

Swansea University
Prifysgol Abertawe

Development and testing of a Particle Swarm Optimizer to handle hard unconstrained and constrained problems

by

Mauro Sebastián Innocente

Submitted to Swansea University in fulfilment of the requirements for the degree of Doctor of Philosophy

Swansea University

Swansea

2010

Summary

The optimization of a system is the quest for its best performance. Thus physical systems tend to reach *minimum* energy states, while biological systems *optimize* their own genetic code and behaviour to *better* cope with the environment. If an artificial system or the model of a natural system can be parameterized, its optimization consists of seeking the best combination of feasible values of those parameters which results in its best performance. This thesis deals with the 'particle swarm optimization' method, which differs from traditional methods in that it poses no restriction to the functions involved. The method was inspired by social behaviour observed in nature, and hence its robustness lies in that it is not deterministically implemented to optimize so that there is no problem-specific implementation that may be inadequate for a different problem.

An extensive study of the coefficients at the core of the method is carried out partly theoretically, partly heuristically, and partly visualizing trajectories. The influence of their settings on the form and speed of convergence is analyzed, and guidelines as to how to obtain the desired behaviour are provided. Different structures of the social network and additional heuristics are studied and tested on unconstrained benchmark problems. Finally, a robust pseudo adaptive constraint-handling mechanism is proposed. The fully working algorithm is tested on a classical benchmark suite of constrained problems and successfully applied to well known engineering problems. Results reported by other authors are also provided for reference.

The 'particle swarm optimizer' developed in this thesis is a global, single-solution, single-objective, gradient-free, population-based method, which is able to handle continuous –exceptionally, discrete variables by rounding-off–, constrained and unconstrained, single-objective problems regardless of whether the functions involved are or are not linear, convex, unimodal, differentiable, smooth, continuous, or even explicit.

Declaration and Statements

DECLARATION

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed ..... (candidate)

Date .....

STATEMENT 1

This thesis is the result of my own investigations, except where otherwise stated. Where correction services have been used, the extent and nature of the correction is clearly marked in a footnote(s).

Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed ..... (candidate)

Date .....

STATEMENT 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ..... (candidate)

Date .....

Table of contents

Summary I

Declaration and Statements III

Table of Contents V

Acknowledgements XV

List of Tables XVII

List of Figures XXXI

Main Acronyms LV

Main Glossary LVII

Chapter 1

INTRODUCTION

1.1. Introduction 1

1.2. Motivation 2

1.3. Objectives 3

1.4. Methodology 4

1.5. Achievements 5

1.6. Layout of the thesis 6

SECTION I

BACKGROUND

Chapter 2

CONSTRAINED OPTIMIZATION

2.1. Introduction 7

2.2. Types of constraints 9

2.3. Optimization problems 10

2.4. Optimization problem formulation 13

2.5. Optimization methods 15
2.6. Constraint-handling 18
2.7. No free lunch theorems for optimization 19
2.8. Closure 22

Chapter 3

EVOLUTIONARY ALGORITHMS

3.1. Introduction 23
3.2. Natural evolution 24
3.3. Evolutionary algorithms 27
3.3.1. Origins and summarized history 27
3.3.2. General evolutionary algorithm 31
3.3.2.1. Initialization 31
3.3.2.2. Fitness evaluation 32
3.3.2.3. Selection 33
3.3.2.4. Genetic alterations 36
3.3.2.5. Termination conditions 37
3.3.3. Genetic Algorithms 38
3.3.3.1. Representation, initialization and fitness evaluation 38
3.3.3.2. Parents' selection 41
3.3.3.3. Genetic alterations 41
3.3.3.4. Fitness evaluation and survivors' selection 44
3.3.4. Genetic Programming 44
3.3.4.1. Representation 44
3.3.4.2. Initialization and fitness evaluation 46
3.3.4.3. Parents' selection 47
3.3.4.4. Genetic alterations 47
3.3.5. Evolution Strategies 50
3.3.5.1. Representation, initialization and fitness evaluation 50
3.3.5.2. Parents' selection 51
3.3.5.3. Genetic alterations 51
3.3.5.4. Fitness evaluation and survivors' selection 55

3.3.6. Evolutionary Programming 56
3.3.6.1. Representation, initialization and fitness evaluation 56
3.3.6.2. Parents' selection 58
3.3.6.3. Genetic alterations 58
3.3.6.4. Fitness evaluation and survivors' selection 59
3.3.7. Differential Evolution 60
3.3.7.1. Representation, initialization and fitness evaluation 60
3.3.7.2. Parents' selection 60
3.3.7.3. Genetic alterations 60
3.3.7.4. Fitness evaluation and survivors' selection 62
3.4. Final remarks and closure 62

Chapter 4

PARTICLE SWARM OPTIMIZATION

4.1. Introduction 65
4.2. Swarm Intelligence 67
4.2.1. Emergence 69
4.2.2. Self-organization 69
4.2.3. Division of labour 71
4.2.4. Stigmergy 71
4.2.5. Biological foundations 72
4.2.5.1. Amoebas 72
4.2.5.2. Social insects 73
4.2.5.3. Fish schools and bird flocks 75
4.2.5.4. Human beings 76
4.2.5.5. Swarm robots 77
4.2.6. Social learning 78
4.3. Ant Colony Optimization 79
4.3.1. The Ant System 80
4.3.1.1. The state transition rule 81
4.3.1.2. The global updating rule 82
4.3.2. The Ant Colony System 82

4.3.2.1. The state transition rule 82
4.3.2.2. The global updating rule 83
4.3.2.3. The local updating rule 84
4.4. Particle Swarm Optimization 85
4.4.1. Social psychology viewpoint 85
4.4.2. Optimization viewpoint 86
4.4.3. Artificial Intelligence viewpoint 86
4.4.4. General viewpoint 87
4.4.5. Basic algorithm 88
4.4.6. Coefficients' setting 90
4.4.6.1. Surfing the waves 91
4.4.6.2. Constriction factor 93
4.4.6.3. Inertia weight 97
4.4.6.4. Other influential work on coefficients settings 99
4.4.7. Neighbourhood topology 100
4.4.8. Constraint-handling 104
4.4.8.1. Methods that use constraints only to evaluate feasibility 105
4.4.8.2. Methods that consider $f(x)$ and $cv(x)$ separately 105
4.4.8.3. Methods that combine $f(x)$ and $cv(x)$ 106
4.4.8.4. Repair algorithms 107
4.4.8.5. Multiobjective-based methods 108
4.4.9. Other features 110
4.4.10. Binary PSO 111
4.5. Final remarks and closure 113

SECTION II

RESEARCH

Chapter 5

SYSTEMATIC COEFFICIENTS STUDY WITH STATIONARY ATTRACTORS

5.1. Introduction 115

5.2. Position equation 117

5.2.1. Recurrence relation 118

5.2.2. Closed-form expression 119

5.2.2.1. Two real-valued roots 121

5.2.2.2. Two complex conjugate roots 122

5.2.2.3. Only one root 124

5.2.3. Discussion 125

5.2.3.1. Two real-valued roots 125

5.2.3.2. Two complex conjugate roots 126

5.2.3.3. Only one root 128

5.3. Deterministic explosion and cyclic behaviour 129

5.4. Stochastic explosion 140

5.5. Convergence graph 148

5.6. On exploitation and exploration 156

5.7. Velocity constraint 158

5.8. Inertia weight 162

5.8.1. Random weight replaced 162

5.8.1.1. Acceleration weight *aw* = 4.00 163

5.8.1.2. Acceleration weight *aw* = 3.80 165

5.8.1.3. Acceleration weight *aw* = 3.60 167

5.8.1.4. Acceleration weight *aw* = 3.40 169

5.8.1.5. Acceleration weight *aw* = 3.20 170

5.8.1.6. Acceleration weight *aw* = 3.00 172

5.8.1.7. Acceleration weight *aw* = 2.80 173

5.8.1.8. Acceleration weight *aw* = 2.60 175
5.8.1.9. Acceleration weight *aw* = 2.40 176
5.8.1.10. Acceleration weight *aw* = 2.20 177
5.8.1.11. Acceleration weight *aw* = 2.00 178
5.8.1.12. Acceleration weight *aw* = 1.80 179
5.8.1.13. Acceleration weight *aw* = 1.60 180
5.8.1.14. Acceleration weight *aw* = 1.20 181
5.8.1.15. Acceleration weight *aw* = 0.80 182
5.8.2. Random weight incorporated 184
5.8.2.1. Acceleration weight *aw* = 4.00 185
5.8.2.2. Acceleration weight *aw* = 3.60 190
5.8.2.3. Acceleration weight *aw* = 3.20 191
5.8.2.4. Acceleration weight *aw* = 3.00 192
5.8.2.5. Acceleration weight *aw* = 2.80 194
5.8.2.6. Acceleration weight *aw* = 2.40 195
5.8.2.7. Acceleration weight *aw* = 2.00 197
5.8.2.8. Acceleration weight *aw* = 1.60 197
5.8.2.9. Acceleration weight *aw* = 1.20 198
5.8.3. Velocity constraint incorporated 198
5.8.3.1. Acceleration weight *aw* = 4.00 199
5.8.3.2. Acceleration weight *aw* = 3.00 206
5.8.3.3. Acceleration weight *aw* = 2.00 208
5.9. Constriction factor 210
5.9.1. Constriction factor *cf* = 0.7298 (*aw* = 4.10, κ = 0.99994) 212
5.9.2. Constriction factor *cf* = 0.547383 (*aw* = 4.10, κ = 0.75) 213
5.9.3. Constriction factor *cf* = 0.364922 (*aw* = 4.10, κ = 0.50) 213
5.9.4. Constriction factor plus velocity constraint 217
5.10. Correlated coefficients 217
5.11. Final remarks and closure 235

Chapter 6

COEFFICIENTS SETTINGS

6.1. Introduction 239

6.2. Single particle 240

6.2.1. PSO with a form of adaptive constriction 240

6.2.2. Randomness 241

6.2.2.1. PSO with reduced ϕ_{max} (PSO-RRM) 243

6.2.2.2. PSO with reduced randomness range 1 (PSO-RRR1) 245

6.2.2.3. PSO with reduced randomness range 2 (PSO-RRR2) 254

6.2.3. Updating attractor 261

6.2.4. Individuality and sociality 274

6.2.5. Preselected sets of settings 282

6.2.5.1. Classical PSO ($aw = \phi_{max}$) 282

6.2.5.2. Constricted PSO ($aw = \phi_{max}$) (*Type 1"*) 283

6.2.5.3. PSO-RRM ($aw = \phi_{max}$) 283

6.2.5.4. PSO-RRR1 ($aw = \phi_{mean}$) 283

6.2.5.5. PSO-RRR2 ($aw = \phi_{mean}$) 283

6.2.5.6. Other authors' settings ($aw = \phi_{max}$) 284

6.3. Swarm of particles 285

6.3.1. Four particles and one dimension 285

6.3.1.1. Classical PSO 285

6.3.1.2. Constricted PSO 290

6.3.1.3. PSO-RRM 293

6.3.1.4. PSO-RRR1 297

6.3.1.5. PSO-RRR2 301

6.3.1.6. Other authors' settings 306

6.3.1.7. Discussion and selected sets of settings 308

6.3.2. Full swarm and multidimensional space 312

6.3.2.1. Measures of clustering and evolution 312

6.3.2.2. Experimental results 316

6.3.2.3. Discussion 333

6.4. Coefficients' settings guidelines 337

6.5. Closure 343

Chapter 7

NEIGHBOURHOODS

7.1. Introduction 345
7.2. Dynamic neighbourhood 346
7.3. Forward topology 381
7.4. Nearest neighbour 413
7.5. Sub-neighbourhoods 431
7.6. Closure 465

Chapter 8

CONSTRAINT-HANDLING

8.1. Introduction 467
8.2. Constrained optimization 467
8.3. Constraint-Handling Techniques 468
8.3.1. Preserving Feasibility with Priority Rules 469
8.3.2. Penalization Method 470
8.4. Pseudo Adaptive Scheme 471
8.4.1. Self-tuned initial relaxation 471
8.4.2. Pseudo Adaptive Tolerance Update 474
8.5. Pseudo Adaptive PFPR 476
8.5.1. Experimental results 476
8.5.2. Discussion 480
8.6. Pseudo Adaptive Penalization 481
8.6.1. Experimental results 481
8.6.2. Discussion 482
8.7. Pseudo Adaptive PFPR with Repair Operator 486
8.7.1. Experimental results 488
8.7.2. Discussion 488
8.8. Engineering applications 489
8.8.1. Pressure Vessel Design (PVD) 496

8.8.2. Welded Beam Design (WBD) 499
8.8.3. Tension/Compression Spring Design (TCSD) 502
8.8.4. Himmelblau's Nonlinear Problem (HNP) 504
8.8.5. 10-Bar Plane Truss 505
8.8.5.1. Stress Constraints 506
8.8.5.2. Stress and Displacements Constraints Case 1 511
8.8.5.3. Stress and Displacements Constraints Case 2 514
8.8.6. 25-Bar Space Truss 518
8.8.6.1. Case 1 (Fleury & Schmit, 1980) 520
8.8.6.2. Case 2 (Park & Ryu, 2004) 534
8.9. Closure 538

SECTION III
CONCLUSIONS

Chapter 9
CONCLUSIONS

9.1. Contributions 541
9.2. Future research 543

References and Bibliography 545

SECTION IV
APPENDICES

Appendix I
AUXILIARY ARITHMETICS

AI.1. Deterministic particle's position equation A1

AI.1.1. Initial conditions A2
AI.1.2. First position derived from initial conditions A3
AI.2. Region in 'ϕ–w' plane for complex roots A5
AI.3. Diagonalization of the system matrix M A6

Appendix II

BENCHMARK TEST PROBLEMS

AII.1. Introduction A9
AII.2. Unconstrained problems A9
AII.2.1. Sphere A9
AII.2.2. Rosenbrock A10
AII.2.3. Rastrigin A12
AII.2.4. Griewank A13
AII.2.5. Schaffer f6 ..... A14
AII.3. Constrained problems A17
AII.3.1. Problem 01 (g01) A18
AII.3.2. Problem 02 (g02) A19
AII.3.3. Problem 03 (g03) A20
AII.3.4. Problem 04 (g04) A20
AII.3.5. Problem 05 (g05) A21
AII.3.6. Problem 06 (g06) A22
AII.3.7. Problem 07 (g07) A23
AII.3.8. Problem 08 (g08) A24
AII.3.9. Problem 09 (g09) A24
AII.3.10. Problem 10 (g10) A25
AII.3.11. Problem 11 (g11) A26
AII.3.12. Problem 12 (g12) A27
AII.3.13. Problem 13 (g13) A27

Appendix III

DIGITAL CONTENT

Acknowledgements

I would like to express my gratitude to my thesis supervisor, Professor Johann Sienz, for his invaluable guidance, advice, encouragement, and friendly support beyond the academic issues, and most of all, for his endless patience. I would also like to thank Maurice Clerc for his readiness to respond to some questions arising from my theoretical research; for his priceless comments and suggestions; and for bringing to my attention some important work from the literature. Comments and suggestions from Professor Vassili Toropov are also appreciated and acknowledged hereby.

I must express my immense gratitude to my parents, Teresa Marín and Armando Innocente, and to my three sisters Daniela Verónica Innocente, Agostina Silvana Innocente, and Carla Georgina Innocente, for their everlasting, unconditional support and encouragement in every matter. Every single one of my achievements would have not been possible without them. I would also like to thank Simon Elliott for his friendly support and his contagious, permanent good mood and positive attitude.

Thanks to Erick Saavedra and Alejandro Diaz de la O for their support and encouragement, as well as their readiness to engage in any work-related or philosophical discussion. In particular, thanks to Erick for spending the time and effort in going through the long chapter 5, and to Alejandro for listening and discussing some mathematical aspects of my research. I would also like to thank Aurelio Arranz-Carreño for his insight and help on any mathematical question rising from my studies, as well as for his friendly support. My appreciation to Eva Mantiñán Fraga and Santiago Waessle for their invaluable friendship, for always being there when needed or wanted, and for their attempts to keep me a social being throughout all these past few years. Thanks to Ayisha Jessa for her selfless and genuine support, especially during the first couple of years of my doctorate. I would also like to thank Laura Rocío Pascual for spending numerous sleepless nights by my side, watching for me during the last few months of my thesis writing. I would like to express my gratitude to María Andrade for her genuine, friendly company and continuous family-like support during my long stay in Swansea.

Finally, I would like to acknowledge the feedback received from the *Applied Mechanics Department* (*Departamento de Mecánica Aplicada*) from the *School of Engineering*

(*Facultad de Ingeniería*) at *Universidad Nacional del Nordeste*, Argentina, with regards to every subject related to numerical methods. In particular, I would like to express my gratefulness to Pablo Alejandro Beneyto for his help and encouragement throughout my undergraduate and postgraduate studies.

List of tables

Table 5.1. Values of $\phi < 4$ that lead to exact cyclic behaviour (for w =1), identified in (Clerc & Kennedy, 2002). The values highlighted are those delimiting the regions of different search type identified in (Ozcan & Mohan, 1998) and (Ozcan & Mohan, 1999) for the particular case of $x^{(0)} = p$ 133

Table 5.2. Values of $\phi < 4$ that lead to exact cyclic behaviour (for w =1) and related data, identified in (Clerc & Kennedy, 2002) 138

Table 5.3. Twenty selected pairs 'ϕ–w' to be considered to illustrate the speed and form of convergence/divergence of the particle's trajectory associated to the different regions of the 'ϕ–w' plane 153

Table 6.1. Qualitative exploration/exploitation trade-offs to be expected from a number of optimizers proposed in section 6.2.5 284

Table 6.2. Qualitative exploration/exploitation trade-offs to be expected from a number of optimizers proposed in section 6.2.5. The approaches/settings are presented in a top-down fashion from more to less explorative performance 309

Table 6.3. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Sphere function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 318

Table 6.4. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Sphere function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 319

Table 6.5. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Sphere function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 320

Table 6.6. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 321

Table 6.7. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 322

Table 6.8. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 323

Table 6.9. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional

Rastrigin function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 324

Table 6.10. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 325

Table 6.11. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 326

Table 6.12. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Griewank function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 327

Table 6.13. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Griewank function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 328

Table 6.14. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Griewank function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 329

Table 6.15. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 330

Table 6.16. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 331

Table 6.17. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours 332

Table 7.1. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Sphere function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to ‘swarm-size – 1’); the WHEEL; and a RANDOM topology 348

Table 7.2. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Sphere function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to ‘swarm-size – 1’); the WHEEL; and a RANDOM topology 349

Table 7.3. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Sphere function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 350

Table 7.4. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 353

Table 7.5. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 354

Table 7.6. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 355

Table 7.7. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 359

Table 7.8. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 360

Table 7.9. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 361

Table 7.10. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Griewank function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 365

Table 7.11. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Griewank function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 366

Table 7.12. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Griewank function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 367

Table 7.13. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 371

Table 7.14. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 372

Table 7.15. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology 373

Table 7.16. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Sphere function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 382

Table 7.17. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Sphere function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 383

Table 7.18. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Sphere function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 384

Table 7.19. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rosenbrock function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 387

Table 7.20. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rosenbrock function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 388

Table 7.21. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rosenbrock function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 389

Table 7.22. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rastrigin function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 393

Table 7.23. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rastrigin function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 394

Table 7.24. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rastrigin function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 395

Table 7.25. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Griewank function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 399

Table 7.26. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Griewank function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 400

Table 7.27. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Griewank function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 401

Table 7.28. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Schaffer f6 function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 405

Table 7.29. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Schaffer f6 function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 406

Table 7.30. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Schaffer f6 function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1') 407

Table 7.31. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Sphere function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 415

Table 7.32. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Sphere function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 416

Table 7.33. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Sphere function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 417

Table 7.34. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Rosenbrock function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 418

Table 7.35. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Rosenbrock function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 419

Table 7.36. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Rosenbrock function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 420

Table 7.37. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Rastrigin function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 421

Table 7.38. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Rastrigin function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 422

Table 7.39. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Rastrigin function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 423

Table 7.40. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Griewank function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 424

Table 7.41. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Griewank function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 425

Table 7.42. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Griewank function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 426

Table 7.43. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Schaffer f6 function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 427

Table 7.44. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Schaffer f6 function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 428

Table 7.45. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Schaffer f6 function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB) 429

Table 7.46. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Sphere function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections are tested, the 'individual' and the 'local' overlapping 434

Table 7.47. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Sphere function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number

of neighbours (from 2 to ‘swarm-size – 1’). Two types of interconnections are tested, the ‘individual’ and the ‘local’ overlapping 435

Table 7.48. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Sphere function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to ‘swarm-size – 1’). Two types of interconnections are tested, the ‘individual’ and the ‘local’ overlapping 436

Table 7.49. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Rosenbrock function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to ‘swarm-size – 1’). Two types of interconnections are tested, the ‘individual’ and the ‘local’ overlapping 439

Table 7.50. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Rosenbrock function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to ‘swarm-size – 1’). Two types of interconnections are tested, the ‘individual’ and the ‘local’ overlapping 440

Table 7.51. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Rosenbrock function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to ‘swarm-size – 1’). Two types of interconnections are tested, the ‘individual’ and the ‘local’ overlapping 441

Table 7.52. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Rastrigin function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to ‘swarm-size – 1’). Two types of interconnections are tested, the ‘individual’ and the ‘local’ overlapping 445

Table 7.53. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Rastrigin function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to ‘swarm-size – 1’). Two types of interconnections are tested, the ‘individual’ and the ‘local’ overlapping 446

Table 7.54. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Rastrigin function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to ‘swarm-size – 1’). Two types of interconnections are tested, the ‘individual’ and the ‘local’ overlapping 447

Table 7.55. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Griewank function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections are tested, the 'individual' and the 'local' overlapping 451

Table 7.56. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Griewank function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections are tested, the 'individual' and the 'local' overlapping 452

Table 7.57. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Griewank function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections are tested, the 'individual' and the 'local' overlapping 453

Table 7.58. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Schaffer f6 function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections are tested, the 'individual' and the 'local' overlapping 457

Table 7.59. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Schaffer f6 function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections are tested, the 'individual' and the 'local' overlapping 458

Table 7.60. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Schaffer f6 function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections are tested, the 'individual' and the 'local' overlapping 459

Table 8.1. Features of the problems in the test suite: number of dimensions, inequality and equality constraints; feasibility ratios (FRs) of the problem with no tolerance, desired tolerance, and initial tolerance; and the mean self-tuned initial inequality and equality tolerances 474

Table 8.2. Statistical results obtained for the 13 problems in the test suite for the PFPR technique and 3 types of tolerance relaxation: none, initially self-tuned with exponential decrease, and initially self-tuned with Pseudo Adaptive decrease 478

Table 8.3. Statistical results obtained for the 13 problems in the test suite for the PFPR technique coupled with the proposed Pseudo Adaptive Scheme. Results

from (Toscano Pulido & Coello Coello, 2004) and (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005) (PESO) are provided for reference 479

Table 8.4. Statistical results obtained for the 13 problems in the test suite for the PM technique and 3 types of tolerance relaxation: none, initially self-tuned with exponential decrease, and initially self-tuned with Pseudo Adaptive decrease 483

Table 8.5. Statistical results obtained for the 13 problems in the test suite for the PM technique coupled with the proposed pseudo adaptive scheme. Results from (Toscano Pulido & Coello Coello, 2004) and (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005) (PESO) are provided for reference 484

Table 8.6. Statistical results for the proposed 'Pseudo Adaptive PM', 'Pseudo Adaptive PFPR' and 'Pseudo Adaptive PFPR with Repair operator' obtained for the 13 problems in the test suite 490

Table 8.7. Statistical results obtained for the 13 problems in the test suite for the 'Pseudo Adaptive PFPR with Repair operator'. Results from (Toscano Pulido & Coello Coello, 2004) and (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005) (PESO) are provided for reference 491

Table 8.8. Statistical results obtained for the 13 problems in the test suite for the 'Pseudo Adaptive PFPR with Repair operator'. Results from (Fuentes Cabrera & Coello Coello, 2007) and (Muñoz Zavala, Hernández Aguirre, Villa Diharce, & Botello Rionda, 2006) (PESO+) are provided for reference 492

Table 8.9. Percentage of successful solutions obtained for the 13 problems in the test suite by the 'Pseudo Adaptive PFPR with Repair operator'. Results from (Zielinski & Laur, 2006) and (Muñoz Zavala, Hernández Aguirre, Villa Diharce, & Botello Rionda, 2006) (PESO+) are provided for reference 493

Table 8.10. Best and mean solutions obtained by the 'Pseudo Adaptive PFPR with Repair operator' and by different authors and different approaches in the literature for problems g01 to g06 494

Table 8.11. Best and mean solutions obtained by the 'Pseudo Adaptive PFPR with Repair operator' and by different authors and different approaches in the literature for problems g07 to g12 495

Table 8.12. Best and mean solutions obtained by the 'Pseudo Adaptive PFPR with Repair operator' and by different authors and different approaches in the literature for problem g13 496

Table 8.13. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Mixed Discrete Pressure Vessel Design (MDPVD) problem 498

Table 8.14. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the MDPVD problem 498

Table 8.15. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Pressure Vessel Design (PVD) problem 498

Table 8.16. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the PVD problem 499

Table 8.17. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Welded Beam Design (WBD) problem 501

Table 8.18. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the WBD problem 501

Table 8.19. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Tension/Compression Spring Design (TCSD) problem 503

Table 8.20. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the TCSD problem 503

Table 8.21. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Himmelblau's Nonlinear Problem (HNP) 505

Table 8.22. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the HLP 505

Table 8.23. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the 10-Bar Truss problem with stress constraints 507

Table 8.24. Coordinates of the best solution found by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the 10-Bar Truss problem with stress constraints 507

Table 8.25. Details of the FE analysis of the optimal design in (Haftka & Gürdal, 1992) of the 10-Bar Truss problem with stress constraints 509

Table 8.26. Details of the FE analysis of the best design of the 10-Bar Truss problem with stress constraints found by the 'MS-SN P.AD. PFPR' PSO 509

Table 8.27. Statistical results obtained by the 'PSO-RRR2-1 Pseudo Adaptive PFPR' and by the Fully Stressed Design (FSD) in (Haftka & Gürdal, 1992) for the 10-Bar Truss problem with stress constraints 509

Table 8.28. Coordinates of the best solution found by the 'PSO-RRR2-1 Pseudo Adaptive PFPR' and by the Fully Stressed Design (FSD) in (Haftka & Gürdal, 1992) for the 10-Bar Truss problem with stress constraints 509

Table 8.29. Details of the FE analysis of the best design of the 10-Bar Truss problem with stress constraints found by the 'PSO-RRR2-1 (100-5000) P.AD. PFPR' 510

Table 8.30. Details of the FE analysis of the fully stressed design in (Haftka & Gürdal, 1992) of the 10-Bar Truss problem with stress constraints 510

Table 8.31. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by Fleury and Schmit (1980) for the 10-Bar Truss problem with stress and two equality constraints (*y*-displacements of nodes '1' and '3') 512

Table 8.32. Coordinates of the best solutions found by the 'MS-SN P.AD. PFPR' and by Fleury and Schmit (1980) for the 10-Bar Truss problem with stress and two equality constraints (*y*-displacements of nodes '1' and '3') 512

Table 8.33. Cross sections and stresses details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) of the 10-Bar Truss problem with stress and displacement constraints (Case 1) 513

Table 8.34. Displacement details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) of the 10-Bar Truss problem with stress and displacement constraints (Case 1) 513

Table 8.35. Cross sections and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 10-Bar Truss problem with stress and displacement constraints (Case 1) 514

Table 8.36. Displacement details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 10-Bar Truss problem with stress and displacement constraints (Case 1) 514

Table 8.37. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by other authors for the 10-Bar Truss problem with stress and four inequality-displacement constraints (nodes '1' to '4') 515

Table 8.38. Coordinates of the best solutions obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by other authors for the 10-Bar Truss problem with stress and four inequality-displacement constraints (nodes '1' to '4') 515

Table 8.39. Cross sections and stresses details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) of the 10-Bar Truss problem with stress and displacement constraints (Case 2) 516

Table 8.40. Displacement details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) of the 10-Bar Truss problem with stress and displacement constraints (Case 2) 517

Table 8.41. Cross sections and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 10-Bar Truss problem with stress and displacement constraints (Case 2) 517

Table 8.42. Displacement details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 10-Bar Truss problem with stress and displacement constraints (Case 2) 517

Table 8.43. Allowable compression stress for the members of the 25-Bar Truss from (Fleury & Schmit, 1980) 520

Table 8.44. Load components on the nodes (Load Case 1), from (Fleury & Schmit, 1980) 520

Table 8.45. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with Load Case 1, as in (Fleury & Schmit, 1980) 520

Table 8.46. Coordinates of the best solution obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with Load Case 1, as in (Fleury & Schmit, 1980) 521

Table 8.47. Cross sections and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with Load Case 1 521

Table 8.48. Displacement details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with Load Case 1 522

Table 8.49. Load components on the nodes (Load Case 2), from 'Table 9 Case 1' in (Li, Huang, Liu, & Wu, 2007) 522

Table 8.50. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with Load Case 2, as in (Li, Huang, Liu, & Wu, 2007) 522

Table 8.51. Coordinates of the best solution obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with Load Case 2, as in (Li, Huang, Liu, & Wu, 2007) 523

Table 8.52. Cross sections and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with Load Case 2 523

Table 8.53. Displacement details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with Load Case 2 524

Table 8.54. Coordinates of an ad-hoc solution for both Load Cases, from the solutions for each independent Load Case obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem as in (Fleury & Schmit, 1980) and in (Li, Huang, Liu, & Wu, 2007) 525

Table 8.55. Cross sections and stresses details from the FE analysis of the ad-hoc design in Table 8.54 for the 25-Bar Truss problem with Load Case 1 525

Table 8.56. Displacement details from the FE analysis of the ad-hoc design in Table 8.54 for the 25-Bar Truss problem with Load Case 1 526

Table 8.57. Cross sections and stresses details from the FE analysis of the ad-hoc design in Table 8.54 for the 25-Bar Truss problem with Load Case 2 526

Table 8.58. Displacement details from the FE analysis of the ad-hoc design in Table 8.54 for the 25-Bar Truss problem with Load Case 2 527

Table 8.59. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with two Load Cases as in (Fleury & Schmit, 1980) and in (Li, Huang, Liu, & Wu, 2007) 529

Table 8.60. Coordinates of the best solution obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem

with two Load Cases as in (Fleury & Schmit, 1980) and in (Li, Huang, Liu, & Wu, 2007) 529

Table 8.61. Cross sections and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with two Load Cases as in (Li, Huang, Liu, & Wu, 2007) and in (Fleury & Schmit, 1980), when loaded with the first Load Case 530

Table 8.62. Displacement details from the same FE analysis as in Table 8.61 530

Table 8.63. Cross sections and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with two Load Cases as in (Li, Huang, Liu, & Wu, 2007) and in (Fleury & Schmit, 1980), when loaded with the second Load Case 531

Table 8.64. Displacement details from the same FE analysis as in Table 8.63 531

Table 8.65. Cross sections and stresses details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) for the 25-Bar Truss problem with two Load Cases as in (Li, Huang, Liu, & Wu, 2007) and in (Fleury & Schmit, 1980), when loaded with the first Load Case 532

Table 8.66. Displacement details from the same FE analysis as in Table 8.65 532

Table 8.67. Cross sections and stresses details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) for the 25-Bar Truss problem with two Load Cases as in (Li, Huang, Liu, & Wu, 2007) and in (Fleury & Schmit, 1980), when loaded with the second Load Case 533

Table 8.68. Displacement details from the same FE analysis as in Table 8.67 533

Table 8.69. Load components on the nodes for the 25-Bar Truss problem as in (Park & Ryu, 2004) 535

Table 8.70. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem as formulated in (Park & Ryu, 2004) 535

Table 8.71. Coordinates of the best solution obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem as formulated in (Park & Ryu, 2004) 535

Table 8.72. Cross sections and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem formulated as in (Park & Ryu, 2004) 536

Table 8.73. Displacement details from the same FE analysis as in Table 8.72 536

Table 8.74. Cross sections and stresses details from the FE analysis of the optimal design in (Park & Ryu, 2004) for the 25-Bar Truss problem formulated as in (Park & Ryu, 2004) 537

Table 8.75. Displacement details from the same FE analysis as in Table 8.74 537

List of figures

Fig. 2.1. A tentative classification of optimization methods 18

Fig. 3.1. Schematic evolution process of a population within a single generational step. It can be viewed as a succession of four mapping functions relating the genotypic information state space and the phenotypic observable state space 25

Fig. 3.2. Flow chart for a general Evolutionary Algorithm 32

Fig. 3.3. Example of binary representation of two individuals for a problem with two variables 38

Fig. 3.4. Example of binary representation of two individuals for a problem with two variables, where a single bit difference in the genotype leads to distant positions in the phenotype 39

Fig. 3.5. Numerical, comparative example of base 10, binary, and Gray codes 39

Fig. 3.6. Example of one-point crossover 42

Fig. 3.7. Example of two-point crossover 42

Fig. 3.8. Examples of the tree-like graphical representation of three individuals standing for different functions 45

Fig. 3.9. Example of crossover in GP 48

Fig. 3.10. Example of point mutation in GP 49

Fig. 3.11. Example of simple (left) and correlated (right) mutations 56

Fig. 3.12. Example of a finite state machine with three states (A, B, C), two inputs (1, 2) and four outputs (I, II, III, IV) 57

Fig. 4.1. Graph representing a four-city symmetric TSP to be solved by an ant algorithm 81

Fig. 4.2. General flowchart for the canonical PSO algorithm 89

Fig. 4.3. a) fully connected topology; b) wheel topology; c) ring topology with neighbourhood-size equal to three 102

Fig. 4.4. 'von Neumann' or 'Square' topology 103

Fig. 4.5. Cut-off at the boundary techniques and bisection method 108

Fig. 4.6. A PSO algorithm optimizing a constrained problem. The constraint-handling techniques are the plain 'preserving feasibility' technique and the 'preserving feasibility with cut-off at the boundary and velocity zeroed-in' technique 109

Fig. 4.7. A PSO algorithm optimizing a constrained problem. The constraint-handling technique is the 'bisection' method 110

Fig. 5.1. Delimitation of the region in the 'ϕ–w' space where the roots of the characteristic polynomial are complex 127

Fig. 5.2. Complex region in the 'ϕ–w' space where the roots are complex conjugates and convergence is ensured 128

Fig. 5.3. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ removed for $w = 1.00$ and $4.10 \geq aw \geq 3.00$ 134

Fig. 5.4. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ removed for $w = 1.00$ and $2.80 \geq aw \geq 1.80$ 135

Fig. 5.5. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ removed for $w = 1.00$ and $1.70 \geq aw \geq 1.10$ 136

Fig. 5.6. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ removed for $w = 1.00$ and $1.00 \geq aw \geq 0.30$ 137

Fig. 5.7. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ removed for $w = 1.00$, $aw = 2 - \sqrt{3}$ in figure A), and $aw = 0.2$ in figure B) 138

Fig. 5.8. Periods associated to $0 < \phi < 4$. The different values of k identify the number of '2π' involved in the cycle 139

Fig. 5.9. Influence of the coefficients $\phi = 1$, $\phi = 2$, and $\phi = 3$ on the particle's trajectory between two consecutive time-steps, disregarding the inertia weight 139

Fig. 5.10. Six possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ included for $w = 1.00$ and $aw = 4.00$ 141

Fig. 5.11. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 1.00$ and $4.20 \geq aw \geq 3.60$. The random weights $U_{(0,1)}$ are replaced by their expected value in the first column, and included in the second one 142

Fig. 5.12. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 1.00$ and $3.40 \geq aw \geq 2.80$. The random weights $U_{(0,1)}$ are replaced by their expected value in the first column, and included in the second one 143

Fig. 5.13. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 1.00$ and $2.60 \geq aw \geq 2.00$. The random weights $U_{(0,1)}$ are replaced by their expected value in the first column, and included in the second one 144

Fig. 5.14. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 1.00$ and $1.80 \geq aw \geq 1.20$. The random weights $U_{(0,1)}$ are replaced by their expected value in the first column, and included in the second one 145

Fig. 5.15. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 1.00$ and $1.00 \geq aw \geq 0.40$. The random

weights $U_{(0,1)}$ are replaced by their expected value in the first column, and included in the second one 146

Fig. 5.16. Six possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ included for $w = 1.00$ and $aw = 3.40$ 147

Fig. 5.17. Relevant curves bounding the convergence region in the 'ϕ–w' plane 150

Fig. 5.18. Convergence region in the 'ϕ–w' plane (blue shaded triangle) 151

Fig. 5.19. Trajectory of a deterministic particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for '$w = -0.50$, $\phi = 0.25$' and '$w = -0.50$, $\phi = 0.75$'. The particle converges despite the negative inertia 151

Fig. 5.20. Region in the 'ϕ–w' plane where convergence is ensured (gray shade). Deterministic explosion is guaranteed within the black triangle 152

Fig. 5.21. Twenty selected pairs 'ϕ–w' to be considered to illustrate the speed and form of convergence/divergence of the particle's trajectory associated to the different regions of the 'ϕ–w' plane 153

Fig. 5.22. Trajectories of the deterministic particle for the pairs A1 to E2 (first two columns) in Fig. 5.21 154

Fig. 5.23. Trajectories of the deterministic particle for the pairs A3 to E4 (last two columns) in Fig. 5.21 155

Fig. 5.24. Comparison of the trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for the deterministic linear explosion ($w = 1.00$; $aw = 4.00$; $U_{(0,1)}$ removed) with and without constraining the velocity. The trajectory becomes cyclic if $v_{max} = fint$ 159

Fig. 5.25. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 1.00$ and $aw = 4.00$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first. Different rows correspond to different values of the v_{max} constraint 160

Fig. 5.26. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 1.00$ and $aw = 4.00$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first. Different rows correspond to different values of the v_{max} constraint 161

Fig. 5.27. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.10 \geq w \geq 0.60$ and $aw = 4.00$. Random weights $U_{(0,1)}$ are replaced by their expected value 163

Fig. 5.28. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.50 \geq w \geq 0.00$ and $aw = 4.00$. Random weights $U_{(0,1)}$ are replaced by their expected value 164

Fig. 5.29. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.10 \geq w \geq 0.60$ and $aw = 3.80$. Random weights $U_{(0,1)}$ are replaced by their expected value 165

Fig. 5.30. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.50 \geq w \geq 0.00$ and $aw = 3.80$. Random weights $U_{(0,1)}$ are replaced by their expected value 166

Fig. 5.31. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.10 \geq w \geq 0.60$ and $aw = 3.60$. Random weights $U_{(0,1)}$ are replaced by their expected value 167

Fig. 5.32. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.50 \geq w \geq 0.00$ and $aw = 3.60$. Random weights $U_{(0,1)}$ are replaced by their expected value 168

Fig. 5.33. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.10 \geq w \geq 0.40$ and $aw = 3.40$. Random weights $U_{(0,1)}$ are replaced by their expected value 169

Fig. 5.34. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.30 \geq w \geq 0.00$ and $aw = 3.40$. Random weights $U_{(0,1)}$ are replaced by their expected value 170

Fig. 5.35. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.10 \geq w \geq 0.80$ and $aw = 3.20$. Random weights $U_{(0,1)}$ are replaced by their expected value 170

Fig. 5.36. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.70 \geq w \geq 0.00$ and $aw = 3.20$. Random weights $U_{(0,1)}$ are replaced by their expected value 171

Fig. 5.37. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.10 \geq w \geq 0.40$ and $aw = 3.00$. Random weights $U_{(0,1)}$ are replaced by their expected value 172

Fig. 5.38. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.30 \geq w \geq 0.00$ and $aw = 3.00$. Random weights $U_{(0,1)}$ are replaced by their expected value 173

Fig. 5.39. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.10 \geq w \geq 0.80$ and $aw = 2.80$. Random weights $U_{(0,1)}$ are replaced by their expected value 173

Fig. 5.40. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.70 \geq w \geq 0.00$ and $aw = 2.80$. Random weights $U_{(0,1)}$ are replaced by their expected value 174

Fig. 5.41. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.00 \geq w \geq 0.00$ and $aw = 2.60$. Random weights $U_{(0,1)}$ are replaced by their expected value 175

Fig. 5.42. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.00 \geq w \geq 0.00$ and $aw = 2.40$. Random weights $U_{(0,1)}$ are replaced by their expected value 176

Fig. 5.43. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.00 \geq w \geq 0.00$ and $aw = 2.20$. Random weights $U_{(0,1)}$ are replaced by their expected value 177

Fig. 5.44. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.00 \geq w \geq 0.00$ and $aw = 2.00$. Random weights $U_{(0,1)}$ are replaced by their expected value 178

Fig. 5.45. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.00 \geq w \geq 0.00$ and $aw = 1.80$. Random weights $U_{(0,1)}$ are replaced by their expected value 179

Fig. 5.46. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.00 \geq w \geq 0.00$ and $aw = 1.60$. Random weights $U_{(0,1)}$ are replaced by their expected value 180

Fig. 5.47. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.00 \geq w \geq 0.00$ and $aw = 1.20$. Random weights $U_{(0,1)}$ are replaced by their expected value 181

Fig. 5.48. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $1.00 \geq w \geq 0.00$ and $aw = 0.80$. Random weights $U_{(0,1)}$ are replaced by their expected value 182

Fig. 5.49. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$, random weights $U_{(0,1)}$ replaced by their expected value '0.50', $w = 0.90$, $aw = 4.00$ (left) and $aw = 0.80$ (right) 183

Fig. 5.50. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ replaced by their expected value '0.50', for $w = 0.10$ and $aw = 4.00$ (left), and for $w = 0.60$ and $aw = 2.00$ (right) 184

Fig. 5.51. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.75 \geq w \geq 0.15$ and $aw = 4.00$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 185

Fig. 5.52. Six possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ included for $w = 0.75$ and $aw = 4.00$ 186

Fig. 5.53. Six possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ included for $w = 0.50$ and $aw = 4.00$ 187

Fig. 5.54. Six possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ included for $w = 0.75$ and $aw = 3.60$ 188

Fig. 5.55. Two quite different possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ included for $w = 0.30$ and $aw = 3.60$ 189

Fig. 5.56. Two quite different possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ included for $w = 0.15$ and $aw = 3.60$ 189

Fig. 5.57. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.75 \geq w \geq 0.15$ and $aw = 3.60$. Random

weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 190

Fig. 5.58. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.75 \geq w \geq 0.15$ and $aw = 3.20$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 191

Fig. 5.59. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.75 \geq w \geq 0.15$ and $aw = 3.00$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 192

Fig. 5.60. Six possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ included for $w = 0.75$ and $aw = 3.00$ 193

Fig. 5.61. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.75 \geq w \geq 0.20$ and $aw = 2.80$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 194

Fig. 5.62. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $0.75 \geq w \geq 0.30$ and $aw = 2.40$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 195

Fig. 5.63. Six possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ and random weights $U_{(0,1)}$ included for $w = 0.75$ and $aw = 2.40$ 196

Fig. 5.64. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.75$; $w = 0.50$; and $aw = 2.00$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 197

Fig. 5.65. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.75$; $w = 0.50$; and $aw = 1.60$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 197

Fig. 5.66. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.75$; $w = 0.50$; and $aw = 1.20$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 198

Fig. 5.67. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.75$ and $aw = 4.00$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected values in the first one. Different rows stand for different v_{max} constraints 200

Fig. 5.68. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.75$, $aw = 4.00$, and random weights

$U_{(0,1)}$ included. The trajectories on the left column have no v_{max} whereas those on the right column have $v_{max} = \mathit{fint} / 2$ 201

Fig. 5.69. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.75$, $aw = 4.00$, and random weights $U_{(0,1)}$ included. The trajectories on the left column have no v_{max} whereas those on the right column have $v_{max} = \mathit{fint} / 2$. The two runs here are consecutive to those in Fig. 5.68 202

Fig. 5.70. Six possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.75$, $aw = 4.00$, random weights $U_{(0,1)}$ included, and $v_{max} = \mathit{fint} / 4$ 203

Fig. 5.71. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.50$ and $aw = 4.00$. Different rows stand for different values of the v_{max} constraint. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 204

Fig. 5.72. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.50$, $aw = 4.00$, and random weights $U_{(0,1)}$ included. The trajectories on the left column have no v_{max} whereas those on the right column have $v_{max} = \mathit{fint} / 2$ 205

Fig. 5.73. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.50$, $aw = 4.00$, and random weights $U_{(0,1)}$ included. The trajectories on the left column have no v_{max} whereas those on the right column have $v_{max} = \mathit{fint} / 2$. The two runs here are consecutive to those in Fig. 5.72 206

Fig. 5.74. Six possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.50$, $aw = 4.00$, random weights $U_{(0,1)}$ included, and $v_{max} = \mathit{fint} / 4$ 207

Fig. 5.75. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.75$ and $aw = 3.00$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected values in the first one. Different rows stand for different v_{max} constraints 208

Fig. 5.76. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $w = 0.75$ and $aw = 2.00$. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected values in the first one. Different rows stand for different v_{max} constraints 209

Fig. 5.77. Values of the constriction factor (*cf*) *type 1"* for a range of values of *aw* and three values of κ 211

Fig. 5.78. Translation of the *type 1"* constriction factor into the classical PSO formulation for $aw \leq 4$ 212

Fig. 5.79. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $cf = 0.7298$ and $aw = 4.10$. Random weights $U_{(0,1)}$ are removed in A); replaced by their expected value in B); and included in the remaining figures 214

Fig. 5.80. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $cf = 0.547383$ and $aw = 4.10$. Random weights $U_{(0,1)}$ are removed in A); replaced by their expected value in B); and included in the remaining figures 215

Fig. 5.81. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $cf = 0.364922$ and $aw = 4.10$. Random weights $U_{(0,1)}$ are removed in A); replaced by their expected value in B); and included in the remaining figures 216

Fig. 5.82. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $cf = 0.7298$; $aw = 4.10$; and v_{max} equal to half the feasible interval. Random weights $U_{(0,1)}$ are removed in A); replaced by their expected value in B); and included in the remaining figures 218

Fig. 5.83. Four positions of the whole cycle of the particle for the settings of the original PSO ($w = 1$ and $aw = 4.00$), with the random weights replaced by their expected value. Initial velocity is set to '0', and 'd' is the distance from the position at time-step t to the weighted average of the two attractors 219

Fig. 5.84. Two positions of the cycle of the particle for $w = 0$ and $aw = 4.00$, with the random weights replaced by their expected value. Initial velocity is set to '0', and 'd' is the distance from the position at time-step t to the weighted average of the two attractors 219

Fig. 5.85. Six consecutive positions of the particle for $w = 0.50$ and $aw = 4.00$, with the random weights replaced by their expected value. Initial velocity is set to '0', and 'd' is the distance from the position at time-step t to the weighted average of the two attractors. The latter is reached in two time-steps 220

Fig. 5.86. Three consecutive positions of the particle for $w = 1/6$ and $aw = 3.00$, with the random weights replaced by '0.50'. Initial velocity is set to '0' and 'd' is the distance from the position at time-step t to the weighted average of the two attractors. The latter is reached in two time-steps 221

Fig. 5.87. Three consecutive positions of the particle for $w = 0$ and $aw = 2.00$, with the random weights replaced by '0.50'. Initial velocity is set to '0' and 'd' is the distance from the position at time-step t to the weighted average of the two attractors. The latter is reached in one time-step 221

Fig. 5.88. Four consecutive positions of the particle for $w = 0.50$ and $aw = 1.00$, with the random weights replaced by '0.50'. Initial velocity is set to '0' and 'd' is the distance from the position at time-step t to the weighted average of the two attractors. The latter is reached in two time-steps 221

Fig. 5.89. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $5.00 \geq aw \geq 3.40$, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are replaced by their expected value 224

Fig. 5.90. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $3.20 \geq aw \geq 1.00$, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are replaced by their expected value 225

Fig. 5.91. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $4.80 \geq aw \geq 4.20$, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 226

Fig. 5.92. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $4.00 \geq aw \geq 3.40$, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 227

Fig. 5.93. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $aw = 3.20$, $aw = 3.00$, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one 228

Fig. 5.94. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $4.80 \geq aw \geq 4.20$, w as in Eq. (5.101), and random weights $U_{(0,1)}$ included. There is no v_{max} in the left column, while it is set to half the feasible interval in the right column 229

Fig. 5.95. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $4.00 \geq aw \geq 3.40$, w as in Eq. (5.101), and random weights $U_{(0,1)}$ included. There is no v_{max} in the left column, while it is set to half the feasible interval in the right column 230

Fig. 5.96. Trajectory of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $aw = 3.20$, $aw = 3.00$, w as in Eq. (5.101), and random weights $U_{(0,1)}$ included. There is no v_{max} in the left column, while it is set to half the feasible interval in the right column 231

Fig. 5.97. Four possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $aw = 4.40$, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and v_{max} set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.94 III) 232

Fig. 5.98. Four possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $aw = 4.20$, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and v_{max} set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.94 IV) 232

Fig. 5.99. Four possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $aw = 4.00$, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and v_{max} set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.95 I) 233

Fig. 5.100. Four possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $aw = 3.80$, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and v_{max} set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.95 II) 233

Fig. 5.101. Four possible trajectories of a particle initialized at $x = 100$ over a 1-dimensional space with stationary attractors at $x = 0$ for $aw = 3.60$, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and v_{max} set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.95 III) 234

Fig. 5.102. Regions in the 'ϕ–w' plane. The light gray area comprises the convergence region, and the dark gray area the divergence region. The parabola in black dotted line bounds the complex region. The blue dotted line is the plot of Eq. (5.102) for $aw = \phi_{max}$. The red solid line is the plot of Eq. (5.103) for $aw = \phi_{max}$ 234

Fig. 6.1. Constricted ϕ as a function of the original, unconstricted ϕ for three different values of κ 241

Fig. 6.2. Region in the plane 'ϕ–w' from where ϕ is sampled in PSO-RRM 244

Fig. 6.3. Region in the plane 'ϕ–w' from where ϕ is sampled in PSO-RRR1 245

Fig. 6.4. Trajectory corresponding to the average behaviour of a deterministic particle initialized at $x = 100$, flying over a 1-dimensional space, with stationary attractors at $x = 0$, for $1.00 \geq w \geq 0.30$. This average behaviour is exhibited by both the PSO-RRM and the PSO-RRR1 247

Fig. 6.5. Trajectory corresponding to the average behaviour of a deterministic particle initialized at $x = 100$, flying over a 1-dimensional space, with stationary attractors at $x = 0$, for $w = 0.20$ (left) and $w = 0.10$ (right). This average behaviour is exhibited by both the PSO-RRM and the PSO-RRR1 248

Fig. 6.6. Trajectory corresponding to the average behaviour of a deterministic particle initialized at $x = 100$, flying over a 1-dimensional space with stationary attractors at $x = 0$, for $w = 0.00$. This average behaviour is exhibited by the PSO-RRM, the PSO-RRR1, and also by the PSO-RRR2 proposed in the next section 248

Fig. 6.7. Trajectory of a particle initialized at $x = 100$, over a 1-dimensional space with stationary attractors at $x = 0$, for $w = 0.80$, corresponding to four consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1 249

Fig. 6.8. Trajectory of a particle initialized at $x = 100$, over a 1-dimensional space with stationary attractors at $x = 0$, for $w = 0.80$, corresponding to two consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1. The runs considered here are consecutive to those in Fig. 6.7 250

Fig. 6.9. Trajectory of a particle initialized at $x = 100$, over a 1-dimensional space with stationary attractors at $x = 0$, for $w = 0.50$, corresponding to four consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1 251

Fig. 6.10. Trajectory of a particle initialized at $x = 100$, over a 1-dimensional space with stationary attractors at $x = 0$, for $w = 0.50$, corresponding to two consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1. The runs considered here are consecutive to those in Fig. 6.9 252

Fig. 6.11. Trajectory of a particle initialized at $x = 100$, over a 1-dimensional space with stationary attractors at $x = 0$, for $w = 0.20$, corresponding to four consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1 253

Fig. 6.12. Trajectory of a particle initialized at $x = 100$, over a 1-dimensional space with stationary attractors at $x = 0$, for $w = 0.20$, corresponding to two consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1. The runs considered here are consecutive to those in Fig. 6.11 254

Fig. 6.13. Region in the plane 'ϕ–w' from where ϕ is sampled in PSO-RRR2 255

Fig. 6.14. Trajectory corresponding to the average behaviour of a deterministic particle initialized at $x = 100$, flying over a 1-dimensional space, with stationary attractors at $x = 0$, for $1.00 \geq w \geq 0.30$, for the PSO-RRR2 257

Fig. 6.15. Trajectory corresponding to the average behaviour of a deterministic particle initialized at $x = 100$, flying over a 1-dimensional space, with stationary attractors at $x = 0$, for $w = 0.20$ (left) and $w = 0.10$ (right), for the PSO-RRR2 258

Fig. 6.16. Trajectory of a particle initialized at $x = 100$, flying over a 1-dimensional space, with stationary attractors at $x = 0$, for $w = 0.80$, corresponding to six consecutive runs, for the PSO-RRR2 258

Fig. 6.17. Trajectory of a particle initialized at $x = 100$, flying over a 1-dimensional space, with stationary attractors at $x = 0$, for $w = 0.50$, corresponding to six consecutive runs, for the PSO-RRR2 259

Fig. 6.18. Trajectory of a particle initialized at $x = 100$, flying over a 1-dimensional space, with stationary attractors at $x = 0$, for $w = 0.20$, corresponding to six consecutive runs, for the PSO-RRR2 260

Fig. 6.19. Trajectory of a particle initialized at $x = 100$, optimizing the 1-dimensional Sphere function, with stationary social attractor at $x = 0$ and dynamic individual attractor initialized at $x = 90$, for $w = 0.70$ (left column) and $w = 0.50$ (right column), $aw = 4.00$, for the classical PSO, corresponding to four consecutive runs 262

Fig. 6.20. Trajectory of a particle initialized at $x = 100$, optimizing the 1-dimensional Sphere function, with stationary social attractor at $x = 0$ and dynamic individual attractor initialized at $x = 90$, for $w = 0.70$ (left column) and $w = 0.50$ (right column), $aw = 4.00$, for the classical PSO, corresponding to two runs consecutive to those in Fig. 6.19 263

Fig. 6.21. Trajectory of a particle initialized at $x = 100$, optimizing the 1-dimensional Sphere function, with stationary social attractor at $x = 0$ and dynamic individual attractor initialized at $x = 90$, for $w = 0.7298$ and $aw = 2.9922$ (left column), and for $w = 0.6204$ and $aw = 2.5435$ (right column), for the C-PSO, corresponding to two consecutive runs 263

Fig. 6.22. Trajectory of a particle initialized at $x = 100$, optimizing the 1-dimensional Sphere function, with stationary social attractor at $x = 0$ and dynamic individual attractor initialized at $x = 90$, for $w = 0.7298$ and $aw = 2.9922$ (left column), and for $w = 0.6204$ and $aw = 2.5435$ (right column), for the C-PSO, corresponding to four runs consecutive to those in Fig. 6.21 264

Fig. 6.23. Trajectory of a particle initialized at $x = 100$, optimizing the 1-dimensional Sphere function, with stationary social attractor at $x = 0$ and dynamic

individual attractor initialized at x = 90, for w = 0.80 and hence aw = 3.60 (left column), and for w = 0.50 and hence aw = 3.00 (right column), for the PSO-RRM, corresponding to four consecutive runs 265

Fig. 6.24. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.80 and hence aw = 3.60 (left column), and for w = 0.50 and hence aw = 3.00 (right column), for the PSO-RRM, corresponding to two runs consecutive to those in Fig. 6.23 266

Fig. 6.25. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.80 and hence w = 0.80 (left column), and for ϕ_{mean} = 1.50 and hence w = 0.50 (right column), for the PSO-RRR1, corresponding to two consecutive runs 266

Fig. 6.26. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.80 and hence w = 0.80 (left column), and for ϕ_{mean} = 1.50 and hence w = 0.50 (right column), for the PSO-RRR1, corresponding to four runs consecutive to those in Fig. 6.25 267

Fig. 6.27. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 2.38 and hence w = 0.80 (left column), and for ϕ_{mean} = 2.00 and hence w = 0.50 (right column), for the PSO-RRR2, corresponding to four consecutive runs 268

Fig. 6.28. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 2.38 and hence w = 0.80 (left column), and for ϕ_{mean} = 2.00 and hence w = 0.50 (right column), for the PSO-RRR2, corresponding to two runs consecutive to those in Fig. 6.27 269

Fig. 6.29. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Schaffer f6 function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.80 and hence w = 0.80 (left column), and for ϕ_{mean} = 1.50 and hence w = 0.50 (right column), for the PSO-RRR1, corresponding to four consecutive runs 271

Fig. 6.30. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Schaffer f6 function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.80 and hence w = 0.80 (left column), and for ϕ_{mean} = 1.50 and hence w = 0.50 (right column), for the PSO-RRR1, corresponding to two runs consecutive to those in Fig. 6.29 272

Fig. 6.31. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Schaffer f6 function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 2.38 and hence w = 0.80 (left column), and for ϕ_{mean} = 2.00 and hence w = 0.50 (right column), for the PSO-RRR2, corresponding to two consecutive runs 272

Fig. 6.32. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Schaffer f6 function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 2.38 and hence w = 0.80 (left column), and for ϕ_{mean} = 2.00 and hence w = 0.50 (right column), for the PSO-RRR2, corresponding to four runs consecutive to those in Fig. 6.31 273

Fig. 6.33. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.70, aw = 4.00, and ip = 1.00 in the left column and ip = 0.00 in the right column, for the classical PSO, corresponding to four consecutive runs 276

Fig. 6.34. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.70, aw = 4.00, and ip = 1.00 in the left column and ip = 0.00 in the right column, for the classical PSO, corresponding to two runs consecutive to those in Fig. 6.33 277

Fig. 6.35. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.50 and hence w = 0.50, and ip = 1.00 in the left column and ip = 0.00 in the right column, for the PSO-RRR1, corresponding to two consecutive runs 277

Fig. 6.36. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.50 and hence w = 0.50, and ip = 1.00 in the left column and ip = 0.00 in the right column, for the PSO-RRR1, corresponding to four runs consecutive to those in Fig. 6.35 278

Fig. 6.37. Trajectory of a particle initialized at x = 100, with stationary attractors at x = 0 and ip = sp = 0.50, for the classical PSO with w = 0.70 and aw = 4.00, randomness included, for six consecutive runs 279

Fig. 6.38. Trajectory of a deterministic particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ = 1.50 and therefore w = 0.50, for increasing values of ip ($0 \leq ip \leq 1$), for the PSO-RRR1 280

Fig. 6.39. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.50 (w = 0.50), for ip = 0.70 (sp = 0.30), for the PSO-RRR1, corresponding to six consecutive runs 281

Fig. 6.40. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-1 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 286

Fig. 6.41. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-1 algorithm with ip = sp = 0.50 and v_{max} = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 286

Fig. 6.42. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-2 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 287

Fig. 6.43. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-2 algorithm with $ip = sp = 0.50$ and $v_{max} = fint / 2 = 100$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 287

Fig. 6.44. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-3 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 288

Fig. 6.45. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-4 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 288

Fig. 6.46. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-5 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 289

Fig. 6.47. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the C-PSO-1 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 291

Fig. 6.48. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the C-PSO-2 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 291

Fig. 6.49. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the C-PSO-1 algorithm with $ip = sp = 0.50$ and $v_{max} = fint / 2 = 100$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 292

Fig. 6.50. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-3 algorithm with $ip = sp = 0.50$ and $v_{max} = fint / 2 = 100$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 292

Fig. 6.51. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRM-1 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 294

Fig. 6.52. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRM-2 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 294

Fig. 6.53. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRM-3 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 295

Fig. 6.54. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRM-4 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 295

Fig. 6.55. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRM-5 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 296

Fig. 6.56. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRM-5 algorithm with $ip = sp = 0.50$ and $v_{max} = 100$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 296

Fig. 6.57. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR1-1 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 298

Fig. 6.58. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR1-2 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 298

Fig. 6.59. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR1-3 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 299

Fig. 6.60. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR1-4 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 299

Fig. 6.61. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR1-5 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 300

Fig. 6.62. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR2-1 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 302

Fig. 6.63. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR2-1 algorithm with $ip = sp = 0.50$ and $v_{max} = 100$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 302

Fig. 6.64. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR2-2 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 303

Fig. 6.65. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR2-2 algorithm with $ip = sp = 0.50$ and $v_{max} = 100$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 303

Fig. 6.66. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR2-3 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 304

Fig. 6.67. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR2-4 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 304

Fig. 6.68. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR2-5 algorithm with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 305

Fig. 6.69. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for the PSO-RRR2-5 algorithm with $ip = sp = 0.50$ and $v_{max} = 100$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 305

Fig. 6.70. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for (Trelea, 2003)'s settings with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 307

Fig. 6.71. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for (Hu, Eberhart, & Shi, 2003)'s settings with $ip = sp = 0.50$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 307

Fig. 6.72. Trajectories of four particles initialized at $x = 100$, $x = 50$, $x = -50$, and $x = -100$, for (Hu, Eberhart, & Shi, 2003)'s settings with $ip = sp = 0.50$ and $v_{max} = 100$, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences coincide with the initial positions 308

Fig. 6.73. Evolution of the mean measures of clustering 'pb_cge' and 'pb_me' for the 'PSO-RRR1-1 Global' optimizing the 30-dimensional Sphere function 317

Fig. 6.74. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 6.3 318

Fig. 6.75. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 6.4 319

Fig. 6.76. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 6.5 320
Fig. 6.77. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 6.6 321
Fig. 6.78. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 6.7 322
Fig. 6.79. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 6.8 323
Fig. 6.80. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 6.9 324
Fig. 6.81. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 6.10 325
Fig. 6.82. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 6.11 326
Fig. 6.83. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 6.12 327
Fig. 6.84. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 6.13 328
Fig. 6.85. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 6.14 329
Fig. 6.86. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 6.15 330
Fig. 6.87. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 6.16 331
Fig. 6.88. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 6.17 332
Fig. 6.89. Suggested region in the 'ϕ–w' plane from where ϕ is to be randomly sampled (blue dotted lines). The regions of suggested upper (ϕ_{max}) and lower (ϕ_{min}) limits of ϕ are shown in green and red dotted lines, respectively 338
Fig. 6.90. Suggested region in the 'ϕ–w' plane from where ϕ is to be randomly sampled in the PSO-RRM. The user only chooses ϕ_{mean} 340
Fig. 6.91. Suggested region in the 'ϕ–w' plane from where ϕ is to be randomly sampled in the PSO-RRR1. The user only chooses ϕ_{mean} 341
Fig. 6.92. Suggested region in the 'ϕ–w' plane from where ϕ is to be randomly sampled in the PSO-RRR2. The user only chooses ϕ_{mean} 342
Fig. 7.1. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 7.1 351
Fig. 7.2. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 7.2 351

Fig. 7.3. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.3 352
Fig. 7.4. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.3 352
Fig. 7.5. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 7.4 356
Fig. 7.6. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.5 356
Fig. 7.7. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.5 357
Fig. 7.8. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.6 357
Fig. 7.9. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.6 358
Fig. 7.10. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.6 358
Fig. 7.11. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 7.7 362
Fig. 7.12. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.8 362
Fig. 7.13. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.8 363
Fig. 7.14. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.9 363
Fig. 7.15. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.9 364
Fig. 7.16. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.9 364
Fig. 7.17. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 7.10 368
Fig. 7.18. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.11 368
Fig. 7.19. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.11 369
Fig. 7.20. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.11 369
Fig. 7.21. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.12 370
Fig. 7.22. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.12 370

Fig. 7.23. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 7.13 374
Fig. 7.24. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.14 374
Fig. 7.25. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.14 375
Fig. 7.26. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.15 375
Fig. 7.27. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.15 376
Fig. 7.28. a) Ring topology with neighbourhood-size equal to three; b) Forward topology with neighbourhood-size equal to three 381
Fig. 7.29. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 7.16 385
Fig. 7.30. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 7.17 385
Fig. 7.31. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.18 386
Fig. 7.32. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.18 386
Fig. 7.33. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 7.19 390
Fig. 7.34. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.20 390
Fig. 7.35. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.20 391
Fig. 7.36. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.21 391
Fig. 7.37. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.21 392
Fig. 7.38. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.21 392
Fig. 7.39. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 7.22 396
Fig. 7.40. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.23 396
Fig. 7.41. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.23 397
Fig. 7.42. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.23 397

Fig. 7.43. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.24 398

Fig. 7.44. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.24 398

Fig. 7.45. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 7.25 402

Fig. 7.46. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.26. 402

Fig. 7.47. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.26 403

Fig. 7.48. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.27 403

Fig. 7.49. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.27 404

Fig. 7.50. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.27 404

Fig. 7.51. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 7.28 408

Fig. 7.52. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.29 408

Fig. 7.53. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.29 409

Fig. 7.54. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.30 409

Fig. 7.55. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.30 410

Fig. 7.56. Two-dimensional search-space with feasible intervals of different size; three particles are allocated to illustrate the normalization of the distances for the nearest neighbour algorithm 414

Fig. 7.57. Example of generating three groups of particles by means of the Nearest Neighbour heuristics 414

Fig. 7.58. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 7.31 415

Fig. 7.59. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 7.32 416

Fig. 7.60. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.33 417

Fig. 7.61. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 7.34 418

Fig. 7.62. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.35 419
Fig. 7.63. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.36 420
Fig. 7.64. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 7.37 421
Fig. 7.65. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.38 422
Fig. 7.66. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.39 423
Fig. 7.67. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 7.40 424
Fig. 7.68. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.41 425
Fig. 7.69. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.42 426
Fig. 7.70. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 7.43 427
Fig. 7.71. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.44 428
Fig. 7.72. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.45 429
Fig. 7.73. Sub-neighbourhoods with INDIVIDUAL overlapping 433
Fig. 7.74. Sub-neighbourhoods with LOCAL overlapping 433
Fig. 7.75. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 7.46 437
Fig. 7.76. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 7.47 437
Fig. 7.77. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.48 438
Fig. 7.78. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.48 438
Fig. 7.79. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 7.49 442
Fig. 7.80. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.50 442
Fig. 7.81. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.50 443

Fig. 7.82. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.51 443
Fig. 7.83. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.51 444
Fig. 7.84. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.51 444
Fig. 7.85. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 7.52 448
Fig. 7.86. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.53 448
Fig. 7.87. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.53 449
Fig. 7.88. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.54 449
Fig. 7.89. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.54 450
Fig. 7.90. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.54 450
Fig. 7.91. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 7.55 454
Fig. 7.92. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.56 454
Fig. 7.93. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.56 455
Fig. 7.94. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.57 455
Fig. 7.95. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.57 456
Fig. 7.96. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.57 456
Fig. 7.97. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 7.58 460
Fig. 7.98. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.59 460
Fig. 7.99. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.59 461
Fig. 7.100. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.60 461
Fig. 7.101. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.60 462

Fig. 8.1. Example of the initialization procedure for three sub-neighbourhoods, one sub-swarm, and nearest neighbourhood procedure for the generation of the sub-neighbourhoods 480

Fig. 8.2. Pseudo Adaptive tolerance (for the PM) for inequality constraint violations in problem g01. The average is among the 25 runs 485

Fig. 8.3. Pseudo Adaptive tolerance (for the PM) for equality constraint violations in problem g03. The average is among the 25 runs 485

Fig. 8.4. Pseudo Adaptive tolerance (for the PM) for inequality and equality constraint violations in problem g05. The average is among the 25 runs 485

Fig. 8.5. Pseudo Adaptive tolerance (for the PM) for equality constraint violations in problem g13. The average is among the 25 runs 486

Fig. 8.6. Evolution of the mean best and average conflict values among 25 runs for problem g05 (for the PM). The shape of the curve is due to the tolerance relaxations 486

Fig. 8.7. Illustration of the Pressure Vessel Design problem 496

Fig. 8.8. Convergence of the mean best solution for the Mixed-Discrete Pressure Vessel Design problem 498

Fig. 8.9. Convergence of the mean best solution for the continuous Pressure Vessel Design problem 499

Fig. 8.10. Illustration of the Welded Beam Design problem 499

Fig. 8.11. Convergence of the mean best solution for the Welded Beam Design problem 501

Fig. 8.12. Illustration of the Tension/Compression Spring Design problem 502

Fig. 8.13. Convergence of the mean best solution for the Tension/Compression Spring Design problem 503

Fig. 8.14. Convergence of the mean best solution for the Tension/Compression Spring Design problem 505

Fig. 8.15. 10-Bar Truss problem 506

Fig. 8.16. Convergence of the mean best solution for the 10-Bar Truss problem with stress constraints 508

Fig. 8.17. Convergence of the mean best solution for the 10-Bar Truss problem with stress constraints found by the ‘PSO-RRR2-1 P.AD. PFPR (100-5000)’ 510

Fig. 8.18. Convergence of the mean best solution for 10-Bar Truss problem with stress and 2 equality constraints (*y*-displacements of nodes ‘1’ and ‘3’) 512

Fig. 8.19. Convergence of the mean best solution for the 10-Bar Truss problem with stress and four inequality-displacement constraints (nodes ‘1’ to ‘4’) 516

Fig. 8.20. 25-Bar Truss problem 519

Fig. 8.21. Convergence of the mean best solution for the 25-Bar Truss problem with Load Case 1 521

Fig. 8.22. Convergence of the mean best solution for the 25-Bar Truss problem with Load Case 2 523

Fig. 8.23. Convergence of the mean best solution for the 25-Bar Truss problem with two Load Cases as in (Fleury & Schmit, 1980) and in (Li, Huang, Liu, & Wu, 2007) 529

Fig. 8.24. Altair Hyperworks (OptiStruct) model of the 25-bar truss problem with Load Case 1 534

Fig. 8.25. Convergence of the mean best solution for the 25-Bar Truss problem as in (Park & Ryu, 2004) 535

Fig. AI.1. Example of the parabola associated to Eq. (AI.41) for $\phi = 4$. The interval where the function is smaller than zero (complex roots in the 'ϕ–w' plane) is between the roots of the parabola A6

Fig. AII.1: Surface plot and colour-map of the 2-dimensional Sphere function within the region $[-100,100]^2$ A10

Fig. AII.2: Surface plot and colour-map of the 2-dimensional Rosenbrock function within the region $[-30,30]^2$ A11

Fig. AII.3: Surface plot and colour-map of the 2-dimensional Rosenbrock function within the region $[-1.5,2.0]^2$ A11

Fig. AII.4: Surface plot and colour-map of the 2-dimensional Rosenbrock function within the region $[0.8,1.2]^2$ A12

Fig. AII.5: Surface plot and colour-map of the 2-dimensional Rastrigin function within the region $[-5.12,5.12]^2$ A13

Fig. AII.6: Surface plot and colour-map of the 2-dimensional Rastrigin function within the region $[-1.0,1.0]^2$ A13

Fig. AII.7: Surface plot and colour-map of the 2-dimensional Griewank function within the region $[-600,600]^2$ A14

Fig. AII.8: Surface plot and colour-map of the 2-dimensional Griewank function within the region $[-60,60]^2$ A15

Fig. AII.9: Surface plot and colour-map of the 2-dimensional Griewank function within the region $[-10,10]^2$ A15

Fig. AII.10: Surface plot and colour-map of the 2-dimensional Schaffer f6 function within the region $[-100,100]^2$ A16

Fig. AII.11: Surface plot and colour-map of the 2-dimensional Schaffer f6 function within the region $[-30,30]^2$ A16

Fig. AII.12: Surface plot and colour-map of the 2-dimensional Schaffer f6 function within the region $[-5,5]^2$ A17

Fig. AII.13: Surface plot and colour-map of the 2-dimensional Schaffer f6 function within the region $[-1,1]^2$ A17

Main Acronyms

AI / CI / SI: Artificial / Computational / Swarm Intelligence.

AL: Artificial Life.

ANN: Artificial Neural Network.

AS / ACO: Ant System / Colony Optimization.

aw: Acceleration weight, where $aw = iw + sw = \phi_{max}$ in classical PSO.

CEs: Constraint Evaluations.

cf: Constriction factor.

CHT: Constraint-Handling Technique.

cv: Constraint violations.

DE: Differential Evolution.

DoE: Design of Experiments.

DoL: Division of Labour.

EA / EP / EC: Evolutionary Algorithm / Programming / Computation.

ES: Evolution Strategy.

FE: Finite Element.

FEs: Function Evaluations.

fint: Feasible interval.

FR: Feasibility Ratio.

FSM: Finite State Machine.

GA / GP: Genetic Algorithm / Programming.

HNP: Himmelblau's Nonlinear Problem.

ip, *sp*: Individuality/Sociality Percentage.

iw, *sw*: Individuality/Sociality weight.

LC: Load Case.

LHS: Latin Hypercube Sampling.

MR / CR: Mutation / Crossover Rate.

MS: Multi Swarm.

nn: Number of neighbours.

NN: Nearest Neighbour heuristics.

NNB: Nearest Neighbour heuristics using the Best particles' experience.

nni, *nnf*: Initial / Final number of neighbours.

PF / PFPR: Preserving Feasibility / Preserving Feasibility with Priority Rules.

PM: Penalization Method.

PSO: Particle Swarm Optimization / Optimizer.

PVD: Pressure Vessel Design.

SNI: Sub-Neighbourhoods with Individual overlapping.

SNL: Sub-Neighbourhoods with Local overlapping.

SO: Self-Organization.

SR: Success Rate.

TCSD: Tension/Compression Spring Design.

Tol_{eq}: Tolerance for equality constraint violations.

Tol_{ineq}: Tolerance for inequality constraint violations.

TSP: Travelling Salesman Problem.

v_{max}: Maximum velocity.

WBD: Welded Beam Design.

Main Glossary

aw:	Acceleration weight, where $aw = iw + sw = \phi_{\max}$ in classical PSO.
CEs:	Number of Constraint functions Evaluations.
cf:	*Type 1"* constriction factor.
Conflict:	Equivalent to *fitness* in EAs.
cv:	Constraint Violations.
$f(\mathbf{x})$:	Objective function (to be minimized). Also *cost* (traditional optimization) *fitness* (EAs) or *conflict* (PSO) *function* within this thesis.
FEs:	Number of objective Function Evaluations.
fint:	Feasible interval.
FR:	Feasibility Ratio.
$g(\mathbf{x})$:	Constraint function (inequality or equality).
ip, *sp*:	Individuality/Sociality Percentage, where $ip, sp \in (0,1)$. They state the part of the acceleration that is assigned to the individuality and to the sociality terms in the velocity update equation.
iw, *sw*:	Individuality/Sociality weight.
m:	Number of particles in the swarm.
n:	Number of dimensions of the search-space (i.e. number of variables).
nn:	Number of neighbours.
nni, *nnf*:	Initial / Final number of neighbours.
SR:	Success Rate.
Tol_{eq}:	Tolerance for equality constraint violations.
Tol_{ineq}:	Tolerance for inequality constraint violations.
$U_{(a,b)}$:	Random number from a uniform distribution in the range [a,b] resampled anew every time it is referenced.

Variables: Design parameters in the optimization problem. Indistinctly referred to as ‘variables’, ‘design variables’, or as ‘object variables’. The term ‘parameters’ is avoided to prevent from confusion with ‘coefficients’.

$\mathbf{v}_{max}$: Velocity constraint (in the general case, it is an *n*-dimensional vector).

ϕ: Acceleration coefficient, where $\phi = iw \cdot U_{(0,1)} + sw \cdot U_{(0,1)}$ in classical PSO. Thus the difference with *aw* is that ϕ is the actual value scaling the difference vector pulling the particle towards the averaged attractor.

ϕ_i / ϕ_s: Individual / Social acceleration coefficient.

χ: Constriction factor.

Chapter 1

INTRODUCTION

1.1. Introduction

The optimization of a system –either specifically intended or simply occurring without purpose or awareness– consists of the attainment or pursuit of its best performance in some sense. The system can be 'natural and inanimate'; 'natural and biological'; 'artificial and tangible'; or 'artificial and abstract'. Thus, 'natural and inanimate systems' optimize themselves as they tend to reach minimum energy states[1]; 'biological systems' optimize their own genetic code[2] and behaviour[3] in their quest for best performance as they strive to survive within the environment; 'artificial and tangible systems' may be optimized, for instance, by seeking the optimum shape of an engineering structure so as to maximize stiffness, minimize weight, and/or minimize cost; 'artificial and abstract systems' are optimized either by solving the optimization of the model of 'artificial and tangible systems' or by optimizing abstract systems 'per se' such as analytical functions, scheduling problems, the design of working plans or control systems, etc.

If the model of the system can be parameterized, an optimization process consists of searching for the best combination of feasible values of those parameters which results in its best performance. Deciding upon the meaning of 'best performance' is not always trivial. Some classical meanings of 'best' may be *minimum energy*; *maximum fitness*; *minimum weight*; *minimum cost*; *maximum benefit*; *minimum error*; *minimum time*; *minimum conflict*; etc. More than one objective may be required to be optimized. Different objectives are often conflicting, making the definition of 'best' more ambiguous. Multi-objective optimization problems are beyond the scope of this thesis.

[1] e.g. crystallization in nature (*optimizing* inner structures to minimize energy)
[2] e.g. biological evolution (*maximizing* fitness of natural organisms)
[3] e.g. colonies of ants finding the *shortest* path from the nest to a food source

1.2. Motivation

While the concept of optimization is precise and specific, optimization problems and processes can be observed everywhere in both natural and artificial systems. In natural systems, they occur without sense of purpose, central control, or even awareness. In artificial systems –including models of natural systems– optimization is intended, although the solving technique may use concepts of natural systems where the link between the procedures being carried out and the resulting optimization process is far from obvious. Such is the case of the *Particle Swarm Optimization* (PSO) method.

Some of these artificial systems are directly optimization problems, such as 'finding the shape of a submarine to minimize friction' or 'finding the shape of a channel to maximize its hydraulic radius'. Others can be indirectly viewed as optimization problems by generating an error function that is to be minimized, such as 'root finding'; 'pattern recognition'; 'solving systems of equations', 'approximations', 'training neural networks', etc. Therefore optimization problems arise in different disciplines such as mathematics; physics; chemistry; general engineering; systems engineering; computer science; economics; genetics; biology; etc. Hence specializing on optimization –as opposed to most scientific specializations– does not lead to a narrow range of applications. On the contrary, it is probably the most 'general specialization', as almost any problem can be posed as an optimization problem.

Traditional optimization methods are limited by restrictions that the functions involved must comply with for the method to be applicable. A problem may be significantly altered to meet those restrictions, drifting away from the original problem. This thesis is concerned with the *Particle swarm Optimization* method, which differs from traditional methods in that it poses no restriction to the functions involved. The method was inspired by the behaviour observed in social animals in nature. The idea is that the achievements of a population overcome the sum of their individual achievements. Useful, individually acquired information is spread throughout the population by means of local interactions. Individuals converge as they imitate their most successful peers. Thus, cooperation overcomes competition. A typical example is a bird flock searching for food: by sharing individually acquired information, birds are able to find a food source without prior knowledge regarding its location. The robustness of the method

lies in that it is not deterministically implemented to optimize, and therefore there is no problem-specific implementation that may be inadequate for a different problem. The optimizing behaviour is an emergent property that is not specifically programmed. All that is needed is a means of evaluating positions in an environment so as to differentiate better from worse. This feature makes the method especially suitable for optimizers to be applied to a wide range of different problems with reasonably good performance without modifications, and with very limited tuning or none at all. In addition, they are able to deal with complex problems where traditional methods fail.

1.3. Objectives

The *Particle Swarm Optimization* method is suitable for unconstrained problems only, whose performance depends critically on the settings of the coefficients in the velocity update equation, as well as on the structure of the social network by means of which the individuals share the acquired information. This thesis intends to carry out a thorough study of the influence of the coefficients on the dynamics of the swarm, and to obtain some range of settings that can be exploited according to the type of behaviour desired. The latter includes robust, general-purpose settings. The same is true for the social network, and therefore this thesis aims to investigate some classical neighbourhood topologies as well as to develop new ones. Their strengths and weaknesses are to be identified so as to allow making the appropriate choice when required. Finding or developing general-purpose neighbourhood structures is also of interest. Finally, the development of successful constraint-handling mechanisms to be incorporated to the algorithm is critical for the applicability of the method to real-life problems, which tend to arise subject to a number of constraints.

The particle swarm optimizer to be developed in this thesis is a global, single-solution, single-objective, gradient-free, population-based method, which is to be able to handle continuous –exceptionally, discrete variables by rounding-off–, constrained and unconstrained, single-objective problems regardless of whether the functions involved are or are not linear, convex, unimodal, differentiable, smooth, continuous, or even explicit. Only a means of measuring the goodness and feasibility of a solution should be re-

quired. The developed optimizer and some of its different features are to be tested on a set of benchmark problems, and applied to a set of well known engineering problems.

1.4. Methodology

The influence of the settings of the basic coefficients in the velocity update equation on the trajectory of the particles is first analyzed within some mathematical framework on an isolated, deterministic particle pulled by stationary attractors. The form and speed of convergence as well as the divergence of this isolated deterministic particle are studied. As complexity is gradually added to the system towards the full particle swarm paradigm, the studies become more heuristic and visual-based. Once the full system is operating, the type of behaviour and the performance are evaluated numerically by testing it on benchmark problems. Sets of coefficients leading to different forms of convergent behaviour are identified. Modifications to the core of the canonical PSO algorithm are proposed, aiming to control both the average behaviour and the strength awarded to randomness on the erratic and swarming trajectory of the particles. Some coefficients' settings and a combination of them are numerically tested on benchmark problems for the global and a local neighbourhood topology. Guidelines as to how to choose the settings of the coefficients in order to obtain the desired behaviour for a given particular problem and available resources are developed.

The structure of the social network is studied, and different classical and proposed neighbourhood topologies –as well as some related heuristics– are numerically tested on unconstrained benchmark problems combined with different coefficients' settings. Their strengths and weaknesses are identified.

Finally, a pseudo adaptive constraint-handling mechanism unrelated to the social behaviour metaphor that inspired the method is proposed and coupled to the algorithm. The fully working method is tested on a classical benchmark suite of constrained problems and compared with results reported by other authors. Successful applications to well known engineering problems are also provided, together with results obtained by other authors in the literature for reference and comparison purposes.

1.5. Achievements

A fully in-house particle swarm optimizer for real-valued –exceptionally discrete or mixed-discrete– constrained and unconstrained problems was developed, implemented, and tested, showing good and robust performance on a wide range of different problems with few or no adaptations and no problem-specific tuning.

An extensive study of the influence of the settings of the coefficients in the velocity update equation –as well as those of the velocity constraint– on the behaviour and performance of the method was carried out partly theoretically, partly heuristically, partly visualizing trajectories, and partly by means of numerical experiments. Convergence conditions were developed and the impact of the coefficients' settings on the speed and form of convergence was analyzed. In turn, the impact of the speed and form of convergence on the performance of the optimizer was discussed.

A reformulation of the basic update equations was proposed, which allows for better control of the strength of randomness and average behaviour desired. Mappings between the proposed and classical formulations were offered, and guidelines were provided for the settings of the coefficients and of the velocity constraint.

Two dynamic neighbourhood topologies were proposed, namely the *ring* and the *forward* topologies *with time-increasing number of neighbours*. A *nearest neighbourhood* heuristics and the *sub-neighbourhood* option were proposed and investigated. Different combinations of neighbourhood topologies and coefficients' settings were considered and tested on a set of unconstrained problems. Conclusions were derived with regards to the behaviour that results from the different combinations. The *dynamic forward topology* appeared to be robust, while the convenience of activating the nearest neighbourhood heuristics and/or the sub-neighbourhood option remains unclear.

A robust pseudo adaptive constraint-handling technique was proposed, implemented, coupled, and successfully tested on a benchmark suite of constrained problems. Results were compared to those obtained using classical techniques and to results reported by other authors in the literature.

Several well known engineering problems were successfully tackled with the in-house particle swarm optimizer developed within this thesis.

1.6. Layout of the thesis

The summary; declaration and statements; table of contents; acknowledgements; list of tables and figures; main acronyms; and main glossary precede the first chapter of this thesis. Chapter 1 presents the motivation and objectives driving the research carried out hereafter, as well as the methodology used and the main achievements. The layout of the thesis is offered at the end of Chapter 1.

The main body of this thesis is structured into four broad sections:

SECTION I comprises three chapters. Chapter 2 is devoted to a review of the formulation of the constrained optimization problem; Chapter 3 offers an overview of the Evolutionary Computation field and its main paradigms; while Chapter 4 presents the Particle Swarm Optimization method, its roots, and its developments.

SECTION II is composed of four chapters. A mathematically relatively formal and systematic study of a simplified system –as undertaken by other authors in the literature– is carried out in Chapter 5; complexity is gradually incremented until the full system is restored in Chapter 6, where three approaches are proposed modifying the core of the PSO algorithm by controlling the range of randomness and by choosing the desired average behaviour; the social network in the population is studied and developed in Chapter 7, proposing a dynamic number of neighbours, a forward topology, and some additional neighbourhood-related heuristics; a pseudo adaptive constraint-handling mechanism is incorporated in Chapter 8, which is tested on a set of constrained benchmark problems, and applied to a set of well-known engineering optimization problems.

SECTION III is exclusively composed of Chapter 9, where conclusions are offered and lines for future research are suggested.

SECTION IV contains three Appendices. Some additional, tedious, burdensome arithmetic work supporting the coefficients' studies are offered in Appendix I; the formulations of two suites of benchmark problems –one unconstrained and one constrained– are included in Appendix II; and finally a digital appendix is provided. The latter contains a digital copy of the thesis; a number of files supporting the research including a number of images; and the outputs of the experimental studies and tests.

The list of References and Bibliography is offered between SECTIONS III and IV.

SECTION I

BACKGROUND

Chapter 2

CONSTRAINED OPTIMIZATION

This chapter consists of a brief review of the optimization field. Some basic concepts are presented, and the continuous constrained optimization problem is posed in a way that is convenient to be tackled with a particle swarm optimizer. A quick overview of different families of optimization problems and methods is offered, without extending the review to discussions on, or descriptions of specific methods. In addition, a few notes with regards to tackling constrained optimization problems with particle swarm optimizers are interlaced with the constrained optimization review.

2.1. Introduction

For problems where the quality of a solution can be quantified in a numerical value, *optimization* is the process of seeking the permitted combination of variables that optimizes that value. Different combinations of *variables* allow trying different candidate solutions, the *constraints* limit the valid combinations, and the *optimality criteria* allow differentiating better from worse.

When solving mathematical problems such as equations, systems of equations or root-finding, there might be one or more solutions, but all of them share the same quality. The equivalent in optimization problems would be to find the very best solution(s) possible (i.e. the exact global optima), which is not always possible in real-world problems. Besides, when comparing solutions in many real-world problems, the classification of a solution being better than another is not always unambiguous. Thus, optimization problems allow for a number of solutions which differ in their degree of goodness. In general, it is not even certain whether the best solution possible has been found –or how good the solution found is– by the time the search is terminated.

The process of identifying objective, variables, and constraints for a given problem is known as modeling. (...) If the model is too simplistic, it will not give useful insights into the practical problem. If it is too complex, it may be too difficult to solve. (Nocedal & Wright, 2006)

Loosely speaking, two main approaches can be considered here. First, a very simplified model can be developed so that available, traditional, methods are able to solve it. Al-

ternatively, more sophisticated models can be designed, and approximate the solving techniques rather than the model. Traditional solving methods require specific features of the functions involved, so that the model may be forced to adopt assumptions and simplifications required to suit the solver. This results in the exact solution of an approximate model (e.g. analytical solutions to structural engineering problems; linear programming; analytical solutions to convex optimization problems). The types of problems that can be tackled by this approach are typically rather limited. The approximate solution of a more precise model is commonly preferred for complex real-world problems. Thus, more precise models are designed, which are to be solved by more modern, approximate methods such as 'modern heuristics' and 'finite element methods'. The latter approach results in an approximate solution of a more sophisticated model.

A meaningful model of the problem is vital. The formulation of the problem involving the design of a cost function that successfully measures the optimality and the definition of the constraint functions are critical to obtain useful solutions. The complete analysis of an optimization problem is performed in four broad stages:

1. Analysis and definition of the problem.
2. Formulation of the problem (development of the model).
3. Solution of the model.
4. Validation of the model.

Only the solution of a well-posed mathematical model is dealt with here, while the other stages are beyond the scope of this thesis. Thus, the problem of optimization is typically posed as a function of some object variables, often in the presence of constraints. The validation of the problem-solving technique –namely a particle swarm optimizer– is performed by testing it on benchmarking problems. The stages of modelling and validating the model are not considered.

Only single-objective optimization problems are handled within this thesis. While the *objective* of the problem is given in plain words, its *formulation* is called the *objective function*, whose output consists of information about the problem as a function of the object variables. Thus, the *objective function* relates the real problem to the model. In turn, the *cost function* –also *evaluation*, *fitness*, or *conflict function*– is the scalar function to be optimized, whose output is a scalar measure of the fulfilment of the objective.

Therefore, the cost function allows evaluating how good a given solution is when compared to others. The objective and cost functions might coincide, or there might be some mapping between them. For simplicity, the names *objective*, *cost*, and *conflict* function will be used indistinctly within this thesis to refer to the function to be optimized. The *problem variables* are also called *unknowns*, *object variables*, *design variables*, or simply *variables*. Since *parameters* might stand for either *variables* or *coefficients*, the use of the term is avoided.

2.2. Types of constraints

The most appropriate technique to handle a constraint usually depends on the type of constraint. Some common types are as follows:

- *Inequality constraint*: Function of the variables that must be smaller than or equal to a constant.
- *Equality constraint*: Function of the variables that must be equal to a constant.
- *Boundary constraint*: Instance of inequality constraints, consisting of functions that define boundaries that contain the feasible space; if the boundary constraints are given by a hyper-rectangle, they are also called *interval* or *side constraints*.

While a constrained optimization problem is defined as the problem of finding the combination of variables that minimizes the cost function while satisfying all constraints, real-world problems sometimes do not lend themselves to such strict conditions. Frequently, all constraints cannot be strictly satisfied simultaneously, and the problem turns into finding a trade-off between minimizing the cost function and minimizing the constraints' violations. Thus, another important classification of the types of constraints is:

- *Hard constraint*: It does not admit any degree of violation.
- *Soft constraint*: There is some given tolerance for the constraint violation.

Since a tolerance is required for particle swarm optimization algorithms to cope with such constraints, only *soft equality constraints* are considered within this thesis. In other words, hard equality constraints cannot be handled.

2.3. Optimization problems

Different types of optimization problems require different approaches to be coped with. The first important difference is in the type of solution sought. Problems whose optimum solution is a scalar or vector of scalars are called *parameter optimization* problems (also *mathematical programming*). Problems whose optimum solution is a function or vector of functions are called *variational* problems. For instance, consider a drainage pipe. A variational problem would be to find its optimum cross section. That is, the function (cross-section) that optimizes a functional (e.g. minimum hydraulic radius). Branches of the optimization field dealing with the latter types of problems are *shape optimization* and *topology optimization*. In mathematics, similar concepts would be to solve an equation or system of equations (solution is a scalar or vector), as opposed to solving differential equations or systems of differential equations (solutions are functions or vectors of functions). In this thesis, only *parameter*[1] *optimization* problems are dealt with. That is, optimization problems whose solutions can be represented by *n*-dimensional vectors. They will be referred to simple as *optimization problems* from here forth. Thus, *optimization problems* can be classified according to the kind of model that can represent them. Considering the characteristics of the variables, they can be classified into *continuous* –i.e. real-valued–, *discrete*, *mixed-discrete*, *binary*, or *combinatorial*. Taking into account the characteristics of the cost and constraint functions, they can be classified into *linear* or *nonlinear*; *convex* or *nonconvex*; *unimodal* or *multimodal*; *differentiable* or *nondifferentiable*; *smooth* or *nonsmooth*; *constrained* or *unconstrained*; *single-objective* or *multiobjective*; *stationary* or *dynamic*; *explicit* or *implicit*; *constraint-satisfaction*; etc.

- **Continuous optimization** problem. Its variables are real-valued, so that they can be represented by position vectors in an *n*-dimensional space. Note that the feasible space and/or the cost and constraint functions need not be continuous. Since there are infinite possible solutions, exhaustive search is not an option. For a review of continuous optimization, refer to any standard text such as (Nocedal & Wright, 2006), (Pedregal, 2004) or (Novo Sanjurjo, 1999).

[1] In this context, the term 'parameters' clearly refers to the variables, whereas in the context of 'parameter tuning' it refers to coefficients. To prevent misunderstandings, the use of the term is avoided in this thesis.

- **Discrete optimization** problem. Its variables can only take a set of discrete values, which is typically finite. Nonetheless, exhaustive search is –in general– not feasible because the number of possible solutions tends to be overwhelming in real-world problems. If the variables are constrained to integer values, it is called *integer optimization* (or *integer programming*) problem. If the discrete structure that represents a solution is other than a vector (e.g. lists, graphs, matroids, etc.), it is called *combinatorial optimization* problem. If some variables are real-valued and some are discrete (or integers), the problem is said to be mixed-discrete (or mixed-integer). For a review of discrete optimization, refer to standard texts on discrete mathematics –e.g. (Johnsonbaugh, 1999)–, integer optimization –e.g. (Wolsey, 1998)– and combinatorial optimization –e.g. (Korte & Vygen, 2006)–.

- **Binary optimization** problem. It is a type of discrete optimization problem where the variables can only take one of two values (true/false, on/off, negative/positive, cold/hot, etc.). Therefore they are usually decision problems. However, almost any problem can be posed as a binary problem (almost anything can be represented to any degree of precision by a binary alphabet) and hence be tackled with a binary algorithm. An example is the use of binary genetic algorithms to handle continuous optimization problems. Even though the search-space consists of a hyper-cube whose finite number of vertices comprises all possible solutions, such a number is typically insurmountable. Hence exhaustive search is not –in general– plausible.

- **Combinatorial optimization** problem. It is a type of discrete problem where the aim is to find the optimal arrangement of elements so as to optimize a result. They are typically formulated in terms of graphs (or other discrete structures) rather than vectors of variables, and the search is permutation-driven. Therefore these problems present a finite number of possible solutions, although this number tends to be intractable for real-world problems. The field of combinatorial optimization is closely linked to graph theory and combinatorics. For a review of combinatorial optimization, refer to (Korte & Vygen, 2006). Some combinatorial problems can be formulated as integer optimization problems (e.g. shortest-path problems).

- **Linear/Nonlinear optimization** problem. A linear problem is that whose cost and constraint functions are all linear. Therefore, the global solution lies on a vertex of a

polyhedron. If the variables are constrained to take integer values, the problem is often referred to as an *integer programming* problem, as short for *integer linear programming* problem.

- **Convex optimization** problem. A problem is said to be convex if both the cost function and the feasible space are convex. This is a desirable feature, which makes the problem much easier to be solved. If a local minimum exists in a convex problem, it is also a global minimum. A linear optimization problem is a particular case of convex optimization problems. For a review of convex optimization, refer to any standard text on continuous optimization such as (Nocedal & Wright, 2006).
- **Unimodal/Multimodal optimization** problem. A problem is said to be unimodal if there is only one optimum and multimodal otherwise.
- **Differentiable/Nondifferentiable optimization** problem. A problem is said to be differentiable if the cost and the constraint functions are continuously differentiable in the domain. Thus, the solving techniques are based on differential calculus.
- **Smooth/Nonsmooth optimization** problem. A problem is said to be smooth if the second derivatives of the cost and constraint functions exist and are continuous.
- **Constrained/Unconstrained optimization** problem. A problem is unconstrained if there are no restrictions to the values that the object variables can take. In other words, the whole of the search-space is feasible. There is some inconsistency in the nomenclature in the literature when only interval constraints are present. In this case, it is not rare to find the problem referred to as *constrained* or *unconstrained*.
- **Single-objective/Multiobjective optimization** problem. If the problem presents only one objective function to be optimized, it is said to be single-objective. If there is more than one objective to be optimized simultaneously, the problem is said to be multiobjective. Different objectives are often in conflict, such as the case of designing an engineering structure with the lightest weight, the greatest stiffness, and the lowest cost. All objectives cannot –in general– be optimized, and the problem consists of finding a trade-off between the objectives. A single solution is not sought in these problems but a set of solutions. The aim is to optimize each objective only up to the point when further optimization of that objective degrades some of the others.

These compromising solutions are called *Pareto-optimal set*, while the plot of the corresponding objectives in the *objectives space* is called the *Pareto front*.

- **Dynamic optimization** problem. It is a problem whose output is a function of time, so that optima change in value and position as a function of it. They would be more accurately defined as optimization problems with dynamic environments. Refer to (Morrison, 2004) for tackling these problems using Evolutionary Algorithms. Note that *dynamic programming* might also refer to the strategy of solving a complex optimization problem by breaking it down into smaller, easier sub-problems.
- **Explicit/Implicit optimization** problem. The cost function is explicitly written as a mathematical function of the object variables. This is not always possible, and sometimes some procedure needs to be carried out to obtain the cost value corresponding to a given vector of object variables (e.g. a finite element model; an artificial neural network; lab tests; etc.).
- **Constraint-satisfaction optimization** problem. Its objective function is constant, and hence the optimization process consists of minimizing the constraint violations. A common case is when all constraints cannot be satisfied simultaneously.

This thesis deals with *continuous* (real-valued) –exceptionally, discrete variables are handled by rounding-off–, *constrained* and *unconstrained*, *single-objective*, *stationary* problems, regardless of whether the functions involved are or are not *linear*, *convex*, *unimodal*, *differentiable*, *smooth*, *continuous*, or even *explicit*. For simplicity, these types of problems are referred to simply as **optimization problems** from here forth.

2.4. Optimization problem formulation

Let $\mathbb{S} \subseteq \mathbb{R}^n$ be an *n*-dimensional search-space, and $\mathbb{F} \subseteq \mathbb{S}$ its feasible part. A **global** minimization problem consists of finding $\mathbf{x}^* \in \mathbb{F}$ such that:

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \ \forall \mathbf{x} \in \mathbb{F} \tag{2.1}$$

where $f(\mathbf{x}^*)$ is a global minimum and $\mathbf{x}^*$ its location. The feasible space $\mathbb{F}$ may be defined as in Eq. (2.2), while the problem can be formulated as in Eq. (2.3):

$$\mathbb{F} = \left\{ \mathbf{x} \in \mathbb{S} \mid g_j(\mathbf{x}) \leq 0 \;\; \forall j \in \{1,...,q\} \;\wedge\; g_j(\mathbf{x}) = 0 \;\; \forall j \in \{q+1,...,q+r\} \right\} \tag{2.2}$$

$$\text{Minimize } f(\mathbf{x}), \quad \text{subject to } \begin{cases} g_j(\mathbf{x}) \leq 0 & ; \quad j = 1, \ldots, q \\ g_j(\mathbf{x}) = 0 & ; \quad j = q+1, \ldots, q+r \\ l_i \leq x_i \leq u_i & ; \quad i = 1, \ldots, n \end{cases} \tag{2.3}$$

where $\mathbf{x} \in \mathbb{S} \subseteq \mathbb{R}^n$ is the vector of variables; $f(\cdot): \mathbb{S} \to \mathbb{R}$ is the cost function; and $g_j(\cdot): \mathbb{S} \to \mathbb{R}$ is the jth (inequality or equality) constraint function, and n is the number of dimensions of the search-space (i.e. the number of object variables).

Since $\max(f(\mathbf{x})) = -\min(-f(\mathbf{x}))$, optimization stands for minimization from here on, and maximization problems are conveniently reformulated. Similarly, the *less than or equal to* constraints are not restrictive, as $g_j(\mathbf{x}) \geq 0$ implies $-g_j(\mathbf{x}) \leq 0$.

The canonical particle swarm optimization algorithm cannot comply with hard equality constraints unless the problem is reformulated to embed them into the cost function (e.g. problems G3 and G11 in (Hu & Eberhart, 2002)), which is problem-dependent and not always possible (e.g. problem G5 in (Hu & Eberhart, 2002)). Therefore, equality constraints need to be relaxed by setting an acceptable tolerance below which the solution is considered feasible. Given that $g_j(\mathbf{x}) = 0$ is equivalent to $\text{abs}(g_j(\mathbf{x})) \leq 0$ and $l_i \leq x_i \leq u_i$ is equivalent to $(\max(0, x_i - u_i) + \max(0, -x_i + l_i)) \leq 0$, the problem in Eq. (2.3) can be reformulated into three groups of inequality constraints as shown in Eq. (2.4), where the tolerances are also incorporated.

$$\text{Minimize } f(\mathbf{x})$$
$$\text{subject to } \begin{cases} g_j(\mathbf{x}) \leq Tol_{ineq} & ; \quad j = 1, \ldots, q \\ \text{abs}(g_j(\mathbf{x})) \leq Tol_{eq} & ; \quad j = q+1, \ldots, q+r \\ \max(0, x_i - u_i) + \max(0, -x_i + l_i) \leq 0 & ; \quad i = 1, \ldots, n \end{cases} \tag{2.4}$$

The amount of constraint violations (*cv*) is calculated as shown in Eq. (2.5):

$$cv = \sum_{j=1}^{q} \max(0, g_j(\mathbf{x})) + \sum_{j=q+1}^{q+r} \text{abs}(g_j(\mathbf{x})) + \sum_{i=1}^{n} [\max(0, x_i - u_i) + \max(0, -x_i + l_i)] \tag{2.5}$$

Local optimum

It is said that $\mathbf{x}^*_{\mathrm{B}}$ is the location of a **local** minimum of the function $f(\cdot)$ if:

$$\exists \varepsilon \in \mathbb{R} > 0 \mid f(\mathbf{x}^*_{\mathrm{B}}) \leq f(\mathbf{x}) \quad \forall \mathbf{x} \in [\mathbb{B}(\mathbf{x}^*_{\mathrm{B}}, \varepsilon) \cap \mathbb{F}] \tag{2.6}$$

where the region $\mathbb{B} \subseteq \mathbb{S} \subseteq \mathbb{R}^n$ is a *hyper-sphere* (or *ball*) as defined in Eq. (2.7):

$$\mathbb{B}(\mathbf{x}^*_{\mathrm{B}}, \varepsilon) = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{x}^*_{\mathrm{B}}\| \leq \varepsilon\} \tag{2.7}$$

where $\|\cdot\|$ is the Euclidean norm.

Most traditional methods to deal with nonlinear problems are iterative and, starting from an initial point $\mathbf{x}_0 \in \mathbb{B}$, are able to find the local optimum within that region.

In binary search-spaces, a region $\mathbb{B}$ is defined as all the points that are separated from $\mathbf{x}^*_{\mathrm{B}}$ by a hamming-distance equal to one –rather than the Euclidean norm–, where the hamming-distance between two points is the number of bits that need to be flipped to move from one to the other. Binary spaces are not considered within this thesis.

2.5. Optimization methods

Optimization methods can be classified according to the **type of problems** they are able to handle: *continuous*, *discrete*, *mixed-discrete*, *binary*, or *combinatorial*; *linear* or *nonlinear*; *convex* or *non-convex*; *unimodal* or *multimodal*; *differentiable* or *non-differentiable*; *smooth* or *non-smooth*; *constrained* or *unconstrained*; *single-objective* or *multiobjective*; *stationary* or *dynamic*; *explicit* or *implicit*; *constraint-satisfaction*; *etc.* Or they can be classified according to the **features of the algorithms**: *global* or *local*; *single-solution* or *multi-solution* (also *niching*); *constrained* or *unconstrained*; *single-objective* or *multiobjective*; *stationary* or *dynamic*; *gradient-based* or *gradient-free*; *deterministic* or *probabilistic* (also *stochastic*); *analytical*, *numerical*, or *heuristic*[2]; *single-based* or *population-based*; *robust*; etc.

[2] *Heuristics* in this context refers to techniques that do not guarantee to find anything, and are usually based on common sense, natural metaphors, or even methods whose behaviour is not fully understood.

- **Global/Local methods**. In practice, no method can guarantee finding a global optimum in non-convex, continuous problems[3]. By *global* method it is meant that it is able to escape poor local optima (e.g. Genetic Algorithms; Particle Swarm Optimization; etc.). Conversely, a *local* method is that which cannot –in a general sense– escape local optima, typically getting stuck in the nearest local optimum when dealing with multimodal problems (e.g. Newton's method; Steepest Descent method; Conjugate Gradient methods; etc.). For some particular cases like convex problems, there are methods such as the Simplex method or Gradient-based methods which may guarantee fining the actual global optimum. The same is true for brute force methods (e.g. exhaustive search) in discrete problems.

- **Multi-solution (niching) methods**. These methods are designed to return a number of optimal solutions rather than only one.

- **Constrained/Unconstrained methods**. *Constrained* methods are able to deal with the optimization of a cost function when only a part of the search-space is feasible, whereas *unconstrained* methods can only handle problems whose whole search-space is feasible. As previously mentioned, a method which is able to deal with side-constrained problems might be referred to as an *unconstrained* method.

- **Single-objective/Multiobjective methods**. *Single-objective* methods can only deal with problems whose formulation involves the optimization of only one scalar cost function. *Multiobjective* methods are designed to handle multiobjective problems, thus returning a *Pareto-optimal set* of solutions rather than a solution vector.

- **Dynamic methods**. These methods are designed to find and track non-stationary optima in dynamic environments.

- **Gradient-based/Gradient-free methods**. *Gradient-based* methods use differential calculus as the main tool throughout the optimization process (e.g. Newton's method; Steepest Descent; Conjugate Gradient methods; Sequential Quadratic Programming; etc.). Hence the problems must typically be smooth, so that gradients and Hessians can be calculated. *Gradient-free* methods usually use the cost function information only, so that there are no requirements with regards to continuity and

[3] However, refer, for instance, to (Cui & Zeng, 2004) for a so-called 'globally convergent' method.

differentiability of the cost and constraint functions (e.g. Simulated Annealing; Genetic Algorithms; Differential Evolution; Particle Swarm Optimization; etc.).

- **Deterministic/Probabilistic (or Stochastic) methods**. *Deterministic* methods are techniques designed based on a precise, deterministic theory. *Probabilistic* methods incorporate some randomness to the optimization process, without turning it into random. Randomness may be incorporated by adding stochastic elements in the problem functions or in the algorithm itself. An example of the latter case is the random weights in the Particle Swarm Optimization algorithm.

- **Analytical/Numerical/Heuristic methods**. Within a limited field of application, *analytical* methods can solve optimization problems in an exact manner (e.g. the Simplex Method). *Numerical* methods usually consist of techniques to approximate the solution to a problem deterministically (e.g. Conjugate Gradient Method). *Heuristic* methods –which are typically also numerical– do not solve the problem deterministically but by means of some procedures not related to the optimization process in an obvious manner (e.g. Particle Swarm Optimization). Both *numerical* and *heuristic* methods find approximate solutions, the former being more precise.

- **Single-based/Population-based methods**. An individual search is carried out in *single-based* methods, so that a single solution per time-step is returned (e.g. Simplex method; Conjugate Gradient methods; Simulated Annealing, Tabu Search; etc.). Conversely, *population-based* methods perform a population of simultaneous, interactive, parallel searches, returning a population of solutions per time-step (e.g. Evolution Strategies; Genetic Algorithms; Particle Swarm Optimization; Ant Colony Optimization; etc.).

- **Robust optimization methods**. These methods are designed to take uncertainties in the input data into account, as well as high sensitivity in some variables. Thus, the *robust* optimal solution is usually not optimal in a traditional sense (i.e. the highest feasible peak; lowest feasible valley; etc.) but the solution is more *robust* to small changes in the values of the solution coordinates.

As can be seen, the classes defined above are not mutually exclusive, so that a given method typically belongs to more than one class. Thus, different classifications can be

proposed by emphasizing different characteristics of the algorithms. A diagram of one possible classification is offered in Fig. 2.1.

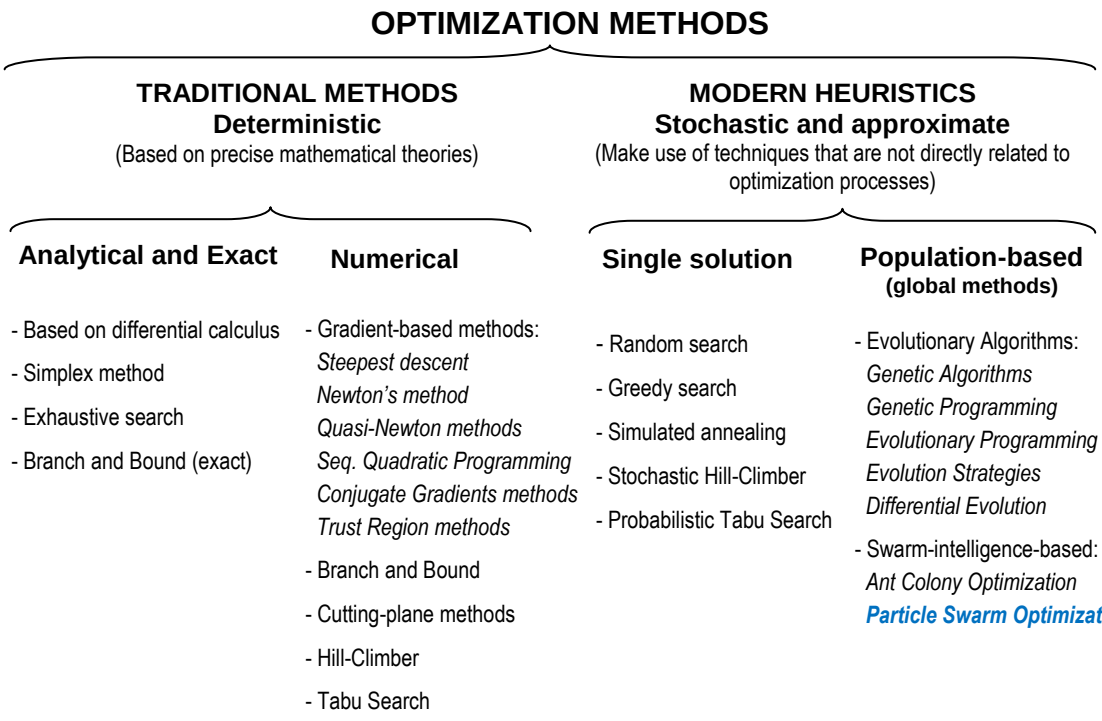


Fig. 2.1. A tentative –among many possible– classification of optimization methods. Note that the Branch and Bound method may be exact or approximate, and that the plain Tabu Search is viewed as traditional rather than heuristics because there are no stochastic elements involved. When stochastic elements are incorporated to methods such as the Hill-Climber and Tabu Search, they are classed as heuristics. A given method usually belongs to more than one class, some of which are not included in this figure (e.g. the simplex method is analytical, exact, convex, global, etc.)

2.6. Constraint-handling

A formal discussion on the classical theory of constrained optimization is beyond the scope of this thesis, as its applicability to the particle swarm optimization (PSO) method is limited. In traditional methods, the constraint-handling technique may be embedded in the algorithm as, for instance, in the *simplex methods* or in some *interior point methods*. This gives birth to families of methods that are specific for constrained problems. Thus, in addition to the necessary conditions for the different traditional unconstrained methods to be applicable –i.e. continuity, differentiability, linearity, convexity, etc.–, there are new conditions for the constraints. That is, the number of constraints, whether they are inequality or equality constraints, linear or nonlinear, continuous, differenti-

able, etc. Other constraint-handling techniques are external to the search engine, such as some *Penalty* and some *Augmented Lagrangian Methods*, which replace the constrained problem by a surrogate, unconstrained one. The constraints are somehow included within the modified objective function. It may be possible to find a local solution by optimizing a single surrogate unconstrained problem, or by optimizing a sequence of unconstrained sub-problems. For a review on the *theory of constrained optimization* as well as traditional constrained optimization algorithms, refer to a standard text on numerical optimization, such as (Nocedal & Wright, 2006).

In Particle Swarm Optimization (PSO), the constraint-handling techniques are external to the method, which is inherently an unconstrained optimization algorithm. The same as for the objective function, no continuity, linearity, or differentiability condition needs to be met by the constraint functions. In addition, dealing with inequality constraints is rather straightforward while it represents one of the main challenges in traditional nonlinear constrained optimization. Conversely, it is harder for the PSO algorithm to cope with equality constraints, and setting a tolerance for their violations is a must. Different families of constraint-handling techniques that can be coupled with the PSO method to cope with constrained problems are briefly discussed in chapter 4.

2.7. No free lunch theorems for optimization

First of all, it must be noted that a formal study and analysis of the *No Free Lunch Theorems* (NFL) for optimization and its implications –including whether or when the theorems hold– are beyond the scope of this thesis. However, given the considerable impact of the first theorem on optimization research –especially on Evolutionary Computation and the development of general-purpose optimizers–, their formulation must be acknowledged and referenced. From here forth, the *no free lunch theorem* will refer to the first theorem in (Wolpert & Macready, 1997), as the second theorem applies to time-dependent cost functions, which are not considered in this thesis.

In short, the *no free lunch theorem* for optimization suggests that, if an algorithm exhibits better performance on a set of problems, it would pay the price of inferior performance on others. So far, the statement is almost self-evident and leaves little room for

disagreement or controversy, but the theorem goes further by asseverating that the average performances of different algorithms on all possible problems are exactly the same. While the theorem is restricted to discrete functions, it is specifically stated that it holds for real-valued problems that are represented and dealt with on digital computers. Refer to (Wolpert & Macready, 1995) and (Wolpert & Macready, 1997) for an in-depth review of the theorem and its demonstrations and implications.

One might expect that there are pairs of search algorithms A and B such that A performs better than B on average, even if B sometimes outperforms A. As an example, one might expect that hill climbing usually outperforms hill descending if one's goal is to find a maximum of the cost function. One might also expect it would outperform a random search in such a context.

(...) *such expectations are incorrect.* (...) *Roughly speaking, we show that* (...) *the average performance of any pair of algorithms across all possible problems is identical. This means in particular that if some algorithm* a_1*'s performance is superior to that of another algorithm* a_2 *over some set of optimization problems, then the reverse must be true over the set of all other optimization problems.* (Wolpert & Macready, 1997)

Mathematically, the (first) *no free lunch theorem* states:

$$\sum_f P\left(d_m^y \mid f, m, a_1\right) = \sum_f P\left(d_m^y \mid f, m, a_2\right) \tag{2.8}$$

where a_i stands for *algorithm i*; m is the number of iterations; f is the function at issue; $P(\cdot)$ stands for the probability; and d_m^y is a measure of performance after m iterations.

It follows that, to improve the average performance, knowledge on the problem and on the optimization method should be taken into account to match problems to solvers.

Perhaps one of the major controversies of this theorem is in what it is meant by *all problems* (*f*). It has been argued, for instance, that it is not all but only practical problems that researchers are interested in, whereas it has been replied that the problems that are of no interest today may be in the future, and so on. Notice that while Wolpert & Macready (1997) specifically include continuous problems that become discrete due to their representation in digital computers, it could be argued back that not all problems can be represented in a computer. In addition, if a general-purpose *digital* algorithm for continuous problems is being sought, the set of all problems is reduced to the set of all

digitalized continuous problems, and the theorem would not be proven to hold. However, this thesis does not concern itself with the implications of the theorem, including whether and/or when it holds.

The argument here is simply that continuous optimization problems do in fact make use of information regarding the problem at hand. To start with, since they are not discrete in nature but only due to their digitalization, it can be expected that if a good solution is found, it is very likely that there is a better one nearby, either following the gradient or the direction towards a better known solution with a sufficiently small step. This would not be a valid *informed guess* in discrete combinatorial problems. But the most important aspect in computational intelligence would be that the aim of the field is precisely to use information regarding the problem at hand, while ideally such information would be gathered and extracted by the algorithm itself.

Let us disregard for the time being the ideal computational-intelligent algorithm, which would self-acquire and exploit information specific to the problem. For more traditional, non-adaptive algorithms, the more they take advantage of a priori knowledge of the features of the problem at hand –i.e. linearity, convexity, differentiability, and so on– the less suitable they are for other problems. In other words, the more problem-specific they become. For instance, a gradient-descent algorithm can tackle a linear problem, but undoubtedly in a less efficient manner than the *simplex method*. Thus, it is argued that the *steepest descent* method is more general-purpose than the *simplex method* because it can tackle both linear and convex-nonlinear problems to some reasonable degree of accuracy. Following the same train of thought, PSO is more general-purpose than both.

Hence, an optimizer is considered *general-purpose* here if it does perform reasonably well –within tolerance– on a wide-range of different problems at a reasonable amount of time. Even if –due to the *NFL* compensation– another solver performs notably better in some problems and notably worse in some others, a general-purpose algorithm would return acceptable solutions on the whole range of validity of its '*general-purposeness*'. In practice, it is frequently the case that relying on some extra computational cost is cheaper and faster than hiring an expert to develop a specific, more efficient solution. Expert, knowledgeable manpower is commonly more expensive than additional computational equipment.

2.8. Closure

A brief overview and discussion on the stages of a full optimization problem analysis was offered, focusing on the solution and validation of the solver while assuming an accurate model is provided. The definition of the different types of constraints was presented, as well as a general review of the main classes of optimization problems.

The type of problems that this thesis is concerned with was discussed, and their initial formulation and convenient re-formulation to be handled by a particle swarm optimizer was proposed. The nomenclature and definition of the main elements involved in the formulation were introduced, a discussion of the families of optimization methods was presented, and a possible classification was offered in Fig. 2.1.

The topic of constraint-handling techniques was only marginally addressed, and it will be discussed further in chapter 4, focusing on those techniques that are relevant to particle swarm optimizers.

Finally, the *no free lunch* theorem was merely acknowledged and just a few relevant comments were made. Since the formal study of this theorem is beyond the scope of this work, the reader is simply referred to the appropriate work for further information.

The canonical particle swarm optimizer is suitable for real-valued variables and unconstrained problems. While discrete or mixed-integer problems can be handled to some extent with some modifications to the basic algorithm –or by its binary version–, constraint-handling techniques need to be incorporated to deal with constrained problems. Multi-objective, dynamic, and multi-solution problems could also be coped with by appropriately modifying the basic algorithm. The *particle swarm optimizer* developed in this thesis is a global, single-solution, constrained and unconstrained, single-objective, stationary (i.e. non-dynamic), gradient-free, probabilistic, heuristic, population-based method, which is able to handle continuous –exceptionally, discrete variables by rounding-off–, constrained and unconstrained, single-objective, stationary problems, regardless of whether the functions involved are or are not linear, convex, unimodal, differentiable, smooth, continuous, or even explicit.

Chapter 3

EVOLUTIONARY ALGORITHMS

A brief review of the Evolutionary Computation field is offered. Basic relevant concepts of natural evolution are presented together with its interpretation as an optimization process. The origins and the history of the development of the initially three mainstream Evolutionary Algorithms is presented –namely Genetic Algorithms, Evolution Strategies and Evolutionary Programming–, and the creation of the Evolutionary Computation field in order to encompass them all is discussed. Some of the posterior paradigms inspired on and derived from them are also mentioned. The main common features and the general procedure for all Evolutionary Algorithms are outlined, followed by a concise description of some of the main current paradigms: Genetic Algorithms; Genetic Programming; Evolution Strategies; Evolutionary Programming; and Differential Evolution. Some minor comments are interlaced in the closure so as to present the link between this about half-century old field and the relatively new Particle Swarm Optimization paradigm.

3.1. Introduction

Evolution is a natural process that organisms undergo to adapt their behaviour in order to survive within a dynamic, competitive environment. Since adaptation is carried out by seeking the best response to the challenges they face, the process of adaptation may be described as the *optimization* of their performance. In the Evolutionary Algorithms' jargon, this is referred to as the process of *fitness maximization*.

Biological evolution takes place at different scales in nature (e.g. chromosomes; cells; individuals; species; populations of species, etc.). Scientists observed that the kinds of problems that these biological organisms are able to cope with are, when posed as optimization problems, the same kinds of problems that traditional algorithms are typically unable to handle (see (Fogel D. B., 2008)). That is, for instance, problems exhibiting nonlinearities; discontinuities; uncertainties; etc. Hence, optimal biological structures observed in nature led scientists to the idea of mimicking biological evolution mechanisms to solve optimization problems. A family of population-based optimization methods inspired by natural evolution is referred to as Evolutionary Algorithms (EAs). They exhibit some intelligent behaviour, which originates from mimicking natural processes that biological organisms undergo. EAs are also used as models of evolution by evolu-

tionary biologists as well as by researchers in Artificial Life (AL) aiming to experiment on new artificial evolutionary worlds. AL is a branch of Artificial Intelligence (AI), which deals with the development of *artificial beings* exhibiting some degree of aliveness in addition to some degree of intelligence.

EAs are like the big brother for the particle swarm optimization (PSO) paradigm. They share some major common features, despite being inspired by different metaphors. Namely, both EAs and PSO evolve a population of individuals which profit from previously acquired knowledge, while making use of stochastic operators to introduce new responses. Therefore, the field of Evolutionary Computation (EC) is introduced, and a few of the most popular paradigms –especially those relevant to optimization problems– are loosely overviewed within this chapter.

3.2. Natural evolution

Keeping in mind that this topic is far beyond the field of expertise of the author of this thesis, a brief review of some basic natural evolution concepts is offered hereafter.

Darwin (1859) claimed that all organisms descend from a common ancestor, and that the adaptive changes of species occur by means of apparently random mutations, where smaller mutations are observed more often than larger ones. If mutations are beneficial, they are preserved by a natural selection mechanism that relies on the ability of some individuals to outlast others, thus increasing the probability of passing their genetic information to the next generation.

Evolution is driven by both reproduction and the challenges posed by the environment. For instance, a population tends to increase in size without environmental restriction until resources become insufficient. Then, the *survival of the fittest* selection mechanism favours the fittest species with higher probability of survival and hence of reproduction. An individual is *fitter* if it is better adapted to the current state of the environment.

Darwin (1859) developed his theory of evolution in a macroscopic fashion, without any knowledge on genetics. The *neo-Darwinian theory of evolution* views living organisms as a duality of *genetic information* present in every cell of an individual, and *external observable features* (behaviour, physiology, morphology, etc.). The *genetic information*

is encoded in a macromolecule known as *deoxyribonucleic acid* (DNA), where the *gene* is the unit of heredity. The aggregation of all genetic information in an organism is referred to as its *genotype*, whereas all its observable features are referred to as its *phenotype*. The theory thinks of *natural selection* as alterations in the frequency of genes in a population. A new field called the *modern synthesis of genetics and evolution* also takes into account other mechanisms of evolution in addition to *natural selection* such as the *genetic drift*, which is beyond the scope of this review.

Thus, the unit of selection is the individual, who possesses a certain genotype that is independent from the environment and from other individuals. The aggregation of all genotypes in the population is known as the *genetic pool*. The interaction between the genotype of an individual and the environment gives shape to its phenotype. Finally, the unit undergoing evolution is a population of individuals. Since evolution takes place during reproduction, the genotype of an individual cannot change except for some infrequent small mutations and some kinds of recombination in unicellular organisms. However, the phenotype might change due to the interaction with the environment.

As quoted by Fogel (1995), Lewontin schematized the evolution process as a *genotypic state space*, a *phenotypic state space* and *four mapping functions*, as shown in Fig. 3.1:

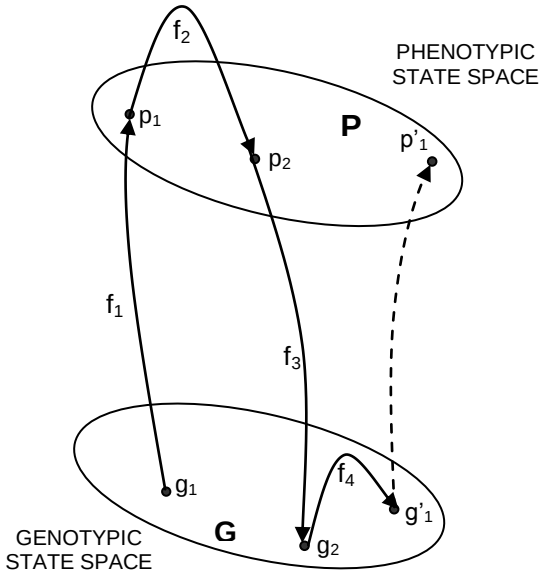


Fig. 3.1. Schematic evolution process of a population within a single generational step. It can be viewed as a succession of four mapping functions (epigenesis; selection; genotypic survival; and alteration) relating the genotypic information state space and the phenotypic observable state space (from (Innocente, 2006), after (Fogel D. B., 1995)).

- **f_1** (*epigenesis* G ⇨ P): It maps the population encoded as g_1 in the genotypic space G to the phenotypic space P as a particular set of traits represented by the set of phenotypes p_1 (observable features, which depend on the genetic code and on the environment). The mapping G ⇨ P is *pleiotropic*, which means that a given genic change may affect more than one phenotypic trait.

- **f_2** (*selection* P ⇨ P): Since natural selection works on phenotypes, **f_2** maps the sets of phenotypes p_1 to p_2. That is, it maps the phenotypes of all individuals to the phenotypes of the selected ones. Thus, the process of natural selection is performed without any knowledge of the information encoded in g_1.

- **f_3** (*genotypic survival* P ⇨ G): It maps the selected set of phenotypes p_2 to the genotypic space G, thus encoding back the phenotypes of the selected individuals as g_2. The mapping P ⇨ G is *polygenic*, which means that the modification of a phenotypic trait may be due to the alterations of several genes.

- **f_4** (*alteration* G ⇨ G): It maps the genotypes g_2 into g'_1, comprising all the genetic changes of the process of evolution corresponding to the current generation. Hence, function **f_4** contains all the rules for the genetic changes. Once the new population g'_1 is encoded in G, the single generation is complete.

Although there are still aspects and mechanisms in the theory of evolution that are not fully understood or unanimously agreed upon, there is –to the best of my knowledge– general consensus on that:

- There must be competition for limited resources between individuals in one or more populations. Populations tend to increase in size while resources are plentiful, so that there is no competition and hence no evolutionary pressure.
- There must be dynamic populations where individuals live, reproduce, and die.
- Evolution occurs during reproduction by means of recombination and mutation.
- Recombination is a process of creating chromosomes by combining genetic material from two or more parents, so that the children's chromosomes are different from those of their parents, although there must be some resemblance.

- Evolution takes place in the chromosomes that encode living beings; hence it takes place in the genotypic state space.
- Natural selection is the mechanism by which the genes within the chromosomes that encode more successful living beings are more likely to survive, since such beings live longer and therefore tend to reproduce more.
- Since the success of individuals depend on their performance in the environment, natural selection takes place in the phenotypic state space.

Refer to (Innocente, 2006) for a brief overview of genetics (appendix 2) and natural evolution (chapter 4). For further reading, also refer to (Davis & Mitchell, 1991), (Fogel D. B., 1995), (Bäck, 1996), (Haupt & Haupt, 2004, pp. 19-22), and (De Jong, 2006).

3.3. Evolutionary algorithms

EAs are all those methods inspired on natural metaphors, which rely on the concepts of *natural selection* and *survival of the fittest*. Although they are most widely used as an optimization tool, they are often claimed to perform adaptation rather than optimization. Since this thesis is concerned with optimization problems, EAs are viewed hereafter as problem solvers rather than models of evolution or artificial evolution implementations.

Considering that the mapping (genotype ⇨ phenotype) is *pleiotropic* and the mapping (phenotype ⇨ genotype) is *polygenic*, the formulation and implementation of such mappings may be very complex. Thus, while selection occurs within the phenotypic space and genetic alterations do within the genotypic space, different paradigms within the EAs family focus on either one space or the other. Loosely speaking, there are two main classes of EAs: the *genetic-based* approach, and the *phenotypic-based* approach.

3.3.1. Origins and summarized history

De Jong (2006) argues that the origins of evolutionary computation can be traced back to some influential ideas of Wright (1932). In1950s, biologists attempted to develop computer simulations of natural genetic systems, even though they apparently did not foresee the possibilities of applying their methodologies to solve optimization problems.

Another early application such as evolving computer programs −i.e. a program that writes programs− can be traced back to (Friedberg, 1959). Three different methods inspired by biological evolution were developed independently, pursuing different objectives: *Genetic Algorithms*; *Evolutionary Programming*; and *Evolution Strategies*.

Genetic Algorithms (GAs)

It is unanimously accepted that the father of *Genetic Algorithms* (GAs) is John Holland, who studied adaptation in nature and its implementation and simulation in computer systems (refer to Holland (1962), (1967), (1975)). Therefore, his original goal was not at problem-solving, which is its main application nowadays. Despite the remarkable simplifications with respect to biological evolution, the method works by performing fitness-based selection on the phenotypes, and genetic modifications in the genotypes. Therefore, it works on the encoding of the variables rather than on the variables themselves, thus comprising a *genetic-based* approach. The genotype was originally encoded in binary code, which presented the problem that small changes in the genotype could lead to great leaps in the phenotypes. Thus, a later improvement to sort out this problem led to the Gray code to represent genotypes. Another problem was that of the precision when dealing with real-valued problems, which resulted in the development of real-coded GAs (refer to section 3.3.3).

Evolution Strategies (ESs)

Unlike GAs, *Evolution Strategies* (ESs) do not intend to model genetic mechanisms but to carry out the evolutionary processes within the *phenotypic* space. Therefore it is a *phenotypic-based* approach, where the link between parents and children rests in the observable features. The paradigm was first introduced by Ingo Rechenberg (1965), (1973) as a means to deal with experimental hydrodynamic optimization problems with real-valued variables. Later, Hans-Paul Schwefel (1975) developed the paradigm further in his Ph.D. thesis. Originally, a population of just one individual was considered, and exploration relied on a mutation operator only. It soon became a family of population-based optimization methods, which nowadays share many features with GAs including the use of recombination in some strategies. For a short overview of the method, refer to (Bäck, Hoffmeister, & Schwefel, 1991).

Evolutionary Programming (EP)

In the same fashion as ESs, *Evolutionary Programming* (EP) do not intend to model genetic mechanisms. Instead, it is a *phenotypic-based* approach in the sense that the descendants are different from –but resemble– their parents in the observable features rather than in the genetic code. That is to say, all modifications are carried out within the phenotypic space. The origins of EP are linked to the attempt to create AI through the simulation of evolutionary processes, where individuals were thought of as finite state machines (FSMs) undergoing evolution. The original work was undertaken by Fogel, Owens, & Walsh (1966), and the aim was at performing predictions. Thus, while traditional AI paradigms concerned themselves with mimicking either human behaviour or the structure of the human brain, Fogel, Owens, & Walsh (1966) viewed evolution as a process of evolving increasingly intelligent organisms, which were represented by FSMs (refer to section 3.3.6). The method was reinvented for real-valued optimization problems by David B. Fogel (1992) in his Ph.D. thesis. The new paradigm resembles the ESs in aspects such as the individuals' representation and the design of the mutation operator. One of the differences is that there is no recombination operator, but this could be viewed as a strategy within the ESs family (refer to section 3.3.6). David B. Fogel (1995) also ventured a useful definition of intelligence as the ability of a (biological or artificial) being to adapt its behaviour to meet its goals in a range of environments, discussing how to achieve this by simulating evolution (Bäck & Schwefel, 1996).

Evolutionary Computation (EC)

In the 1990s, different groups of scientists concerned with these mainstream evolution-based methods started gathering together to share ideas and insights. Thus, the broader field of Evolutionary Computation (EC) was born to encompass all these families of algorithms. Therefore, the general name adopted to refer to any method somehow inspired by biological evolution is now *Evolutionary Algorithm* (EA). From then forth, crossbreeding of ideas and development of hybrids and new paradigms has been of common practice. Examples of successful EAs developed in the 1990s within the EC framework are *Genetic Programming* (GP) and *Differential Evolution* (DE). There are also other well-established paradigms whose development was strongly influenced by the EAs, even though they are not based on evolution per se. Well known examples are

Memetic Algorithms; *Cultural Algorithms*; *Artificial Immune Systems*; *Ant Colony Optimization*; *Particle Swarm Optimization*; etc. Most of these methods will not be discussed further in this thesis due to space and time constraints.

Genetic Programming (GP)

Friedberg (1959)'s earlier work and the GAs approach inspired Koza (1990) to develop the *Genetic Programming* (GP) paradigm. The main objective was to design programs that would evolve themselves. In other words, the method is a procedure for the automatic generation of computer programs. Therefore, the goal is to solve the problem of solving problems by designing programs that write programs.

Differential Evolution (DE)

Inspired on the ESs paradigm, a new increasingly popular method called *Differential Evolution* (DE) was proposed by Storn and Price (1995), (1996) to handle real-valued problems. Unlike ESs, there is a *differential mutation* operator instead of performing the mutation from a predefined probability distribution. Hence the scheme is claimed to be completely self-organizing. However, the mutation is performed so that the mutated individual (*mutant vector*) is unrelated to the individual being mutated (*target vector*). Therefore, the name mutation for that operator seems arguable. Later, different mutation strategies were proposed, some of which do consider the *target vector* (refer to section 3.3.7). Originally, each *target vector* would undergo a fitness-based competition for survival with its corresponding *mutant vector*. That is, there would be as many pairs of vectors competing for survival at each generation as individuals in the population. Later, a supposed crossover operator was introduced, which consisted of a probability threshold that needs to be passed component-wise for that component of the so-called *trial vector* to take the value of the corresponding component of the *mutant vector* (also *donor vector*). The crossover operator resembles a discrete crossover between the *target vector* and the *mutant vector* rather than between two different vectors in the population. It appears that the DE method is highly sensitive to the scheme chosen for the mutation process, to the mutation weight (F), and to the crossover rate (CR). Nevertheless, it is one of the stochastic optimization methods for real-valued problems which is becoming

more popular due to the good performance reported. Refer to section 3.3.7 for a more detailed description of the paradigm.

Particle Swarm Optimization (PSO)

The *Particle Swarm Optimization* (PSO) method was invented by Kennedy & Eberhart (1995), inspired by earlier bird flock simulations framed within the field of social psychology. Hence, the origin of the paradigm is more or less contemporary to that of DE. The method is included here in order to highlight its links and similarities to EAs, even though it does not mimic natural evolution processes. Nonetheless, the method was invented in the 1990s while all the different paradigms inspired by biological evolution were being encompassed by the newly created EC field, and new paradigms and hybrids were being developed. It seems clear that the EAs played a critical role in the conception of the PSO paradigm. In the same fashion as all EAs, PSO is a population-based, stochastic, global optimization method whose ability to optimize is an emergent property. Since the remainder of the thesis, from Chapter 4 forth, is devoted to the PSO method, the latter will not be discussed any further within this chapter.

For further details on the history of the evolution of the *Evolutionary Computation* field, refer to (Fogel D. , 1998) and (De Jong, 2006, pp. 23-31).

3.3.2. General Evolutionary Algorithm

A high level flow chart for a general Evolutionary Algorithm is offered in Fig. 3.2. The shadowed boxes are the stages where evolution actually takes place.

3.3.2.1. Initialization

Just like any other population-based method, nearly all EAs start with the initialization of a population of candidate solutions. The simplest procedure consists of a purely random initialization. Any other Design of Experiments (DoE)–based procedure would do the job. If some prior knowledge on the environment is available, it should always be exploited. A data structure must be defined to represent candidate solutions in a computer, for which no best choice can be generalized for all problems and for all EAs.

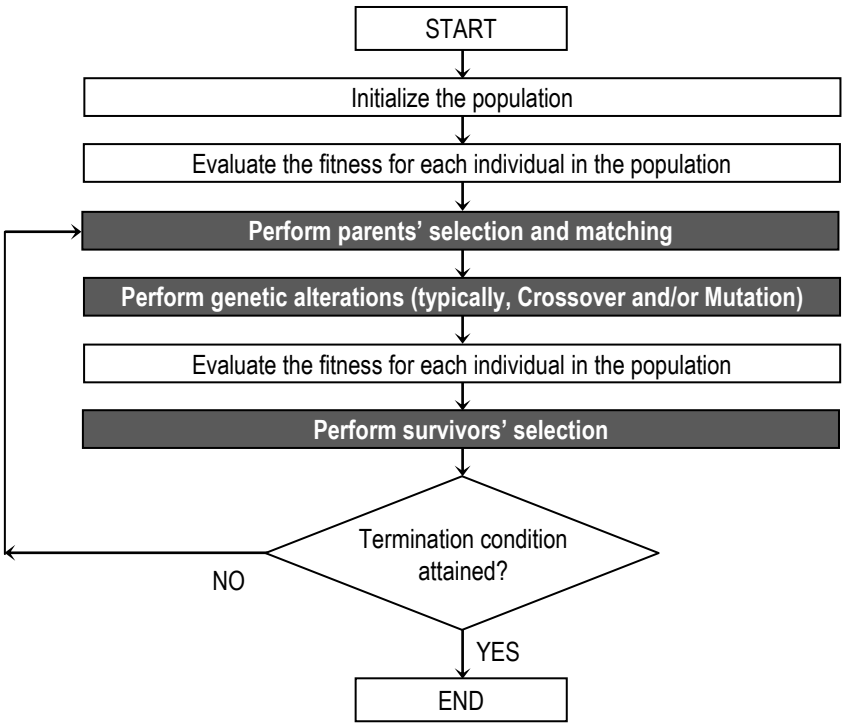


Fig. 3.2. Flow chart for a general Evolutionary Algorithm. The shadowed boxes are the stages where evolution takes place. Some EAs present only one selection stage, so that the other is as if all individuals were selected.

3.3.2.2. Fitness evaluation

Next, the performance of each individual in the population must be computed somehow. The simplest case consists of a cost function in a well-posed optimization problem. In the EAs jargon, the function to be maximized is referred to as the *fitness function* due to the metaphor that inspired these methods. Even though it is not strictly correct, it is not uncommon to find in the literature EAs aimed at minimizing a so-called *fitness function*. Technically, the *fitness function* should always be maximized. Thus, for minimization problems, the *fitness function* should be computed, for instance, as in Eq. (3.1):

$$fitness(\mathbf{x}) = -f(\mathbf{x}) \tag{3.1}$$

Alternatively, if the problem does not allow for an explicit *fitness function*, some means to evaluate the individuals' performances must be available (e.g. neural network; lab experiments; numerical model of the problem being optimized; etc.). Ultimately, since these methods do not rely on differential calculus, all that is needed is a means to tell

whether one candidate solution is better than another. That is, the fitness assigned to each individual allows for the comparative evaluation of their performances.

3.3.2.3. Selection

Two types of selection stages can be defined in EAs. The first one is called the *parents' selection* stage, which selects the individuals that are to undergo genetic alterations. The other is the *survivors' selection* stage, where the individuals that are to survive for the next generation are selected. The latter takes place after all genetic alterations. At least one of the selection stages must be fitness-based (i.e. survival of the fittest). Some EAs present one, some the other, and some both stages. Note that individuals are not modified during any selection stage, so that the search-space is not further explored.

Parents' selection and matching

Parents' selection

According to the paradigm, either all individuals or only those who are probabilistically fitter within the current population are selected for reproduction. Different paradigms, and even different versions of the same paradigm, perform this stage differently. In GAs, the fittest individuals within the population are probabilistically selected. Each individual is allowed to be selected more than once, and all selected individuals are guaranteed to breed. In EP, ESs and DE, the whole population (μ individuals) is selected at this stage, where all individuals are allowed, but not necessarily guaranteed, to breed.

Parents' matching

The number of parents per child, ρ, is set *a priori*. Once the individuals that are allowed to become parents (μ) are selected, ρ out of the μ individuals are randomly chosen to mate and thus produce a child. In canonical GAs, there are two parents per reproduction –which can breed either one or two children–, and the *pairing* (i.e. *matching*) is random. In other EAs such as ESs, ρ parents are successively selected out of the μ individuals at random, who breed one child. All individuals are typically as likely to be selected. The selection is λ–fold, hence λ mating groups composed of ρ parents are generated, which will breed one descendant each per recombination.

Survivors' selection

This stage takes place after all genetic alterations. In canonical GAs, every descendant is selected, whereas only the fittest individuals are deterministically selected in EP, ESs and DE. Notice that compliance with constraint functions is embedded in the concept of fitness when applicable. In the simplest EAs, selection is performed so that the size of the population is kept constant. In algorithms like canonical GAs, which replace the whole population at every generation, an additional survivors' selection strategy called *elitism* can be incorporated to avoid the loss of good solutions during reproduction. *Elitism* consists of ensuring that the fittest individual at every generation is awarded a 100% probability of survival. The strategy may be extended to any number of fittest individuals, so that only a fraction of the population is replaced at each generation.

Popular selection procedures

Roulette-wheel selection

This is one of the most popular procedures for the parents' selection in GAs. It consists of a probabilistic proportional fitness-based selection, where the number of times that an individual is selected is probabilistically proportional to its current fitness with respect to the aggregated fitness of the whole population. The procedure is as follows:

1. The probability that each individual has of being selected is calculated.
2. Individuals are listed, and their cumulative probabilities are computed. Hence the cumulative probability of the first individual equals its actual probability of being selected, whereas the cumulative probability of the last individual equals '1'.
3. A random number between '0' and '1' is generated from a uniform distribution, and the first individual whose cumulative probability is higher than the random number generated is selected.

The cumulative probabilities can be thought of as a wheel, the probability that each individual has of being selected as a portion of it, and the random number generated as a spin of the wheel. Thus, each spin selects one individual. For different reasons, it might happen that several or all individuals present similar fitness values. In cases like the latter, the plain roulette wheel selection does not work properly because individuals are

more or less equally likely to be selected, so that improvement of the population's average fitness becomes more difficult. The procedure can be modified so that the selection of the fittest individuals is guaranteed. In this case the wheel does not have a single pointer but as many pointers as individuals are to be selected, equally separated. Thus, the complete selection is performed in a single spin of the wheel.

Ranked-based roulette wheel selection

Individuals are ranked according to their fitness, from worst to best. The probability that each individual has of being selected is computed as its position in the rank divided by the summation of all ranks. Thus, the probabilities of selection are still fitness-based, but not proportional to the fitness.

Tournament selection

This is a probabilistic fitness-based selection scheme. Since all individuals are selected in the parents' selection stage and the survivors' selection is deterministic in ESs, EP and DE, this selection scheme is mostly used for the parents' selection in GAs. The procedure is as follows:

1. Iteratively, two individuals are selected at random from the whole population.
2. Their performance is compared in term of their fitness. Note that the constraint violations are assumed to be considered in their fitness values.
3. The fitter individual survives.
4. Since only the winner of the tournament is selected, the procedure is repeated as many times as individuals are to be selected.

This procedure can be generalized to any number of competitors per tournament. In this case, the fitness of each individual is compared to those of all the other competitors, and the number of times each individual defeats the others in the tournament is computed. The winner of the tournament is the individual with the most winnings.

Other selection schemes

Several other selection schemes can be found in the literature, such as the *Boltzmann selection*; *Disruptive selection*; *non-fitness-based ranking selection*; etc. A complete

survey of the different selection schemes in EAs is beyond the scope of this thesis. For further reading, refer to (Herrera, Lozano, & Verdegay, 1998); (Bäck, Fogel, & Michalewicz, 2000) (papers 22 to 27); (Gen & Cheng, 2000); (De Jong, 2006); and (Alves Da Silva & Falcão, 2008); among others.

3.3.2.4. Genetic alterations

Genetic operators alter the individuals so as to explore new regions in the search-space. There are two main families: *mutation* and *crossover* (also called *recombination*). While *crossover* is just one type of *recombination* in natural biology, the terms are used indistinctly within this thesis because no other type of recombination is used in EAs. Nevertheless, other types of genetic operators may be thought of, and many variations within the existing types may –and have been– developed.

Mutation operator

In some EAs, mutation is applied to every individual in the population. In others, it is only applied to the selected parents, and with some probability called the *mutation rate* (MR). In any case, once an individual is selected to undergo mutation, the operator performs some small perturbations to some or all of its coordinates. Usually, such perturbations are obtained from some probability distributions –typically *uniform* or *normal*–, and they are independent from one another for each gene. There are exceptions such as the correlated mutations in ESs and EP or the differential mutation in DE, which is not obtained from a probability distribution (refer to section 3.3.7). Not only does mutation allow exploration but it also adds new information to the genetic pool. It mimics nature in the sense that smaller changes occur more often than greater ones, and hence can be viewed as an individual's random walk in the vicinity of its current location (Sastry, Goldberg, & Kendall, 2005). However, small changes made by mutation in the genotypes of *genetic-based* algorithms may take any size in the phenotype. Appropriate corrections need to be made to sort out this problem. The role of mutation in EAs such as canonical ESs and EP is to drive the search. In contrast, it is a background operator in paradigms like GAs and GP, which is performed with some low MR aiming to explore lost and unexplored regions of the search-space. *It ensures that the probability of reaching any point in the search-space is never zero* (Herrera, Lozano, & Verdegay, 1998).

Recombination (crossover) operator

The crossover operator creates new individuals by combining parts from two or more other individuals. It comprises a means of sharing information between different chromosomes. Thus, it allows exploration without adding genetic information that was not included in the previous population. The hope is that the combination of two good individuals results in a better one. Crossover is not commonly applied to all selected parents in the mating pool, as a probability threshold called the *crossover rate* (CR) must be passed. Many different crossover types can be found in the literature, which depend greatly on the representation of the individual adopted. Some paradigms like EP do not perform crossover, and therefore the search is driven by mutation only. It is sometimes claimed that this is because an individual in EP stands for a species rather than an individual, and different species do not mate. Other paradigms like DE may perform a misleadingly called crossover, but in fact it is another mechanism as no recombination between different individuals in the population in the same generation is performed.

For an overview of different *individuals' representation*; *selection mechanisms*; and *genetic operators* for the general EA, refer to (De Jong, 2006, pp. 115-209).

3.3.2.5. Termination conditions

Different conditions and combinations of conditions can be thought of for the search to be terminated. Those associated to the convergence of the search in terms of the location of the solution are not as straightforward as in point-to-point searches, since the solutions found in consecutive time-steps may belong to different individuals in the population. The most immediate termination conditions used in EAs involve the maximum computational cost permitted for the search, which is commonly measured in terms of a maximum number of fitness function evaluations. Other conditions to terminate the search may involve a maximum number of generations; a lower limit for a measure of diversity in the population both in terms of the fitness function and of the coordinates of the individuals (further improvement is unlikely); the objective fulfilled (e.g. a feasible solution found in a constraint-satisfaction problem; or finding a zero-error in an error-minimizing problem); etc.

3.3.3. Genetic Algorithms

(…) *with GAs we are not optimizing; we are creating conditions in which optimization occurs, as it may have occurred in the natural world*… (Davis & Mitchell, 1991)

3.3.3.1. Representation, initialization and fitness evaluation

The individuals' representation is a key issue in GAs because they directly (...) *manipulate a coded representation of the problem and because the representation schema can severely limit the window by which a system observes its world*. (Koza, 1992)

The original, canonical GA presents a fixed-length binary representation of individuals; generational replacement (i.e. individuals' life-span limited to a generation); one-point crossover with high crossover rate (CR); mutation with low mutation rate (MR); and probabilistic proportional parents' selection scheme with random pairing (matching). Examples of two individuals encoded in binary strings are offered in Fig. 3.3.

1	1	0	1	0	0	0	1	1	0	1	0

1	1	0	1	0	0	0	1	1	0	1	1

Fig. 3.3. Example of binary representation of two individuals for a problem with two variables. Since each variable is represented by six binary bits, only $2^6 = 64$ discrete values can be generated for each variable. For a real-valued problem, the first and last values among the 64 possible are the lower and upper limits of the search-space, and linear interpolation is typically used between the uniformly distributed discrete values.

The binary string representation stands for a highly simplified genotype composed of a single chromosome. Since GAs is a genetic-based approach, the genetic operations are performed on this representation while selection is performed on the phenotypes. The individual's binary string is decoded to a real-valued representation, which stands for its phenotype. Its performance is evaluated in terms of the fitness function, which is to be considered in the *survival of the fittest* scheme. The binary search is carried out within a binary hypercube, where each vertex represents one possible solution. The hypercube and its mappings to a discrete ($\mathbb{Z}^n$) and a continuous ($\mathbb{R}^n$) space are shown in Eq. (3.2).

$$I : \{0,1\}^{ng} \rightarrow \mathbb{S} \subseteq \mathbb{Z}^n \qquad \text{or} \qquad I : \{0,1\}^{ng} \rightarrow \mathbb{S} \subseteq \mathbb{R}^n \tag{3.2}$$

where *ng* is the dimension of the binary hypercube –i.e. the number of genes in the chromosome–, and *n* is the dimension of the (phenotypic) search-space. Of course, no

mapping is necessary for binary problems. The fitness function is typically defined as shown in Eq. (3.3) for discrete and real-valued problems.

$$f:\{0,1\}^{ng} \rightarrow \mathbb{S} \subseteq \mathbb{Z}^n \rightarrow \mathbb{R} \qquad \text{or} \qquad f:\{0,1\}^{ng} \rightarrow \mathbb{S} \subseteq \mathbb{R}^n \rightarrow \mathbb{R} \tag{3.3}$$

One important problem for the binary encoding is that the flip of a single bit in the string typically results in important variations in the phenotype. For instance, the two individuals represented in Fig. 3.3 differ only in the last gene corresponding to the second variable. In this case, they also happen to be close in the phenotypic space, as they are consecutive integers once decoded. However, there is also only one bit difference between the two individuals in Fig. 3.4, while they are far away in the phenotypic space after decoding. This is called a *Hamming cliff*.

1	1	0	1	0	0	0	1	1	0	1	0

0	1	0	1	0	0	0	1	1	0	1	0

Fig. 3.4. Example of binary representation of two individuals for a problem with two variables, where a single bit difference in the genotype leads to distant positions in the phenotype.

One solution is the use of the *Gray code* (also *reflected binary code*), which is such that two consecutive numbers in the phenotype always differ in one bit in the genotype. That is, they are separated by a Hamming distance equal to '1' in the Gray binary hypercube. Refer to Fig. 3.5 for a small comparative example of plain and Gray binary codes.

Base 10	Binary code	Gray code
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101

Fig. 3.5. Numerical, comparative example of base 10, binary, and Gray codes. As can be seen, consecutive numbers in base 10 always differ in only one bit in Gray code (Hamming distance equal to one).

It is clear that the GA paradigm is discrete in nature, and hence works very well on discrete problems, where the search consist of seeking the best combination of discrete values or the best feasible permutation. In spite of the fact that it can be adapted well to cope with real-valued problems, it has been widely proved that *real-number* encodings

work better for real-valued and/or constrained optimization problems (Gen & Cheng, 2000). Besides, real-valued chromosomes for problems with real-valued variables allow the use of larger domains without sacrificing precision –thus obtaining the solution to the full machine-precision– as well as the fine-tuning of the search. They are also faster, as they do not need to successively encode and decode solutions. For a summary of the advantages of real-coded GAs with respect to the binary versions, refer to (Herrera, Lozano, & Verdegay, 1998, pp. 285-287). For real-valued encoding, individuals are represented by *n*-dimensional floating point vectors as in Eq. (3.4), where each gene represents an object variable. In this case, the *genotype* and the *phenotype* coincide. Other problem-dependent data structure to represent individuals may be, for instance, *vectors of integers*, *order lists*, or the *syntax trees* used in GP (see section 3.3.4).

$$\mathbf{x}_i^g = \left(x_{i,1}^g \quad \ldots \quad x_{i,j}^g \quad \ldots \quad x_{i,n}^g \right) \tag{3.4}$$

where:

$\mathbf{x}_i^g \in \mathbb{F} \subseteq \mathbb{S} \subseteq \mathbb{R}^n$	:	Position vector of the ith individual at g^{th} generation.
$x_{i,j}^g$	:	j^{th} coordinate of the ith individual at g^{th} generation.
$\mathbb{F}$, $\mathbb{S}$, $\mathbb{R}$	:	Feasible space, Search Space, and Real numbers.
n	:	Dimensionality (i.e. number of object variables).
m	:	Number of individuals in the population.
g	:	Index to identify the generation.
i	:	Index to identify the individual.
j	:	Index to identify the dimension.

Refer to (Gen & Cheng, 2000) for a discussion on the properties that the genotype-to-phenotype encoding must exhibit.

A population of a few hundred individuals is typical in GAs. If the computational cost is flexible, unnecessarily large populations imply unnecessarily high computational cost, whereas populations smaller than necessary compromise the quality of the solution evolved. Given a fixed permitted computational cost, a trade-off must be found: the larger the population size the more parallel the search becomes; while the longer the search-length the longer evolution has to do its work.

Typically, initialization consists of randomly spreading the individuals throughout the search-space. If available, domain specific or problem knowledge should be exploited.

3.3.3.2. Parents' selection

The most widespread parents' selection scheme in GAs is the roulette-wheel discussed in section 3.3.2.3, where the probability that the ith individual has of being selected among all m individuals to breed is as shown in Eq. (3.5).

$$p(\mathbf{x}_i) = \frac{f(\mathbf{x}_i)}{\sum_{j=1}^{m} f(\mathbf{x}_j)} \tag{3.5}$$

For other selection schemes, refer to (Bäck, 1996); (Gen & Cheng, 2000); (Sastry, Goldberg, & Kendall, 2005); (De Jong, 2006); among others.

3.3.3.3. Genetic alterations

Crossover

The predefined number of parents that are to be involved in each breeding process is ρ out of the μ selected parents. Therefore the crossover operator acts on ρ individuals at a time by recombining their existing genetic material without widening the genetic pool. Typically, in GAs, $\rho = 2$ and either one or two descendants are bred per crossover.

In other words, once the parents' selection is through, a mating pool of μ individuals has been generated. Only some of them will actually perform crossover, as a probabilistic threshold must be passed. For each breeding process, ρ parents from the mating pool are randomly paired. The process is repeated until all individuals in the mating pool have been matched and the complete offspring population has been generated. The number of parents in the mating pool and the number of times this process is repeated depend on the characteristics of the crossover operator. For instance, say that two parents ($\rho = 2$) are involved in each mating process. The exchange of genetic material during crossover naturally leads to the breeding of two descendants. In this case, the number of parents in the mating pool equals the number of individuals in the population ($\mu = m$), and the number of times crossover is performed per generation equals half the population-size.

There are schemes where only the fitter child is kept, so that only one child is bred per crossover. Then, the number of parents is doubled, and so is the number of applications of the crossover operator per generation, which now equals the population-size.

The way crossover is performed gives rise to different techniques, which may depend on the data structure of the individual. The original operator is the *1-point crossover*, where a position between two bits is randomly chosen from a uniform distribution, and the chromosomes of both parents are split in two parts. One of these parts is exchanged to give birth to two new children of the same size. An extension of this is the *k-point crossover*. A graphical example of the *1-point crossover* is offered in Fig. 3.6, and an example of a *two-point crossover* is shown in Fig. 3.7.

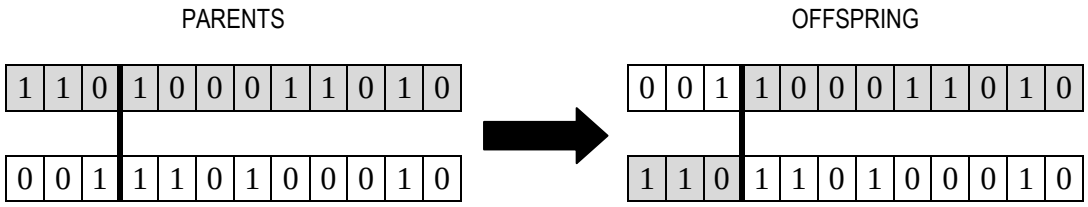


Fig. 3.6. Example of one-point crossover.

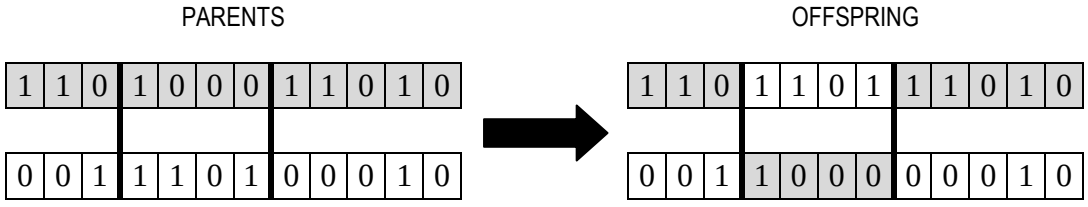


Fig. 3.7. Example of two-point crossover.

Another common operator is the *uniform crossover*, where each gene of the descendant is randomly taken from any one of the parental chromosomes. If two parents are to breed two descendants per crossover, the procedure consists of randomly swapping analogous genes between the two parental chromosomes. In the so-called *half uniform crossover*, half of the non-matching homologous genes are swapped.

The *position-based crossover* is an adaptation of the *uniform crossover* for individuals encoded as an ordered list. It consists of randomly choosing a number of genes from one parent, and passing it on to the corresponding gene in the descendant. The blanks are filled in from the other parent, from left to right. Hence the genes passed on from the

second parent will not be stored, in general, in the homologous genes of the child (i.e. in the genes of the same index). Refer to (Gen & Cheng, 2000, p. 349) for further reading. Sastry, Goldberg, & Kendall (2005) discuss a similar operator called *cycle crossover*.

Different crossover operators can be defined for real-coded GAs. For instance, the *flat crossover* is as follows: given two parents, each gene of the descendant is generated at random from a uniform distribution, using the homologous genes of two randomly paired parents as upper and lower bounds. In turn, the *discrete crossover* is equivalent to the *uniform crossover* discussed for binary GAs, where each gene of the descendant is directly taken from one of its parents at random. Another common operator for real-valued GAs is the *arithmetical crossover*, where two descendants $\mathbf{x}^{o1}$ and $\mathbf{x}^{o2}$ are bred from two parents $\mathbf{x}^{p1}$ and $\mathbf{x}^{p2}$ as shown in Eq. (3.6):

$$\begin{aligned} \mathbf{x}^{o1} &= \lambda \cdot \mathbf{x}^{p1} + (1-\lambda) \cdot \mathbf{x}^{p2} \\ \mathbf{x}^{o2} &= \lambda \cdot \mathbf{x}^{p2} + (1-\lambda) \cdot \mathbf{x}^{p1} \end{aligned} \tag{3.6}$$

where λ is constant or dependant on the generations. In the first case, the operator is *uniform*, while in the second it is *non-uniform*. For other recombination operators for real-coded GAs, refer to (Herrera, Lozano, & Verdegay, 1998) and (Haupt & Haupt, 2004, pp. 57-60). For further reading on crossover operators for binary GAs, also refer to (Bäck, 1996); (Sastry, Goldberg, & Kendall, 2005); and (Gen & Cheng, 2000).

Mutation

Mutation was introduced by Holland (quoted in (Mitchell, 1999)) as a background operator to be applied after crossover, whose main purpose was to introduce variety when facing loss of diversity and exploration by crossover only is no longer possible. It is typically given a low probability of occurrence. Usually $0.001 \leq \mathrm{MR} \leq 0.005$.

The mutation operator acts on a single individual, and is expected to perform a sort of explorative local search in the neighbouring area thus generating new genetic material. The way to implement this greatly depends on the data structure of the individuals. In canonical, binary GA, the mutation operator consists of flipping every gene in the chromosome, whereas its design is not so straightforward in real-coded GAs. Numerous alternatives can be found in the literature, the simplest of which is the *random mutation*,

where the value of each gene is randomly generated from a uniform distribution within the feasible interval of the variable. Hence it is not really a mutation but a re-generation operator. For a good overview of different mutation operators for real-valued GAs, refer to (Herrera, Lozano, & Verdegay, 1998, pp. 292-294).

3.3.3.4. Fitness evaluation and survivors' selection

Once all the genetic alterations have been carried out, the fitness of the newly created individuals has to be computed somehow, typically by means of a fitness function. The survivors' selection is not performed in canonical GAs, or it can be viewed as selecting the whole offspring population for the next generation. The complete replacement of the current population by its progeny is referred to as *generational replacement*. Other alternative GAs perform some sort of elitism, or some other replacement strategy.

Discussing the hard theory behind GAs (e.g. the *building blocks hypothesis*, the *schema theorem*, etc.) is beyond the scope of this overview. Refer to (Davis & Mitchell, 1991); (Bäck, 1996); (Herrera, Lozano, & Verdegay, 1998); (Sastry, Goldberg, & Kendall, 2005); (De Jong, 2006, pp. 115-209); (Fogel D. B., 2008); among others.

3.3.4. Genetic Programming

GP can be thought of as the application of GAs to write computer programs. Hence it is a computer program that writes other computer programs. Naturally, appropriate data structures and genetic operators are required. Computer programs are traditionally written in a deterministic manner, profiting from previously acquired knowledge on the specific targeted problem. Conversely, GP writes computer programs without such knowledge. *Genetic Programming is a systematic method for getting computers to automatically solve a problem starting from a high-level statement of what needs to be done.* It (...) *iteratively transforms a population of computer programs into a new generation of computer programs by applying analogs of naturally occurring genetic operations.* (Koza & Poli, 2005)

3.3.4.1. Representation

Although any standard programming language such as *Fortran*, *C* or *Matlab* could be used to implement a GP algorithm, the LISP (short for LISt Processing) programming

language commonly used in AI is better suited for the task. Koza (1990) offers a few reasons for choosing such a language. Therefore, (...) *the search space is the hyperspace of LISP "symbolic expressions" (called S-expressions) composed of functions and terminals appropriate to the problem domain* (Koza, 1994).

Thus, individuals are represented by tree-like structures called *syntax trees* (also *parse trees*), and the evolutionary search is carried out over the hyperspace of valid trees. A *syntax tree* is stored in LISP in the form of an ordered list of elements (functions, numbers, names, sub-lists, etc.), where the number and type of elements need not be set in advance (Haupt & Haupt, 2004, p. 196). Every individual comprises a potential solution to the problem in hand, standing for a whole syntactically valid executable computer program. The *syntax tree* is composed of genes –also referred to as *nodes* or *points*– and *links*. There are two types of genes in GP:

Terminal genes: They are the ending nodes, which represent constants and variables, standing for the leaves of the trees (no branches coming out).

Function genes: They are intermediate nodes in the trees, which represent functions such as arithmetic operations (+, –, %), mathematical functions (*sin*, *cos*, *log*, *sqrt*, *max*, *min*), Boolean (logical) operations (*and*, *or*, *not*), conditional operations (*if-then-else*), etc. The so-called *links* coming out of the *function genes* connect the latter to *terminal genes* or to *sub-trees* standing for the arguments of the function.

Illustrative examples of three individuals represented by *syntax trees* are offered in Fig. 3.8, where the represented functions are shown above each tree.

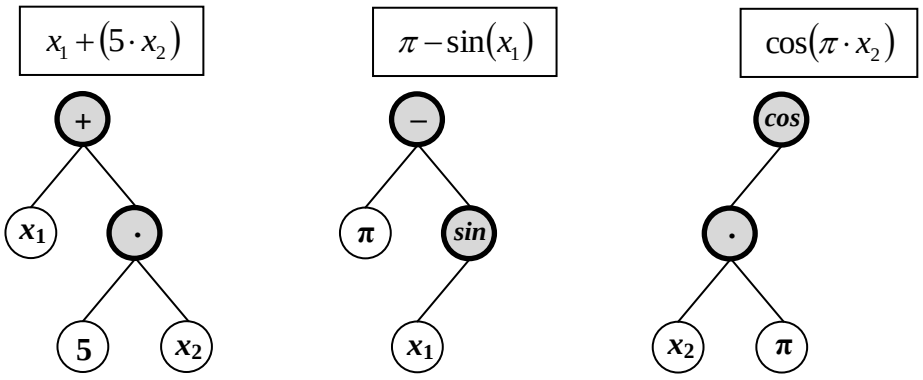


Fig. 3.8. Examples of the tree-like graphical representation of three individuals standing for different functions. The standard expression is shown above each tree, while the storage in LISP language is in the form of an ordered list.

The nodes of the trees stand for the *genes*, and the connections between nodes are the *links*. The shaded nodes are *function genes*, whereas the non-shaded ones are *terminal genes*. The uppermost node is called the *root*, and the lower level sub-trees are called *branches*. These trees can be represented in *prefix notation* such as LISP *S-expressions*, where functions always precede their arguments (Koza & Poli, 2005). For instance, the *prefix notation* for the first individual in Fig. 3.8 would be (+ *x*1 (· 5 *x*2)).

Prior to the initialization of the population, the sets of available *terminal* and *function genes* for the individuals must be specified. This could limit the algorithm to the extent of being unable to evolve a good solution. It is always preferable to include more rather than fewer genes than necessary, as useless genes would simply die out throughout evolution. Following a path in a tree, the depth of a node is the minimal number of nodes from the root to the node in hand. The maximum and minimum depth of the *syntax trees* must also be specified in advance.

3.3.4.2. Initialization and fitness evaluation

Loosely speaking, each individual in the initial population is a list of random functions and terminals shaping a computer program. Once the sets of genes have been specified, the population of individuals needs to be initialized. Two traditional procedures are the *grow* method and the *full* method. The former generates trees of diverse shapes with no minimum depth guaranteed; while the latter results in all trees having the maximum depth, but this also implies some loss of diversity. The so-called *ramped half-and-half* method comprises a convenient combination of both. Refer to (Koza, 1992, pp. 92-93).

- *Grow method*. The maximum depth (*md*) permitted for the individuals must be specified. One individual is created at a time, beginning from the root of every tree. Every node is first decided to be either a function or a terminal gene, and the appropriate type of gene is randomly selected from the corresponding set. If it is a function gene, it is given as many children (terminals or rooted sub-trees) as the arity of the function (number of arguments it requires). The initialization process starts again for each child unless the *md* level has been reached, in which case a terminal gene is randomly selected. This method provides a variety of structures in the population, but does not guarantee individuals of at least a certain depth.

- *Full method*. The terminal genes are guaranteed to be at a specified depth (*d*). Every gene at a depth lower than *d* is randomly selected from the set of function genes only. If the depth equals *d*, the gene is selected form the set of terminal ones. Therefore every tree has a depth equal to *d*.
- *Ramped half-and-half.* The maximum depth must be specified. The population is uniformly divided into (*md* – 1) groups. Half of each group is initialized by the *grow method* and half by the *full method*. For the first group, the depths *md* and *d* are set to two, increased to three for the second, and so on. Thus, diversity in the population is guaranteed, and individuals of only one level cannot be generated.

The evaluation of an individual's fitness is not straightforward. It can be evaluated in terms of the mean squared error of the outputs of the generated program with respect to a set of preselected benchmarking problems, or in terms of the accuracy in approximating a set of points in a polynomial interpolation. It usually comes at a very high computational cost. Although it depends on the problem in hand, it is common practice to use big populations; in general, from 500 to a few thousand individuals.

3.3.4.3. Parents' selection

The parents' selection procedure is the same as in GAs. Hence any of the selections schemes previously discussed are applicable. The most common ones are the *roulette wheel* and its variations, and the *tournament* selection. The important issue is that the selection scheme is probabilistically fitness-based, as the GP paradigm is not greedy.

3.3.4.4. Genetic alterations

The most important –and frequently the only– genetic operator in GP is crossover. This is because it is more or less straightforward, and it does not present much difficulty in generating valid tree structures. There are a few alternatives for the mutation operator, which may be applied after or instead of crossover. There are versions of GP that allow an individual to be subject to both genetic alterations, while others only allow one at a time. In the latter case, the probability thresholds are set so that the individual selected in the parents' selection scheme has a chance to be cloned and passed unmodified to the next generation, OR undergo crossover, OR undergo mutation. Note that *perform re-*

production in GP merely refers to cloning the individual. Other genetic operators may be designed, such as the *architecture altering operations* discussed in (Koza & Poli, 2005), which modifies the architecture of the selected individual program.

Crossover

It is similar to crossover in binary GAs and to *discrete crossover* in real-coded GAs. A child is generated with part of the genetic code of one parent and part of the other, with no new genetic material created. That is, a crossover point is chosen at random on a *link* of each parent, and the sub-trees rooted are interchanged. An example of a possible crossover between the first two individuals in Fig. 3.8 is shown in Fig. 3.9, where two children were bred. Note that the depth of individuals might change during crossover. Similar to binary GAs, the CR is typically set between 0.6 and 0.9, the size of the population is kept constant, and there is generational replacement.

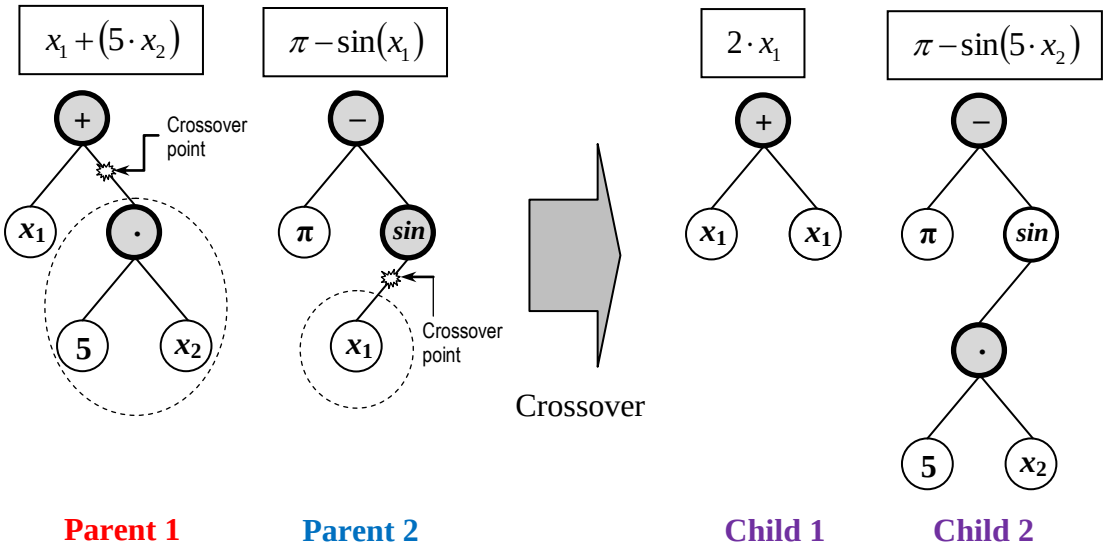


Fig. 3.9. Example of crossover in GP. The genetic material of both children is contained in the parents' genome, while the depth of the children bred might differ from that of their parents.

Mutation

It is a secondary operator. Therefore, it is typically awarded a low MR to maintain diversity in the population. The same as for the crossover operator, the definition of a mutation operator is not straightforward in GP. Some of the most common mutation operators are as follows (Innocente, 2006):

- *Point mutation*. A single gene is replaced by a randomly generated single gene of the same type. That is to say, a terminal gene must be replaced by another terminal gene, and a function gene must be replaced by a function gene of the same arity. An example is offered in Fig. 3.10.
- *Expansion mutation*. A terminal gene is replaced by a randomly generated sub-tree.
- *Collapse mutation*. A sub-tree is replaced by a randomly generated terminal gene.
- *Sub-tree mutation*. A sub-tree is replaced by a new, randomly generated sub-tree.
- *Per-mutation*. Two genes of the same type within the individual are permutated.

Note that the *expansion*, *collapse* and *sub-tree* mutations can be viewed as a crossover between the individual at issue and a new randomly generated one.

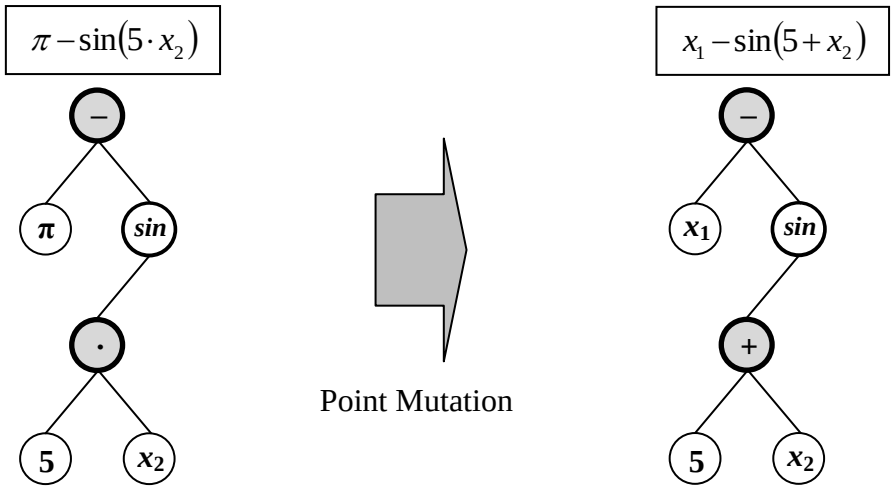


Fig. 3.10. Example of point mutation in GP. Although it is unlikely that two genes of the same individual at mutated at once, the mutation of one function gene and of one terminal gene is performed for illustrative purposes.

Fitness evaluation and survivors' selection

Once the genetic alterations have been performed, the fitness of the new population is evaluated. Since there is *generational replacement*, the survivors' selection simply consists of all individuals in the offspring population being selected for the next generation.

For a concise tutorial on GP, refer to (Koza & Poli, 2005). For an extensive review, refer to (Koza, 1993) and its sequels (four volumes altogether), and to (Ferreira, 2006).

3.3.5. Evolution Strategies

They comprise a family of phenotypic EAs aimed at real-valued optimization problems.

3.3.5.1. Representation, initialization and fitness evaluation

The (1+1)-ES

The original ES was composed of a single individual and is now referred to as the **(1+1)-ES**. The method consists of randomly generating an initial solution represented by an *n*-dimensional vector, each of whose components stands for an object variable. The individual is cloned at every generation, and the clone is subject to a mutation operator. The latter consists of introducing variation in the variables by adding a vector of small random numbers from a zero-mean normal distribution. This is in line with observations in nature: children are similar to their parents, while smaller changes occur more often than larger ones. The fitness of the new individual is evaluated, and the best between *father* and *child* survives. One of the innovative features was the incorporation of the mutation parameters within the individual, so that they would also be subject to evolution. Initially, the vector of standard deviations was constant over time. Later, Rechenberg (1973) postulated the *1/5-success-rule* (refer to Eq. (3.9) in section 3.3.5.3) to update the standard deviations, based on results obtained in the study of the optimum standard deviations combined with the probabilities of a successful mutation (Bäck, Hoffmeister, & Schwefel, 1991, pp. 2-3). The individual is represented as in Eq. (3.7).

$$\mathbf{p}^{(t)} = \left(\mathbf{x}^{(t)}, \boldsymbol{\sigma}^{(t)}\right) \quad \in \quad \mathbb{R}^n \text{ x } \mathbb{R}^n \tag{3.7}$$

The (μ+1)-ES

This is the first population-based ES, therefore allowing sexual reproduction. It consists of breeding one child from two parents selected with the same probability among all μ individuals, and then choosing the best μ among the (μ+1) individuals in the extended population for survival. The representation of the ith individual is shown in Eq. (3.8).

$$\mathbf{p}_i^{(t)} = \left(\mathbf{x}_i^{(t)}, \boldsymbol{\sigma}_i^{(t)}\right) \quad \in \quad \mathbb{R}^n \text{ x } \mathbb{R}^n \tag{3.8}$$

The (μ+λ)-ES and the (μ,λ)-ES

These strategies were derived from the **(μ+1)-ES**, but here λ descendants are bred from ρ parents randomly selected with the same probability among all μ individuals. Thus, μ among the aggregated (μ+λ) individuals are selected in the **(μ+λ)-ES**, whereas μ among the $\lambda \geq \mu$ descendants are selected in the **(μ,λ)-ES** to survive for the next generation. The representation of each individual is the same as in Eq. (3.8).

3.3.5.2. Parents' selection

The (1+1)-ES

There is no parents' selection in this strategy, as there is only one single individual.

The (μ+1)-ES

Two parents are randomly selected among all μ individuals in the population. Every individual has the same probability of being chosen regardless of its fitness.

The (μ+λ)-ES and the (μ,λ)-ES

In these strategies, ρ out of the μ individuals in the population are randomly selected to breed one single descendant, where every individual has the same probability of being selected. Typically, ρ = 2 or ρ = μ. This is a λ-fold operator, as these strategies require λ descendants while only one is generated from every set of ρ parents selected.

3.3.5.3. Genetic alterations

The (1+1)-ES

Mutation

The standard deviations are updated first by the *1/5-success-rule*, and then the mutated standard deviations are used to mutate the current solution. The *1/5-success-rule* is as shown in Eq. (3.9).

$$\boldsymbol{\sigma}'^{(t)} = m\left(\boldsymbol{\sigma}^{(t)}\right) = \begin{cases} cd \cdot \boldsymbol{\sigma}^{(t)} & \text{if } p_s^{(t)} < 1/5 \\ ci \cdot \boldsymbol{\sigma}^{(t)} & \text{if } p_s^{(t)} > 1/5 \\ \boldsymbol{\sigma}^{(t)} & \text{if } p_s^{(t)} = 1/5 \end{cases} \tag{3.9}$$

where $p_s^{(t)}$ is the frequency of successful mutations measured over a certain interval of trials, and usually cd <1 and ci >1. Schwefel (1981) suggests using the values cd = 0.82 and ci = 1/cd for the adjustment, which should take place every n mutations (Bäck, Hoffmeister, & Schwefel, 1991). Thus, the clone is mutated as in Eq. (3.10).

$$\mathbf{x}'^{(t)} = \mathbf{x}^{(t)} + \mathbf{N}_{(0,\boldsymbol{\sigma}'^{(t)})} \tag{3.10}$$

where $\mathbf{N}_{(0,\boldsymbol{\sigma}'^{(t)})}$ stands for a vector whose components are independent random numbers generated from zero-mean normal distributions with standard deviations equal to the corresponding components of the vector $\boldsymbol{\sigma}'^{(t)}$ obtained from Eq. (3.9).

The (μ+1)-ES

Crossover

The crossover between the two selected parents breeds a child by randomly choosing each component from either one or the other parent (*discrete crossover*), and is applied both to the object variables and to the standard deviations. Thus,

$$\begin{aligned} & p'^{(t)}_{\mu+1,j} = \begin{cases} p^{(t)}_{p1,j} & \text{if} \quad U_{(0,1)} \le 0.5 \\ p^{(t)}_{p2,j} & \text{otherwise} \end{cases} \\ & j = 1,\ldots,2 \cdot n \\ & \mathbf{p}'^{(t)}_{\mu+1} = \left(\mathbf{x}'^{(t)}_{\mu+1} \quad \boldsymbol{\sigma}'^{(t)}_{\mu+1}\right) \in \mathbb{R}^{2 \cdot n} \end{aligned} \tag{3.11}$$

where n is the dimension of the search-space, $p1$ and $p2$ are the indices identifying the parents, j is the index identifying the component of the individual (including coordinates and corresponding standard deviations), $\mathbf{p}'^{(t)}_{\mu+1}$ is the descendant generated, and $U_{(0,1)}$ is a random number generated from a uniform distribution between '0' and '1'.

Mutation

In the same fashion as in the **(1+1)-strategy**, the mutation is applied only to the object variables of the descendant, whereas the update of the standard deviations follows the *1/5-success rule* in Eq. (3.9). Thus, the child generated by recombination in Eq. (3.11) is mutated as in Eq. (3.12), where $\mathbf{N}_{(0,\boldsymbol{\sigma}''^{(t)})}$ is as explained before.

$$\mathbf{x}''^{(t)}_{\mu+1} = \mathbf{x}'^{(t)}_{\mu+1} + \mathbf{N}_{(0,\boldsymbol{\sigma}''^{(t)})} \tag{3.12}$$

After all the genetic alterations have been performed, the descendant is as in Eq. (3.13).

$$\mathbf{p}''^{(t)}_{\mu+1} = \left(\mathbf{x}''^{(t)}_{\mu+1} \quad \boldsymbol{\sigma}''^{(t)}_{\mu+1}\right) \tag{3.13}$$

The (μ+λ)-ES and the (μ,λ)-ES

Crossover

This is a λ-fold operator. It is performed λ times, breeding a single descendant per application. Typically, the crossover operators used for the object variables and for the strategy parameters are not the same. A *discrete crossover* is usually preferred for the object variables whereas an *intermediate crossover* is preferred for the strategy parameters. The *discrete crossover* consists of choosing the components of the child from the ρ selected parents at random, while the *intermediate crossover* consists of the arithmetic average of the parents' corresponding strategy parameters[1]. Hence the higher the number of parents involved in breeding the higher the mixing of the genetic information. Consider two individuals to be crossed-over, and the hyper-space spanned between them. By means of *discrete crossover*, a child can only be placed in one of the vertices of the hyper-polyhedron enclosing the hyper-space, while *intermediate crossover* places the child in the middle of the line that links the parents' positions. Therefore *intermediate crossover* is better at exploiting the space inside the hyper-polyhedron at the expense of the loss of diversity, which happens to be one of the main features EAs rely on. Conversely, *discrete crossover* is more robust in that it maintains higher diversity.

[1] Notice the resemblance to the *arithmetical crossover* in real-valued GAs. If the number of parents is $\rho = 2$ here, and $\lambda = 0.5$ in the arithmetical crossover in GAs, the latter and the *intermediate crossover* here coincide.

Other crossover operators can be found in (Bäck, 1996), where the child is placed in different areas of the aforementioned hyper-space spanned between the parents. Since a child cannot be placed outside such hyper-space by means of traditional crossover operators, they result in volume reduction. In contrast, the mutation operator widens the genetic pool. Thus, for the case of two parents, *discrete* crossover for the object variables, and *intermediate* crossover for the strategy parameters, the temporary population of λ descendants is obtained as shown in Eq. (3.14).

$$\begin{aligned} x'^{(t)}_{i,j} &= \begin{cases} x^{(t)}_{p1,j} & \text{if} \quad U_{(0,1)} \le 0.5 \\ x^{(t)}_{p2,j} & \text{otherwise} \end{cases} \\ \sigma'^{(t)}_{i,j} &= \frac{\sigma^{(t)}_{p1,j} + \sigma^{(t)}_{p1,j}}{2} \\ \mathbf{p}'^{(t)}_{i} &= \left(\mathbf{x}'^{(t)}_{i} \quad \boldsymbol{\sigma}'^{(t)}_{i} \right) \in \mathbb{R}^{2 \cdot n} \\ i &= 1,\ldots,\lambda \quad ; \quad j = 1,\ldots,n \end{aligned} \tag{3.14}$$

Mutation

It is typically applied to the object variables and to the standard deviations following different rules, as shown in Eq. (3.15). Thus, each object variable is mutated by adding a random number obtained from a zero-mean normal distribution, whereas the standard deviations undergo multiplicative logarithmic normal mutations. Bear in mind that the standard deviations are mutated before mutating the object variables.

$$\begin{aligned} & \mathbf{p}_i{}''^{(t)} = m\left(\mathbf{p}_i{}'^{(t)}\right) = \left(\mathbf{x}''^{(t)}_{i} \quad \boldsymbol{\sigma}''^{(t)}_{i} \right) \quad ; \quad \mathbf{p}_i{}''^{(t)} \in I = \mathbb{R}^n \text{ x } \mathbb{R}^n \\ & \left. \begin{aligned} \sigma'^{(t)}_{i,j} &= \sigma'^{(t)}_{i,j} \cdot e^{\tau \cdot N_{(0,1)} + \tau' \cdot \overline{N}_{(0,1)}} \\ x''^{(t)}_{i,j} &= x'^{(t)}_{i,j} + \sigma'^{(t)}_{i,j} \cdot N_{(0,1)} \end{aligned} \right\} \quad ; \quad \begin{cases} i = 1,\ldots,\lambda \\ j = 1,\ldots,n \\ \overline{N}_{(0,1)} = \text{constant} \quad \forall j \end{cases} \end{aligned} \tag{3.15}$$

where:

$N_{(0,1)}$: Random real number obtained from a zero-mean normal distribution with standard deviation equal to "1", resampled anew for each coordinate, for each individual and for each generation (i.e. every time it is referenced).

$\overline{N}_{(0,1)}$: Random real number obtained from a zero-mean normal distribution with standard deviation equal to "1", resampled anew for each individual and for each generation but constant for all dimension within an individual.

τ, τ' : Coefficients that rule the adaptive mutation step-sizes. Some recommended values are $\tau = \frac{1}{\sqrt{2 \cdot \sqrt{n}}}$ and $\tau' = \frac{1}{\sqrt{2 \cdot n}}$ (see (Bäck & Schwefel, 1996)).

Given that the standard deviations for different variables are independent from one another, improvement by mutation (plus selection) resembles a hill-climbing process zigzagging along the gradient (Bäck, Hoffmeister, & Schwefel, 1991). In order to improve the rate of progress, (Scwefel, 1981) extended the mutation operator to deal with *correlated mutations* by incorporating a new vector of strategy parameters $\boldsymbol{\alpha}_i^{(t)}$ (also referred to as $\boldsymbol{\theta}_i^{(t)}$). In this case, the representation of the individual consists now of the concatenation of three distinct vectors, as shown in Eq. (3.16).

$$\mathbf{p}_i^{(t)} = \left(\mathbf{x}_i^{(t)}, \boldsymbol{\sigma}_i^{(t)}, \boldsymbol{\alpha}_i^{(t)}\right) \tag{3.16}$$

The vectors $\boldsymbol{\sigma}$ *and* $\boldsymbol{\alpha}$ *represent the complete covariance matrix of the n-dimensional normal distribution, where the covariances are given by rotation angles describing the coordinate rotations necessary to transform an uncorrelated mutation vector into a correlated one.* (Bäck & Schwefel, 1996)

The new vector is also subject to evolution, although it undergoes a different mutation process. For a visualization of the implications of searching with independent and correlated mutations, refer to Fig. 3.11. Refer to (Schwefel, 1981); (Bäck, 1996); (Bäck, Hoffmeister, & Schwefel, 1991); (Fogel, Angeline, & Fogel, 1995); (Bäck & Schwefel, 1996) for details on the implementation of correlated mutations and the theory behind.

3.3.5.4. Fitness evaluation and survivors' selection

Once the offspring population has been bred, their fitness is evaluated. For the most frequent case of *n*-dimensional real-valued vectors and an analytically posed objective function, it simply consists of evaluating such a function for each individual. For other representations, the computation of the fitness value is problem-dependent.

In the canonical **(μ, λ)-ES**, the survivors' selection operator deterministically selects the best μ out of the λ children for the next generation. Note that this strategy is not elitist, which helps escape local optima by temporal deterioration. Thus, the life-span of an individual is limited to one generation. In the canonical **$(\mu+\lambda)$-ES** instead, the survivors'

selection is deterministic too, but it selects the best μ out of the (μ+λ) individuals, thus guaranteeing constant improvement (*elitism*). Hence the life-span of an individual is not limited. This imply that good solutions are not lost, but also that an individual who has reached a local optimum becomes an attractor difficult to escape from.

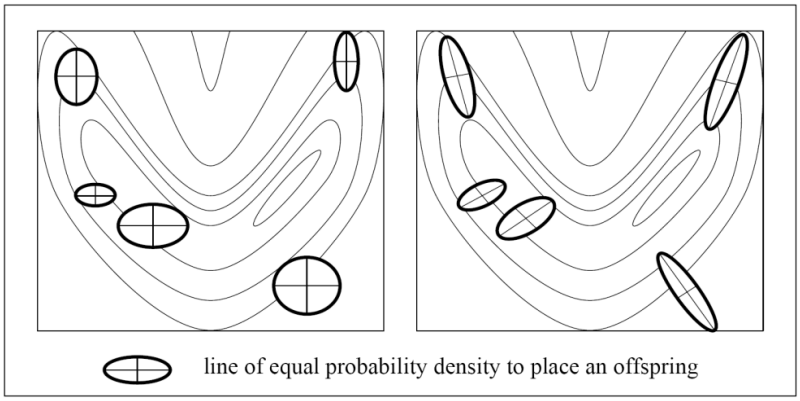


Fig. 3.11. Example of simple (left) and correlated (right) mutations. Source: (Bäck, Hoffmeister, & Schwefel, 1991).

3.3.6. Evolutionary Programming

It comprises one of the most popular phenotypic EAs. Although it was originally aimed at AI, its main current application lies on the real-valued optimization field.

3.3.6.1. Representation, initialization and fitness evaluation

Finite State Machine representation

The original objective was to generate AI through simulated evolution. The initial representation of the individuals –as envisioned in (Fogel, Owens, & Walsh, 1966)– consisted of finite state machines (FSM) aimed at performing predictions. Typically, the prediction depends on both the input and the current state of the machine. In order to validate or assess the performance of a given FSM –and hence allow its optimization–, some small part of the environment that is to be presented to the machine needs to have been already observed. That is, a set of the correct outputs corresponding to a set of inputs must be available. Thus, the given machine is presented a sequence of symbols (inputs), and returns another sequence (outputs). The performance is evaluated as a meas-

ure of the accuracy of each prediction (e.g. aggregation of mean squared errors on all sequences in the test set). An example of a FSM aimed at predictions is offered in Fig. 3.12. The table presents the outputs corresponding to each pair (state, input). Notice that the machine also changes states (A, B, C). The new state is the one shown in the next column. The graph is rather self-explanatory, where the inputs (0, 1) are shown to the left and the outputs (I, II, III, IV) to the right of the slash. The machine is in state **A** when it is presented the sequence **(1, 0, 0, 1, 0, 1, 1, 1, 0)**, to which it returns the sequence **(II, II, IV, I, II, I, III, II, II)**.

STATE	A	A	B	B	C	B	C	A	A
INPUT	1	0	0	1	0	1	1	1	0
OUTPUT	II	II	IV	I	II	I	III	II	II

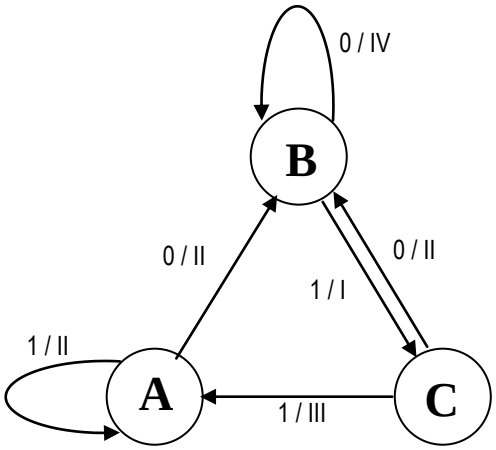


Fig. 3.12. Example of a finite state machine with three states (A, B, C), two inputs (1, 2) and four outputs (I, II, III, IV). The table above shows that, for an initial state **A**, if the sequence of inputs **(1, 0, 0, 1, 0, 1, 1, 1, 0)** is presented to this machine, it returns the sequence **(II, II, IV, I, II, I, III, II, II)**. This type of machine is typically used for predictions.

This representation is not as straightforward as a binary or real-valued vector, and hence the object variables are not so obvious. *There are five modes of varying an individual finite state machine: (1) add a state, (2) delete a state, (3) change the start state, (4) change an output symbol, (5) change a next-state transition* (Michalewicz & Fogel, 2004, p. 176).

Thus, the size of the individual might not be fixed as in most EAs, as the size and architecture of the machine may be allowed to vary.

In this context, optimizing a finite state machine means finding the design that returns the best predictions. If the predictions are straightforward and hence their accuracy is not an issue, optimizing might mean finding the design that does the job with the fewer

number of states. In any case, the meaning of an optimum FSM needs to be reflected in the fitness function. Thus, the architecture of each FSM is randomly initialized, and the experienced environment is presented to it. Its fitness is computed as a function of the accuracy of its performance.

Vector of real-valued object variables

David B. Fogel (1992) reformulated the method for real-valued optimization problems. This modern version of EP resembles the ESs paradigm is several aspects, namely the representation of the individuals, the mutation operator, the selection schemes, and more importantly, the improvement of the self-adaptation of the strategy parameters. Hence the strategy parameters evolve as the algorithm self-adapts to the environment. It is fair to note that the evolution of the strategy parameters was incorporated to the ESs approximately 20 years earlier. Thus, individuals are encoded in *n*-dimensional floating point vectors as shown in Eq. (3.8). In the same fashion as in ESs, the individual might incorporate in its representation an additional vector of strategy parameters in case of correlated mutations, in which the representation of the individual is as in Eq. (3.16).

Thus, a population of μ individuals is randomly initialized, and their fitness values are evaluated. Beware that only the part of the individual which comprises a candidate solution is involved in the fitness evaluation.

3.3.6.2. Parents' selection

Every individual in the population is selected to undergo mutation. This is one of the few differences with ESs.

3.3.6.3. Genetic alterations

Mutation

This is the only genetic alteration in EP. Therefore individuals do not really reproduce by are cloned, and these clones are later mutated. The object variables and the strategy parameters also undergo mutation following different rules. The former are mutated by adding a random number obtained from a zero-mean normal distribution whereas the latter are mutated by a similar additive but self adaptive rule, as shown in Eq. (3.17).

$$\mathbf{p}_i'^{(t)} = m\left(\mathbf{p}_i^{(t)}\right) = \left(\mathbf{x}_i'^{(t)} \quad \boldsymbol{\sigma}_i'^{(t)}\right) \quad ; \quad \mathbf{p}_i'^{(t)} \in I = \mathbb{R}^n \text{ x } \mathbb{R}^n$$
$$\left.\begin{aligned} \sigma_{i,j}'^{(t)} &= \sigma_{i,j}^{(t)} + k \cdot \sigma_{i,j}^{(t)} \cdot N_{(0,1)} \\ x_{i,j}'^{(t)} &= x_{i,j}^{(t)} + \sigma_{i,j}'^{(t)} \cdot N_{(0,1)} \end{aligned}\right\} \quad ; \quad \begin{cases} i = 1,\ldots,\mu \\ j = 1,\ldots,n \\ k \cong 0.2 \end{cases} \tag{3.17}$$

where:

$N_{(0,1)}$: Random real number obtained from a zero-mean normal distribution with standard deviation equal to "1", resampled anew every time it is referenced.

k : Scaling factor: $k \approx 0.2$ (Bäck & Schwefel, 1996, p. 22).

Alternatively, the update of the standard deviations is performed by the multiplicative logarithmic normal mutation shown in Eq. (3.15) for ESs instead of the additive self-adaptation shown in Eq. (3.17). This is reproduced and modified for EP in Eq. (3.18).

$$\left.\begin{aligned} \sigma_{i,j}'^{(t)} &= \sigma_{i,j}^{(t)} \cdot e^{\tau \cdot N_{(0,1)} + \tau' \cdot \overline{N}_{(0,1)}} \\ x_{i,j}'^{(t)} &= x_{i,j}^{(t)} + \sigma_{i,j}'^{(t)} \cdot N_{(0,1)} \end{aligned}\right\} \quad ; \quad \begin{cases} i = 1,\ldots,\mu \\ j = 1,\ldots,n \\ \overline{N}_{(0,1)} = \text{constant} \quad \forall j \end{cases} \tag{3.18}$$

The strategy of correlated mutations is also viable in EP, so that the individual may also be represented by three concatenated vectors. Refer to (Scwefel, 1981); (Bäck, 1996); (Bäck, Hoffmeister, & Schwefel, 1991); (Fogel, Angeline, & Fogel, 1995); (Bäck & Schwefel, 1996) for details on the correlated mutations and the theory behind.

3.3.6.4. Fitness evaluation and survivors' selection

Once the μ cloned individuals are mutated, their fitness is somehow evaluated according to the problem at issue. An intermediate population is created by the aggregation of the original and the cloned-mutated populations. For the survivors' selection, μ out of the (μ+μ) individuals in the intermediate population are selected for the next generation. A probabilistic fitness-based selection scheme was originally implemented, although the tournament selection scheme is more popular nowadays. Nevertheless, any of the previously discussed selection schemes is a valid alternative. This is another difference with ESs, whose survivors' selection is deterministic (and might or might not be *elitist*).

3.3.7. Differential Evolution

3.3.7.1. Representation, initialization and fitness evaluation

DE is a population-based method for *n*-dimensional real-valued search-spaces. Hence individuals are encoded in *n*-dimensional floating point vectors as shown in Eq. (3.19).

$$\mathbf{x}_i^g = \left(x_{i,1}^g \quad \ldots \quad x_{i,j}^g \quad \ldots \quad x_{i,n}^g \right) \tag{3.19}$$

where:

$\mathbf{x}_i^g \in \mathbb{F} \subseteq \mathbb{S} \subseteq \mathbb{R}^n$	:	Position vector of the ith individual at g^{th} generation.
$x_{i,j}^g$	:	j^{th} coordinate of the ith individual at g^{th} generation.
$\mathbb{F}$, $\mathbb{S}$, $\mathbb{R}$	:	Feasible space, Search Space, and Real numbers.
n	:	Dimensionality (i.e. number of object variables).
m	:	Number of individuals in the population.
g	:	Index to identify the generation.
i	:	Index to identify the individual.
j	:	Index to identify the dimension.

The population size is recommended to be ten times the number of object variables as a first approximation. The typical initialization is random with uniform distribution. Once the initial positions are defined, their fitness is evaluated.

3.3.7.2. Parents' selection

All individuals in the population are selected to undergo genetic alterations.

3.3.7.3. Genetic alterations

Mutation

There is a mutation operator and a crossover operator, where the former is typically applied first. The mutation operator here is *differential* instead of the alteration by noise generated from a predefined probability distribution in traditional EAs. All individuals

in the population undergo mutation, and a temporary *mutant population* is generated. The ith *mutant vector* –also *donor vector*– at the gth generation is shown in Eq. (3.20).

$$\mathbf{v}_i^g = \left(v_{i,1}^g \quad \ldots \quad v_{i,j}^g \quad \ldots \quad v_{i,n}^g\right) \tag{3.20}$$

The original differential mutation is as in Eq. (3.21).

$$\mathbf{v}_i^g = \mathbf{x}_{r_1}^g + F \cdot \left(\mathbf{x}_{r_2}^g - \mathbf{x}_{r_3}^g\right) \tag{3.21}$$

where r_1 and r_2 are random, mutually different integers generated in the range [1,m] (also different from i), and F is the *mutation weighting factor* (advice: F = 0.8; typically between 0.5 and 1). The *mutant vector* (or *donor vector*) is independent from the individual i undergoing mutation (*target vector*). In addition, a third vector is considered in Eq. (3.21), which is referred to as the *base vector*. Some other popular strategies to perform the *differential mutation* are outlined in Eqs. (3.22) to (3.25).

$$\mathbf{v}_i^g = \mathbf{x}_{best}^g + F \cdot \left(\mathbf{x}_{r_1}^g - \mathbf{x}_{r_2}^g\right) \tag{3.22}$$

$$\mathbf{v}_i^g = \mathbf{x}_i^g + F \cdot \left(\mathbf{x}_{best}^g - \mathbf{x}_i^g\right) + F \cdot \left(\mathbf{x}_{r_1}^g - \mathbf{x}_{r_2}^g\right) \tag{3.23}$$

$$\mathbf{v}_i^g = \mathbf{x}_{best}^g + F \cdot \left(\mathbf{x}_{r_1}^g - \mathbf{x}_{r_2}^g\right) + F \cdot \left(\mathbf{x}_{r_3}^g - \mathbf{x}_{r_4}^g\right) \tag{3.24}$$

$$\mathbf{v}_i^g = \mathbf{x}_{r_1}^g + F \cdot \left(\mathbf{x}_{r_2}^g - \mathbf{x}_{r_3}^g\right) + F \cdot \left(\mathbf{x}_{r_4}^g - \mathbf{x}_{r_5}^g\right) \tag{3.25}$$

Note that the *target vector* $\mathbf{x}_i^g$ is considered for the generation of the mutant vector $\mathbf{v}_i^g$ in the strategy described by Eq. (3.23), whereas the best individual in the current generation $\mathbf{x}_{best}^g$ is considered in the strategies described by Eqs. (3.22) to (3.24).

Although every individual undergoes mutation, whether some component of the *mutant* individual participates in the survivors' selection is decided by the crossover operator.

Crossover

It is a discrete recombination between a *target vector* and its corresponding *mutant vector* to generate a *trial vector*. The ith *trial vector* at the gth generation is as in Eq. (3.26).

$$\mathbf{u}_i^g = \left(u_{i,1}^g \quad \ldots \quad u_{i,j}^g \quad \ldots \quad u_{i,n}^g\right) \tag{3.26}$$

One of the most common operators is the *binomial crossover* shown in Eq. (3.27),

$$u_{i,j}^g = \begin{cases} v_{i,j}^g & \text{if } U_{(0,1)} \leq \text{CR} \quad \text{or} \quad j = j_{\text{rand}} \\ x_{i,j}^g & \text{otherwise} \end{cases} \tag{3.27}$$

where the jth component of the *trial vector* equals the corresponding component of the *mutant vector* if the CR is passed, and equal to that of the *target vector* otherwise. In order to ensure that the *trial vector* differs from the *target vector*, a random integer j_{rand} is generated to apply Eq. (3.27). Advice: CR = 0.90; typically between 0.50 and 1.00.

Refer to (Price, Storn, & Lampinen, 2005) for a comprehensive study of the paradigm.

3.3.7.4. Fitness evaluation and survivors' selection

For the survivors' selection, each *trial vector* is compared to the corresponding *target vector*, and the best one survives for the next generation. In general, the comparison is fitness-based –especially for unconstrained problems–, but there might be variations depending on the constraint-handling technique implemented. For instance, see (Huang, Qin, & Suganthan, 2006). For other state-of-the-art DE algorithms, refer to (Takahama & Sakai, 2006) and in (Mezura-Montes, Velázquez-Reyes, & Coello Coello, 2006).

3.4. Final remarks and closure

Nomenclature was kept as uniform as possible, although certain different names are standard in different paradigms to refer to the same variables. For instance, the number of individuals in a population is referred to as m whenever possible. However, μ is a widespread, standard denomination in EP and ESs.

Some of the more relevant concepts underlying natural evolution and its simulation were discussed, highlighting how they are related to optimization processes. The different origins and objectives of the three mainstreams –GAs, ESs, and EP– were outlined, and the creation of the broader Evolutionary Computation field to encompass all EAs

was pointed out. Thus, a high-level flowchart for a general EA was offered. Since there has been noticeable crossbreeding of techniques between these paradigms for several decades, they became more alike. At present, the main applications for all of them are related to optimization. Therefore the common features of all EAs were first presented, and all three paradigms were later discussed in some detail. References to their similarities and differences were made throughout the discussions. In turn, the GP is viewed as a particular case of a GA with a specific individual representation and appropriately redeveloped genetic operators. This and the finite state machines in the original EP paradigm are examples of how EAs can be adapted to cope with problems other than *n*-dimensional real-valued optimization problems. Finally, the newer DE paradigm was presented and discussed. It is claimed to innovate with a novel mutation operator followed by a crossover which does not involve recombination between two parents. In my opinion, the tandem mutation-crossover in DE resembles a peculiar type of crossover.

To summarize their strengths and weaknesses, it could be argued that the binary GA is better in dealing with discrete problems, whereas the real-valued GAs, ESs, and DE are direct competitors for real-valued problems. They all involve *n*-dimensional real-valued vectors, crossover, mutation, and at least one of the selection schemes is fitness-based. EP only performs mutation, which makes convergence slower. However, the advantage of driving the search only by mutation is that the individual representation is more flexible, and even the string-lengths may vary. Finally, GP is a special case which deals with evolving computer programs by developing appropriate representations and genetic operators. There have been endless discussions among researchers regarding the convenience of performing only mutation, only crossover, or both, in tandem or in parallel so that no individual undergoes both at the same time-step. To the best of my knowledge, no final conclusion in this regard has been reached.

The main EC paradigms were discussed in some details because they have been and still are a strong influence for the conception and development of the PSO paradigm. There are clearly many common features, they compete for many problems, and there is a remarkable crossbreeding of ideas. Note that newer differential mutation operators in DE involve the best solution found so that the latter becomes a sort of attractor resembling PSO, whereas mutation operators are used in PSO to maintain diversity (resembling EAs). In addition, practically the same constraint-handling techniques may be used.

Chapter 4

PARTICLE SWARM OPTIMIZATION

Swarm intelligence is the branch of artificial intelligence that studies the collective behaviour that emerges from decentralized and self-organized systems. The basic concepts underlying swarm intelligence are discussed; some examples of social behaviour in the animal kingdom are presented; and a few, relevant concepts in social psychology are included, which guide the dissertation towards a thorough understanding of the basic concepts underlying the particle swarm optimization paradigm. The Ant Colony Optimization method is briefly reviewed as one of the successful applications of swarm intelligence to optimization. Finally, the Particle Swarm Optimization paradigm is discussed in some details, emphasizing the study of the influence of the coefficients in the velocity update equation on the dynamics of the system; the neighbourhood topologies; and the incorporation of constraint-handling techniques.

4.1. Introduction

Evolutionary Algorithms (EAs) and Particle Swarm Optimization (PSO) are both population-based methods that rely on the interaction amongst relatively simple individuals. However, they lean on very different interacting mechanisms: while EAs rely on competition amongst individuals in a population, PSO does on cooperation between them in order to achieve goals that are not possible individually. Thus, populations of simple individuals are able to perform tasks far beyond their individual capabilities.

As unflattering as it may be to refer to ourselves as *relatively non-intelligent individuals*, human beings also belong to a system of that sort. Humanity achievements greatly exceed those of the most brilliant, intelligent, prominent human individuals. Although we may have individual purposes, all of them as well as our actions interact with each other and with the environment in such a way that the outcome is unpredictable, and the behaviour of the system as a whole emerges purposelessly, without central control. Perhaps the main concept is that of emergence, where the behaviour of a system can become astonishingly complex and unpredictable from very simple rules at the level of each interacting component. The concept is present in many scientific areas, some of them seemingly unrelated. Namely, *chaos theory*; *complexity theory* in physics; *self-organized criticality* in statistical physics; *emergent behaviour* in physical and biologi-

cal systems; *genetics* and *evolution*; *collective behaviour and spread of culture* in social psychology; *swarm intelligence* (SI) in artificial intelligence (AI); *connectionism* in AI, in cognitive psychology, in cognitive science, and in neuroscience; etc.

PSO is a SI-based paradigm originally developed as a simulation of social behaviour, in spite of being currently applied almost exclusively to deal with complex mathematical optimization problems. The ability of the paradigm to optimize is an emergent property rather than a deterministically implemented one. As an optimization tool, the method has proven itself robust and general-purpose in the sense that it can cope with numerous problems of notably different characteristics with few or even no adaptation. Its strength derives from the fact that the method is not specifically implemented to optimize, and therefore no a priori problem-specific knowledge is deterministically exploited in its implementation. Instead, simple rules of social behaviour are implemented, seemingly unrelated to optimization tasks. As an optimization algorithm, it is extremely difficult to set the mathematical foundations of the method due to the complexity added by multiple randomly weighted interactions between individuals in a population. Some interesting attempts have been made to study simplified versions of the algorithm, which allowed a better understanding of the dynamics of the system, and in particular of the influence of the settings of the parameters involved. The initial (published) attempts to build a theoretical framework were presented by Ozcan and Mohan (1998), (1999), while the most prominent and extensive contributions in this regard are probably those undertaken by French mathematician Maurice Clerc (Clerc, 1999), (Clerc & Kennedy, 2002), (Clerc, 2004), (Clerc, 2006a), (Clerc, 2006b), (Clerc, 2008a), (Clerc, 2008b), etc. Some other relevant contributions to the theoretical studies of PSO were presented in (van den Bergh, 2001), (Trelea, 2003), (Jiang, Luo, & Yang, 2007), (Helwig & Wanka, 2008), (Mendes, Kennedy, & Neves, 2004), (Kennedy, 2008), and (Poli, 2008).

The other main aspect of the PSO algorithm that strongly influences its performance is the behavioural links between the individuals in the population, which are commonly referred to as the topology or structure of the neighbourhood. This affects the speed of spread of information, which in turn governs the speed of convergence of the algorithm as a whole together with the coefficients' settings. Some of the most relevant works on the neighbourhoods topologies are (Kennedy, 1998), (Kennedy, 1999), (Suganthan, 1999), (Richards & Ventura, 2003), (Mendes, 2004), (Li, 2004), (Kennedy & Mendes,

2006), (Clerc, 2006a, pp. 87-101), (Mohais, 2007), (Abraham, Liu, & Chang, 2006), (Miranda, Keko, & Duque, 2008), (Akat & Gazi, 2008), among others.

In order to deal with constrained problems, some external technique needs to be incorporated into the algorithm. Given the similarities, most of the constraint-handling techniques (CHTs) used in EAs are easily adaptable. Refer, for instance, to (Coello Coello, 1999), (Coello Coello, 2000), (Hu & Eberhart, 2002), (Farmani & Wright, 2003), (Xie, Zhang, & Bi, 2004), (Takahama & Sakai, 2005), (Takahama, Sakai, & Iwane, 2006), (Fuentes Cabrera & Coello Coello, 2007), (Helwig & Wanka, 2007), (Innocente & Sienz, 2008), and (Venter & Haftka, 2008).

Other aspects to be considered are the size of the population (see (Carlisle & Dozier, 2001), (Trelea, 2003), (Auger, et al., 2007), (van den Bergh & Engelbrecht, 2001), (DeBao & ChunXia, 2009), and (DeBao & ChunXia, 2009)); the form of initialization (see (Clerc, 2008c), (Helwig & Wanka, 2008)); the number of sociality terms (see (Foryś & Bochenek, 2004), (Mendes, Kennedy, & Neves, 2004)); the synchrony of the updates (parallel or sequential); the search length; etc.

4.2. Swarm Intelligence

The first mainstream in AI was the symbolic paradigm, where the units of knowledge are represented by symbols which are then combined and handled directly by rules of binary logic. While the advent of fuzzy logic allowed introducing uncertainty, the connectionist paradigm –mainly consisting of the artificial neural network (ANN) approach to intelligence– already incorporated the concept of emergence in the field. However, intelligence was still viewed as an individual trait, as the interacting components were artificial neurons within an individual. Although the intelligent behaviour was emergent, it was developed by mimicking the structure of the individual human brain.

The term ‘Swarm Intelligence’ (SI) was coined in (Beni & Wang, 1989) to denote a class of ‘cellular robotic systems’ (Beni, 2005). (...) *Later, the term moved on to cover a wide range of studies from optimization to social insect studies, losing its robotics context in the meantime.* (Sahin, Girgin, Bayindir, & Turgut, 2008)

Today, SI is a relatively modern discipline, which may be defined as the branch of AI that deals with the collective behaviour that emerges from decentralized and self-organized systems, where self-organization occurs without a sense of purpose. It is the property of a system as a whole, whose individual parts interact locally with one another and with their environment, inducing the emergence of coherent global patterns that individuals are not aware of. Thus, instead of a traditional sophisticated central controller that regulates the global behaviour, the latter emerges from cooperative interactions of unsophisticated entities. The main applications are associated to *optimization* and *robotics*. In fact, the use of SI in *multi-robots system* is known as *swarm robotics*, which deals with large numbers of cheap and dispensable robots that intercommunicate locally with no central control. Thus, if some individuals are lost or malfunctioning, the task can still be performed provided the number of *swarm robots* that remain operative stay above a critical threshold necessary for emergence to occur.

The analysis of an emergent system becomes more complex if the individual components exhibit some intelligence of their own. In addition, if the analyzer is part of the system, perspective and objectivity are lost. Conversely, it is easy to observe that individual lives in colonies of social insects are not essential for the functioning of the system. Furthermore, emergence is far more evident in these types of organisms where individual intelligence is negligible in relation to the emergent intelligence of the system, which is known as *Social Intelligence*, *Collective Intelligence*, or *Swarm Intelligence*.

However, it is fair to note that despite describing the collective behaviour of decentralized and self-organized systems of both biological and artificial beings, the term SI is mainly used within the frame of optimization and AI. SI commonly refers to (...) *any attempt to design algorithms and distributed problem-solving devices inspired by the collective behaviour of social insect colonies and other animal societies* (Bonabeau, Dorigo, & Theraulaz, 1999). Thus, it comprises a relatively new mainstream in AI –in addition to the symbolic paradigm and ANNs–, where intelligence is viewed as a collective trait.

The same as intelligence, life can be viewed as a collective rather than an individual attribute. In addition, both intelligence and life can be thought of as having degrees rather than in absolute terms. Thus, an individual worker ant possesses a rather negligible degree of intelligence while also not being fully alive, since it is sterile, and the ability to

reproduce is one of the critical properties of a living organism. Instead, the colony as a whole has the ability to produce offspring so as to perpetuate itself, and can therefore be thought of as fully alive. It also exhibits a highly adaptive behaviour, which is a trait of intelligence[1]. These types of *life forms* are typically known as *super-organisms*, and their intelligence is emergent and superior to the aggregation of those of its components.

Arguably, the most successful SI-based optimization methods are Ant Colony Optimization (ACO) and PSO.

4.2.1. Emergence

Emergence is a key concept in many somewhat related research fields such complexity, chaos, cellular automaton, and SI. Strangely enough, there is no general agreement on a concise definition of the concept. Roughly speaking, a property of a system is said to be *emergent* if it rises from lower level interactions of numerous simple elements of the system in a non-obvious manner. In artificial systems, the property is emergent if it is not specifically implemented in a deterministic fashion. The resulting emergent property or behaviour of the system is of considerably higher complexity than those whose interactions generated it. Thus, the collective behaviour of fish schools and bird flocks, as well as the self-organizing patterns in cellular automata, are emergent properties of the systems, but not of their individual components. An emergent property is a characteristic of a group of interacting elements that cannot be inferred by analyzing an individual in isolation. In artificial systems, they cannot be specifically implemented.

4.2.2. Self-organization

Bonabeau et al. (1999) argue that self-organization (SO) is a set of dynamical mechanisms in a system whereby structures appear at the global level from local interactions of its components at a lower level. In other words, SO is the set of mechanisms that lead to the emergent pattern. In self-organizing systems, the internal organization increases automatically without the influence of external elements.

[1] David B. Fogel (1995) defined intelligence as the ability of a (biological or artificial) being to adapt its behaviour to meet its goals in a range of environments.

Kennedy & Eberhart (2001) offer a list of attributes commonly associated with SO:

- Self-organizing systems usually exhibit what appears to be spontaneous order.
- SO can be viewed as a system's incessant attempts to organize itself into ever more complex structures, even in the face of the incessant forces of dissolution described by the second law of thermodynamics.
- The overall system state of a self-organizing system is an emergent property.
- Interconnected system components become organized in a productive or meaningful way based on local information.
- Complex systems can self-organize.
- The SO process works near the *edge of chaos*.

In turn, Bonabeau et al. (1999) state that SO relies on four basic ingredients:

- *Positive feedback*. It is the response of a system when an action that affects it induces it to respond in the same direction of change. For example, the case of the pheromone-laying and pheromone-following behaviour observed in colonies of ants. A positive feedback that is not controlled by a negative feedback may run out of control, leading to the collapse of the system.
- *Negative feedback*. It is the response of the system when the action that acts on it induces it to respond in a reverse direction of change. This is a process that tends to keep things constant, helping to stabilize the emergent collective patterns and to prevent a system from crashing. The exhaustion of food sources, the saturation of available workers, and the pheromone evaporation in ant colonies are good examples of stabilizing negative feedback in biological systems.
- *Fluctuations*. They comprise stochastic processes that introduce innovation, thus enabling a system to explore for new solutions.
- *Multiple interactions*: Like emergence, SO can only be generated among a number of individuals. There is usually a minimum number required for SO to take place.

4.2.3. Division of labour

The Division of Labour (DoL) is a decentralized process by which different individuals take over different tasks in parallel, without any central leader assigning the jobs. This cooperative, self-organizing phenomenon typically occurs in highly structured insect societies such as ant, bee, wasp, or termite colonies. In such developed social structures, different tasks such as guarding the nest, breeding, brood-care, and foraging need to be performed in parallel. This also results in individuals becoming more efficient as they gain experience by repetitively performing the same tasks.

In groups of social animals composed of only one type of individuals, DoL is not a common phenomenon. Conversely, in more structured societies where individuals are not all anatomically equivalent, DoL usually takes place by means of genetic pressures. It is common to find different genetic groups –say queen, foragers, breeders, guardians– which exhibit predisposition to perform the task they are genetically more suitable for. However, the DoL process presents plasticity to the environmental challenges. If some external force changes the situation, and more individuals are required for a given task than there are suitable individuals available, individuals genetically less prepared for the task step up, and adapt their behaviour to the needs of the colony. Thus, the number of individuals required is matched even when some of them are stretching their capabilities. And the most striking part of this is that it all happens without central control.

4.2.4. Stigmergy

This is a mechanism of indirect communication by means of individual alterations to the environment. The modification introduced by an individual stimulates the next action, either by the same or by a different individual. Thus, the environment stands for a work-state memory, so that the work can be continued by any member of the group. The individual work is a behavioural response to the state of the environment, which is in turn modified by such a work. Trail-laying and trail-following in insect colonies is a typical example of this kind of communication.

4.2.5. Biological foundations

The SI discipline found its inspiration on the observation of the cooperative behaviour and resulting achievements of animal societies composed of relatively non-intelligent individuals, in particular –but not exclusively– social insects.

In nature there is always a trade-off between the advantages of gathering together and cooperate to achieve greater results, and the competition for the available resources when they are not plentiful. While EAs rely on competition and survival of the fittest to improve the average performance of the population, PSO relies on cooperation between the so-called particles. Therefore, in order to illustrate the advantages of cooperation in social organisms in nature, the behaviour of a few social biological organisms that influenced the development of the SI field is briefly presented hereafter.

It is important to remark that the behaviour of social animals, whether consciously or not, is not necessarily individualistic. One example is that of a specialist ant taking on a different job it is not genetically prepared for, seeking the benefits of the colony rather than its own. Another example is that of noisy group vocalizations to inform the members of a group about the size of its population so as to regulate the rate of reproduction and avoid overpopulation and shortness of resources, which goes against the individualistic animal instinct compelling them to pass their genes to the next generation. Another example widespread in the animal kingdom is parents ready to risk their lives for their offspring, against the survival instinct. Yet another illustrative example is that of several prays that move in herds giving a warning call in the presence of a predator, thus becoming more vulnerable. Or during reproduction, when the mother becomes an easier pray or a handicapped predator during pregnancy. These and other examples of natural social behaviour show that social beings may renounce to some extent, consciously or not, to their individual convenience to seek the benefit of the group.

4.2.5.1. Amoebas

Perhaps the simplest social behaviour observed in the animal kingdom is that of a group of amoebas that self-organize to optimize their individual probability of survival, as discussed in (Kennedy & Eberhart, 2001). The amoeba is a single-celled organism that moves by alternating softening and hardening of the protoplasm, feeds on bacteria, and

reproduces by cell-division. When food becomes scarce, the amoebas that cannot find nutrients start emitting a chemical substance. In turn, the amoebas that detect the presence of this substance start emitting the substance themselves, while moving towards the areas where the concentration of the substance is higher. When they meet other amoebas, they merge with them, eventually forming an aggregate single organism that can crawl around. This organism produces reproductive spores, which are released when a more favourable location is found. Some spores eventually become amoebas, which would then be situated in a more suitable environment. Thus, amoebas communicate through stigmergy, and switch their behaviour from individual to social and vice versa according to the challenges posed by the environment.

4.2.5.2. Social insects

The case of the amoebas is probably the simplest kind of SI-based behaviour, where SO of individuals of the same type takes place under environmental pressure. There are also cases of insects in nature that are typically solitary insects until the environment occasionally forces them together, and trigger a *gregarious* phase that leads to group formations and drastic change of behaviour. Such is the case of the locusts, which under those conditions aggregate to produce massive collective migrations. The phenomenon is not easy to predict, but it is associated to the environment pushing together these insects that, if possible, tend to avoid each other.

SO and SI are more frequent and most impressive in highly structured social groups such as bee, ant, wasp, or termite colonies. In these cases, there are groups of anatomically different individuals, which are genetically predisposed to perform certain tasks. Thus, the DoL is influenced by both the needs of the colony and the genetic structure of the insects. Typical tasks within a colony that are performed by genetically different individuals involve breeding, protecting the nest, and searching for food. If external factors make it necessary, the DoL adapts to the needs to the swarm.

Despite the only few hundred brain cells that an ant is equipped with, and the individual rather random behaviour it exhibits, when interacting with other ants by means of a set of simple rules, their self-organizing emergent behaviour somehow enables them to build complex nest structures and near-optimal networks of highways connecting food

sources to the nest. Some termites in the north-east of Argentina are able to build domed structures simply by following a small and simple set of rules that may be as follows:

1. Take some dirt in your mouth and moisten it.
2. Follow the strongest pheromone trail while depositing pheromone as you move.
3. Deposit the moistened dirt where the smell is strongest.

Since termites, like ants, individually display almost random behaviour, the first movements would seem random until a number of pillars are initiated. This process presents positive feedback effect, since the pillars are placed where the pheromone concentration is higher, thus becoming more powerful attractors as the pillars grow. Since the termites are attracted by several pillars, they frequently end up in a critical point where the attraction to either one or the other of two opposing pillars is equally strong, making a random choice. Thus, the termites tend to approach both pillars from the sides that face one another, resulting in more dirt being deposited on those opposite sides of the pillars. As the pillars ascend, they tend to get closer to one another, eventually meeting and forming an arch. When several pillars meet, the dome is formed. Since the pheromone trails evaporates, a certain minimum number of termites involved in laying a trail is required for the positive feedback effect to take place. This prevents the formation of a great number of pillars, as well as the work on abandoned trails.

Like termites building a dome, foraging ants communicate through stigmergy. They initially leave the nest in a random search for food, depositing pheromones as they move. If they find pheromone trails, they follow the one with the highest concentration while laying their own pheromones. Thus, this trail-laying and trail-following behaviour allows finding the shortest path from the nest to a food source, as the concentration of pheromones would be higher for shorter paths. There is always a small probability that an ant loses a trail, which allows for further exploration.

The other type of communication displayed by social insects is a direct, individual-to-individual communication, such as the dancing of a worker bee when it finds a food source. The complexity of the dance is such that it encodes the abundance, the distance, and the direction of the food source so as to induce potential recruits to leave the colony and exploit the food source advertised. The whole recruitment system self-tunes to control the degrees of exploration and exploitation of known foraging sites.

Notice the important difference between SO in bee and ant colonies, and those in amoebas and locusts. The former are highly structured, and individuals need to self-organize as they would not survive otherwise. Individuals in the group are not all genetically the same, and there typically exists a DoL. Amoeba and locusts are rather solitary beings which self-organize into social structures under environmental pressure only, and individuals are all of the same type. For further reading, refer to (Bonabeau, Dorigo, & Theraulaz, 1999) and (Beekman, Sword, & Simpson, 2008).

4.2.5.3. Fish schools and bird flocks

Some kinds of fish schools and bird flocks orderly move about in a rather majestic fashion, as opposed to the swarming behaviour observed in insects. For instance, if a predator approaches a fish school, those who first notice the threat change direction. Suddenly, every fish in the school changes direction almost at the same time, so as to match their neighbours' new velocities. Some models of this behaviour have been proposed, suggesting that a single fish is attracted to a school, and that the attraction increases (while the rate of increase decreases) with the size of the school.

Bird flocks behave in a similar fashion, for which many different models have been proposed. A well-known simulation of bird flocks was developed by Reynolds (1987), who proposed three basic rules for each so-called *boid* (artificial bird) to follow:

- Pull away before crashing into another bird (*collision avoidance*).
- Try to match your neighbours' velocities (*velocity matching*).
- Try to move towards the centre of the flock (*flock centering*).

The implementation of these rules resulted in realistic flock-like behaviour. Although the rules are entirely artificial, it is evident that natural animals try to avoid collision, while it is believed that a bird within a flock tries to keep the same distance to all its neighbours. Furthermore, it has been noticed that this kind of behaviour is more frequent in preys than in predators, and since a fish at the edge of the school is more likely to be caught, it is only natural that it would try to move towards the centre of the group.

Another influential simulation of bird flocks is that of Heppner & Grenander (1990), who observed that natural bird flocks do not have a leader, and therefore there is no cen-

tral control. They implemented a simulation similar to that of Reynolds, but now the birds are also attracted to a roost, and an occasional random force was implemented seldom deflecting the birds' direction, just like a gust of wind would do. The intensity of the attraction was programmed to increase with the decrease of the distance to the roost. The result was a realistic flock-like choreography. To summarize, the artificial birds in (Heppner & Grenander, 1990):

- are attracted towards one another unless they get too close;
- if they are too close, they repel;
- have the tendency to keep a target velocity;
- are occasionally knocked off by a random force resembling a gust of wind;
- are attracted to a roost, whose position is known, where the closer they get to the roost the stronger the attraction.

It is important to note that these simulations of bird flocks were especially influential in the development of the PSO basic algorithm in (Kennedy & Eberhart, 1995).

4.2.5.4. Human beings

We humans are the most social of animals: we live together in families, tribes, cities, nations, behaving and thinking according to the rules and norms of our communities, adopting the customs of our fellows, including the facts they believe and the explanations they use to tie those facts together. Even when we are alone, we think about other people, and even when we think about inanimate things, we think using language—the medium of interpersonal communication. (Kennedy & Eberhart, 2001, p. xiii)

Despite possessing a very powerful brain, a single human being that is born and left alone in the world cannot learn much of it in his whole lifetime. The achievements of humanity by far exceed those of even the greatest human minds. There are, nevertheless, important differences with respect to the other social animals. The higher complexity and capabilities of the human mind result in the existence of individual objectives that distort the systems discussed before. In addition, humans learn from experience and observation, and change their behaviour accordingly. Finally, there are different levels of social influence in addition to the direct contact with their peers. Namely, the behav-

iour of a human being is also affected by social rules and by culture, which is a kind of storage of previous successful experiences of other individuals with whom there has never been any kind of direct interaction. Nevertheless, although individual intelligence may not be fully disregarded in human beings, it is still in a different scale when compared to the emergent capabilities of social groups, societies, and humanity.

4.2.5.5. Swarm robots

In the early days of robotics, the symbolic paradigm dominated the approaches to AI, so that the robot minds were typically implemented in a central executive processor. Hence the systems required inference engines to respond to the inputs received by the robot's sensors. The problem was that, for increasingly complex environments the system required more symbols and the inference engine more and more complex chains of logic. As a result, the robot would need a long time to decide upon a response to the stimuli.

The advent of multi-robot systems allowed working on robotics without a central controller. Brooks (quoted in (Kennedy & Eberhart, 2001)) proposed simple, independent, specialized robots, each of which would perform the tasks it was specifically programmed to. While this eliminates the central control, each robot is still an autonomous agent whose behaviour is exactly as deterministically programmed to be. The behaviour of a so-called multi-agent system is more or less the sum of each agent's contribution. Conversely, the application of SI to multi-robot systems gives birth to *Swarm Robotics*, where every robot is the same, and the achievements of the system go way beyond the aggregation of the contributions of its simple parts.

The numerous terms arising from the different approaches in multi-robot systems makes a concise definition of swarm robotics necessary, so as to differentiate it from other multi-robot approaches such as *distributed robotics*, *collective robotics*, or *cellular robotics*. Merely having multiple interacting robots does not necessarily imply SO, emergence, or SI.

Swarm Robotics is the study of how a large number of relatively simple physically embodied agents can be designed such that a desired collective behaviour emerges from the local interactions among the agents and between the agents and the environment. (Sahin E. , 2005)

Broadly speaking, swarm robotic systems should exhibit three properties:

Robustness. The system should maintain its operability even in the case of disturbances from the environment or malfunction (or loss) of some individuals. Of course, some minimum number of units must remain operative.

Flexibility. The individuals of the swarm must be able to coordinate their behaviours to cope with different tasks. The cooperative behaviour should be adaptive.

Scalability. The system must be able to operate for a wide range of increasing swarm-sizes without excessive impact on its performance.

Note that these are also properties of biological swarms and of self-organizing and SI-based systems. For a concise review, refer to (Sahin, Girgin, Bayindir, & Turgut, 2008).

4.2.6. Social learning

It has been argued before that the social behaviour of human beings is more complex than those observed in the other social beings discussed earlier in this chapter. Some of the studies and experiments in social psychology that somehow influenced the development of the PSO paradigm are discussed hereafter in this section.

In 1936, Sherif[2] reported experiments demonstrating the convergence of people's perceptions. He placed subjects in a dark room with a stationary spot of light projected on a wall. When asked in isolation, the individuals tended to report that the spot had been moving[3], although the range of the movement reported varied from person to person. However, when they were asked to make the report in public, the reports tended to converge. In 1956, Asch[4] reported that when subjects in an experiment were faced with the dilemma of giving the obvious true answer versus agreeing with the group, about a third of them chose to agree with the group in spite of knowing that the answer was plainly wrong. In 1965, Bandura[5] announced the discovery of the 'no-trial learning', arguing that humans can learn a task without even trying it, by observing somebody else doing it with successful results. Note that the tendency to seek agreement manifested in Sherif's experiments, the conformism observed in Asch's experiments, and Bandura's social

[2] Quoted in (Kennedy & Eberhart, Swarm Intelligence, 2001).
[3] This is due to the 'autokinetic effect'. Refer to (Kennedy & Eberhart, 2001, p. 202).
[4] Quoted in (Kennedy & Eberhart, Swarm Intelligence, 2001).
[5] Quoted in (Kennedy & Eberhart, Swarm Intelligence, 2001).

learning, all support the belief that whenever people interact, they become more similar to one another. This is the key concept underlying some models of social behaviour such us 'Axelrod's Culture Model' and the PSO paradigm.

Latané[6] suggested in his 'social impact theory' that the influence of a group of people over an individual is a function of the strength, the immediacy, and the number of people in the group. The strength is just a kind of social persuasiveness, and the immediacy is inversely proportional to the distance. The influence increases –although the rate of increase decreases– with the number of individuals in the group.

For further reading on the aspects of, and studies in social psychology that influenced the development of the PSO paradigm, refer to (Kennedy & Eberhart, 2001). For further reading on SI, refer, for instance, to (Bonabeau, Dorigo, & Theraulaz, 1999), (Kennedy & Eberhart, 2001), (Blum & Li, 2008), (Sahin, Girgin, Bayindir, & Turgut, 2008), and (Trianni, Nolfi, & Dorigo, 2008).

4.3. Ant Colony Optimization

The Ant Colony Optimization (ACO) method was proposed by Marco Dorigo (1992) in his Ph.D. thesis. The original algorithm was inspired by the foraging behaviour observed in some colonies of Argentinian ants, which were able to find the nearly closest path from their nest to a food source without any previous knowledge. The basic foraging mechanism –already introduced in section 4.2.5.2– may be summarized as follows:

- Without relying on previous knowledge, each ant leaves the nest and starts an independent random walk, while depositing a trail of pheromones.
- If an ant detects a pheromone trail, it follows the smell.
- If more than one trail is detected, the probabilistic choice to follow the one with the higher concentration is made. The stronger the scent the higher the probability.
- While no pheromone is found, the choice is random.
- Since choices are probabilistic, it is always possible that an ant loses a trail, in which case it explores for new alternative paths.

[6] Quoted in (Kennedy & Eberhart, Swarm Intelligence, 2001).

- Once an ant finds a food source, it returns to the nest, also laying a trail of pheromones on the journey back. The richer the food source the higher the concentration of pheromones deposited.
- The shorter the path between the nest and the source, the higher the amount of pheromones that accumulates in a given period of time. Thus, the paths followed by every ant tend to converge to the shortest path.
- Since pheromones evaporate, paths which are not popular are lost.

The ant behaviour is governed by the needs of the colony rather than by the goals of the individual. Thus, the emergent global behaviour of the colony allows the latter to find the shortest path between the nest and the food source, while a single isolated ant following these simple mechanisms would simply die off.

The Travelling Salesman Problem (TSP) consists of finding the shortest tour visiting every other city only once, and returning to the initial city. It can be represented by a completely connected undirected graph of nodes (cities) and edge weights (distances between cities). Considering that the problem can be viewed as a finite set of paths, it can be handled directly by mimicking the ants foraging behaviour. Hence the original ACO algorithm –the 'Ant System'– was developed to solve the TSP and any other problem that can be formulated as the search for shortest paths through graphs.

4.3.1. The Ant System

A graph representing a four-city symmetric TSP is offered in Fig. 4.1. Since the problem represented is symmetric, the distance between cities *i* and *j* is the same as the one between *j* and *i*. That is, $\delta_{ij} = \delta_{ji}$. The total cost or distance of the tour is the aggregation of all the partial distances. Variations of the problem include asymmetries. In order to solve this problem using an Ant System (AS) algorithm, there must be some concentration of pheromones associated to every edge throughout the whole search. The pheromone concentration between two edges *i* and *j* is also symmetric: $\tau_{ij} = \tau_{ji}$. The basic algorithm works by randomly initializing a colony of ants, followed by every ant completing a tour by choosing the cities according to a 'state transition rule'. Once all tours are completed, the pheromone trails are updated according to a 'global updating rule'.

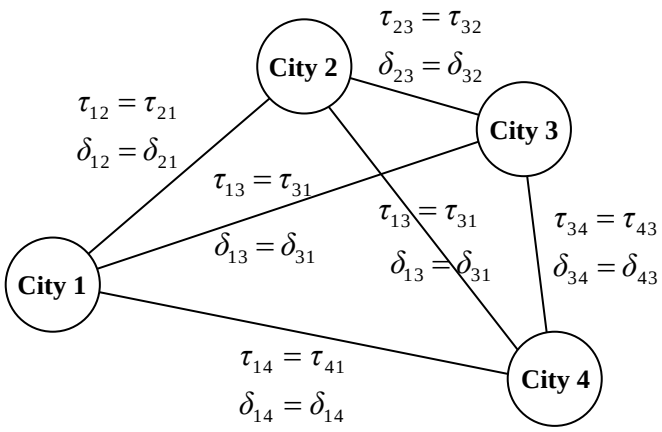


Fig. 4.1. Graph representing a four-city symmetric TSP to be solved by an ant algorithm.

4.3.1.1. The state transition rule

As mentioned before, an ant in one city chooses the next one in its tour by making a choice that probabilistically favours the edges with higher concentrations of pheromones. Note that pheromones are updated after the tour is completed, so that higher pheromones in one edge are associated with the edge being part of some of the shortest tours, but not necessarily the shortest distance to the next city.

Given that cities can only be visited once, there is a list of feasible (or of banned) cities for the next move. The calculation of the probability $P_{kij}^{(t)}$ with which the ant k in city i chooses to move to city j at time t is given by the state transition rule in Eq. (4.1).

$$P_{kij}^{(t)} = \begin{cases} \dfrac{\tau_{ij}^{(t)} \cdot \left[\eta_{ij}\right]^{\beta}}{\sum_{s \in \mathcal{N}_{ki}} \tau_{is}^{(t)} \cdot \left[\eta_{is}\right]^{\beta}} & \text{if } j \in \mathcal{N}_{ki} \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

where:

$\tau_{ij}^{(t)}$: Pheromone concentration in the edge linking cities i and j at time step t.

$\eta_{ij} = \dfrac{1}{\delta_{ij}}$: Inverse of distance between cities i and j.

$\beta > 0$: Parameter setting importance of pheromone in relation to distance.

$\mathcal{N}_{ki}$: Set of feasible cities that remain to be visited by ant *k* located at city *i*.

A small amount of pheromone is assigned to every edge at the beginning. As can be observed, Eq. (4.1) favours both shorter paths and higher concentrations of pheromone.

4.3.1.2. The global updating rule

Once every ant in the swarm has built up its own tour, the concentration of pheromone is updated on all edges according to the global updating rule shown in Eq. (4.2):

$$\tau_{ij}^{(t+1)} = (1-\alpha)\cdot\tau_{ij}^{(t)} + \sum_{k=1}^{m}\Delta\tau_{kij}^{(t)}$$
$$\Delta\tau_{kij}^{(t)} = \begin{cases} \dfrac{1}{L_k^{(t)}} & \text{if } l_{ij} \in L_k^{(t)} \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

where:

$0<\alpha<1$: Pheromone decay parameter (evaporation).

$L_k^{(t)}$: Length of the complete tour performed by ant *k*.

l_{ij} : Length of the edge that joins the cities *i* and *j*.

m : Number of ants in the colony.

Thus, Eq. (4.2) updates the concentration of pheromone taking into account both the pheromone-laying and the pheromone evaporation phenomena.

4.3.2. The Ant Colony System

Dorigo & Gambardella (1997) reported that the AS becomes too slow for more than 30 cities. Therefore they proposed a modified version called the ant colony system (ACS). The latter differs from the AS in the implementations of the 'state transition rule' and of the 'global updating rule', and in that it incorporates a 'local updating rule' as well.

4.3.2.1. The state transition rule

The probabilities for the next move in the ACS are calculated as in Eq. (4.3), which offers a balance between exploitation and a biased exploration of new solutions.

$$\text{if } U_{(0,1)} \le q_0: \quad j = \arg\max_{u \in \mathcal{N}_{ki}} \left\{ \tau_{iu}^{(t)} \cdot \left[\eta_{iu}\right]^{\beta} \right\}$$

$$\text{otherwise:} \quad \begin{cases} P_{kij}^{(t)} = \dfrac{\tau_{ij}^{(t)} \cdot \left[\eta_{ij}\right]^{\beta}}{\sum_{s \in \mathcal{N}_{ki}} \tau_{is}^{(t)} \cdot \left[\eta_{is}\right]^{\beta}} & \text{if } j \in \mathcal{N}_{ki} \\ P_{kij}^{(t)} = 0 & \text{otherwise} \end{cases} \tag{4.3}$$

where:

$U_{(0,1)}$: Pseudo-random number generated from a uniform distribution within [0,1], resampled anew each time it is referenced

$0 < q_0 < 1$: Parameter of the system that regulates the relative importance of exploitation $(q \le q_0)$ versus a biased exploration $(q > q_0)$.

$\tau_{ij}^{(t)}$: Pheromone concentration in the edge linking cities *i* and *j* at time-step *t*.

$\beta > 0$: Parameter setting importance of pheromone in relation to distance.

$\eta_{ij} = \dfrac{1}{\delta_{ij}}$: Inverse of the distance between cities *i* and *j*.

$\mathcal{N}_{ki}$: Set of feasible cities that remain to be visited by ant *k* located in city *i*.

If $q_0 = 1$, the very best valid edge is chosen deterministically. If $q_0 = 0$, the state transition rule becomes the same as for the AS in Eq. (4.1).

This new 'state transition rule' allows a better exploitation of the promising areas of the graph, speeding up convergence –of course, at the cost of lower exploration–.

4.3.2.2. The global updating rule

As opposed to the AS, the global updating rule in the ACS only updates the concentration of pheromone on the edges corresponding to the best tour found so far by any ant. That is, from the first up to the current time-step. Making use of the evolutionary computation jargon, a '*non-elitist*' alternative consists of updating the level of pheromone on the best tour within the current iteration rather than on the best tour found so far. The latter would release the 'convergence pressure' thus enhancing exploration. Still, this update takes place once every ant has completed its tour.

Notice that this 'global updating rule' may increase the pheromone concentration of a tour at a given time-step even when no ant walked that path. The modified 'state transi-

tion rule' and 'global updating rule' increase the convergence pressure, which guide the search towards the neighbourhood of the best tour found so far faster.

$$\tau_{ij}^{(t+1)} = (1-\alpha)\cdot\tau_{ij}^{(t)} + \alpha\cdot\Delta\tau_{ij}^{(t)}$$
$$\Delta\tau_{ij}^{(t)} = \begin{cases} \dfrac{1}{L_{gb}} & \text{if } l_{ij} \in L_{gb} \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

where:

$0 < \alpha < 1$: Pheromone regulation parameter (no longer just for the decay).

L_{gb} : Length of the complete globally best tour found so far by any ant.

l_{ij} : Length of the edge that joins the cities *i* and *j*.

4.3.2.3. The local updating rule

(…) *The effect of local updating is to make the desirability of the edges change dynamically* (…) *without local-updating all ants would search in a narrow neighbourhood of the best previous tour.* (Dorigo & Gambardella, 1997)

The local updating rule presents some more resemblance to what actually happens with real ants: the pheromone trail is updated every time an ant passes through. Thus each ant influences the ants that walk the same edge '*later on*' within the same time-step. The local updating rule of the ACS is as given in Eq. (4.5):

$$\tau_{ij}^{(t)} = (1-\rho)\cdot\tau_{ij}^{(t)} + \rho\cdot\Delta\tau$$
$$\Delta\tau = \tau_0 \tag{4.5}$$

where:

$0 < \rho < 1$: Pheromone regulation parameter.

τ_0 : Initial pheromone level.

Two other alternative settings for $\Delta\tau$ were also considered in (Dorigo & Gambardella, 1997) for their experiments. The one in Eq. (4.5) returned one of the two best results.

Today, the family of ACO algorithms includes numerous variants, some seeking performance improvement –which differ mainly in the pheromone updating rules–, and some aiming for different types of combinatorial problems. In fact, there is some recent work on ACO algorithms applied to continuous problems (refer to (Blum & Li, 2008)).

The names given to the different ACO algorithms vary considerably in the literature. They can be referred to as 'ant algorithms'; 'ant colony algorithms'; 'ant-system'; 'ant colony system'; 'ACO meta-heuristic'; etc. Some of these names refer to the same type of algorithms, and some refer to different variants and adaptations. For further reading, see (Dorigo, 1992), (Bonabeau, Dorigo, & Theraulaz, 1999), (Dorigo & Gambardella, 1997), (Dorigo & Stützle, 2004), (Engelbrecht, 2005), (Blum & Li, 2008).

4.4. Particle Swarm Optimization

The PSO method was invented by social-psychologist James Kennedy and electrical-engineer Russell C. Eberhart (1995), inspired by earlier bird flock simulations framed within the field of social psychology. In particular, Reynolds' boids (1987) and Heppner and Grenander's artificial birds (1990) strongly influenced their early developments. Therefore, the method is closely related to other simulations of social processes and experimental studies in social psychology, while also having strong roots in optimization and AI. In the same fashion as ANNs can be viewed as models of the human brain or as general mapping devices; and GAs can be viewed as models of genetic evolution or as optimization algorithms; the PSO paradigm can be thought of either as a model of social behaviour or as a problem-solving technique. Its applications, nevertheless, are mainly as an optimization method.

4.4.1. Social psychology viewpoint

From the 'social psychology' point of view, the method performs some simulation of social behaviour within some social network. The most successful individuals become dynamic leaders, while the less successful ones become followers. Nonetheless, roles may switch at any time so that the system is not centralized.

As briefly discussed in section 4.2.6, some of the experimental studies in social psychology that influenced the development of the method are Lewin's field theory; Gestalt theory; Sherif's and Asch's experiments; Latané's social impact theory; Bandura's no-trial learning; simulations of the spread of culture in a population; and simulations of the behaviour of social animals such as bird flocks and social insects. For further reading, refer to (Kennedy & Eberhart, 2001, pp. 187-284).

4.4.2. Optimization viewpoint

From the 'optimization' perspective, PSO is a gradient-free search method suitable for optimization problems whose solutions can be represented as points in an *n*-dimensional space. While variables need to be real-valued in its original version, binary and other discrete versions of the method have also been proposed. Refer, for instance, to (Kennedy & Eberhart, 1997), (Kennedy & Eberhart, 2001, pp. 289-299), (Mohan & Al-Kazemi, 2001), and (Clerc, 2004).

Since the method is not designed to optimize but to carry out procedures that are not directly related to the optimization problem, it is frequently referred to as a 'modern heuristics'. Optimization occurs, nevertheless, without obvious links between the implemented technique and the resulting optimization process. It seems clear that the inspiration of the method was also strongly influenced by the earlier EAs, with which this method –and many others– shares numerous characteristics.

Gradient information is not required, which enables the method to deal with non-differentiable and even discontinuous problems. Therefore there is no restriction to the characteristics of the objective function for the approach to be applicable. In fact, the function does not even need to be explicit.

4.4.3. Artificial Intelligence viewpoint

From the AI point of view, the PSO paradigm belongs to different branches such as *Artificial Life* (AL), *Computational Intelligence* (CI), and *Swarm Intelligence* (SI). The latter is concerned with the study of the collective behaviour that emerges from decentralized and self-organized systems. Thus, PSO is a bottom-up SI-based approach be-

cause its ability to optimize is not specifically implemented in the code but emerges from local interactions among its individual parts and with the environment, which occurs without a sense of purpose or central control. The system's intelligent behaviour emerges in a higher level than the individuals', evolving solutions without using the programmers' expertise on the subject matter. Hence the problem *per se* is not solved but AI entities are programmed, which are expected to find a solution themselves.

4.4.4. General viewpoint

Either a 'modern heuristics' or a 'SI-based optimizer', the PSO algorithm is not designed to optimize but to perform a sort of simulation of a social milieu, where the ability of the population (*swarm*) to optimize its performance emerges from the cooperation among individuals (*particles*). While this makes it difficult to understand how optimization is actually performed[7], it shows astonishing robustness in handling many kinds of complex problems that it was not specifically designed for. It has the disadvantage that its theoretical bases are very difficult to be understood deterministically. Nevertheless, considerable theoretical work has been carried out on simplified versions of the algorithm, extrapolated to the full version, and supported by experimental results. Refer, for instance, to (Ozcan & Mohan, 1998);(Ozcan & Mohan, 1999); (Clerc, 1999); (van den Bergh, 2001); (Clerc & Kennedy, 2002); (Trelea, 2003); (Clerc, 2004); (Mendes, Kennedy, & Neves, 2004); (Clerc, 2006a); (Clerc, 2006b); (Jiang, Luo, & Yang, 2007);(Clerc, 2008a); (Clerc, 2008b); (Helwig & Wanka, 2008); (Kennedy, 2008); (Poli, 2008); and chapters 5 and 6 of this thesis. For a comprehensive review of the method, the reader may refer to (Kennedy & Eberhart, 2001), (Engelbrecht, 2005), (Clerc, 2006a), and (Bratton & Kennedy, 2007).

The function to be minimized is commonly called 'fitness function' because of the influence of the EAs. It is more appropriately referred to as the 'conflict function' from here forth due to the social-psychology metaphor that inspired the method. That is, each individual searches the space of beliefs, seeking the minimization of the conflicts among the beliefs it holds by using the information gathered by both its own experience and those of others. Individuals seek agreement by clustering in the space of beliefs,

[7] Refer to (Kennedy, 2008) and (Clerc, 2008) for advanced reading on the subject.

which is −broadly speaking− the result of individuals imitating their most successful peers, thus becoming more similar to one another as the search progresses. The clustering is delayed by their own previous successful experiences, which each individual is reluctant to disregard.

4.4.5. Basic algorithm

While the emergent optimization properties of the PSO algorithm result from local interactions among particles in a swarm, the behaviour of a single particle can be summarized in three sequential processes:

Evaluation. A particle evaluates its position in the environment, given by the associated value of the conflict function. Following the social psychology metaphor, this stands for the conflict among its current set of beliefs.

Comparison. Once the particle's position in the environment is evaluated, it is not straightforward to tell how good it is. Experiments and theories in social psychology suggest that humans judge themselves by comparing to others, thus telling better from worse rather than good from bad. Thus, the particle compares the conflict among its current set of beliefs to those of its neighbours.

Imitation. The particle imitates those whose performances are desirable or superior in some sense. In the basic PSO algorithm, only the most successful neighbour is imitated.

These three processes are implemented within PSO, where the only sign of individual intelligence is a small memory. The basic update equations are as follows:

$$\begin{cases} v_{ij}^{(t)} = w \cdot v_{ij}^{(t-1)} + iw \cdot U_{(0,1)} \cdot \left(pbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) + sw \cdot U_{(0,1)} \cdot \left(lbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) \\ x_{ij}^{(t)} = x_{ij}^{(t-1)} + v_{ij}^{(t)} \end{cases} \tag{4.6}$$

where:

$v_{ij}^{(t)}$: j^{th} coordinate of the velocity of particle i at time-step t.

$x_{ij}^{(t)}$: j^{th} coordinate of the position of particle i at time-step t.

$U_{(0,1)}$: Random number from a uniform distribution in the range [0,1] resampled anew every time it is referenced.

w, iw, sw : Inertia, individuality, and sociality weights, respectively.

$pbest_{ij}^{(t)}$: j^{th} coordinate of the best position found by particle i by time-step t.

$lbest_{ij}^{(t)}$: j^{th} coordinate of the best position found by any particle in the neighbourhood of particle i by time-step t.

Thus, the performance of a particle in its current position is evaluated in terms of the *conflict function*. In order to decide upon its next position, the particle compares its current conflict to those associated with its own and with its neighbours' best experiences. Finally, the particle imitates its own best experience and the best experience in its neighbourhood to some extent. The general flow-chart is shown in Fig. 4.2.

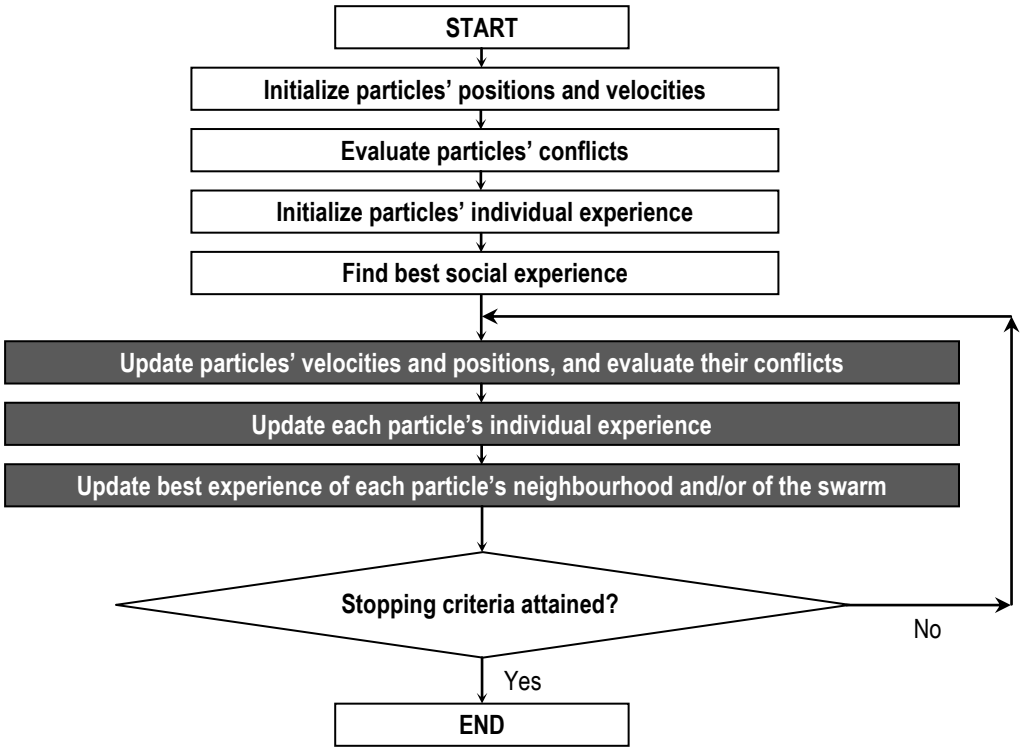


Fig. 4.2. General flowchart for the canonical PSO algorithm.

As shown in Eq. (4.6), there are three coefficients that rule the dynamics of the swarm: the inertia (w), the individuality (iw), and the sociality (sw) weights, where iw and sw are also referred to as learning weights and their aggregation the acceleration weight (aw). The relative importance given to iw and sw leads to more self-confident or more conformist behaviour, while the random weights introduce creativity: since they are re-sampled anew for every time-step, particle, dimension, and term in Eq. (4.6), the particles display uneven trajectories that allow for better exploration. In addition, resampling

them anew for the individuality and the sociality terms (together with $iw = sw$) leads to the particles alternating self-confident and conformist behaviour. Every particle also tends to keep its current velocity, where the strength of this tendency is governed by *w*. The relative importance between *w* and *aw* results –broadly speaking– in more explorative or more exploitative behaviour of the swarm. Different settings of the coefficients in Eq. (4.6) notably affect the behaviour of the swarm.

Regarding the name chosen for the paradigm, Kennedy & Eberhart (1995) argue that, although each member of the population is mass-less and volume-less –typical characteristics of a point–, the velocity and its acceleration are more appropriately applied to a **particle**. As to the term **swarm**, the emergent behaviour resembles a **swarm** rather than the original bird flock inspiration. In addition, they claim that the method adheres to the five principles of **swarm intelligence** (SI) articulated by Millonas[8]:

1. *The population should be able to carry out simple space and time computations.*
2. *The population should be able to respond to quality factors in the environment.*
3. *The population should not commit its activities along excessively narrow channels.*
4. *The population should not change its mode of behaviour every time the environment changes.*
5. *The population must be able to change behaviour mode when it's worth the computational price.*

4.4.6. Coefficients' settings

In the original algorithm, Kennedy & Eberhart (1995) did not include the inertia weight, and suggested setting $iw = sw = 2$. However, the particles tended to diverge rather than cluster, performing the so-called *explosion*. It was found that if the components of the particles' velocities were bounded as in Eq. (4.7), the *explosion* could be controlled.

$$\text{if } \operatorname{abs}\left(v_{ij}^{(t)}\right) > v_{\max\, j} > 0 \quad \Rightarrow \quad v_{ij}^{(t)} = \operatorname{sign}\left(v_{ij}^{(t)}\right) \cdot v_{\max\, j} \tag{4.7}$$

Later, Shi & Eberhart (1998a) proposed the incorporation of the inertia weight (*w*) to control the explosion, while Clerc & Kennedy (2002) studied the trajectory of a simplified one-particle system with stationary attractors and no random weights, developing a constriction factor (χ) to both control the explosion and ensure convergence.

[8] Quoted in (Kennedy & Eberhart, Particle Swarm Optimization, 1995)

4.4.6.1. Surfing the waves

Ozcan & Mohan (1998), (1999) carried out the first attempt to theoretically formalize the dynamic behaviour of the original PSO system (w = 1), although James Kennedy may have informally presented a similar analysis previously, in 1998. They simplified the algorithm by considering a single deterministic particle on a one-dimensional space, pulled by a stationary attractor p. Hence the attractor is as in Eq. (4.8):

$$p \equiv pbest \equiv gbest \tag{4.8}$$

Notice that there are no sub-indices and the font is italic because there is only one particle and one dimension. Thus, calling the acceleration coefficient

$$\phi = \phi_1 + \phi_2, \tag{4.9}$$

the velocity and position update equations become as in Eq. (4.10):

$$\begin{cases} v^{(t)} = v^{(t-1)} + \phi \cdot \left(p - x^{(t-1)} \right) \\ x^{(t)} = x^{(t-1)} + v^{(t)} \end{cases} \tag{4.10}$$

Ozcan & Mohan (1998) assumed that p and ϕ were constants, and the initial conditions shown in Eq. (4.11).

$$\begin{aligned} v^{(t=0)} &= v^{(0)} \\ x^{(t=0)} &= x^{(0)} \end{aligned}, \tag{4.11}$$

Thus, the recurrence relation in Eq. (4.12) was obtained, which they solved using generating functions to yield the closed form of the particle's position shown in Eq. (4.13).

$$x^{(t)} = \left(2 - \phi\right) \cdot x^{(t-1)} - x^{(t-2)} + \phi \cdot p \tag{4.12}$$

$$x^{(t)} = \alpha \cdot \left[\frac{2 - \phi + \delta}{2} \right]^t + \beta \cdot \left[\frac{2 - \phi - \delta}{2} \right]^t + p \tag{4.13}$$

where

$$\begin{aligned}\delta &= \sqrt{\phi^2 - 4\cdot\phi} \\ \beta &= \frac{\left(x^{(0)} - p\right)\cdot\left(\delta+\phi\right)}{2\cdot\delta} - \frac{v^{(0)}}{\delta} \\ \alpha &= x^{(0)} - p - \beta\end{aligned} \tag{4.14}$$

Ozcan & Mohan (1998) studied the trajectory of the particle governed by Eqs. (4.13) and (4.14) for the particular case where:

$$x^{(0)} = p \tag{4.15}$$

That is, the particle has already visited the attractor. Under those assumptions, they analyzed the trajectory for a few particular values of ϕ that comprise boundary cases.

Real δ

δ is real for $\phi = 0$ and for $\phi \geq 4$ (of course, $\phi < 0$ is irrelevant). The case of $\phi = 0$ is also of no interest, as the particle is not pulled by the attractor and hence moves in a straight line with the initial velocity. For $\phi \geq 4$, the trajectory of the particle is oscillatory with increasing amplitudes, bounded by exponential functions, as described by Eq. (4.13).

Complex δ

δ is a complex number for $0 < \phi < 4$. Again, for the particular case of Eq. (4.15), Ozcan & Mohan (1998) presented the trajectory equation for complex δ as in Eq.(4.16):

$$x^{(t)} = \frac{2\cdot v^{(0)}}{\|\delta'\|}\cdot\sin\left(\operatorname{atan}\left(\frac{\|\delta'\|}{2-\phi}\right)\cdot t\right) \tag{4.16}$$

This equation implies that the trajectory of a simple particle in complex δ zone is a sinusoidal wave and our choice of parameters determines the amplitude and the frequency of the wave. (Ozcan & Mohan, 1998)

The domain of ϕ was divided in four regions and five cases of interest:

Case 1: $0 < \phi < \left(2-\sqrt{3}\right)$. $\|\delta'\| < 1$, and the amplitude of the sine wave increases as ϕ decreases.

Case 2: $\left(2-\sqrt{3}\right)\leq\phi\leq 2$. $\|\delta'\|$ increases from '1' to '2' as ϕ increases, and hence the amplitude of the sine wave decreases with ϕ.

Case 3: $\phi=2$. $\|\delta'\|=2$, and the trajectory equation becomes $x^{(t)}=v^{(0)}\cdot\sin\left(\frac{\pi}{2}\cdot t\right)$.

Case 4: $2<\phi<\left(2+\sqrt{3}\right)$. $\|\delta'\|$ decreases from '2' to '1' as ϕ increases, and hence the amplitude of the sine wave increases with ϕ.

Case 5: $\left(2+\sqrt{3}\right)<\phi<4$. $\|\delta'\|>1$, and the amplitude of the sine wave increases with ϕ.

Note that their analyses and conclusions are valid under the assumption of $x^{(0)}=p$. The analysis was extended in (Ozcan & Mohan, 1999) to consider the individual and the social attractors separate in the expressions; sub-indices in the equations to account for multi-dimensional spaces; the study of the step sizes; and the trajectory closed form expression for complex δ in polar coordinates –including the cosine term (dropped off under the assumption $x^{(0)}=p$)–. Thus, they argue that the particle does not really fly over the search-space but rather surfs it on sine waves. The particle is then attracted by the weighted average of the two best experiences, moving in step sizes randomly obtained from a sinusoidal wave. According to the different cases analyzed, the type of wave caught would be determined by the random weights used in the random averaging. They also claim that the v_{max} constraint seems to help the particle jump onto another wave.

4.4.6.2. Constriction factor

In a similar fashion as Ozcan & Mohan (1998), (1999), Clerc & Kennedy (2002) analyzed the trajectory of the deterministic particle for the original PSO algorithm. That is, without the inertia weight. In addition, they developed constriction factors aiming to control the explosion and ensure convergence. They built a system of two recurrence relations of first order that describes the simplified system by using the change of variables in Eq. (4.17):

$$y^{(t)}=p-x^{(t)} \tag{4.17}$$

Thus, the system of recurrence relations is as shown in Eq. (4.18):

$$\begin{cases} v^{(t)} = v^{(t-1)} + \phi \cdot y^{(t-1)} \\ y^{(t)} = -v^{(t-1)} + (1-\phi) \cdot y^{(t-1)} \end{cases} \tag{4.18}$$

The system can be expressed in matrix notation as in Eqs. (4.19) and (4.20):

$$P^{(t)} = \begin{bmatrix} v^{(t)} \\ y^{(t)} \end{bmatrix} = M \cdot P^{(t-1)} = M^t \cdot P^{(0)} \tag{4.19}$$

$$M = \begin{pmatrix} 1 & \phi \\ -1 & (1-\phi) \end{pmatrix} \tag{4.20}$$

The eigenvalues of the system matrix *M* are offered in Eq. (4.21).

$$\begin{aligned} e_1 &= 1 - \frac{\phi}{2} + \frac{\sqrt{\phi^2 - 4 \cdot \phi}}{2} = 1 - \frac{\phi}{2} + \frac{\gamma}{2} \\ e_2 &= 1 - \frac{\phi}{2} - \frac{\sqrt{\phi^2 - 4 \cdot \phi}}{2} = 1 - \frac{\phi}{2} - \frac{\gamma}{2} \end{aligned} \tag{4.21}$$

By diagonalizing the matrix of the system *M*, Clerc & Kennedy (2002) showed that the position of a particle at any time-step would depend only on the initial conditions and on its eigenvalues raised to the power of the corresponding time-step.

In a similar fashion as Ozcan & Mohan (1998) solved the recurrence relation, Clerc & Kennedy (2002) considered continuous[9] rather than discrete[10] time, and solved the corresponding second order differential equations. The roots of the characteristic polynomial are the same as the eigenvalues of the system matrix *M* in Eq. (4.21). Hence the analytic representation of the system is as follows:

$$\begin{cases} v^{(t)} = c_1 \cdot e_1^{\,t} + c_2 \cdot e_2^{\,t} \\ y^{(t)} = \dfrac{1}{\phi} \cdot \left(c_1 \cdot e_1^{\,t} \cdot (e_1 - 1) + c_2 \cdot e_2^{\,t} \cdot (e_2 - 1) \right) \end{cases} \tag{4.22}$$

[9] The 'analytic point of view'

[10] The 'algebraic point of view'

where c_1 and c_2 are calculated according to the initial conditions. Since time is continuous here, if at least one root is negative, the two variables of the system become complex for non-integer values of time *t*.

Going back to discrete time, if at least one of the eigenvalues is not smaller than '1', Clerc & Kennedy (2002) proposed to build a surrogate system whose eigenvalues (e'_1 and e'_2) do comply with the convergence condition. For that purpose, they proposed the incorporation of five coefficients to the system, whose values could be chosen so as to ensure convergence. Such a system is shown in Eq. (4.23):

$$\begin{cases} v^{(t+1)} = \alpha \cdot v^{(t)} + \beta \cdot \phi \cdot y^{(t)} \\ y^{(t+1)} = -\gamma \cdot v^{(t)} + (\delta - \eta \cdot \phi) \cdot y^{(t)} \end{cases} \tag{4.23}$$

Hence the system matrix is as in Eq. (4.24):

$$M = \begin{pmatrix} \alpha & \beta \cdot \phi \\ -\gamma & (\delta - \eta \cdot \phi) \end{pmatrix} \tag{4.24}$$

Then, if the system in Eq. (4.18) does not comply with the convergence condition, then constriction coefficients (also factors) are applied as in Eq. (4.25):

$$\begin{aligned} e'_1 &= \chi_1 \cdot e_1 \\ e'_2 &= \chi_2 \cdot e_2 \end{aligned} \tag{4.25}$$

where the eigenvalues of the surrogate system are forced to have a magnitude smaller than '1'. Thus the constriction factors are calculated as the ratio between the eigenvalues of the new system and those of the original system. Refer to (Clerc & Kennedy, 2002) for the details of the calculations.

According to how the added coefficients (α, β, γ, δ, η) are correlated, Clerc & Kennedy (2002) studied different classes (also types) of constriction; namely, *class 1* and its derivations *class 1'* and *class 1"*, and *class 2*. For details on these classes, refer to their original work (Clerc & Kennedy, 2002). The only constriction type considered in this thesis is the *type 1"* constriction because it is the only one that maintains the original, intuitive concept of the velocity as the difference between two consecutive positions.

Constriction *type 1"*

For this constriction type, the added coefficients are correlated as in Eq. (4.26):

$$\alpha = \beta = \gamma = \eta \tag{4.26}$$

For simplicity, Clerc & Kennedy (2002) also suggest

$$\delta = 1 \tag{4.27}$$

Therefore, with χ instead of α, the system matrix is as in Eq. (4.28):

$$M = \begin{pmatrix} \chi & \chi \cdot \phi \\ -\chi & (1 - \chi \cdot \phi) \end{pmatrix} \tag{4.28}$$

From Eq. (4.25),

$$\chi < \min\left(\frac{1}{|e_1|}, \frac{1}{|e_2|} \right) \tag{4.29}$$

Thus, if $\phi > 4$ in the original system, the constriction factor in Eq. (4.30) is applied.

$$\chi = \frac{2 \cdot \kappa}{\left| 2 - \phi - \sqrt{\phi^2 - 4 \cdot \phi} \right|} \qquad 0 < \kappa < 1 \tag{4.30}$$

Translating this into the original PSO equation yields Eq. (4.31):

$$\begin{cases} v_{ij}^{(t)} = \chi \cdot \left(v_{ij}^{(t-1)} + iw \cdot U_{(0,1)} \cdot \left(pbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) + sw \cdot U_{(0,1)} \cdot \left(lbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) \right) \\ x_{ij}^{(t)} = x_{ij}^{(t-1)} + v_{ij}^{(t)} \end{cases} \tag{4.31}$$

Since ϕ is a random variable, it is typically replaced in practice by the acceleration weight[11], which comprises its upper bound (safe side). Thus the constriction factor to be applied in Eq. (4.31) is commonly calculated as shown in Eq. (4.32):

[11] Note that the term 'acceleration coefficient' may refer in the literature to either one of the individuality and sociality weights, whereas 'acceleration weight' in this thesis stands for their aggregation.

$$\chi = \begin{cases} \dfrac{2 \cdot \kappa}{\left| 2 - (iw + sw) - \sqrt{(iw + sw)^2 - 4 \cdot (iw + sw)} \right|} & \text{if } (iw + sw) \geq 4 \\ \kappa & \text{otherwise} \end{cases} \qquad (4.32)$$

$$0 < \kappa < 1$$

4.4.6.3. Inertia weight

Shi & Eberhart (1998a) theorized that the inertia of the particles with respect to the acceleration terms should be adapted to the different problems. Therefore a new parameter called 'inertia weight' (*w*) was introduced into the original PSO algorithm. Today, the most widespread formulation includes the inertia weight in the velocity update. Such a formulation is referred to as the classical or canonical PSO method.

This w plays the role of balancing the global search and local search. It can be a positive constant or even a positive linear or nonlinear function of time. (Shi & Eberhart, 1998a)

Shi & Eberhart (1998a) ran experiments with the canonical PSO algorithm optimizing the 2-dimensional Schaffer f6 function (refer to Appendix 2) within a feasible space delimitated by $[-100,100]^2$. They used a swarm of 20 particles; $v_{max} = 2$; $iw = sw = 2$; and the maximum number of time-steps $t_{max} = 4000$. Keeping these settings fixed, different values of the inertia weights were evaluated. The conclusions were that it is a good idea to choose *w* from the range [0.9,1.2]. To reach this conclusion, they took into account the number of iterations required to find the global optimum, and the number of failures in finding it. It was also reported that a linearly decreasing inertia weight from 1.4 at the beginning to 0 at the 4000th iteration produced no failure, and the average number of iterations required to find the optimum was smaller than when using a fixed inertia weight in the range [0.9,1.2]. Therefore, not only does *w* allow balancing the relative importance between the particles' inertia and acceleration for different problems, but it also enables the system to perform a wider search at the beginning, while gradually narrowing the search as time progresses.

However, the limitation of the maximum velocity acts as a constraint for the exploration abilities of the algorithm: if v_{max} is set too low, the algorithm behaves as a local search

method no matter which w is chosen. Conversely, if v_{max} is set larger, the exploration ability is mainly ruled by the inertia weight.

Note that the terms 'exploration' and 'exploitation' are frequently used here in their general semantic rather than in their technical meaning, as it is also widely used in the literature. For instance, Ben Ghalia (2008) states that (...) *Exploration is the ability of the search algorithm to explore various regions of the search space in order to locate promising good solutions. Exploitation is the ability to conduct a thorough search within a smaller area recognized as promising in finding the optimal solution.* For a formal definition of exploration and exploitation, the reader may refer to (Naudts & Schippers, 1999), although the applicability of those definitions is limited to a specific search algorithm, non-population-based, and with no memory. Differently, Clerc (2008a) proposes a means of quantifying exploitation in PSO as the ratio between the number of particles inside a given local exploitation area (defined in terms of the particles' individual best experiences) and the total number of particles in the swarm.

Since both v_{max} and w control the exploration ability –as well as the explosion– of the algorithm, Shi & Eberhart (1998b) suggest the removal of v_{max}, passing all the control of the global exploration ability to w. Since a larger w leads to better exploration and a smaller w leads to better exploitation, a decreasing w seems to be a reasonable choice.

Thus, Shi & Eberhart (1998b) ran experiments to analyze the performance of the algorithm for different combinations of w and v_{max}. For each setting, the algorithm was run 30 times, optimizing the Schaffer f6 benchmark function. The results showed that for increasing values of v_{max}, decreasing values of w were required to find the global optimum without any failure, and faster. However, the decrease stopped at $w = 0.80$, to the extent that the best w did not change when v_{max} was virtually removed $(v_{max} = x_{max})$. Hence Shi & Eberhart (1998b) suggest that $w = 1$ is a good choice for a small v_{max}, while $w = 0.80$ is appropriate for a large v_{max}. If a convenient setting for v_{max} is not evident, they suggest setting $v_{max} = x_{max}$ and $w = 0.80$ as a starting point. Note that the suggestion of $v_{max} = x_{max}$ assumes symmetric interval constraints. Generalizing, v_{max} should be set to half the feasible interval in each dimension. Shi & Eberhart (1998b)

also tested a time-decreasing inertia weight, from 1 to 0.40 in the first 1500 time steps, keeping it constant for the remainder of the search. This last setting resulted in the best performance with regards to robustness, convergence rate and variance. This last setting is more in line with the current trend to keep the inertia weight smaller than one.

It must be noted, however, that the conclusions in (Shi & Eberhart, 1998a) and in (Shi & Eberhart, 1998b) were derived from the test on a single, 2-dimensional problem.

Shi & Eberhart (1999) continued to experiment with linearly decreasing inertia weights, and compared their results to those previously obtained in (Angeline, 1998), where a comparison between the original PSO and a well-developed EP algorithm was offered. The conclusions were that the results obtained by the PSO algorithm with a linearly decreasing inertia weight from 0.90 to 0.40 were noticeably better than those obtained by the original PSO and by the EP algorithm in (Angeline, 1998). Surprisingly, the algorithm did not appear to be sensitive to the population size. However, although the algorithm displays fast convergence, the linearly time-decreasing inertia weight results in the lack of exploration at the last stages of the search.

4.4.6.4. Other influential work on coefficients settings

Other influential work on the study of the influence of the coefficients in the dynamics of the particle swarm optimization algorithm are those in (van den Bergh, 2001) and in (Trelea, 2003). Both works are highly recommended to the reader.

van den Bergh (2001) took an approach similar to that in (Ozcan & Mohan, 1998), solving the recurrence relation including the inertia weight. They demarcated a convergence region, and carried out a series of experiments to map the values of the roots of the characteristic polynomial with the maximum magnitude.

In turn, Trelea (2003) took an approach more similar to that in (Clerc & Kennedy, 2002) in the sense that he solved a system of two first order linear recurrence relations rather than one second order linear recurrence relation, although the variables involved are not the same. He also added coefficients to every term of the two basic PSO update equations (velocity and position), although the redundant ones were nicely removed afterwards, returning to the canonical PSO equations. These equations maintain the original concept of two consecutive positions differing in the velocity, as opposed to constriction

other than *Type 1"* in (Clerc & Kennedy, 2002). Trelea (2003) thought of the PSO system in terms of the 'dynamic system theory', and argued that the necessary and sufficient condition for the equilibrium point[12] to be stable is that both eigenvalues of the system matrix have a magnitude smaller than '1'. Thus, he obtained a 'convergent domain'; a 'harmonic oscillatory domain'; and a 'zigzagging domain'.

Although derived from different analyses, using different solving techniques, and obtaining different expressions, the convergence regions obtained in (Trelea, 2003) and in the next chapter of this thesis coincide. In turn, the convergent area reported in (van den Bergh, 2001) is included in the former two, only missing a triangle where the inertia weight take on negative values (no practical use after all).

A few authors dared take the next step and incorporate randomness in the theoretical analyses of PSO. While this is beyond the scope of this thesis, the reader is encouraged to refer to (Clerc, 2006b), (Jiang, Luo, & Yang, 2007), and (Poli, 2008).

Adaptive coefficients are not considered in this thesis, and the subject is left for future work. A popular, fully adaptive PSO is the so-called TRIBES in (Clerc, 2006a) and in (Cooren, Clerc, & Siarry, 2009)[13], while Chen et al. (2007) considered an adaptive constriction factor. For additional theoretical work on PSO, refer to (Cui & Zeng, 2004), (Clerc, 2006b), (Kennedy, 2008), (Clerc, 2008b), and (Helwig & Wanka, 2008).

4.4.7. Neighbourhood topology

Kennedy & Eberhart (1995) developed the original PSO considering that each particle is able to interact with any other particle in the swarm, thus forming a fully connected social network. This type of neighbourhood –commonly known as 'global', 'fully connected', or 'star' topology– offers a fast convergence rate at the expense of exploration. All particles are pulled by the same social attractor, so that there is a risk of premature convergence, unless it is frequently updated in scattered regions of the search-space.

Eberhart & Kennedy (1995) proposed the first local topology, where each particle is able to interact with only *k* other socially connected particles. This topology may be re-

[12] Position coinciding with the attractor and velocity equal to zero

[13] Also in: http://clerc.maurice.free.fr/pso/

ferred to as the '*k*-best topology' or as the 'ring topology' with *k* neighbours. Some authors call 'ring topology' to the particular case of '*k*-best topology' where *k* = 2. When a neighbourhood topology is referred to plainly as 'local topology', it typically refers to the ring structure. This neighbourhood takes the form of the fully connected topology when $k = m - 1$, with *m* being the number of particles in the swarm. Each neighbourhood is then composed of $k + 1$ particles, *k* of which overlap with the next neighbourhood thus producing the transfer of information between them. Therefore every particle is –if not directly– at least indirectly connected to every other. The last neighbourhood overlaps with the first. Note that the neighbourhoods are typically defined topologically, so that neighbouring particles are not necessarily near one another in the search-space.

Eberhart & Kennedy (1995) ran experiments with the original PSO (*w* = 1) for one global and two local ring topologies. The first local one with two neighbours (*k* = 2) and the other with 6 neighbours (*k* = 6). They claimed that the local versions are more reluctant to getting trapped in local optima because a number of groups of particles spontaneously separate and explore different regions, although this comes at the cost of a higher number of iterations, on average, to meet a given error level.

Carlisle & Dozier (2001) also experimented on different neighbourhood sizes, and arguably claimed that the global neighbourhood should be preferred as it requires less work to achieve the same results.

While countless topologies can be designed, three typical ones are shown in Fig. 4.3. The topology labelled 'a)' is the original *fully connected topology*, where every particle is connected to every other. Thus the neighbourhood-size equals the swarm-size (*m*), and every particle is informed by $m - 1$ neighbours. The topology labelled 'b)' is the so-called *wheel topology*, where only one particle –the centre of the wheel– is connected with all others, so that the size of its neighbourhood equals the swarm-size (*m*). In contrast, all other particles in the swarm are connected with this particle only, so that their neighbourhoods are composed of two particles: the particle itself, and the centre of the wheel. The topology labelled "c)" is the *ring topology* with two neighbours, where every particle is connected to two topologically immediate particles, and every neighbourhood overlaps with the previous and with the next ones in two particles (i.e. neighbourhood-size = 3). The ring topology can be extended to any number of

neighbours, where the overlapping is always given by (neighbourhood-size – 1) particles. Therefore, there are as many neighbourhoods as there are particles.

It is self-evident that the local versions are better suited to avoid premature convergence and escape poor local optima, but convergence might be so slow that either a fine-grain search does not take place and compliance with constraints is more likely to fail or a higher computational cost is required. A global neighbourhood speeds up convergence, and helps ensure that feasible solutions are found and the search is fine-grained, but it is more likely to get trapped in suboptimal solutions. Compared to the latter, the wheel topology has the speed of spread of information reduced by the centre of the wheel, which acts as a sort of buffer.

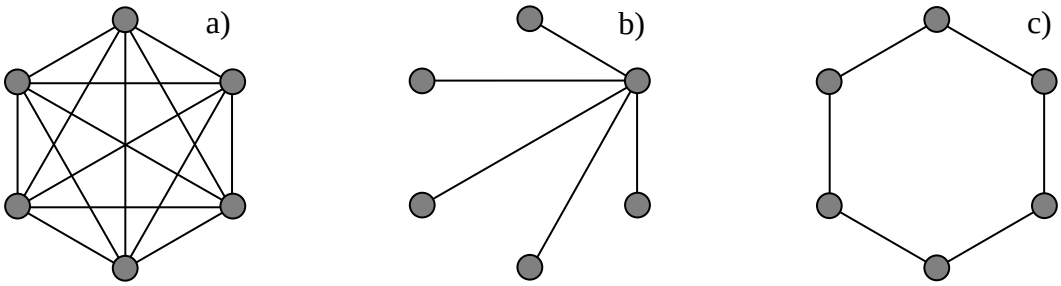


Fig. 4.3. a) fully connected topology, where neighbourhood-size equals swarm-size; b) wheel topology, where neighbourhood-size equals swarm-size for one particle and two for the rest; c) ring topology with neighbourhood-size equal to three.

Some of the experiments and studies in social psychology that influenced the development of the PSO paradigm –originally intended as a simulation of social behaviour– have been acknowledged and summarized in sections 4.2.6 and 4.4.1. James Kennedy (1999) carried out a systematic, experimental and statistical study of the influence of the social networks in the performance of the algorithm. Such a study was also inspired by theories and studies in social psychology; namely, Milgram (1967)'s experiment on the number of links required to connect two people selected at random; Granovetter (1973)'s *weak ties theory* arguing that information travelling distant acquaintances is very important because it may bring innovation to the ('strongly-tied') group; and Hutchins (1995)' *mega-minds* study showing that groups of networks converge on optima when there are a moderate number of connections among them whilst they converge on poor solutions when the cognitive structures are highly connected; plus Watts and Stro-

getz (1998)'s *small world effect* arguing that randomizing a small proportion of connections in a regular ring lattice maintains a high level of clustering[14] but greatly reduces average distance. All these works are cited in (Kennedy, 1999).

Therefore, Kennedy (1999) investigated and experimented on the global, ring (k = 2), wheel, and random topologies. Several degrees of small-world shortcuts were studied for the ring and wheel topologies (meaningless on the others). It was found that the neighbourhood topology has a critical effect on the performance of the algorithm, and that the effect was problem-dependent.

Kennedy & Eberhart(2001) also carried out experiments in order to compare the ring and the wheel topologies. The results suggested that the appropriateness of a neighbourhood topology is problem-dependent, in agreement with (Kennedy, 1999).

Other neighbourhood topologies which are also becoming somewhat classical are the 'von Neumann' or 'Square' topology in (Engelbrecht, 2005, p. 109) and in (Kennedy & Mendes, 2006) formed by arranging the population in a grid and connecting neighbours above, below, to the right, and to the left; and the 'Stochastic Star' topology in (Miranda, Keko, & Duque, 2008). The 'von Neumann' topology is shown in Fig. 4.4 whereas the 'Stochastic Star' topology is a sort of generalization of the 'global topology': in each iteration and for each dimension of the search-space, there is a probability 'p' that a particle will not access the **gbest** information and therefore would move only under the effects of the inertia and the individual memory (**pbest**).

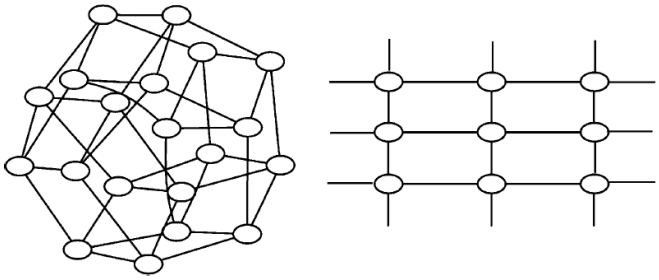


Fig. 4.4. 'von Neumann' or 'Square' topology (from (Kennedy & Mendes, 2006)).

While the optimal neighbourhood topology is problem-dependent, Suganthan (1999) proposed dynamic neighbourhoods where the social attractor varied from the local to

[14] Clustering is defined by the average number of neighbours that any two connected nodes have in common.

the swarm's best experience. Thus, the social best experience to be considered in the velocity update of a given particle *i* at a given iteration *it* is the global best if the inequality in Eq. (4.33) is true. Otherwise, a neighbourhood is defined for each particle *i* as all particles *j* which comply with the inequalities shown in Eq. (4.34).

$$\frac{3\cdot it + 0.6\cdot maxit}{maxit} \geq 0.9 \tag{4.33}$$

$$\frac{dist(\mathbf{x}_j, \mathbf{x}_i)}{\max\left(dist(\mathbf{x}_j, \mathbf{x}_i)\right)_i} < \frac{3\cdot it + 0.6\cdot maxit}{maxit} < 0.9 \tag{4.34}$$

where $dist(\mathbf{x}_j,\mathbf{x}_i)$ is the distance between the particle *i* and its potential neighbour *j*. Other dynamic neighbourhood topologies in the literature are those in (Richards & Ventura, 2003), (Abraham, Liu, & Chang, 2006), (Mohais, 2007), and (Akat & Gazi, 2008).

Maurice Clerc (2006a, pp. 87-101) offers an innovative view of the sociometry in PSO, as the swarm and the storage of the particles' best experiences are thought of as two different swarms: the 'explorer swarm', and the 'memory swarm'. (...) *This more complicated representation offers more freedom of configuration, for example by modifying the number of memories or by making them communicate directly with each other.* (Clerc, 2006a)

Numberless neighbourhood topologies are possible, and the optimal design is problem-dependent. For a comprehensive review of neighbourhood topologies, refer to (Kennedy, 1998), (Kennedy, 1999), (Suganthan, 1999), (Richards & Ventura, 2003), (Mendes, 2004), (Li, 2004), (Kennedy & Mendes, 2006), (Clerc, 2006a, pp. 87-101), (Mohais, 2007), (Abraham, Liu, & Chang, 2006), (Miranda, Keko, & Duque, 2008), (Akat & Gazi, 2008), among others.

4.4.8. Constraint-handling

The PSO method requires a constraint-handling technique (CHT) incorporated to be able to deal with constrained problems. These CHTs can be classified according to the way constraints are treated (refer to (Koziel & Michalewicz, 1999), (Engelbrecht, 2005, pp. 29-30), (Takahama, Sakai, & Iwane, 2006), (Innocente & Sienz, 2008), and (Worasucheep, 2008)). The following 6 groups are considered here:

4.4.8.1. Methods that use constraints only to evaluate feasibility

These methods allow infeasible solutions, although they are ignored. The search is started with one or more feasible solutions, and all infeasible solutions generated are plainly disregarded. An example of this is the *preserving feasibility strategy* (PF) initially proposed by (Hu & Eberhart, 2002) –also used by (Hu, Eberhart, & Shi, 2003)–, which was the first method to be applied to the PSO algorithm, to the best of my knowledge. This technique requires the feasibility of the initial swarm, and the search is not guided through infeasible space. While particles can fly over infeasible regions, they are pulled back to feasible space as infeasible positions are not stored in memory. Its advantages are that a feasible solution is guaranteed, and that it only requires two small modifications to the unconstrained algorithm: successive random initialization until a feasible swarm is generated, and the feasibility condition for the update of the best experiences. Its drawbacks are that the random initialization might be extremely time-consuming or impossible for low feasibility ratios of the search-space, and the lack of exploration of infeasible space. It may not be possible to explore feasible islands.

4.4.8.2. Methods that consider $f(\mathbf{x})$ and $cv(\mathbf{x})$ separately

These methods coincide with the previous group except when comparing two infeasible solutions, whose comparison is based on their constraint violations (*cv*). Therefore, infeasible space is also explored, without additional objective function evaluations (FEs). Thus, comparisons between feasible solutions are based on the objective function; a feasible solution is always preferred over an infeasible one; and infeasible solutions are compared based on constraint violations (*cv*). As a safety mechanism, if the latter are the same, comparisons are based on the objective function. The problem is viewed as a bi-objective problem, where the objective of finding feasible solutions is given absolute priority over the objective of minimizing the objective function.

Toscano Pulido & Coello Coello (2004) proposed a mechanism to handle constraints in PSO that can be viewed as a PF with the addition of comparisons between infeasible particles carried out based on their aggregated normalized constraint violations. The normalization is performed by dividing each individual constraint violation of a particle by the highest corresponding constraint violation in the population. Thus, the overall

violation is the aggregation of all the normalized ones. Some of several similar mechanisms are the *feasibility-based rule* in (He & Wang, 2007); the *ε Constrained Method* implemented in (Takahama & Sakai, 2006) and in (Takahama, Sakai, & Iwane, 2006) –where the *ε Constrained Method* already includes a tolerance for the constraint violations–; and the CHTs in (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005) and in (Muñoz Zavala, Hernández Aguirre, Villa Diharce, & Botello Rionda, 2006). Likewise, Wang & Yin (2008) implemented a related mechanism that also considers the objective function values and constraint violations separately, but ranking and selection operations are performed and objective function information may also be used to guide the search through infeasible space[15].

The main drawbacks of these methods are that the objective function information is mainly ignored in highly constrained problems, and that a very poor feasible solution would be given priority over a good, near-feasible solution.

4.4.8.3. Methods that combine *f*(x) and *cv*(x)

These methods can also be viewed as optimizing two objectives, but now minimizing constraint violations is not given absolute priority but the two objectives are combined into a single value. Popular techniques are the 'penalization methods' (PMs), which transform the original constrained problem into an unconstrained one by penalizing the objective function associated with infeasible solutions. Thus, the relative priority given to the objectives is somewhat weighted. Many different kinds of PMs can be found in the literature according to the way the penalization is calculated.

The advantage of these methods is that they use both objective and constraint functions information within the infeasible space to smoothly guide the search towards more promising areas. Since constrained problems are treated as unconstrained once infeasible solutions are penalized, they work well on highly constrained problems. The drawback is that they are sensitive to the tuning of at least a couple of problem-dependent penalization coefficients, performing badly when the solution lies on the boundaries. High penalizations might lead to infeasible regions not being explored converging to

[15] Feasible solutions are ranked first according to their objective function values, followed by infeasible solutions according to their nondomination levels. If the latter are the same, priority is given to either smaller constraint violation or to the smaller objective function value (two ranking operations are implemented).

non-optimal but feasible solutions, whereas low penalizations might lead to the system evolving solutions that are violating constraints but present themselves as better than feasible solutions. However, research on adaptive coefficients is extensive in the literature (e.g. (Parsopoulos & Vrahatis, 2002) and (Coello Coello, 2000)). A classical additive, constant penalization scheme –linked to the amount of infeasibility– is shown in Eqs. (4.35) and (4.36),

$$f_p(\mathbf{x}) = f(\mathbf{x}) + \sum_{j=1}^{m} \left[k_j \cdot \left(f_j(\mathbf{x}) \right)^{\alpha_j} \right] \tag{4.35}$$

$$f_j(\mathbf{x}) = \begin{cases} \max\left\{0, g_j(\mathbf{x})\right\} & ; \quad 1 \le j \le q \\ \text{abs}\left(g_j(\mathbf{x})\right) & ; \quad q < j \le m \end{cases} \tag{4.36}$$

where $f(\mathbf{x})$ is the conflict function; $f_p(\mathbf{x})$ is the penalized conflict function; $f_j(\mathbf{x})$ is the amount of violation of jth constraint; and k_j and α_j are penalization coefficients. The latter may be constant, time-varying, or adaptive, and they can be the same or different for different constraints. Typically, k_j is set to high and α_j to small values.

4.4.8.4. Repair algorithms

Some procedures are applied so that an infeasible candidate solution is repaired by relocating it to a nearby feasible position. Simple instances of these techniques are the 'reflection method' (Foryś & Bochenek, 2004), and 'the cut-off at the boundary' technique to handle interval constraints only, and the 'bisection' method for interval and inequality constraints ((Innocente, 2006) and (Innocente & Sienz, 2008)). The repairing of a particle flying out of bounds by two different 'cut-off at the boundary' techniques and by the 'bisection' method is illustrated in Fig. 4.5.

The effect of the 'cut-off at the boundary' technique (with velocity reset to zero) can be observed in Fig. 4.6 (below), while that of the 'bisection' method is shown in Fig. 4.7. In Fig. 4.6 (above), the plain PF technique handles the interval constraints for comparative purposes. The 'cut-off at the boundary' technique works very well if the solution is on the boundary, but particles get stuck on the boundaries when the solution is near but not on them.

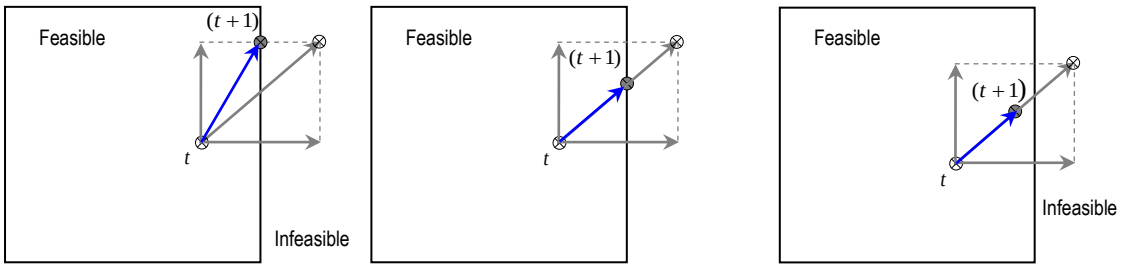


Fig. 4.5. Cut-off at the boundary techniques (left and centre) and bisection method (right). On the left, the solution is relocated on the boundary the nearest possible to the attempted infeasible location. In the centre, the vector of displacement is cut off so that its direction remains unchanged. On the right, the vector of displacement is successively divided by two until the new solution is feasible.

Some techniques like the 'cut-off at the boundary' reset the particle's velocity to zero when the trajectory is altered by the confinement technique. Clerc (2007) studied different methods to confine the particles to the feasible intervals. Since they are biased, the idea is two combine two methods to obtain a less biased one.

It is fair to note that Helwig and Wanka (2007), (2008) proved that for a standard PSO, the particles tend to move outside the boundaries with overwhelming probability in the first iteration. Hence a method of confinement would be highly beneficial. Note, however, that the PF method does not evaluate the objective function for infeasible particles. And sometimes it is not entirely undesirable to allow the particles to move outside the boundaries so as to repeatedly overfly the solution when the latter is near the boundary.

4.4.8.5. Multiobjective-based methods

Constraint violations are viewed as additional objectives to be minimized. Thus, constrained single-objective problems can be tackled using multiobjective (MO) techniques. Venter & Haftka (2008) proposed a PSO algorithm combined with a bi-objective formulation of the problem, where the additional objective is the minimization of a measure of constraint violation. The problem is solved and a Pareto front is obtained. The solution is the point on the front with the best true objective and zero constraint violation. de Freitas Vaz et al. (2006) also proposed a PSO algorithm combined with a MO-based CHT.

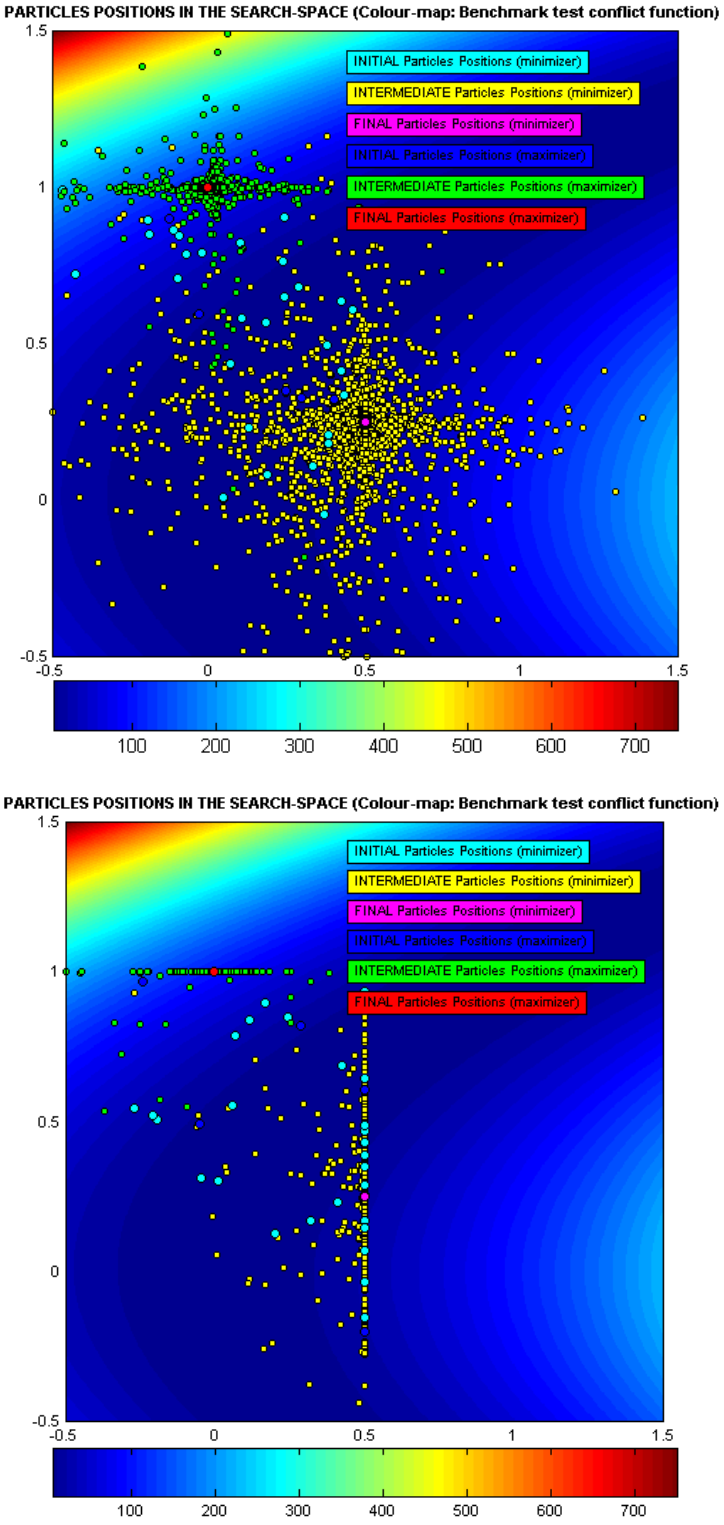


Fig. 4.6. A PSO algorithm optimizing a constrained problem, where interval constraints are $-0.50 \leq x_1 \leq 0.50$ and $-0.50 \leq x_2 \leq 1.00$. The constraint-handling techniques are the plain 'preserving feasibility' technique (above) and the 'preserving feasibility with cut-off at the boundary and velocity zeroed-in' technique (below). There is a minimizer (yellow intermediate positions) and a maximizer (green intermediate positions). The search is carried out along 4000 time-steps in both cases.

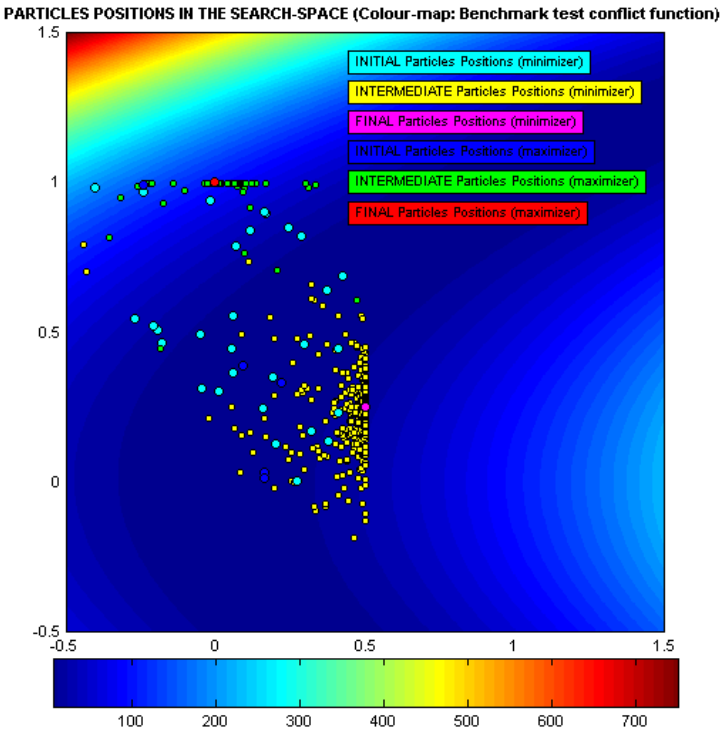


Fig. 4.7. A PSO algorithm optimizing a constrained problem, where interval constraints are $-0.50 \leq x_1 \leq 0.50$ and $-0.50 \leq x_2 \leq 1.00$. The constraint-handling technique is the 'bisection' method. There is a minimizer (yellow intermediate positions) and a maximizer (green intermediate positions). The search is carried out along 4000 time-steps.

4.4.9. Other features

Another feature that needs to be defined in order to run a PSO algorithm is the size of the swarm. Kennedy & Eberhart (2001) suggest from 10 to 50 particles, while Carlisle & Dozier (2001) argue that a swarm of 30 particles appears to be a good choice.

The Standard PSO (Auger, et al., 2007) calculates the swarm-size as a function of the dimensionality of the problem, as shown in Eq. (4.37),

$$m = 10 + 2 \cdot \sqrt{2 \cdot d} \tag{4.37}$$

where m is the number of particles in the swarm, and d is the number of dimensions of the search-space.

However, the swarm-size should also depend on the FEs available, on the coefficients' settings, and on the neighbourhood structure. That is, if the latter two are such that fast convergence is favoured, then a bigger swarm would be advisable. Also on the decision

of the user as to whether it is better to have a longer search or a more parallel search for the problem at issue. Further research on the size of the swarm in PSO can be found in (van den Bergh & Engelbrecht, 2001) and (DeBao & ChunXia, 2009)). The latter proposed an adaptive population size.

In the simplest PSO algorithm, the particles are randomly scattered over the search-space at the initial time-step, while their velocities are randomly initialized within the intervals $[-v_{\max}, v_{\max}]$ or set to zero. Any technique for the design of experiments (DoE) can be used to improve the initialization of the particles' positions. One would expect that these techniques, e.g. Latin Hypercube Sampling (LHS), would improve the performance of the algorithm at least for unconstrained or boundary constrained problems. Alternatively, two initial populations may be generated: **p** and **pbest**, instead of one population and its velocities. This thesis is not concerned with initializations in PSO. For reading on the subject, refer to (Clerc, 2008c) and (Helwig & Wanka, 2008).

The social best experiences may be updated only once per time-step, after all individual best experiences have been updated (parallel/synchronous update); or every time a particle's best experience is updated (sequential/asynchronous update). In general, the second approach results in marginally faster convergence, while the vast majority of researchers choose the first approach.

Canonical PSO has only one sociality term: the best experience in the neighbourhood. A second sociality term may be added so that the particle is attracted to both the best experience in the neighbourhood and the best in the whole swarm. Mendes, Kennedy, & Neves (2004) go further with their *Fully-Informed PSO*, where every particle is influenced to some extent by all its neighbours.

4.4.10. Binary PSO

The method was originally proposed in (Kennedy & Eberhart, 1997) as a variation of the original PSO. That is, no inertia weight or constriction factor considered.

The search-space is now a binary *n*-dimensional hyper-cube $\mathcal{S} = \{0,1\}^n$; the individuals are represented by binary, fixed-length bit-strings; and the conflict function is defined as $c: \{0,1\}^n \to \mathcal{R}$. While the metaphor of bird flocks no longer applies, it appears that the

metaphor of cognitive processes is still valid. The update equations are as shown in Eqs. (4.38) to (4.40):

$$v_{ij}^{(t)} = w \cdot v_{ij}^{(t-1)} + iw \cdot U_{(0,1)} \cdot \left(pbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) + sw \cdot U_{(0,1)} \cdot \left(gbest_{j}^{(t-1)} - x_{ij}^{(t-1)} \right) \tag{4.38}$$

$$p_{ij}^{(t)} = \frac{1}{1+e^{-v_{ij}^{(t)}}} \in [0,1] \subset \mathcal{R} \tag{4.39}$$

$$x_{ij}^{(t)} = \begin{cases} 1 & \text{if} \quad U_{(0,1)} < p_{ij}^{(t)} \\ 0 & \text{otherwise} \end{cases} \tag{4.40}$$

where $p_{ij}^{(t)}$ stands for the probability of a bit adopting the state '1'. In other words, $p_{ij}^{(t)}$ is the probability that the individual *i* has of holding the feature *j* at time-step *t*. All the others parameters in Eq. (4.38) remain the same as in classical PSO.

Although the position is restricted to the vertices of a binary hypercube, the velocity equation remains unchanged. It does not make any sense to think of $v_{ij}^{(t)}$ as a velocity but rather as a measure of the likelihood for individual *i* to hold the belief *j*. Notice that if $v_{ij}^{(t)} < 0 \;\Rightarrow\; p_{ij}^{(t)} < 0.5$, whereas if $v_{ij}^{(t)} > 0 \;\Rightarrow\; p_{ij}^{(t)} > 0.5$.

To summarize, an individual seeks consistency among its beliefs, which is attained by minimizing the conflicts among them. Thus, the conflict function receives the beliefs as inputs and returns a scalar that stands for the level of conflict among those beliefs that are held together.

Although an individual that holds a certain belief does not change its mind immediately when it notices that someone else's beliefs are more consistent, it is influenced by that observation. This is consistent with Eq. (4.38), whose first term stands for the tendency to keep the belief the individual has, while the second and third terms tend to move the probability threshold upwards or downwards if the belief is or is not, respectively, held by the best previous individual and social experiences.

If the current probability is either too high or too low, it might take a long time to change the activation status of the feature. For values of $v_{\max} = \pm 10$ and greater in absolute value, the sigmoid function in Eq. (4.39) saturates. Hence Kennedy & Eberhart

(2001) suggested setting $w=1$, $(iw+sw)=4$, and $v_{max}=\pm 4$. Thus there is always a chance of at least $p^{(t)}=0.018$ that a bit will change state. While the constraint v_{max} prevents the explosion in classical PSO, it prevents the complete loss of diversity in this binary PSO in a similar fashion as the mutation does in binary GAs. Higher values reduce the probability of new vectors. Hence the higher the v_{max} the lower the exploration.

Kennedy & Eberhart (2001) reported a general better performance of their binary PSO in comparison to a standard GA, a mutation-only GA, and a crossover-only GA, when dealing with multimodal problems. Kennedy (1998) implemented a binary PSO to optimize a directed S-digraph inference problem, where the aim is to find the optimal state of the binary nodes in a recurrent inference network. The network in (Kennedy, 1998) was composed of 9 nodes and a complex pattern of connections. His binary PSO was able to optimize such a problem with a population of only 20 so-called 'EleMentals' (particles), where some member of the population could always find the optimum within the first 20 time steps. The final 'states of minds' of the 'EleMentals' showed the formation of cultures. No binary PSO is dealt with in this thesis.

4.5. Final remarks and closure

A fairly extensive review of the PSO method was presented, including its foundations and the links to the three broad fields with which the method is associated: *Swarm Intelligence*, *Social Psychology* and *Mathematical Optimization*.

In order to better understand how the paradigm works, a brief discussion on *Swarm Intelligence* was offered, and a few important phenomena in the field such as *emergence*, *self-organization*, *division of labour* and *stigmergy* were discussed.

While the simulation of social behaviour was initially the aim of the PSO paradigm, the emergent behaviour observed in animal societies is clearly the link between *Swarm Intelligence* and *Social Psychology* from the PSO point of view. Thus, the social behaviour that occurs in a number of animal societies –including human beings– was also presented, in addition to the collective behaviour that is imposed on swarm robots. Given that human beings present remarkably higher complexity than say ant colonies or bird flocks, some influential experiments and theories in social psychology were also

included in the review. All this helps understand how the method works, and brings multidisciplinary new ideas into PSO research as to how to improve its performance. Good examples of this are the *EleMentals* in (Kennedy, 1998) and the *small world* and *mega-minds* concepts brought into the sociometry in PSO by Kennedy (1999). Bear in mind that the neighbourhood topology in PSO is still a social network despite being applied almost exclusively to mathematical optimization, and much insight and new ideas can be imported from the *Social Psychology* discipline.

A couple of simple Ant Colony Optimization algorithms were introduced simply to present the other popular Swarm-Intelligence-based paradigm which has also achieved great success as an optimization tool.

With regards to the PSO algorithm *per se*, some attempts to formalize the dynamics of the swarm via the study of the influence of the coefficients' settings in the trajectory of a single, isolated, deterministic particle were described. This allows gaining insight into the system itself, as well as some understanding as to how the coefficients may be conveniently set, and what to expect from different settings. In particular, it is of evident interest to understand how the so-called explosion may be controlled without giving up on the explorative power of the PSO algorithm.

In addition, the neighbourhood topology has a strong impact on the performance of the method. By controlling the speed of spread of information in the swarm, the neighbourhood structure is critical in controlling the speed of convergence of the optimizer as a whole, as well as the type of trajectory the particles undergo. Therefore, a review of some studies on neighbourhood topologies from the literature was also offered.

Finally, due to the fact that the aim of this work is to develop a fully working PSO algorithm for Constrained Optimization Problems, different groups of constraint-handling techniques from the literature were discussed in some details.

Chapter 5 and Chapter 6 present my own studies on the influence of the coefficients' settings on the behaviour and performance of the PSO method. Chapter 7 deals with the study of some neighbourhood topologies, and Chapter 8 with constraint-handling.

SECTION II

RESEARCH

Chapter 5

SYSTEMATIC COEFFICIENTS STUDY WITH STATIONARY ATTRACTORS

This chapter is concerned with the influence of the settings of the velocity constraint and of the coefficients involved in the particles' velocity update equation on the dynamics of a single particle pulled by stationary attractors. The individuality and sociality weights are kept the same, so that only the settings of the acceleration weight are analyzed. The equation of the particle's position is obtained as a function of the coefficients and of two initial conditions. There are three cases according to whether the roots of the characteristic polynomial of the recurrence relation of the particle's position are two and real-valued; only one and real-valued; or complex conjugates. Hence three equations are obtained, and the convergence/divergence conditions are stated. The deterministic explosion and the cyclic behaviour of the original PSO are discussed, and the concept of stochastic explosion is proposed by visualizing divergent trajectories due to randomness. The conditions of convergence are then studied in more detail, and a region of convergence in the plane 'ϕ–w' is offered. A short discussion on the concepts of 'exploration' and 'exploitation' and their meanings within this thesis is provided. The effectiveness of the velocity constraint in controlling the explosion without improving convergence is discussed and visualized. Different settings of the inertia weight for a number of settings of the acceleration weight are studied, showing its ability to both control the explosion and improve convergence. First, the random weights are replaced by their expected value so as to remove randomness from the effect of the inertia weight. Then the random weights are reincorporated and the trajectories corresponding to the same coefficients' settings are compared. A visual study of the combined effect of the inertia weight and the velocity constraint follows. A brief discussion on the constriction factor *Type 1"* is offered and a few settings are tested and compared. The combined effect of constriction and velocity constraint is also visually analyzed. Finally, a 'ϕ–w' relationship is presented, which is expected to favour convergence. Visual analyses of the trajectory of the particle are carried out. The overall objective of this study is to identify a range of usable coefficients' settings, and the type of behaviour to be expected in terms of the particle's trajectory and convergence towards the stationary attractor. Further complexity is incorporated in the next chapter.

5.1. Introduction

The settings of the coefficients in the velocity update equation of the particle swarm algorithm have a critical effect on the behaviour exhibited by the swarm. Thus, a particle might converge or diverge; exhibit oscillations with different frequencies and damping steepness; converge from one side without much oscillation; converge smoothly by a sustained decrease in the oscillations amplitude; exhibit local explosions followed by convergence; etc. Even if convergence is ensured, different behaviours lead to different capabilities of the algorithm. In this chapter, the dynamics of a single particle pulled by

stationary attractors is studied. The individuality and sociality weights are kept equal to one another so as to analyze the settings of the acceleration weight instead.

The closed-form for the particle's position is obtained as a function of the coefficients and of two initial conditions: either the positions in the first two time-steps, or the position and the velocity in the first time-step. There can be three different cases according to whether the roots of the characteristic polynomial of the second-order linear recurrence relation are 1) two and real-valued; 2) only one and real-valued; or 3) complex conjugates. Thus, three closed-forms of the particle's position are offered –one for each case– and the conditions for convergence and divergence are stated.

The so-called 'deterministic explosion' and the 'cyclic behaviour' of the original PSO are discussed, and then the concept of 'stochastic explosion' is proposed by visualizing divergent trajectories of a particle whose coefficients do not lead to a deterministic explosion. The conditions of convergence are then studied in more detail, and a region of convergence is presented bounded by inequalities in the plane 'ϕ–w'. The convergence region obtained coincides with the one in (van den Bergh, 2001), although further explained and bounded, in agreement with (Trelea, 2003). It is proved here that the module of the complex conjugate roots is simply the square root of the inertia weight, which therefore must be smaller than one to ensure convergence for complex conjugate roots.

The meanings of the terms 'exploration' and 'exploitation' in the literature as well as those assigned to them within this thesis are briefly discussed.

The effect of the velocity constraint is analyzed and visualized, showing its effectiveness in controlling the explosion by narrowing the part of the search-space being explored without improving the particle's convergence capabilities.

Different settings of the inertia weight for a number of settings of the acceleration weight are visually studied. Its ability to control the explosion and also improve convergence is visualized in terms of the particle's trajectory, thus confirming the convergence region in the plane 'ϕ–w' previously obtained. First, the random weights are replaced by their expected value so as to remove randomness from the 'pure' effect of the inertia weight. Later, the random weights are reincorporated and the trajectories corresponding to the same settings –with and without randomness– are compared. The next step is to study the combined effect of the inertia weight and the velocity constraint.

A brief discussion of the constriction factor *Type 1"* proposed in (Clerc & Kennedy, 2002) is offered, and a few settings are tested and compared. The other constriction types developed in (Clerc & Kennedy, 2002) are beyond the scope of this study because they modify the classical PSO (intuitive) equations. The combined effect of the constriction factor *Type 1"* and the velocity constraint is also visually analyzed.

Finally, a proposed relationship between the inertia and the acceleration weights is presented, which was developed without considering the convergence per se. Nevertheless, it is still expected to favour convergence. Visual analyses of the trajectory of the particle are carried out so as to confirm the expectations.

The main objective of this study is to obtain a range of coefficients' settings that are usable, and the type of behaviour to be expected in terms of the particle's trajectory and convergence towards the stationary attractor. Further complexity, namely updating attractors and particles' interactions, are incorporated in the next chapter. Note that the studies here are extensive and visual to a great extent.

5.2. Position equation

Ozcan and Mohan (1998), (1999) attempted the first formal analysis of the particles' trajectories, although James Kennedy may have informally presented related work previously in Vancouver, 1998. Despite acknowledging the inertia weight proposed by Shi and Eberhart (1998b), they only concerned themselves with the original algorithm (i.e. $w=1$), and studied different settings of the acceleration coefficient ϕ, where:

$$\phi = \phi_i + \phi_s = iw \cdot U_{(0,1)} + sw \cdot U_{(0,1)} \tag{5.1}$$

Thus, Ozcan & Mohan (1998) studied a simplified system of one particle flying over a one-dimensional space, where the individual and social best experiences coincide and the coefficient ϕ is kept constant. First, a recurrence relation was formulated, which was solved obtaining a closed-form for the particle's position using the initial conditions $x^{(0)}$ and $v^{(0)}$. They studied the trajectory of the particle in the 'real' and in the 'complex' domains, analyzing special cases according to the value of ϕ. The equations and the corresponding analyses were further simplified by assuming the initial position $x^{(0)}$ coincid-

ing with the attractor *p*, which in turn coincided with the individual and with the social best experiences. More general expressions for the particle's position incorporating the inertia weight and considering the more general case of $p \neq x^{(0)}$ are developed hereafter. The resulting equations are analyzed and compared to the conclusions offered in (Ozcan & Mohan, 1998); (Ozcan & Mohan, 1999); and (van den Bergh, 2001). The latter already included the inertia weight in his studies.

5.2.1. Recurrence relation

For convenience, the general expression of the particles' velocity update equation is reproduced in Eq. (5.2).

$$\begin{cases} v_{ij}^{(t)} = w \cdot v_{ij}^{(t-1)} + iw \cdot U_{(0,1)} \cdot \left(pbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) + sw \cdot U_{(0,1)} \cdot \left(lbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) \\ x_{ij}^{(t)} = x_{ij}^{(t-1)} + v_{ij}^{(t)} \end{cases} \tag{5.2}$$

where *w*, *iw*, *sw* are the inertia, individuality and sociality weights, respectively; $x_{ij}^{(t)}$ and $v_{ij}^{(t)}$ are j^{th} component of the position and the velocity of particle *i*, respectively; *pbest* and *lbest* stand for the best individual and social experiences, respectively; and $U_{(0,1)}$ is a random number from a uniform distribution between '0' and '1' resampled anew every time it is referenced.

For future reference, the relation between the acceleration weight (*aw*) and the acceleration coefficient (ϕ) is shown in Eq. (5.3):

$$0 < \phi = \left(\phi_i + \phi_s \right) = \left(iw \cdot U_{(0,1)} + sw \cdot U_{(0,1)} \right) \leq \left(aw = iw + sw \right) \tag{5.3}$$

Thus, $\phi = aw$ if the random weights are removed (or equal to their maximum value '1'), whereas $\phi = 0.5 \cdot aw$ if they are replaced by their expected value '0.5'. Notice that, if the algorithm incorporates randomness, then ϕ is random whereas *aw* is not.

While Ozcan & Mohan (1999) extended their previous studies by considering multi-dimensional space and the two attractors separately, it is preferred here to maintain a single attractor and a uni-dimensional space to keep the expressions simpler, since:

- The two attractors are stationary and ϕ is constant, so that the particle is in fact attracted towards a single stationary point (weighted average of the attractors).
- The relative values of the individuality and sociality weights are not under study.
- The PSO algorithm maintains different dimensions independent from one another.

As opposed to (Ozcan & Mohan, 1998), it is not considered here that $pbest \equiv lbest$ but that the actual attractor is given by a weighted average of both best experiences:

$$p = \frac{\phi_i \cdot pbest + \phi_s \cdot lbest}{\underbrace{\phi_i + \phi_s}_{\phi}} \tag{5.4}$$

Thus, the system can be simplified as shown in Eq. (5.5).

$$\begin{aligned} v^{(t)} &= w \cdot v^{(t-1)} + \phi \cdot \left(p - x^{(t-1)}\right) \\ x^{(t)} &= x^{(t-1)} + v^{(t)} \end{aligned} \tag{5.5}$$

Replacing $v^{(t)}$ in $x^{(t)}$ in Eq. (5.5), and given that $v^{(t-1)} = x^{(t-1)} - x^{(t-2)}$, the *second order non-homogeneous linear recurrence relation* in Eq. (5.6) is obtained.

$$\boxed{x^{(t)} + \left(\phi - w - 1\right) \cdot x^{(t-1)} + w \cdot x^{(t-2)} = \phi \cdot p} \tag{5.6}$$

5.2.2. Closed-form expression

The characteristic polynomial for the homogeneous recurrence relation associated to Eq. (5.6) is given by Eq. (5.7), whose roots are shown in Eqs. (5.8) to (5.10):

$$r^2 + \left(\phi - w - 1\right) \cdot r + w = 0 \tag{5.7}$$

$$r = \frac{\left(1 + w\right)}{2} - \frac{\phi}{2} \pm \frac{\sqrt{\phi^2 - \left(2 \cdot w + 2\right) \cdot \phi + \left(w - 1\right)^2}}{2} \tag{5.8}$$

$$\boxed{\gamma = \sqrt{\phi^2 - \left(2 \cdot w + 2\right) \cdot \phi + \left(w - 1\right)^2}} \tag{5.9}$$

$$\boxed{r_1 = \frac{(1+w)}{2} - \frac{\phi}{2} + \frac{\gamma}{2}} \quad ; \quad \boxed{r_2 = \frac{(1+w)}{2} - \frac{\phi}{2} - \frac{\gamma}{2}} \quad ; \quad \boxed{r_1 - r_2 = \gamma} \tag{5.10}$$

Two solutions of the homogeneous recurrence relation are shown in Eq. (5.11).

$$x_1^{(t)} = r_1^t \quad ; \quad x_2^{(t)} = r_2^t \tag{5.11}$$

If $r_1 \neq r_2$, the solutions in Eq. (5.11) are linearly independent, and hence the general solution is given by the linear combination in Eq. (5.12).

$$x_H^{(t)} = C_1 \cdot r_1^t + C_2 \cdot r_2^t = C_1 \cdot \left(\frac{1+w}{2} - \frac{\phi}{2} + \frac{\gamma}{2}\right)^t + C_2 \cdot \left(\frac{1+w}{2} - \frac{\phi}{2} - \frac{\gamma}{2}\right)^t \tag{5.12}$$

The general solution of the non-homogeneous recurrence relation in Eq. (5.6) is given by the general solution of the associated homogeneous recurrence relation plus a particular solution. From Eq. (5.6), it can be observed that a particular solution of the recurrence relation is of the form $x_P^{(t)} = C$ (constant). Therefore,

$$C + (\phi - w - 1) \cdot C + w \cdot C = \phi \cdot p \quad \Rightarrow \quad C = p \tag{5.13}$$

Thus, the general solution of the non-homogeneous recurrence relation is, in general, as in Eq. (5.14).

$$x^{(t)} = C + C_1 \cdot r_1^t + C_2 \cdot r_2^t = p + C_1 \cdot \left(\frac{1+w}{2} - \frac{\phi}{2} + \frac{\gamma}{2}\right)^t + C_2 \cdot \left(\frac{1+w}{2} - \frac{\phi}{2} - \frac{\gamma}{2}\right)^t \tag{5.14}$$

There are three cases for the general solution of the recurrence relation in Eq. (5.6), depending on the value of γ in Eq. (5.9):

1. $\gamma^2 > 0$: The roots of the characteristic polynomial are real and different.
2. $\gamma^2 < 0$: The roots of the characteristic polynomial are complex conjugate numbers.
3. $\gamma^2 = 0$: The roots of the characteristic polynomial are real and the same.

5.2.2.1. Two real-valued roots

The general solution for the recurrence relation in Eq. (5.6) is as in Eq. (5.14), where two initial conditions are required for the definition of the constants. Those conditions may be given by the first two positions $x^{(0)}$ and $x^{(1)}$. Thus, for $t = 0$:

$$x^{(0)} = p + C_1 + C_2 \qquad \Rightarrow \qquad C_2 = x^{(0)} - p - C_1 \tag{5.15}$$

In turn, for $t = 1$:

$$x^{(1)} = p + C_1 \cdot r_1 + C_2 \cdot r_2 \tag{5.16}$$

$$p + C_1 \cdot r_1 + \left(x^{(0)} - p - C_1\right) \cdot r_2 - x^{(1)} = 0 \tag{5.17}$$

$$C_1 \cdot \left(r_1 - r_2\right) + x^{(0)} \cdot r_2 - p \cdot r_2 + p - x^{(1)} = C_1 \cdot \gamma + r_2 \cdot \left(x^{(0)} - p\right) + p - x^{(1)} = 0 \tag{5.18}$$

$$C_1 = \frac{r_2 \cdot \left(p - x^{(0)}\right) - \left(p - x^{(1)}\right)}{\gamma} \tag{5.19}$$

Introducing Eq. (5.19) into (5.15) yields:

$$C_2 = \frac{\gamma \cdot \left(x^{(0)} - p\right)}{\gamma} - \frac{r_2 \cdot \left(p - x^{(0)}\right)}{\gamma} + \frac{\left(p - x^{(1)}\right)}{\gamma} \tag{5.20}$$

$$C_2 = \frac{-\left(p - x^{(0)}\right) \cdot \left(\gamma + r_2\right) + \left(p - x^{(1)}\right)}{\gamma} \tag{5.21}$$

$$C_2 = \frac{-r_1 \cdot \left(p - x^{(0)}\right) + \left(p - x^{(1)}\right)}{\gamma} \tag{5.22}$$

Thus, solving Eq. (5.14) for the two initial conditions $x^{(0)}$ and $x^{(1)}$ yields the closed-form equation for the particle's position shown in Eq. (5.23):

$$\boxed{x^{(t)} = p + \frac{r_2 \cdot \left(p - x^{(0)}\right) - \left(p - x^{(1)}\right)}{\gamma} \cdot r_1^t + \frac{-r_1 \cdot \left(p - x^{(0)}\right) + \left(p - x^{(1)}\right)}{\gamma} \cdot r_2^t} \tag{5.23}$$

where the roots r_1 and r_2 are as in Eqs. (5.9) and (5.10).

Alternatively, the initial conditions may be given as $x^{(0)}$ and $v^{(0)}$, where $x^{(1)} = x^{(0)} + v^{(1)}$ and $v^{(1)} = w \cdot v^{(0)} + \phi \cdot \left(p - x^{(0)}\right)$. Therefore $x^{(1)}$ and $v^{(0)}$ are related as shown in Eq. (5.24).

$$v^{(0)} = \frac{x^{(1)} - (1-\phi) \cdot x^{(0)} - \phi \cdot p}{w} \tag{5.24}$$

van den Bergh (2001, pp. 267-269) developed a similar closed-form, but his expressions involve three initial conditions $x^{(0)}$, $x^{(1)}$, $x^{(2)}$ rather than the two essential ones $x^{(0)}$, $x^{(1)}$ (or $x^{(0)}$, $v^{(0)}$). It is evident that the particle converges towards the attractor p if the absolute values of both roots are smaller than one, whereas it diverges if at least one is greater.

5.2.2.2. Two complex conjugate roots

The general solution for the recurrence relation in Eq. (5.6) can be given by Eq. (5.23), but it can be more conveniently written in terms of real values. If $\gamma^2 < 0$, let us call:

$$\boxed{\gamma' = \sqrt{-\gamma^2} = \sqrt{-\phi^2 + (2 \cdot w + 2) \cdot \phi - (w-1)^2}} \tag{5.25}$$

$$r_1 = \left(\frac{(1+w)}{2} - \frac{\phi}{2}\right) + \left(\frac{\gamma'}{2}\right) \cdot i \quad ; \quad r_2 = \left(\frac{(1+w)}{2} - \frac{\phi}{2}\right) - \left(\frac{\gamma'}{2}\right) \cdot i \tag{5.26}$$

These roots can be written in polar coordinates (ρ,θ) in the complex plane, where:

$$\rho = \sqrt{\left(\frac{1+w-\phi}{2}\right)^2 + \left(\frac{\gamma'}{2}\right)^2}$$

$$\Rightarrow \quad \rho = \frac{\sqrt{(1 + 2 \cdot w - 2 \cdot \phi + w^2 - 2 \cdot w \cdot \phi + \phi^2) + (-\phi^2 + 2 \cdot w \cdot \phi + 2 \cdot \phi - w^2 + 2 \cdot w - 1)}}{2}$$

$$\Rightarrow \quad \rho = \frac{\sqrt{(2 \cdot w) + (2 \cdot w)}}{2} \quad \Rightarrow \quad \boxed{\rho = \sqrt{w}} \tag{5.27}$$

$$\boxed{\theta = \operatorname{atan}\left(\frac{\gamma'}{1+w-\phi}\right)} \tag{5.28}$$

$$\cos(\theta)=\frac{1}{\sqrt{w}}\cdot\left(\frac{1+w-\phi}{2}\right) \quad ; \quad \sin(\theta)=\frac{1}{\sqrt{w}}\cdot\left(\frac{\gamma'}{2}\right) \tag{5.29}$$

Note that if $(1+w-\phi)=0$, the complex roots only have imaginary components. Hence:

$$\theta=\frac{\pi}{2} \quad ; \quad \cos(\theta)=0 \quad ; \quad \sin(\theta)=1 \tag{5.30}$$

Therefore the roots of the polynomial can be written as in Eq. (5.31):

$$r_1=\rho\cdot(\cos(\theta)+i\cdot\sin(\theta)) \quad ; \quad r_2=\rho\cdot(\cos(\theta)-i\cdot\sin(\theta)) \tag{5.31}$$

By Euler's formula, $\cos(\theta)+i\cdot\sin(\theta)=e^{\theta\cdot i}$ and $\cos(\theta)-i\cdot\sin(\theta)=e^{-\theta\cdot i}$, so that:

$$r_1^t=\rho^t\cdot e^{t\cdot\theta\cdot i}=\rho^t\cdot(\cos(t\cdot\theta)+i\cdot\sin(t\cdot\theta)) \tag{5.32}$$

$$r_2^t=\rho^t\cdot e^{-t\cdot\theta\cdot i}=\rho^t\cdot(\cos(t\cdot\theta)-i\cdot\sin(t\cdot\theta)) \tag{5.33}$$

If r_1^t and r_2^t are two linearly independent solutions of the homogeneous recurrence relation associated to the non-homogeneous one in Eq. (5.6), then so are

$$\left(\frac{r_1^t+r_2^t}{2}\right)=\rho^t\cdot\cos(t\cdot\theta) \quad \text{and} \quad \left(\frac{r_1^t-r_2^t}{2\cdot i}\right)=\rho^t\cdot\sin(t\cdot\theta) \tag{5.34}$$

Thus, the general solution for Eq. (5.6) is written in terms of real values as follows:

$$x^{(t)}=C+C'_1\cdot\left(\frac{r_1^t+r_2^t}{2}\right)+C'_2\cdot\left(\frac{r_1^t-r_2^t}{2\cdot i}\right)=p+\rho^t\cdot(C'_1\cdot\cos(t\cdot\theta)+C'_2\cdot\sin(t\cdot\theta)) \tag{5.35}$$

$$\Rightarrow \quad x^{(t)}=p+\left(\sqrt{w}\right)^t\cdot(C'_1\cdot\cos(t\cdot\theta)+C'_2\cdot\sin(t\cdot\theta)) \tag{5.36}$$

The initial conditions are given by the first two positions $x^{(0)}$ and $x^{(1)}$. Thus, for t = 0:

$$x^{(0)}=p+C'_1 \quad \Rightarrow \quad C'_1=x^{(0)}-p \tag{5.37}$$

In turn, for t = 1:

$$x^{(1)} = p + \sqrt{w} \cdot \left(\left(x^{(0)} - p \right) \cdot \cos(\theta) + C'_2 \cdot \sin(\theta) \right) \tag{5.38}$$

$$\frac{x^{(1)} - p}{\sqrt{w}} - \left(x^{(0)} - p \right) \cdot \cos(\theta) = C'_2 \cdot \sin(\theta) \tag{5.39}$$

$$C'_2 \cdot \frac{1}{\sqrt{w}} \cdot \left(\frac{\gamma'}{2} \right) = \frac{x^{(1)} - p}{\sqrt{w}} - \left(x^{(0)} - p \right) \cdot \frac{1}{\sqrt{w}} \cdot \left(\frac{1 + w - \phi}{2} \right) \tag{5.40}$$

$$C'_2 = \frac{2 \cdot \left(x^{(1)} - p \right) - \left(x^{(0)} - p \right) \cdot (1 + w - \phi)}{\gamma'} \tag{5.41}$$

Thus, the equation for the particle's position when $\gamma^2 < 0$ is as shown in Eq. (5.42):

$$x^{(t)} = p + \left(\sqrt{w} \right)^t \cdot \left(- \left(p - x^{(0)} \right) \cdot \cos(t \cdot \theta) + \left(\frac{(1 + w - \phi) \cdot \left(p - x^{(0)} \right) - 2 \cdot \left(p - x^{(1)} \right)}{\gamma'} \right) \cdot \sin(t \cdot \theta) \right) \tag{5.42}$$

where γ' is as in Eq. (5.25) and θ is as in Eqs. (5.28) or (5.30). As can be observed, the particle converges towards the attractor p for $w < 1$ regardless of ϕ when $\gamma^2 < 0$ (i.e. the roots of the characteristic polynomial are complex conjugates). Again, the initial conditions may be given as $x^{(0)}$ and $v^{(0)}$ rather than $x^{(0)}$ and $x^{(1)}$, where $v^{(0)}$ is as in Eq. (5.24).

5.2.2.3. Only one root

From Eqs. (5.9) and (5.10), this condition implies that $\gamma^2 = 0$. Therefore the only root is real-valued, as shown in Eq. (5.43).

$$r = r_1 = r_2 = \frac{1 + w - \phi}{2} \tag{5.43}$$

Hence the general solution for Eq. (5.6) cannot be as in Eq. (5.23) because another solution for the homogeneous recurrence relation is required. It can be proved that if r^t is a solution, so is $\left(t \cdot r^t \right)$.

$$x^{(t)} = C + C_1 \cdot r^t + C_2 \cdot t \cdot r^t = p + \left(C_1 + C_2 \cdot t\right) \cdot \left(\frac{1 + w - \phi}{2}\right)^t \tag{5.44}$$

The initial conditions are given by the first two positions $x^{(0)}$ and $x^{(1)}$. Thus, for $t = 0$:

$$x^{(0)} = p + C_1 \qquad \Rightarrow \qquad C_1 = -\left(p - x^{(0)}\right) \tag{5.45}$$

In turn, for $t = 1$:

$$x^{(1)} = p + \left(-\left(p - x^{(0)}\right) + C_2\right) \cdot \left(\frac{1 + w - \phi}{2}\right) \tag{5.46}$$

$$\Rightarrow \qquad C_2 = \left(p - x^{(0)}\right) - \frac{2 \cdot \left(p - x^{(1)}\right)}{1 + w - \phi} \tag{5.47}$$

Thus, the equation for the particle's position when $\gamma^2 = 0$ is as shown in Eq. (5.48).

$$\Rightarrow \qquad \boxed{x^{(t)} = p + \left[-\left(p - x^{(0)}\right) + \left(\left(p - x^{(0)}\right) - \frac{2 \cdot \left(p - x^{(1)}\right)}{1 + w - \phi}\right) \cdot t\right] \cdot \left(\frac{1 + w - \phi}{2}\right)^t} \tag{5.48}$$

Thus, the particle still diverges if the absolute value of the root equals '1'. This phenomenon was explained differently in (Clerc & Kennedy, 2002, p. 61) for $w = 1$, $\phi = 4$.

5.2.3. Discussion

Once obtained the closed-form expressions of the particle's position for the three cases of γ, the analysis of such expressions, some special cases, and references to similar work in the literature are provided in this section.

5.2.3.1. Two real-valued roots

If $\gamma^2 > 0$, there are two, different, real-valued roots for the characteristic polynomial. The trajectory of the particle is then determined by Eq. (5.23), rewritten in Eq. (5.49) for convenience. The roots r_1 and r_2 are as in Eq. (5.10), while γ is as in Eq. (5.9).

$$x^{(t)} = p + \frac{r_2 \cdot \left(p - x^{(0)}\right) - \left(p - x^{(1)}\right)}{\gamma} \cdot r_1^t + \frac{-r_1 \cdot \left(p - x^{(0)}\right) + \left(p - x^{(1)}\right)}{\gamma} \cdot r_2^t \tag{5.49}$$

Clearly, the particle converges exponentially if $\max\left(|r_1|,|r_2|\right)<1$, where $|\cdot|$ is the absolute value. Conversely, it diverges exponentially if $\max\left(|r_1|,|r_2|\right)>1$, which is in agreement with findings by Ozcan and Mohan (1998), (1999) for $w=1$ and $\phi>4$. Note that the original PSO system does not allow convergence, since $\max\left(|r_1|,|r_2|\right)<1$ is not possible for $w=1$ regardless of ϕ. In contrast, the incorporation of the inertia weight allows for exponential convergence. The steepness of the divergence or convergence depends on the initial conditions, the location of the attractor *p*, and the settings of ϕ and *w*.

5.2.3.2. Two complex conjugate roots

If $\gamma^2<0$, the roots of the characteristic polynomial are complex conjugates. The trajectory of the particle is then determined by Eq. (5.42) –rewritten in Eq. (5.50) for convenience–, where $\gamma'=\sqrt{-\gamma^2}$ and θ is as in Eq. (5.28).

$$x^{(t)} = p + \left(\sqrt{w}\right)^t \cdot \left(-\left(p - x^{(0)}\right) \cdot \cos\left(t \cdot \theta\right) + \left(\frac{\left(1 + w - \phi\right) \cdot \left(p - x^{(0)}\right) - 2 \cdot \left(p - x^{(1)}\right)}{\gamma'} \right) \cdot \sin\left(t \cdot \theta\right) \right) \tag{5.50}$$

In the same fashion as for real-valued roots, convergence occurs when the magnitudes of the roots are smaller than one. Recall that Eq. (5.23) is also valid here. If the roots are represented in polar coordinates as in Eq. (5.31), the module ρ of the complex roots has to be smaller than one to ensure convergence. Therefore, from Eq. (5.27), $w<1$ ensures convergence when the roots of the characteristic polynomial are complex conjugates. This can be observed at once in Eq. (5.42), which is reproduced in Eq. (5.50).

The region of the plane 'ϕ–w' where the roots are complex conjugates can be bounded by the two curves plotted in Fig. 5.1, which are obtained by setting $\gamma=0$ in Eq. (5.9). Thus, the complex region is defined as in Eq. (5.51), where the signs of the inequalities are obtained by setting $\gamma<0$. Refer to Appendix I for further details.

$$\boxed{(\phi+1)-2\cdot\sqrt{\phi}<w<(\phi+1)+2\cdot\sqrt{\phi}} \tag{5.51}$$

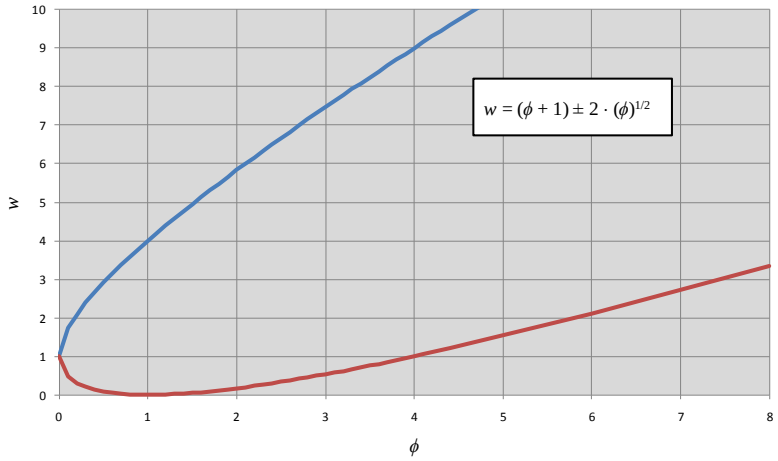


Fig. 5.1. Delimitation of the region in the 'ϕ–w' space where the roots of the characteristic polynomial are complex. Any combination 'ϕ–w' that falls between the two curves results in two complex conjugate roots.

Since only the sub-region where there is convergence is of practical interest, the part of the complex region where $w < 1$ is plotted in Fig. 5.2. Thus, the 'ϕ–w' pairs which lead to both complex roots and convergent behaviour are those enclosed by the red lines.

Since Ozcan and Mohan (1998), (1999) studied the algorithm without w, their analyses were limited to the horizontal line upper-bounding the complex region in Fig. 5.2. In that context, the real-valued roots would lead to divergence and should be discouraged or controlled. Therefore their studies were focused on $0<\phi<4$. By assuming $x^{(0)} = p$, they identified different search types according to the amplitude of the sine waves. Thus, for $0<\phi<2$, γ' increases with increasing ϕ. Therefore the amplitude of the sine wave decreases as ϕ increases, as can be seen in Eq. (5.50). Recall that the cosine term is dropped from their equation. Conversely, for $2<\phi<4$, γ' decreases with increasing ϕ, so that the amplitude of the sine wave increases as ϕ increases. In turn, $\gamma'<1$ for $0<\phi<\left(2-\sqrt{3}\right)$ and $\left(2+\sqrt{3}\right)<\phi<4$, so that the amplitudes are magnified. Notice that the more general case of $x^{(0)} \neq p$ affects this behaviour. While the theoretical studies by Ozcan and Mohan (1998), (1999) comprise a milestone in the formal study of the particles' trajectories in PSO, it is undesirable for the particles to surf the waves. Instead, an inertia weight $w < 1$ or constriction factor (Clerc & Kennedy, 2002) should be incorporated. More on the cyclic behaviour is discussed in section 5.3.

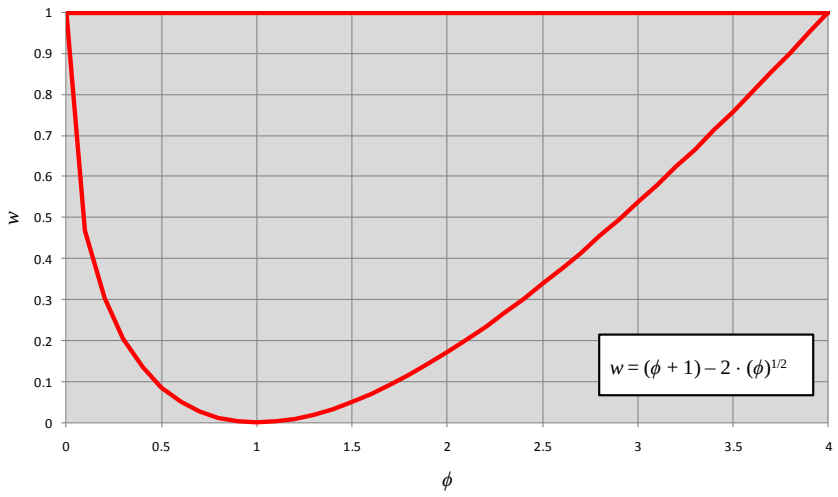


Fig. 5.2. Complex region in the 'ϕ–w' space where the roots are complex conjugates and convergence is ensured.

5.2.3.3. Only one root

If $\gamma^2 = 0$, there is a single root for the characteristic polynomial. The trajectory of the particle is then determined by Eq. (5.48), rewritten in Eq. (5.52).

$$x^{(t)} = p + \left[-\left(p - x^{(0)}\right) + \left(\left(p - x^{(0)}\right) - \frac{2 \cdot \left(p - x^{(1)}\right)}{1 + w - \phi} \right) \cdot t \right] \cdot \left(\frac{1 + w - \phi}{2} \right)^t \tag{5.52}$$

Therefore there is convergence for $\left| \frac{1 + w - \phi}{2} \right| < 1$. Furthermore, if $\frac{1 + w - \phi}{2} < 0$, the particle moves in opposite directions in consecutive time-steps.

While Ozcan and Mohan (1998), (1999) discussed the case of $\phi = 0$ as a special case, it has no practical interest as the particle would not be pulled by the attractor. The other special case with one single root they analyzed is the original PSO: $w = 1$ and $\phi = 4$. In the latter case, Eq. (5.52) becomes Eq. (5.53).

$$x^{(t)} = p + \left[-\left(p - x^{(0)}\right) + \left(\left(p - x^{(0)}\right) + \left(p - x^{(1)}\right) \right) \cdot t \right] \cdot (-1)^t \tag{5.53}$$

Thus, the particle moves in opposite directions in consecutive time-steps, diverging with linearly-increasing amplitudes. This is a boundary case, as smaller values of ϕ lead to cyclic behaviour while greater values lead to exponential explosion (for $w = 1$).

The set of 'ϕ–w' pairs that result in one single root of the characteristic polynomial are those plotted in Fig. 5.1. Those which also result in convergent behaviour comprise the lower bound of the complex region in Fig. 5.2. Notice that the particular cases of $\phi = 0$ and $\phi = 4$ comprise the two uppermost points in the parabola, where $w = 1$.

5.3. Deterministic explosion and cyclic behaviour

In a similar fashion as Ozcan and Mohan (1998), (1999), Clerc and Kennedy (2002) analyzed the trajectory of the deterministic particle for the algorithm without inertia weight. However, their studies aimed further, and a constriction factor was proposed to control the explosion. Their approach consisted of building a system of two recurrence relations of first order rather than one single recurrence relation of second order. Following their same train of thought, the equations are derived hereafter with the inclusion of the inertia weight. The following auxiliary variable is used:

$$y^{(t)} = p - x^{(t)} \tag{5.54}$$

Thus, the system of recurrence relations is as shown in Eq. (5.55):

$$\begin{cases} v^{(t)} = w \cdot v^{(t-1)} + \phi \cdot y^{(t-1)} \\ p - y^{(t)} = p - y^{(t-1)} + w \cdot v^{(t-1)} + \phi \cdot y^{(t-1)} \end{cases} \Rightarrow \quad \boxed{\begin{cases} v^{(t)} = w \cdot v^{(t-1)} + \phi \cdot y^{(t-1)} \\ y^{(t)} = -w \cdot v^{(t-1)} + (1-\phi) \cdot y^{(t-1)} \end{cases}} \tag{5.55}$$

The system can be expressed in matrix notation as in Eqs. (5.56) and (5.57):

$$P^{(t)} = \begin{bmatrix} v^{(t)} \\ y^{(t)} \end{bmatrix} = M \cdot P^{(t-1)} = M^{t} \cdot P^{(0)} \tag{5.56}$$

$$M = \begin{pmatrix} w & \phi \\ -w & (1-\phi) \end{pmatrix} \tag{5.57}$$

Given that

$$\det(M - I \cdot e) = \det\begin{pmatrix} w-e & \phi \\ -w & (1-\phi-e) \end{pmatrix} = e^2 + (\phi - w - 1) \cdot e + w = 0\,, \tag{5.58}$$

the eigenvalues of M are the same as the roots of the characteristic polynomial of the linear second order recurrence relation in Eq. (5.6), where γ is as in Eq. (5.9).

$$\boxed{\begin{aligned} e_1 &= \frac{(1+w)}{2} - \frac{\phi}{2} + \frac{\sqrt{\phi^2 - (2\cdot w+2)\cdot\phi + (w-1)^2}}{2} = \frac{(1+w)}{2} - \frac{\phi}{2} + \frac{\gamma}{2} \\ e_2 &= \frac{(1+w)}{2} - \frac{\phi}{2} - \frac{\sqrt{\phi^2 - (2\cdot w+2)\cdot\phi + (w-1)^2}}{2} = \frac{(1+w)}{2} - \frac{\phi}{2} - \frac{\gamma}{2} \end{aligned}} \tag{5.59}$$

Clerc and Kennedy (2002) derived the same equations for the particular case of $w = 1$:

$$e_1 = 1 - \frac{\phi}{2} + \frac{\sqrt{\phi^2 - 4\cdot\phi}}{2} = 1 - \frac{\phi}{2} + \frac{\gamma}{2} \quad ; \quad e_2 = 1 - \frac{\phi}{2} - \frac{\sqrt{\phi^2 - 4\cdot\phi}}{2} = 1 - \frac{\phi}{2} - \frac{\gamma}{2} \tag{5.60}$$

By diagonalizing the matrix of the system (M), Clerc and Kennedy (2002) showed that the position of a particle at any time-step would depend only on the initial conditions and on its eigenvalues raised to the power of the corresponding time-step. This is in agreement with the equations derived in section 5.2.2 (refer, in particular, to Eq. (5.23)).

Thus, including the inertia weight and following the same train of thought as in (Clerc & Kennedy, 2002), there exists a matrix A such that

$$A^{-1}\cdot M\cdot A = D = \begin{pmatrix} e_1 & 0 \\ 0 & e_2 \end{pmatrix} \tag{5.61}$$

provided there are two linearly independent eigenvectors associated to the eigenvalues in Eq. (5.59) (i.e. $\gamma \neq 0$), where

$$A = \begin{pmatrix} (1-w-\phi-\gamma) & (1-w-\phi+\gamma) \\ 2\cdot w & 2\cdot w \end{pmatrix} \tag{5.62}$$

Clearly, A^{-1} does not exist if $\gamma = 0$, as there would be only one family of eigenvectors:

$$e = \frac{1+w-\phi}{2} \quad (\text{for } \gamma = 0) \tag{5.63}$$

Refer to Appendix I for further details on Eqs. (5.61) and (5.62). Thus, from Eq. (5.61),

$$M = A \cdot D \cdot A^{-1} \tag{5.64}$$

Including Eq. (5.64) into Eq. (5.56),

$$P^{(t)} = A \cdot D \cdot A^{-1} \cdot P^{(t-1)} \quad \Rightarrow \quad A^{-1} \cdot P^{(t)} = D \cdot A^{-1} \cdot P^{(t-1)} \tag{5.65}$$

Therefore, calling $Q^{(t)} = A^{-1} \cdot P^{(t)}$,

$$Q^{(t)} = D \cdot Q^{(t-1)} \quad \Rightarrow \quad \boxed{Q^{(t)} = D^t \cdot Q^{(0)}} \tag{5.66}$$

Eq. (5.66) implies that the position of the particle at any time-step depends on the initial conditions and settings ($x^{(0)}$, $v^{(0)}$, w and ϕ within $Q^{(0)}$) –which are constants in t–, and on the eigenvalues raised to the power of the corresponding time-step: e_1^t and e_2^t. Hence there is convergence if both eigenvalues are smaller than one and divergence if at least one of them is greater than one. The question is what happens if $|e_1| = |e_2| = 1$.

Case 1: $e_1 = e_2$

Then $\gamma = 0$ and A is singular so that A^{-1} does not exist and Eqs. (5.61) to (5.66) do not apply. Eq. (5.48) shows that, in general, the particle diverges linearly. If $e_1 = e_2 = -1$ as in the particular case of w = 1 and ϕ = 4, the particle diverges by moving in opposite directions in consecutive time-steps. Clerc and Kennedy (2002) showed that if $P^{(0)}$ is an eigenvector, the particle oscillates between two positions (no explosion): $P^{(t+1)} = -P^{(t)}$. The same is true in Eq. (5.48) if the two terms multiplying t cancel each other out.

Case 2: $e_1 = -e_2$

From Eq. (5.59),

$$w = \phi - 1 \quad \Rightarrow \quad \boxed{e_i = \pm\sqrt{(1-\phi)}} \tag{5.67}$$

If $\phi > 1$, $e_1 = -e_2$ is imaginary. If, in addition, $|e_1| = |e_2| = 1$, then $\phi = 2$ and $e_1 = -e_2 = i$.

Case 3: $|e_1| = |e_2|$ with $e_1 \neq e_2$ and $e_1 \neq -e_2$

The eigenvalues are complex conjugates. If, in addition, $|e_1| = |e_2| = 1$,

$$|e_i| = \rho = \sqrt{\left(\frac{1+w-\phi}{2}\right)^2 + \left(\frac{\gamma'}{2}\right)^2} = 1 \qquad \Rightarrow \qquad \boxed{w=1} \tag{5.68}$$

which is the horizontal line upper-bounding the complex region of interest in Fig. 5.2. That is, the original PSO ($w=1$ and $0<\phi<4$):

$$\rho = \sqrt{\left(\frac{2-\phi}{2}\right)^2 + \left(\frac{\sqrt{4\cdot\phi-\phi^2}}{2}\right)^2} = 1 \qquad ; \qquad \forall 0<\phi<4 \ \wedge \ w=1 \tag{5.69}$$

Therefore the second case ($e_1 = -e_2$) comprises an instance of the third case.

If the eigenvalues are complex, they can be represented in polar coordinates, in the same fashion as the roots in Eqs. (5.32) and (5.33)). For $\rho = 1$,

$$\boxed{\begin{aligned} e_1^{\,t} &= e^{t\cdot\theta\cdot i} = \cos(t\cdot\theta) + i\cdot\sin(t\cdot\theta) \\ e_2^{\,t} &= e^{-t\cdot\theta\cdot i} = \cos(t\cdot\theta) - i\cdot\sin(t\cdot\theta) \end{aligned}} \tag{5.70}$$

Thus, following the studies in (Clerc & Kennedy, 2002), the original PSO ($w=1$) exhibits a cyclic or pseudo-cyclic behaviour for $0 < \phi < 4$.

To summarize, if at least one of the eigenvalues has a module greater than one, there is a deterministic divergence (explosion); if both modules are smaller than one, there is convergence; if $|e_1| = |e_2| = 1$ and $e_1 = e_2$, there is a deterministic explosion (exceptional initial conditions may lead to unlikely cyclic behaviour); and finally, if $|e_1| = |e_2| = 1$ and $e_1 \neq e_2$, the eigenvalues are complex conjugates and the trajectory is (pseudo) cyclic.

This thesis is not concerned with the particle's cyclic behaviour per se, as it is in fact argued that it should be discouraged. Instead, convergence should be ensured by means of inertia weights and/or constriction factors. However, given that the cyclic behaviour is a boundary case between convergent and divergent behaviour, some analyses and empirical studies on the subject are offered hereafter.

The (pseudo) cyclic behaviour takes place only for $w=1$ and $0<\phi<4$, where the roots of the characteristic polynomial are complex conjugates with $\rho = 1$. In turn, $\phi = 4$

leads –in general– to linear explosion, and $\phi > 4$ results in exponential explosion. In order to illustrate this and to visually analyze the influence of ϕ on the divergent and cyclic behaviours, a sequential series of plots of the trajectory of the deterministic particle for $w = 1$ and $0.2 < \phi < 4.1$ is offered in Fig. 5.3 to Fig. 5.7. The particle is initialized at $x^{(0)} = 100$, $v^{(0)} = 0$, and $p = 0$. Thus, the exponential explosion can be observed in Fig. 5.3 A); the linear explosion in Fig. 5.3 B); and different cyclic and pseudo-cyclic behaviours in the remaining figures. Recall that $\phi = aw$ if the random weights are removed whereas $\phi = aw/2$ if they are replaced by their expected value (see Eq. (5.3)).

Clerc and Kennedy (2002) identified the values of ϕ which result in cyclic behaviour, stating that every other value leads to pseudo-cyclic behaviour. Some of those values are shown in Table 5.1, where the associated values of the smallest period, cycle, θ, and γ' are also provided. The rows highlighted are those which were also identified by Ozcan and Mohan (1998), (1999) as limits for different search types. Thus, for $\phi < 2 - \sqrt{3}$ and $\phi > 2 + \sqrt{3}$ it happens that $\gamma' < 1$, and the opposite is true in between. For the particular case they studied –where $x^{(0)} = p$ and therefore the cosine term in Eq. (5.42) drops–, values of $\gamma' < 1$ result in the magnification of the amplitude of the sine waves. A precise study of the characteristics of the sine waves for varying values of ϕ is of no interest in this study, as the cyclic behaviour is to be discouraged.

The trajectories in Fig. 5.3 to Fig. 5.7 which happen to be exactly cyclic are framed within red lines, and the values of the period, cycle, and θ are provided.

Table 5.1. Values of ϕ < 4 that lead to exact cyclic behaviour (for w =1), identified in (Clerc & Kennedy, 2002). The values highlighted are those delimiting the regions of different search type identified in (Ozcan & Mohan, 1998) and (Ozcan & Mohan, 1999) for the particular case of $x^{(0)} = p$ (the cosine term drops off).

ϕ	smallest *period*	*cycle*	θ	γ'
$2 - 3^{1/2}$	**12**	**2π**	**1/6 π**	**1**
1	6	2π	1/3 π	1.73205
$(5 - 5^{1/2}) / 2$	5	2π	2/5 π	1.90211
2	**4**	**2π**	**1/2 π**	**2**
3	3	2π	2/3 π	1.73205
$(5 + 5^{1/2}) / 2$	5	4π	4/5 π	1.17557
$2 + 3^{1/2}$	**12**	**10π**	**5/6 π**	**1**

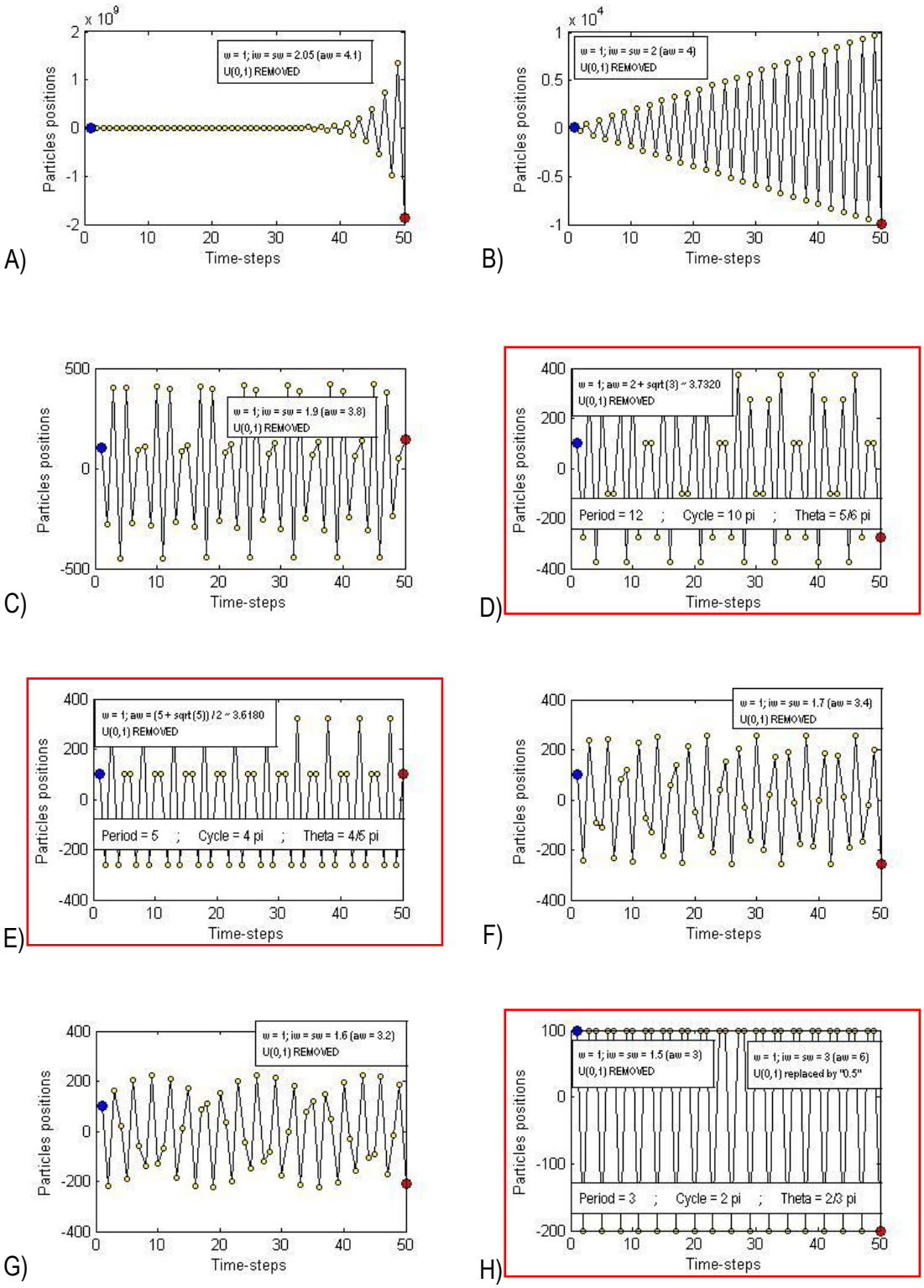


Fig. 5.3. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ removed for w = 1.00 and 4.10 ≥ aw ≥ 3.00. Cyclic trajectories are highlighted by a red frame. Figures A) and B) feature the exponential and linear explosions, respectively, whereas the remaining figures depict cyclic or pseudo-cyclic behaviour.

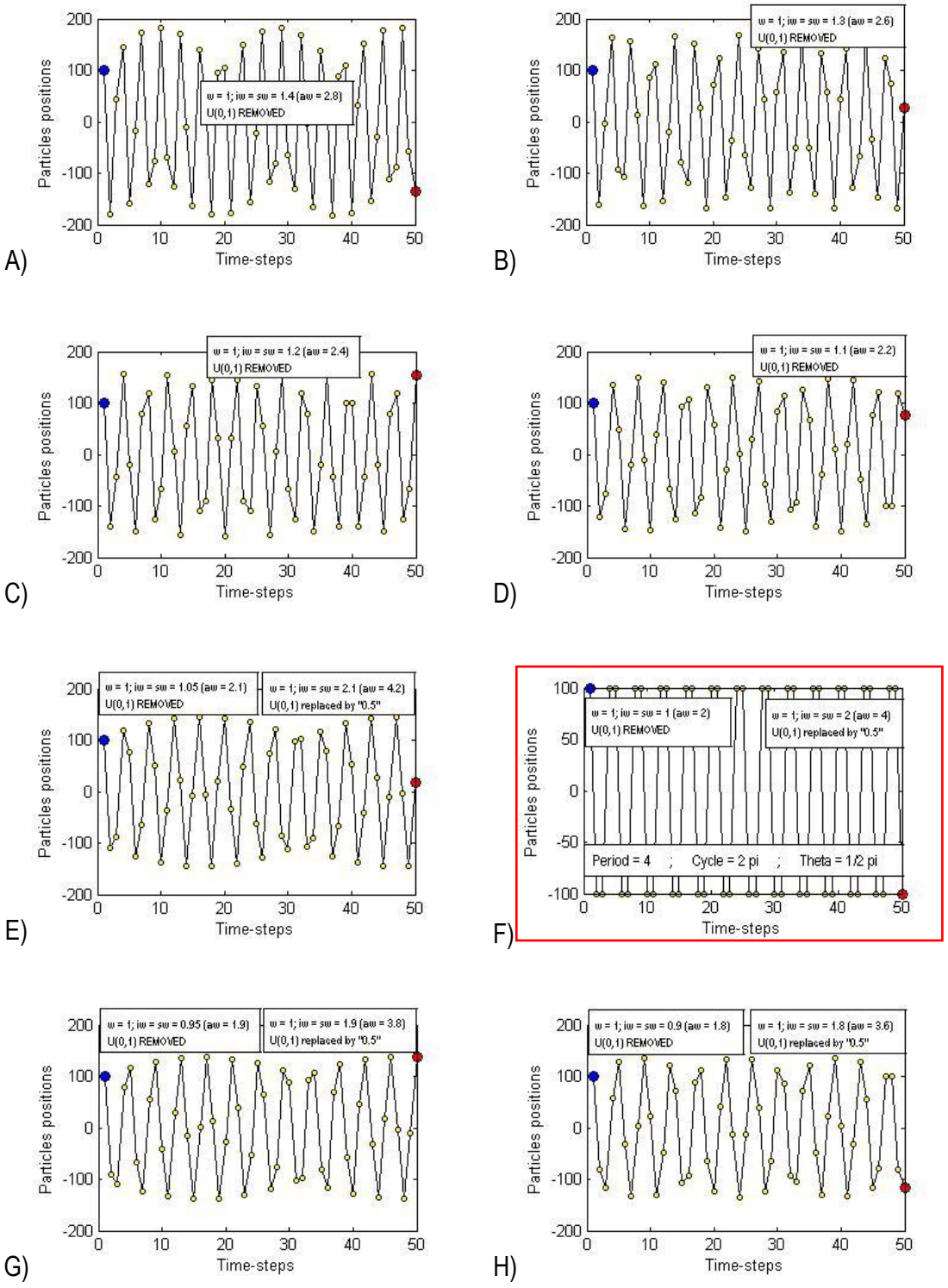


Fig. 5.4. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ removed for w = 1.00 and 2.80 ≥ aw ≥ 1.80. Cyclic trajectories are highlighted by a red frame.

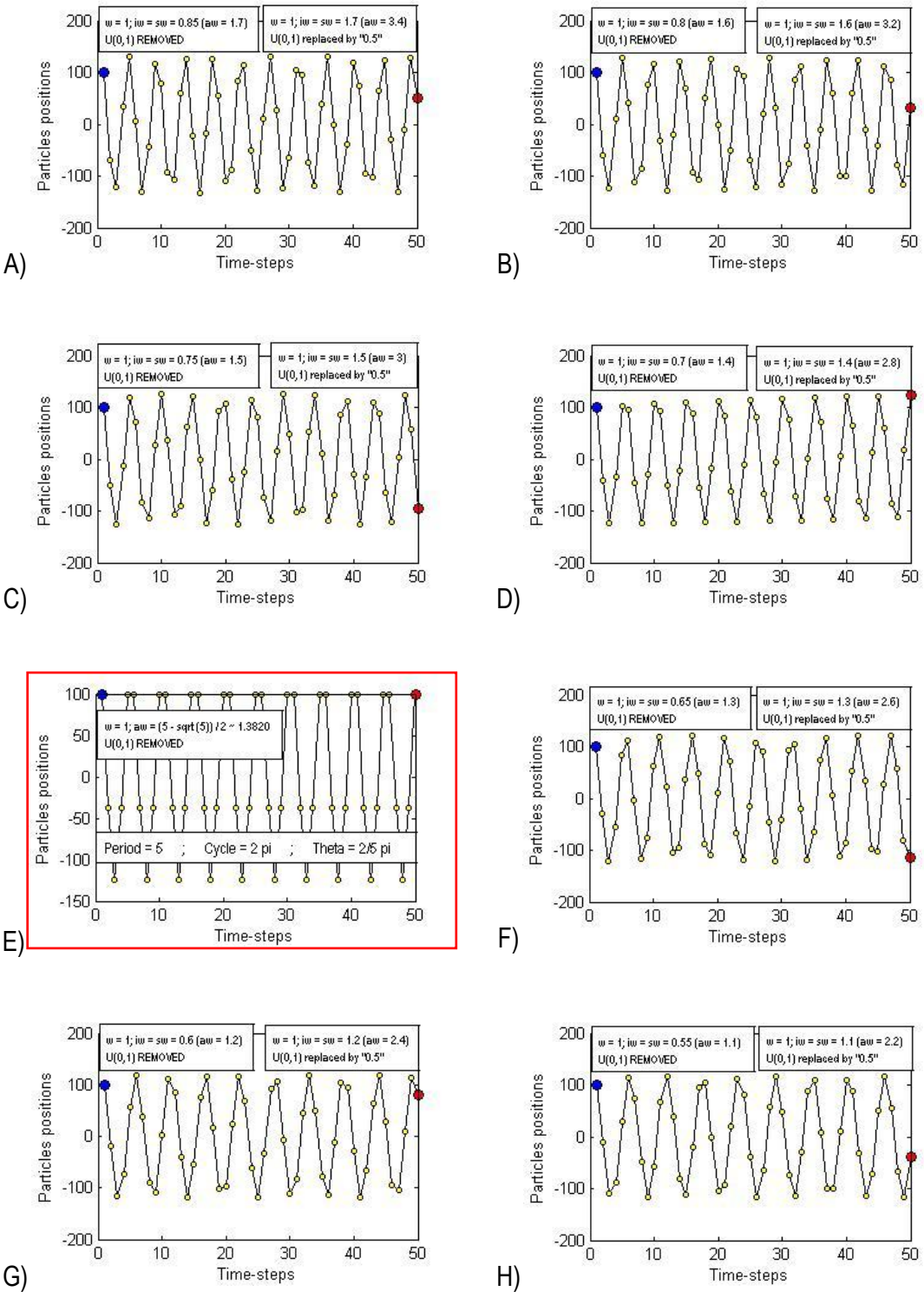


Fig. 5.5. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ removed for w = 1.00 and 1.70 ≥ aw ≥ 1.10. Cyclic trajectories are highlighted by a red frame.

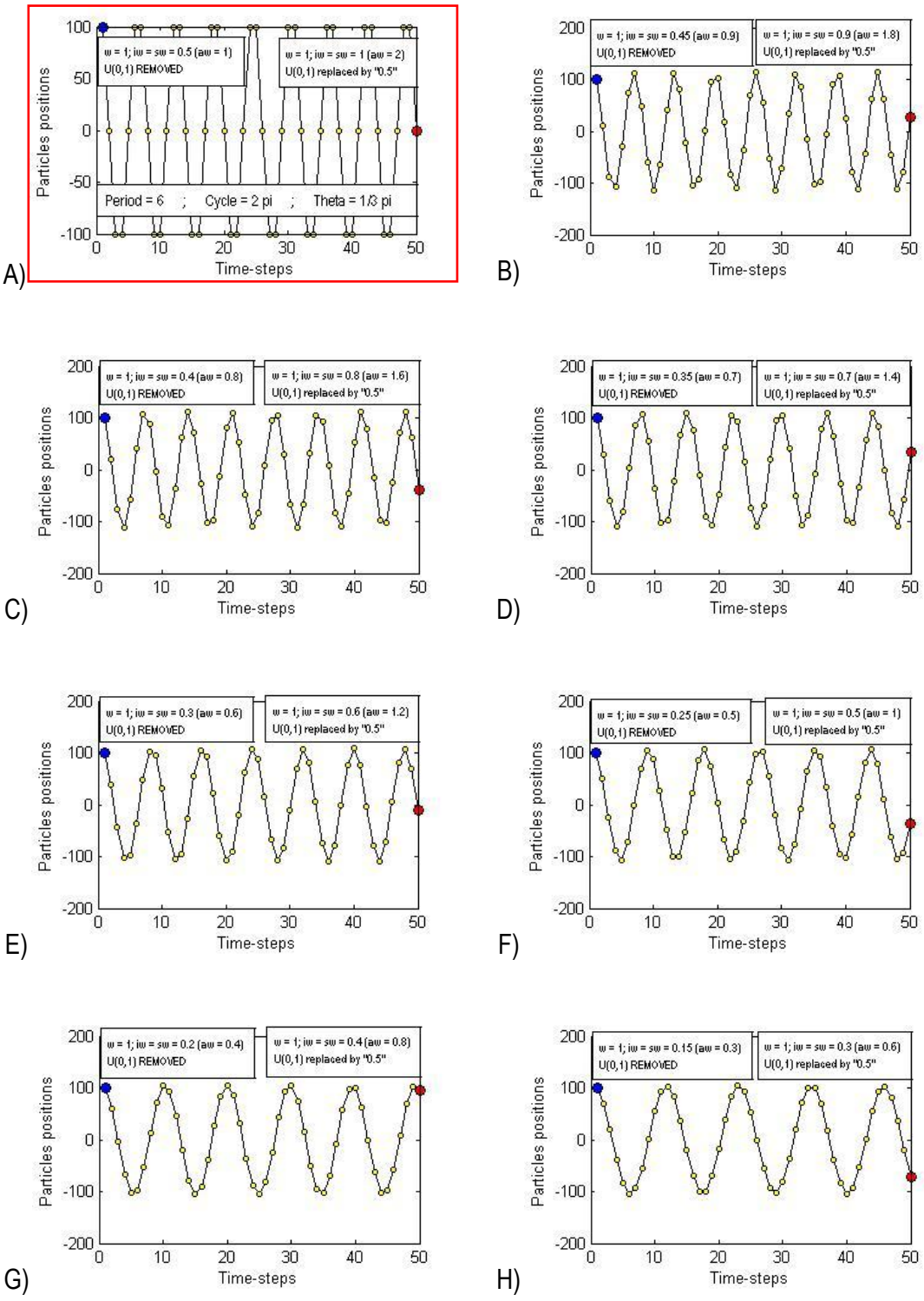


Fig. 5.6. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ removed for w = 1.00 and 1.00 ≥ aw ≥ 0.30. Cyclic trajectories are highlighted by a red frame.

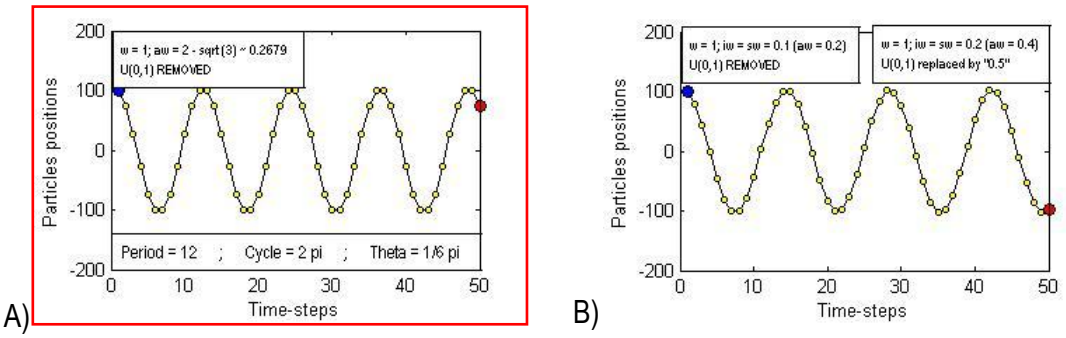


Fig. 5.7. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ removed for w = 1.00, aw = $2-\sqrt{3}$ in figure A), and aw = 0.2 in figure B). The cyclic behaviour on the left is highlighted by a red frame.

Table 5.1 is extended in Table 5.2 to include greater multiple periods and cycles for which the values of ϕ also lead to cyclic behaviour. Given ϕ, from Eq. (5.29) and for $w=1$: $\theta = \text{acos}\left(1-\frac{\phi}{2}\right)$ and $period = \frac{cycle}{\theta} = \frac{2\cdot\pi\cdot k}{\theta}$, where k is an integer. In turn, γ' is as in Eq. (5.25). The curves '$\phi - period$' are plotted in Fig. 5.8 for five values of k. Of course, integer periods imply cyclic behaviour. Note, for instance, the value $\phi = 2$, which results in cyclic trajectory for all five cycles, with the smallest *period* being 4 time-steps for a cycle of 2π. This can also be observed in Fig. 5.4 F), where the whole cycle is completed in 4 time-steps. Notice that $\gamma' = 2$ in this particular case, which means that the root is imaginary ($\theta = \pi/2$).

Table 5.2. Values of ϕ < 4 that lead to exact cyclic behaviour (for w =1) and related data, identified in (Clerc & Kennedy, 2002).

ϕ	*period*	*cycle*	θ	γ'
$2 - 3^{1/2}$	12, 24, ...	2π, 4π, ...	1/6 π	1
1	6, 12, 18, 24, 30, ...	2π, 4π, 6π, 8π, 10π, ...	1/3 π	1.73205
$(5 - 5^{1/2}) / 2$	5, 10, 15, 20, 25, ...	2π, 4π, 6π, 8π, 10π, ...	2/5 π	1.90211
2	4, 8, 12, 16, 20, ...	2π, 4π, 6π, 8π, 10π, ...	1/2 π	2
3	3, 6, 9, 12, 15, ...	2π, 4π, 6π, 8π, 10π, ...	2/3 π	1.73205
$(5 + 5^{1/2}) / 2$	5, 10, ...	4π, 8π, ...	4/5 π	1.17557
$2 + 3^{1/2}$	12, ...	10π, ...	5/6 π	1

From here on, *deterministic explosion* will refer to the divergence observed in Fig. 5.3 A) and B). That is, when the divergence is due to at least one of the roots of the characteristic polynomial –the eigenvalues of system matrix– being greater than one.

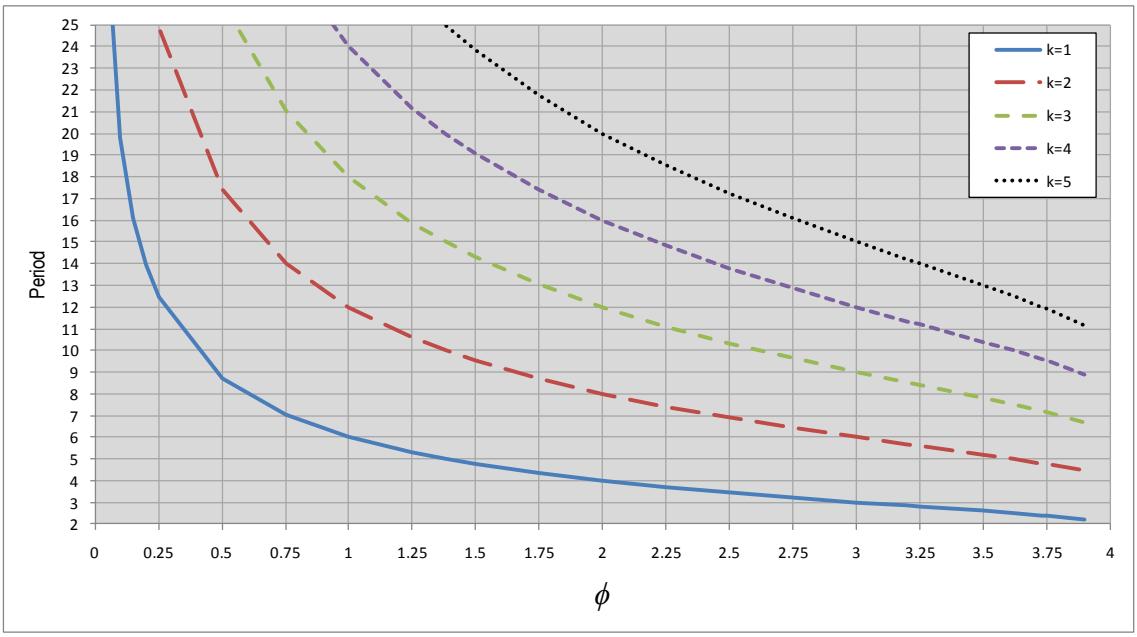


Fig. 5.8. Periods associated to $0 < \phi < 4$. The different values of k identify the number of '2π' involved in the cycle. For instance, for $\phi = 2$, the period equals 4 time-steps in a 2π cycle and 20 time-steps in a 10π cycle.

As to the (pseudo) cyclic behaviour, it can be observed that even when the trajectory is perfectly cyclic such us the cases in Fig. 5.3 D), E), and H), the particle explores regions of the search-space that are far from the attractor. In other words, although the deterministic explosion does not take place, the pulling influence of the attractor alone (i.e. disregarding the inertia weight) over the particle ends up taking it farther rather than closer to itself, as illustrated in Fig. 5.9. Common sense throws doubt on the usage of $\phi > 2$ (bear in mind the difference between ϕ and aw, as posed in Eq. (5.3)).

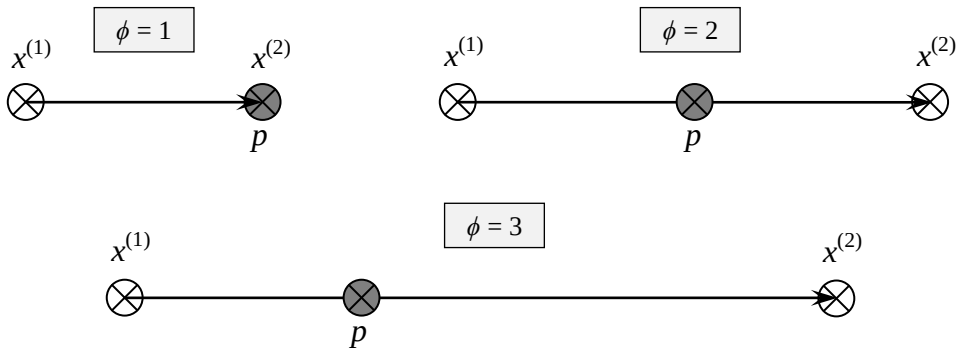


Fig. 5.9. Influence of the coefficients $\phi = 1$, $\phi = 2$, and $\phi = 3$ on the particle's trajectory between two consecutive time-steps, disregarding the inertia weight.

5.4. Stochastic explosion

If once the random weights are re-incorporated into the algorithm an explosion occurs despite the magnitude of both roots/eigenvalues being smaller than one, such divergence is referred to here as a 'stochastic explosion'.

In the original PSO as proposed in (Kennedy & Eberhart, 1995), the settings are $w=1$ and $iw=sw=2$. This leads to the explosion of the system. Since $\phi\in[0,4]$, this cannot be the 'deterministic explosion' discussed in section 5.3 but a 'stochastic explosion'. Given that the system is unstable for cyclic behaviour, it is not too surprising that a perturbation like the introduction of randomness results in the collapse of the system. However, this 'stochastic explosion' can still be observed for $w<1$ and ϕ fully within the convergence region. In either case, it is not proved that the divergence would continue indefinitely. The stochastic explosions that are clearly temporary, with the particle being pulled back towards the attractor, will be referred to as 'local explosions'.

Theoretical studies of the PSO algorithm's behaviour in the presence of randomness are beyond the scope of this thesis. Only few authors, to the best of my knowledge, dare take this challenge. Jiang et al. (2007) studied the convergence of an isolated particle using stochastic process theory, viewing the particles' position as a stochastic vector. By studying the convergence of the expectation and of the variance of the particle's position, they claim to have derived the 'stochastic convergent condition' of the particle swarm system. Clerc (2006b) studied the stagnation phenomenon in PSO (no improvement observed over several time-steps). In that extensive formal study, he analyzed the distribution of velocities of a particle with stochastic forces. In turn, Poli (2008) presented a method to determine the characteristics of the sampling distribution of a PSO algorithm, and its changes as particles search for better individual best experiences.

Six trajectories of the particle with $w=1$; $aw=4$; random weights re-incorporated; and stationary attractors are shown in Fig. 5.10. The first one is obtained by setting the pseudo-random number generator in Matlab to its original state, and the others comprise consecutive runs. As can be observed, there is a stochastic explosion rather than a cyclic average trajectory as could be expected given that $\phi\in[0,4]$ (cyclic range). It can also be seen that the steepness and size of the stochastic explosion varies from run to run.

Due to the reasons illustrated in Fig. 5.9, this thesis does only exceptionally concern itself with values of $\phi_{\text{mean}} > 2$. That is, in general, $0 < \phi_{\text{mean}} \leq 2$ ($0 < aw \leq 4$).

The main objective in this section is to illustrate how the random weights may lead to a stochastic explosion as the trajectories of the particle with the random weights replaced by their expected value '0.5' are (pseudo) cyclic. Thus, Fig. 5.11 to Fig. 5.15 offer direct comparisons of the trajectories with the random weights included on the right columns and with their expected values on the left ones. Two graphs in the same row share the same acceleration weight.

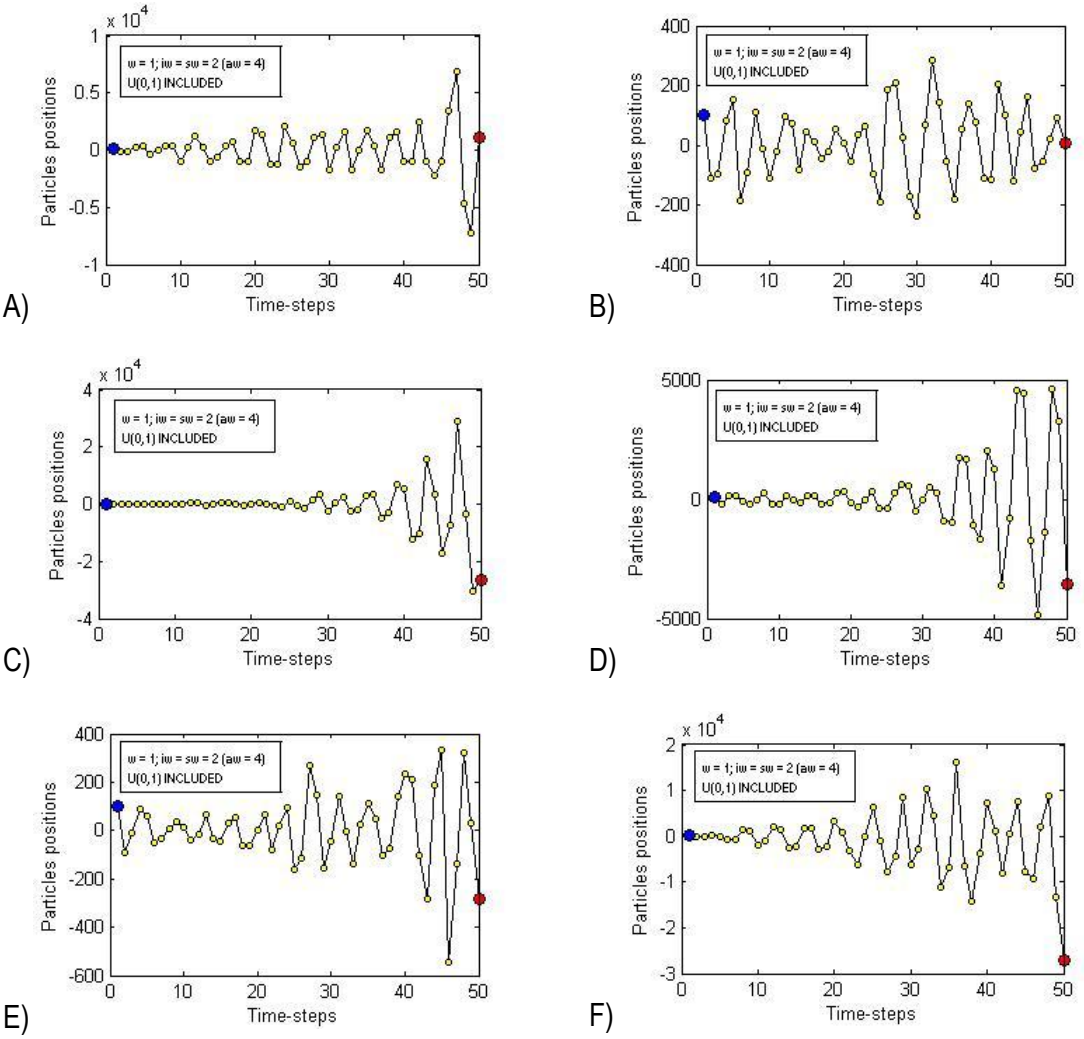


Fig. 5.10. Six possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ included for w = 1.00 and aw = 4.00. The first one corresponds to the first run once the pseudo-random number generator is set to its initial state, while the others correspond to consecutive runs.

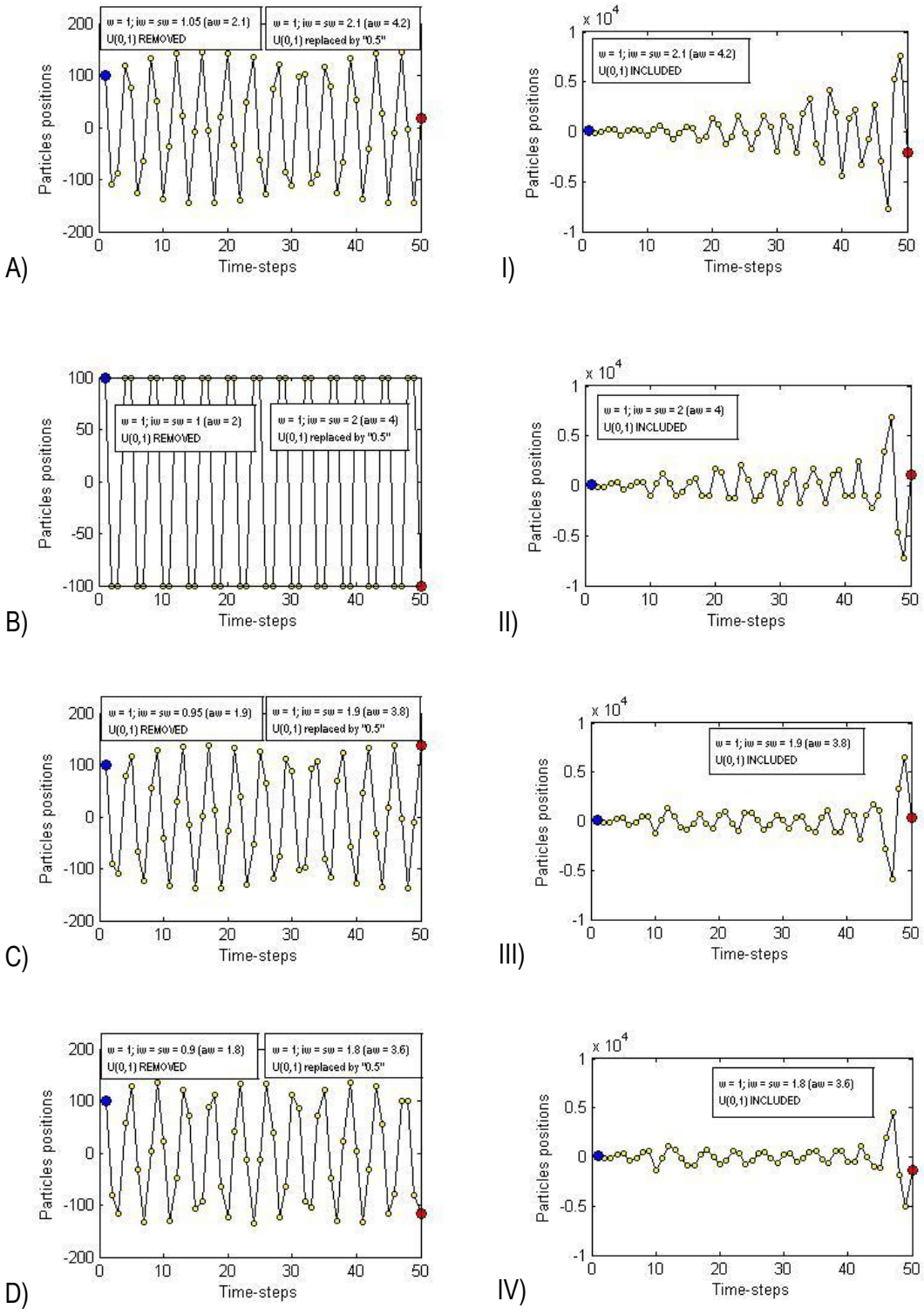


Fig. 5.11. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 1.00 and 4.20 ≥ aw ≥ 3.60. The random weights $U_{(0,1)}$ are replaced by their expected value (i.e. '0.5') in the first column, and included in the second one. The equivalent values of the coefficients if the random weights were removed instead are presented in the left text box on the left column.

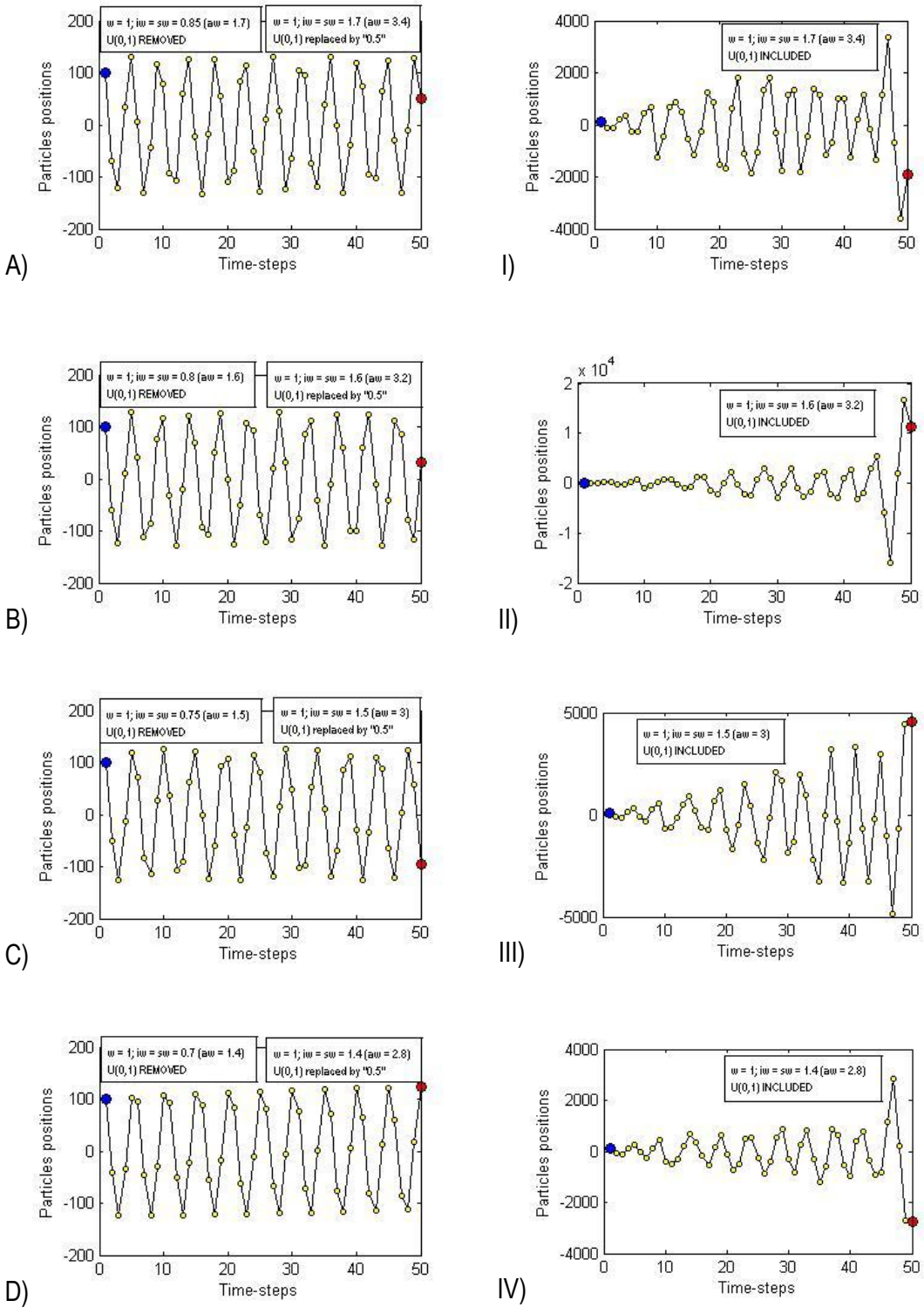


Fig. 5.12. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 1.00 and 3.40 ≥ aw ≥ 2.80. The random weights $U_{(0,1)}$ are replaced by their expected value (i.e. '0.5') in the first column, and included in the second one. The equivalent values of the coefficients if the random weights were removed instead are presented in the left text box on the left column.

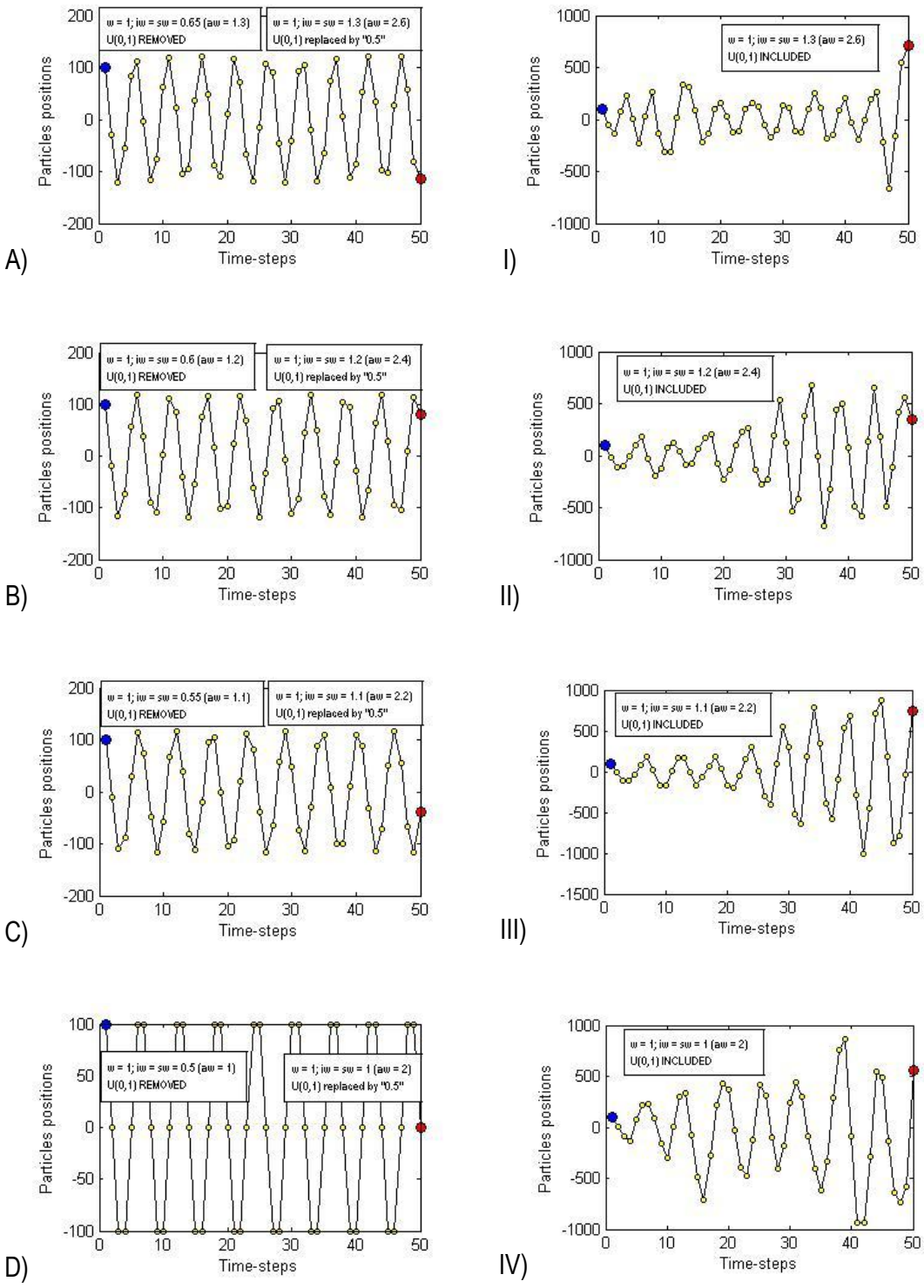


Fig. 5.13. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 1.00 and 2.60 ≥ aw ≥ 2.00. The random weights $U_{(0,1)}$ are replaced by their expected value (i.e. '0.5') in the first column, and included in the second one. The equivalent values of the coefficients if the random weights were removed instead are presented in the left text box on the left column.

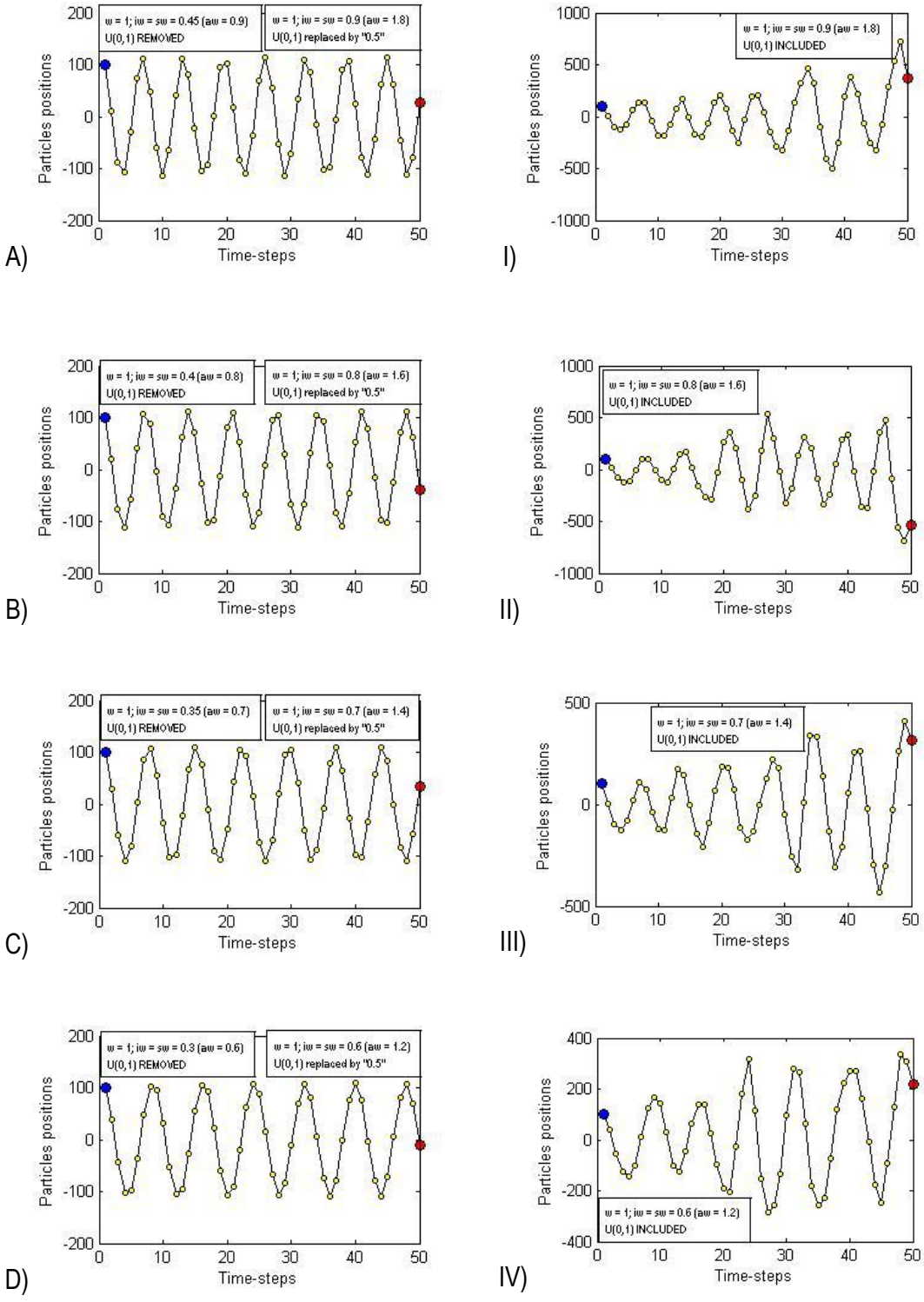


Fig. 5.14. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 1.00 and 1.80 ≥ aw ≥ 1.20. The random weights $U_{(0,1)}$ are replaced by their expected value (i.e. '0.5') in the first column, and included in the second one. The equivalent values of the coefficients if the random weights were removed instead are presented in the left text box on the left column.

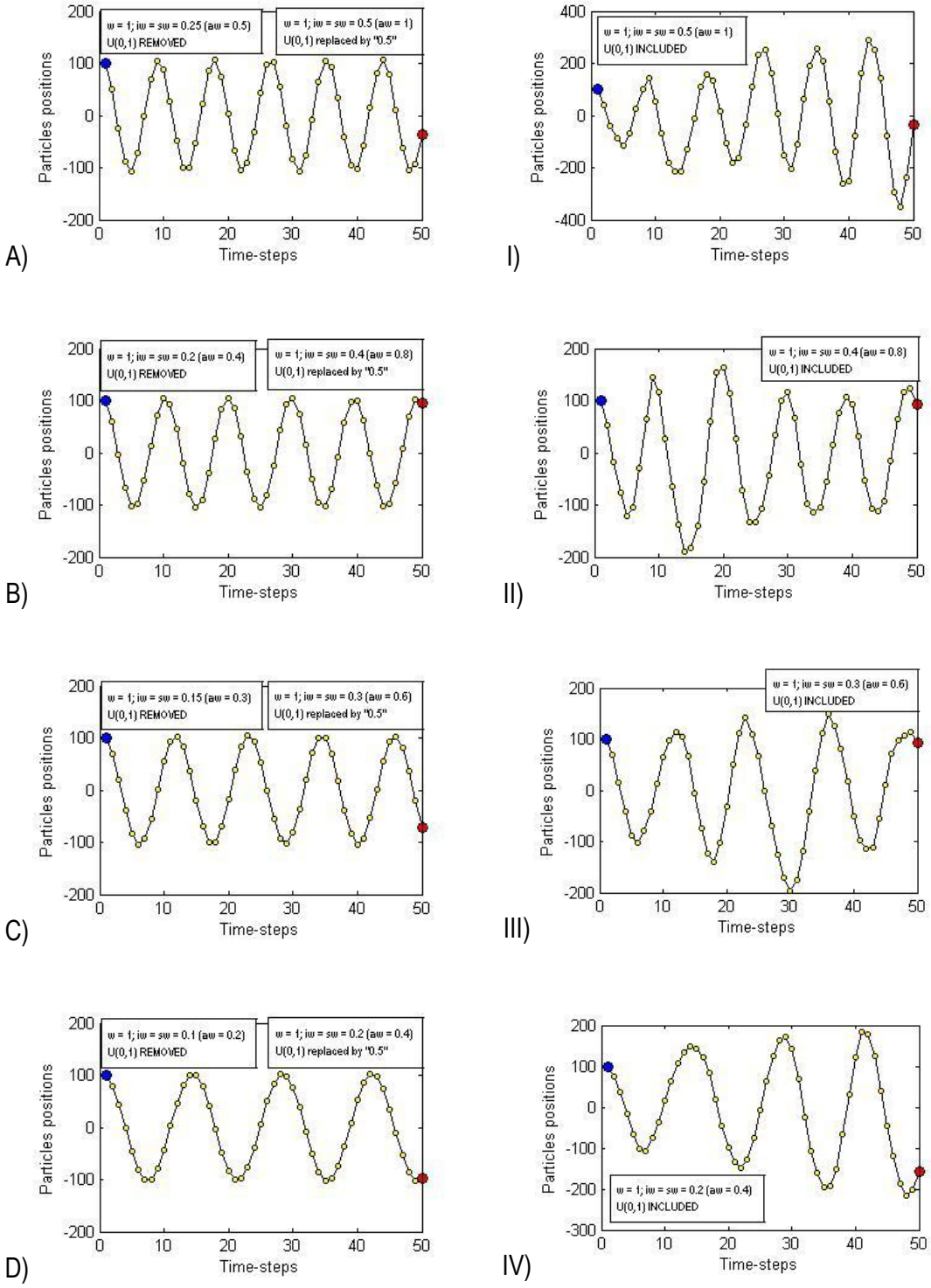


Fig. 5.15. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 1.00 and 1.00 ≥ aw ≥ 0.40. The random weights $U_{(0,1)}$ are replaced by their expected value (i.e. '0.5') in the first column, and included in the second one. The equivalent values of the coefficients if the random weights were removed instead are presented in the left text box on the left column.

Fig. 5.11 to Fig. 5.15 show that the stochastic explosion occurs for the whole range of ϕ studied, although its size tends to decrease as ϕ decreases. This is to be expected, as smaller values of ϕ reduce the influence of randomness. The figures with random weights included were chosen to be representative of the most common behaviour, as different runs return different trajectories. Six runs are performed for every trajectory involving random weights presented in this chapter, and some others. All those images are gathered in a digital appendix. Fig. 5.16 shows six trajectories of the particle with $w = 1$; $aw = 3.4$; random weights re-incorporated; and stationary attractors.

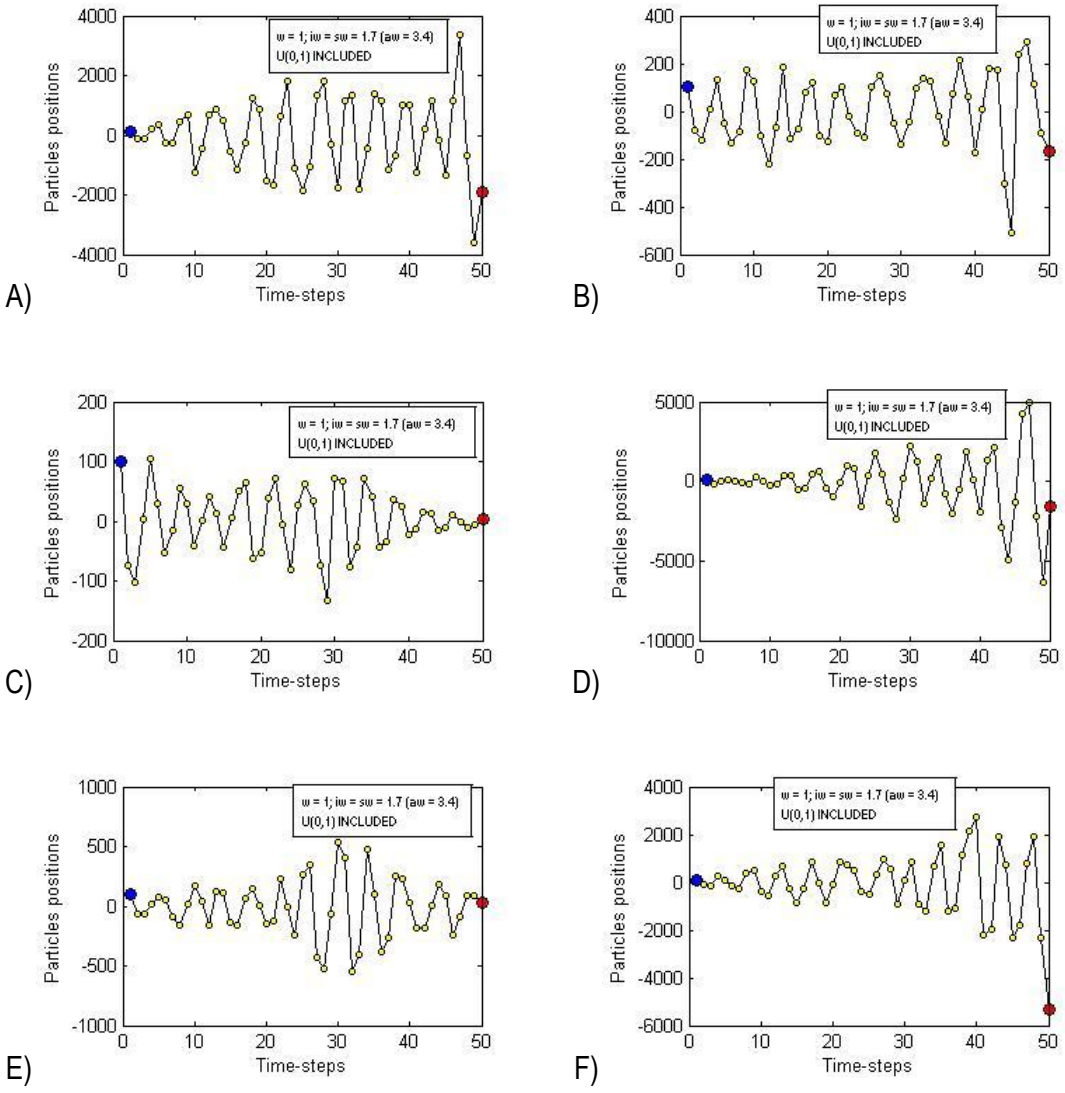


Fig. 5.16. Six possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ included for w = 1.00 and aw = 3.40. The first one corresponds to the first run once the pseudo-random number generator is set to its initial state, while the others correspond to consecutive runs.

5.5. Convergence graph

As discussed throughout section 5.2, convergence occurs when the magnitude of both roots of the characteristic polynomial (eigenvalues of the system matrix *M*) are smaller than one. Refer, in particular, to Eqs. (5.23), (5.42), and (5.48). Therefore, for real-valued roots, convergence is ensured if:

$$\boxed{-1<\frac{1+w-\phi\pm\sqrt{\phi^2-(2\cdot w+2)\cdot\phi+(w-1)^2}}{2}<1} \tag{5.71}$$

Thus, from the rightmost inequality,

$$1+w-\phi\pm\sqrt{\phi^2-(2\cdot w+2)\cdot\phi+(w-1)^2}<2 \tag{5.72}$$

$$-1+w-\phi\pm\sqrt{\phi^2-(2\cdot w+2)\cdot\phi+(w-1)^2}<0 \tag{5.73}$$

The worst case scenario in Eq. (5.73) is for the positive square root. That is to say that if the positive square root satisfies Eq. (5.73), so does the negative one. Therefore,

$$(-1+w-\phi)<-\sqrt{\phi^2-(2\cdot w+2)\cdot\phi+(w-1)^2} \tag{5.74}$$

$$(-1+w-\phi)^2>\left(-\sqrt{\phi^2-(2\cdot w+2)\cdot\phi+(w-1)^2}\right)^2 \tag{5.75}$$

$$\boxed{\phi>0} \tag{5.76}$$

Note that for Eq. (5.74) to hold, it must be true that $(-1+w-\phi)<0$. Hence,

$$\boxed{w<\phi+1} \tag{5.77}$$

It follows that satisfying Eqs. (5.76) and (5.77) guarantees that the two (real-valued) roots of the characteristic polynomial satisfy the rightmost inequality in Eq. (5.71).

From the leftmost inequality in Eq. (5.71),

$$1+w-\phi\pm\sqrt{\phi^2-(2\cdot w+2)\cdot\phi+(w-1)^2}>-2 \tag{5.78}$$

$$3+w-\phi\pm\sqrt{\phi^2-(2\cdot w+2)\cdot\phi+(w-1)^2}>0 \tag{5.79}$$

The worst case scenario in Eq. (5.79) is for the negative square root. That is to say that if the negative square root satisfies Eq. (5.79), so does the positive one. Therefore,

$$(3+w-\phi)>\sqrt{\phi^2-(2\cdot w+2)\cdot\phi+(w-1)^2} \tag{5.80}$$

$$(3+w-\phi)^2>\left(\sqrt{\phi^2-(2\cdot w+2)\cdot\phi+(w-1)^2}\right)^2 \tag{5.81}$$

$$\boxed{w>\frac{\phi}{2}-1} \tag{5.82}$$

Note that for Eq. (5.80) to hold, it must be true that $(3+w-\phi)>0$. Hence,

$$\boxed{w>\phi-3} \tag{5.83}$$

It follows that satisfying Eqs. (5.82) and (5.83) guarantees that the two (real-valued) roots of the characteristic polynomial satisfy the leftmost inequality in Eq. (5.71).

If the roots of the characteristic polynomial are complex conjugates rather than real-valued, the convergence condition is that their module is smaller than one. As shown in Eq. (5.27), the module equals $\rho=\sqrt{w}$, which translates into the convergence condition shown in Eq. (5.84) for complex conjugate roots:

$$\boxed{w<1} \tag{5.84}$$

Therefore, the set of inequality conditions bounding a region in the 'ϕ–w' plane of ensured convergence of the isolated, deterministic particle is offered in Eq. (5.85), while the boundaries of those inequalities are plotted in Fig. 5.17.

Considering the redundancies and that only Eq. (5.84) is valid within the complex region (inside the red parabola in Fig. 5.17), the conditions can be reduced to those shown in Eq. (5.86).

$$\begin{boxed}\begin{aligned}&\phi > 0\\&w < \phi + 1\\&w > \frac{\phi}{2} - 1\\&w > \phi - 3\\&w < 1\end{aligned}\end{boxed} \tag{5.85}$$

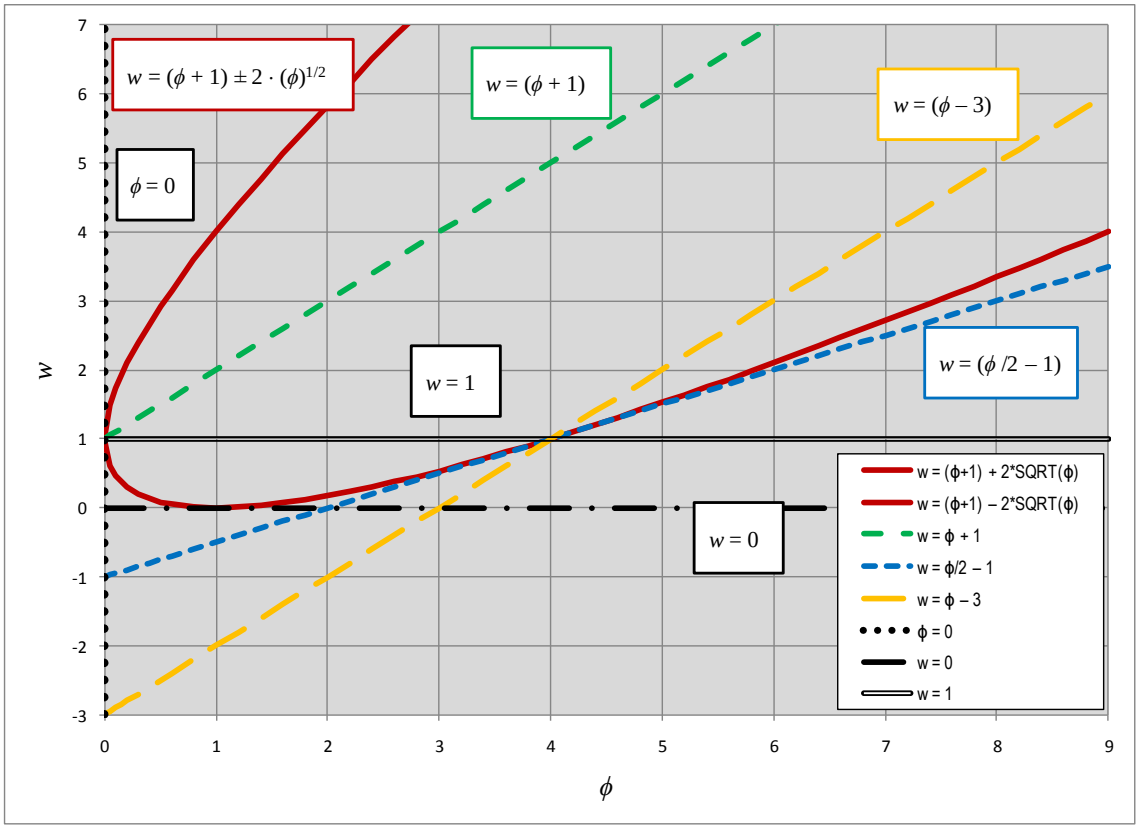


Fig. 5.17. Relevant curves bounding the convergence region in the 'ϕ–w' plane.

$$\begin{boxed}\begin{aligned}&\phi > 0\\&w > \frac{\phi}{2} - 1\\&w < 1\end{aligned}\end{boxed} \tag{5.86}$$

Thus, the convergence region is given by the blue-shaded triangle in Fig. 5.18.

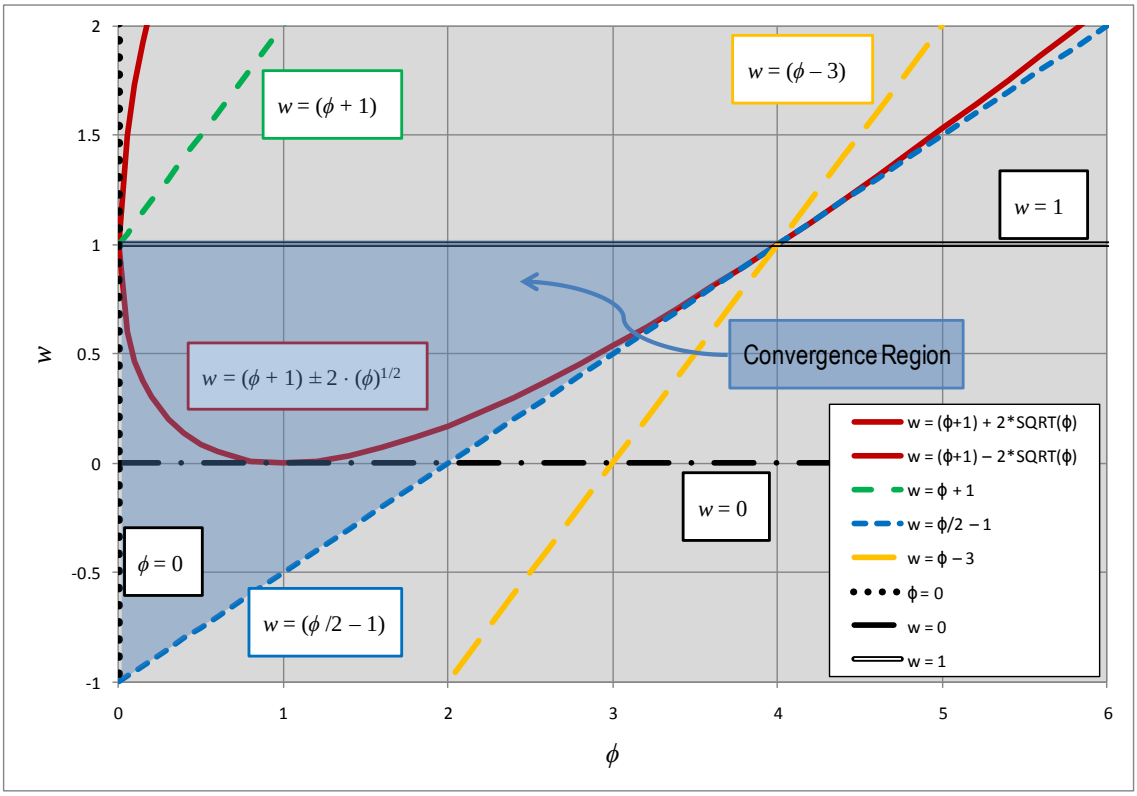


Fig. 5.18. Convergence region in the 'ϕ–w' plane (blue shaded triangle).

In general, $w<1$ is of no practical interest as it does not make much sense to speak of negative inertia. However, it is interesting to observe the how the particle converges nonetheless, as shown in Fig. 5.19 for '$w = –0.50$, $\phi = 0.25$' and '$w = –0.50$, $\phi = 0.75$'.

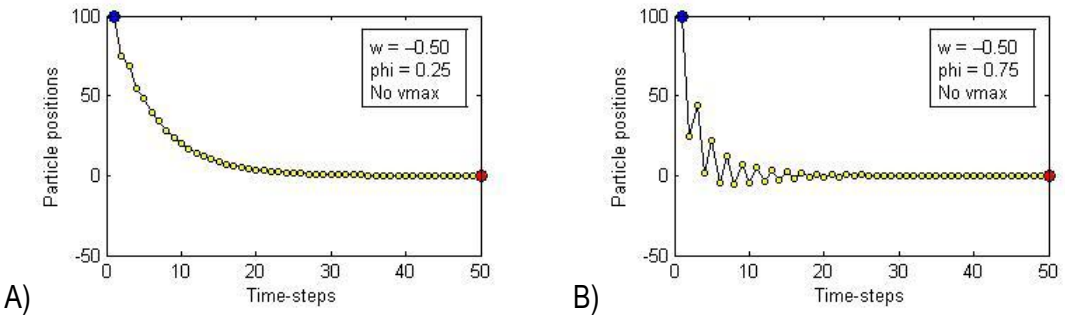


Fig. 5.19. Trajectory of a deterministic particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 'w = –0.50, ϕ = 0.25' and 'w = –0.50, ϕ = 0.75'. The particle converges despite the negative inertia.

Disregarding the region of the convergence triangle in Fig. 5.18 where $w < 1$, the convergence region of the 'ϕ–w' plane is reduced to the gray shaded area in Fig. 5.20.

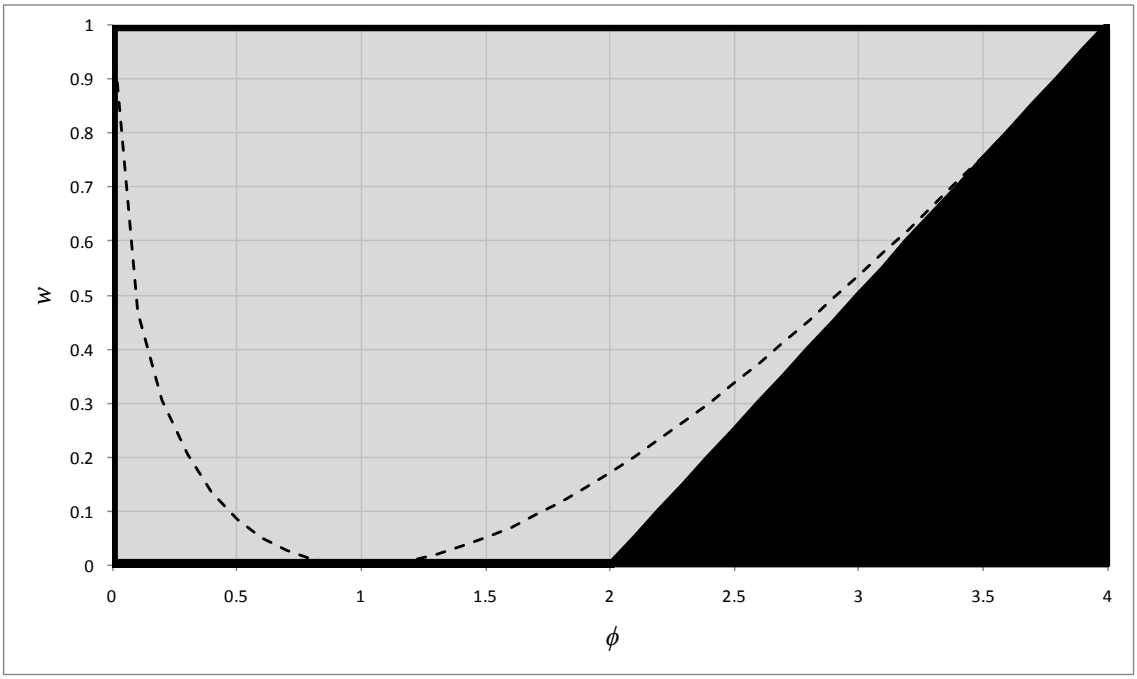


Fig. 5.20. Region in the 'ϕ–w' plane where convergence is ensured (gray shade). The region inside the dotted parabola comprises the complex roots of the characteristic polynomial which lead to convergence. Deterministic explosion is guaranteed within the black triangle. The top horizontal line with $\phi < 4$ leads to (pseudo) cyclic behaviour.

In summary, the convergence region of practical interest is given by the gray area in Fig. 5.20, which is delimitated by the inequalities offered in Eq. (5.87).

$$\boxed{\begin{aligned} &\phi > 0 \\ &w > \frac{\phi}{2} - 1 \\ &0 \le w < 1 \end{aligned}} \tag{5.87}$$

Ensured convergence is not the only important aspect of the particle's trajectory. The speed and form of convergence define the manner in which the search is carried out, and therefore have a critical impact on the final performance of the optimizer. Different combinations of w and ϕ within the convergence region result in different amplitudes, frequencies, and speed of damping of the oscillations in the particle's trajectories.

Twenty pairs 'ϕ–w' resulting from all combinations of five values of w and four values of ϕ are considered to illustrate the speed and form of convergence/divergence of the (deterministic) particle's trajectory associated to the different regions of the 'ϕ–w' plane. These selected values are shown in Table 5.3 and Fig. 5.21.

Table 5.3. Twenty selected pairs 'ϕ–w' to be considered to illustrate the speed and form of convergence/divergence of the particle's trajectory associated to the different regions of the 'ϕ–w' plane.

		ϕ			
		0.5	**2**	**3.5**	**4**
w	**1**	A1	A2	A3	A4
	0.8	B1	B2	B3	B4
	0.5	C1	C2	C3	C4
	0.2	D1	D2	D3	D4
	0	E1	E2	E3	E4

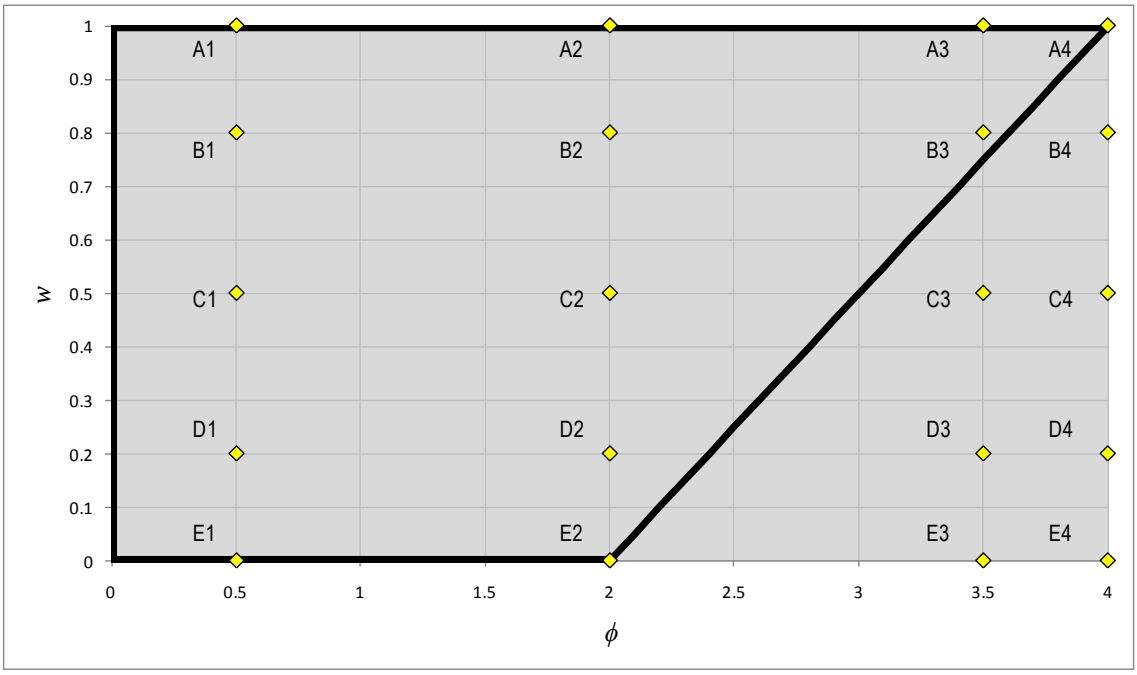


Fig. 5.21. Twenty selected pairs 'ϕ–w' to be considered to illustrate the speed and form of convergence/divergence of the particle's trajectory associated to the different regions of the 'ϕ–w' plane. Note that only 8 of these pairs are within convergence region. The other 12 pairs lead to either (pseudo) cyclic or divergent trajectories.

The trajectories of the particle resulting from the twenty selected pairs 'ϕ–w' shown in Table 5.3 and Fig. 5.21 are offered in Fig. 5.22 and Fig. 5.23, where both attractors are fixed at $x = 0$. Notice that Fig. 5.22 shows the trajectories corresponding to the first two columns of points in Fig. 5.21 whereas Fig. 5.23 shows the trajectories corresponding to the other two columns. Hence, if these figures are assembled next to each other, a grid of trajectories is obtained corresponding to their homologous pairs 'ϕ–w' in Fig. 5.21.

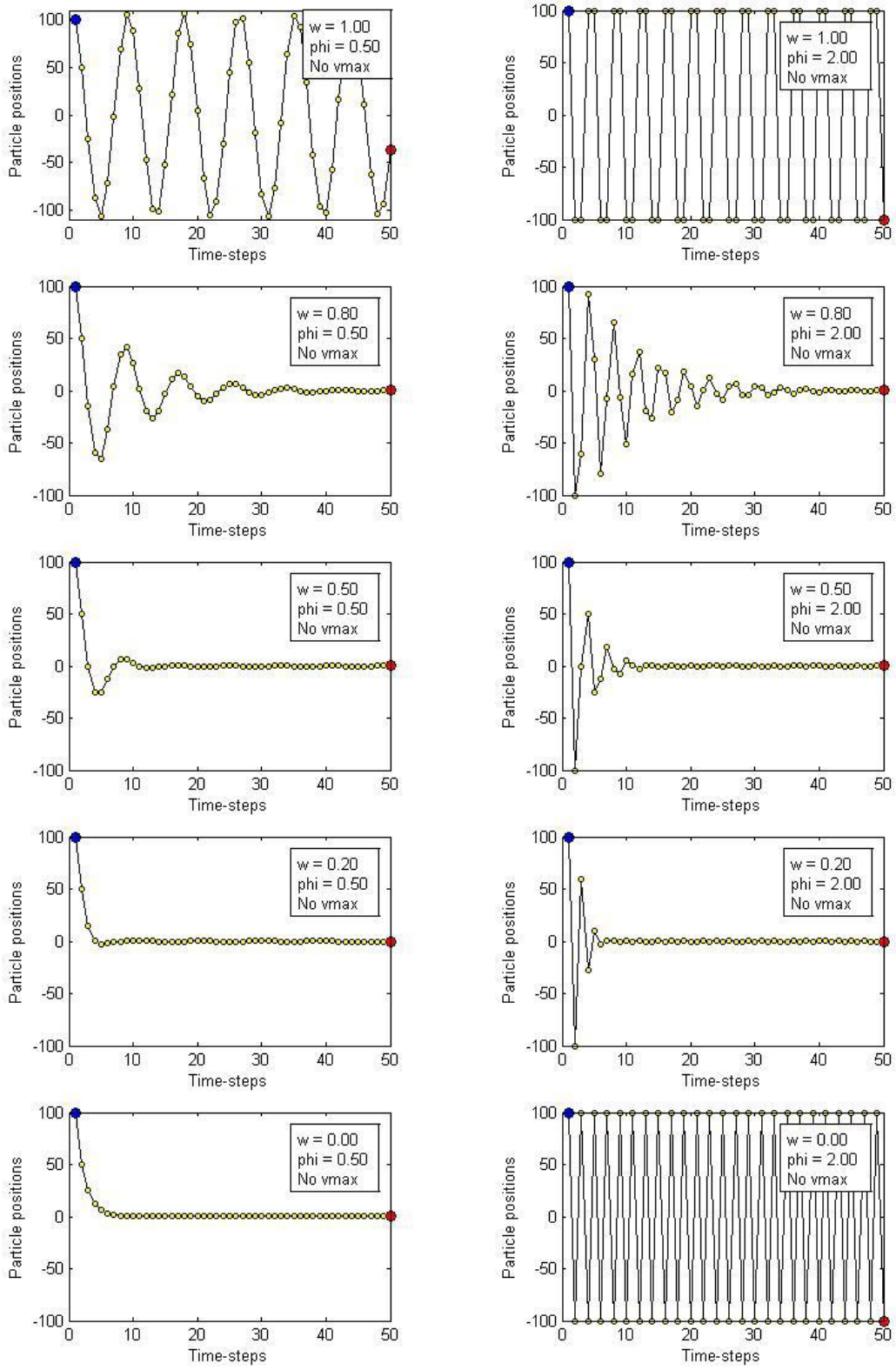


Fig. 5.22. Trajectories of the deterministic particle for the pairs A1 to E2 (first two columns) in Fig. 5.21.

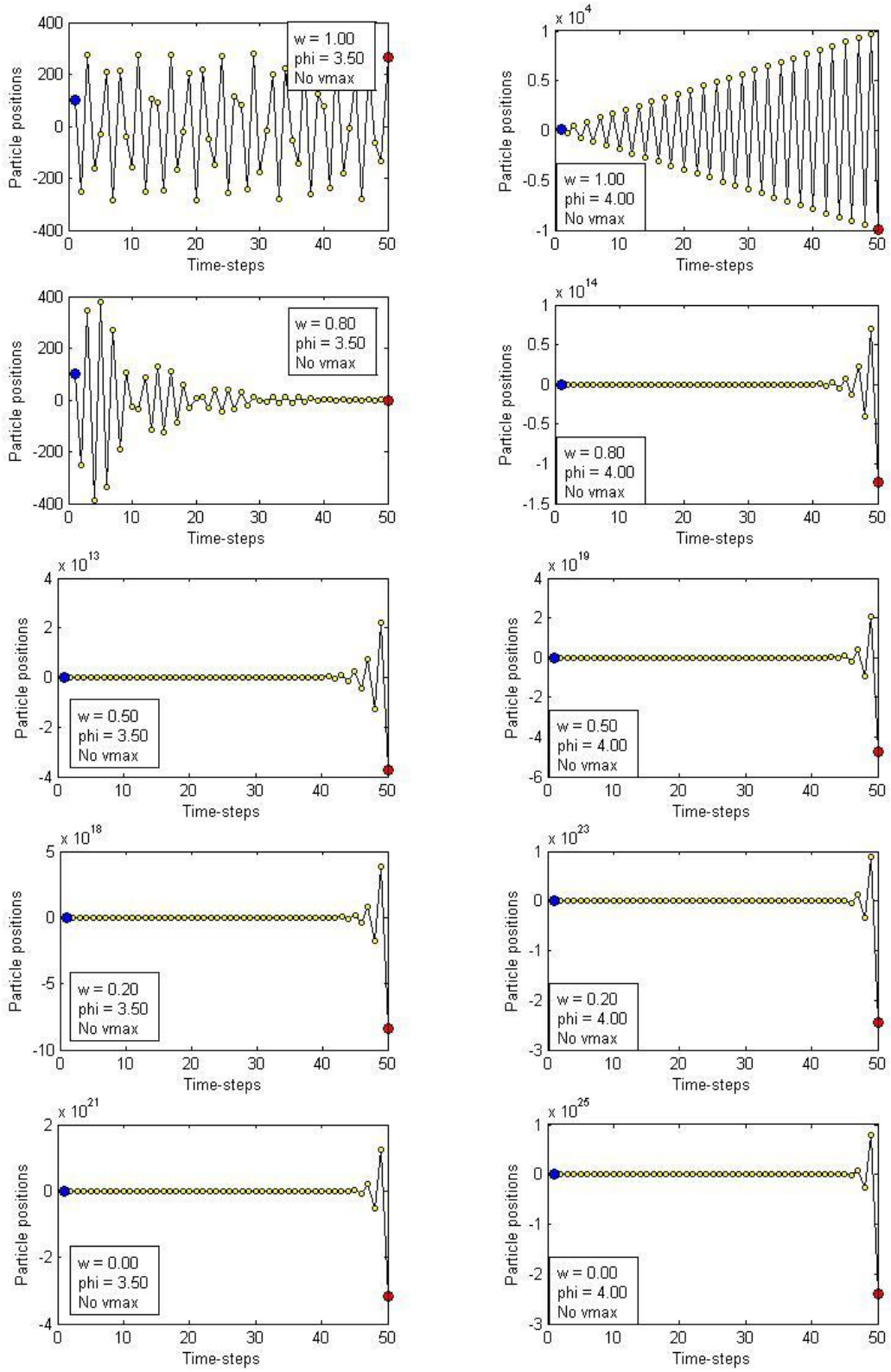


Fig. 5.23. Trajectories of the deterministic particle for the pairs A3 to E4 (last two columns) in Fig. 5.21.

Fig. 5.22 and Fig. 5.23 show that the points within the convergence region (B1, B2, B3, C1, C2, D1, D2, and E1) effectively lead to convergence towards the attractor. Conversely, the points within the divergence region (B4, C3, C4, D3, D4, E3, E4) lead to the particle diverging from the attractor, where the lower the inertia weight (w) for a given acceleration coefficient (ϕ) −and the greater the ϕ for a given w− the greater the explosion. The points on the boundary –namely A1, A2, A3, A4, and E2− do not lead to convergence. According to the magnitudes of the roots of the characteristic polynomial associated to the recurrence relation of the particle's position, they may lead to cyclic, pseudo-cyclic, or divergent trajectories. Loosely speaking, points A1, A2 and A3 lead to (pseudo) cyclic trajectories because the roots are complex conjugates with a module equal to one; point A4 leads to a linearly divergent trajectory because both roots are real-valued and equal to '−1'; and point E2 leads to a cyclic trajectory because one root equals '0' and the other equals '−1' which results in the particle moving in opposite directions in consecutive time-steps always keeping the same distance from the attractor.

With regards to the different speeds of convergence, it can be observed that the closer to the boundaries of the convergence region the slower the convergence. Regarding the form of convergence, the greater the inertia weight (w) in relation to the acceleration coefficient (ϕ) the lower the frequency of the oscillations in the trajectory. In other words, low w with high ϕ leads to high frequencies (i.e. the attractor being overflown a higher number of times) whereas high w with low ϕ results in low frequencies. Setting them both to small values is not advisable, as exploration is affected, while setting them both to great values (approaching point A4 in Fig. 5.21) leads to slow convergence.

5.6. On exploitation and exploration

The terms 'exploitation' and 'exploration' are frequently used in the search algorithms' literature. They are typically loosely defined, and mostly used in a semantic rather than in a technical fashion. Although Naudts and Schippers (1999) attempted to provide an unambiguous definition of these terms, they only apply to algorithms that are not population-based and do not possess any sort of memory. Nonetheless, it seems fair to discuss some of their definitions. They consider two types of exploitation and two types of exploration, providing some hints as to how they may be quantified.

Representational exploitation: The representation of the current solution is used to generate neighbouring candidate successors (resembling mutation in EAs). As to its quantification, (...) *there is more representational exploitation when the neighbourhood plays a more restrictive role.* (Naudts & Schippers, 1999)

Neighbourhood exploration: Potential successors are selected from the neighbourhood and evaluated (resembling mutation in EAs). Its quantification may be given by the number of distinct candidate solutions that are evaluated.

Objective exploitation: The actual successor is selected based on the information obtained from the neighbourhood exploration (resembling fitness-based survivors' selection in EAs). The quantification of the objective exploitation may be given by the selection pressure of the algorithm.

Generational exploration: The successor is accepted.

According to these definitions, there is a kind of representational exploitation in PSO: the next position of a particle is generated from the current one. However, the actual solution is not the current position of a particle but the global best. In addition, the new position may be located far from the current one. The neighbourhood exploration is a bit more complex in PSO, since the neighbourhoods are typically topological and involving the 'memory swarm' as well. The objective exploitation is indirect, as the new position replaces the current one regardless of their conflicts (i.e. fitness). However, the new positions of the 'memory swarm' are indeed conflict-based. The quantification of this exploitation would be given by the (randomly weighed) coefficients in the velocity update equations: a set of coefficients that drives the particle closer to the attractor would imply a higher objective exploitation.

Thus, the definitions in (Naudts & Schippers, 1999) may be adapted to PSO to some extent. However, they do not stand for what those terms mean within this thesis. In population-based methods, the terms 'exploration' and 'exploitation' are typically used to qualitatively describe the type of search: 'exploration' would mean that the individuals are looking for potentially good areas in the search-space, whereas 'exploitation' would mean that the individuals are performing a fine-grain search within any one of these identified promising areas. Hence, however loosely defined, 'exploitation' refers to the exploration of a small neighbourhood surrounding a given (usually good) solu-

tion. The question is, of course, how to define this small neighbourhood[1]. It is important to note that there is no attempt here to quantify exploration or exploitation; in fact, there is not even a crisp limit defined between them. That is to say, a low exploitation implies a high exploration and vice versa. Thus exploration is associated to the concept of diversity: the higher the diversity the more explorative the behaviour. The diversity is measured according to some proposed measures of the degree of clustering late in Chapter 6. Therefore, the lower the value of those measures the lower the exploration (i.e. the higher the exploitation) and the higher the value of those measures the higher the exploration (i.e. the lower the exploitation). This implies that an algorithm which returns a higher number of distinct solutions for a given search-length is more explorative. Hence a set of coefficients in PSO which lead to faster convergence is more exploitative, as the particles implode to a solution sooner. As a consequence, exploitative behaviour will be, in general, unable to escape local attractors.

A definition of exploitation/exploration somewhat similar to the meaning these terms have in this thesis is given by Beyer (1998), who views exploitation as the ability to step into the local gradient direction and exploration as the ability to leave the gradient path.

While increasing degrees of clustering of the particles mean increasing exploitation in this thesis, Clerc (2008a) interestingly proposes measuring the exploitation by defining local exploitation areas (*LEAs*) around the particles' best experiences, and then computing the so-called 'exploitation rate'. The latter is the ratio between the number of particles located within a *LEA* and the total number of particles in the swarm. It is not clear to me, however, how these *LEAs* are defined: to the best of my understanding, these areas would be decreasing as the particles' best experiences cluster closer except for those whose *LEAs* involve the boundaries of the search-space in their definitions.

5.7. Velocity constraint

For $\phi > 0$, the explosion does not take place by means of consecutive small steps moving away from the attractor but by means of increasing amplitudes of the oscillations. In fact, in the case of a deterministic explosion with at least one of the roots smaller '−1',

[1] Note that 'neighbourhood' refers to a region of the search-space, and not to the 'social network' in PSO.

the particle moves in opposite directions in consecutive time-steps with increasing amplitudes. Therefore, an external constraint to control the explosion consists of restricting the maximum displacement permitted for a particle between consecutive time-steps. More precisely, given that different dimensions are treated independently in PSO, the restriction is actually on the components of the displacements. The effect of this so-called '$v_{\max}$ constraint' (also 'velocity clamping') on a 'deterministic explosion' of a deterministic particle is shown in Fig. 5.24.

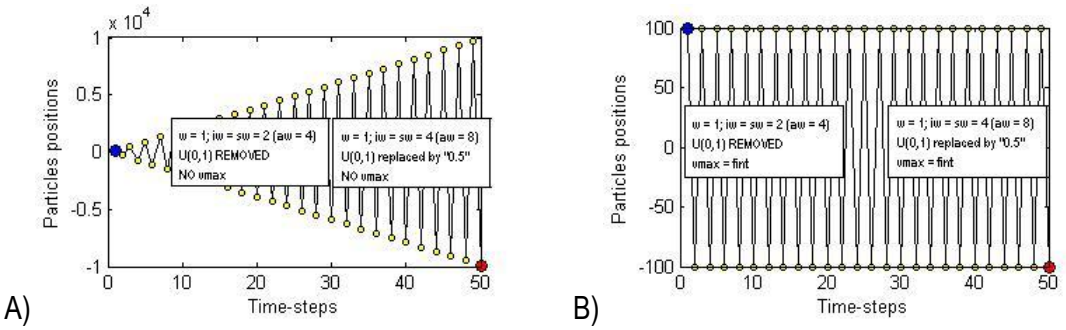


Fig. 5.24. Comparison of the trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for the deterministic linear explosion (w = 1.00; aw = 4.00; $U_{(0,1)}$ removed) with and without constraining the velocity. The trajectory becomes cyclic if the velocity is constrained to the feasible interval *fint*.

It is of more practical relevance to analyze the effect of the '$v_{\max}$ constraint' on the 'stochastic explosions'. From Fig. 5.20 and Fig. 5.9, it is advisable that $\phi_{\max} \leq 4$ and $\phi_{\text{mean}} \leq 2$. In the original PSO algorithm, $w = 1$ and $aw = 4$, so that $0 \leq \phi \leq 4$ and $\phi_{\text{mean}} = 2$. For these settings, a stochastic explosion occurs, as shown in Fig. 5.10 and Fig. 5.25 I). In order to observe the influence of the $v_{\max}$ constraint on the trajectory of the particle, the latter is plotted in Fig. 5.25 and Fig. 5.26 for different settings of $v_{\max}$. Figures sharing the same row share the same acceleration weight. Furthermore, in order to assess the deviation caused by the random weights, every experiment is carried out for the particle with the random weights on the right, and with their expected values on the left. The particle is initialized at $x = 100$ with $v^{(0)} = 0$ and $p = 0$ in every case, and the feasible interval is set to $fint = 200$. Typically, the size of the constraint is linked to the size of the feasible interval.

Clearly, the restriction of the step-sizes is successful in preventing the explosion. However, it does nothing with regards to convergence, and the restriction simply narrows

down the region of the search-space around the attractor where the search is focused. The trajectory is still (pseudo) cyclic, with the amplitudes of the cycles limited by v_{max}. In fact, the peak-to-peak amplitude is approximately equal to $2 \cdot v_{max}$.

If the settings are such that convergence is ensured, the v_{max} constraint is optional, and it just helps prevent function evaluations (FEs) too far from the region of interest. If the roots of the characteristic polynomial are not both smaller than '1', the restriction to the step-sizes or an alternative method to control the deterministic explosion is compulsory.

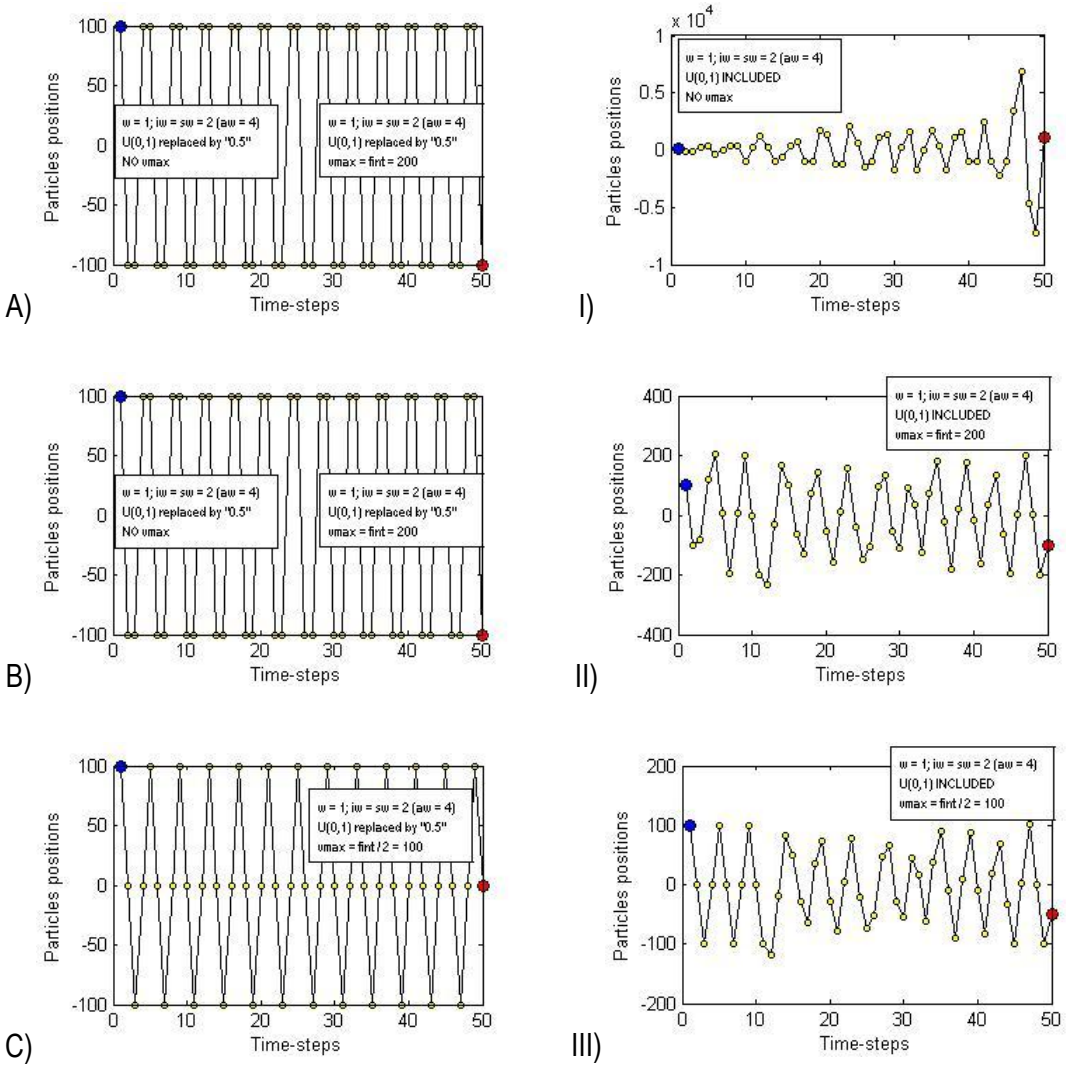


Fig. 5.25. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 1.00 and aw = 4.00. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value (i.e. '0.50') in the first. Different rows correspond to different values of the v_{max} constraint, the first one being unconstrained.

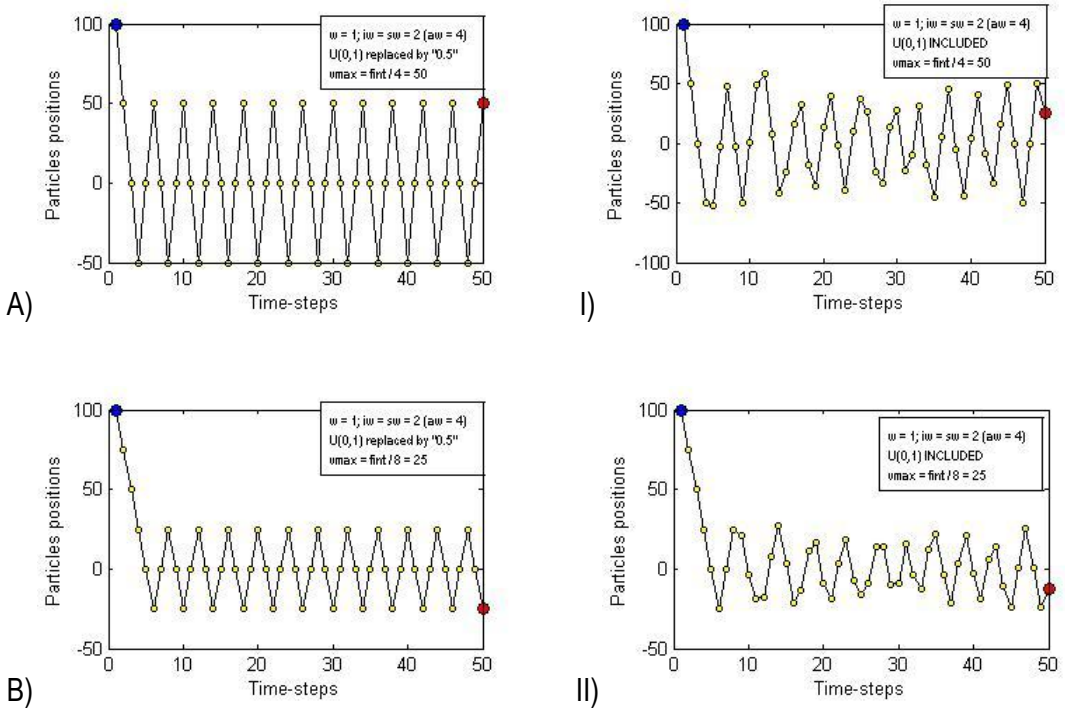


Fig. 5.26. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 1.00 and aw = 4.00. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value (i.e. '0.50') in the first. Different rows correspond to different values of the v_{max} constraint.

Notice that the majority of realistic problems, even when referred to as unconstrained, involve at least interval constraints (also 'boundary constraints' or 'side constraints'). That is to say, there is a feasible interval in each dimension of the search-space. In these cases, handling the boundary constraints is of course another means of controlling the explosion of the system without ensuring convergence or high degrees of exploitation. Beware that interval constraints can be handled by specific techniques (e.g. the 'cut-off at the boundary' or the 'bisection' methods discussed in chapter 4), or by any general constraint-handling technique (e.g. the 'penalization method'). Helwig and Wanka (2008) showed that, for some classical coefficients' settings, neighbourhood topologies, particles' and velocities' initializations, particles whose initial position (also initial best individual experience) do not coincide with their neighbourhood best experiences would leave the search-space at the first iteration with overwhelming probability. The probability increases with the dimensionality of the problem. Handling the boundary constraints helps avoid this initial local explosion. However, they found that different boundary constraint-handling techniques result in significant performance differences.

5.8. Inertia weight

As previously discussed, an inertia weight $w<1$ can be used to ensure convergence. However, different combinations of *w* and ϕ settings result in notably different dynamic behaviour and capabilities. Namely, convergence speed, reluctance to getting trapped in sub-optimal solutions, ability to perform fine-grain search, etc.

5.8.1. Random weight replaced

Fig. 5.20 allows choosing coefficients that would ensure convergence, but the type of resulting dynamic behaviour is not immediately obvious. The aim here is to visually analyze the characteristics of the convergent trajectory for different values of $w<1$ and $0<aw\leq 4$. The random weights are replaced by their expected value ($\phi = aw/2$) so as to study the convergent trajectories without the disruption introduced by randomness.

It is a common mistake to assume that the smaller the inertia weight (*w*) the fastest the convergence, as the search would become more like a local search. This is not necessarily true, and the setting of *w* that results in the fastest convergence depends on the setting of ϕ. The trajectories of the deterministic particle for $aw=4$ (i.e. $\phi=2$), as in the original PSO, and $1.10\geq w\geq 0$ is offered in Fig. 5.27 and Fig. 5.28. Clearly, convergence takes place for $w<1$ at different rates. Initially, convergence speed increases as *w* decreases until $w\cong 0.30$. From then on, the amplitude of the oscillations increase again until the trajectory becomes cyclic when $w=0$. Thus, the trajectory is cyclic in both extremes, although the attractor is overflown twice as many times for $w=0$ than for $w=1$. The period is 2 time-steps in the former case and 4 time-steps in the latter.

For a series of decreasing acceleration weights from $aw=4.00$ to $aw=0.80$, Fig. 5.27 to Fig. 5.48 show the trajectory of the particle for various settings of the inertia weight (*w*). Thus, the effect of different settings of *w* for a given *aw* can be observed.

For $4.00\geq aw\geq 2.00$, the value of *w* that leads to the fastest convergence decreases as *aw* decreases. Thus, it is approximately equal to '0.30' for $aw=4.00$ (see Fig. 5.28 C)); '0.20' for $aw=3.40$ (see Fig. 5.34 B)); '0.10' for $aw=2.80$ (see Fig. 5.40 G)); and '0.00' for $aw=2.00$ (see Fig. 5.44 H)). In the latter case, the particle reaches the attrac-

tor in one time-step. For $aw < 2.00$, the w that leads to the fastest convergence increases as aw decreases. It is approximately equal to '0.20' for aw = 0.80 (Fig. 5.48 G)).

5.8.1.1. Acceleration weight aw = 4.00

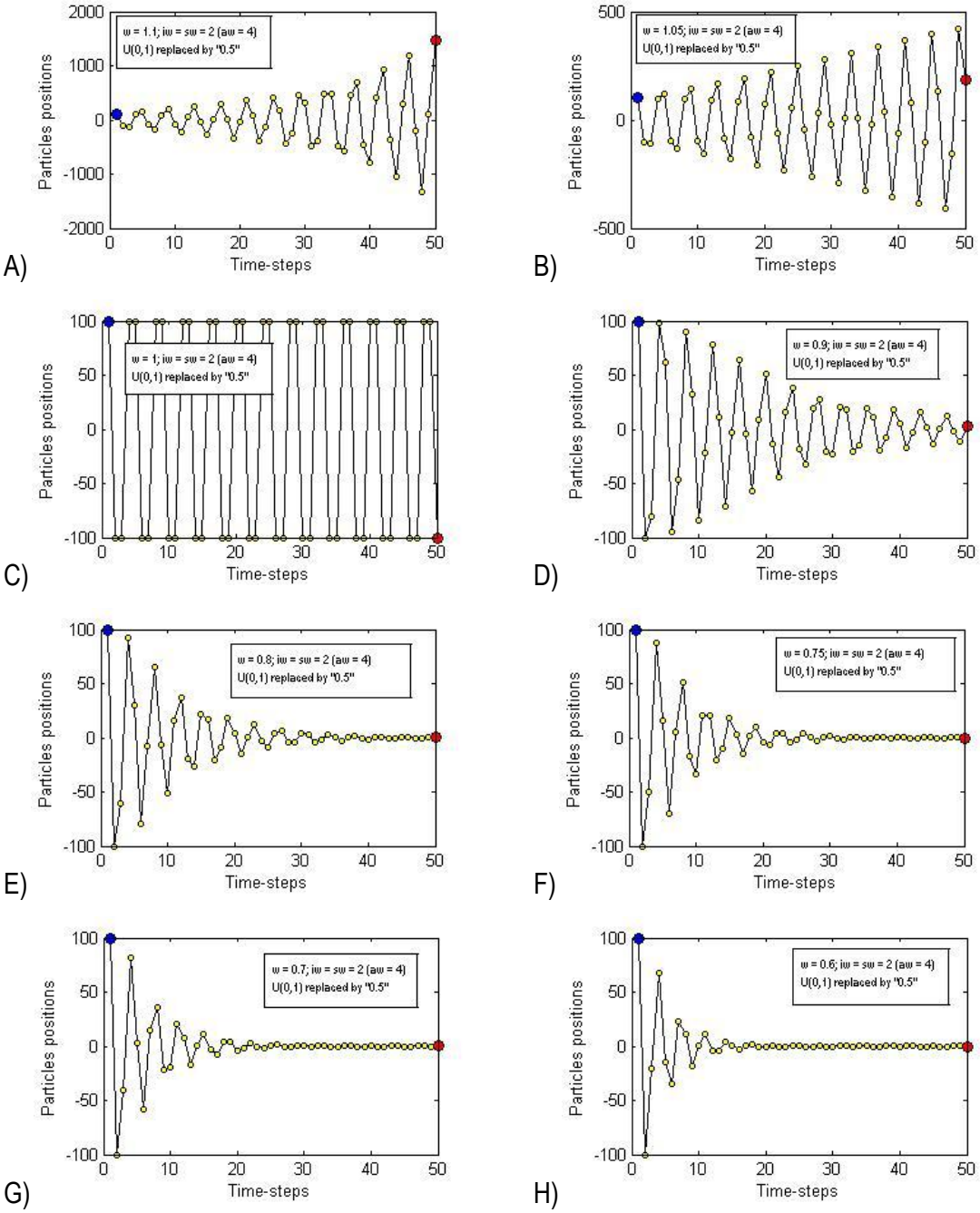


Fig. 5.27. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.10 ≥ w ≥ 0.60 and aw = 4.00. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

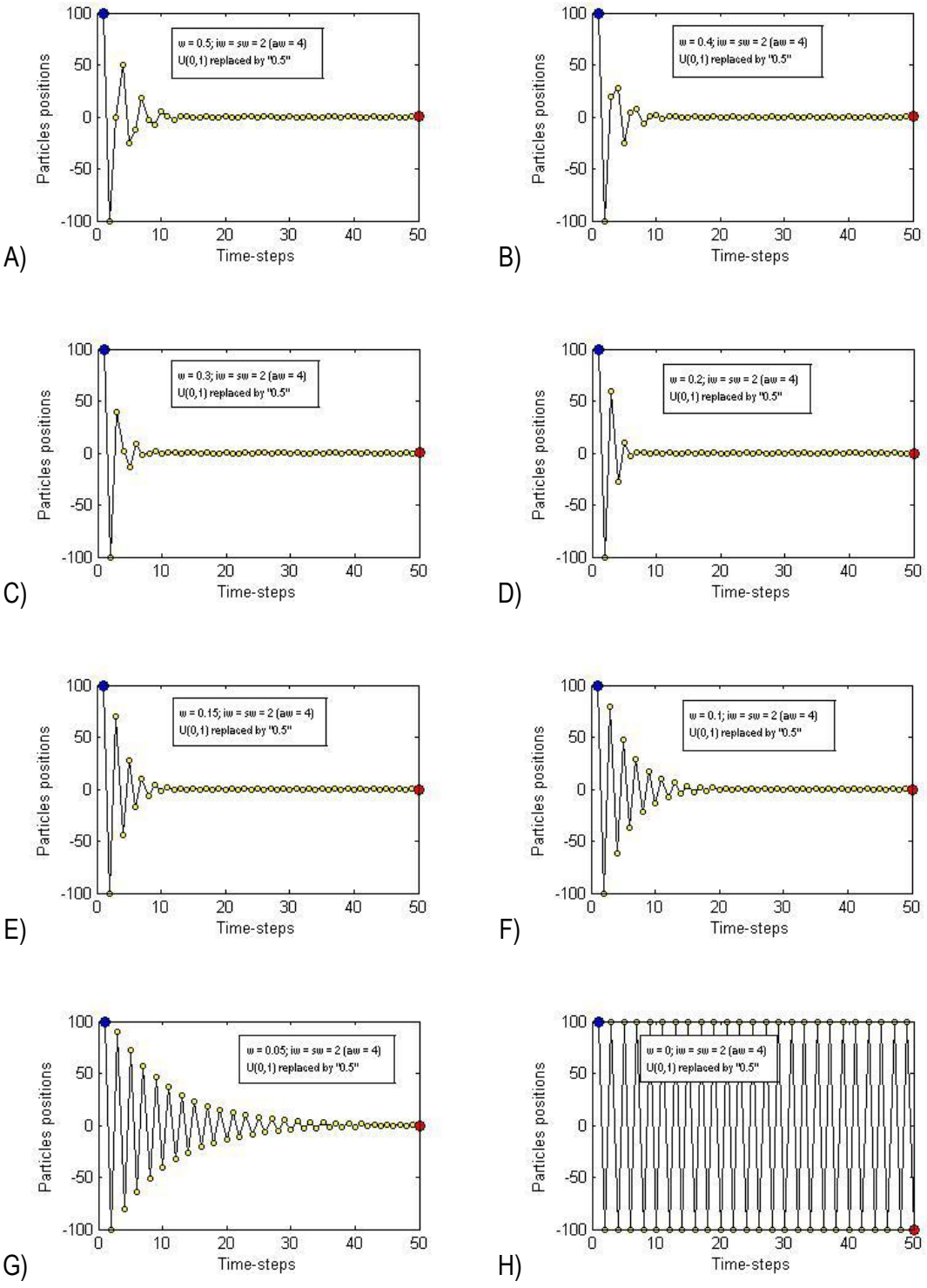


Fig. 5.28. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.50 ≥ w ≥ 0.00 and aw = 4.00. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.2. Acceleration weight *aw* = 3.80

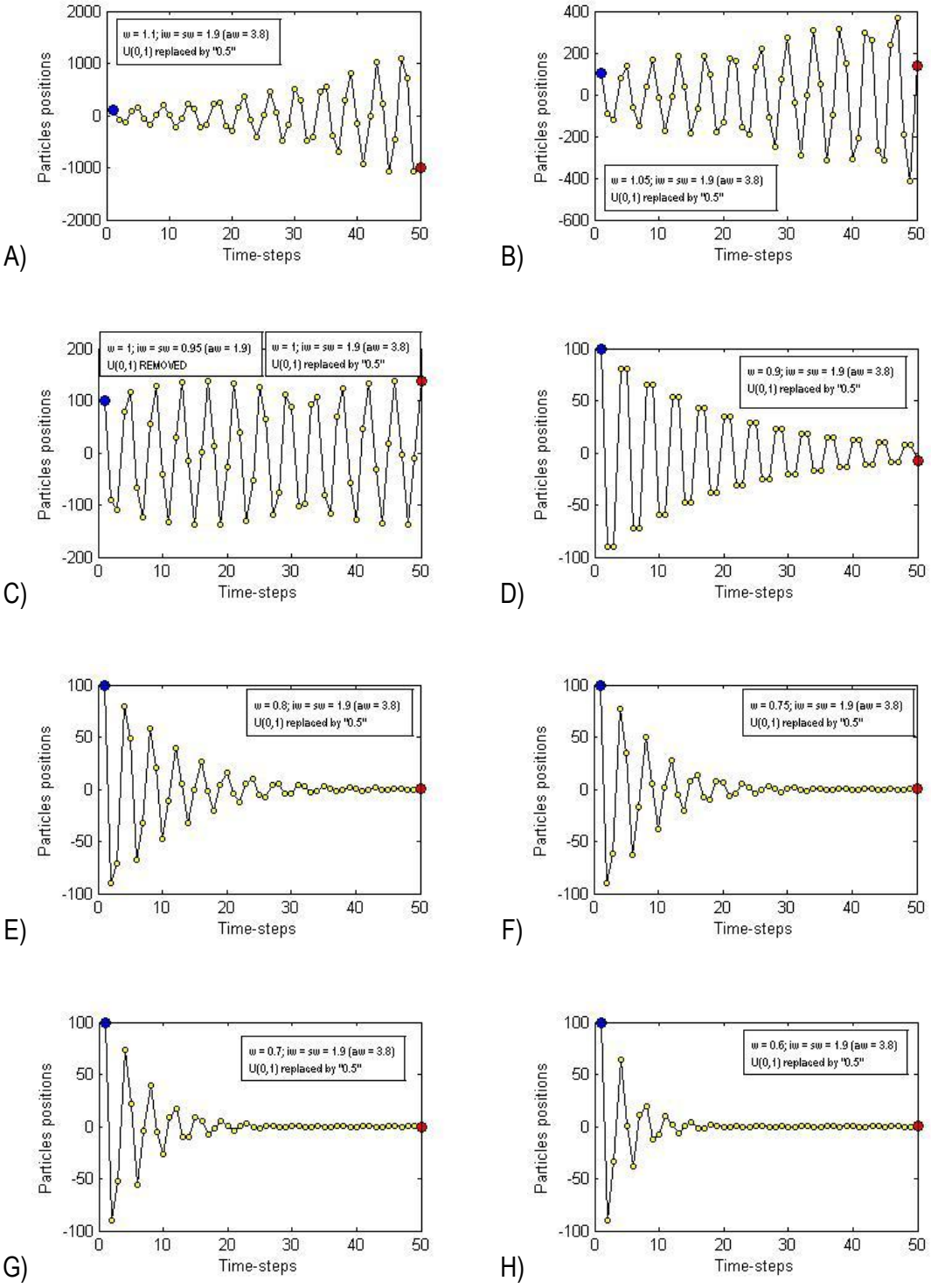


Fig. 5.29. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.10 ≥ w ≥ 0.60 and aw = 3.80. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

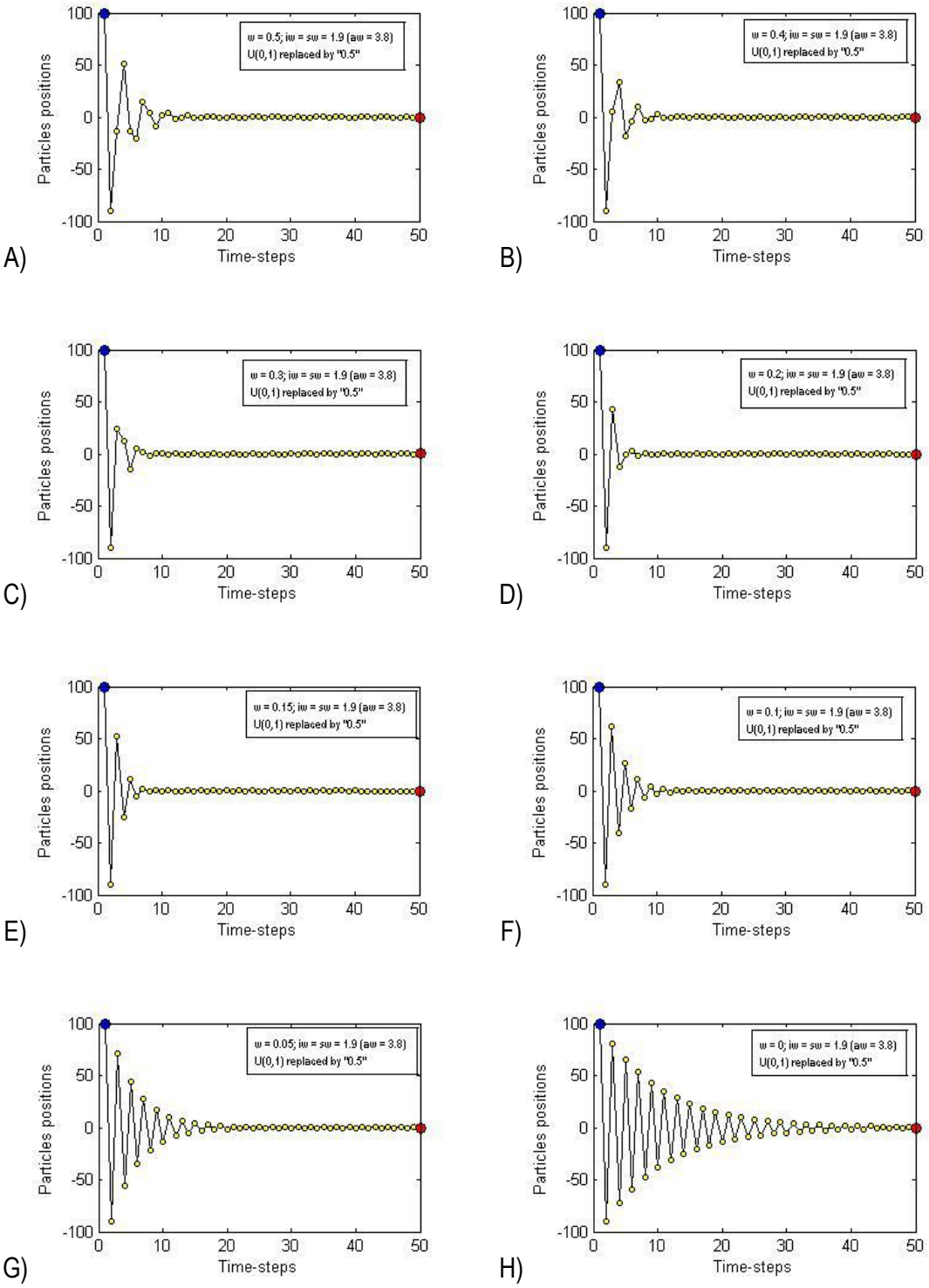


Fig. 5.30. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.50 ≥ w ≥ 0.00 and aw = 3.80. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.3. Acceleration weight *aw* = 3.60

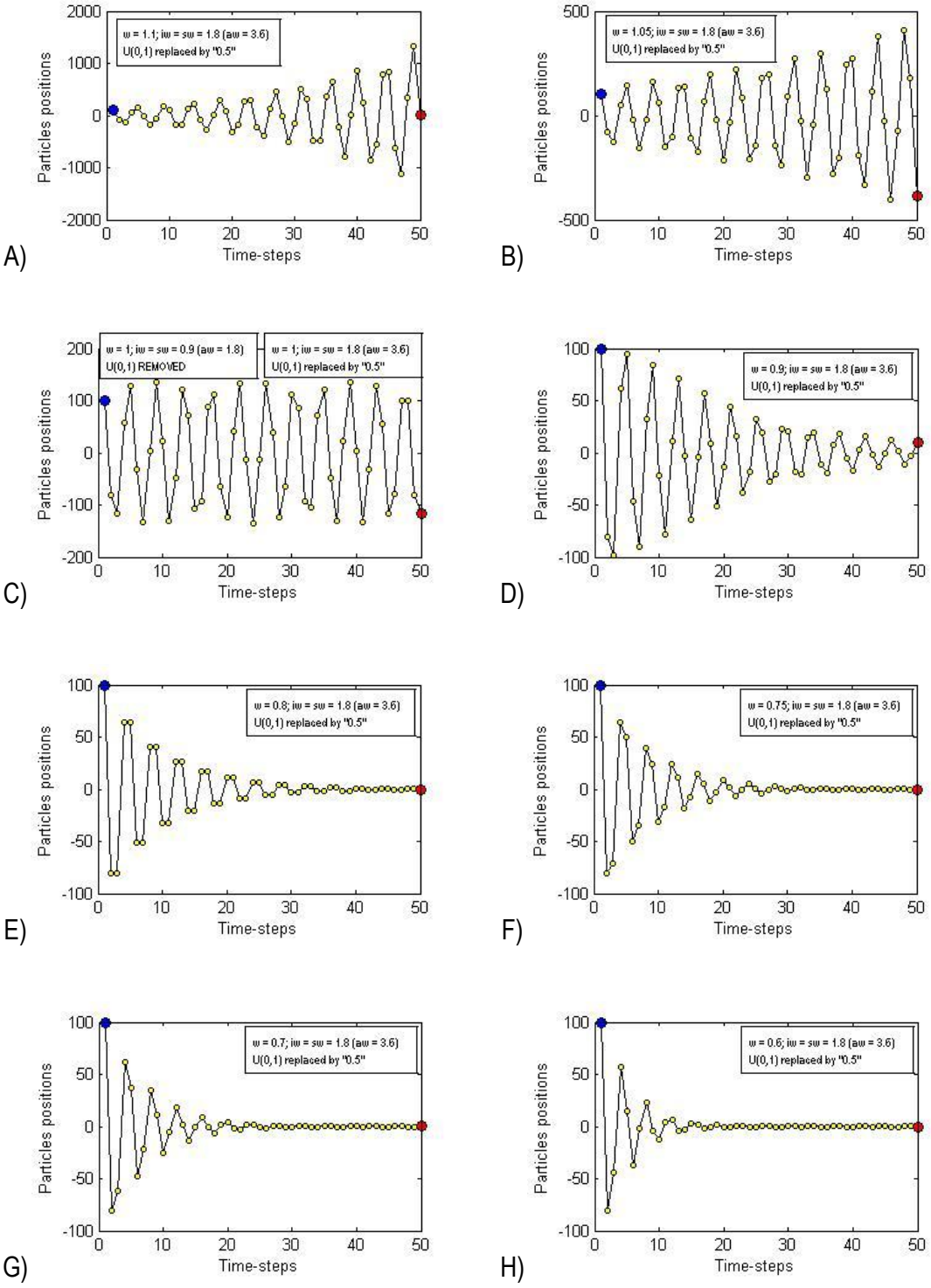


Fig. 5.31. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.10 ≥ w ≥ 0.60 and aw = 3.60. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

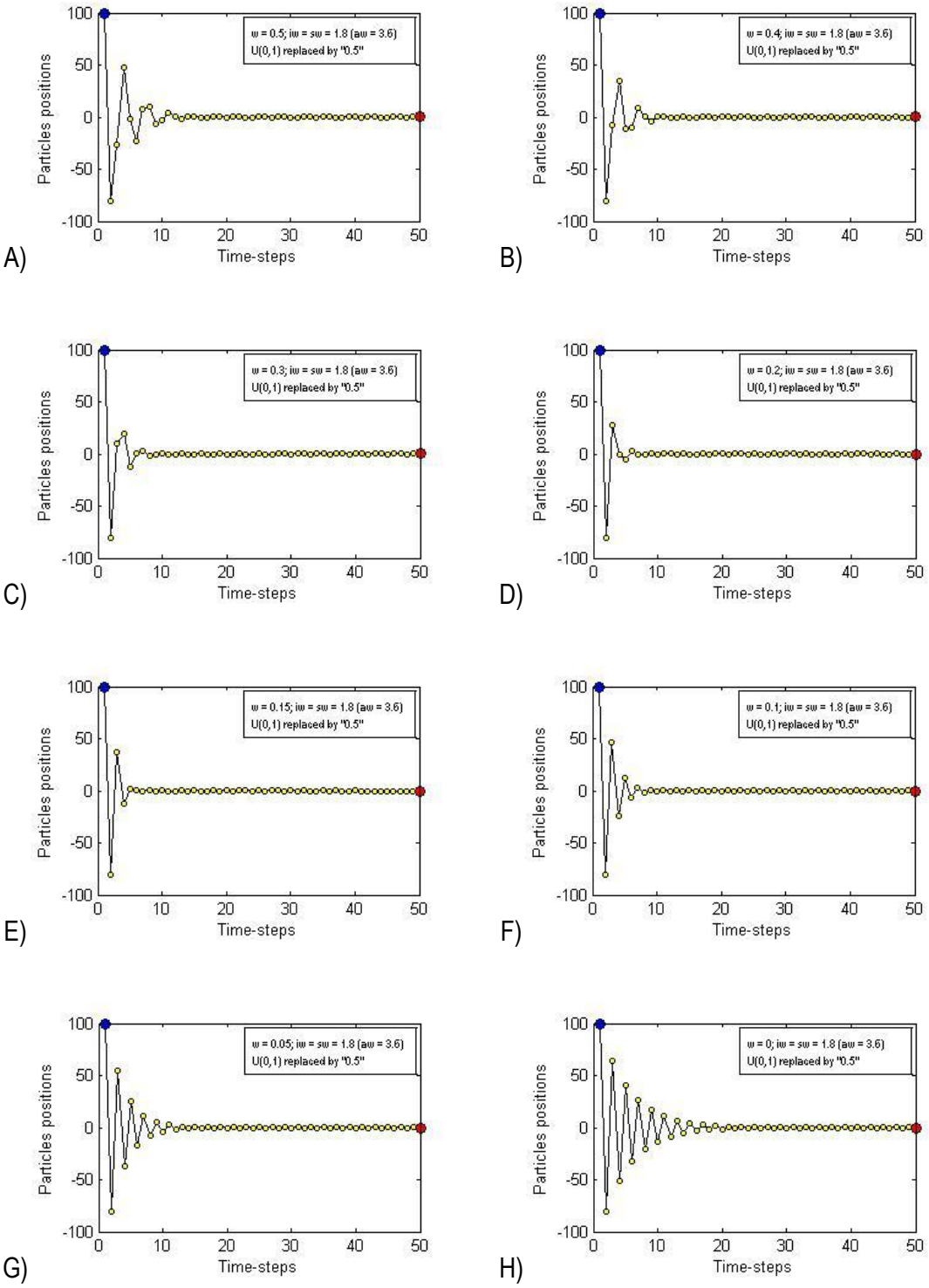


Fig. 5.32. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.50 ≥ w ≥ 0.00 and aw = 3.60. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.4. Acceleration weight *aw* = 3.40

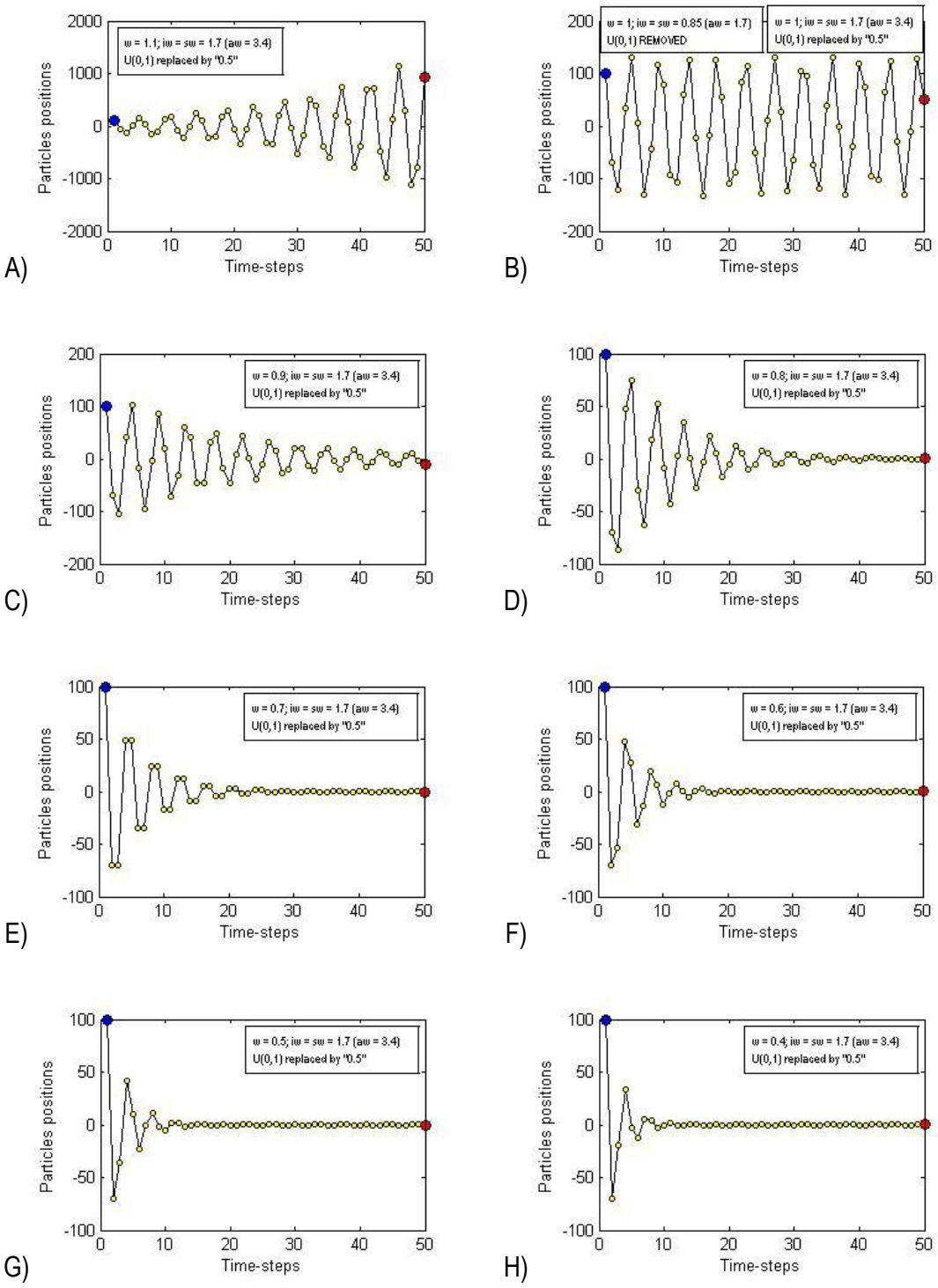


Fig. 5.33. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.10 ≥ w ≥ 0.40 and aw = 3.40. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

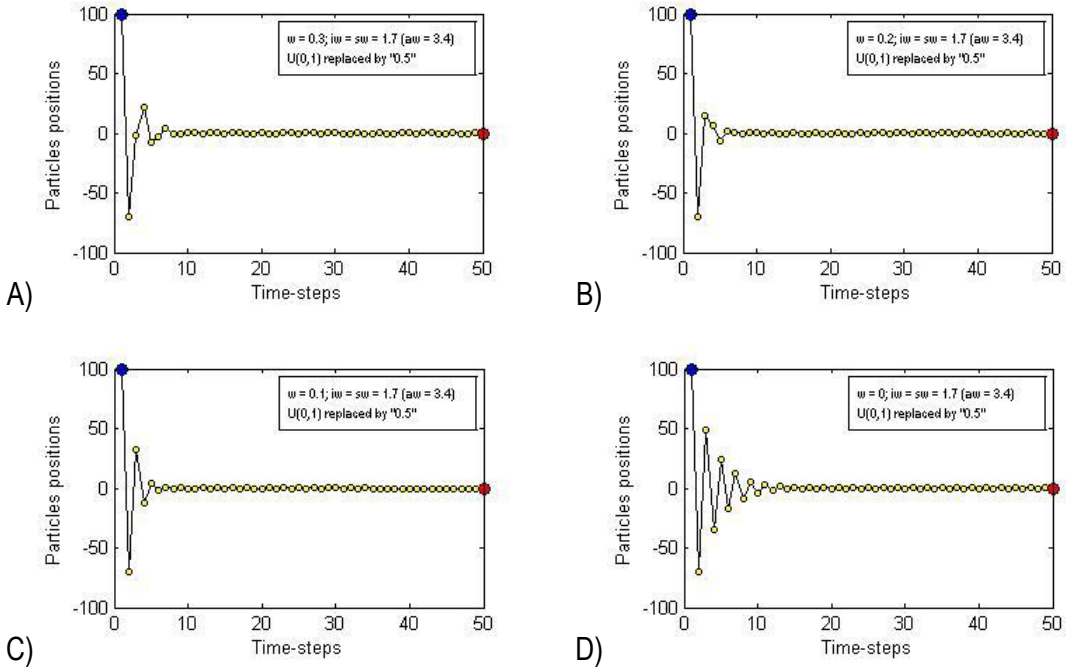


Fig. 5.34. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.30 ≥ w ≥ 0.00 and aw = 3.40. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.5. Acceleration weight *aw* = 3.20

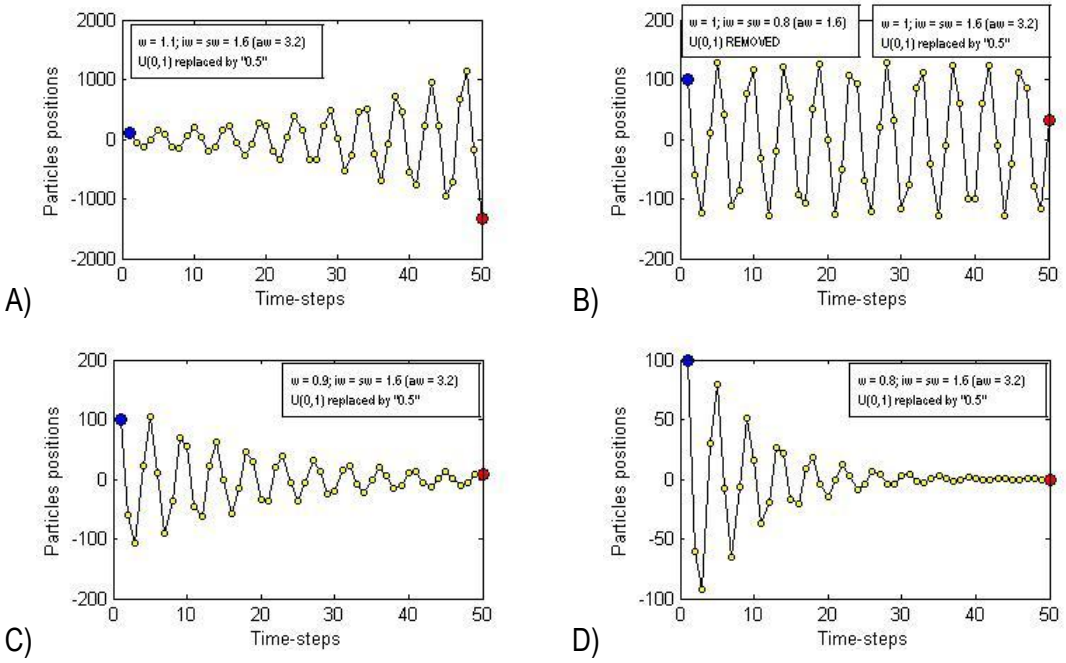


Fig. 5.35. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.10 ≥ w ≥ 0.80 and aw = 3.20. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

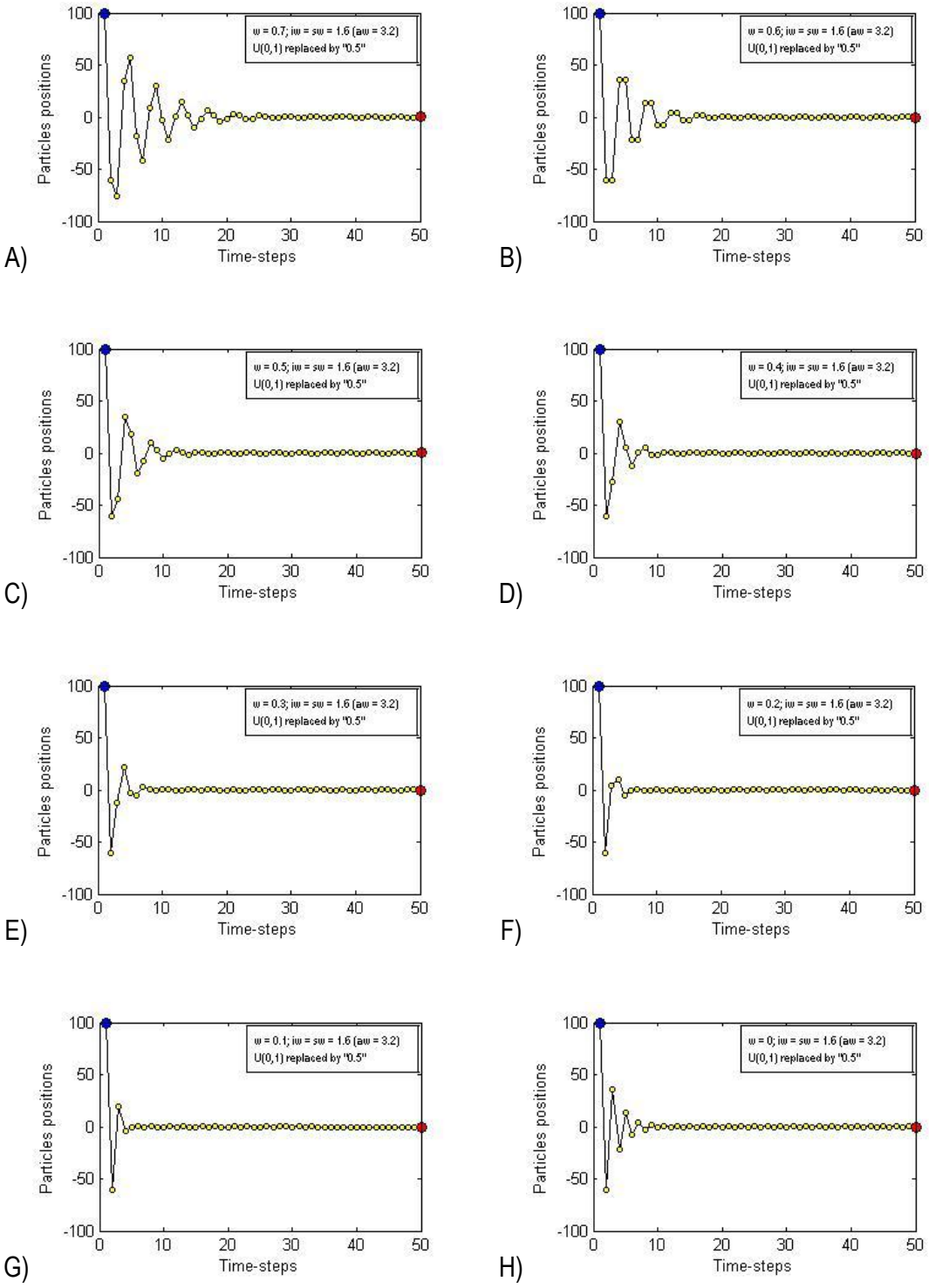


Fig. 5.36. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.70 ≥ w ≥ 0.00 and aw = 3.20. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.6. Acceleration weight *aw* = 3.00

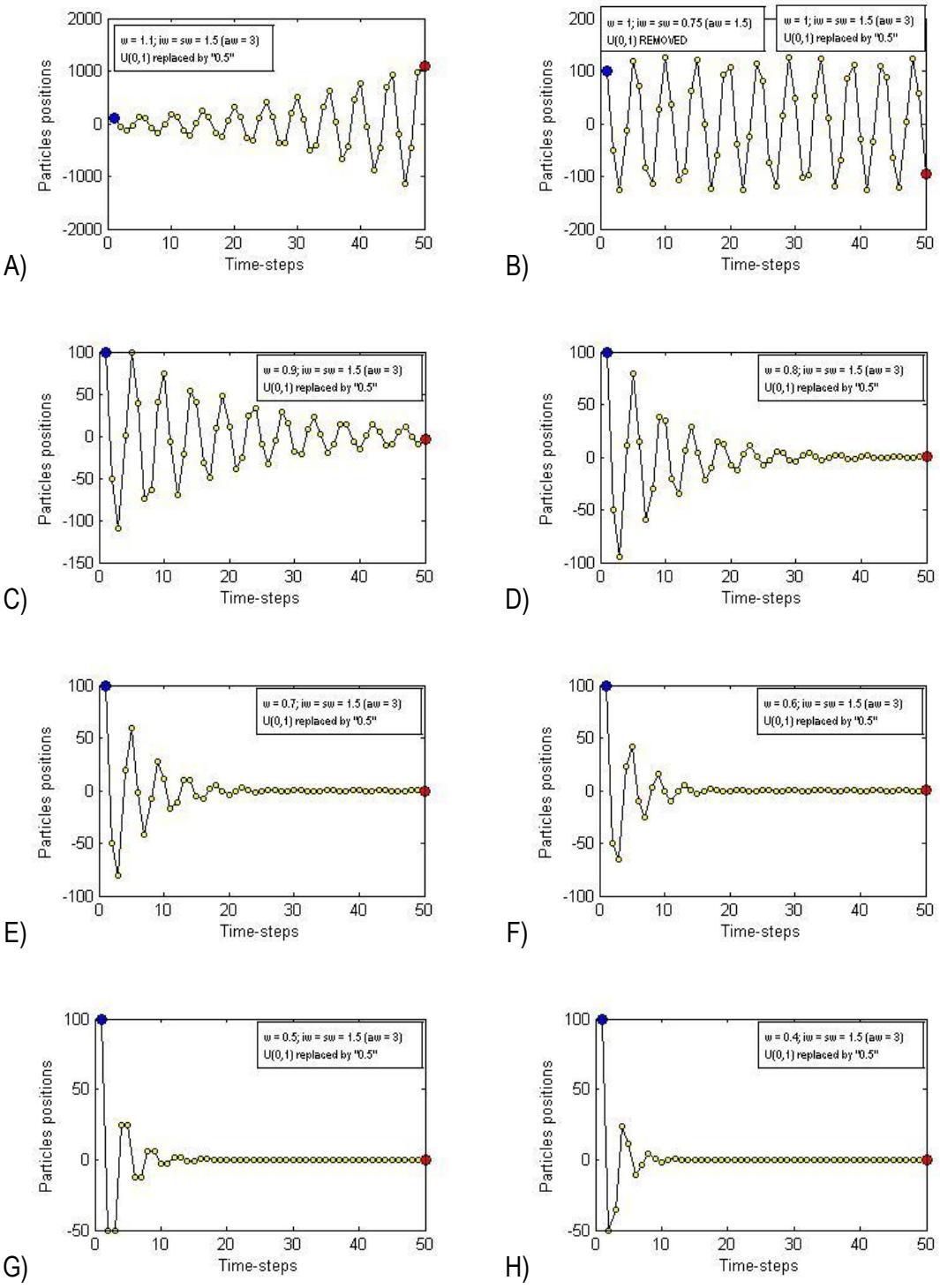


Fig. 5.37. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.10 ≥ w ≥ 0.40 and aw = 3.00. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

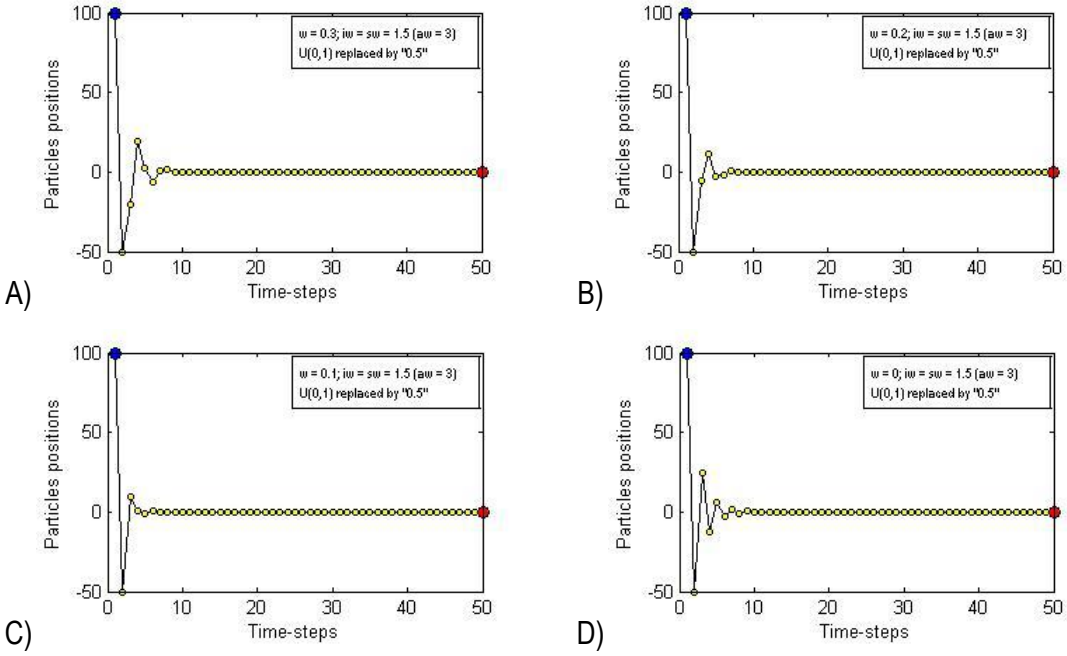


Fig. 5.38. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.30 ≥ w ≥ 0.00 and aw = 3.00. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.7. Acceleration weight *aw* = 2.80

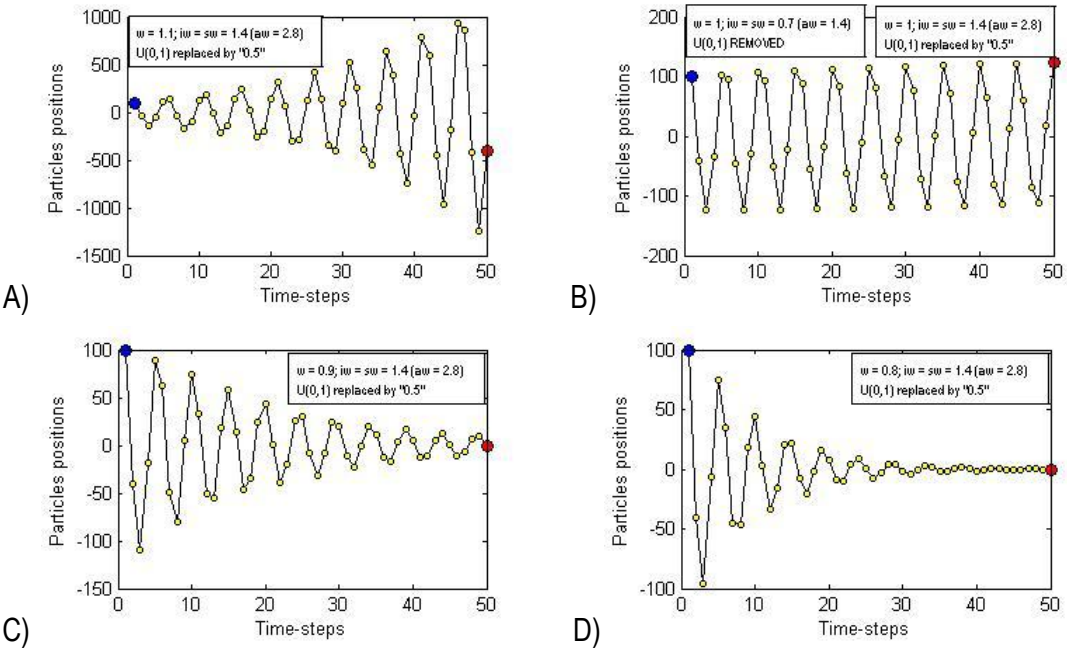


Fig. 5.39. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.10 ≥ w ≥ 0.80 and aw = 2.80. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

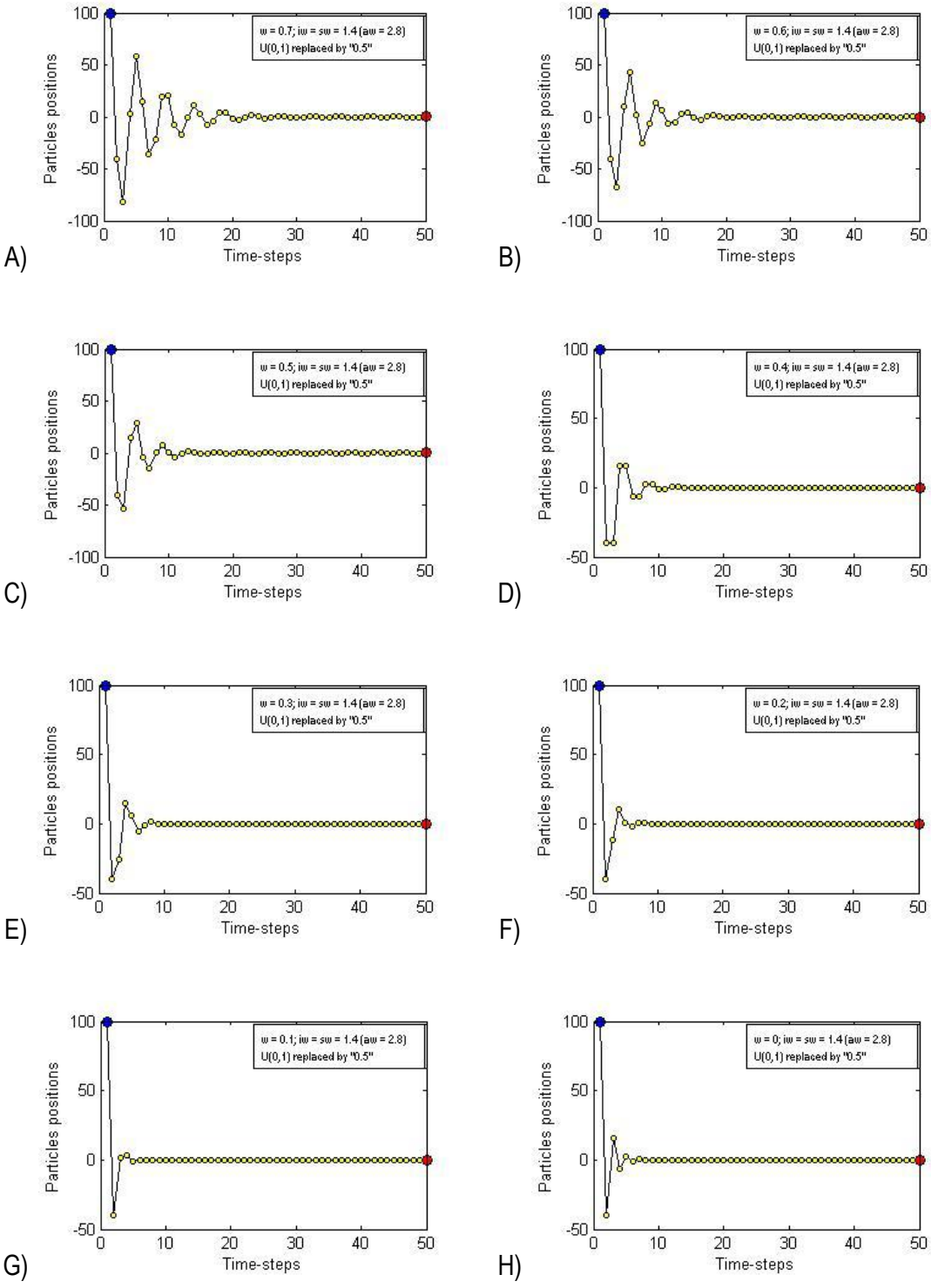


Fig. 5.40. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.70 ≥ w ≥ 0.00 and aw = 2.80. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.8. Acceleration weight *aw* = 2.60

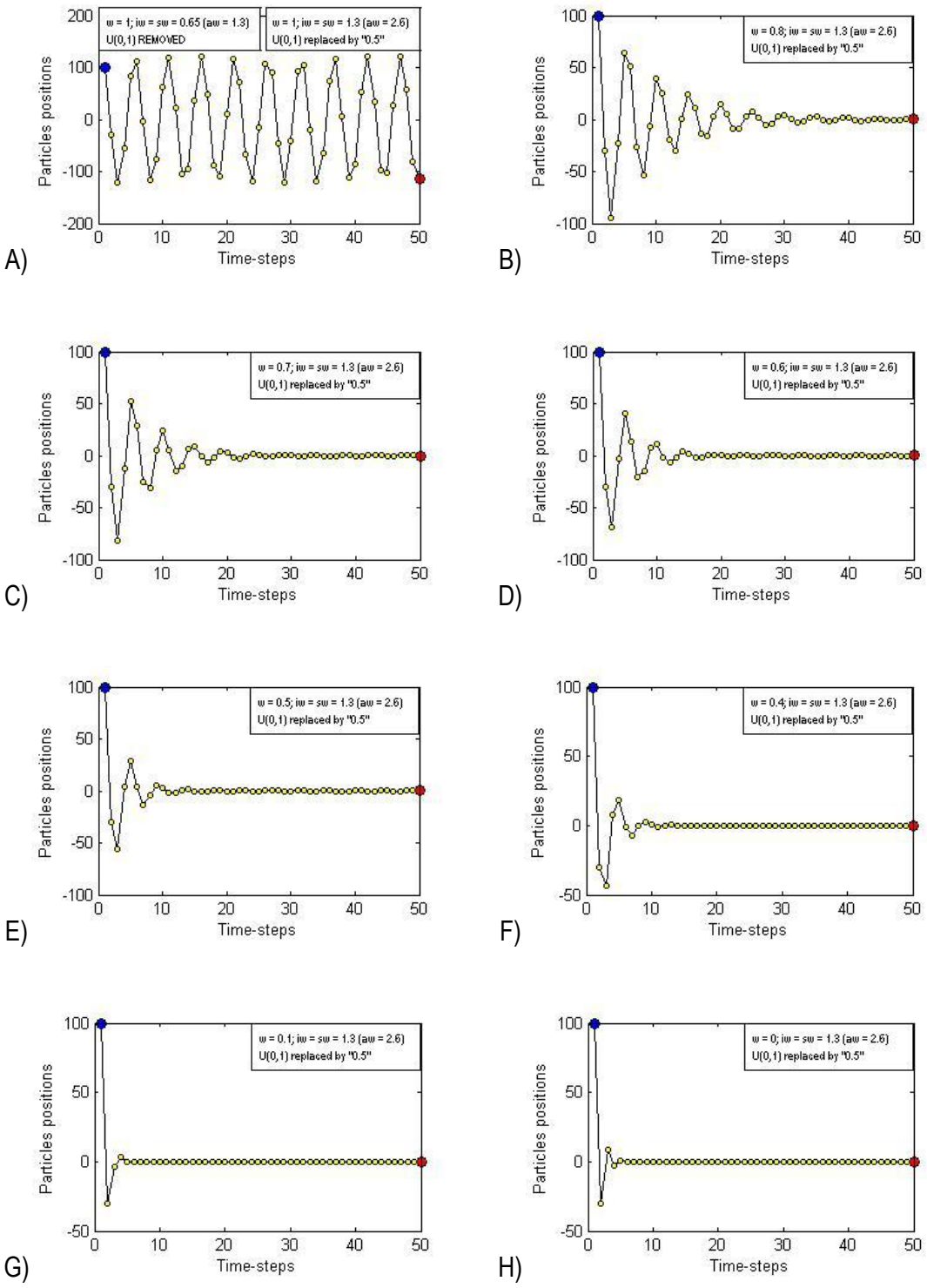


Fig. 5.41. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.00 ≥ w ≥ 0.00 and aw = 2.60. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.9. Acceleration weight *aw* = 2.40

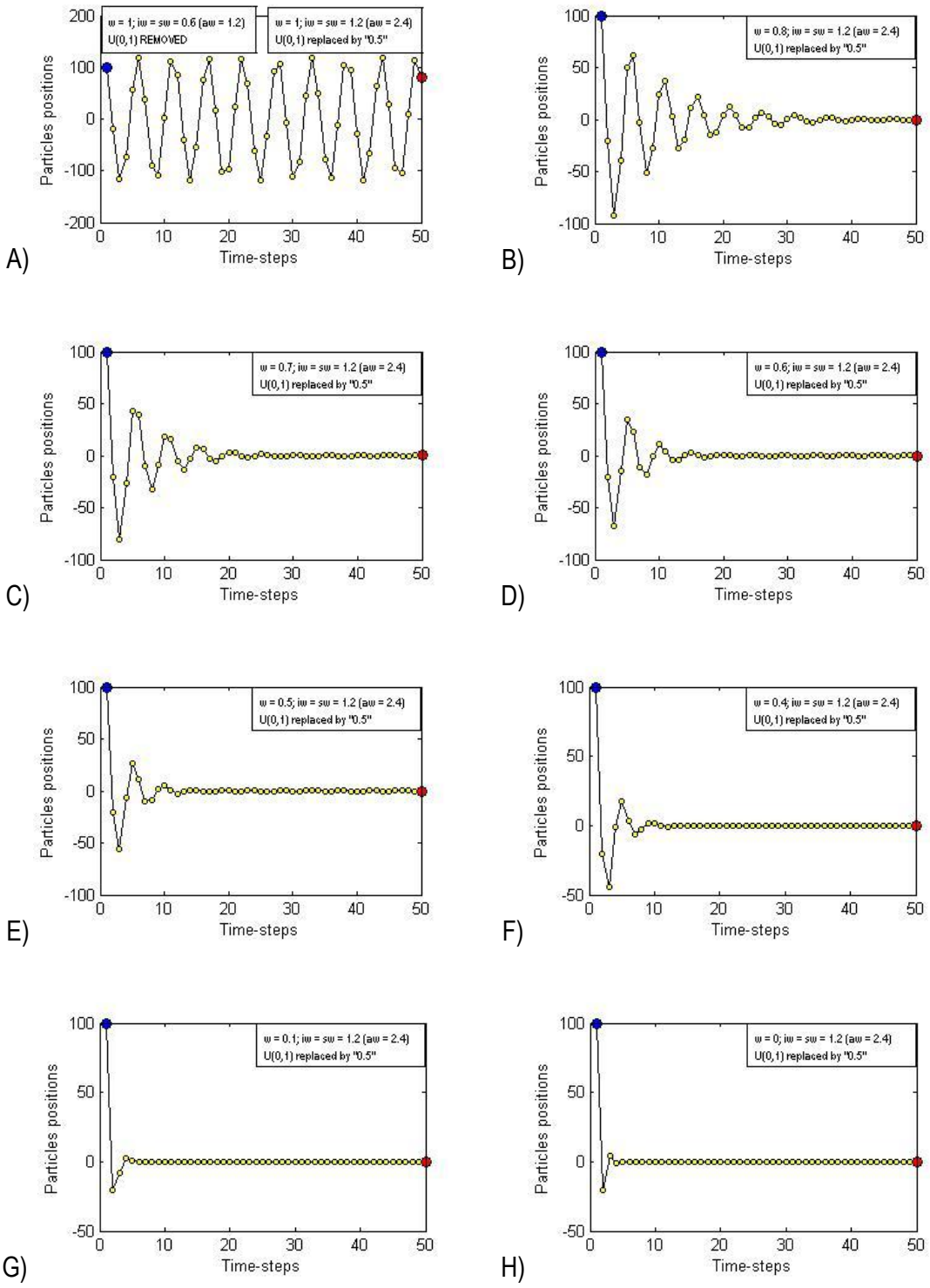


Fig. 5.42. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.00 ≥ w ≥ 0.00 and aw = 2.40. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.10. Acceleration weight *aw* = 2.20

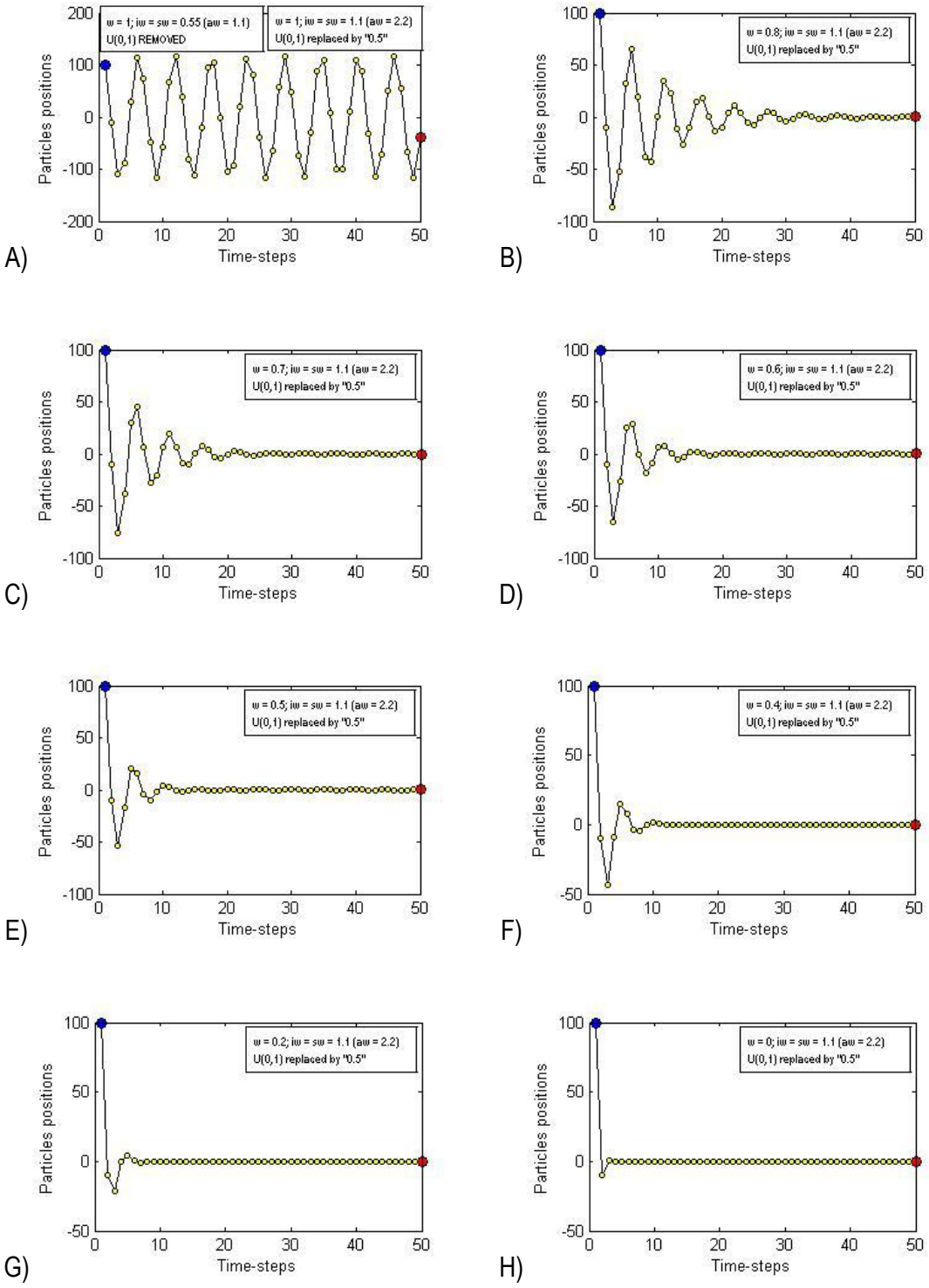


Fig. 5.43. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.00 ≥ w ≥ 0.00 and aw = 2.20. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.11. Acceleration weight *aw* = 2.00

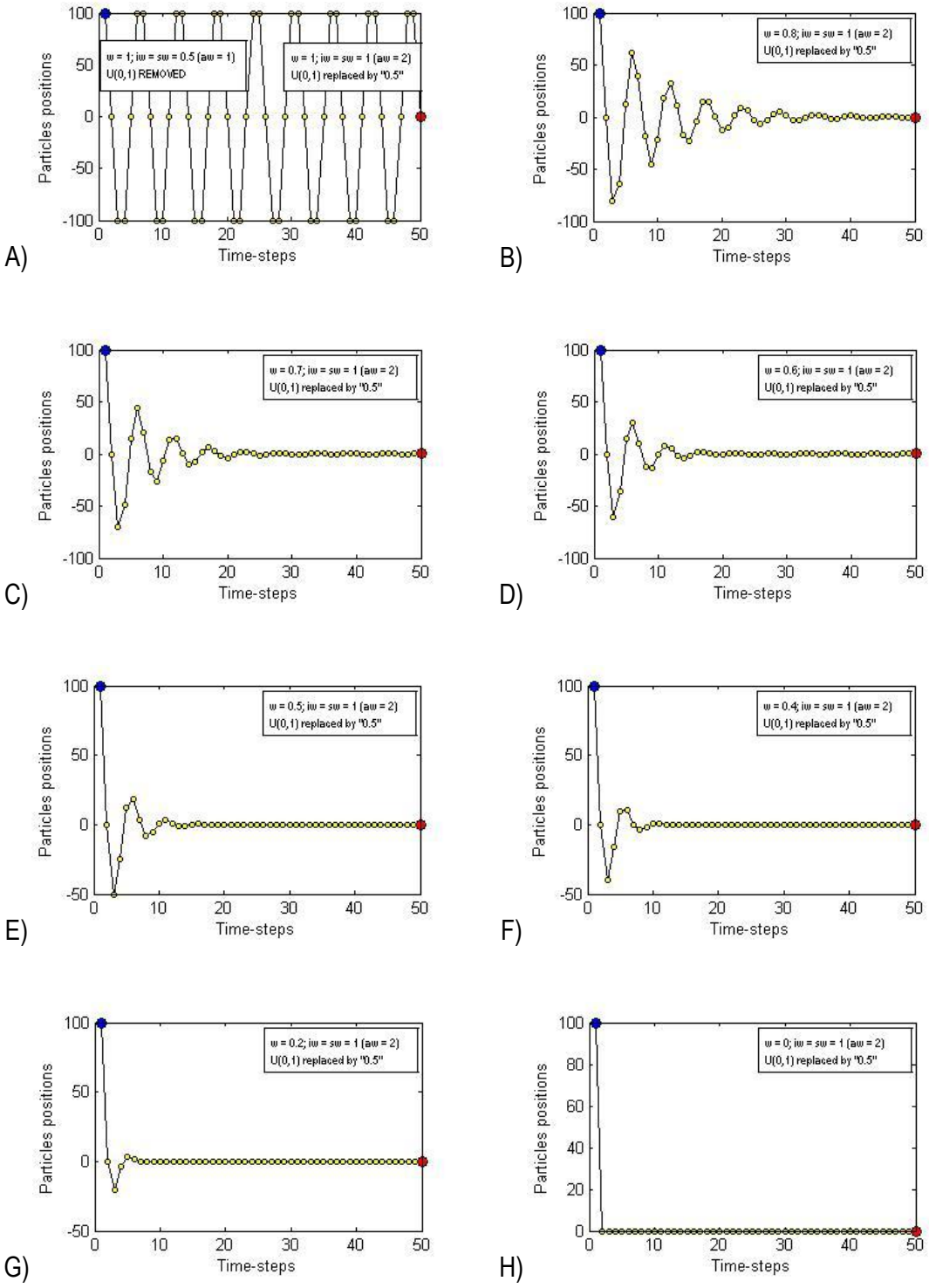


Fig. 5.44. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.00 ≥ w ≥ 0.00 and aw = 2.00. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.12. Acceleration weight *aw* = 1.80

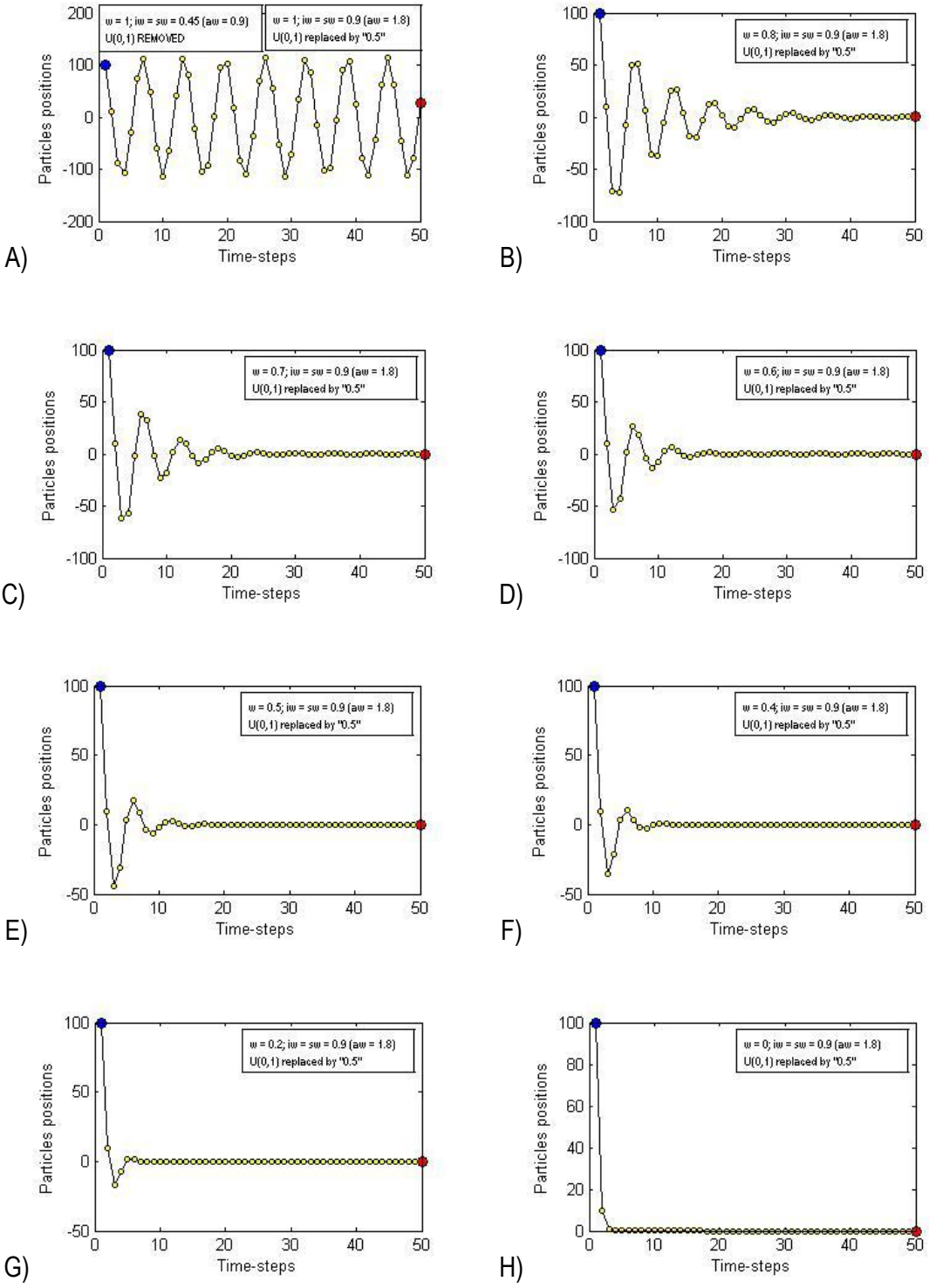


Fig. 5.45. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.00 ≥ w ≥ 0.00 and aw = 1.80. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.13. Acceleration weight *aw* = 1.60

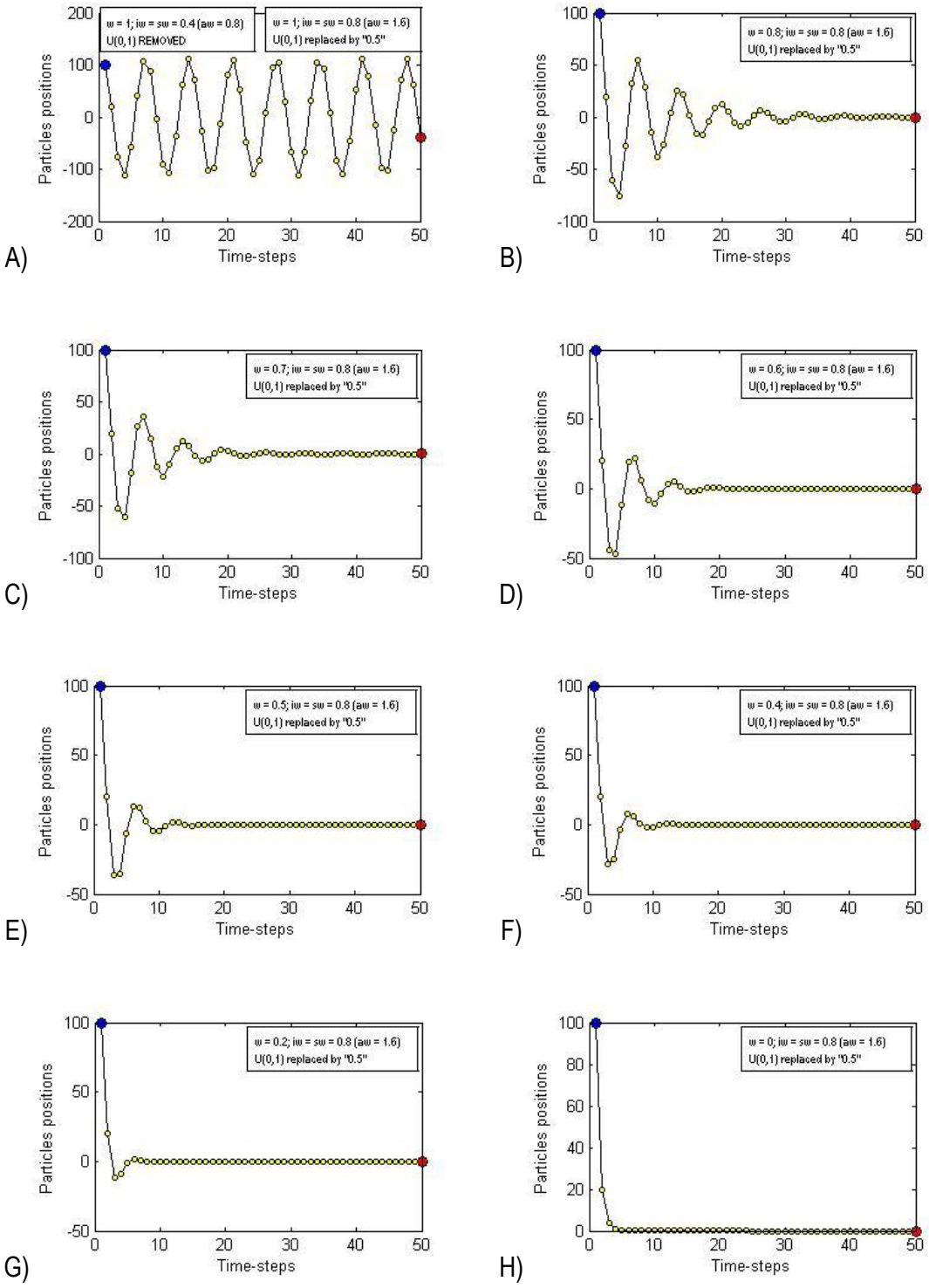


Fig. 5.46. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.00 ≥ w ≥ 0.00 and aw = 1.60. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.14. Acceleration weight *aw* = 1.20

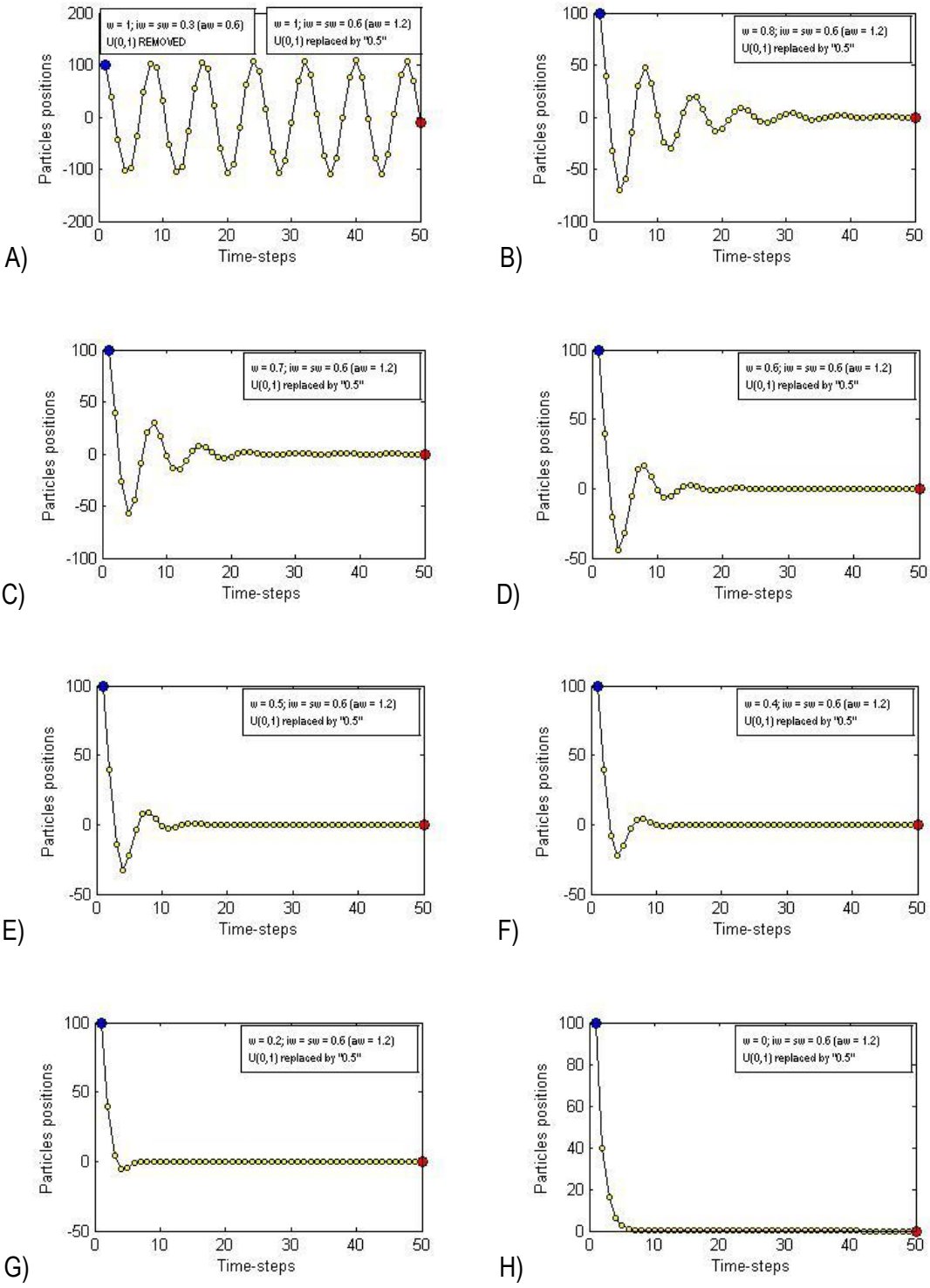


Fig. 5.47. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.00 ≥ w ≥ 0.00 and aw = 1.20. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

5.8.1.15. Acceleration weight *aw* = 0.80

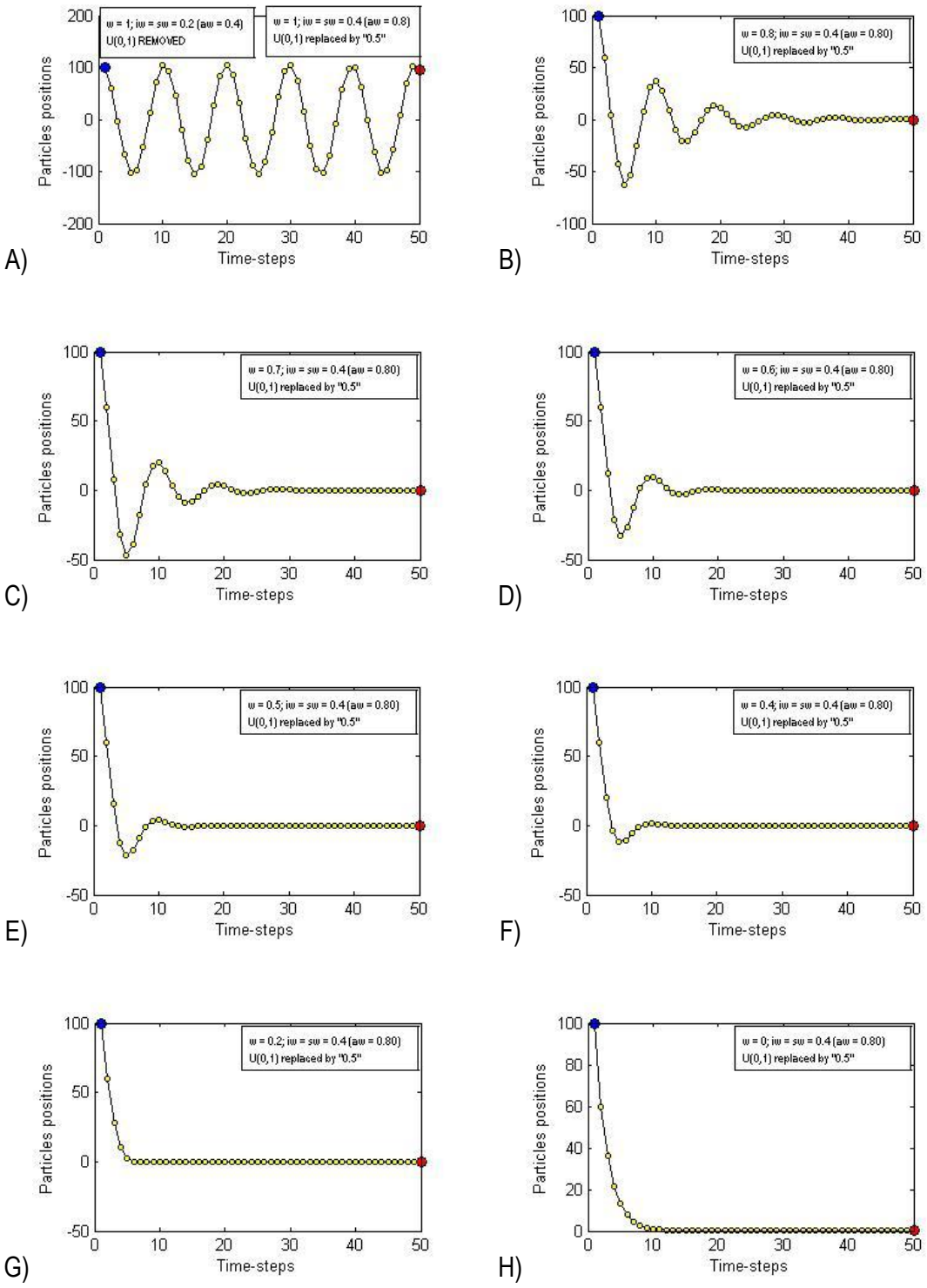


Fig. 5.48. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 1.00 ≥ w ≥ 0.00 and aw = 0.80. Random weights $U_{(0,1)}$ are replaced by the average of the uniform distribution used to generate them (i.e. their expected value: '0.50').

It is important to note that the incorporation of randomness would be expected to widen the search area, and that the attractors are not stationary in the full algorithm. Therefore, a deterministic particle that is not able to converge in the 50 time-steps analyzed here would arguably take too long to fine-cluster in a full PSO system with random weights and moving attractors. It appears that, for the whole range of *aw* analyzed, the inertia weight should not be kept above '0.80' unless it decreases over time. Of course this is subjective, and depends on the available resources and acceptable computational cost. Two trajectories of the deterministic particle with $w = 0.90$ are shown in Fig. 5.49, for a high $aw = 4.00$ on the left, and for a low $aw = 0.80$ on the right. In turn, too low a *w* would dampen the momentum too quickly hence discouraging exploration.

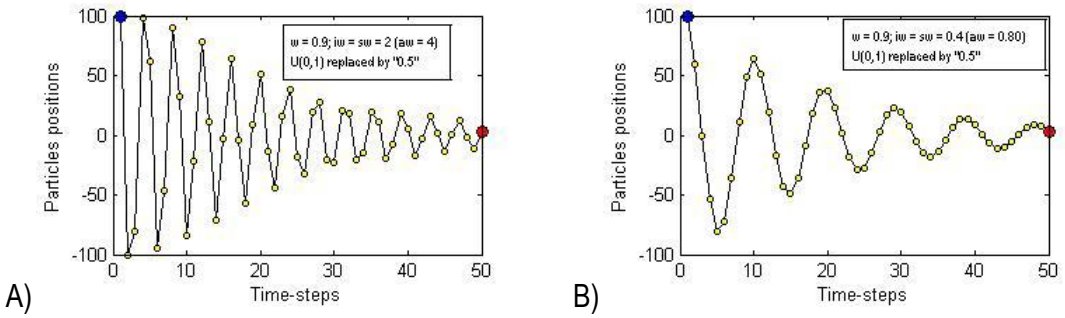


Fig. 5.49. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0, random weights $U_{(0,1)}$ replaced by their expected value '0.50', and w = 0.90. For the figure on the left, aw = 4.00. For the figure on the right, aw = 0.80.

If either *w* or *aw* are set to a small value, the other should not be small as well, so that at least one of them keeps some momentum. It is important to note, however, that they keep momentum in a very different manner. A trajectory for a low *w* and a high *aw*, and another for a high *w* and a low *aw*, are shown in Fig. 5.50.

It does not seem advisable to set $aw < 2.00$ because the attractor would be approached mainly from one side rather than repeatedly overflown, especially if *w* is also small. If *w* is set to a high value to compensate, the particle will approach the attractor from both sides but with a low frequency, as in Fig. 5.49 B) and Fig. 5.50 B). Notice that the particle in Fig. 5.50 overflies the attractor 10 times in the first 10 time-steps for $w = 0.10$ and $aw = 4.00$, whereas it does only 4 times for $w = 0.60$ and $aw = 2.00$. Furthermore, for a unimodal function and a non-stationary attractor, the current position would also be the attractor, as there would be constant improvement, and only the inertia weight

times the previous velocity would keep the movement. Therefore if $w < 1.00$, the step-sizes would decrease exponentially and it might even be impossible for the particle to reach the attractor. A practical example of this can be found in (Clerc, 2006a, pp. 212-214). Of course, in a multi-particle system, only one particle per time-step –if any– would undergo this situation. Nevertheless, the problem of the quick loss of momentum would still persist, and it is therefore advisable to keep $aw \geq 2.00$.

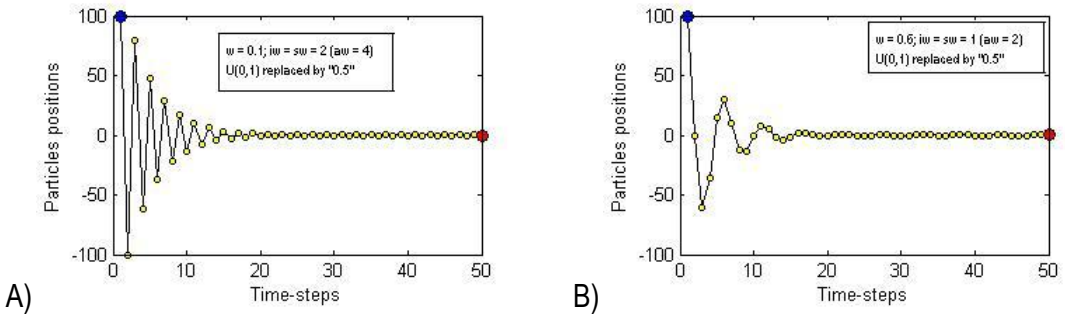


Fig. 5.50. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ replaced by their expected value '0.50', for w = 0.10 and aw = 4.00 (left), and for w = 0.60 and aw = 2.00 (right).

5.8.2. Random weight incorporated

Complexity has to be gradually re-incorporated to the simplified model of a single, deterministic particle with stationary attractors. The first step is to re-incorporate the random weight so as to visually study how similar or different from the analyses in section 5.8.1 the behaviour of the particle is. Only some selected coefficients taken from the previous section are analyzed for obvious reasons.

The first column in Fig. 5.51 shows a series of trajectories of the particle with stationary attractors and the random weights replaced by their expected value, for $aw = 4.00$ and various settings of the inertia weight. The second column shows the trajectories for the same settings except that the random weights are re-incorporated. Thus, two figures in the same row share the same inertia and acceleration weights. As usual, six runs were performed for the cases with the random weights included, whose images can be found in the digital appendix. Fig. 5.52 shows six trajectories for $w = 0.75$ and $aw = 4.00$, while Fig. 5.53 shows six trajectories for $w = 0.50$ and $aw = 4.00$.

5.8.2.1. Acceleration weight *aw* = 4.00

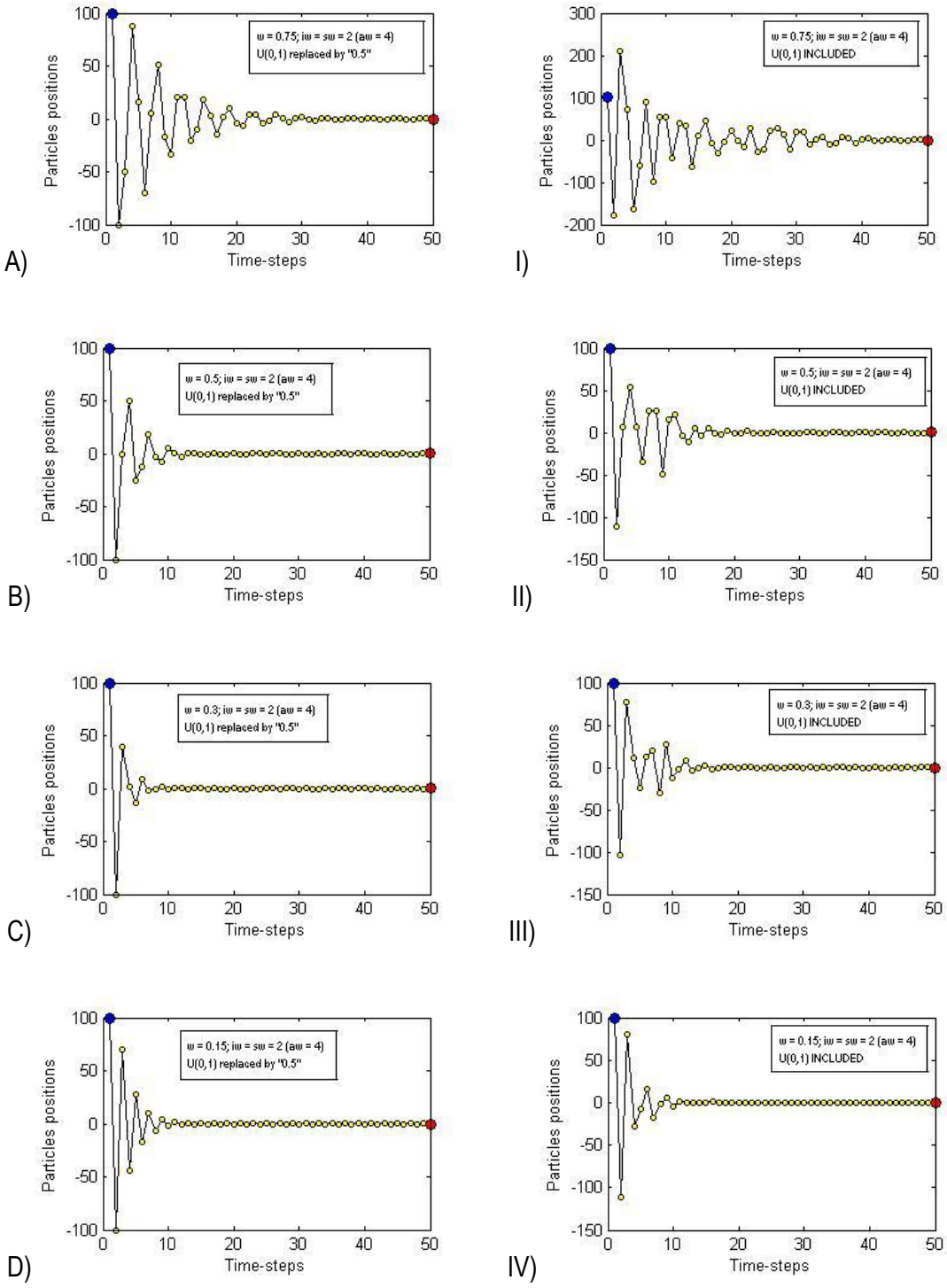


Fig. 5.51. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.75 ≥ w ≥ 0.15 and aw = 4.00. Random weights $U_{(0,1)}$ are included in the second column, and replaced by the average of the uniform distribution used to generate them (i.e. their expected value '0.50') in the first one.

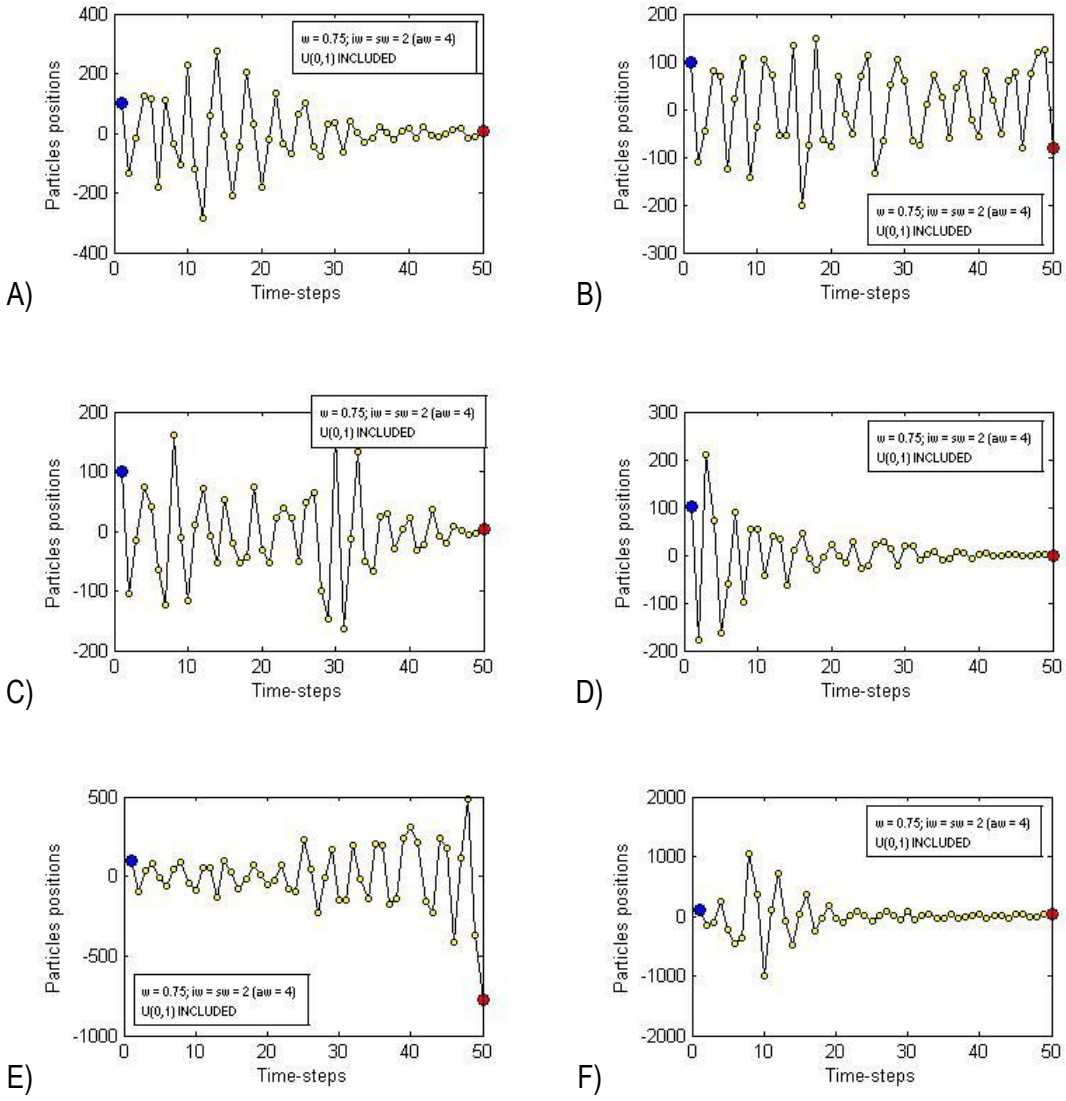


Fig. 5.52. Six possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ included for w = 0.75 and aw = 4.00.

It would be a mistake to assume that the greater the percentage of the ϕ interval that falls within the convergent region the faster the convergence. Recall that the different points within the convergent area in Fig. 5.20 lead to different values of the roots of the characteristic polynomial, and therefore to different convergence speed. The horizontal line for $w = 1.00$ and the top of the black triangle are points with the module of the roots equal to '1'. An experimentally obtained map representing the magnitude of the root with the maximum magnitude can be found in (van den Bergh, 2001, p. 84).

Consider, for instance, the setting $w = 0.75$ and $aw = 4.00$ for which ϕ will be within the convergent area 87.5% of the time. Compare that to the setting $w = 0.50$ and

$aw = 4.00$ for which ϕ will be within the convergent area 75% of the time. However, as can be observed in Fig. 5.51 A), B), I) and II), the convergence speed is notably higher for $w = 0.50$ both with the random weights and with their expected value. This is even more notorious when comparing all six runs in Fig. 5.52 to those in Fig. 5.53. A sense of why this happens can be obtained by noting that the module of the roots for $w = 0.75$ is approximately equal to 0.8660 whereas the module for $w = 0.50$ is approximately equal to 0.7071 (the smaller the module the faster the convergence). A more accurate measure would be given by integrating the modules for the whole range of ϕ.

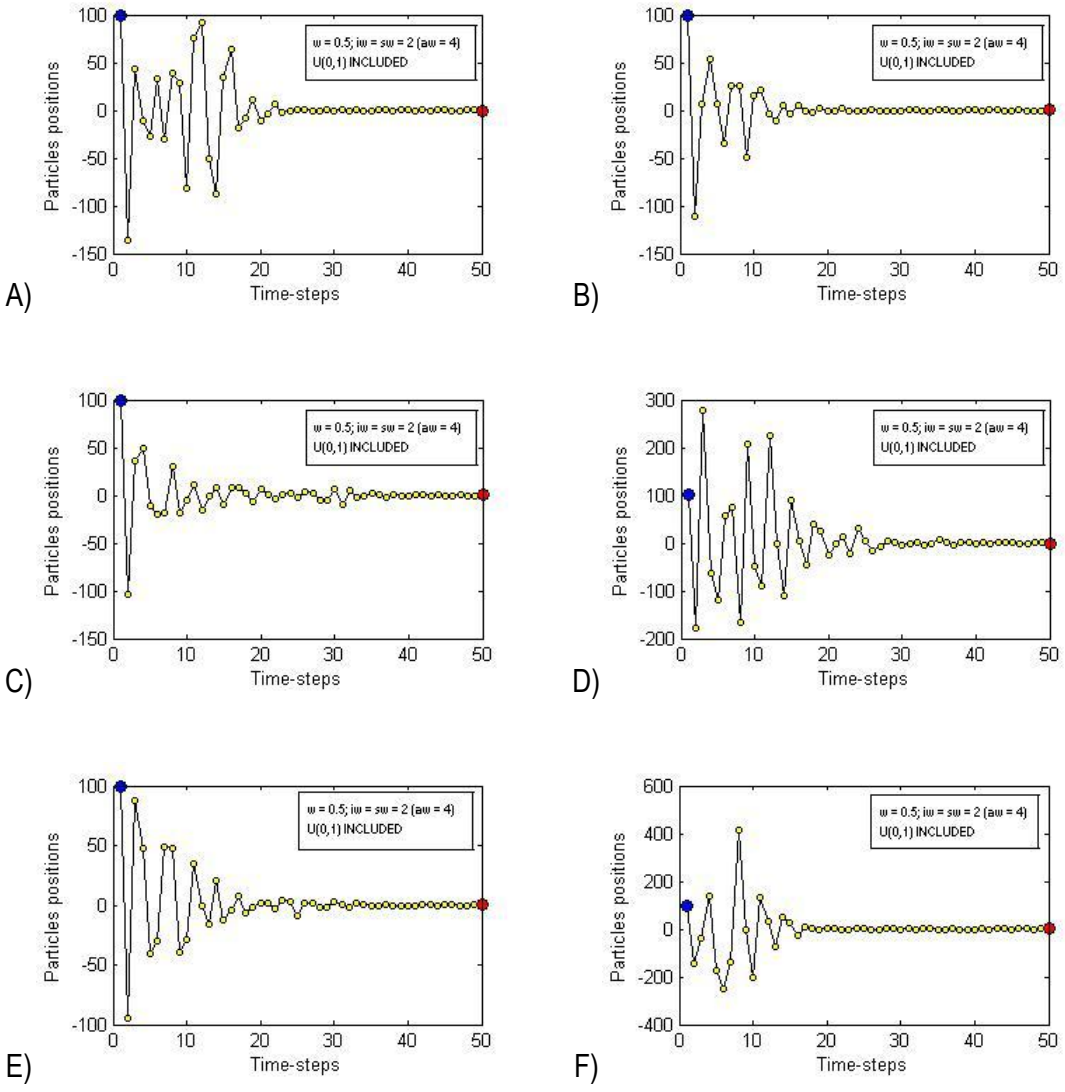


Fig. 5.53. Six possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ included for w = 0.50 and aw = 4.00.

As can be observed in Fig. 5.52 E), the stochastic explosion can still occur with an inertia weight as small as 0.75. For $w \leq 0.60$ (not included here), all six runs end up clustering by 50th time-step. Even for inertia weights as low as 0.30 and 0.15, some explosion might still occur due to the embedded randomness, although the particle is eventually pulled back to the attractor. Refer to the digital appendix for the images.

Six trajectories for $w = 0.75$ and $aw = 3.60$ are shown in Fig. 5.54. Comparing the latter with Fig. 5.52 ($w = 0.75$ and $aw = 4.00$), it can be observed that convergence is noticeably improved by a small decrease in the acceleration weight.

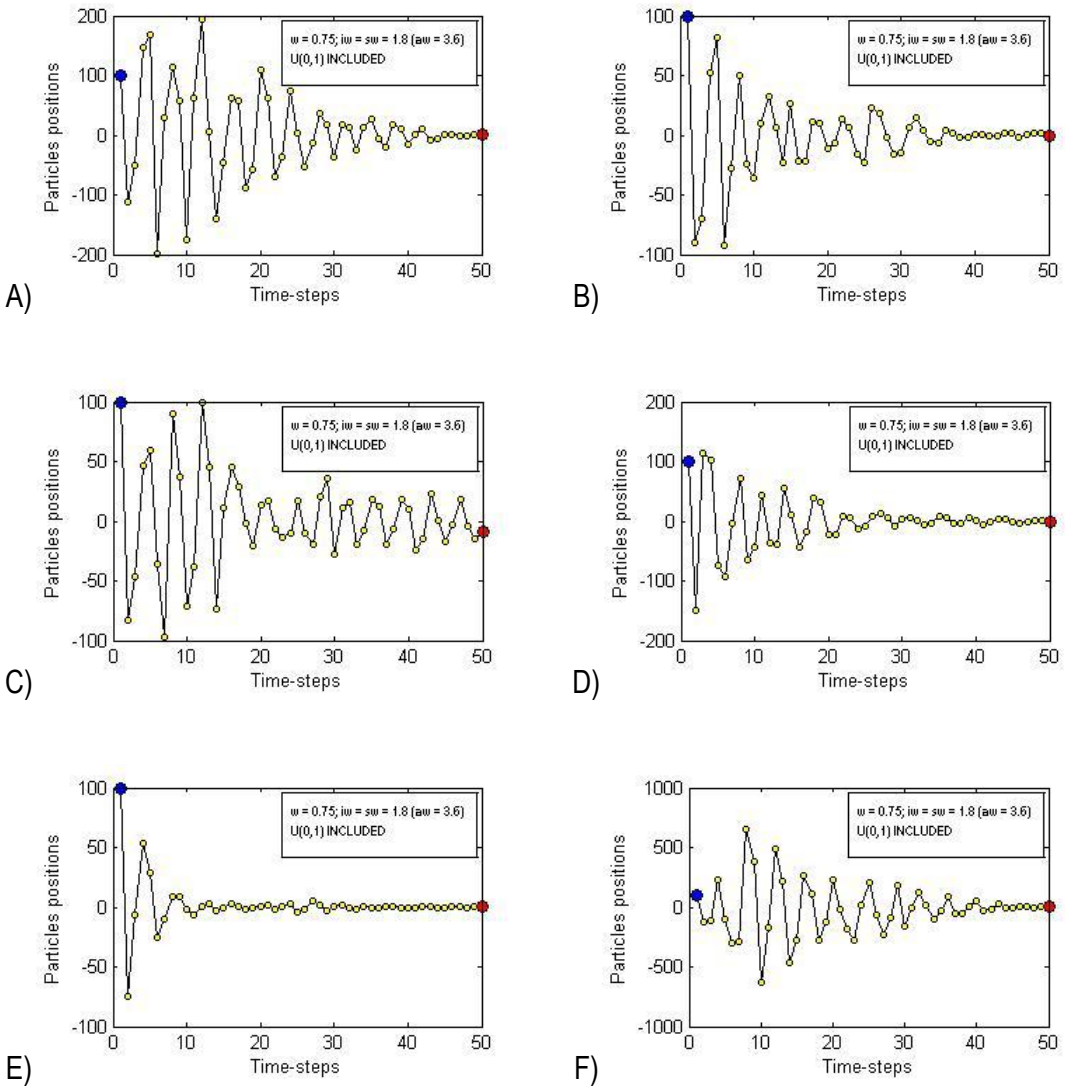


Fig. 5.54. Six possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ included for w = 0.75 and aw = 3.60.

For aw = 3.60, ϕ will be within the convergent area 97.22% of the time, against the 87.5% of the time for $aw = 4.00$. In addition, since $w = 0.75$ in both cases, the interval of ϕ for $aw = 3.60$ is entirely contained within the one for $aw = 4.00$, only disregarding the worst, rightmost bit ('0.4-long' segment within the black triangle in Fig. 5.20).

As w is decreased for the same $aw = 3.60$, the percentage of the ϕ interval that is within the divergent black triangle increases. This leads to a less uniform trajectory of the particle with random weights. Fig. 5.55 shows two quite different trajectories for the same settings: $w = 0.30$, $aw = 3.60$ (random weights included). The same is true in Fig. 5.56 for $w = 0.15$. Note, nevertheless, that the convergence is still faster than for $w = 0.75$ despite having coefficients ϕ within the divergent area more often. Refer to the digital appendix for the other four runs carried out for the cases in Fig. 5.55 and Fig. 5.56.

Fig. 5.57 to Fig. 5.59 show trajectories of the particle with random weights included on the right columns and replaced by their expected value (i.e. '0.5') on the left columns, for $aw = 3.60$, $aw = 3.20$, and $aw = 3.00$, respectively, and various settings of w.

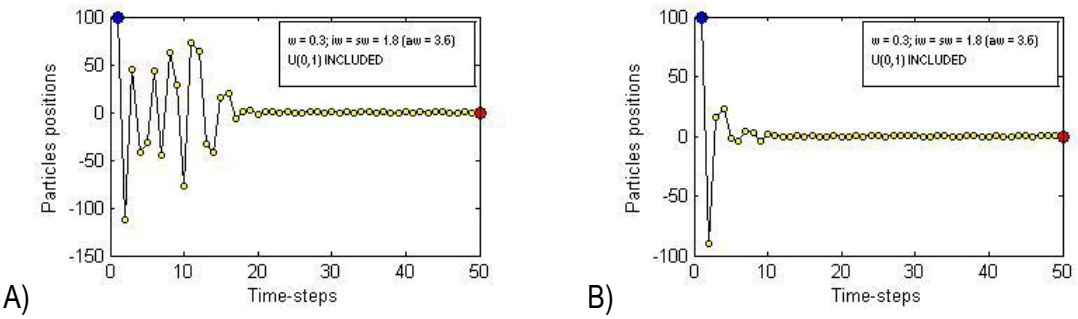


Fig. 5.55. Two quite different possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ included for w = 0.30 and aw = 3.60.

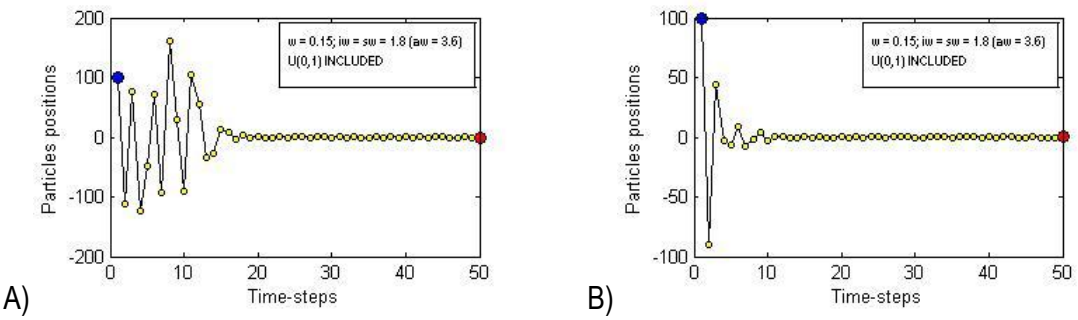


Fig. 5.56. Two quite different possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ included for w = 0.15 and aw = 3.60.

5.8.2.2. Acceleration weight *aw* = 3.60

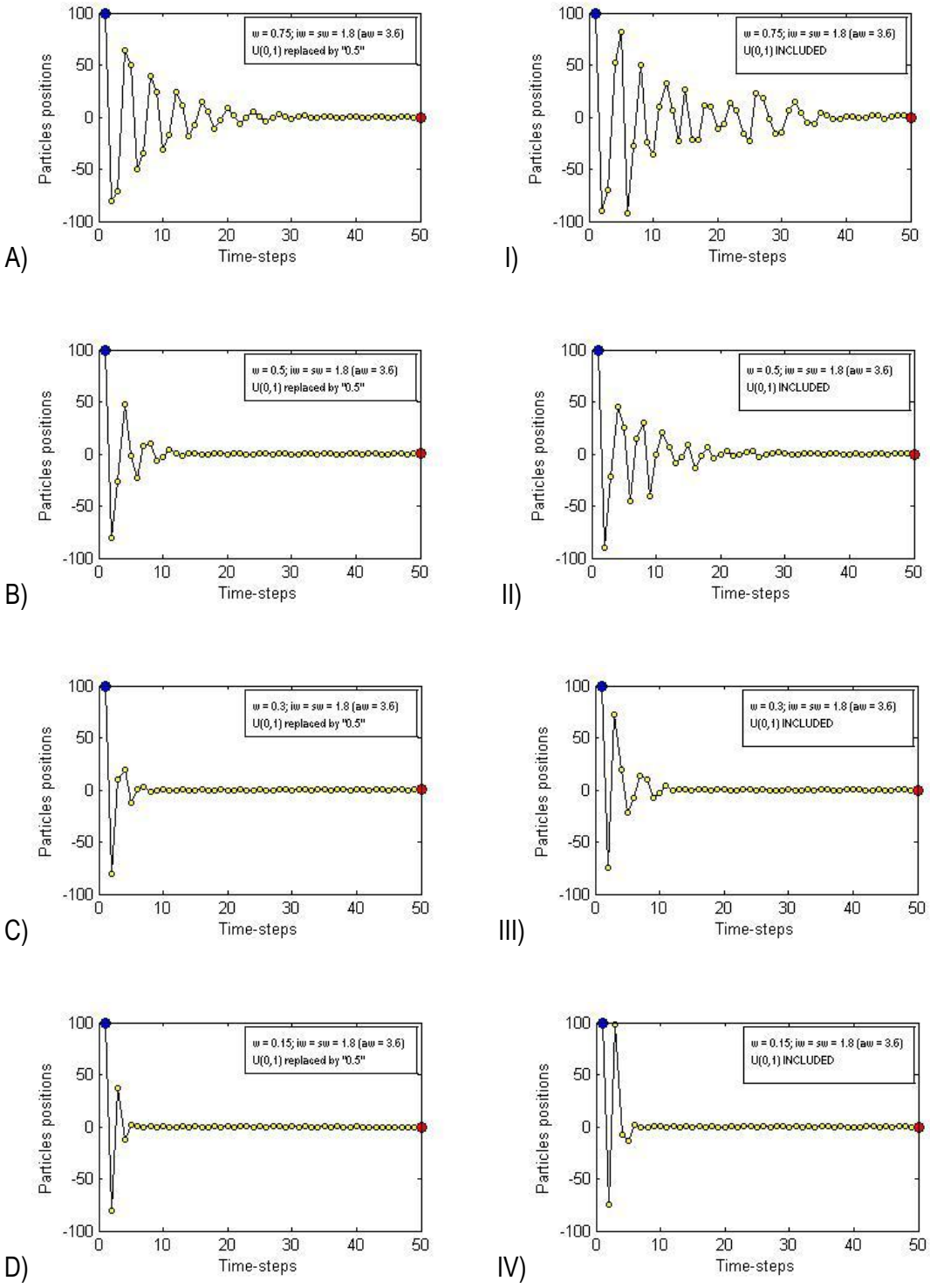


Fig. 5.57. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.75 ≥ w ≥ 0.15 and aw = 3.60. Random weights $U_{(0,1)}$ are included in the second column, and replaced by the average of the uniform distribution used to generate them (i.e. their expected value '0.50') in the first one.

5.8.2.3. Acceleration weight *aw* = 3.20

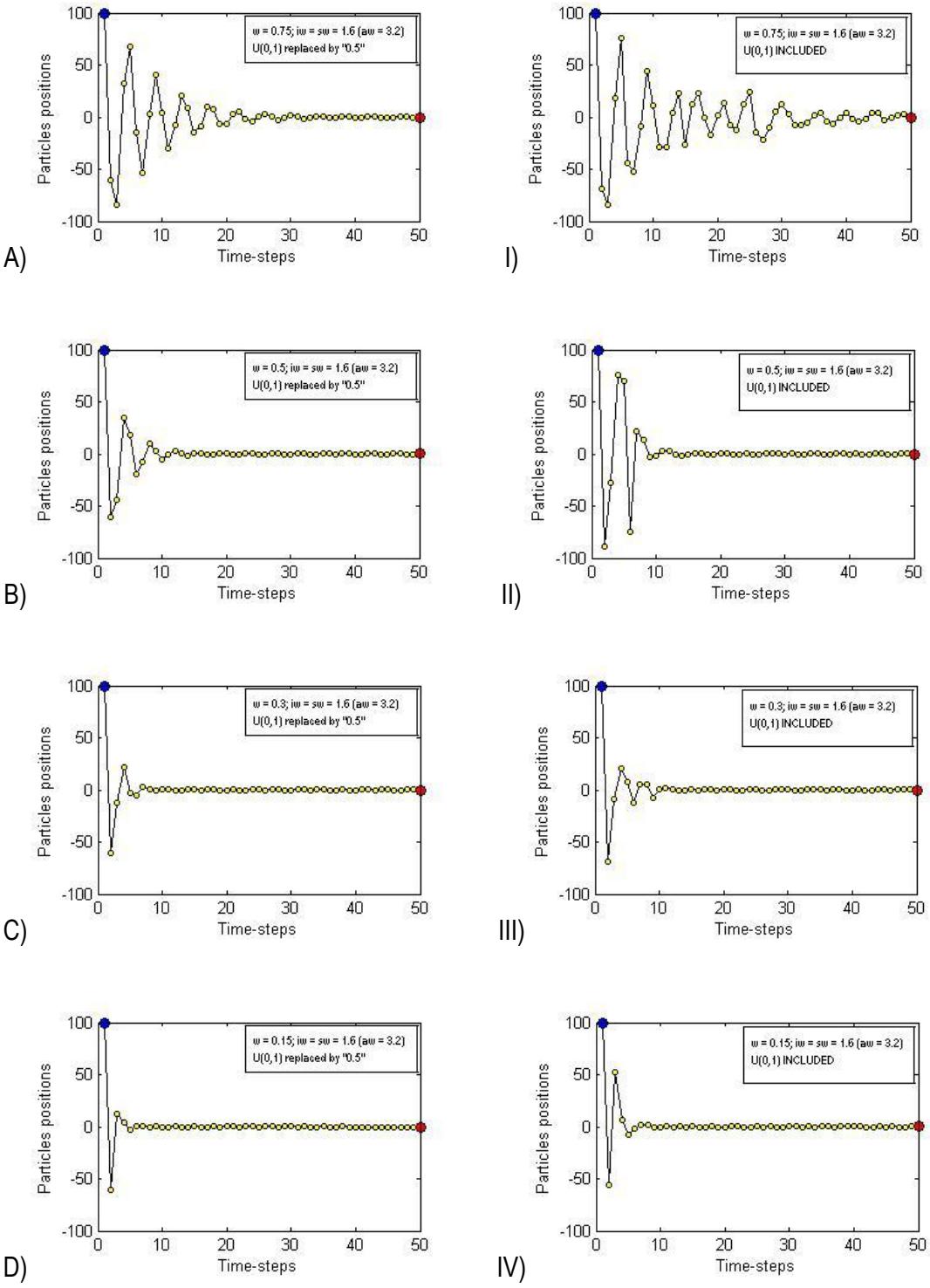


Fig. 5.58. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.75 ≥ w ≥ 0.15 and aw = 3.20. Random weights $U_{(0,1)}$ are included in the second column, and replaced by the average of the uniform distribution used to generate them (i.e. their expected value '0.50') in the first one.

5.8.2.4. Acceleration weight *aw* = 3.00

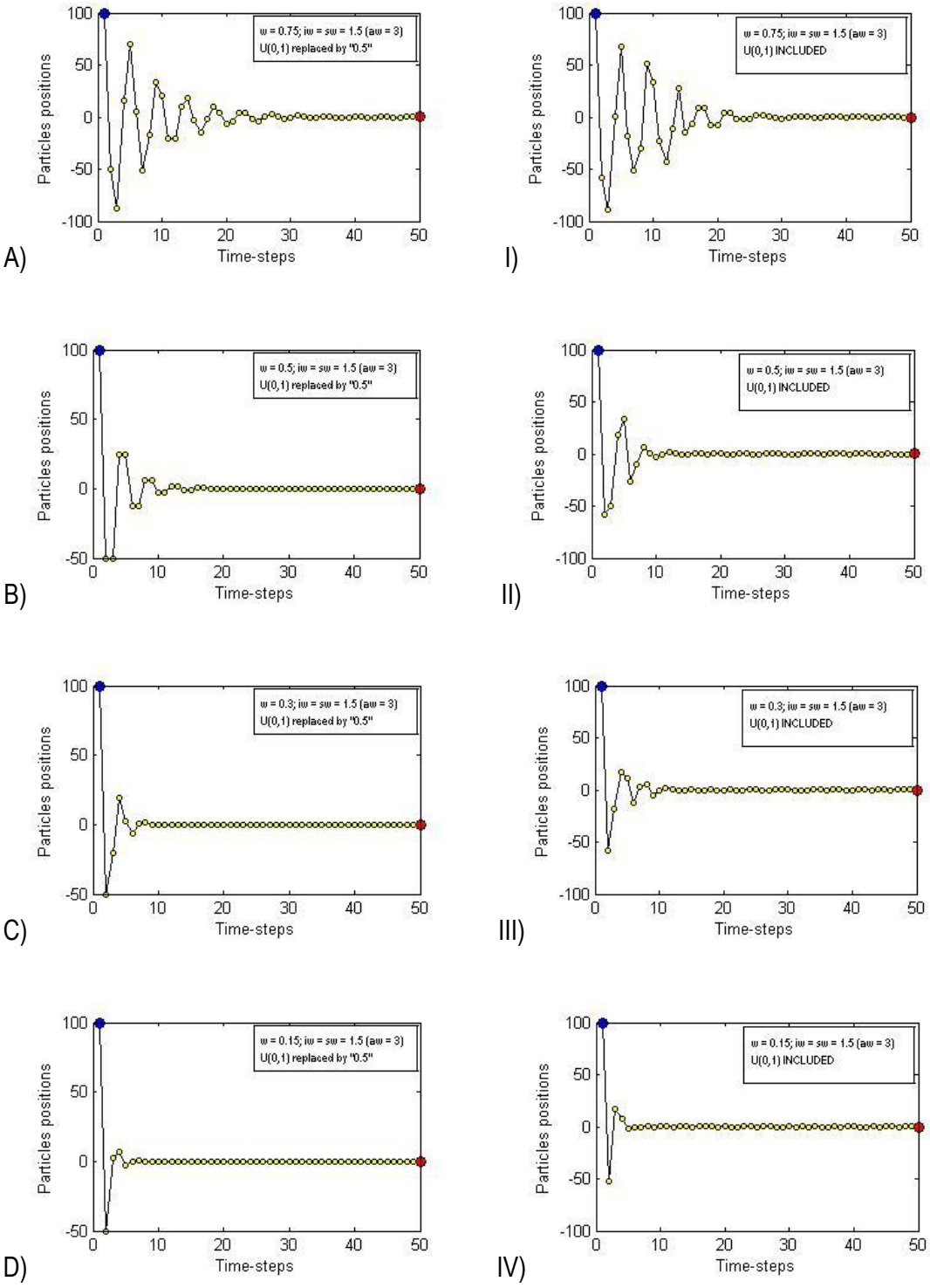


Fig. 5.59. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.75 ≥ w ≥ 0.15 and aw = 3.00. Random weights $U_{(0,1)}$ are included in the second column, and replaced by the average of the uniform distribution used to generate them (i.e. their expected value '0.50') in the first one.

As argued for the trajectories without randomness in section 5.8.1, the particle loses momentum too quickly for low inertia weights as the acceleration weight decreases. If extremely fast convergence is sought, settings like in Fig. 5.59 IV) are a good choice, but they are not robust and premature convergence is to be expected. Six trajectories for $w = 0.75$ and $aw = 3.00$ are shown in Fig. 5.60 (the particle converges every time).

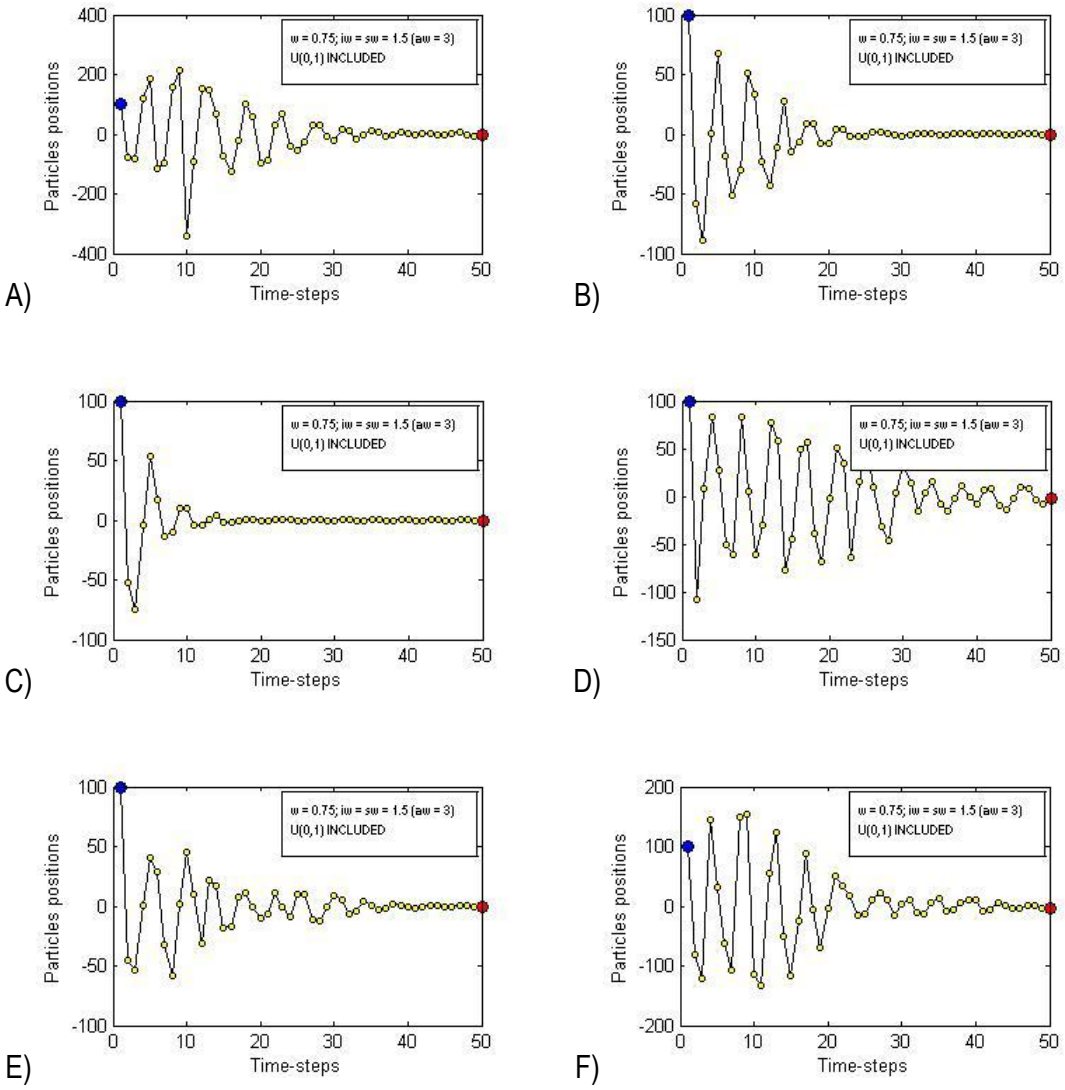


Fig. 5.60. Six possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ included for w = 0.75 and aw = 3.00.

Fig. 5.61 and Fig. 5.62 show trajectories of the particle with random weights included on the right columns and replaced by their expected value (i.e. '0.5') on the left ones, for $aw = 2.80$ and $aw = 2.40$, respectively, and various settings of w.

5.8.2.5. Acceleration weight *aw* = 2.80

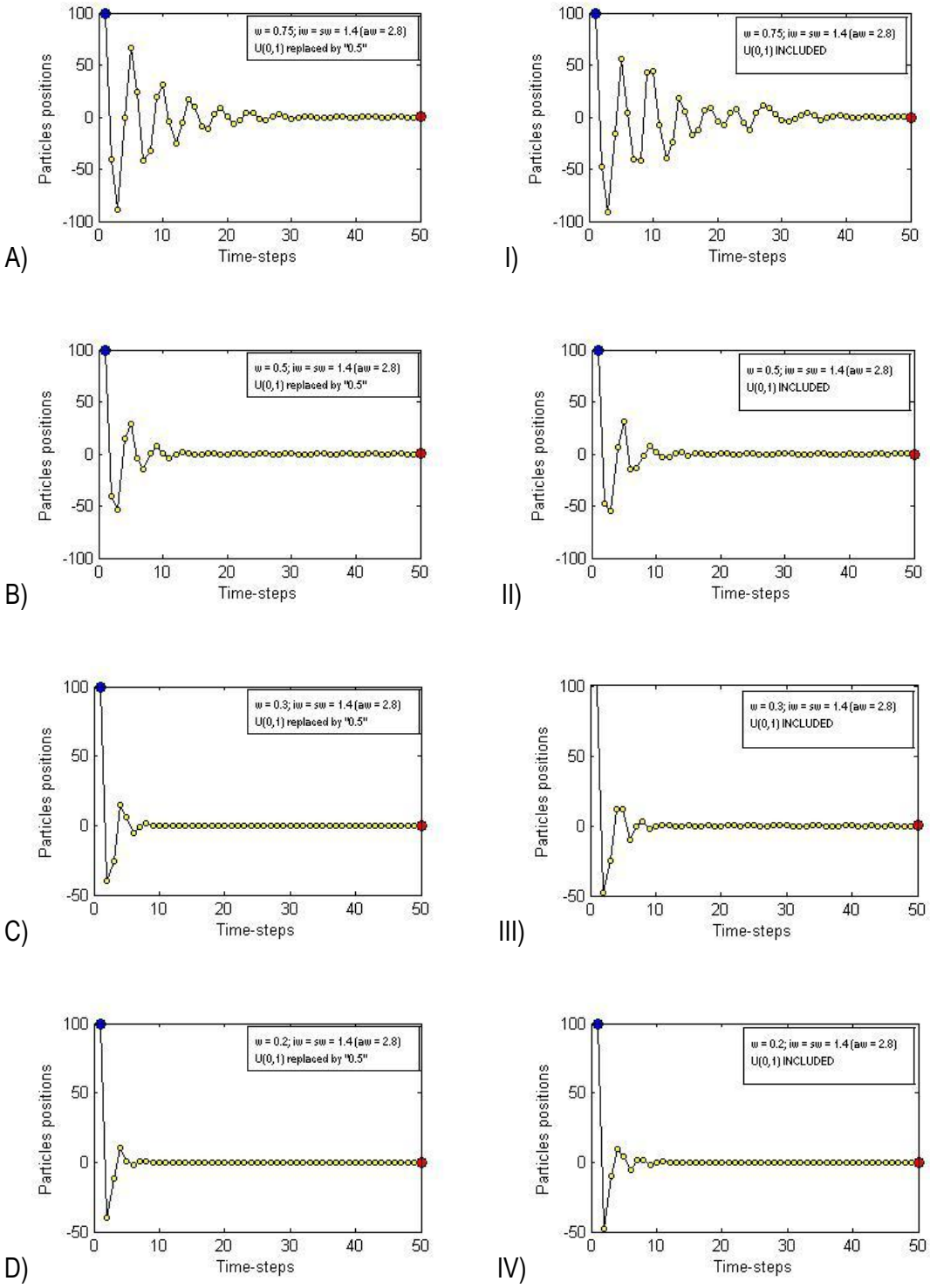


Fig. 5.61. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.75 ≥ w ≥ 0.20 and aw = 2.80. Random weights $U_{(0,1)}$ are included in the second column, and replaced by the average of the uniform distribution used to generate them (i.e. their expected value '0.50') in the first one.

5.8.2.6. Acceleration weight *aw* = 2.40

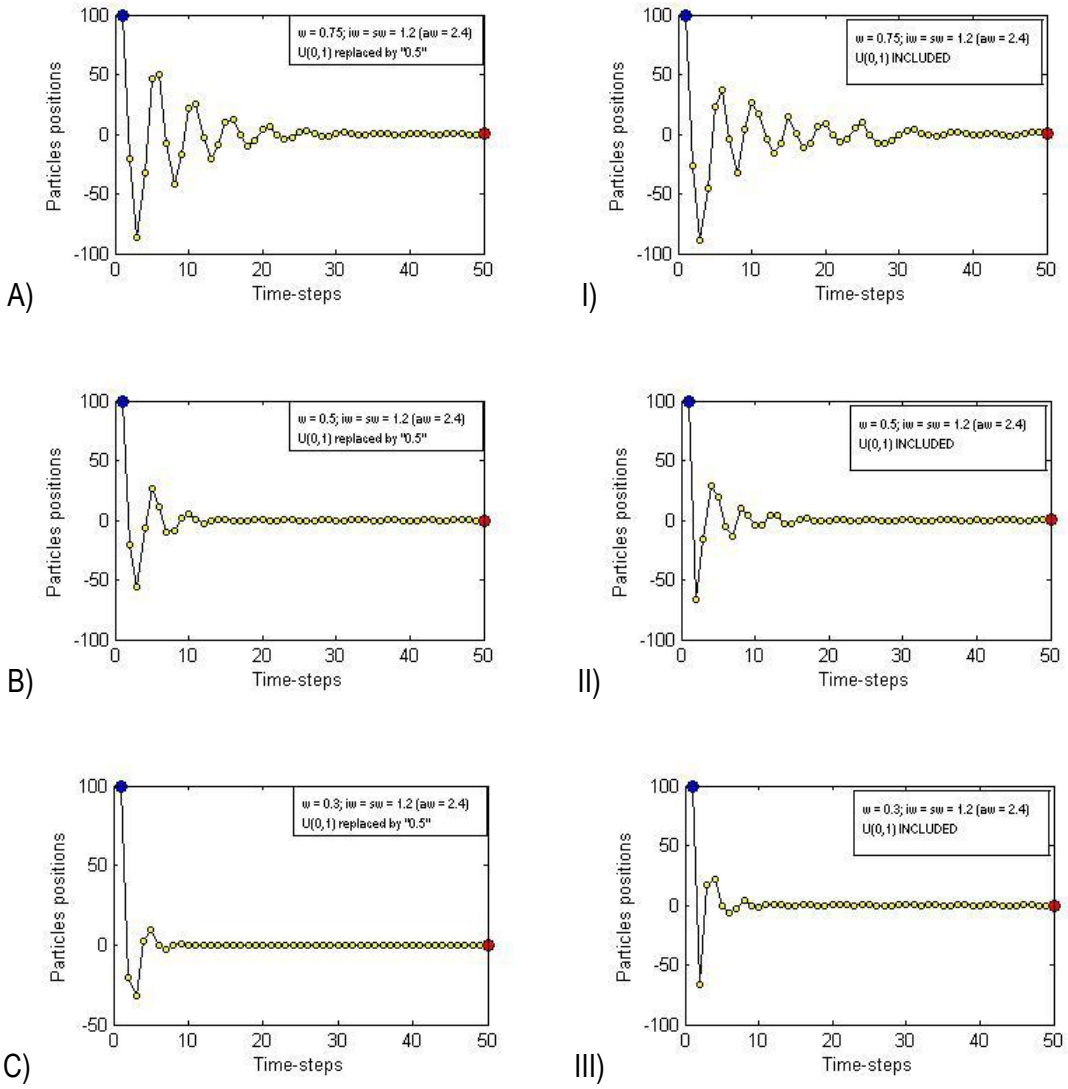


Fig. 5.62. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 0.75 ≥ w ≥ 0.30 and aw = 2.40. Random weights $U_{(0,1)}$ are included in the second column, and replaced by the average of the uniform distribution used to generate them (i.e. their expected value '0.50') in the first one.

These are around the smallest values advisable for the acceleration weight. For medium to small inertia weights, there is always fast convergence. However, for $w \cong 0.75$ there is still a reasonable trade-off between exploration and exploitation, with a final good convergence despite the randomness re-introduced. This can be observed in Fig. 5.63, which shows six trajectories of the particle for $w = 0.75$ and $aw = 2.40$ with the random weights included, corresponding to consecutive runs.

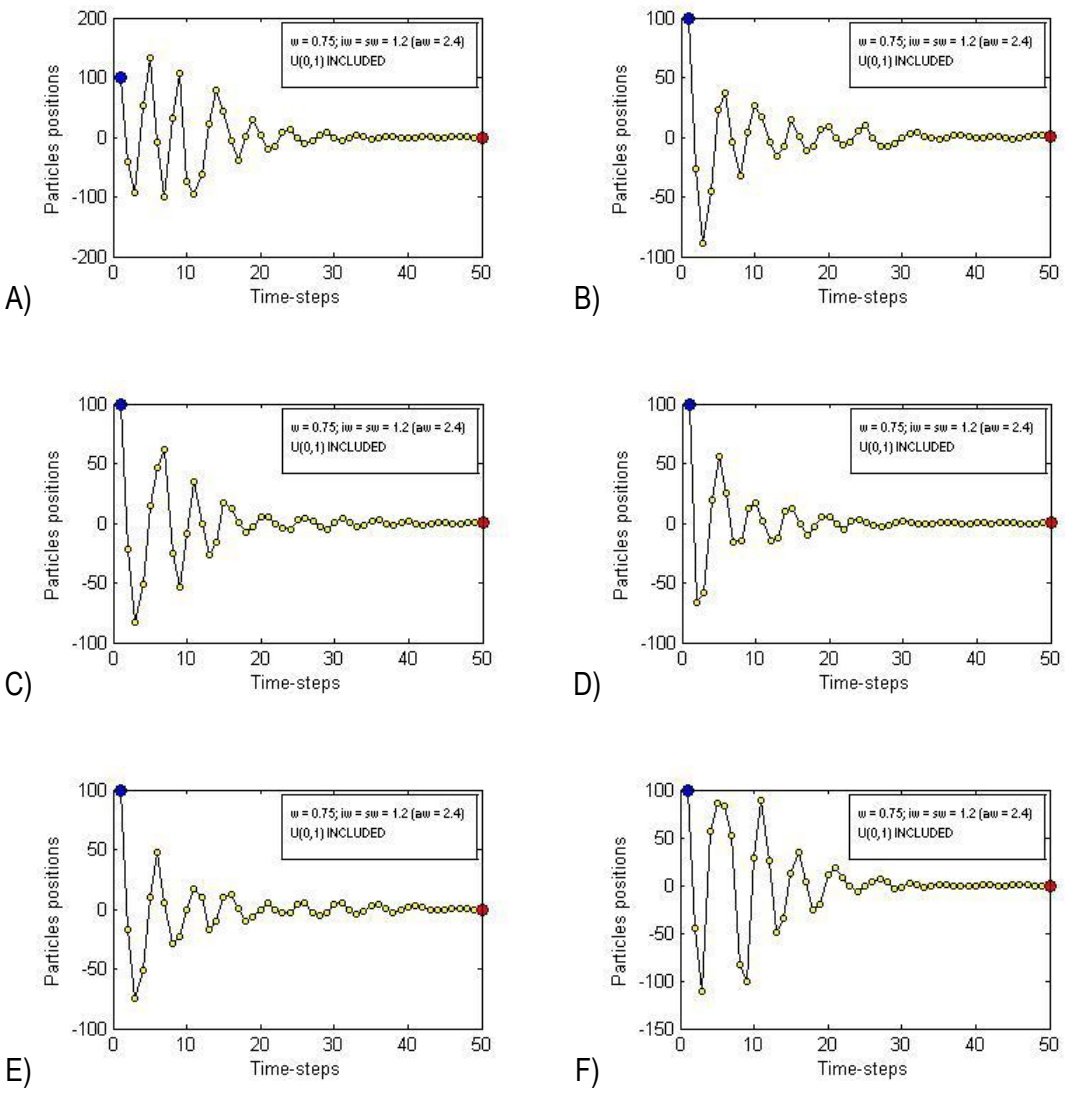


Fig. 5.63. Six possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 and random weights $U_{(0,1)}$ included for w = 0.75 and aw = 2.40.

Note that the settings in Fig. 5.59 I) and Fig. 5.61 I) are very similar to those suggested in (Clerc, 2006a, p. 49) ($w = 0.70$; $aw = 2.86$), and to a popular setting derived from the constriction factor *type 1"* in (Clerc & Kennedy, 2002) ($w \cong 0.73$; $aw \cong 2.99$).

Just for the sake of illustration, the trajectories of the particle for lower acceleration weights are offered in Fig. 5.64 to Fig. 5.66. Only two inertia weights are considered, as smaller values are already out of the question. Again, the trajectories corresponding to the random weights included are on the right columns, and those corresponding to their expected values are on the left ones.

5.8.2.7. Acceleration weight *aw* = 2.00

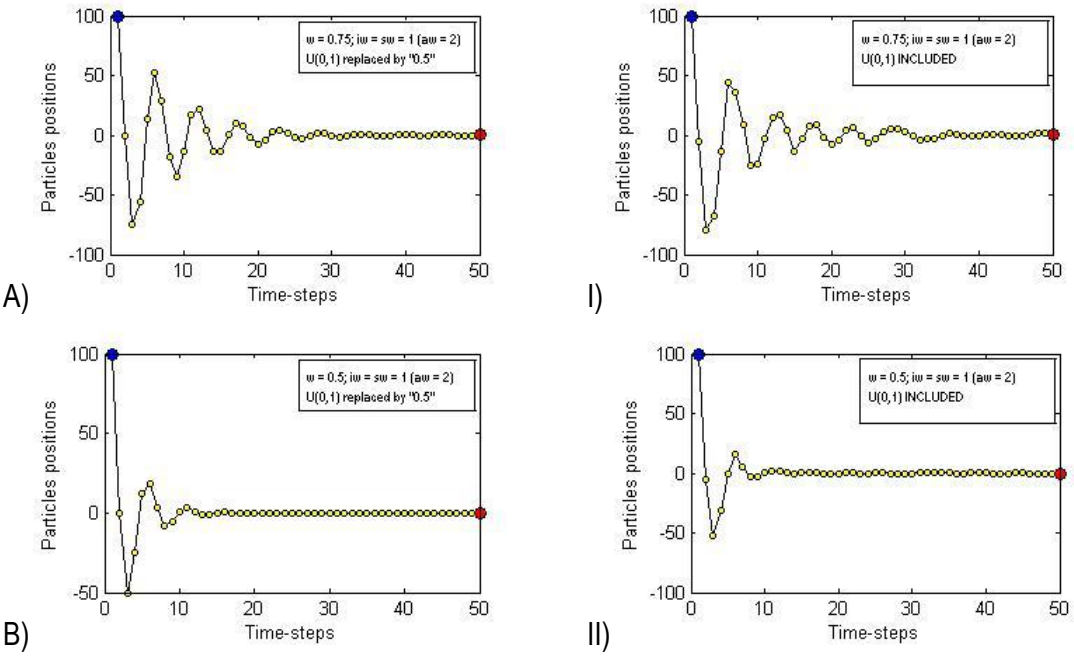


Fig. 5.64. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.75; w = 0.50; and aw = 2.00. Random weights $U_{(0,1)}$ are included in the second column, and replaced by the average of the uniform distribution used to generate them (i.e. their expected value '0.50') in the first one.

5.8.2.8. Acceleration weight *aw* = 1.60

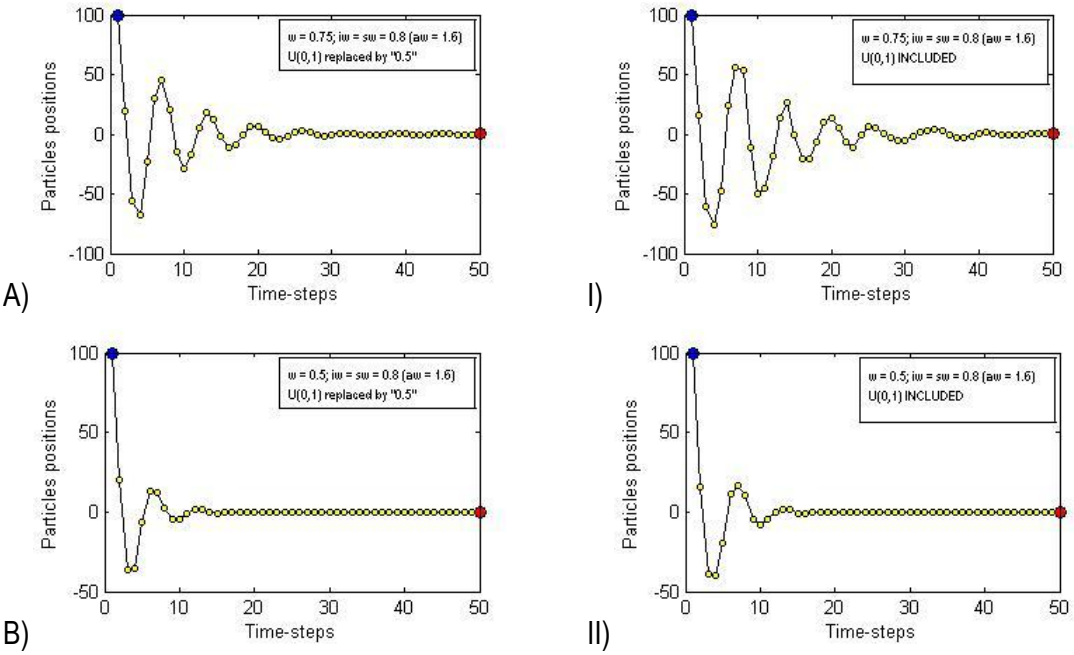


Fig. 5.65. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.75; w = 0.50; and aw = 1.60. Random weights $U_{(0,1)}$ are included in the second column, and replaced by the average of the uniform distribution used to generate them (i.e. their expected value '0.50') in the first one.

5.8.2.9. Acceleration weight *aw* = 1.20

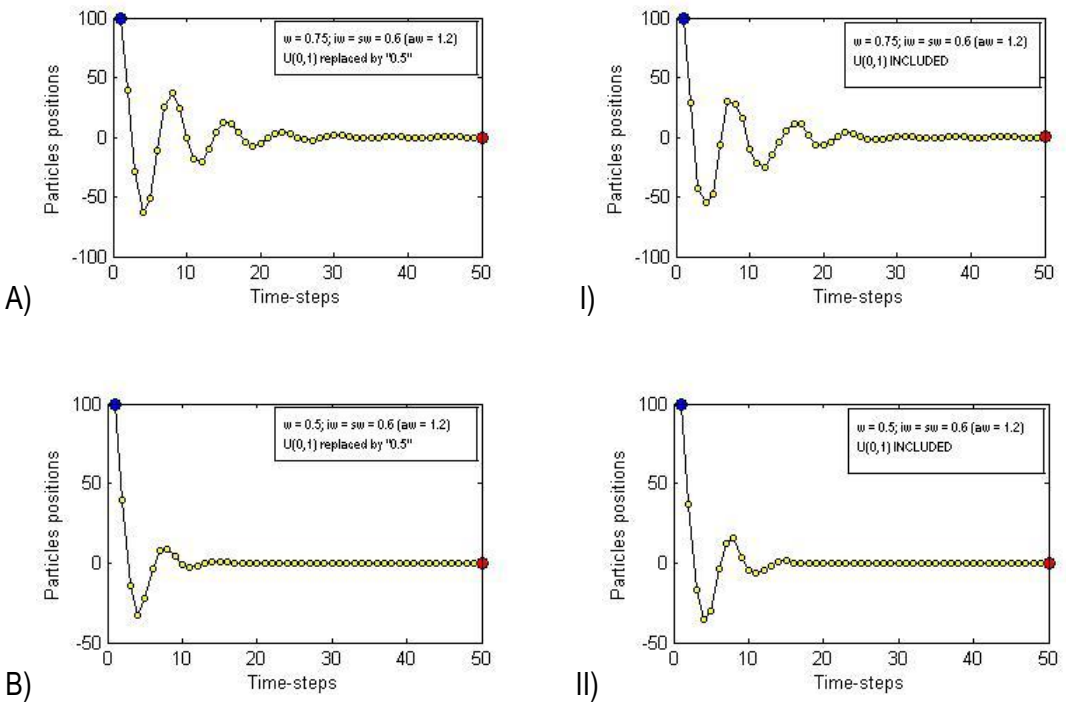


Fig. 5.66. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.75; w = 0.50; and aw = 1.20. Random weights $U_{(0,1)}$ are included in the second column, and replaced by the average of the uniform distribution used to generate them (i.e. their expected value '0.50') in the first one.

For very small values of the acceleration weight, the influence of the randomness decreases, and the trajectories for the random weights included and those for them replaced by their expected values are more alike.

5.8.3. Velocity constraint incorporated

For $w<1$ and $aw\leq 4$, the implementation of the velocity constraint is not essential. Nonetheless, it may help avoid evaluations of the particle's position far from the region of interest by preventing local explosions. However, if the size of the displacement in each dimension is over-restricted, the explorative capabilities of the algorithm may be compromised, and the normal dynamics of the swarm over-affected. Think, for instance, of a particle that is far from its attractor, and therefore the calculated displacement is big. Suppose the velocity constraint is set to a small value. Then, the displacement in that dimension will have the same value in a number of consecutive time-steps, disre-

garding the randomness on which the PSO algorithm relies. In addition, a small value may also prevent a particle from jumping from a feasible island to another in a constrained problem, or from one peak/valley to another in multimodal problems.

In order to avoid the problem-dependent setting of v_{max}, it is of common practice to link it to the interval constraints of the search-space. As shown in Fig. 5.25 III) in section 5.7, setting v_{max} as in Eq. (5.88) narrows the explored search-space approximately to the area bounded by the interval constraints, thus preventing local explosions without excessively interfering with the normal dynamics of the system. Therefore Eq. (5.88) comprises a good setting, especially when the constraint is not essential ($w < 1$).

$$v_{\max\, j} = 0.5 \cdot \left(x_{\max\, j} - x_{\min\, j} \right) = 0.5 \cdot \mathit{fint} \tag{5.88}$$

where *j* identifies the dimension and *fint* stands for 'feasible interval'. Note that the value may be set differently for different dimensions in multidimensional problems.

5.8.3.1. Acceleration weight *aw* = 4.00

For $w = 0.75$ and $aw = 4.00$, the trajectories of the particle without v_{max}, and with v_{max} set to half and to a quarter of the feasible interval (*fint*) is offered in Fig. 5.67. The random weights are included in the right column, and replaced by their expected value in the left one. Note that for $v_{max} = 0.25 \cdot \mathit{fint}$, the first two time-steps are the same size. As can be observed, $v_{max} = 0.50 \cdot \mathit{fint}$ keeps the particle within the region of interest without over-constraining its trajectory, whereas $v_{max} = 0.25 \cdot \mathit{fint}$ further narrows the region of the search-space explored. An even smaller setting for v_{max} would appear unnecessarily over-restrictive.

Aiming for a more comprehensive visualization of the trajectory with random weights included, six consecutive runs were performed for $w = 0.75$ and $aw = 4.00$ with v_{max} set to half the feasible interval on the one hand, and without the constraint on the other. The trajectories are shown in Fig. 5.68 and Fig. 5.69, where consecutive rows imply consecutive runs. The trajectories on the left column do not have the velocity constrained, whereas the ones on the right column have it restricted as in Eq. (5.88). Same row indicates same initial state of the pseudo-random number generator.

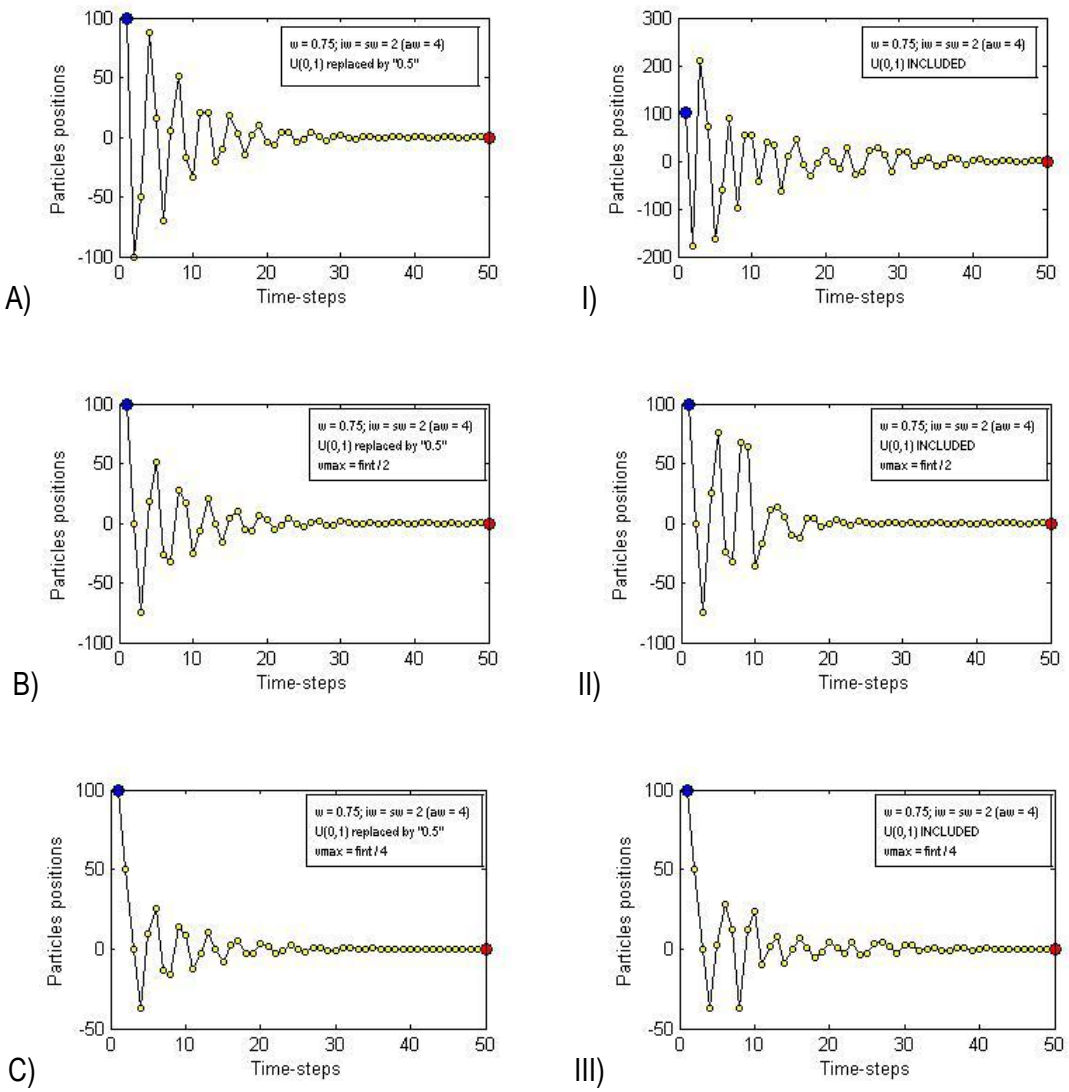


Fig. 5.67. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.75 and aw = 4.00. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected values in the first one. Different rows stand for different v_{max} constraints, the first one being unrestricted.

Fig. 5.69 A) and I) clearly show how constraining the size of the displacement helps prevent the explosion without having any direct effect on the convergence capabilities.

The effect of further constraining the displacement to a quarter of the feasible interval is shown in Fig. 5.70, where the trajectories corresponding to six consecutive runs are displayed. By comparing Fig. 5.70 E) to Fig. 5.69 I) –corresponding to the same initial state of the pseudo-random number generator–, it can be clearly observed again that a smaller v_{max} further narrows the region of the search-space being explored but does not improve the convergence capabilities.

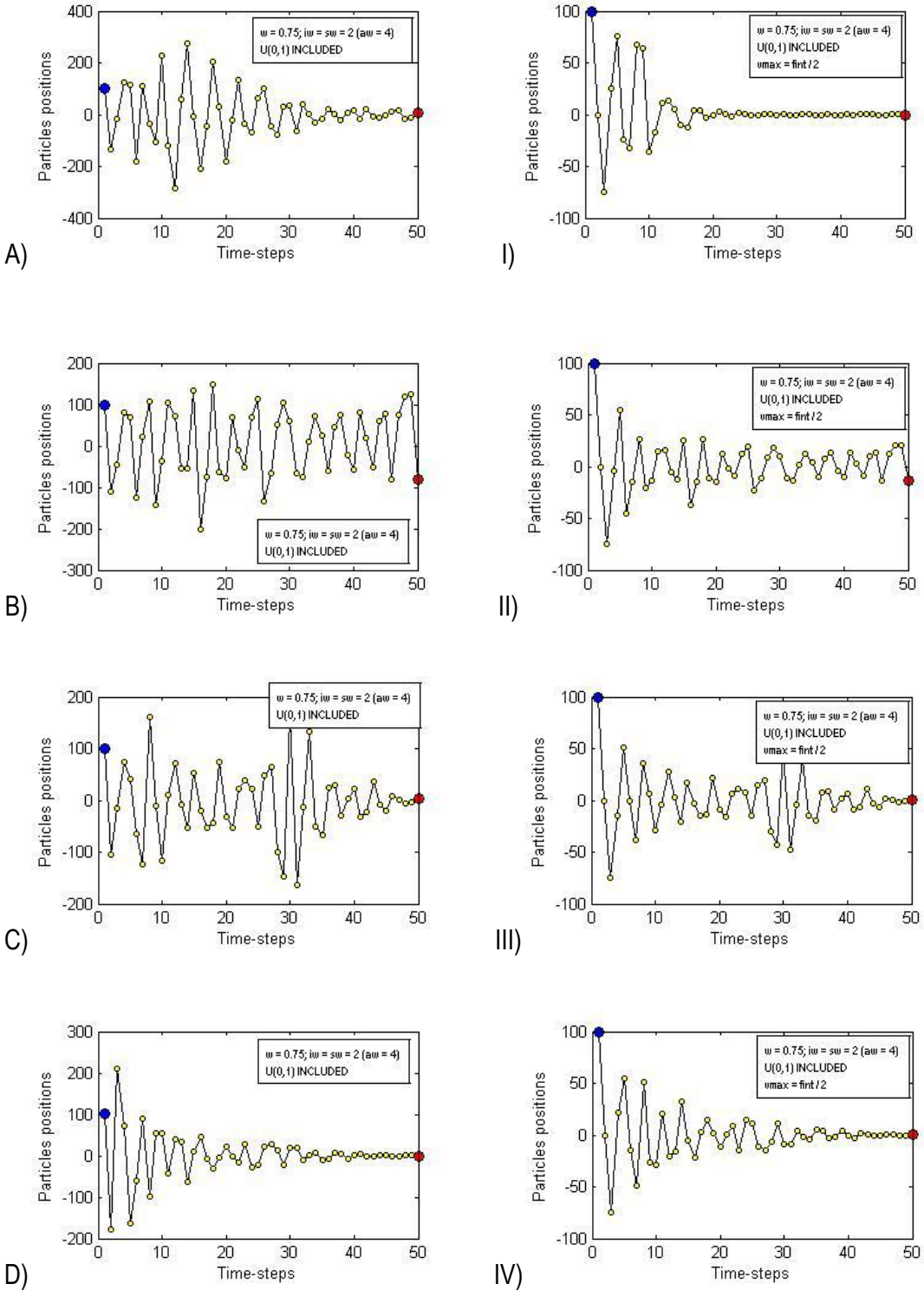


Fig. 5.68. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.75, aw = 4.00, and random weights $U_{(0,1)}$ included. The trajectories on the left column have no restriction to the velocity, whereas those on the right one have it constrained to half the feasible interval ($fint$). Consecutive rows correspond to consecutive runs, so that the figures in the same row have the same initial state of the pseudo-random number generator.

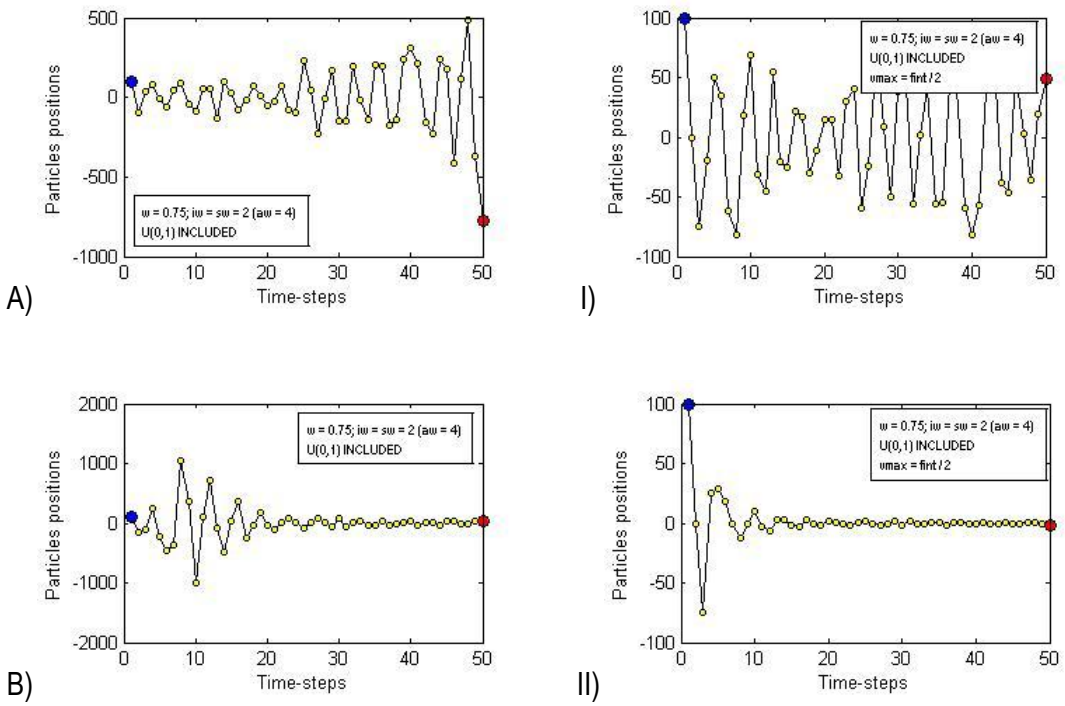


Fig. 5.69. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.75, aw = 4.00, and random weights $U_{(0,1)}$ included. The trajectories on the left column have no restriction to the velocity, whereas those on the right one have it constrained to half the feasible interval (*fint*). Consecutive rows correspond to consecutive runs, so that the figures in the same row have the same initial state of the pseudo-random number generator. The two runs here are consecutive to those in Fig. 5.68.

It is interesting to observe the effect of $v_{\max}$ on the trajectory of a particle whose settings lead to faster convergence. Thus, for the same $aw = 4.00$ and $w = 0.50$, the trajectories of the particle without $v_{\max}$, and with $v_{\max}$ set to half and to a quarter of the feasible interval is offered in Fig. 5.71. The random weights are included in the right column, and replaced by their expected value in the left one. As can be observed, $v_{\max} = 0.50 \cdot fint$ is already more than enough a constraint to keep the particle within the region of interest. The better the convergence ability of the coefficients' settings the less necessary it is to restrict the velocity. In any case, constraining it to half the feasible interval appears harmless at worst.

For a more accurate visual analysis, six consecutive runs were performed for $w = 0.50$, $aw = 4.00$, random weights included, and $v_{\max}$ set to half the feasible interval on the one hand, and without the $v_{\max}$ constraint on the other. The trajectories are shown in Fig. 5.72 and Fig. 5.73, where consecutive rows stand for consecutive runs. The trajectories

on the left column do not have the velocity constrained, whereas the ones on the right column have it restricted as in Eq. (5.88). Figures in the same row correspond to the same initial state of the pseudo-random number generator. Clearly, local explosion is contained every time, without a quick loss of momentum as a result of the constraint.

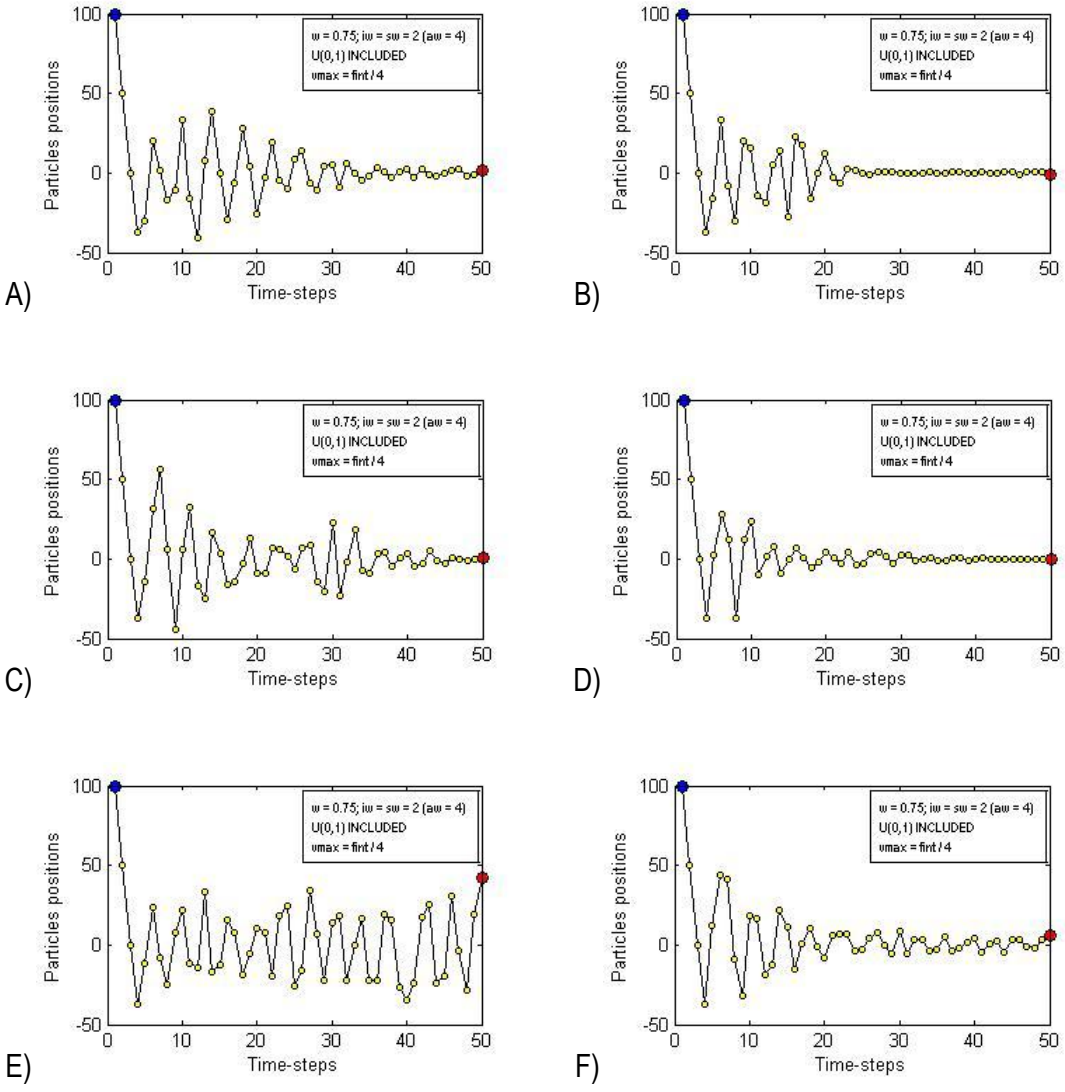


Fig. 5.70. Six possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.75, aw = 4.00, and random weights $U_{(0,1)}$ included. The maximum displacement permitted is restricted to a quarter of the feasible interval (i.e. $v_{max} = fint / 4$). The first one corresponds to the first run once the pseudo-random number generator is set to its initial state, while the others correspond to consecutive runs.

The effect of further constraining the displacement to a quarter of the feasible interval is shown in Fig. 5.74, where the trajectories corresponding to six consecutive runs are displayed. As it can be observed, there might be some quick loss of momentum, leaving an

important part of the search-space unexplored. Loosely speaking, this is an undesirable behaviour, especially for a global version of the optimizer. Nevertheless, it might still be considered for some local topologies or special applications (e.g. *swarm robots*).

Further decreasing the size of the maximum displacement permitted does not seem advisable. Numerical testing is required for better assessment of the influence of the size of v_{max} on full, multi-particle PSO systems, and different neighbourhood topologies.

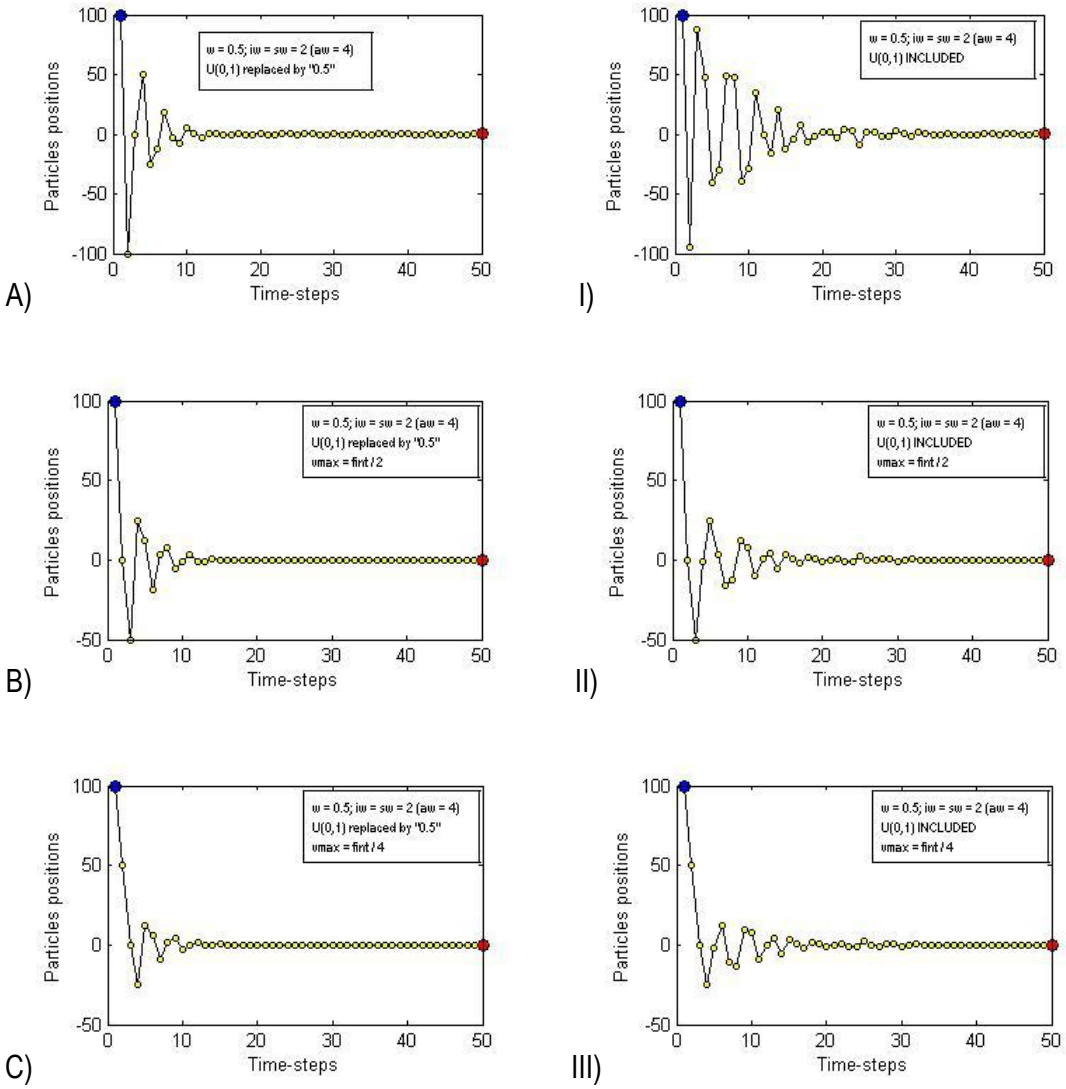


Fig. 5.71. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.50 and aw = 4.00. Different rows stand for different values of the v_{max} constraint. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value '0.50' in the first one.

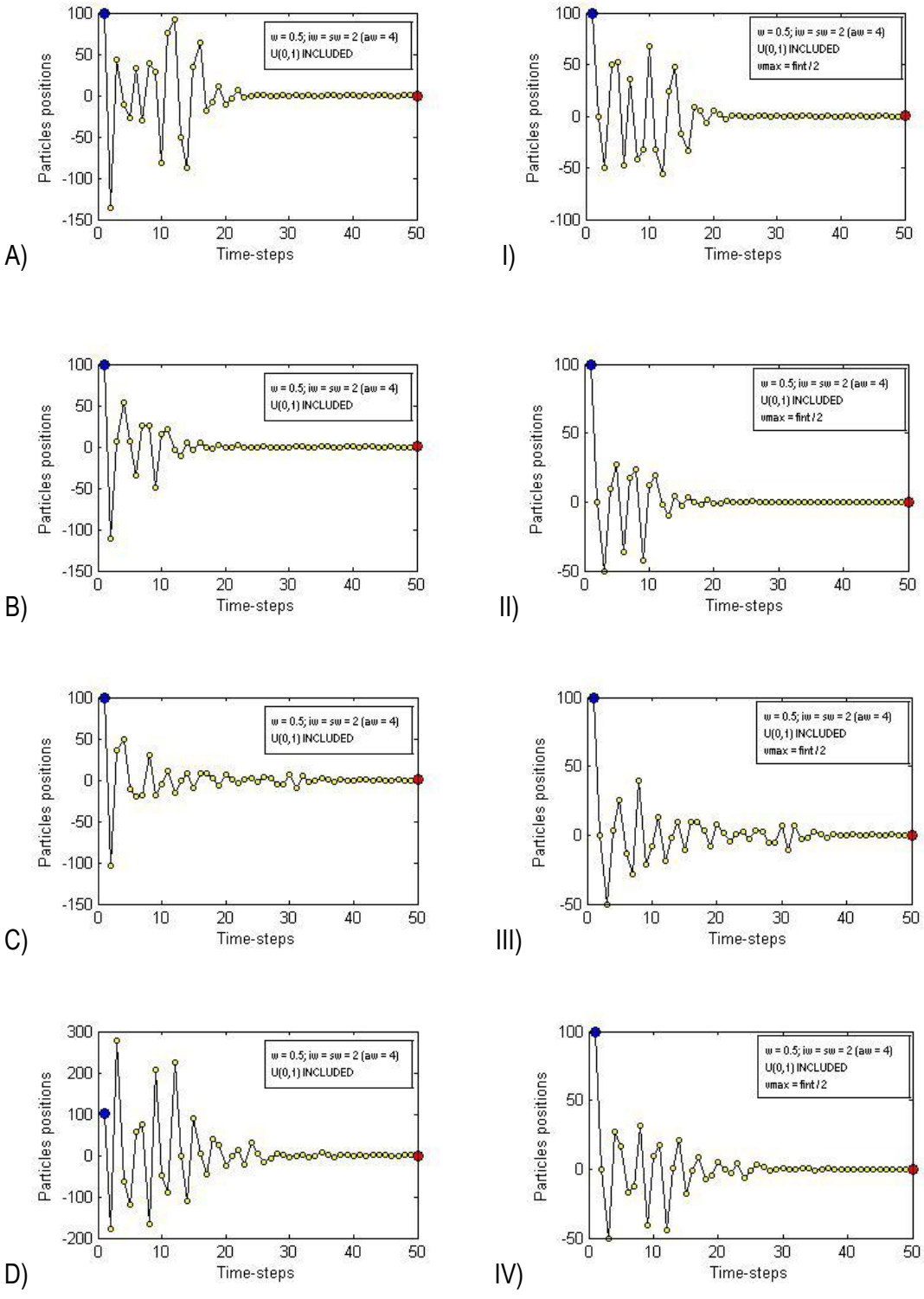


Fig. 5.72. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.50, aw = 4.00, and random weights $U_{(0,1)}$ included. The trajectories on the left column have no restriction to the velocity, whereas those on the right one have it constrained to half the feasible interval ($fint$). Consecutive rows correspond to consecutive runs, so that the figures in the same row have the same initial state of the pseudo-random number generator.

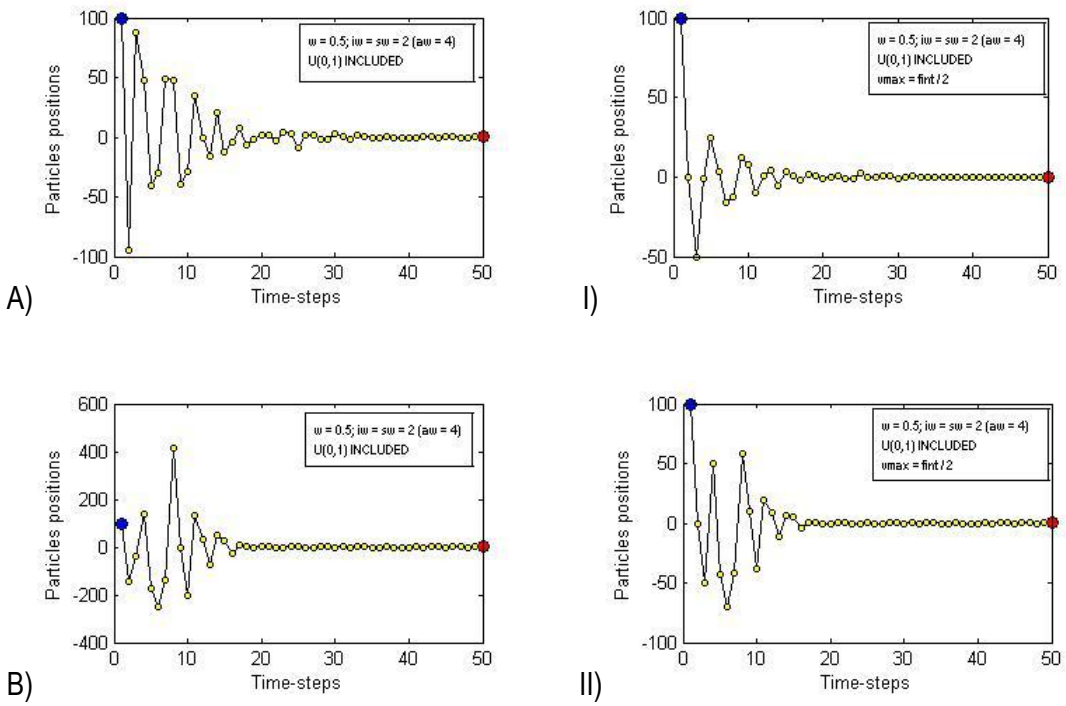


Fig. 5.73. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.50, aw = 4.00, and random weights $U_{(0,1)}$ included. The trajectories on the left column have no restriction to the velocity, whereas those on the right one have it constrained to half the feasible interval (*fint*). Consecutive rows correspond to consecutive runs, so that the figures in the same row have the same initial state of the pseudo-random number generator. The two runs here are consecutive to those in Fig. 5.72.

5.8.3.2. Acceleration weight *aw* = 3.00

In the cases with $aw = 4.00$, ϕ was only part of the time within the convergence region in Fig. 5.20 (87.5% of the time for $w = 0.75$ and 75% for $w = 0.50$). It is interesting to observe that local explosions are still possible even if ϕ is all the time within the convergence region. It is easy to see that for $aw > 2$, there is always a chance that a sequence of $\phi > 1$ is generated (i.e. random weights greater than '0.50'). This would temporarily send the particle farther from rather than closer to the attractor. This was illustrated in Fig. 5.9, and an example is offered in Fig. 5.75 for $w = 0.75$ and $aw = 3.00$. The trajectories of the particle without v_{max}, and with v_{max} set to half, and to a quarter of the feasible interval are shown. The random weights are included in the right column, and replaced by their expected value in the left one. A local explosion is shown in Fig. 5.75 I), which is controlled by the v_{max} constraints in Fig. 5.75 II) and Fig. 5.75 III). Recall that six runs were performed for every trajectory that involves randomness. It is fair

to note that the other five runs did not show any local explosion. Those images are not included here but can be found in the digital appendix.

With regards to the size of the constraint, again $v_{\max} = 0.50 \cdot fint$ restricts the explored search-space to approximately the feasible space, whereas $v_{\max} = 0.25 \cdot fint$ restricts it to approximately half the feasible space. Thus, the maximum peak-to-peak amplitude is approximately bounded by $2 \cdot v_{\max}$.

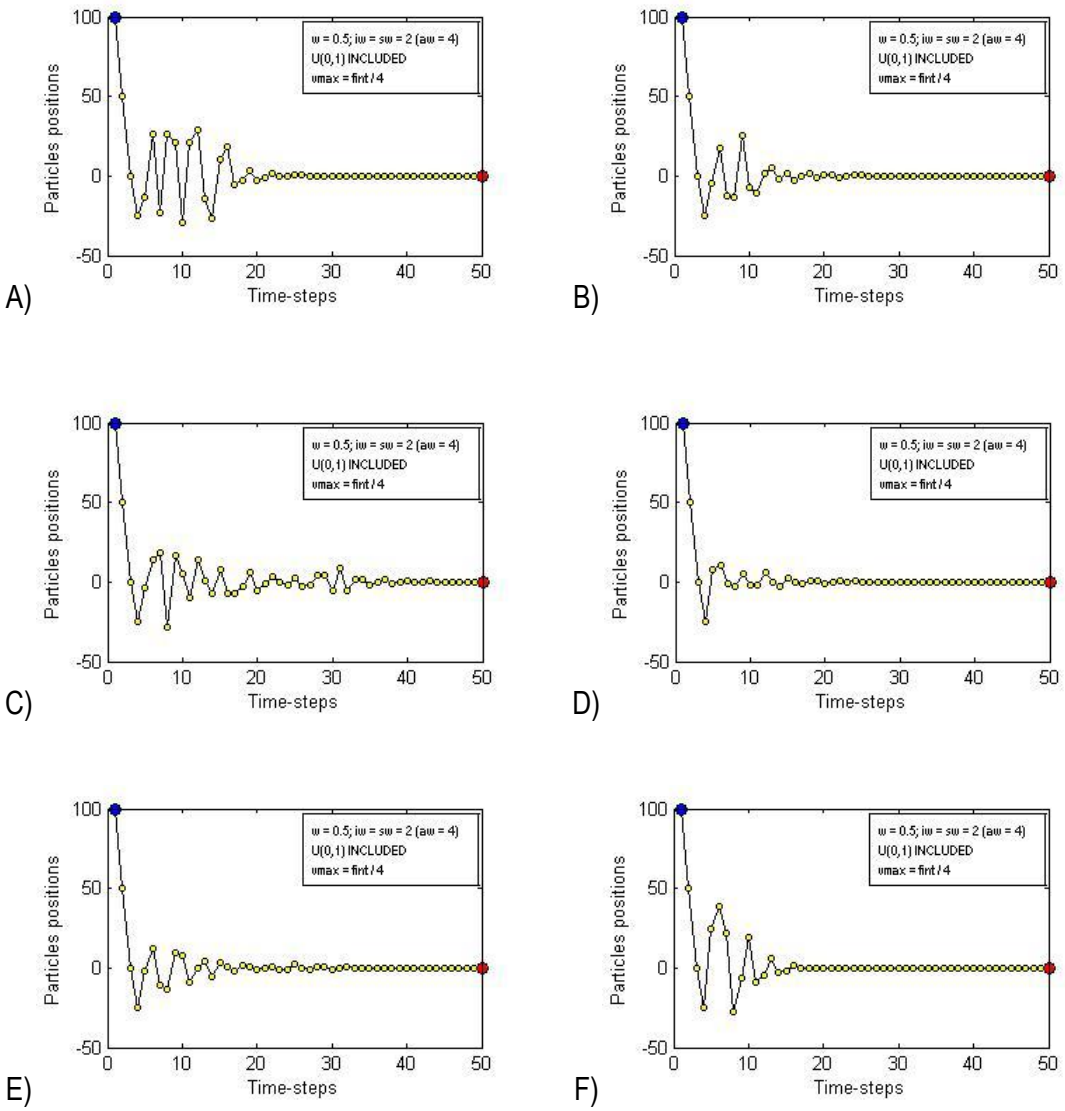


Fig. 5.74. Six possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.50, aw = 4.00, and random weights $U_{(0,1)}$ included. The maximum displacement permitted is restricted to a quarter of the feasible interval (i.e. $v_{\max}$ = $fint$ / 4). The first one corresponds to the first run once the pseudo-random number generator is set to its initial state, while the others correspond to consecutive runs.

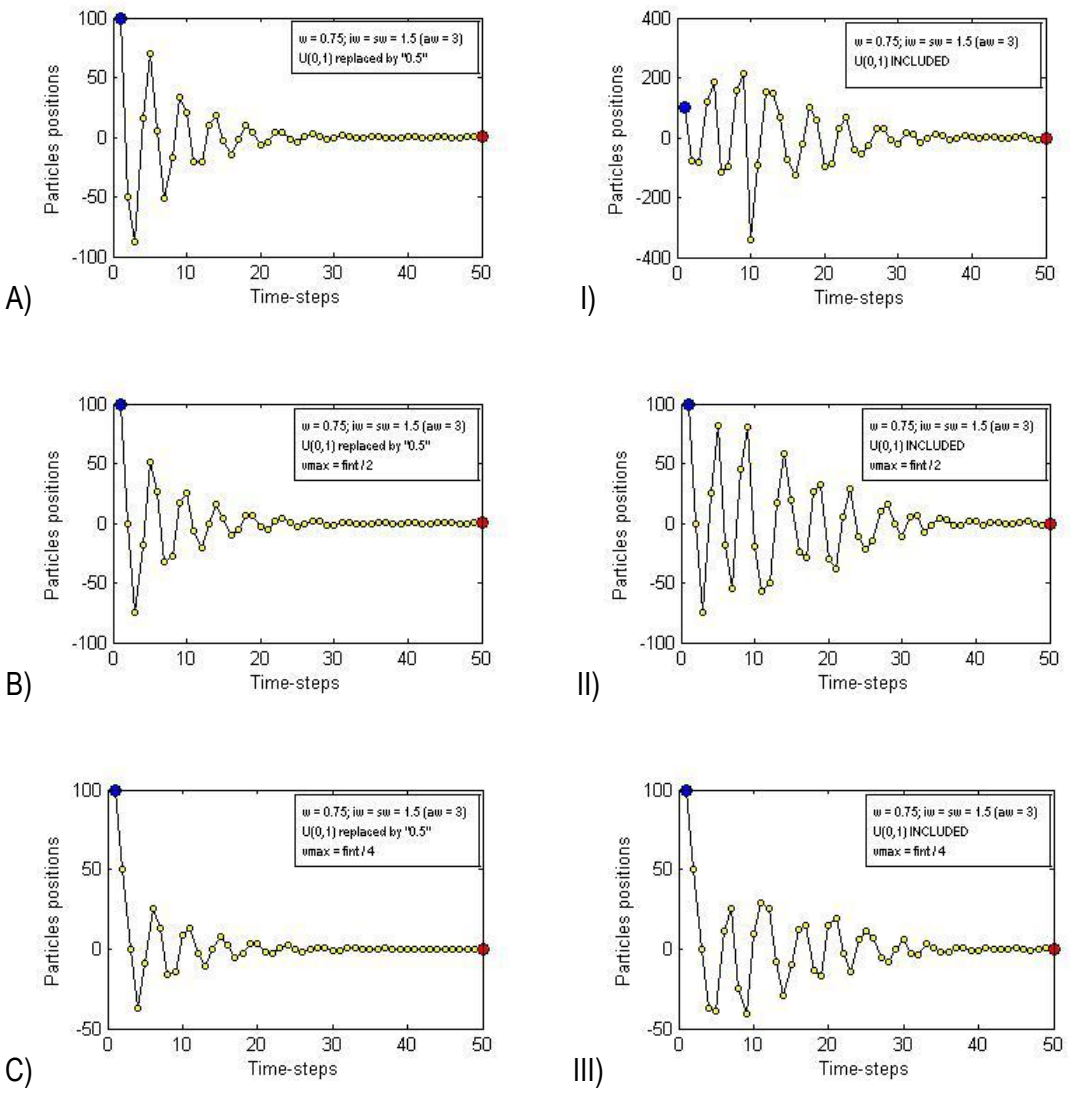


Fig. 5.75. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.75 and aw = 3.00. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected values in the first one. Different rows stand for different v_{max} constraints, the first one being unrestricted.

5.8.3.3. Acceleration weight *aw* = 2.00

For acceleration weights this small or smaller, it is unlikely that a particle is driven outside the feasible space. Six runs were performed for $w = 0.75$ and $aw = 2.00$, five of which showed the particle never leaving the feasible space. The one that barely did is shown in Fig. 5.76 I). Notice that for that to happen, the random weight $U_{(0,1)} \cong 1$ just when the particle was at one extreme of the feasible space (7th time-step) thus pulling it to almost the other extreme. In addition, the next $U_{(0,1)}$ was smaller in the 8th time-step,

which, combined with a high $w = 0.75$ took the particle outside the feasible space. As can be observed, that is not likely to occur very often, and when it does, the particle is not driven too far and it is pulled back by the following time-step. For the other six runs with and without v_{max} constraint, refer to the digital appendix.

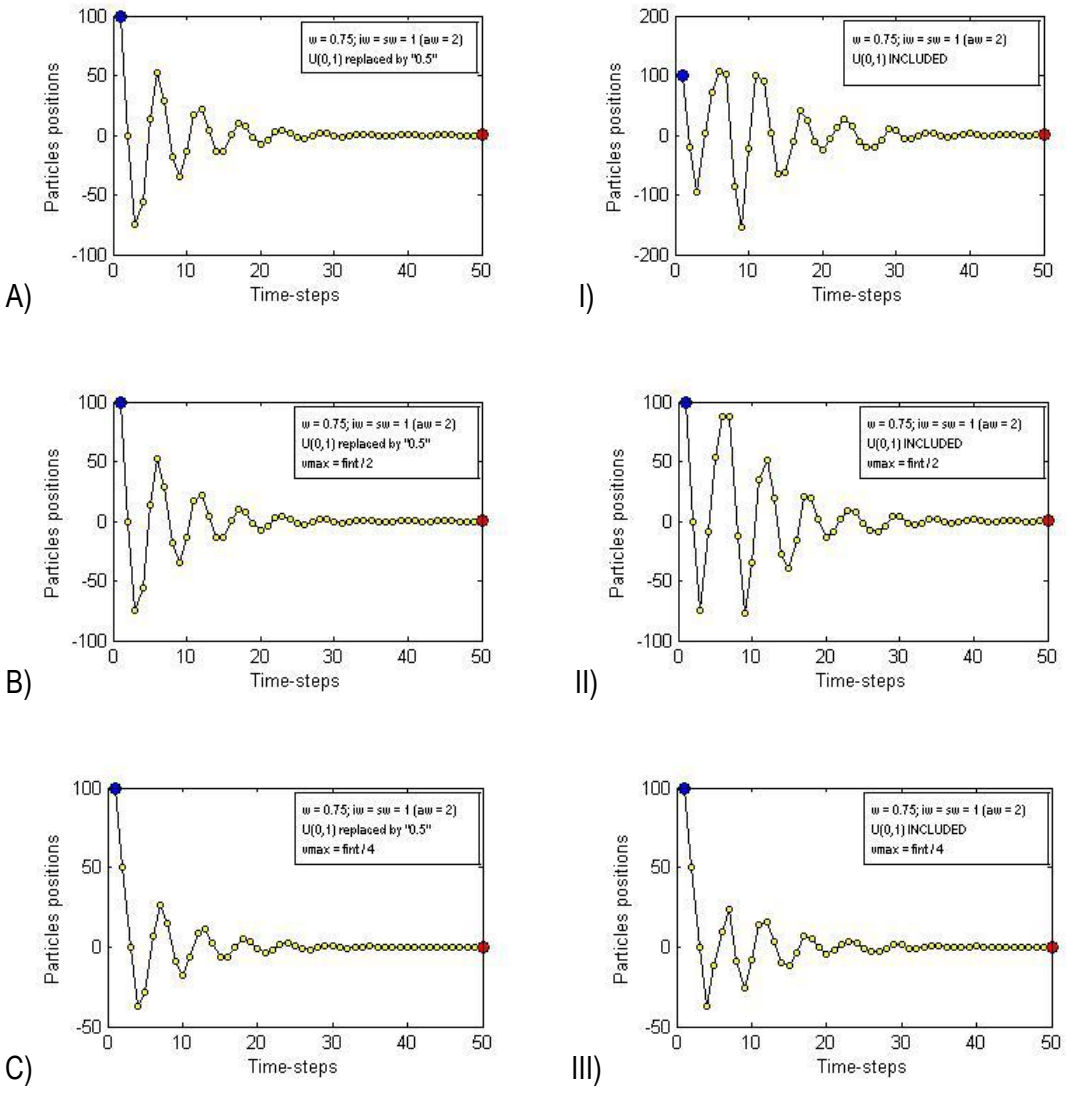


Fig. 5.76. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for w = 0.75 and aw = 2.00. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected values in the first one. Different rows stand for different v_{max} constraints, the first one being unrestricted.

It is self-evident that the v_{max} constraint should not be too restrictive for such small values of the acceleration weight. Of course, even less if the inertia weight is small as well, although it has been previously stated that keeping them both small is not advisable in the first place.

5.9. Constriction factor

Only the simplest constriction factor is analyzed here, which is referred to *Type 1"* by Clerc and Kennedy (2002). This is because it comprises a minor variation of the classical –and also of the original– PSO algorithm. In the original one, explosion was not controlled ($w=1$). Shi and Eberhart (1998a) simply included an inertia weight multiplying the first term in the original velocity update equation, keeping the two terms involving the attractors unchanged. This is the most widespread version today, and it is therefore referred to as 'classical' here. The *Type 1"* constriction factor proposed by Clerc and Kennedy (2002) –whose formulation is shown in Eqs. (5.89) and (5.90)– incorporates a coefficient multiplying all three terms in the original velocity update equation. Note that the constricted version can easily be reduced to the classical one.

$$cf = \begin{cases} \dfrac{2\cdot\kappa}{(iw+sw)-2+\sqrt{(iw+sw)^2-4\cdot(iw+sw)}} & \text{if } (iw+sw)\geq 4 \\ \kappa & \text{otherwise} \end{cases} \tag{5.89}$$

$$\begin{cases} v_{ij}^{(t)} = cf\cdot\left(v_{ij}^{(t-1)} + iw\cdot U_{(0,1)}\cdot\left(pbest_{ij}^{(t-1)} - x_{ij}^{(t-1)}\right) + sw\cdot U_{(0,1)}\cdot\left(lbest_{ij}^{(t-1)} - x_{ij}^{(t-1)}\right)\right) \\ x_{ij}^{(t)} = x_{ij}^{(t-1)} + v_{ij}^{(t)} \end{cases} \tag{5.90}$$

where $0<\kappa<1$. For the single particle with stationary attractors,

$$cf = \begin{cases} \dfrac{2\cdot\kappa}{aw-2+\sqrt{aw^2-4\cdot aw}} & \text{if } aw\geq 4 \\ \kappa & \text{otherwise} \end{cases} \tag{5.91}$$

$$\begin{cases} v^{(t)} = cf\cdot\left(v^{(t-1)} + \phi\cdot\left(p - x^{(t-1)}\right)\right) \\ x^{(t)} = x^{(t-1)} + v^{(t)} \end{cases} \tag{5.92}$$

While the constriction factor is typically referred to as 'χ', it is called '*cf*' in this thesis for ease of notation. Thus, '*cf*' refers to the *Type 1"* constriction factor only. Note that the random variable ϕ is replaced by $\phi_{\max} = aw = (iw + sw)$ in the calculation of the constriction factor in Eq. (5.89). The values of *cf* associated with the range $0.00 \leq aw \leq 8.00$ for three values of the constant κ are shown in Fig. 5.77.

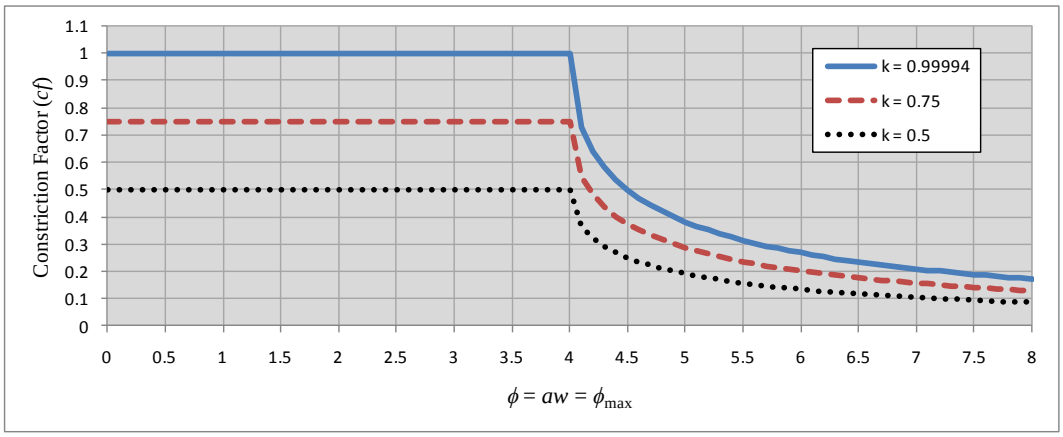


Fig. 5.77. Values of the constriction factor (*cf*) *type 1"* for a range of values of aw and three values of κ. If aw is set much greater than '4.00' for this type of constriction, there is a quick loss of momentum that leads to premature convergence. On the other hand, if $aw < 4.00$ and κ is too close to '1.00', convergence is extremely slow.

For the *type 1"* constriction, the matrix of the system is given by Eq. (5.93):

$$M = \begin{pmatrix} cf & cf \cdot \phi \\ -cf & (1 - cf \cdot \phi) \end{pmatrix} \tag{5.93}$$

Notice the similarity to the matrix of the system derived for the algorithm with the inertia weight in Eq. (5.57). If the value of ϕ is set so as to include *cf*, then the *cf* in the left column would be the inertia weight. Also compare Eqs. (5.90) and (5.2) to observe the minor differences between the two formulations. For the other types of constriction, refer to (Clerc & Kennedy, 2002).

It is important to note that following the extensive –and impressive– analyses in (Clerc & Kennedy, 2002) is not straightforward, and mistakes (or typos) can be found in the literature when using their constriction formulae. For instance, the *type 1"* constriction factor is said to be calculated as the square root of the expression in Eq. (5.91) in (van den Bergh, 2001, p. 63), (Engelbrecht, 2005, p. 147), and even in (Clerc & Kennedy, 2002, p. 70). It is also stated that κ could take on the value '1' in (Clerc & Kennedy, 2002, p. 70) and in (Kennedy & Eberhart, 2001, p. 339), when that would imply that there is no convergence for $\phi < 4$ but the (pseudo) cyclic behaviour discussed in section 5.3. Thus, according to Eq. (5.91), $cf = \kappa < 1$ for $aw < 4$. Translating these settings into the classical PSO formulation, $w = \kappa$ and the '*constricted aw*' $= \kappa \cdot aw$.

The pairs '*constricted aw–w*' for $\kappa \in (0,1)$ and for '*unconstricted aw*' = 4 (blue, dotted line) and '*unconstricted aw*' = 3 (red, dotted line) are shown in Fig. 5.78. Therefore, ϕ will vary between '0' and the corresponding dotted line. As can be observed, ϕ will be within the convergent region 100% of the time, and mostly within the complex region. Recall that the gray area represents the convergent region, and the sub-set within the parabola the region where the roots of the characteristic polynomial are complex. Also note that κ should not be set to a very small value to avoid premature convergence.

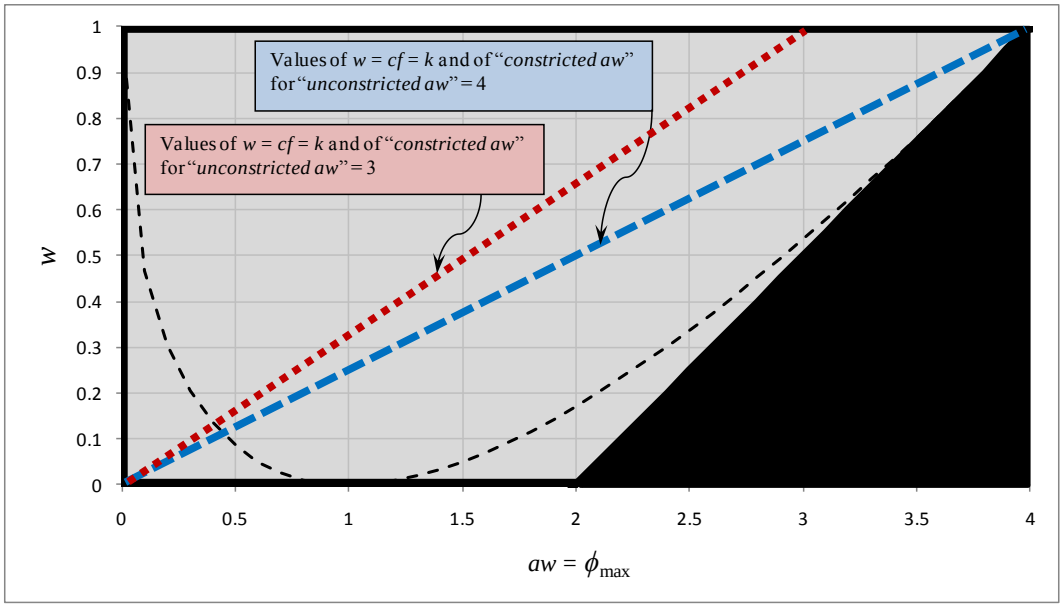


Fig. 5.78. Translation of the *type 1"* constriction factor into the classical PSO formulation for $aw \leq 4$. Thus, $w = cf = \kappa$ and the '*constricted aw*' = $\kappa \cdot$ '*unconstricted aw*'. The pairs of values '*constricted aw–w*' are shown for the whole range of κ for the particular cases of '*unconstricted aw*' = 4 (blue, dotted line) and '*unconstricted aw*' = 3 (red, dotted line). Thus, ϕ varies from '0' to the corresponding dotted line, completely within the convergence region (gray area in the graph), and mostly within the complex region (inside the parabola).

For '*unconstricted aw*' > 4, the constriction is too strong, as shown in Fig. 5.77. Hence *aw* only marginally greater than '4' is advisable in practice for this constriction type.

5.9.1. Constriction factor *cf* = 0.7298 (aw = 4.10, κ = 0.99994)

A very popular setting for the *type 1"* constriction factor is given by $\kappa = 0.99994$ and $iw = sw = 2.05$ ($aw = 4.10$) resulting in $cf = 0.7298$. Translating these settings into the classical PSO yields $w = 0.7298$ and $iw = sw = 1.49609$ ($aw = 2.99218$). That is to say, the '*constricted aw*' = 2.99218, while the '*unconstricted aw*' = 4.10.

Trajectories of the particle with stationary attractors are shown in Fig. 5.79 for these settings with the random weights removed in figure A); replaced by their expected value '0.5' in figure B); and included in the remaining six figures. The upper text-boxes show the settings in terms of the constriction factor, while the lower text-boxes show the translation into the classical formulation with inertia weight. Fig. 5.79 A) shows the trajectory of the deterministic particle for $\phi = \phi_{\max}$, whereas Fig. 5.79 B) shows the trajectory for $\phi = \phi_{\text{mean}}$. As expected, the trajectories converge. Note the similarity of the trajectories with random weights included in Fig. 5.79 C) to H) to those in Fig. 5.60.

5.9.2. Constriction factor *cf* = 0.547383 (aw = 4.10, κ = 0.75)

Setting $\kappa = 0.75$ and $aw = 4.10$ results in $cf = 0.547383$. Translating these settings into the classical PSO yields $w = 0.547383$ and $aw = 2.24427$. That is to say, the '*constricted aw*' = 2.24427, while the '*unconstricted aw*' = 4.10.

Trajectories of the particle with stationary attractors are shown in Fig. 5.80 for these settings with the random weights removed in figure A); replaced by their expected value '0.5' in figure B); and included in the remaining six figures. The upper text-boxes show the settings in terms of the constriction factor, while the lower text-boxes show the translation into the classical formulation with inertia weight. Fig. 5.80 A) shows the trajectory of the deterministic particle for $\phi = \phi_{\max}$, whereas Fig. 5.80 B) shows the trajectory for $\phi = \phi_{\text{mean}}$. As expected, the trajectories converge much faster than in Fig. 5.79.

5.9.3. Constriction factor *cf* = 0.364922 (aw = 4.10, κ = 0.50)

Setting $\kappa = 0.50$ and $aw = 4.10$ results in $cf = 0.364922$. Translating these settings into the classical PSO yields $w = 0.364922$ and $aw = 1.49618$.

Trajectories of the particle with stationary attractors are shown in Fig. 5.81 for these settings with the random weights removed in figure A); replaced by their expected value in figure B); and included in the remaining figures. The loss of momentum is too fast for these settings, risking premature convergence. The particle does not exhibit explorative capabilities. Clearly, such small values of κ are not recommendable.

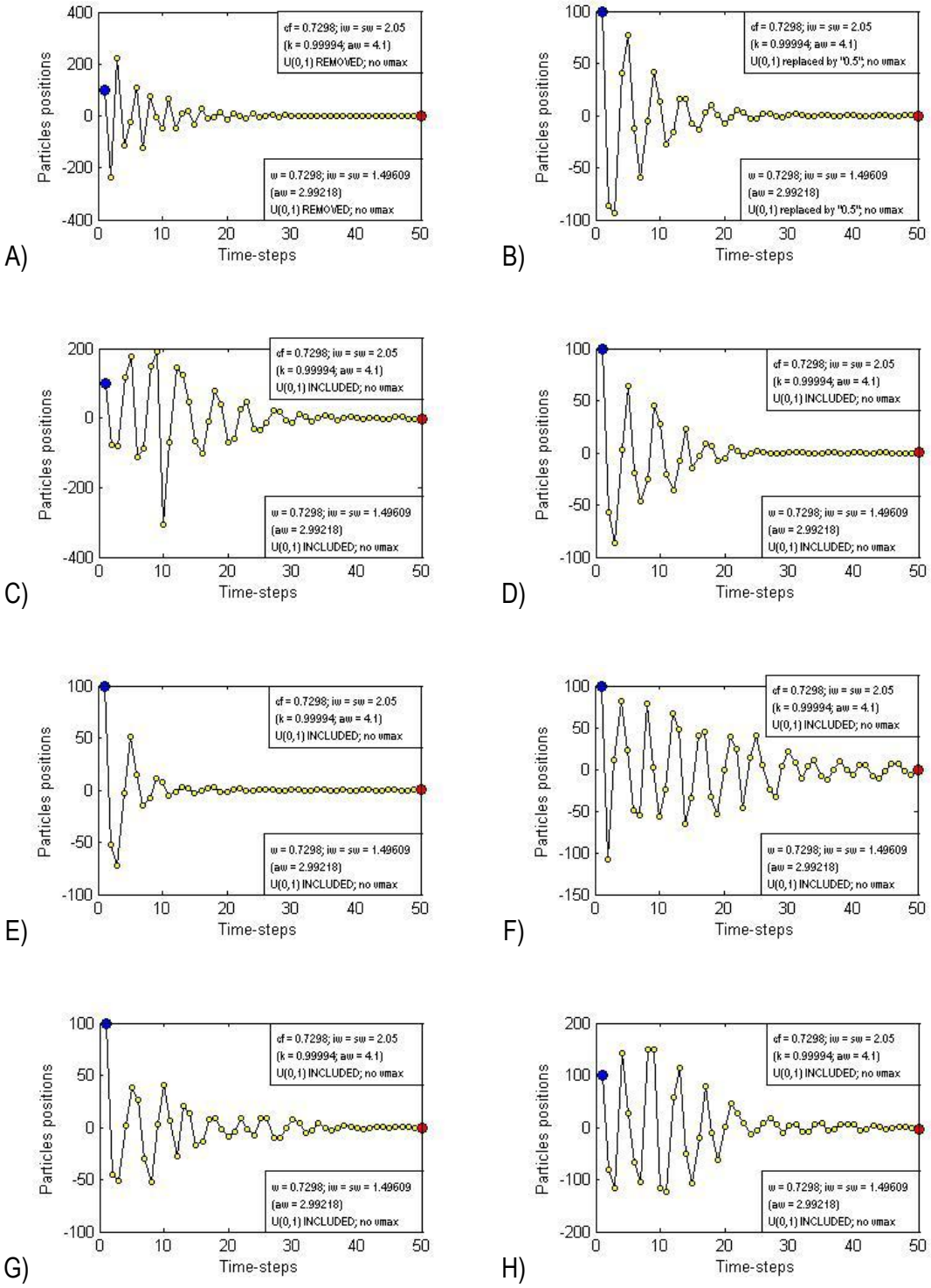


Fig. 5.79. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for cf = 0.7298 and aw = 4.10. Random weights $U_{(0,1)}$ are removed in A); replaced by their expected value in B); and included in the remaining figures. The upper text box in each figure offers the settings for the constriction factor, and the lower one the equivalence in terms of the inertia weight formulation.

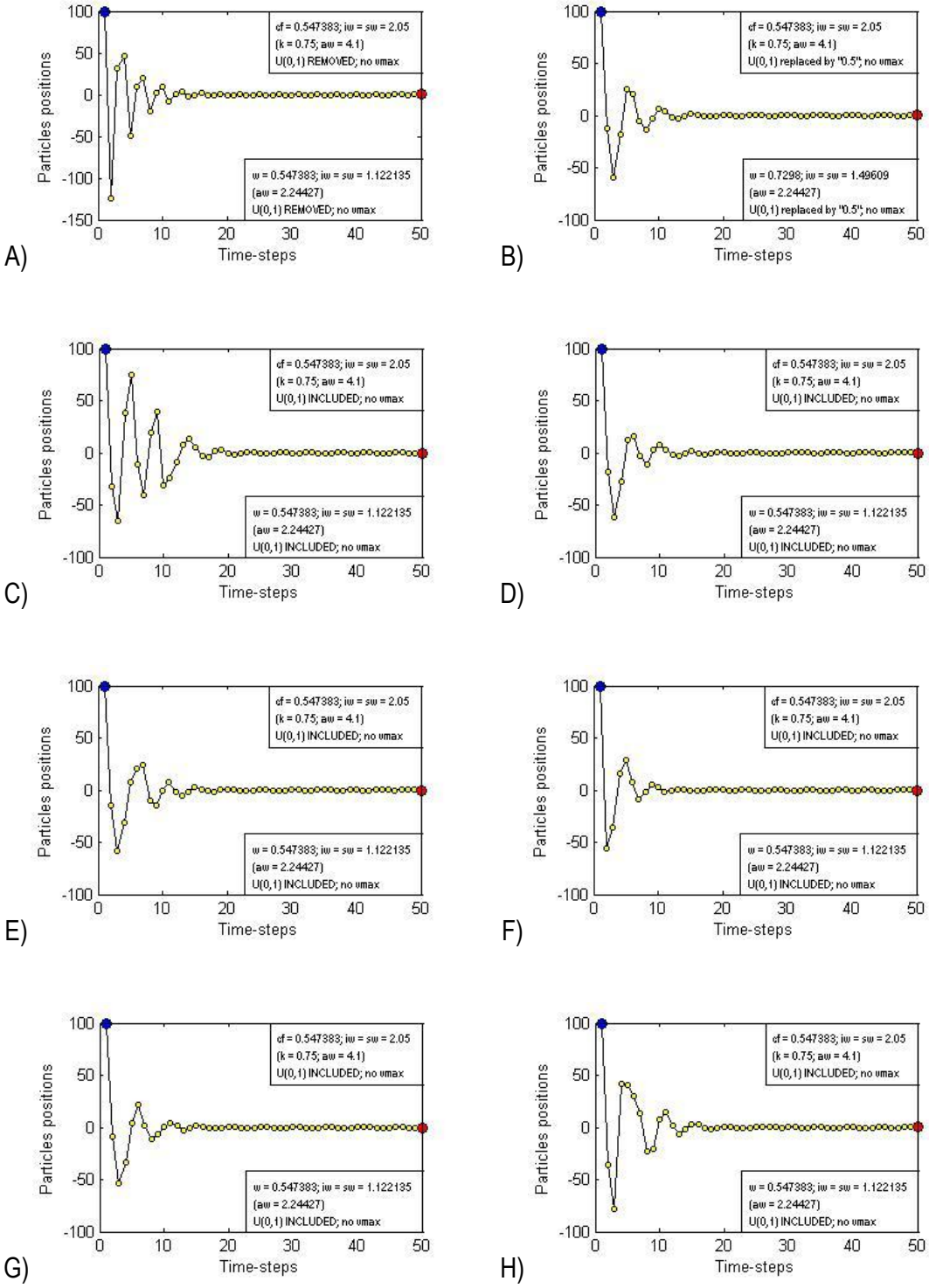


Fig. 5.80. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for cf = 0.547383 and aw = 4.10. Random weights $U_{(0,1)}$ are removed in A); replaced by their expected value in B); and included in the remaining figures. The upper text box in each figure offers the settings for the constriction factor, and the lower one the equivalence in terms of the inertia weight formulation.

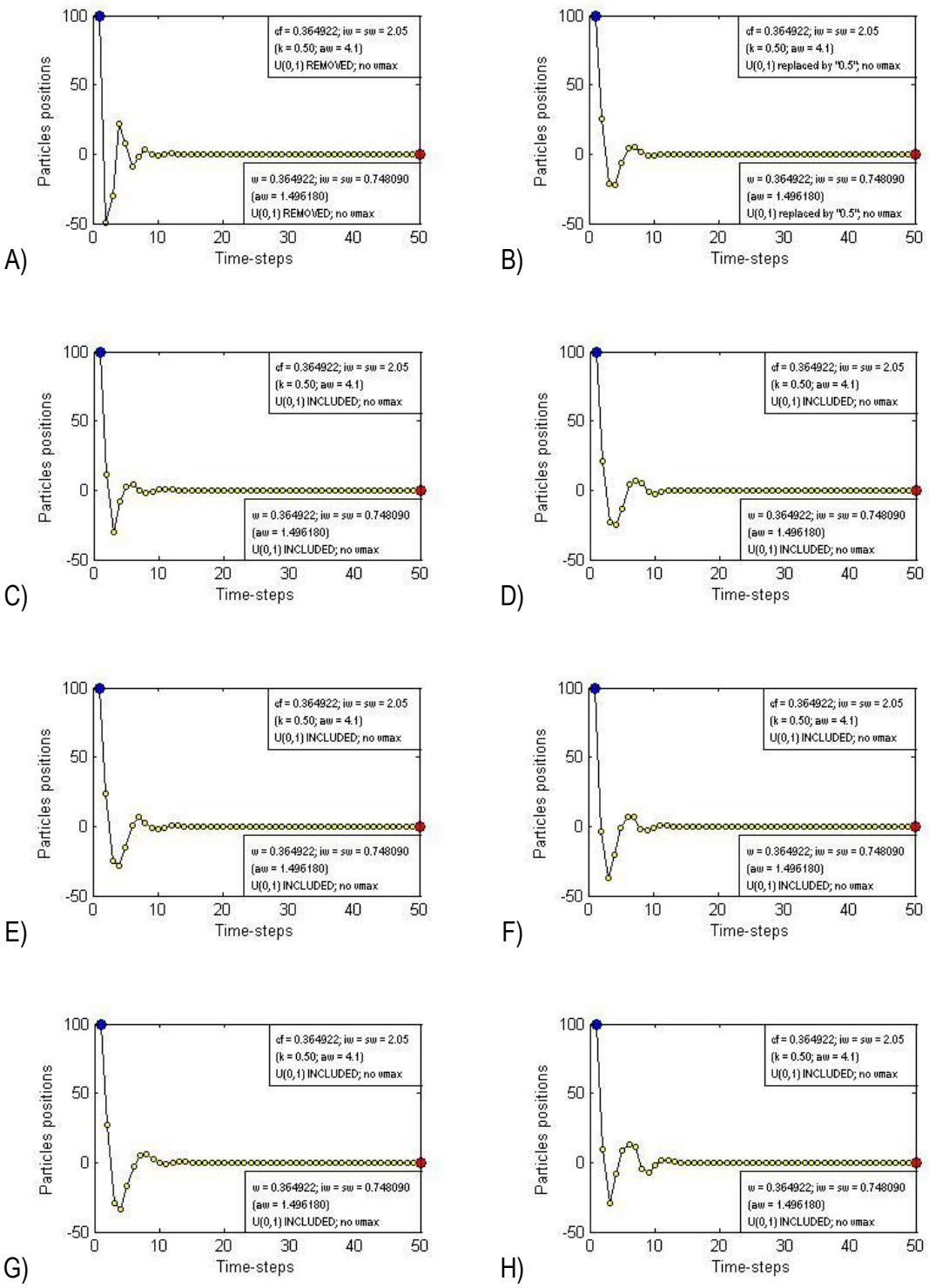


Fig. 5.81. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for cf = 0.364922 and aw = 4.10. Random weights $U_{(0,1)}$ are removed in A); replaced by their expected value in B); and included in the remaining figures. The upper text box in each figure offers the settings for the constriction factor, and the lower one the equivalence in terms of the inertia weight formulation.

5.9.4. Constriction factor plus velocity constraint

As can be observed in Fig. 5.79, the particle may occasionally be driven outside the feasible space for a few time-steps for high values of κ (and *aw* not too much greater than '4'). Therefore, as argued in section 5.8.3, the incorporation of a $v_{\max}$ constraint that is not too restrictive is still convenient.

For the same settings used in section 5.9.1, namely $\kappa = 0.99994$ and $aw = 4.10$ resulting in $cf = 0.7298$, the trajectories of the particle with stationary attractors are shown in Fig. 5.82 with the random weights removed in figure A); replaced by their expected value in figure B); and included in the remaining six figures. In every case, the maximum displacement is restricted to half the feasible interval: $v_{\max} = 0.50 \cdot fint$. The upper text-boxes show the settings in terms of the constriction factor, while the lower text-boxes show the translation into the classical formulation. Fig. 5.82 A) shows the trajectory of the deterministic particle for $\phi = \phi_{\max}$, whereas Fig. 5.82 B) shows the trajectory for $\phi = \phi_{\text{mean}}$.

Comparing Fig. 5.82 to Fig. 5.79, it can be observed that the few positions of the particle outside the feasible space were prevented in every case by setting $v_{\max} = 0.50 \cdot fint$.

Recall that translating these settings into the classical PSO yields $w = 0.7298$ and $iw = sw = 1.49609$ ($aw = 2.99218$). That is to say, the '*constricted aw*' = 2.99218, while the '*unconstricted aw*' = 4.10.

5.10. Correlated coefficients

In the original PSO (*w* = 1 and *aw* = 4), the average trajectory of the particle should be cyclic, with *period* = 4, as shown in Fig. 5.4 F). It has already been discussed that the incorporation of the random weight leads to a stochastic explosion. However, if the explosion is controlled by $v_{\max} = 0.50 \cdot fint$, the particle is still rather reluctant to converge because its behaviour tends to be cyclic. This average cyclic behaviour is illustrated in vector form on a 2-dimensional space in Fig. 5.83, where the four positions of the full cycle are shown. The small red dot between **gbest** and **pbest** is the weighted average stationary attractor **p**.

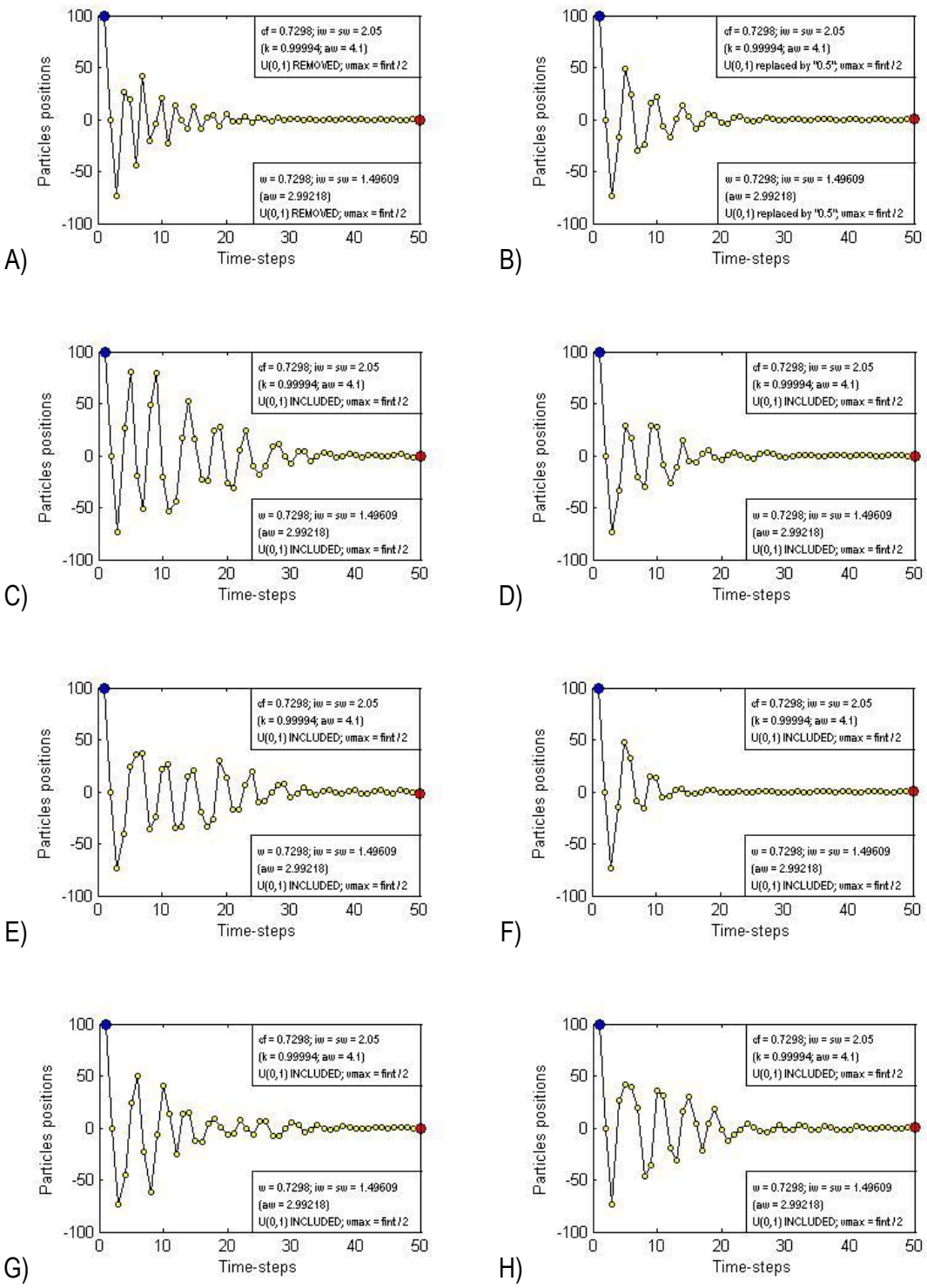


Fig. 5.82. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for cf = 0.7298; aw = 4.10; and $v_{\max}$ equal to half the feasible interval ($fint$ / 2). Random weights $U_{(0,1)}$ are removed in A); replaced by their expected value in B); and included in the remaining figures. The upper text boxes offer the settings for the constriction factor, and the lower ones the equivalence in terms of the inertia weight formulation.

While it was initially assumed that the smaller the inertia weight the faster the convergence, $w = 0$ would again lead to average cyclic behaviour with *period* = 2 (see Fig. 5.28 H)). The average cyclic behaviour for $w = 0$ and $aw = 4$ is illustrated in Fig. 5.84, where the two positions of the cycle are shown.

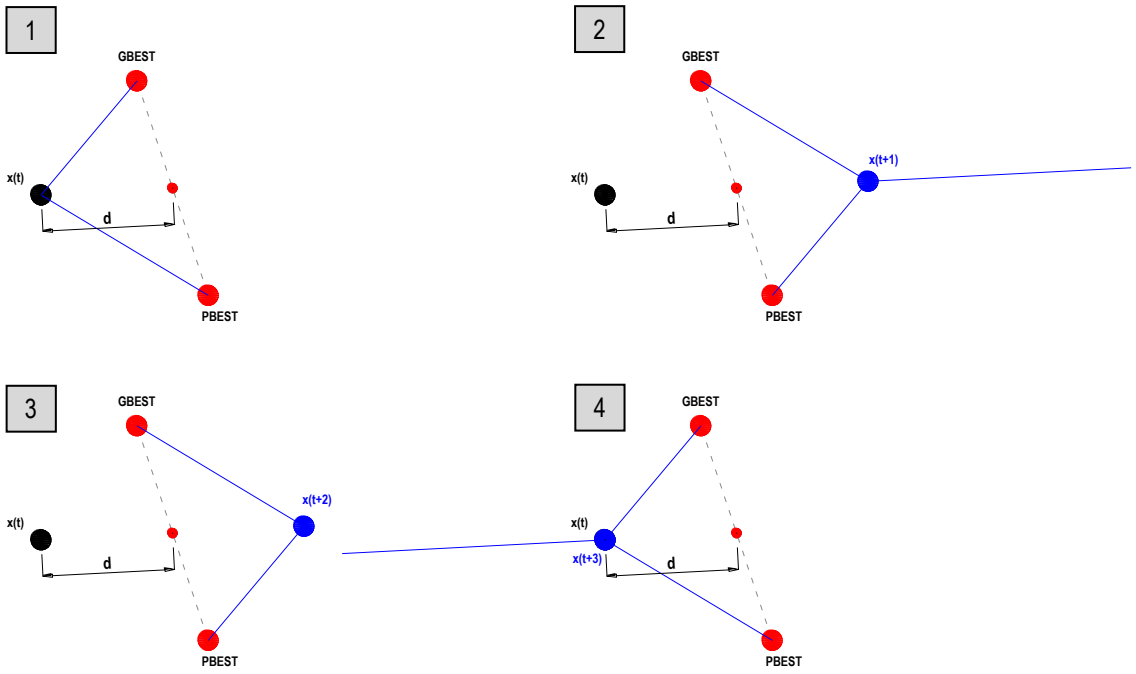


Fig. 5.83. Four positions of the whole cycle of the particle for the settings of the original PSO (w = 1 and aw = 4.00), with the random weights replaced by their expected value '0.50'. Initial velocity is set to '0', and 'd' is the distance from the position at time-step t to the weighted average of the two attractors.

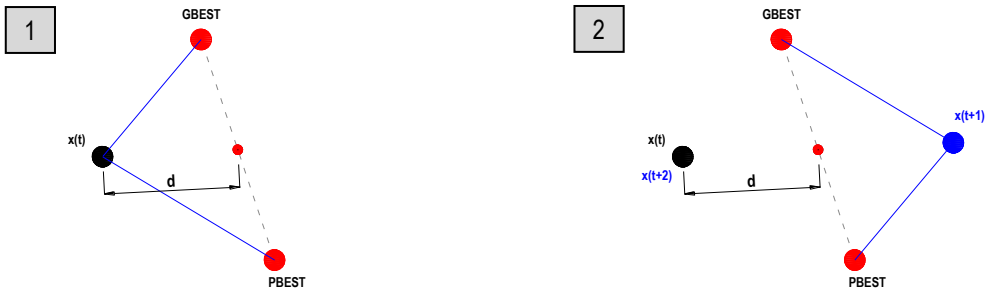


Fig. 5.84. Two positions of the cycle of the particle for w = 0 and aw = 4.00, with the random weights replaced by their expected value '0.50'. Initial velocity is set to '0', and 'd' is the distance from the position at time-step t to the weighted average of the two attractors.

It is self-evident, though, that $w=1$ would make convergence more difficult than $w=0$ once the random weights are incorporated.

Aiming to keep the original $aw=4.00$ so as to maintain the tendency of the particle to continuously overfly the attractor, but at the same time improve its ability to converge towards it, Innocente (2006) performed a rudimentary geometrical analysis of the trajectory with $aw=4.00$ similar to the one in Fig. 5.83, but looking for an inertia weight that would drive the particle to the attractor in two time-steps. The analysis resulted in $w=0.50$ (for $aw=4.00$). The trajectory of the particle is illustrated in vector form on a 2-dimensional space in Fig. 5.85, where the attractor is reached in two time-steps. Six positions of the particle are depicted, which still has some momentum left.

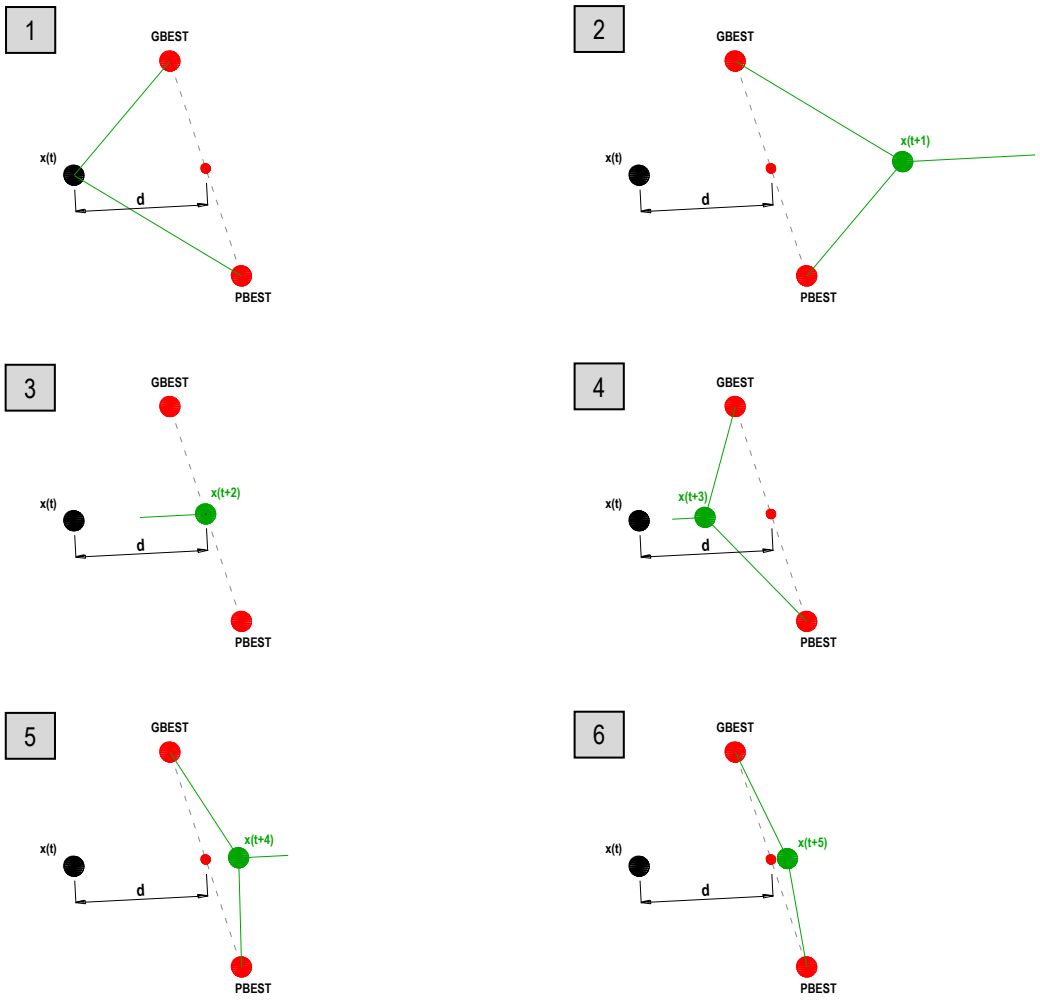


Fig. 5.85. Six consecutive positions of the particle for w = 0.50 and aw = 4.00, with the random weights replaced by their expected value '0.50'. Initial velocity is set to '0' and 'd' is the distance from the position at time-step t to the weighted average of the two attractors. The latter is reached in two time-steps.

Even for settings that result in convergent trajectories, the greater the inertia and the acceleration weights the slower the convergence. Thus, aiming to find different settings that would lead to convergence but at different speeds, similar geometrical analyses were performed in (Innocente, 2006) for different values of *aw*. Some of them are illustrated in Fig. 5.86 to Fig. 5.88, where the attractor is again reached in two time-steps.

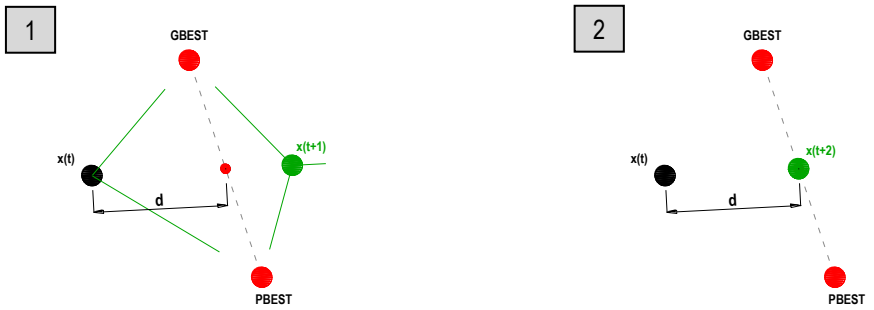


Fig. 5.86. Three consecutive positions of the particle for w = 1/6 and aw = 3.00, with the random weights replaced by '0.50'. Initial velocity is set to '0' and 'd' is the distance from the position at time-step t to the weighted average of the two attractors. The latter is reached in two time-steps. Note that the first two positions are in the figure on the left.

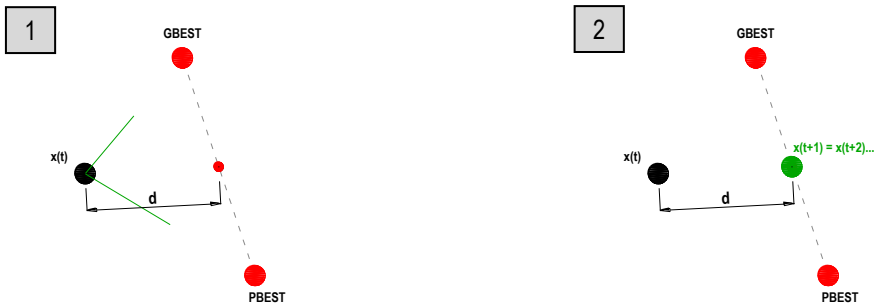


Fig. 5.87. Three consecutive positions of the particle for w = 0 and aw = 2.00, with the random weights replaced by '0.50'. Initial velocity is set to '0' and 'd' is the distance from the position at time-step t to the weighted average of the two attractors. The latter is reached in one time-step. Successive positions coincide because there is no inertia.

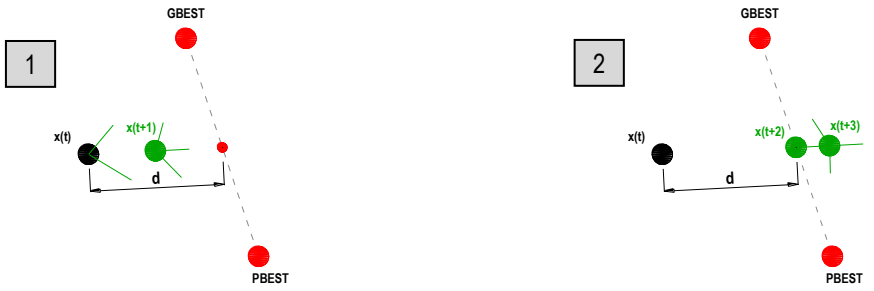


Fig. 5.88. Four consecutive positions of the particle for w = 0.50 and aw = 1.00, with the random weights replaced by '0.50'. Initial velocity is set to '0' and 'd' is the distance from the position at time-step t to the weighted average of the two attractors. The latter is reached in two time-steps. Note that there are two positions per figure.

Innocente (2006) carried out five geometrical analyses (all for $aw \geq 2$), and obtained five pairs '*aw*–*w*'. By direct interpolation, a (fourth degree) polynomial relationship between them was set, which was expected to favour convergence. Although the analyses were not mathematically rigorous, and nothing was said about convergence per se, the approach ensured that the settings of the inertia and acceleration weights were balanced. In fact, for inertia weights $w < 1$ and assuming –like in the geometrical analyses above– that $aw = \phi_{\max} = 2 \cdot \phi_{\text{mean}}$, the average behaviour of the pairs '*aw*–*w*' obtained are within the convergent region in Fig. 5.20.

The analysis is generalized hereafter so as to obtain an analytical relationship rather than a polynomial interpolation of a discrete number of geometrical analyses. Recall that the aim is to reach the attractor **p** –weighted average of **pbest** and **gbest**– in two time-steps, where the distance from $\mathbf{x}^{(t)}$ to **p** is called **d**. That is to say,

$$\begin{cases} \mathbf{p} - \mathbf{x}^{(t)} = \mathbf{d} \\ \mathbf{x}^{(t+2)} - \mathbf{x}^{(t)} = \mathbf{d} \end{cases} \tag{5.94}$$

Given that the initial velocity (i.e. at time-step *t*) is set to zero,

$$\begin{aligned} \mathbf{x}^{(t+1)} &= \mathbf{x}^{(t)} + \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)} \right) \\ \mathbf{x}^{(t+1)} &= \mathbf{x}^{(t)} + \phi \cdot \mathbf{d} \end{aligned} \tag{5.95}$$

Thus, the position of the particle at the next time-step (*t*+2) is as shown in Eq. (5.96):

$$\mathbf{x}^{(t+2)} = \mathbf{x}^{(t+1)} + w \cdot \left(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)} \right) + \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t+1)} \right) \tag{5.96}$$

From Eqs. (5.95) and (5.96),

$$\begin{aligned} \mathbf{x}^{(t+2)} &= \mathbf{x}^{(t)} + \phi \cdot \mathbf{d} + w \cdot \left(\mathbf{x}^{(t)} + \phi \cdot \mathbf{d} - \mathbf{x}^{(t)} \right) + \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)} - \phi \cdot \mathbf{d} \right) \\ \mathbf{x}^{(t+2)} &= \mathbf{x}^{(t)} + \phi \cdot \mathbf{d} + w \cdot \phi \cdot \mathbf{d} + \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)} - \phi \cdot \mathbf{d} \right) \end{aligned} \tag{5.97}$$

Therefore:

$$\mathbf{x}^{(t+2)} - \mathbf{x}^{(t)} = \phi \cdot \mathbf{d} + w \cdot \phi \cdot \mathbf{d} + \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)} - \phi \cdot \mathbf{d} \right) \tag{5.98}$$

From Eqs. (5.98) and (5.94),

$$\begin{aligned}\mathbf{d} &= \phi\cdot\mathbf{d} + w\cdot\phi\cdot\mathbf{d} + \phi\cdot(\mathbf{d} - \phi\cdot\mathbf{d}) \\ 1 &= \phi + w\cdot\phi + \phi\cdot(1-\phi)\end{aligned} \tag{5.99}$$

Hence,

$$\begin{aligned}\phi + w\cdot\phi + \phi - \phi^2 &= 1 \\ 2 + w - \phi &= \frac{1}{\phi}\end{aligned} \tag{5.100}$$

Finally,

$$\boxed{w = \frac{1}{\phi} - 2 + \phi} \tag{5.101}$$

Eq. (5.101) stands for the desired behaviour for a given ϕ. However $0 \le \phi \le aw$. Therefore, for this to be the average behaviour, ϕ is in fact ϕ_{mean} with $aw = 2\cdot\phi_{\text{mean}}$. Thus,

$$\boxed{w = \frac{2}{aw} - 2 + \frac{aw}{2}} \tag{5.102}$$

Trajectories of the particle with stationary attractors, random weights replaced by their expected value, no $v_{\max}$, aw varying from '5' to '1', and w as in Eq. (5.102) are shown in Fig. 5.89 and Fig. 5.90. Notice that even the trajectories involving $aw > 4$ end up converging. It can be observed that w decreases as aw decreases up until $aw = 2$. From then on, w increases again because aw is not big enough to reach the attractor on its own in two time-steps (see Fig. 5.88).

If Eq. (5.101) is not thought of as the average behaviour but the one corresponding to $\phi = \phi_{\max} = aw$, Eq. (5.103) is obtained instead of Eq. (5.102). That is,

$$\boxed{w = \frac{1}{aw} - 2 + aw} \tag{5.103}$$

This means that the behaviour imposed in Eq. (5.94) is not the average behaviour but the one corresponding to $\phi_{\max}$. Therefore the average behaviour would exhibit oscillations with lower frequencies.

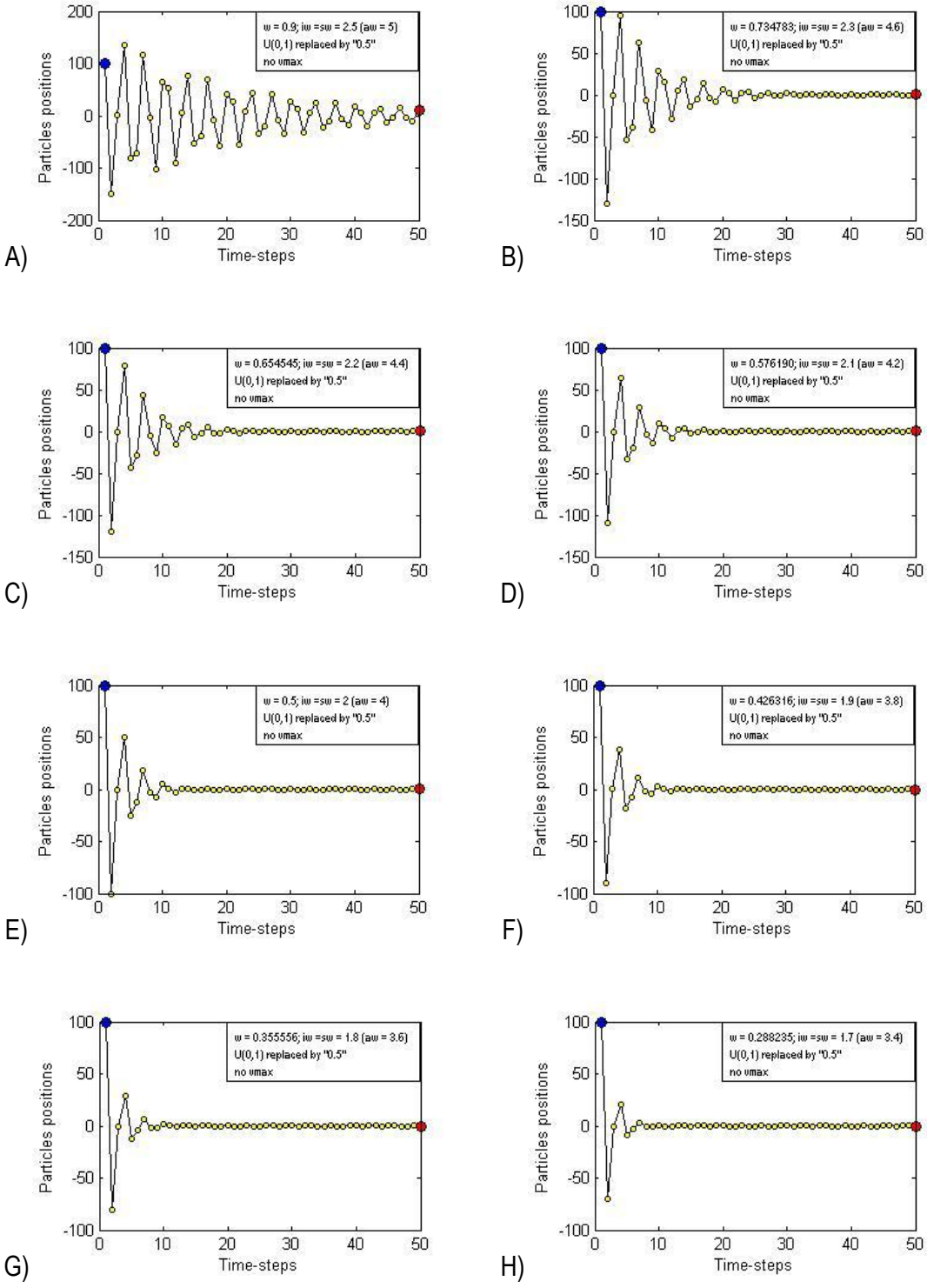


Fig. 5.89. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 5.00 ≥ aw ≥ 3.40, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are replaced by their expected value.

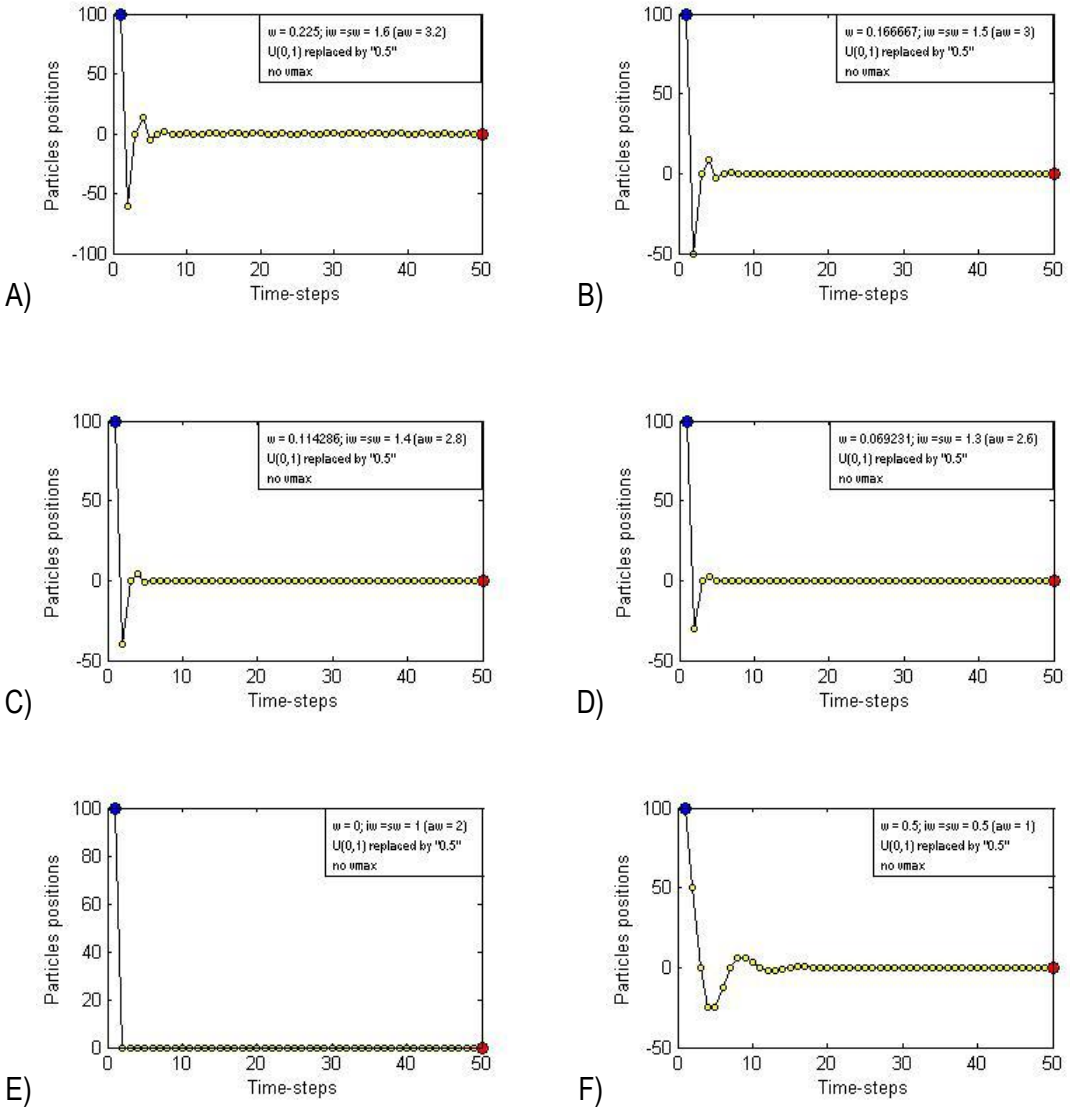


Fig. 5.90. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for $3.20 \geq aw \geq 1.00$, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are replaced by their expected value.

Fig. 5.89 and Fig. 5.90 show how the deterministic particle converges for the whole range of aw tested. Note that this is true even for $aw > 4$ ($\phi_{mean} > 2$), in which cases aw drives the particle farther from the attractor and it is w which brings it back. Oscillations are damped too quickly for small values of aw (e.g. $aw \leq 3$). In order to observe the influence of randomness, trajectories of the particle with stationary attractors, no v_{max}, $4.8 \geq aw \geq 3$, and w as in Eq. (5.102) are shown in Fig. 5.91 to Fig. 5.93. The random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one. The other five trajectories corresponding to the experiments performed for each of the cases involving random weights can be found in the digital appendix.

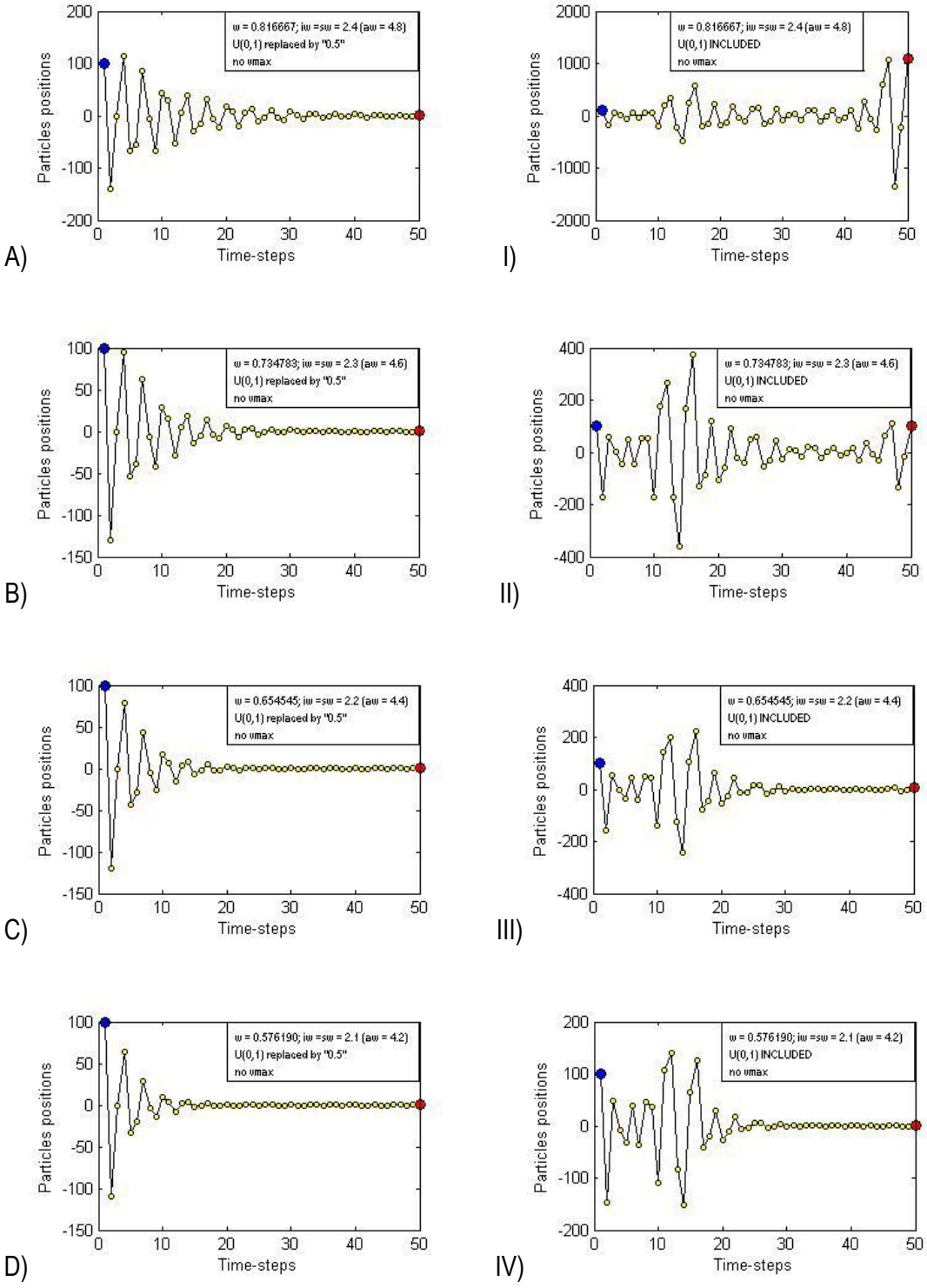


Fig. 5.91. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 4.80 ≥ aw ≥ 4.20, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one. Other five trajectories for each of the cases with $U_{(0,1)}$ included can be found in the digital appendix.

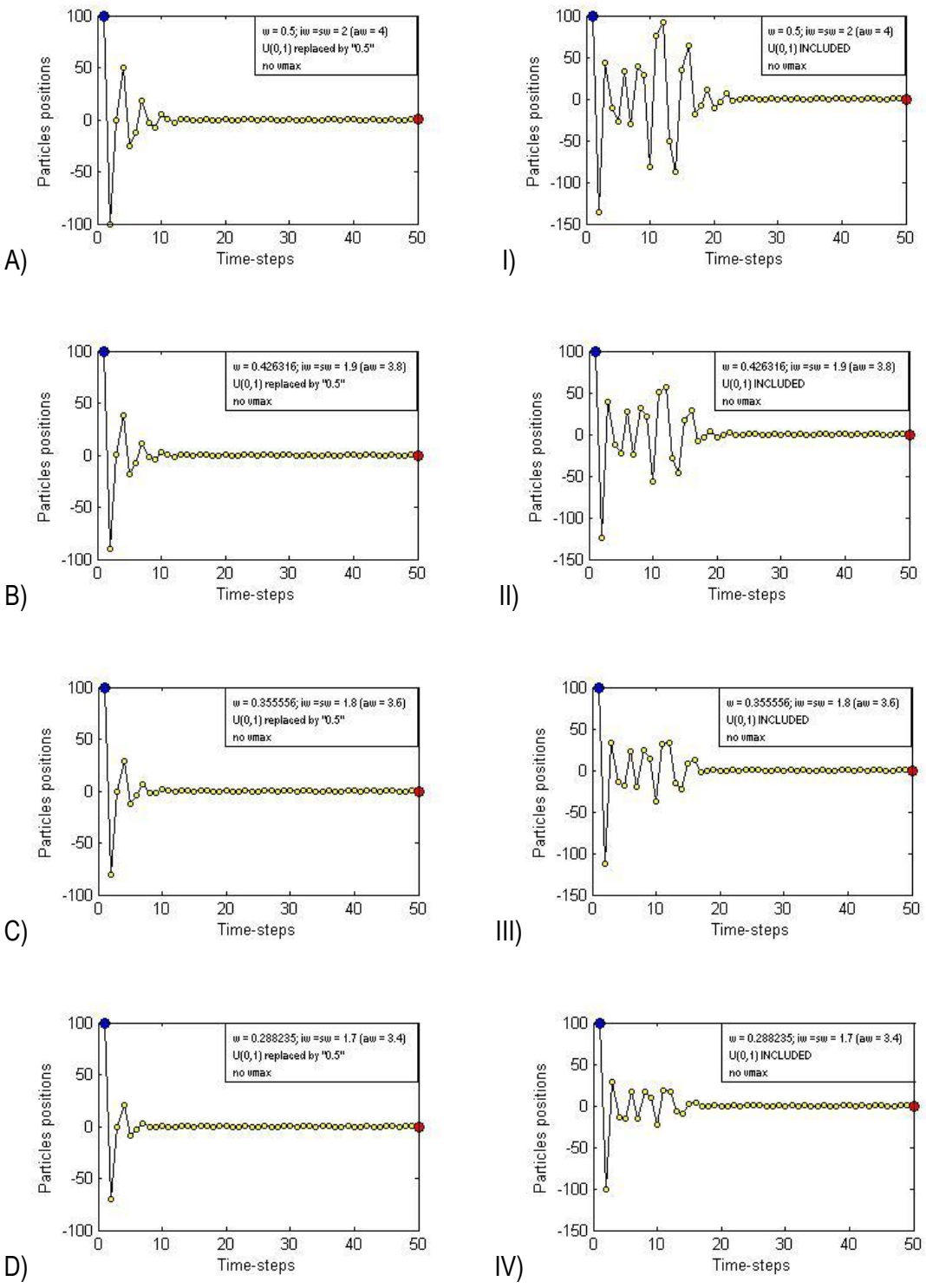


Fig. 5.92. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 4.00 ≥ aw ≥ 3.40, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one. Other five trajectories for each of the cases with $U_{(0,1)}$ included can be found in the digital appendix.

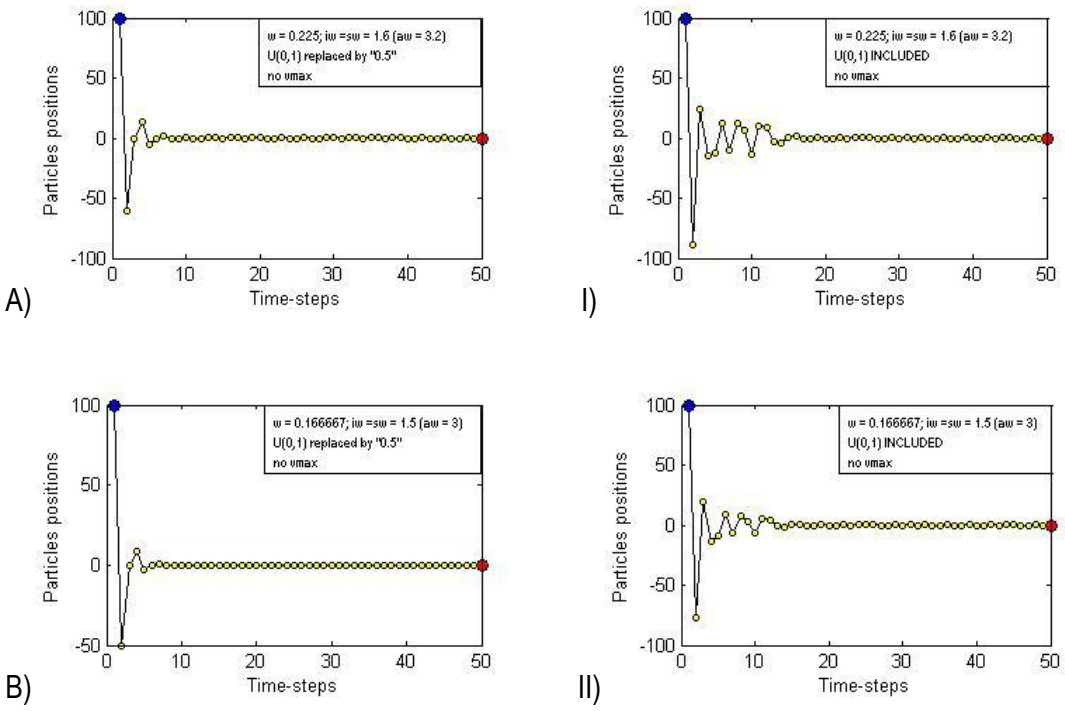


Fig. 5.93. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for aw = 3.20, aw = 3.00, w as in Eq. (5.101), and no v_{max}. Random weights $U_{(0,1)}$ are included in the second column, and replaced by their expected value in the first one. Other five trajectories for each of the cases with $U_{(0,1)}$ included can be found in the digital appendix.

As can be observed in Fig. 5.91, the random weights lead to a stochastic explosion for the highest values of aw tested. Nonetheless, quite surprisingly, there are still values of $4 < aw < 4.5$ that result in convergent trajectories, even with the random weights re-incorporated, for w calculated as in Eq. (5.102). For the cases with $aw \leq 4$, the oscillations show higher amplitudes and somewhat more erratic behaviour for the particle with random weights incorporated compared to its deterministic counterpart. This is not necessarily undesirable, as PSO relies in this sort of behaviour. Besides, the particle rarely leaves the feasible space and is quickly pulled back in. Small values of aw like those in Fig. 5.93 may be useful when extremely fast convergence is required, but are not desirable for a stand-alone, general optimizer.

To prevent the particle from being driven too far from the attractor, the loose velocity constraint $v_{max} = 0.50 \cdot fint$ is implemented. The trajectories are shown in Fig. 5.94 to Fig. 5.96, in the right columns. The trajectories corresponding to the same settings but without the v_{max} constraint are shown in the left column for comparison.

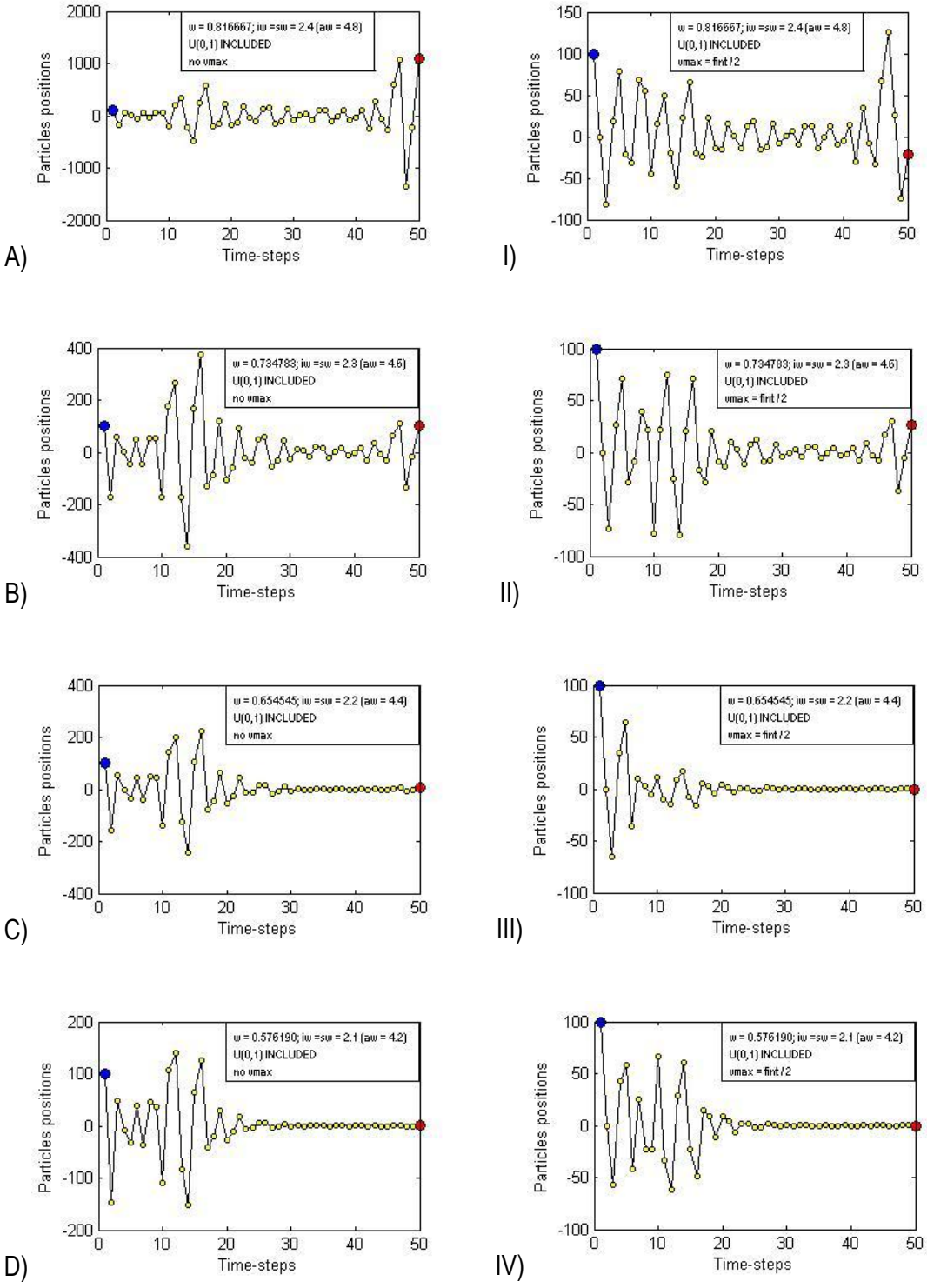


Fig. 5.94. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 4.80 ≥ aw ≥ 4.20, w as in Eq. (5.101), and random weights $U_{(0,1)}$ included. There is no v_{max} in the left column, while it is set to half the feasible interval in the right column. Six trajectories corresponding to consecutive runs for each case can be found in the digital appendix.

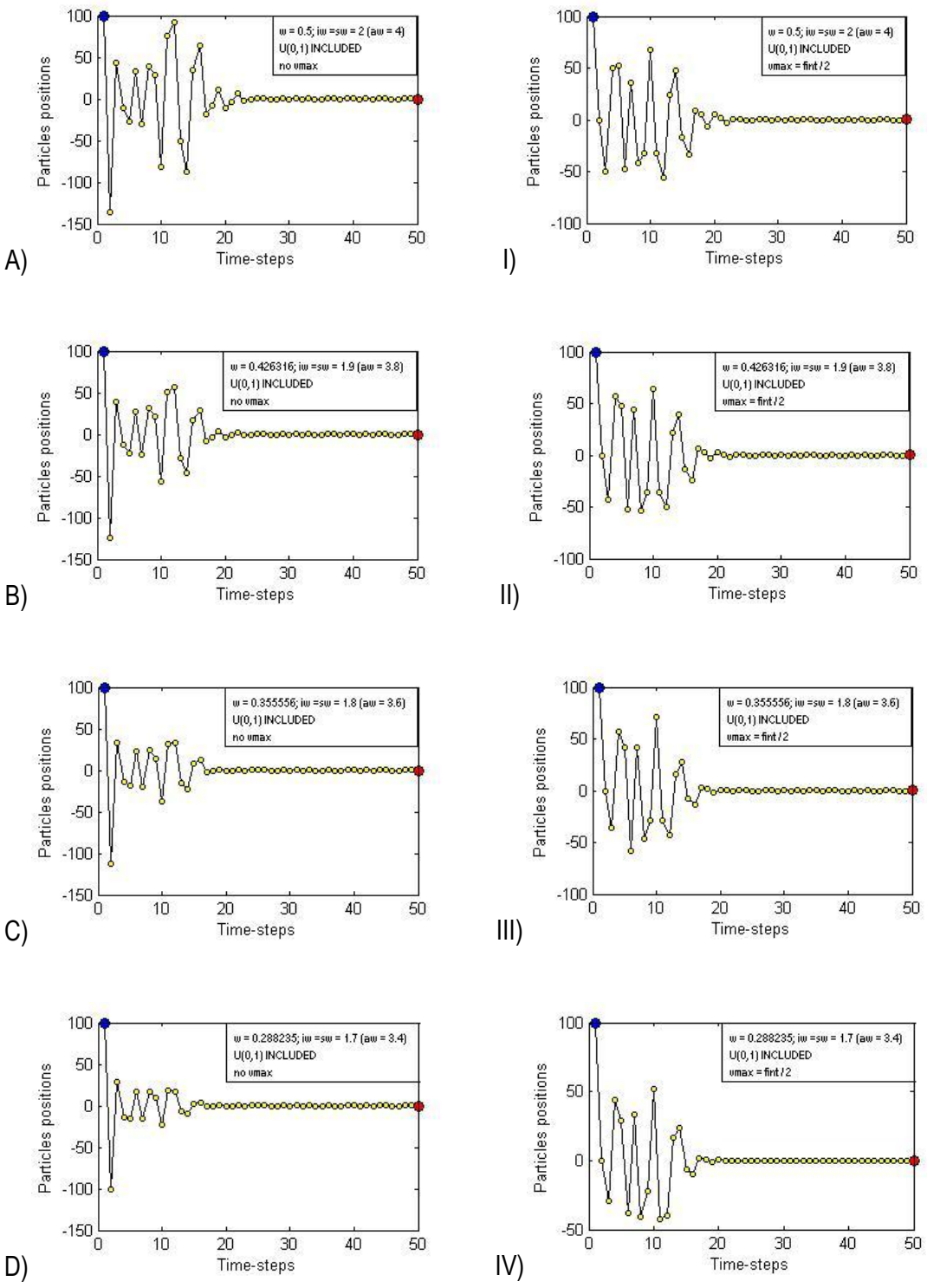


Fig. 5.95. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for 4.00 ≥ aw ≥ 3.40, w as in Eq. (5.101), and random weights $U_{(0,1)}$ included. There is no $v_{\max}$ in the left column, while it is set to half the feasible interval in the right column. Six trajectories corresponding to consecutive runs for each case can be found in the digital appendix.

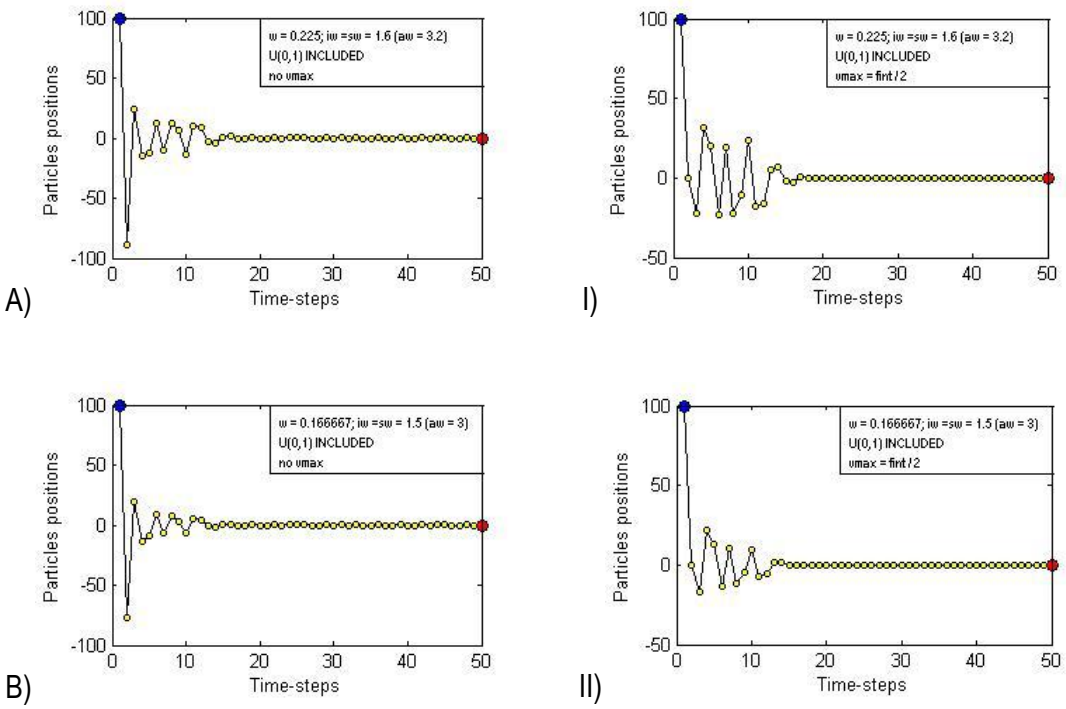


Fig. 5.96. Trajectory of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for aw = 3.20, aw = 3.00, w as in Eq. (5.101), and random weights $U_{(0,1)}$ included. There is no v_{max} in the left column, while it is set to half the feasible interval in the right column. Six trajectories corresponding to consecutive runs for each case can be found in the digital appendix.

The v_{max} constraint prevents the explosion, as it can be clearly seen in Fig. 5.94. It may also prevent some few evaluations of the particle's position a bit far from the attractor (e.g. see Fig. 5.95 A) and I)). In the range analyzed here, convergence consistently takes place for $aw \leq 4.40$, and the amplitude of the oscillations –and hence the size of the space explored– decreases as aw decreases. These results can be observed in all six runs performed for each case. Four additional ones for some selected aw values are offered in Fig. 5.97 to Fig. 5.101.

The convergence area in the plane 'ϕ–w' showed in Fig. 5.20 is offered again in a light gray shadow in Fig. 5.102. The deterministic divergence area is shadowed in dark gray, and the parabola delimitating the complex region is showed in a black dotted line. Note that the inequalities bound the convergence area. The blue dotted line comprises the plot of Eq. (5.102) for $aw = \phi_{max}$, where $0 \leq \phi \leq aw$. Thus ϕ_{mean} is within the convergence region, and so is ϕ most of the time. The red solid line may be either Eq. (5.103) for $aw = \phi_{max}$ or the average behaviour of Eq. (5.102) for $aw = 2 \cdot \phi_{mean}$.

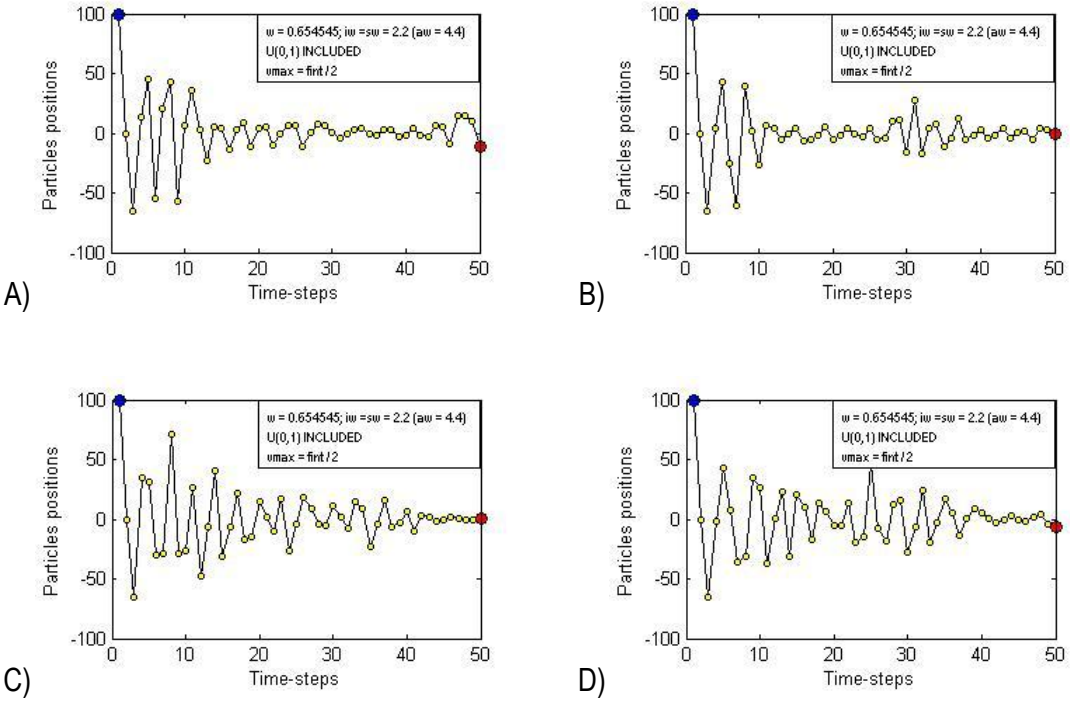


Fig. 5.97. Four possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for aw = 4.40, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and $v_{\max}$ set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.94 III). A 6th run can be found in the digital appendix.

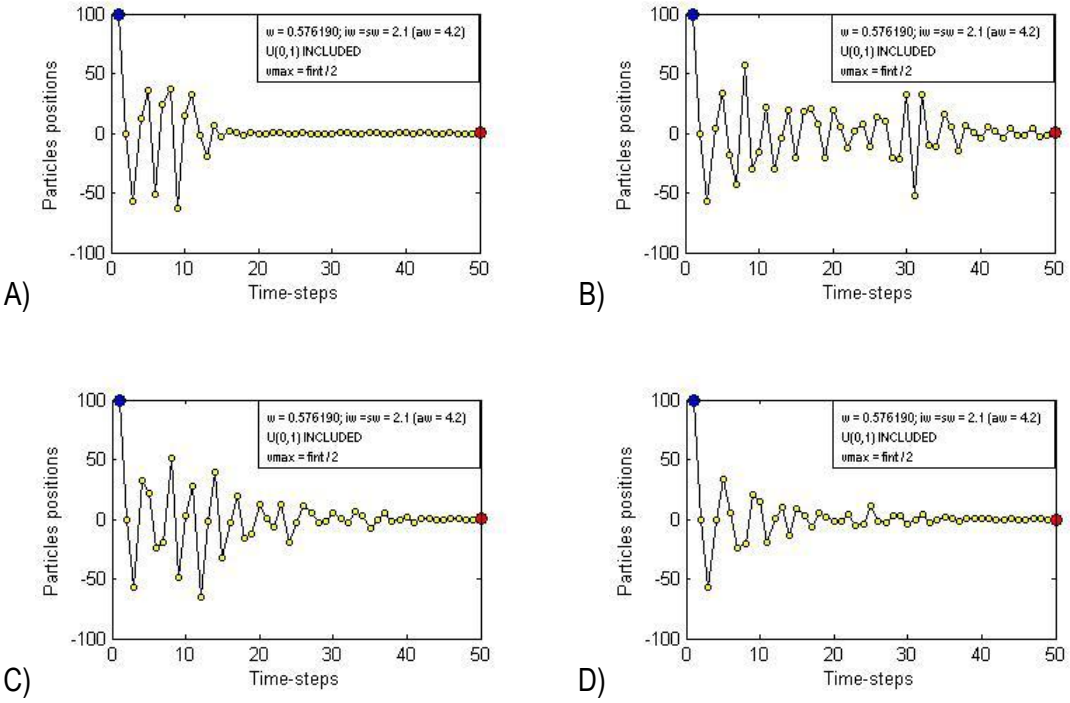


Fig. 5.98. Four possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for aw = 4.20, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and $v_{\max}$ set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.94 IV). A 6th run can be found in the digital appendix.

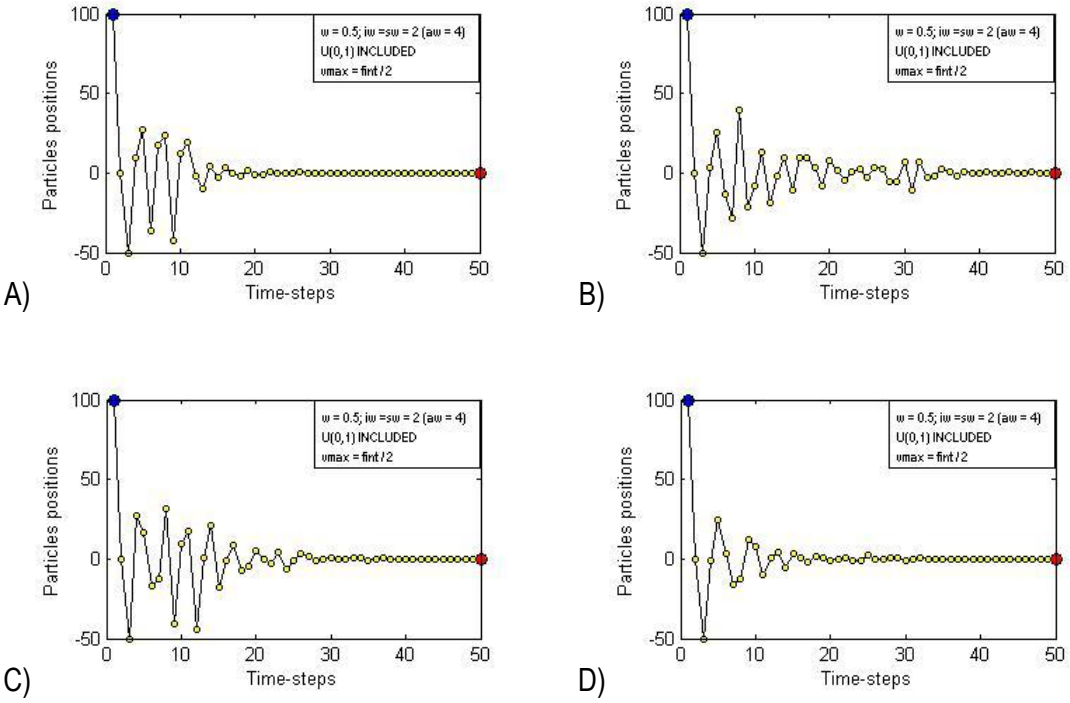


Fig. 5.99. Four possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for aw = 4.00, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and v_{max} set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.95 I). A 6th run can be found in the digital appendix.

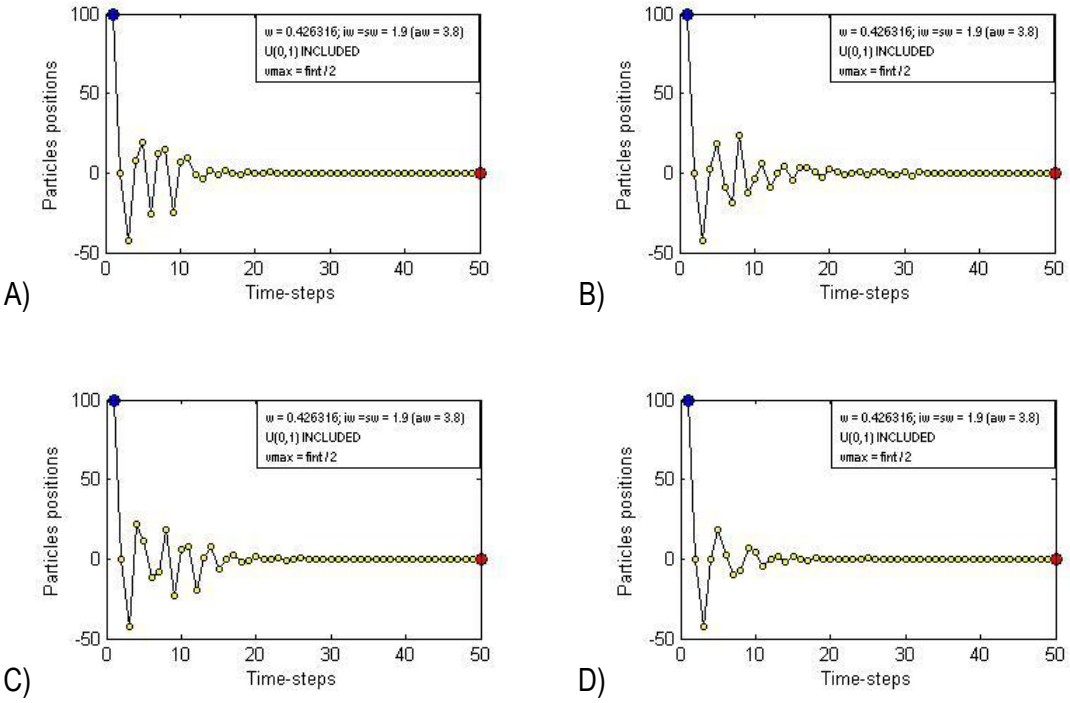


Fig. 5.100. Four possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for aw = 3.80, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and v_{max} set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.95 II). A 6th run can be found in the digital appendix.

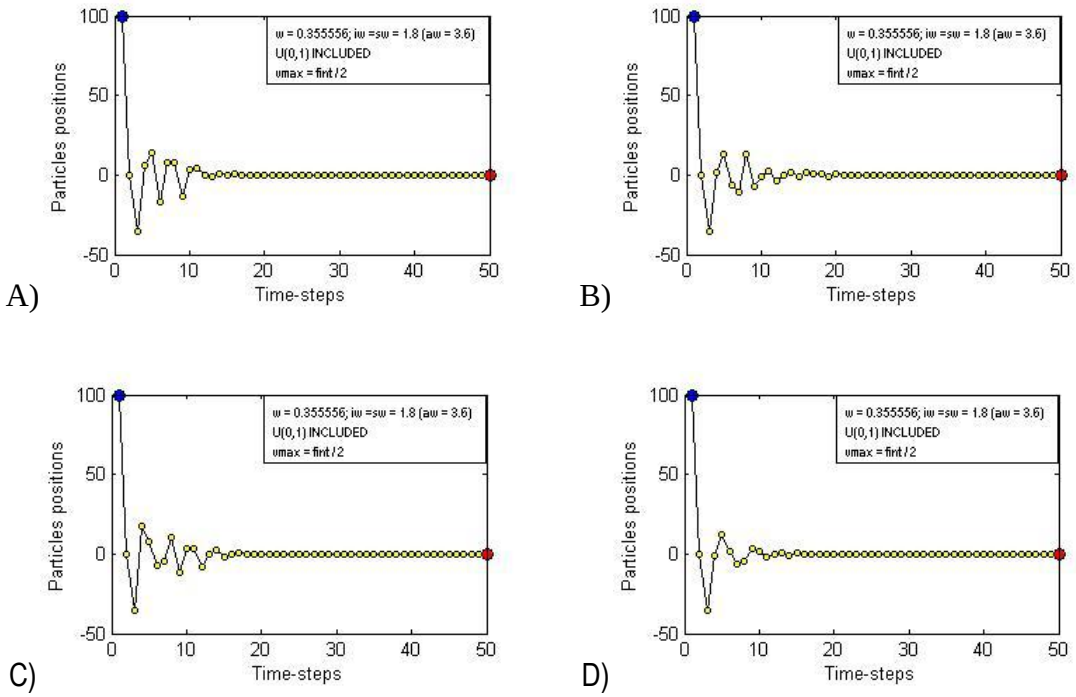


Fig. 5.101. Four possible trajectories of a particle initialized at x = 100 over a 1-dimensional space with stationary attractors at x = 0 for aw = 3.60, w as in Eq. (5.101), random weights $U_{(0,1)}$ included, and v_{max} set to half the feasible interval. They correspond to runs consecutive to that in Fig. 5.95 III). A 6th run can be found in the digital appendix.

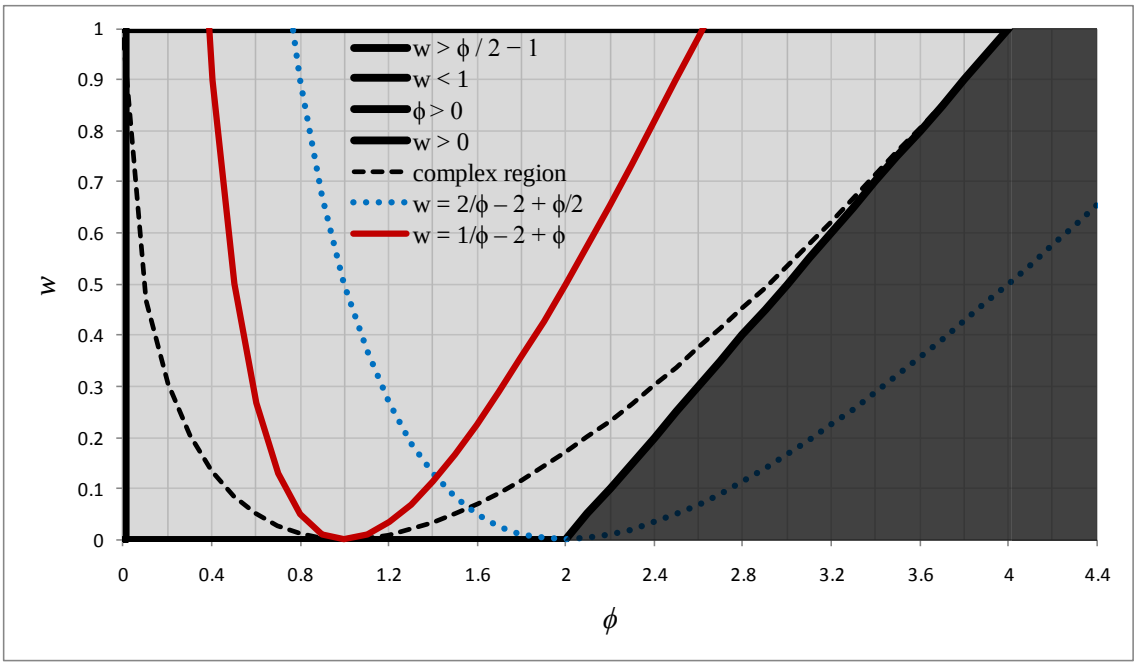


Fig. 5.102. Regions in the 'ϕ–w' plane. The light gray area comprises the convergence region, and the dark gray area the divergence region. The parabola in black dotted line bounds the complex region. The blue dotted line comprises the plot of Eq. (5.102) for $aw = \phi_{max}$ so that ϕ is within the convergence region most of the time. The red solid line may be either Eq. (5.103) for $aw = \phi_{max}$ or the average behaviour of Eq. (5.102) for $aw = 2 \cdot \phi_{mean}$.

5.11. Final remarks and closure

An extensive study of the influence of the settings of the coefficients in the velocity update equation as well as those of the velocity constraint on the behaviour of the isolated particle with stationary attractors have been carried out, partly theoretically, partly heuristically, and partly by simply visualizing trajectories.

Setting the velocity constraint as half the feasible interval is, at worst, harmless, and it is therefore advisable. For some special cases, a quarter of the feasible interval could be considered. Smaller values appear unnecessarily over-restrictive unless convergence is not controlled by the settings of the coefficients but fully driven by progressively decreasing the size of the maximum displacement permitted.

Controlling the explosion only by means of the velocity constraint is not convenient, as it does nothing or little to improve convergence[2]. Therefore, coefficients that lead to divergence or even cyclic behaviour should be discouraged. Although cyclic behaviour is not all that bad in itself, it is not stable, and the random weights tend to turn it into some stochastic explosion. In addition, the search does not concentrate on the most promising areas. Therefore, it is highly advisable to keep at least the average coefficients within the convergence region in Fig. 5.20 (also in Fig. 5.78 and Fig. 5.102).

The region in the 'ϕ–w' plane that leads to convergence was obtained from the closed-form of the particle's position, ensuring that the latter tends to that of the attractor. However, the speed and form of convergence directly depends on the values of the roots of the characteristic polynomial. Thus, according to the form and speed of convergence sought for the particular problem, coefficients from different regions in the convergence map should be selected. Although some authors recommend the use of $w>1$ –e.g. (Shi & Eberhart, 1998a) and (Ozcan & Mohan, 1999) suggest $0.9 \le w \le 1.2$–, it is highly advisable to keep $0<w<1$. A negative w does not make much sense despite convergence because the idea is to maintain some of the velocity from the last movement. In turn, by common sense, $w>1$ would not stand for an inertia term but for a third acceleration one, as new energy would be introduced. Since this coefficient is applied on the previous velocity, it would go against the convergence criteria for which the velocity has to

[2] This is so unless the size of the velocity constraint progressively decreases, thus forcing convergence. Decreasing the velocity to zero is a condition of convergence, as the particle ceases to move.

tend to zero as the particle's position tends to the attractor. From a more analytical point of view, values of $w>1$ lead to the deterministic explosion regardless of the acceleration coefficient ϕ and of the acceleration weight *aw*. Thus, it is strongly recommended to keep $w<1$. Even in the latter case, the higher the inertia weight the greater the magnitude of the roots and therefore the slower the convergence and the lower the frequency of the oscillations. This helps prevent premature convergence in detriment of the ability to perform a fine-grain search. In turn, too low an inertia weight (*w*) should be discouraged, as the particle loses momentum too fast unless *aw* is also small. In the latter case, the particle approaches the attractor mainly from one side, which is obviously undesirable. Loosely speaking, it would be arguably recommendable to choose $0.3\leq w\leq 0.9$, according to the behaviour desired.

A similar common sense analysis would suggest that the acceleration weight should be kept $aw\leq 4$, so that $\phi_{\text{mean}}\leq 2$. This means that the 'average acceleration' would not take the particle farther from the attractor but, at most, the same distance on the other side. The variation introduced by the random weights would allow the particle to move closer and farther. Therefore, it is recommended here that $2\leq aw\leq 4$ even when some of them may fall within the black triangle in Fig. 5.20 (divergence region). As long as ϕ_{mean} is comfortably within the convergence region, the particle is expected to converge. Besides, the velocity constraint prevents possible local explosions. Consider, for instance, the setting $w=0.5$ and $aw=4$, which results in fast convergence and fine-grain search. In contrast, settings like $w=0.9$ and $aw=3.8$ lead to much slower convergence even when the whole range of ϕ is within the convergence area (but the magnitudes of the roots are close to '1'). In turn, a value of $aw<2$ would imply that the particle, on *average*, would not be able to reach –let alone overfly– the attractor due to the acceleration, and a high inertia weight should be used to compensate. Nevertheless, the frequency of the oscillations would be low.

To summarize, within the suggested convergence ranges $0.3\leq w\leq 0.9$ and $2\leq aw\leq 4$, loosely speaking, the lower the *aw* for a given *w* –and the lower the *w* for a given *aw*– the faster the convergence. That is, the more exploitative the behaviour. Conversely, the higher the *aw* for a given *w* –and the higher the *w* for a given *aw*– the stronger the reluctance to premature convergence. That is, the more explorative the behaviour. In turn,

low *w* plus high *aw* lead to high frequencies in the oscillations, and the opposite is true for high *w* and low *aw*. In general, higher frequencies are preferred for exploitation.

The acceleration weight calculated as in Eq. (5.102) favours fast convergence and high frequencies –especially for $aw \leq 4$–, so that it is recommended when exploitation is being sought. Be aware that only the right branch of the blue dotted parabola in Fig. 5.102 should be used, as values of $aw < 2$ are advised against. In general, *aw* is not recommended to be greater than '4'. However, it could be said that for $4.0 < aw \leq 4.4$, *w* as in Eq. (5.102), and $v_{\max} = 0.50 \cdot fint$, the particle still converges more or less in a neatly.

If Eq. (5.103) is used instead of (5.102) so that the dynamics imposed in section 5.10 takes place for $\phi_{\max}$ rather than ϕ_{mean} (red solid parabola in Fig. 5.102), the speed of convergence notably increases, the frequency of the oscillations decreases, and the space explored is narrowed to the vicinity of the attractor. Therefore this is not recommended unless extremely fast convergence and exploitation is desired.

The constriction factor is effective in controlling the explosion and ensuring convergence. It is important to note that, while the constriction factor was developed in terms of ϕ, *aw* is typically used instead in its calculation, as shown in Eq. (5.89). This means that the strength of the constriction of the random variable ϕ is the one corresponding to its maximum possible value regardless of its actual value. A few words on a type of adaptive constriction are offered in the next chapter. The strength of the constriction also depends directly on the value of κ, which is typically set close to '1' for $aw > 4$. The acceleration weight is commonly set marginally greater than '4' because the constriction is too strong otherwise, as can be observed in Fig. 5.77. If $aw < 4$, then κ should be decreased, as it comprises the constriction factor itself (see Fig. 5.78).

Further discussions on the coefficients settings are offered in the next chapter, as complexity is gradually added to the algorithm.

Chapter 6

COEFFICIENTS SETTINGS

The possibility of a form of adaptive constriction factor more in line with its theoretical development is considered. Then, still studying the isolated particle with stationary attractors, some restrictions to the limits of the random weights are investigated so as to obtain a desired average behaviour without abandoning the convergence region. Two average behaviours –and three restrictions to the random weights associated– are proposed. In order to study the trajectory of the isolated particle with a dynamic attractor, its individual best experience is made dynamic. Therefore the weighted attractor becomes dynamic without introducing particles' interactions, and the influence of a moving attractor on the trajectory of the particle is analyzed. The effect of varying the individuality and sociality weights is studied by visualizing the trajectories of an isolated particle with dynamic individual best and stationary social best experiences. Some coefficients settings are preselected, and their influence on a small swarm of four interacting particles in 1-dimensional space is analyzed. The full PSO system with three coefficients' settings (together with a multi-swarm combination of them) and two neighbourhood topologies –one global and one local– are tested on a benchmark suite of multidimensional problems. Finally, some guidelines are provided as to how to set the coefficients to obtain a given desired behaviour of the system.

6.1. Introduction

An extensive analysis on the influence of the inertia and acceleration weight –and a few words about the constriction factor– as well as the velocity constraint on the behaviour of the isolated particle pulled by stationary attractors was offered in the previous chapter. Initially the particle was deterministic, and then random weights were reinstated as a first step to re-incorporate the complexity of the full PSO algorithm. The region of the plane 'ϕ–w' that leads to convergence, as well coefficients leading to different types of trajectories and speeds of convergence, were identified.

The first part of this chapter is still concerned with the isolated particle, but complexity is progressively added. First, a brief discussion about the random weights is offered, which was not considered in the previous chapter. This leads to the discussion of the possibility of using a form of adaptive constriction factor which would seem to be more in line with its theoretical development, since the constriction was developed in terms of the variable ϕ rather than of the constant aw, commonly used in its computation.

Then, a modified version of the PSO algorithm is proposed, where the limits of the random weights are restricted so as to obtain a desired average behaviour without ever leaving the convergence region in the 'ϕ–w' plane. Two alternative average behaviours and three associated restrictions to the random weights are proposed.

Succeeding that, the weighted attractor is made dynamic by incorporating a non-stationary individual best experience. Thus, the stationary global experience can be viewed as the best experience in the neighbourhood, and the effect of an updating attractor in the still isolated particle can be observed. Finally, some empirical studies of setting different individuality and sociality weights are presented.

The second part of this chapter is concerned with the full, multi-particle PSO system. First, the effect of some preselected coefficients' settings on the dynamics of a small swarm of interacting particles flying over 1-dimensional space is visually studied. Then, selected settings profiting from the previous studies are tested and compared on multi-dimensional benchmark problems. Since the neighbourhood topology and the coefficients settings work together to provide the exploration/exploitation balance, the global and one local topologies are considered in the experiments. This chapter ends with some guidelines as to how to set the coefficients to obtain the desired behaviour of the system.

6.2. Single particle

The studies of the effect that the coefficients settings have on the behaviour of the particle are simplified by considering it isolated, even if the studies are not theoretical. Since there is still some more complexity that can be incorporated and studied before moving on to the full multi-particle system, the isolated particle is studied further in this section.

6.2.1. PSO with a form of adaptive constriction

The constricted PSO was proposed in (Clerc & Kennedy, 2002) so that, when the magnitudes of the eigenvalues of the system matrix are not both less than one (convergence condition), a constriction factor is applied. Such constriction modifies the system matrix so that the new eigenvalues comply with the convergence condition. However, the constriction factor is proposed as a function of ϕ, which is a random variable. Typically, ϕ

is simply replaced by $aw = \phi_{max}$ in the formula so as to ensure that the eigenvalues comply with the convergence condition in the whole ϕ interval. However, this is over-restrictive. As shown in the graph 'ϕ–*cf*' in section Fig. 5.77, the strength of the constriction increases remarkably as ϕ grows greater than '4' (for a given κ). Therefore all the values of ϕ smaller than the ϕ_{max} used in the computation of *cf* are over-constricted (for *unconstricted* $\phi_{max} > 4$). It is reasonable to think of a constriction factor specific for each random ϕ (i.e. for each particle, for each dimension, for each time-step). However, this would alter the uniform distribution of the *constricted* ϕ in the range [0,4] (favouring greater values) due to the discontinuity in Fig. 5.77, as can be inferred from Fig. 6.1.

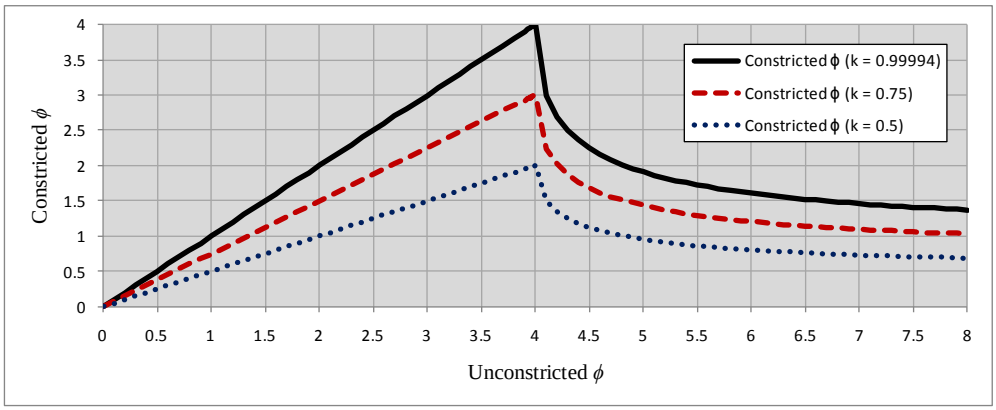


Fig. 6.1. Constricted ϕ as a function of the original, unconstricted ϕ for three different values of κ.

Note that the adaptiveness of *cf* here is with respect to ϕ, in line with its theoretical developments. The idea was to calculate the constriction corresponding to the actual ϕ rather than to ϕ_{max}. This thesis does not deal with more traditional adaptive coefficients schemes such as that in (Chen, Lee, Liao, & Dai, 2007). Thus, it appears more reasonable to maintain a fixed value of *cf* calculated for $\phi = aw = \phi_{max}$, making sure that neither κ is much smaller than '1' nor the *unconstricted* ϕ much greater than '4'. Smaller values of κ and greater values of ϕ result in too strong a constriction (i.e. small *cf*).

6.2.2. Randomness

While the previous chapter was concerned with the influence of the inertia and the acceleration weights on the dynamics of the particle, nothing has been said so far with regards to the random weights. Classical PSO typically multiply the individuality and so-

ciality weights by a random number between '0' and '1' generated from a uniform distribution, referred to as $U_{(0,1)} \in [0,1]$. This means that $\phi \in [0, aw]$ in classical PSO, and $\phi \in [0, cf \cdot aw]$ in constricted PSO. Thus, it has always been assumed that the variable ϕ is bounded by '0' and 'ϕ_{max}', and that the probability distribution is rectangular.

In several sections in the previous chapter, the average behaviour of the isolated particle was studied, identifying some desirable ranges of w and ϕ_{mean} (seeking different forms of convergence). When the random weights $U_{(0,1)}$ were reincorporated, deviations of the expected behaviour were observed, sometimes quite noticeable ones. Since $\phi \in [0, aw]$, the greater the ϕ_{mean} studied the greater the aw, hence the greater the range of ϕ and the greater the deviation from the average behaviour. This need not be like that, and the randomness can be bounded. An alternative that was not extensively investigated –and therefore not included in this thesis– is to view the randomness as a sort of noise to the desired average behaviour. Thus, the individuality and sociality weights would be chosen according to the average behaviour sought, and a random number from a zero-mean normal distribution and standard deviation according to the degree of noise desired could be implemented. This approach would allow the general behaviour to be more controlled, while randomness is still relied on to avoid getting stuck in some particular pattern throughout the search. Similarly, the rectangular distribution is maintained for the studies in this thesis, but different limits for the acceleration coefficient ϕ are proposed. The idea is to decide upon the average behaviour desired associated to a given ϕ_{mean}, and then introduce the random weights so that

$$\phi_{mean} = \frac{\phi_{max} - \phi_{min}}{2} \tag{6.1}$$

and ϕ is never outside the convergence region.

By seeking two different kinds of average behaviours, three approaches are proposed hereafter. Notice that this allows removing one variable to be set by the user, as w and aw are correlated. Thus the user has to set only one of them and the other is automatically calculated. This also provides the user with a simple range of the only variable to be manually set, where the higher the setting the higher the exploration and the lower the setting the higher the exploitation.

6.2.2.1. PSO with reduced ϕ_{max} (PSO-RRM)

The idea here was to find a relationship between the inertia and the acceleration weight that would 'kill the momentum' once the particle has overflown the attractor. In other words, if the particle overflies the attractor from time-step *t* to time-step *t*+1, then the velocity $\mathbf{v}^{(t+2)} = 0$ so that the particle does not tend to keep flying away from the attractor in the following time-step. This implies that $\mathbf{x}^{(t+2)} = \mathbf{x}^{(t+1)}$. While it is not desirable in a deterministic algorithm to re-evaluate the same position, the randomness would make $\mathbf{x}^{(t+2)} \neq \mathbf{x}^{(t+1)}$. Thus, the conditions imposed for the average behaviour are offered in Eq. (6.2), while the resulting relationship between *w* and ϕ_{mean} is shown in Eq. (6.5).

$$\begin{cases} \mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{v}^{(t+1)} \\ \mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)}\right) \\ \mathbf{x}^{(t+2)} = \mathbf{x}^{(t+1)} \end{cases} \tag{6.2}$$

where it is assumed that $\mathbf{v}^{(t)} = \mathbf{0}$, as in section 5.10. Thus, imposing $\mathbf{v}^{(t+2)} = 0$ yields

$$\begin{aligned} &\mathbf{v}^{(t+2)} = w \cdot \mathbf{v}^{(t+1)} + \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t+1)}\right) = 0 \\ &w \cdot \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)}\right) = -\phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t+1)}\right) \end{aligned} \tag{6.3}$$

Replacing $\mathbf{x}^{(t+1)}$ in Eq. (6.3) from Eq. (6.2),

$$\begin{aligned} &w \cdot \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)}\right) = -\phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)} - \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)}\right)\right) \\ &w \cdot \phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)}\right) = -\phi \cdot \left(\mathbf{p} - \mathbf{x}^{(t)}\right) + \phi^2 \cdot \left(\mathbf{p} - \mathbf{x}^{(t)}\right) \\ &w \cdot \phi = -\phi + \phi^2 \end{aligned} \tag{6.4}$$

Finally, the relationship sought is given by Eq. (6.5):

$$\boxed{w = \phi - 1} \tag{6.5}$$

Eq. (6.5) is the middle line between the left and right boundaries of the convergence region in Fig. 5.20 (also in Fig. 5.78 and in Fig. 5.102). Thus, if the range of the random weights is sought to be maximized without ever leaving the convergence region and maintaining the ϕ_{mean} as in Eq. (6.5), then ϕ_{min} = 0 and ϕ_{max} coincides with the hypotenuse of the black triangle in Fig. 5.20. This results in the equation of the velocity update

virtually unmodified with respect to that of the classical PSO: by relating the inertia and acceleration weights as in Eq. (6.6) –see also Eq. 5.82 in section 5.5– and ϕ_{mean} is as in Eq. (6.5), the PSO velocity update equation remains the same.

$$w = \frac{aw}{2} - 1 \tag{6.6}$$

Fig. 6.2 shows the average behaviour given by Eq. (6.5), and the maximum and minimum values of ϕ for each w given by $\phi_{\min} = 0$ and $\phi_{\max}$ as in Eq. (6.6). The blue dotted horizontal lines show the region of the plane 'ϕ–w' from where ϕ is sampled in this approach. The limits $w_{\max} = 0.8$ and $w_{\min} = 0.1$ are arbitrary, as the condition is $w \in (0,1)$.

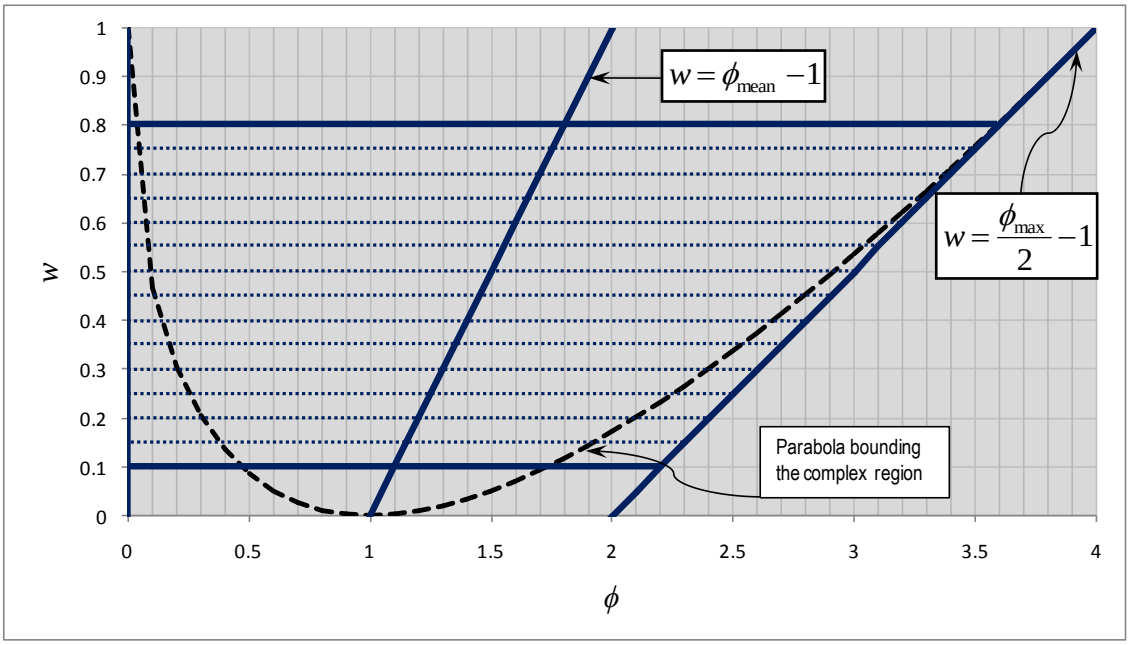


Fig. 6.2. Region in the plane 'ϕ–w' from where ϕ is sampled in PSO-RRM (blue dotted horizontal lines). The middle line rules the average behaviour, while the whole range of ϕ is within the convergence region. The upper and lower limits of w are arbitrary: greater values lead to extremely slow convergence and smaller values to extremely fast convergence. Thus, the approach consists of the user simply choosing $2.20 \leq aw \leq 3.60$, where the lower the aw the more *exploitative* the behaviour, and the higher the aw the more *explorative* the behaviour. This is equivalent to the user choosing $0.10 \leq w \leq 0.80$, where higher inertia weights lead to more extensive exploration.

Thus, the procedure for the PSO-RRM consists of the user simply choosing either the acceleration weight $aw \in [2.20,3.60]$ or the inertia weight $w \in [0.10,0.80]$, where the higher the value selected the more explorative the behaviour, and the lower the value the more exploitative the behaviour. Therefore only one coefficient needs to be set.

6.2.2.2. PSO with reduced randomness range 1 (PSO-RRR1)

The PSO-RRM approach proposed in the previous section was developed seeking the average behaviour given by Eq. (6.5). However, since that happened to be the middle line between $\phi = 0$ and $w = 0.5 \cdot \phi - 1$ (hypotenuse of black triangle in Fig. 5.20) and the range of ϕ was the maximum possible without leaving the convergence region, the approach can be viewed as classical PSO with a relationship between w and aw.

If the same average behaviour is sought but the randomness is viewed as a noise to the desired average behaviour, the range of ϕ can be reduced. The range is arbitrarily set here to half the maximum range, and the approach is called PSO-RRR1. The region in the plane 'ϕ–w' from where the variable ϕ is sampled –for a given w– in this approach is shown in blue, dotted, horizontal lines in Fig. 6.3.

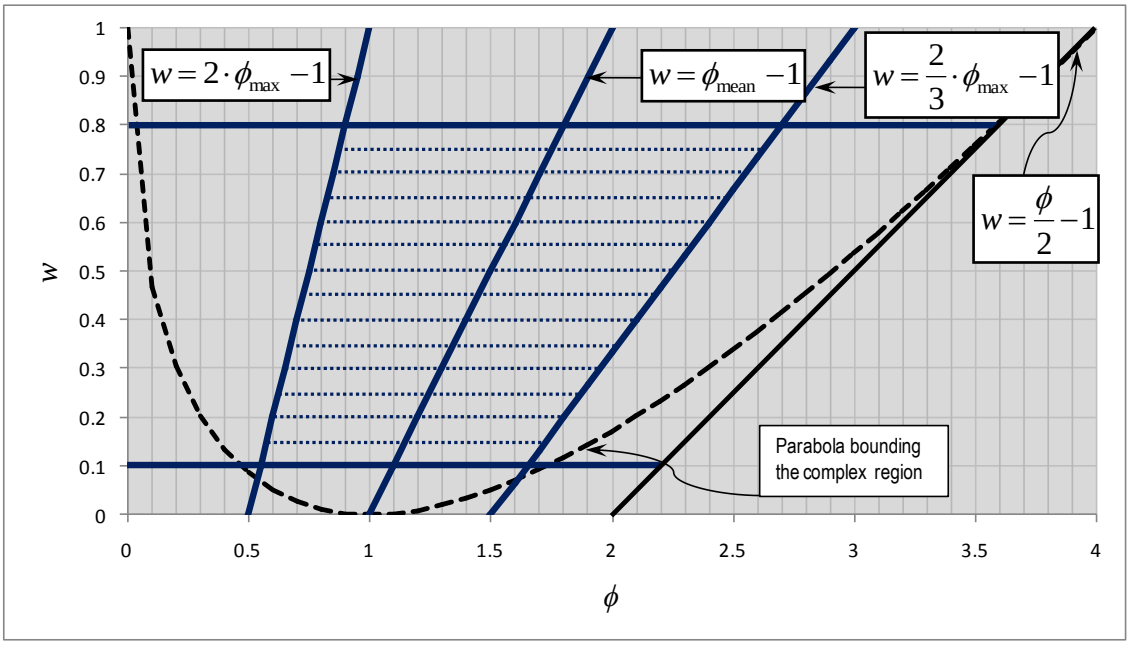


Fig. 6.3. Region in the plane 'ϕ–w' from where ϕ is sampled in PSO-RRR1 (blue dotted horizontal lines). The middle line rules the average behaviour, while the whole range of ϕ is within the convergence region. The upper and lower limits of w are arbitrary: greater values lead to extremely slow convergence and smaller values to extremely fast convergence. Thus, the approach consists of the user simply choosing $1.10 \leq \phi_{mean} \leq 1.80$, where the lower the ϕ_{mean} the more *exploitative* the behaviour, and the higher the ϕ_{mean} the more *explorative* the behaviour. This is equivalent to the user choosing $0.10 \leq w \leq 0.80$, where higher inertia weights lead to more extensive exploration.

Fig. 6.3 shows the equations for ϕ_{min}, ϕ_{mean}, and ϕ_{max}, where the random variable ϕ is always sampled within the complex and convergent region in the plane 'ϕ–w'. The same as for the PSO-RRM, the upper and lower limits for w are arbitrary. Higher values lead to extremely slow convergence, and the opposite is true for lower values.

Given that the PSO-RRR1 does not allow the probability of $\phi = 0$, the equation for the velocity update presents a marginal modification with respect to that of the classical PSO in Eq. (5.2). Thus, the updates of the velocity and position in PSO-RRR1 are performed as shown in Eq. (6.7).

$$\begin{cases} v_{ij}^{(t)} = w \cdot v_{ij}^{(t-1)} + \phi_i \cdot \left(pbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) + \phi_s \cdot \left(lbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) \\ w = \phi_{\text{mean}} - 1 \\ \phi_i = \dfrac{iw}{iw + sw} \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \\ \phi_s = \dfrac{sw}{iw + sw} \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \\ \phi_{\max} = \dfrac{3}{2} \cdot (w+1) \\ \phi_{\min} = \dfrac{1}{2} \cdot (w+1) \\ x_{ij}^{(t)} = x_{ij}^{(t-1)} + v_{ij}^{(t)} \end{cases} \tag{6.7}$$

Recall that the average behaviour of both the PSO-RRM and the PSO-RRR1 are given by Eq. (6.5). Eq. (6.7) shows that it is not *iw* and *sw* but their relative values which matter. Therefore, the user simply needs to weigh the acceleration, stressing either the *self-confidence* (individuality) or the *conformism* (sociality) of each particle, as shown in Eq. (6.8). Setting equal strength to both terms results in $ip = sp = 0.50$.

$$\begin{aligned} & ip = \frac{iw}{iw + sw} \in [0,1) \quad ; \quad sp = 1 - ip \\ & \phi_i = ip \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \\ & \phi_s = sp \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \end{aligned} \tag{6.8}$$

Notice that the formulation in Eq. (6.7) also applies to classical PSO except that $\phi_{\min} = 0$ and $\phi_{\max} = aw$. Thus, the procedure for the PSO-RRR1 approach consists of the user simply choosing either $\phi_{\text{mean}} = [1.10, 1.80]$ or $w = [0.10, 0.80]$ and $ip \in [0,1)$, where the higher the values selected the more explorative the behaviour. Conversely, the lower the values selected the more exploitative the behaviour. Therefore, two coefficients need to be set, which meanings are straightforward.

The trajectories of the deterministic isolated particle for decreasing values of ϕ_{mean} –and therefore decreasing values of w– are offered in Fig. 6.4 and Fig. 6.5. Recall that $\phi_{\text{mean}} = 0.50 \cdot aw$ for the PSO-RRM.

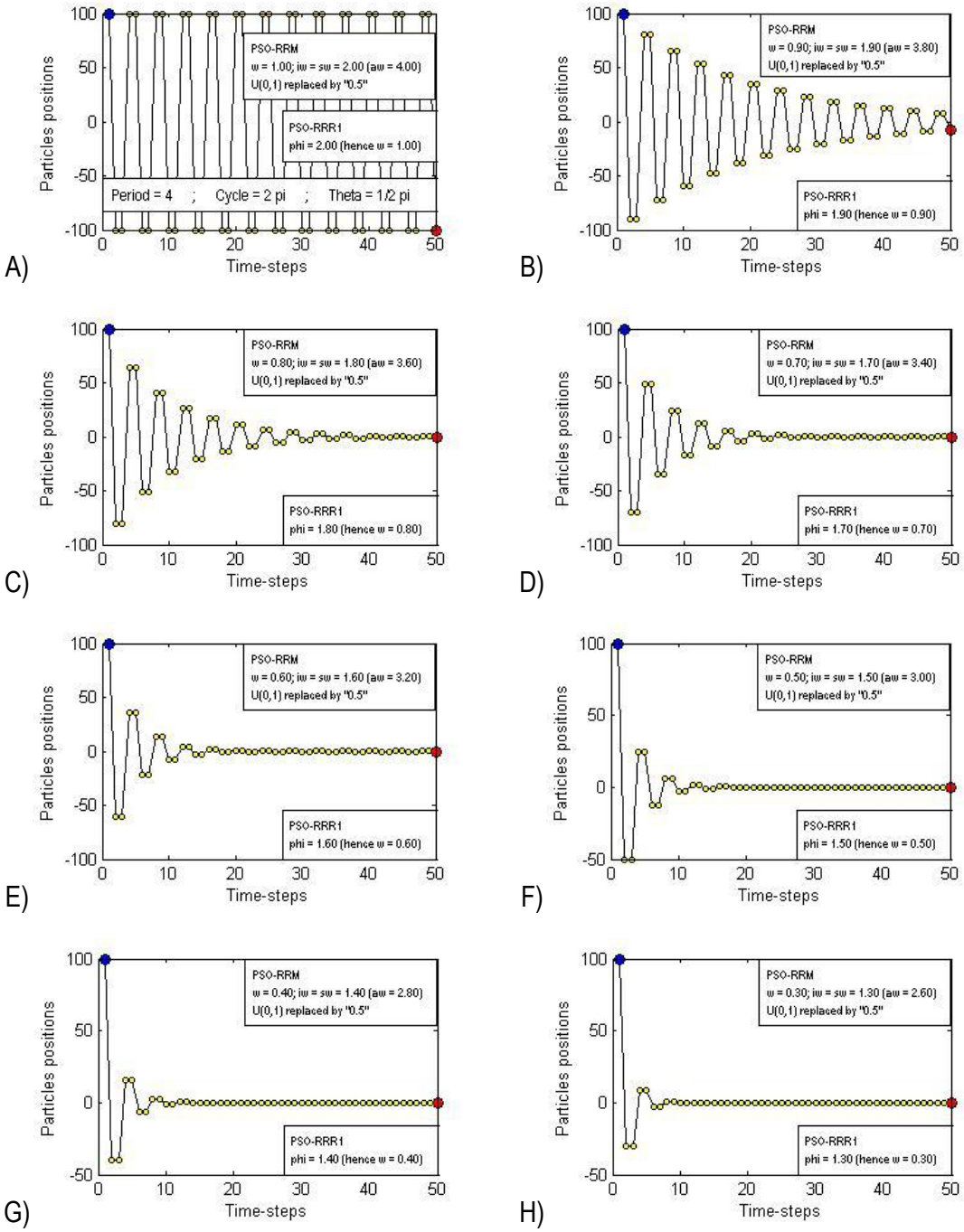


Fig. 6.4. Trajectory corresponding to the average behaviour of a deterministic particle initialized at x = 100, flying over a 1-dimensional space, with stationary attractors at x = 0, for 1.00 ≥ w ≥ 0.30. This average behaviour is exhibited by both the PSO-RRM and the PSO-RRR1.

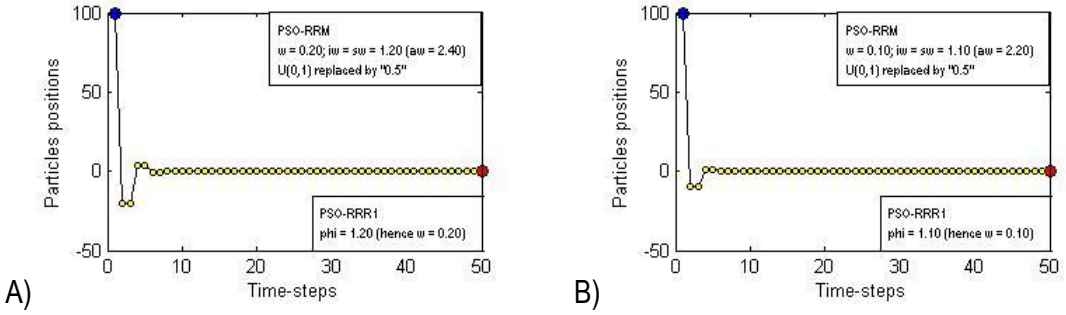


Fig. 6.5. Trajectory corresponding to the average behaviour of a deterministic particle initialized at x = 100, flying over a 1-dimensional space, with stationary attractors at x = 0, for w = 0.20 (left) and w = 0.10 (right). This average behaviour is exhibited by both the PSO-RRM and the PSO-RRR1.

Notice that Fig. 6.4 A) is the average behaviour of the original PSO algorithm ($w=1$ and $aw=4$). As can be observed in all the average trajectories offered in Fig. 6.4 and Fig. 6.5, the position of the particle when it has just overflown the attractor is repeated in the following time-step, while it overflies the attractor again in the time-step after that. This was the condition imposed in Eq. (6.2), yielding Eq. (6.5). Thus, the average behaviour of both the PSO-RRM and the PSO-RRR1 is the same as that of the original PSO, except that the amplitudes of the oscillations are time-decreasing. As mentioned before, the smaller the ϕ_{mean} (aw for the PSO-RRM) –and therefore the smaller the w– the steeper the decrease of the amplitudes. That is, the faster the convergence.

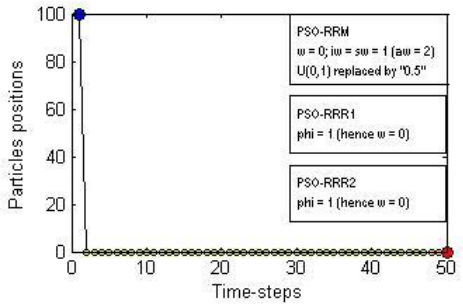


Fig. 6.6. Trajectory corresponding to the average behaviour of a deterministic particle initialized at x = 100, flying over a 1-dimensional space with stationary attractors at x = 0, for w = 0.00. This average behaviour is exhibited by both the PSO-RRM (aw = 2) and the PSO-RRR1 (ϕ = 1), as well as the PSO-RRR2 proposed in the next section.

Six trajectories corresponding to consecutive runs once randomness is reincorporated are offered in Fig. 6.7 and Fig. 6.8 for $w=0.80$, for the PSO-RRM (hence $aw=3.60$) on the left column, and for the PSO-RRR1 (hence $\phi_{\text{mean}}=1.80$) on the right column.

Trajectories in the same row correspond to the same initial state of the pseudo random number generator.

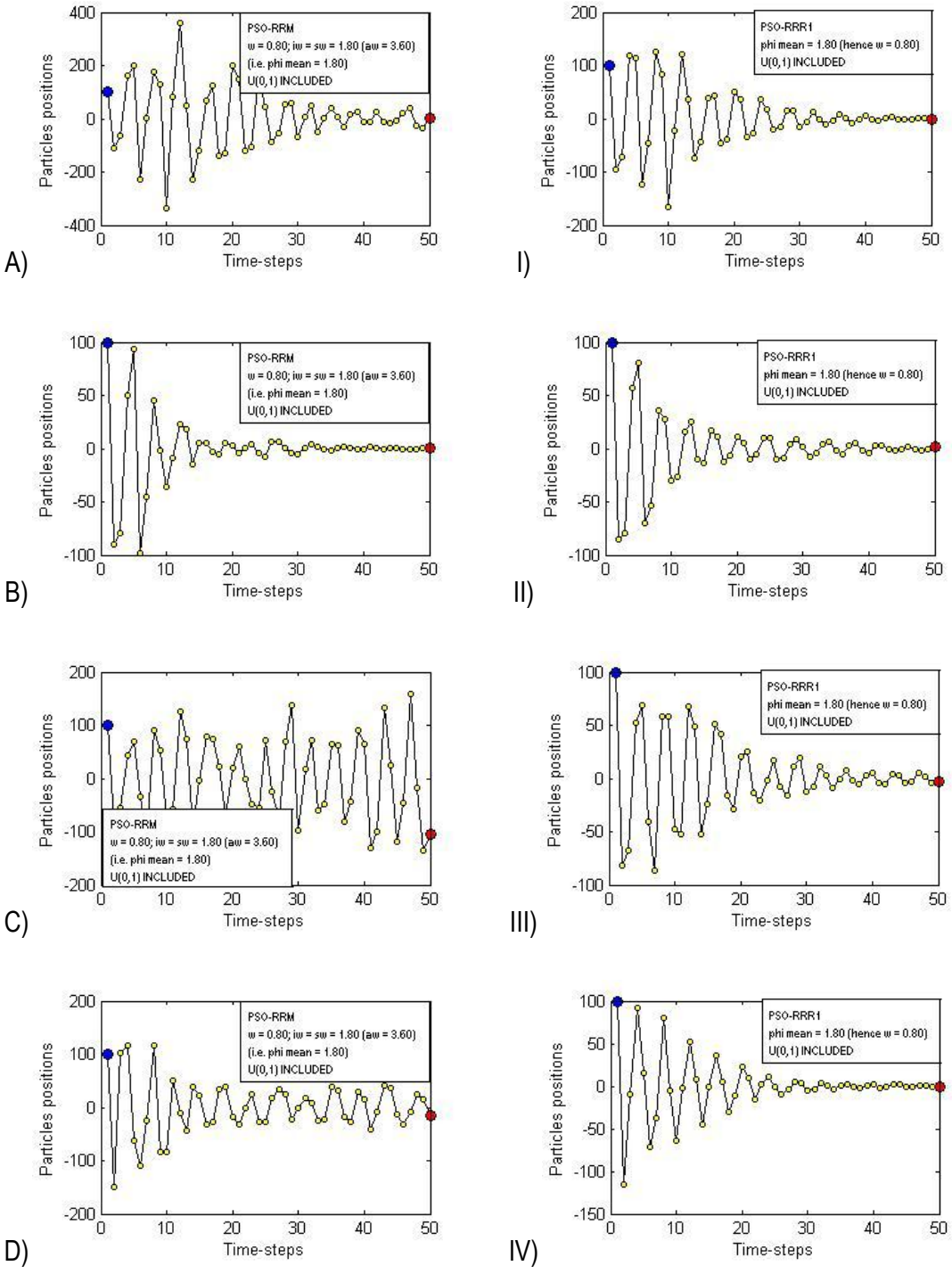


Fig. 6.7. Trajectory of a particle initialized at x = 100, over a 1-dimensional space with stationary attractors at x = 0, for w = 0.80, corresponding to four consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1. Trajectories in the same row correspond to the same initial state of the pseudo random number generator.

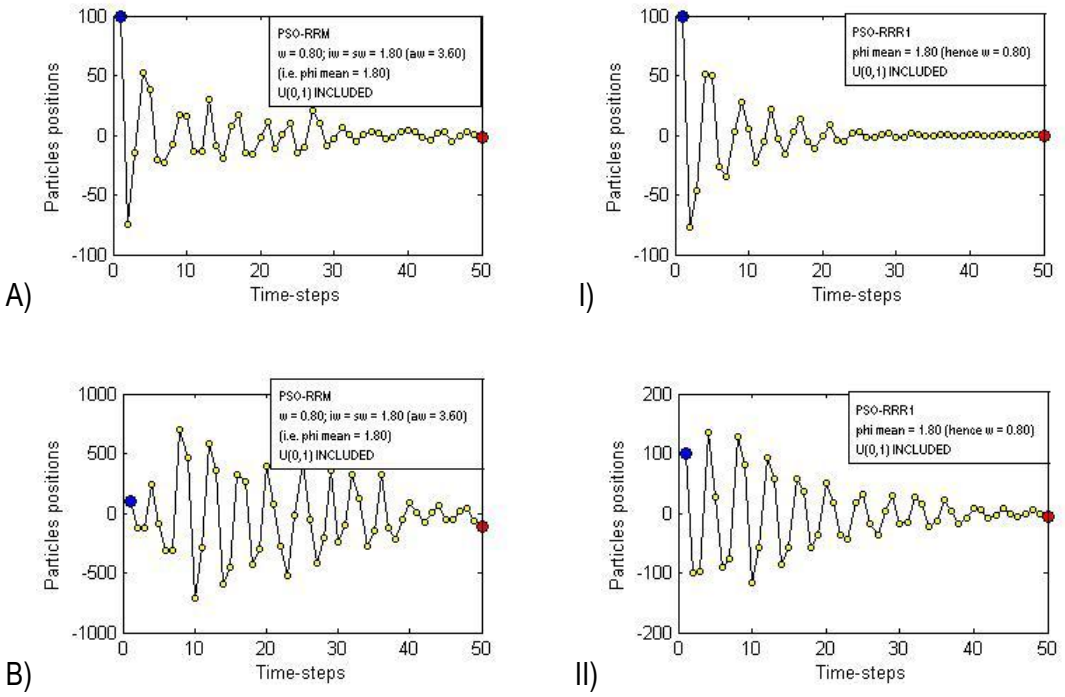


Fig. 6.8. Trajectory of a particle initialized at x = 100, over a 1-dimensional space with stationary attractors at x = 0, for w = 0.80, corresponding to two consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1. Trajectories in the same row correspond to the same initial state of the pseudo random number generator. The runs considered here are consecutive to those in Fig. 6.7.

As can be observed, despite having the same average behaviour (i.e. the same ϕ_{mean}), the PSO-RRR1 exhibit a more desirable and consistent behaviour, where the trajectories deviate much less from the deterministic counterpart without giving up on randomness. Notice that convergence takes place every time –as opposed to Fig. 6.7 C)–, there is no local explosion –as opposed to Fig. 6.7 A) and Fig. 6.8 B)–, and the form and speed of convergence are more similar to one another for all six runs of the PSO-RRR1. It is important to note that no velocity constraint is implemented here, and yet explosion and convergence are conveniently controlled for an inertia weight as high as '0.80' and a ϕ_{mean} as high as '1.80'. Recall that $w = 2.00$ and $\phi_{\text{mean}} = 2.00$ for the original PSO.

Six trajectories corresponding to consecutive runs once randomness is reincorporated are offered in Fig. 6.9 and Fig. 6.10 for $w = 0.50$, for the PSO-RRM (hence $aw = 3.00$) on the left column, and for the PSO-RRR1 (hence $\phi_{\text{mean}} = 1.50$) on the right column. Trajectories in the same row correspond to the same initial state of the pseudo random number generator.

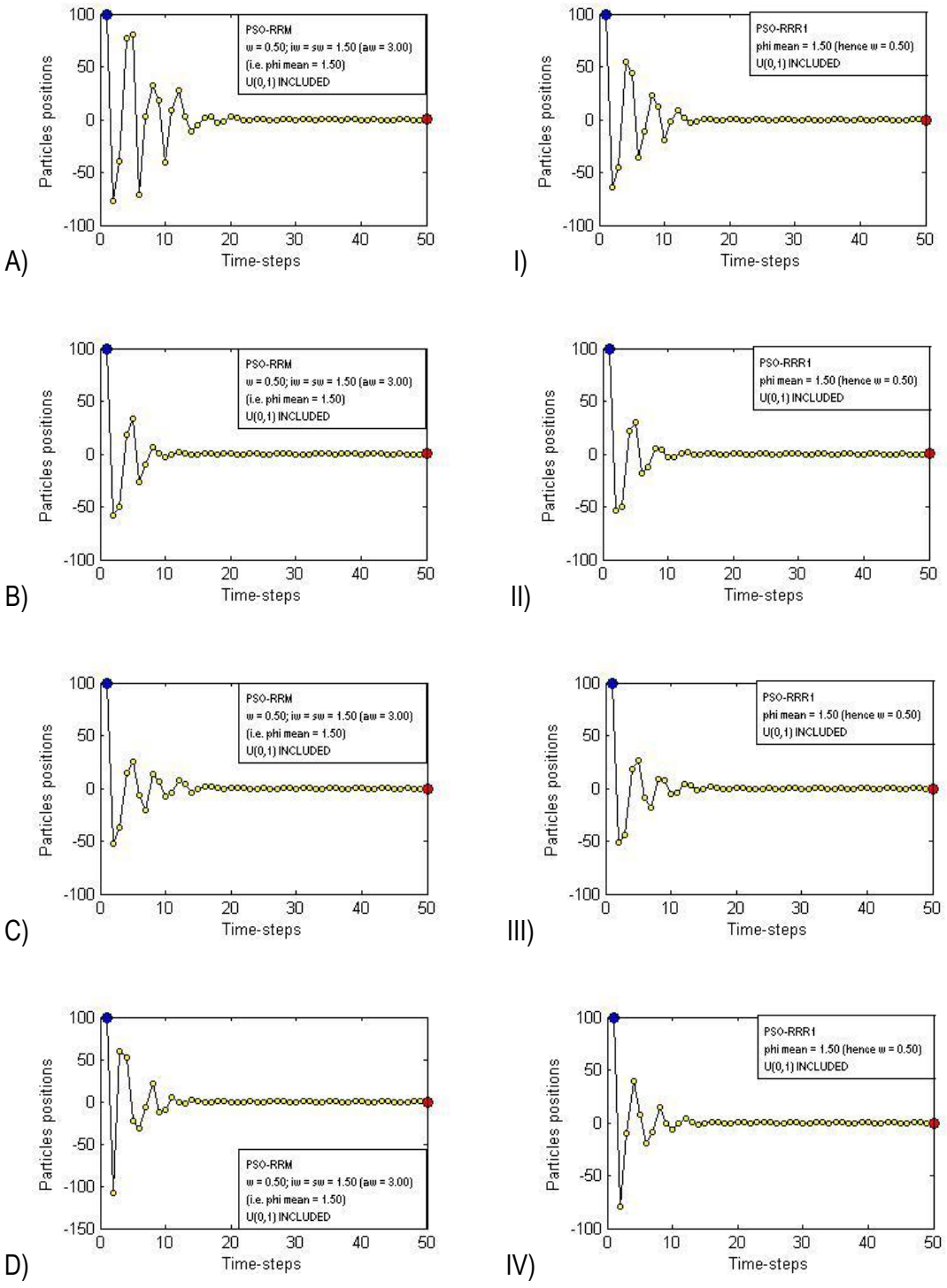


Fig. 6.9. Trajectory of a particle initialized at x = 100, over a 1-dimensional space with stationary attractors at x = 0, for w = 0.50, corresponding to four consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1. Trajectories in the same row correspond to the same initial state of the pseudo random number generator.

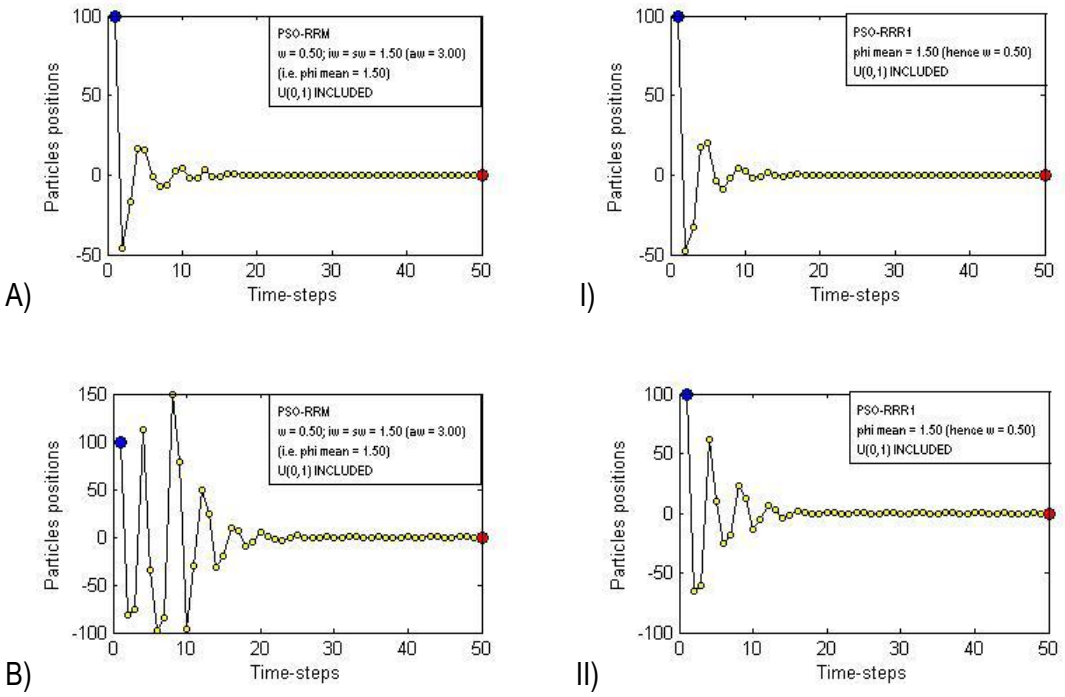


Fig. 6.10. Trajectory of a particle initialized at x = 100, over a 1-dimensional space with stationary attractors at x = 0, for w = 0.50, corresponding to two consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1. Trajectories in the same row correspond to the same initial state of the pseudo random number generator. The runs considered here are consecutive to those in Fig. 6.9.

As mentioned before, if ϕ_{mean} is as in Eq. (6.5), the lower the ϕ_{mean} –and hence the lower the w– the faster the convergence. Thus convergence is noticeably faster in Fig. 6.9 and Fig. 6.10 in comparison to that in Fig. 6.7 and Fig. 6.8. In addition, the form and speed of convergence are more similar for different runs because the interval $[\phi_{\min}, \phi_{\max}]$ decreases as ϕ_{mean} and w decrease, and no local explosion is observed. Thus trajectories deviate less from their deterministic counterparts. Nevertheless, the trend is the same as in Fig. 6.7 and Fig. 6.8, and trajectories are more consistent for the PSO-RRR1. For instance, compare figures A) to B), and figures I) to II) in Fig. 6.10.

In order to compare both approaches for extremely fast convergent settings, six trajectories corresponding to consecutive runs with randomness reincorporated are offered in Fig. 6.11 and Fig. 6.12 for $w = 0.20$, for the PSO-RRM (hence $aw = 2.40$) on the left column, and for the PSO-RRR1 (hence $\phi_{\text{mean}} = 1.20$) on the right one. Trajectories in the same row correspond to the same initial state of the pseudo random number generator.

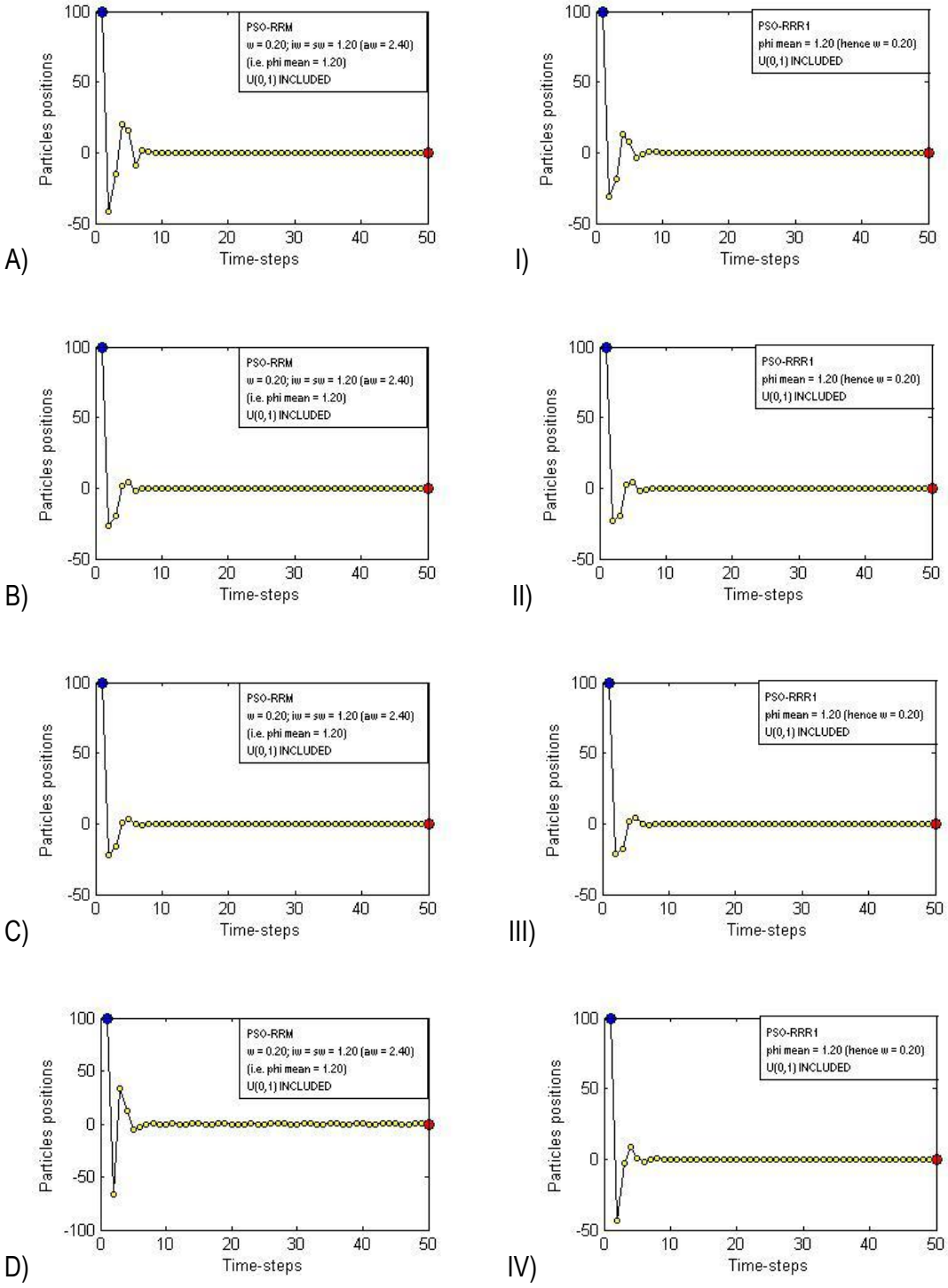


Fig. 6.11. Trajectory of a particle initialized at x = 100, over a 1-dimensional space with stationary attractors at x = 0, for w = 0.20, corresponding to four consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1. Trajectories in the same row correspond to the same initial state of the pseudo random number generator.

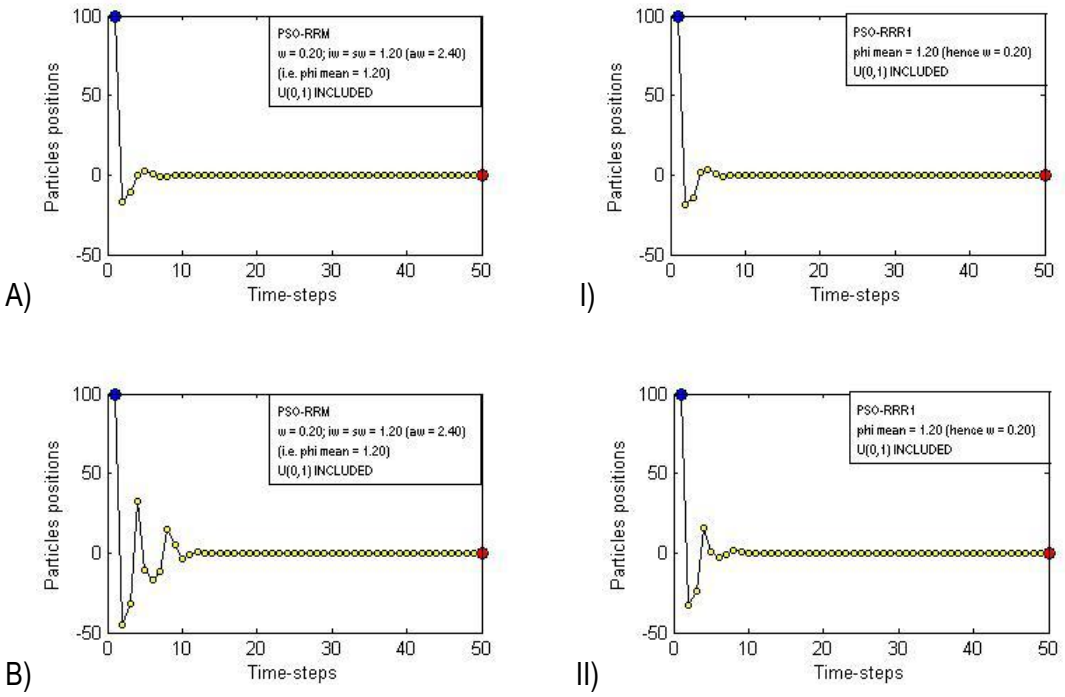


Fig. 6.12. Trajectory of a particle initialized at x = 100, over a 1-dimensional space with stationary attractors at x = 0, for w = 0.20, corresponding to two consecutive runs. The trajectories on the left column correspond to the PSO-RRM and the ones on the right to the PSO-RRR1. Trajectories in the same row correspond to the same initial state of the pseudo random number generator. The runs considered here are consecutive to those in Fig. 6.11.

As expected, the difference in the trajectories for the approaches decreases as the interval $[\phi_{\min}, \phi_{\max}]$ decreases, although the PSO-RRR1 still remains more consistent. Recall that these small settings are not advised for stand-alone optimizers.

6.2.2.3. PSO with reduced randomness range 2 (PSO-RRR2)

The average behaviour imposed by Eq. (6.5) in the PSO-RRM and the PSO-RRR1 was developed so that the inertia and the acceleration cancelled each other out in the time-step following the overflying of the attractor (see Eq. (6.2)). This was a way to conveniently balance the strength of the inertia and that of the acceleration. Another way to balance them but which results in higher frequencies of the trajectory is the relationship between ϕ and w proposed in Eq. (5.101) in section 5.10. This leads to higher accelerations for a given w than for the PSO-RRM and PSO-RRR1 because the acceleration is such that not only cancels the inertia but also drives the particle to the attractor's location. Of course, once randomness is reincorporated, this behaviour is altered.

In section 5.10, this average behaviour was considered with $\phi \in [0, aw]$, which resulted in ϕ being sampled from vast regions of the plane 'ϕ–w', some of which were within the divergence region.

The PSO-RRR2 approach proposed here combines the average behaviour in Eq. (5.101) with a reduced randomness range criterion. Thus, the variable ϕ is randomly sampled from an interval that has Eq. (5.101) as its midpoint, and the hypotenuse of the divergence triangle as its rightmost point. This is the greatest interval fully within the convergence region with the average behaviour sought. The region of the plane 'ϕ–w' from where ϕ is randomly sampled –for a given w– is shown in Fig. 6.13. The upper and lower limits for w are arbitrary. While it is advised not to violate those limits, some greater and smaller values are considered when visually studying the average behaviour.

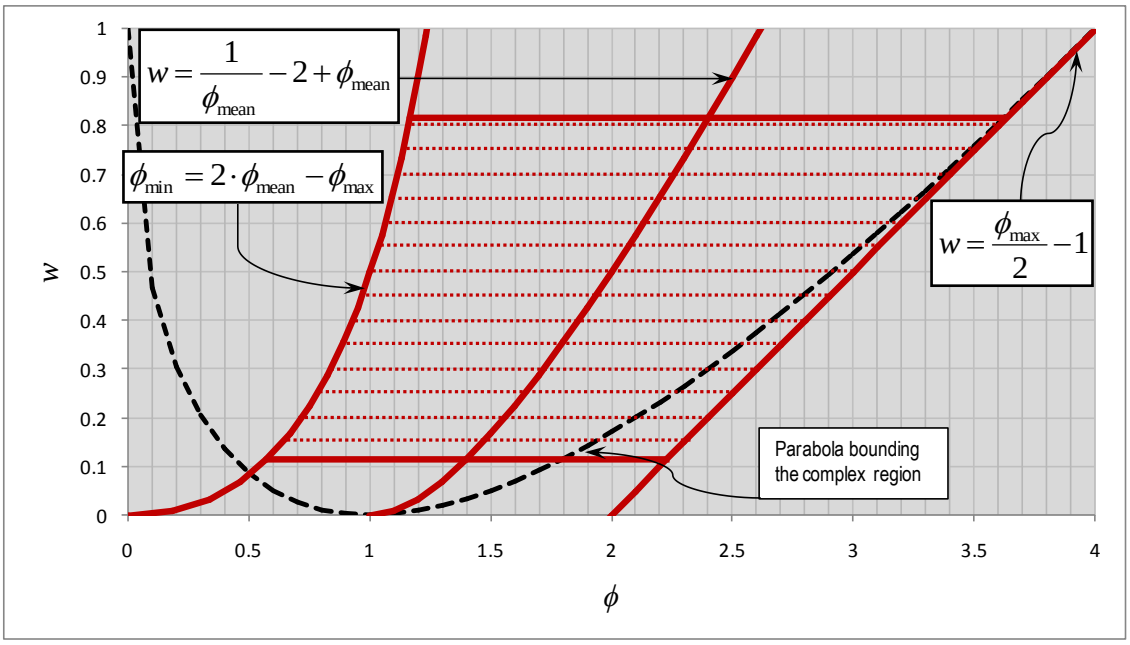


Fig. 6.13. Region in the plane 'ϕ–w' from where ϕ is sampled in PSO-RRR2 (red dotted horizontal lines). The middle line rules the average behaviour, while the whole range of ϕ is within the convergence region. The upper and lower limits of w are arbitrary: greater values lead to extremely slow convergence and smaller values to extremely fast convergence. Thus, the approach consists of the user simply choosing $1.40 \leq \phi_{mean} \leq 2.40$, where the lower the ϕ_{mean} the more *exploitative* the behaviour, and the higher the ϕ_{mean} the more *explorative* the behaviour.

In the same fashion as the PSO-RRR1, the PSO-RRR2 does not allow the probability of $\phi = 0$. Therefore the equation for the velocity update presents a marginal modification with respect to that of the classical PSO in Eq. (5.2). Thus, the updates of the velocity and position in PSO-RRR2 are carried out as shown in Eq. (6.9).

$$\begin{cases} v_{ij}^{(t)} = w \cdot v_{ij}^{(t-1)} + \phi_i \cdot \left(pbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) + \phi_s \cdot \left(lbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) \\ w = \dfrac{1}{\phi_{\text{mean}}} - 2 + \phi_{\text{mean}} \\ \phi_i = \dfrac{iw}{iw+sw} \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \\ \phi_s = \dfrac{sw}{iw+sw} \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \\ \phi_{\max} = 2 \cdot \left(w+1 \right) \\ \phi_{\min} = 2 \cdot \phi_{\text{mean}} - \phi_{\max} \\ \\ x_{ij}^{(t)} = x_{ij}^{(t-1)} + v_{ij}^{(t)} \end{cases} \tag{6.9}$$

Since it is not *iw* and *sw* but their relative values which matter, the user simply needs to weigh the acceleration stressing either the *self-confidence* (individuality term) or the *conformism* (sociality term) of each particle, as shown in Eq. (6.10). Setting the same strength to both terms results in $ip = sp = 0.50$.

$$\begin{aligned} ip &= \frac{iw}{iw+sw} \in [0,1) \quad ; \quad sp = 1 - ip \\ \phi_i &= ip \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \\ \phi_s &= sp \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \end{aligned} \tag{6.10}$$

Thus, the procedure for the PSO-RRR2 approach consists of the user simply choosing $\phi_{\text{mean}} = [1.40, 2.40]$ and $ip \in [0,1)$, where the higher the values selected the more explorative the behaviour, and the lower the values selected the more exploitative the behaviour. Therefore, two coefficients need to be set, whose meanings are straightforward.

The trajectories of the deterministic isolated particle for decreasing values of ϕ –and hence decreasing values of *w*– are offered in Fig. 6.14 and Fig. 6.15. The trajectories once randomness is reincorporated are offered in Fig. 6.16 for $w \cong 0.80$, in Fig. 6.17 for $w \cong 0.50$, and in Fig. 6.18 for $w \cong 0.20$ to allow comparisons with the PSO-RRM and the PSO-RRR1.

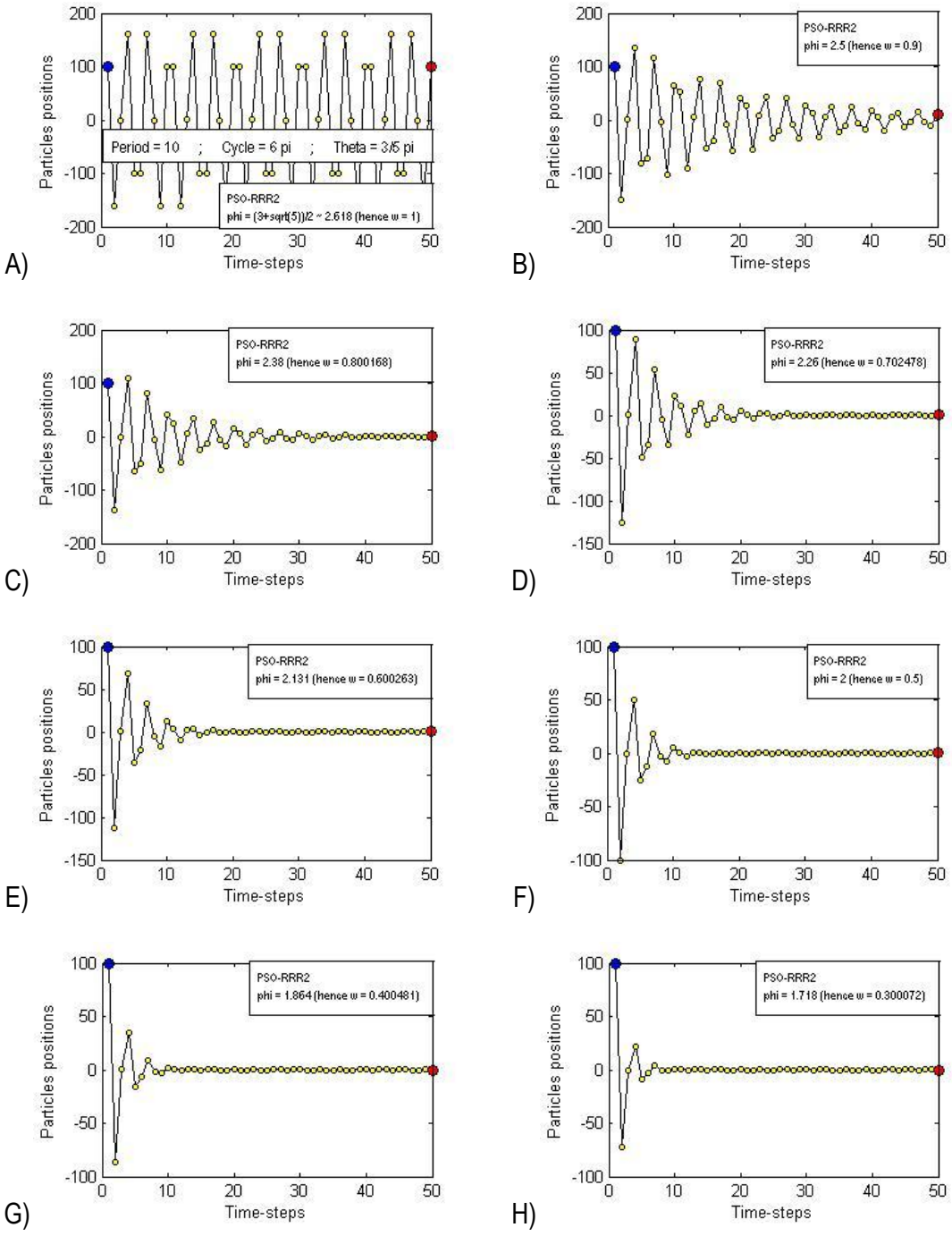


Fig. 6.14. Trajectory corresponding to the average behaviour of a deterministic particle initialized at x = 100, flying over a 1-dimensional space, with stationary attractors at x = 0, for 1.00 ≥ w ≥ 0.30, for the PSO-RRR2.

Fig. 6.14 A) shows that the PSO-RRR2 also leads to a perfectly cyclic behaviour for $w=1$, whose amplitude is higher than that in Fig. 6.4 A). Although the *period* is greater (10 time-steps), the *pseudo frequency* is higher. That is to say, the attractor is overflown a higher number of times.

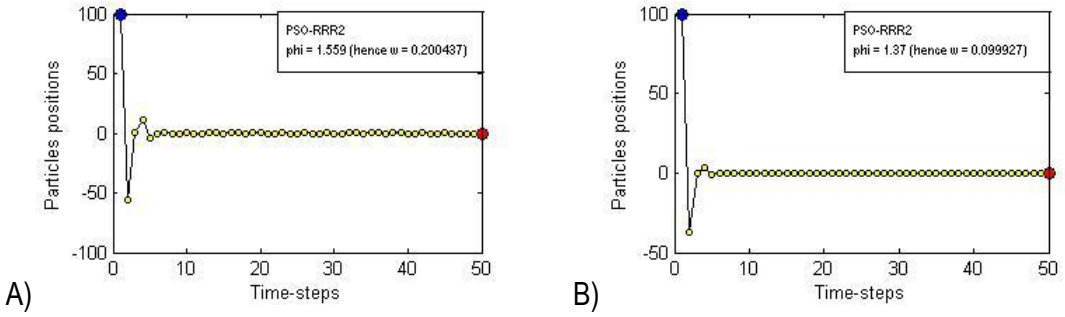


Fig. 6.15. Trajectory corresponding to the average behaviour of a deterministic particle initialized at x = 100, flying over a 1-dimensional space, with stationary attractors at x = 0, for w = 0.20 (left) and w = 0.10 (right), for the PSO-RRR2.

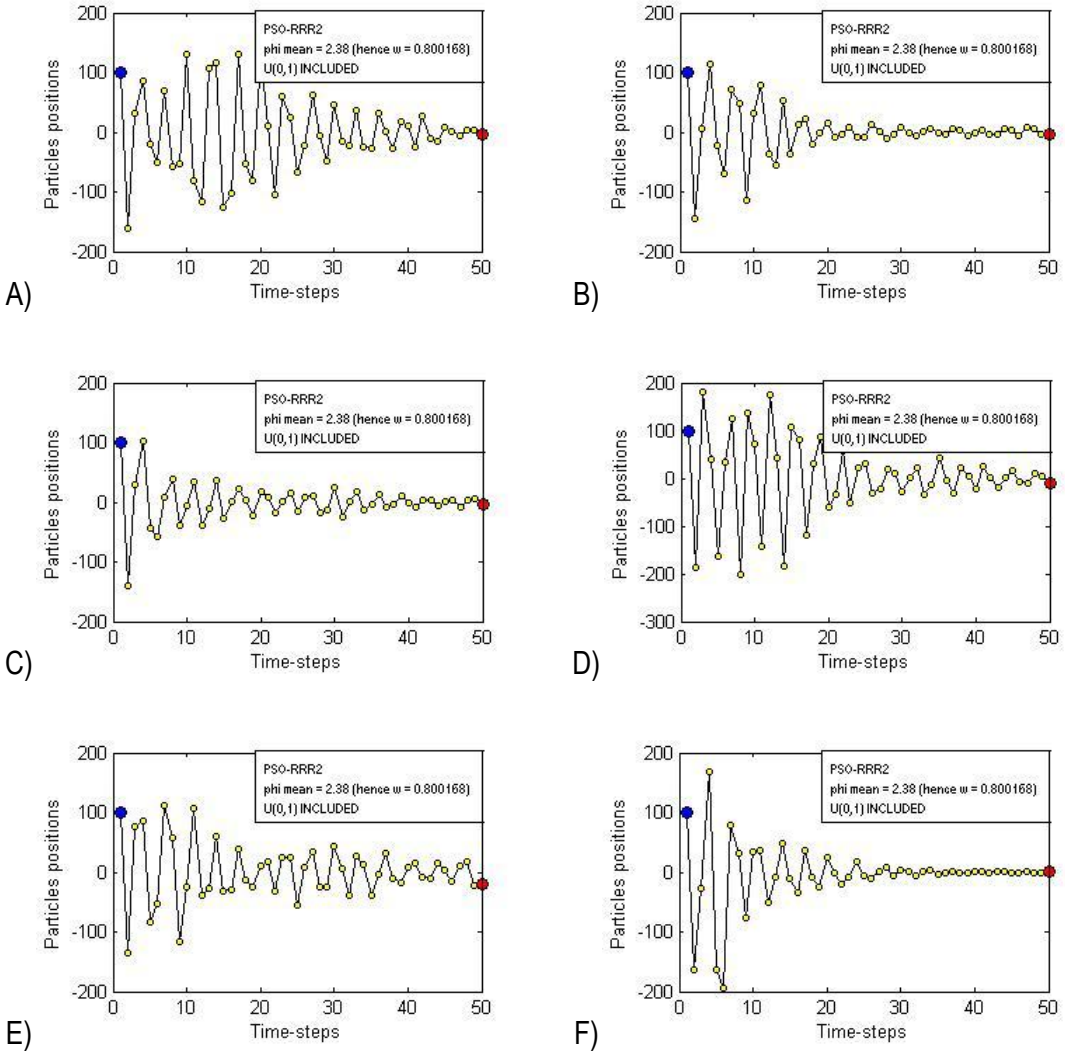


Fig. 6.16. Trajectory of a particle initialized at x = 100, flying over a 1-dimensional space, with stationary attractors at x = 0, for w = 0.80, corresponding to six consecutive runs, for the PSO-RRR2.

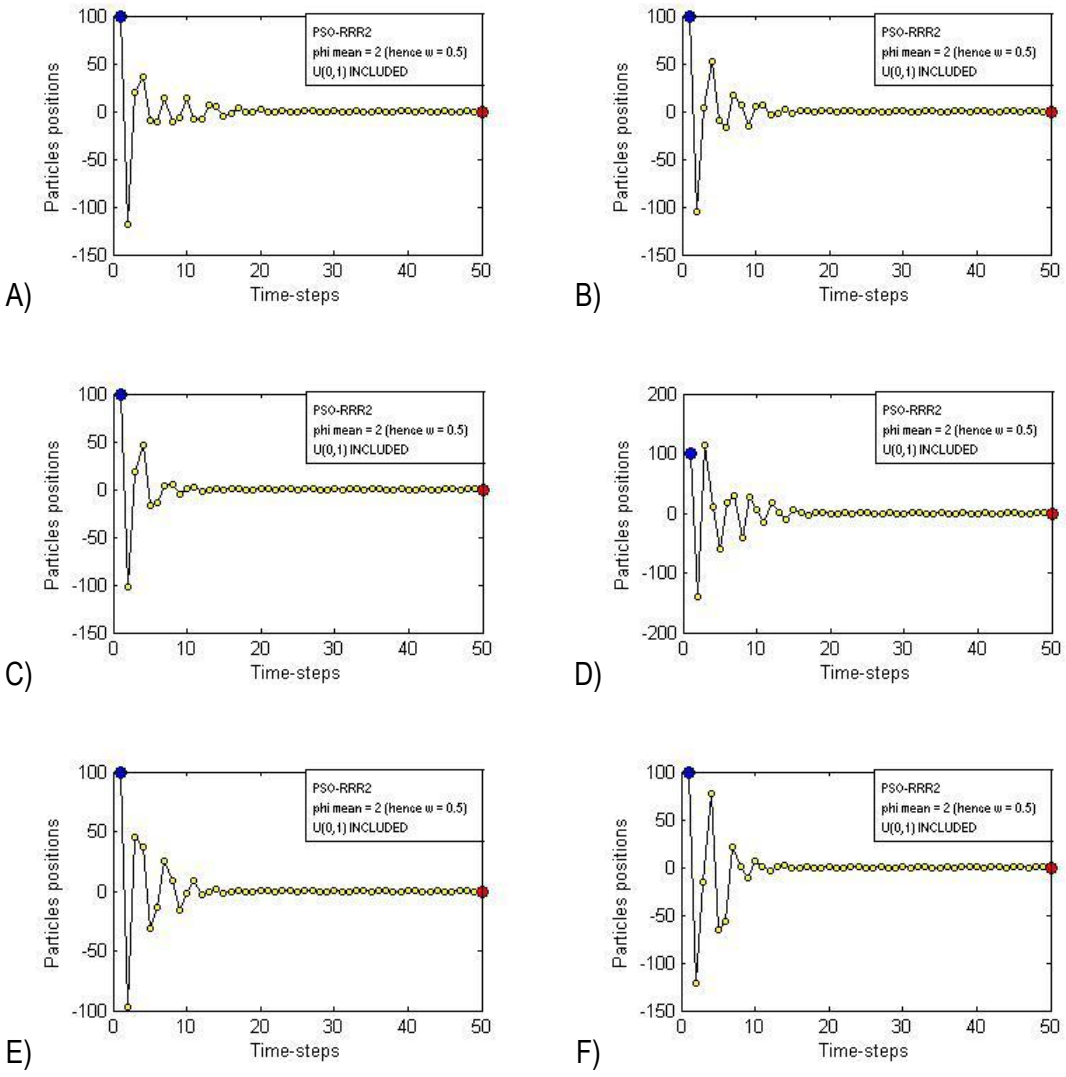


Fig. 6.17. Trajectory of a particle initialized at x = 100, flying over a 1-dimensional space, with stationary attractors at x = 0, for w = 0.50, corresponding to six consecutive runs, for the PSO-RRR2.

Notice that the deterministic particle reaches the attractor at the third time-step, as it was imposed in Eq. (5.94), while the momentum keeps the oscillations going. Also note that the three approaches, namely the PSO-RRM, the PSO-RRR1, and the PSO-RRR2 converge for $w = 0$, for which $\phi_{\text{mean}} = 1$ (see Fig. 6.6).

The same as for the PSO-RRR1, here convergence occurs in every one of the six runs, even for an inertia weight as high as '0.80' (see Fig. 6.16). Trajectories from different runs of the PSO-RRR2 are however more dissimilar from one another than those of the PSO-RRR1, but more similar to each other than those of the PSO-RRM. This is consistent with the fact that the range of ϕ and ϕ_{mean} are greater for the PSO-RRR2 than for the

PSO-RRR1; while the range of ϕ is smaller for the PSO-RRR2 than for the PSO-RRM. The same trend is observed for $w \cong 0.50$ in Fig. 6.17, and for $w \cong 0.20$ in Fig. 6.18.

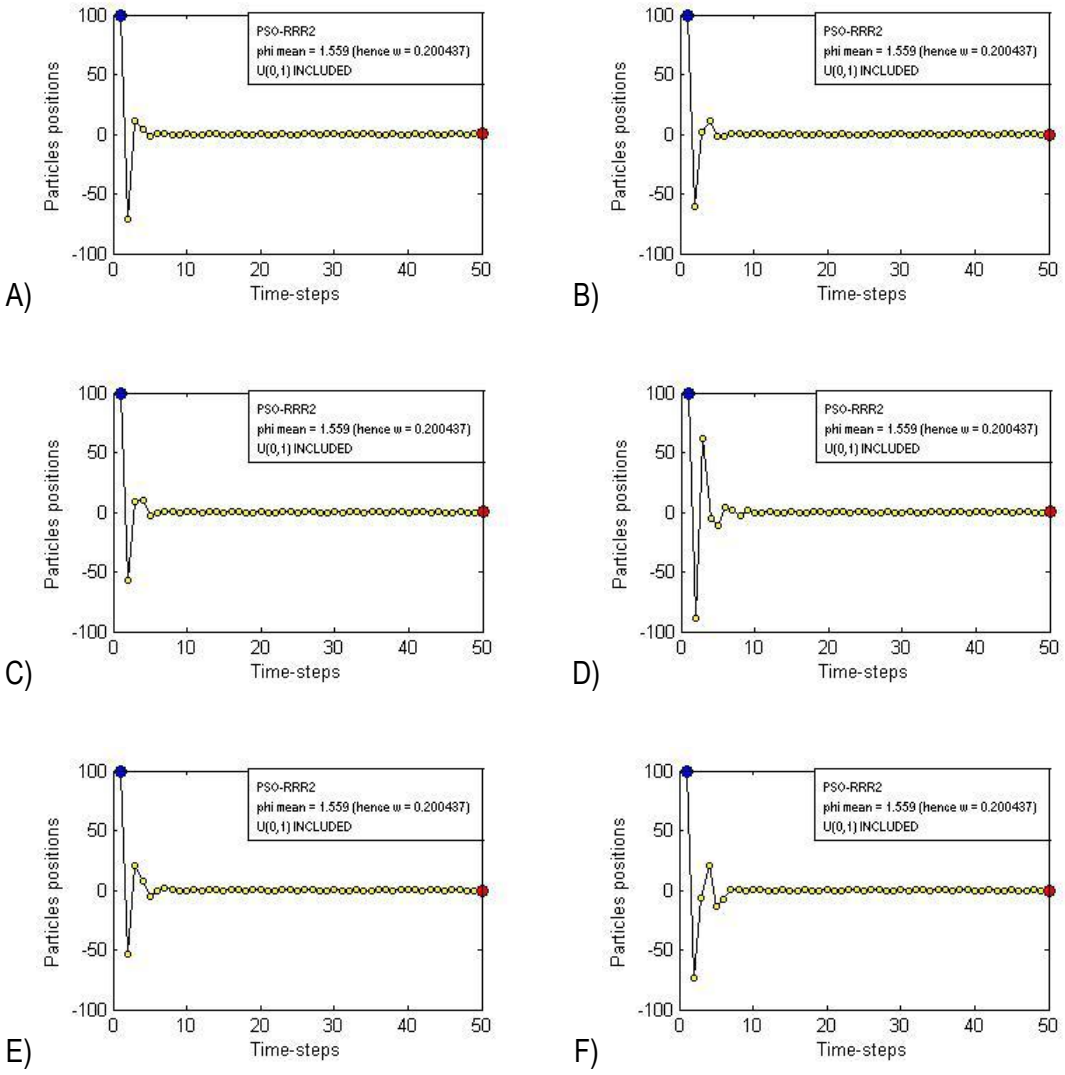


Fig. 6.18. Trajectory of a particle initialized at x = 100, flying over a 1-dimensional space, with stationary attractors at x = 0, for w = 0.20, corresponding to six consecutive runs, for the PSO-RRR2.

Thus, the main differences between the PSO-RRR1 and the PSO-RRR2 are that the latter shows trajectories with marginally higher amplitudes and pseudo frequencies[1], as well as higher deviations between trajectories corresponding to different runs. They both lead to convergence even for high values of the inertia weight. Trajectories corresponding to numerous other coefficients' settings and randomness included for the PSO-RRM, the PSO-RRR1, and the PSO-RRR2 can be found in the digital appendix.

[1] Recall that by higher pseudo frequencies it is simply meant that the attractor is overflown more frequently.

6.2.3. Updating attractor

The influence of the coefficients' settings on the trajectory of the isolated particle with stationary attractors with and without randomness has been extensively studied in chapter 5 and in the previous sections of this chapter. The next step towards the complexity of the full algorithm consists of making the attractor non-stationary.

In order to disregard the particles' interactions, the social attractor (**lbest**) is still kept stationary thus allowing the analysis of the trajectory of an individual particle. In turn, the individual attractor (**pbest**) is dynamic, therefore making the overall attractor also dynamic. Recall that each particle can be thought of as pulled by a single attractor consisting of a randomly weighted average of **lbest** and **pbest**, where the strength of the attraction is also randomly weighted.

It would seem reasonable to expect that making **pbest** the actual individual best experience of the particle instead of forcing it to take its ultimate location from the beginning would delay convergence as **pbest** gradually approaches **lbest**. However, if **pbest** and the particle's position are located at the same side from **lbest** –i.e. **lbest** is not located between them– then the momentum generated is actually smaller than it would be in the case of both attractors fixed to the actual solution. This is not far from what happens in the full system, as each particle tends to be most of the time closer to its **pbest** than to its **lbest**. Hence making **pbest** dynamic does not necessarily delay convergence.

Some of the variations to the basic PSO algorithm discussed before are the *classical PSO* (PSO), the *constricted PSO* (C-PSO), the *PSO with reduced randomness maximum* (PSO-RRM), the *PSO with reduced randomness range 1* (PSO-RRR1), and the *PSO with reduced randomness range 2* (PSO-RRR2). A couple of coefficients' settings for each of these variations are used in the experiments that follow, one favouring exploration and the other favouring non-extreme exploitation. The trajectories corresponding to six consecutive runs of an isolated particle optimizing the one-dimensional *sphere function*, attracted towards a stationary **lbest** and a dynamic **pbest**, with random weights included, for a couple of coefficients' settings for each of the five aforementioned variations of the PSO algorithm, are shown in Fig. 6.19 to Fig. 6.28. In these figures, different rows show different consecutive runs for the same settings, while different columns

correspond to different settings. Two trajectories sharing the same row share the same initial state of the pseudo random number generator.

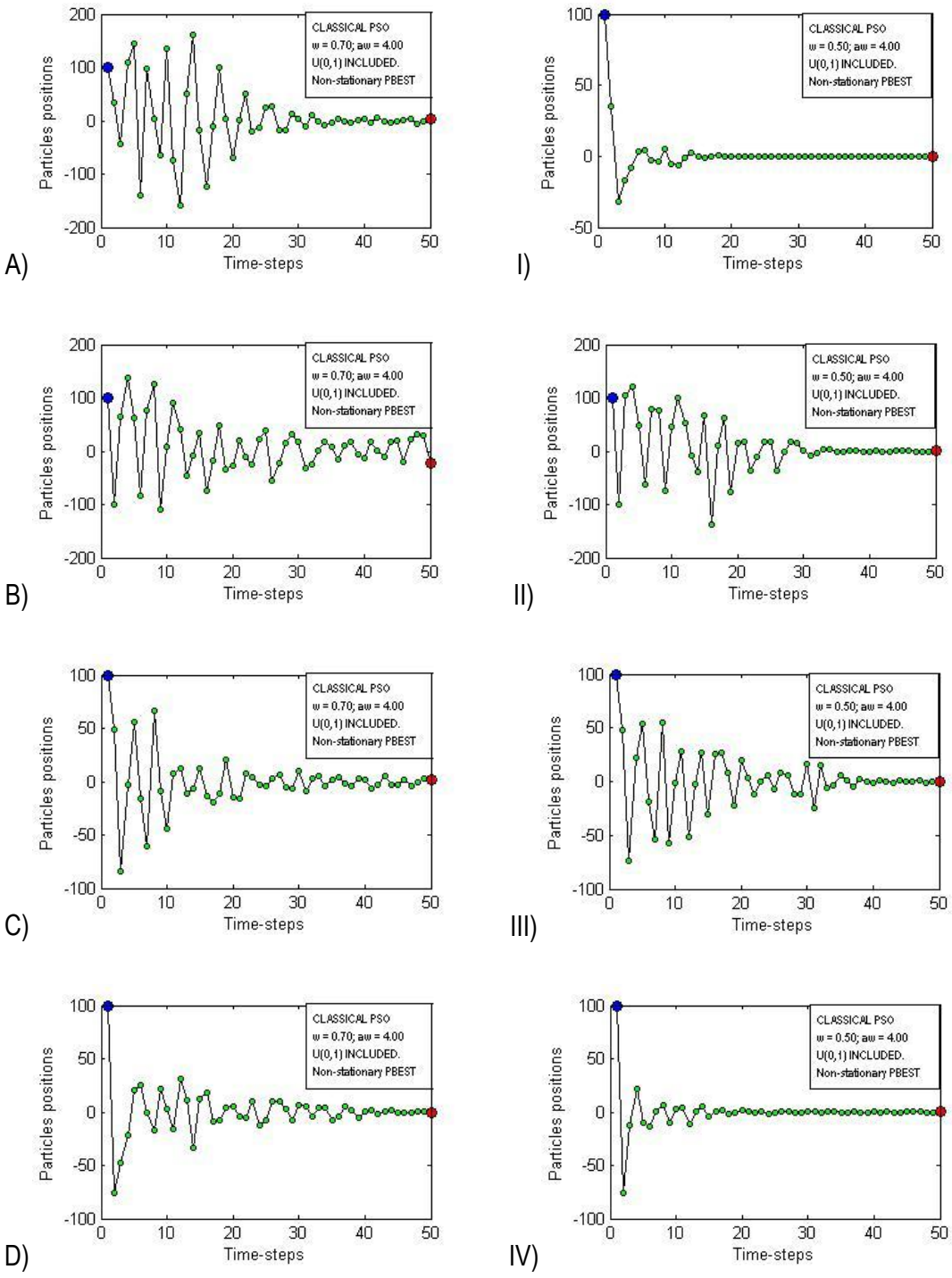


Fig. 6.19. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.70 (left column) and w = 0.50 (right column), aw = 4.00, for the classical PSO, corresponding to four consecutive runs. Same row indicates same initial state of the pseudo random number generator.

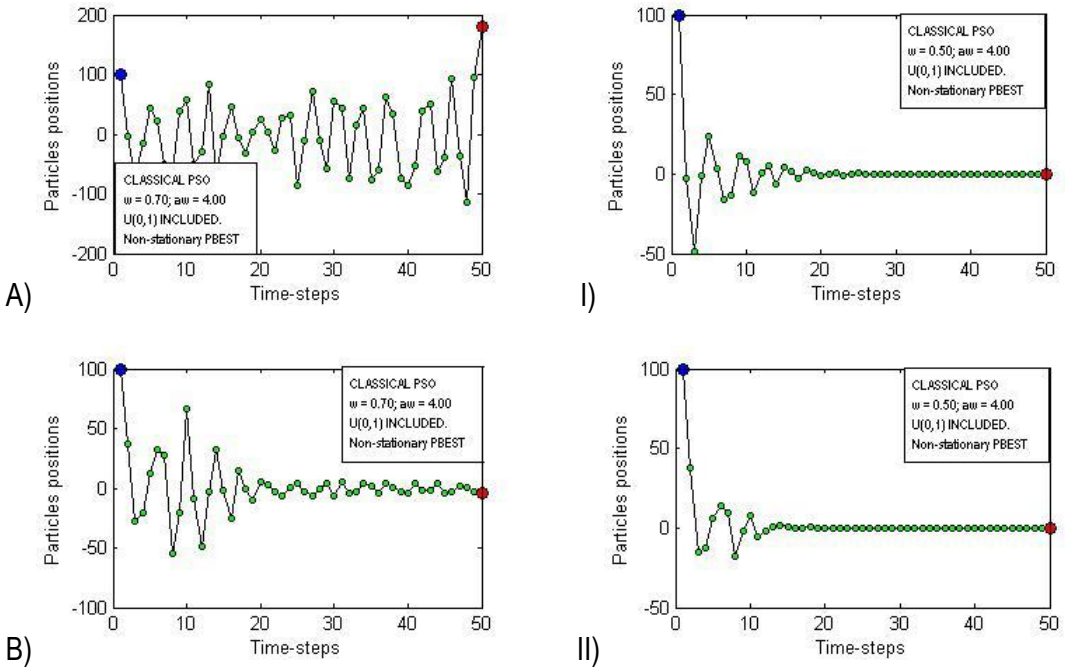


Fig. 6.20. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.70 (left column) and w = 0.50 (right column), aw = 4.00, for the classical PSO, corresponding to two runs consecutive to those in Fig. 6.19. Same row indicates same initial state of the pseudo random number generator.

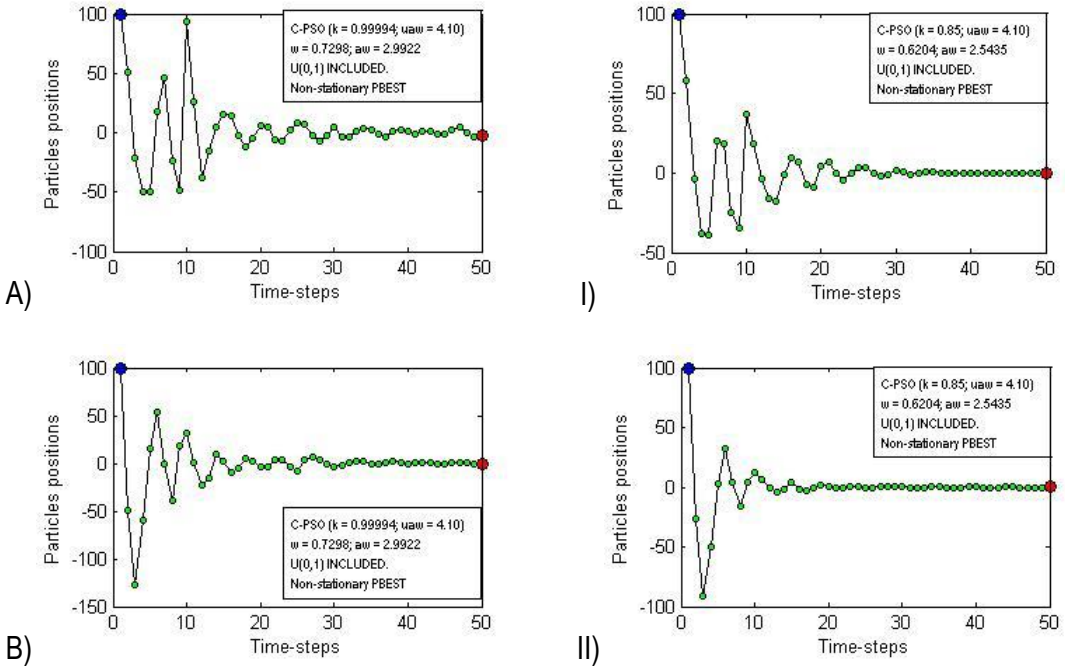


Fig. 6.21. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.7298 and aw = 2.9922 (left column), and for w = 0.6204 and aw = 2.5435 (right column), for the C-PSO, corresponding to two consecutive runs. Same row indicates same initial state of the pseudo random number generator.

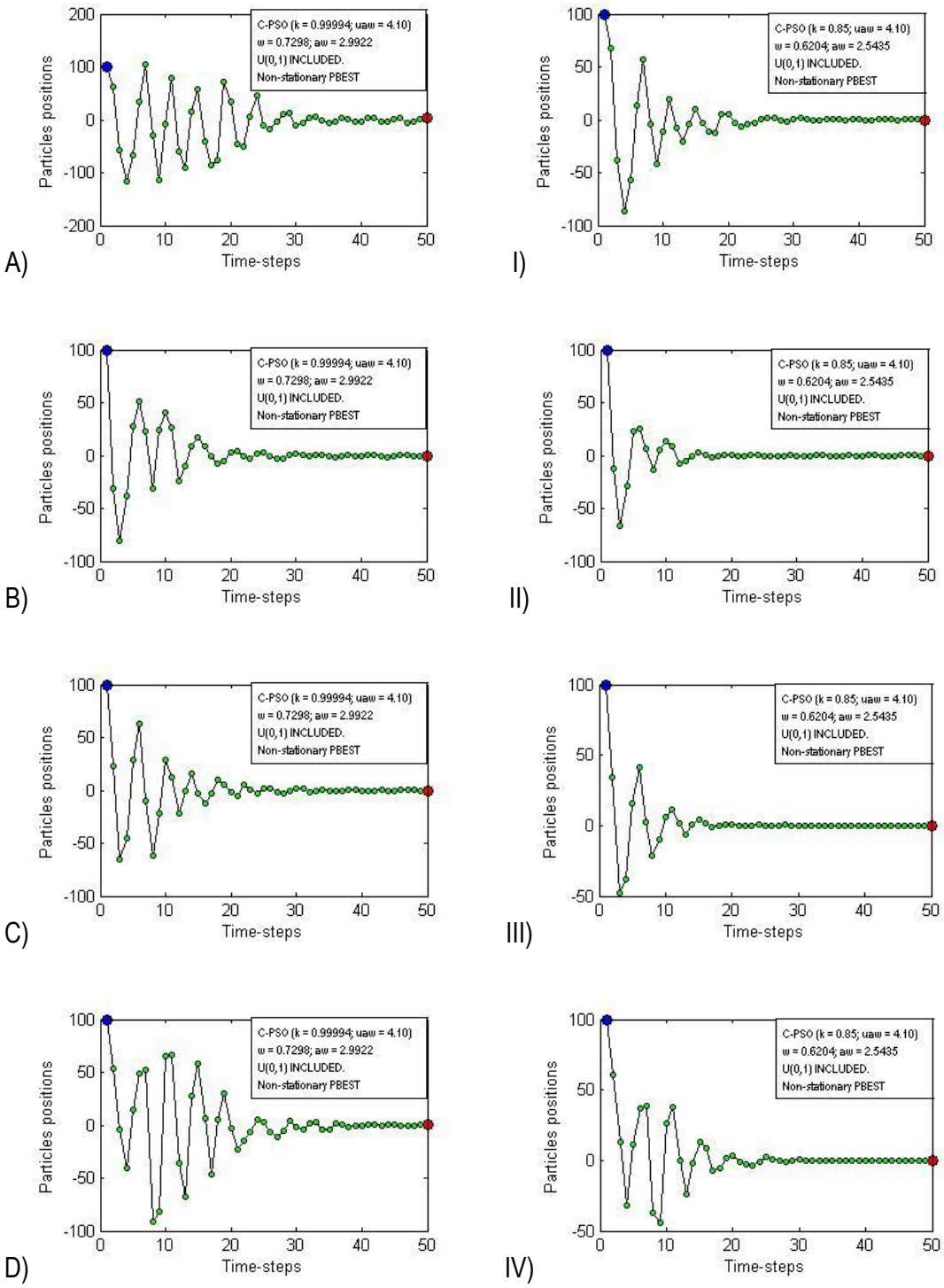


Fig. 6.22. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.7298 and aw = 2.9922 (left column), and for w = 0.6204 and aw = 2.5435 (right column), for the C-PSO, corresponding to four runs consecutive to those in Fig. 6.21. Same row indicates same initial state of the pseudo random number generator.

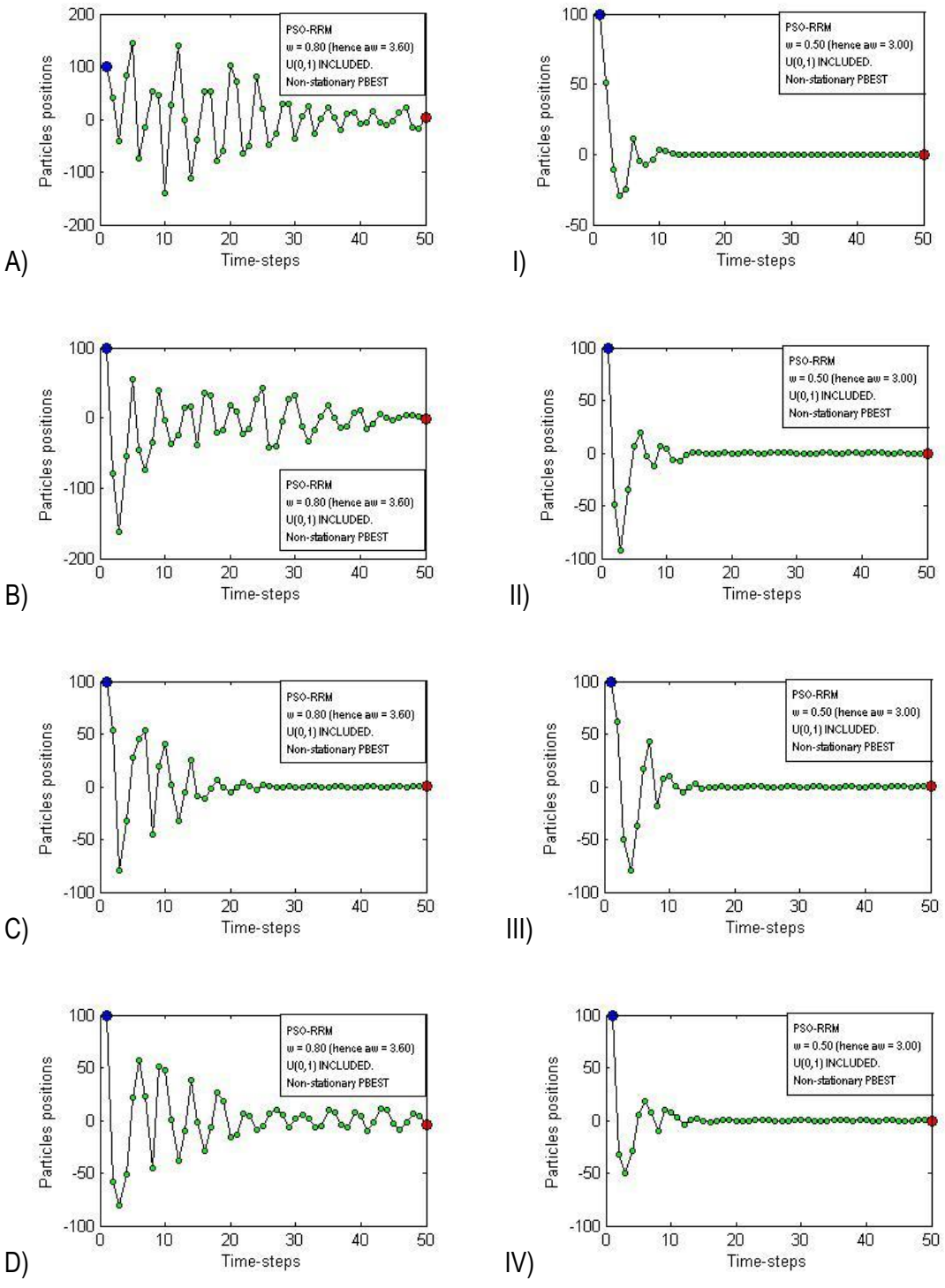


Fig. 6.23. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.80 and hence aw = 3.60 (left column), and for w = 0.50 and hence aw = 3.00 (right column), for the PSO-RRM, corresponding to four consecutive runs. Same row indicates same initial state of the pseudo random number generator.

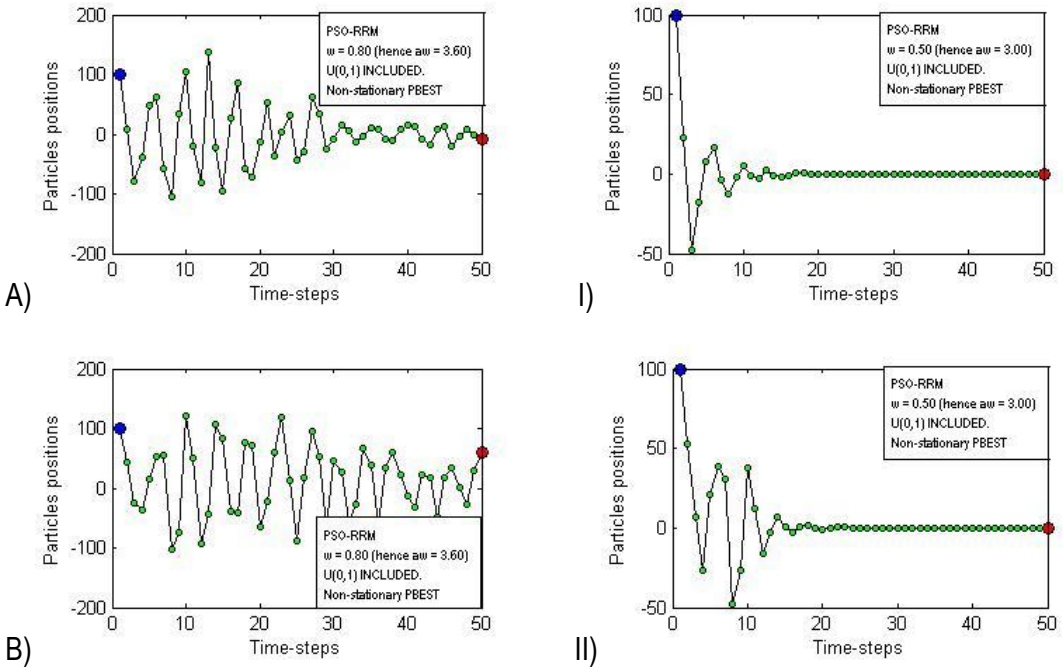


Fig. 6.24. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.80 and hence aw = 3.60 (left column), and for w = 0.50 and hence aw = 3.00 (right column), for the PSO-RRM, corresponding to two runs consecutive to those in Fig. 6.23. Same row indicates same initial state of the pseudo random number generator.

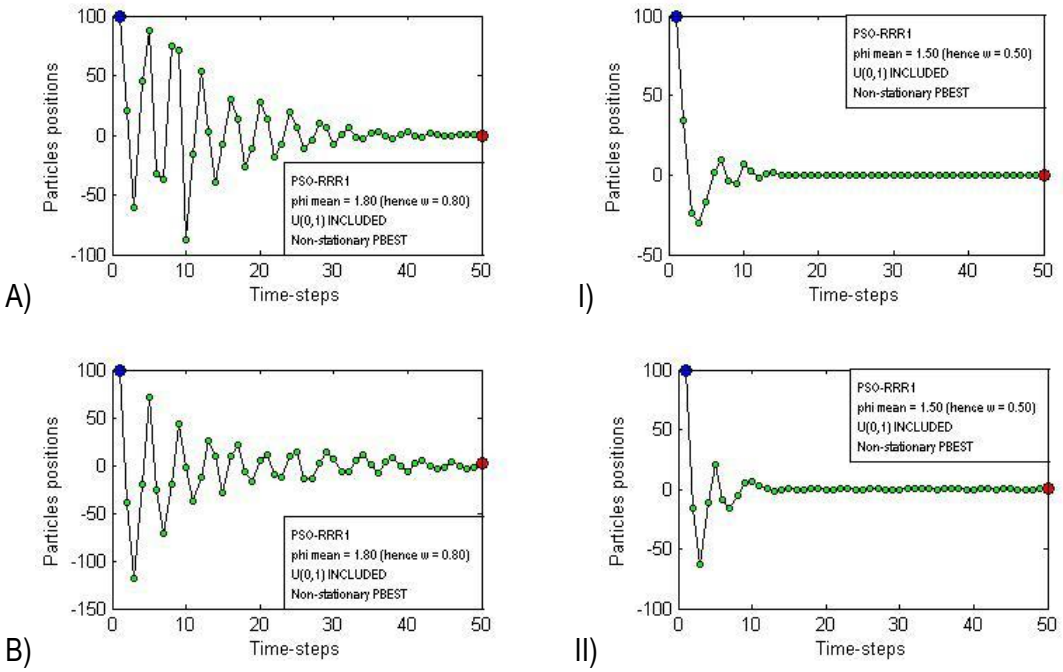


Fig. 6.25. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.80 and hence w = 0.80 (left column), and for ϕ_{mean} = 1.50 and hence w = 0.50 (right column), for the PSO-RRR1, corresponding to two consecutive runs. Same row indicates same initial state of the pseudo random number generator.

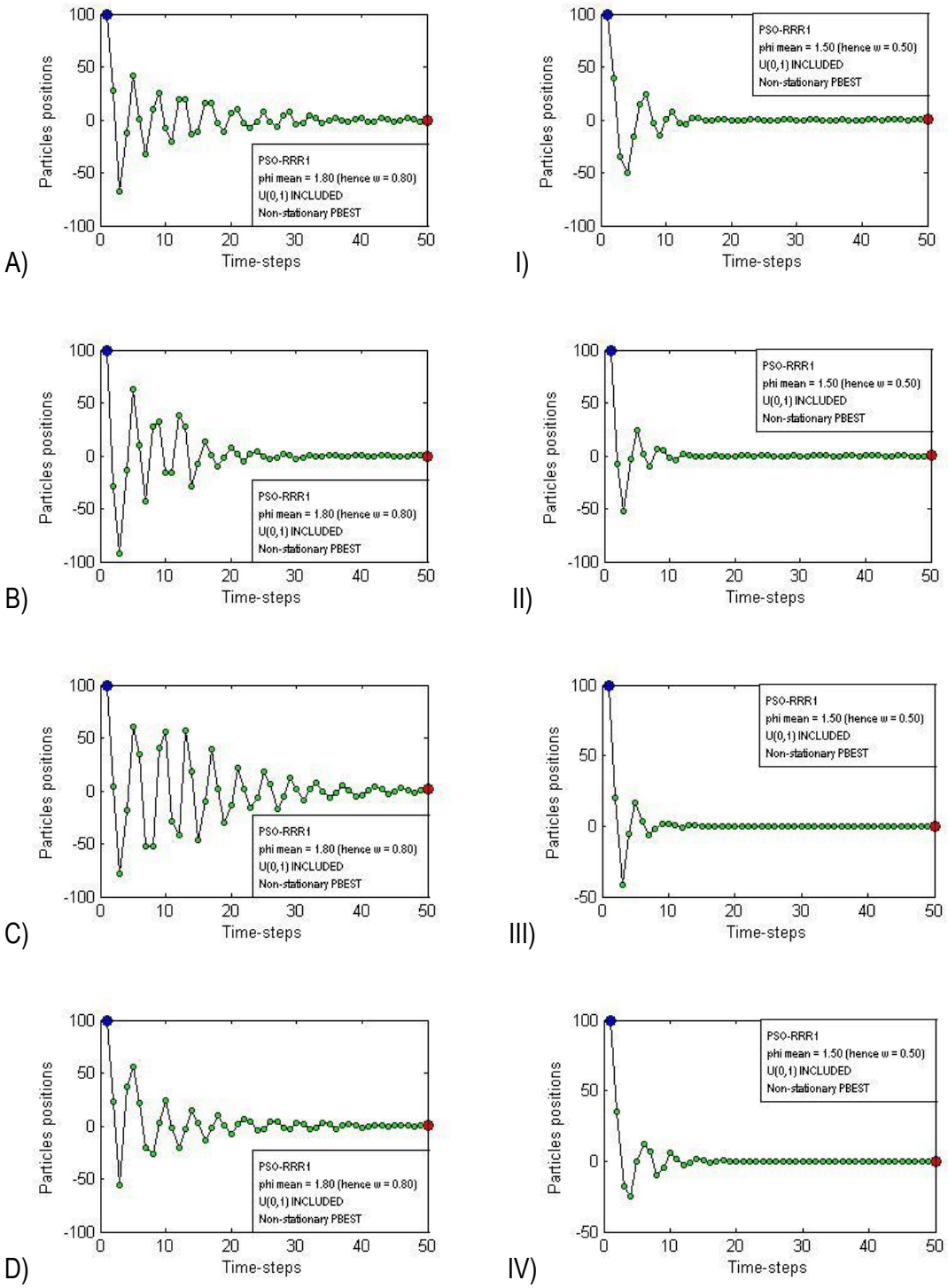


Fig. 6.26. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.80 and hence w = 0.80 (left column), and for ϕ_{mean} = 1.50 and hence w = 0.50 (right column), for the PSO-RRR1, corresponding to four runs consecutive to those in Fig. 6.25. Same row indicates same initial state of the pseudo random number generator.

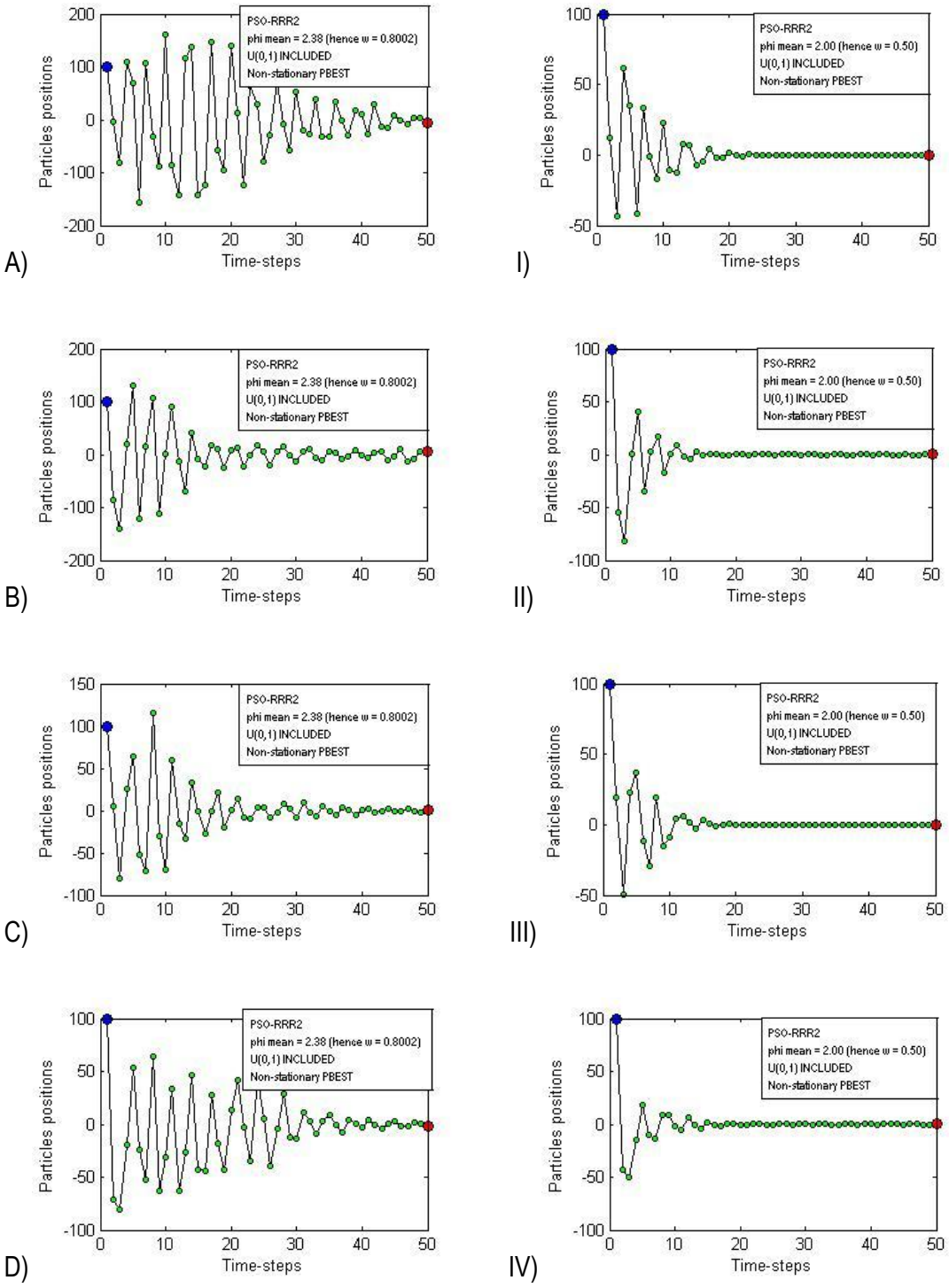


Fig. 6.27. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 2.38 and hence w = 0.80 (left column), and for ϕ_{mean} = 2.00 and hence w = 0.50 (right column), for the PSO-RRR2, corresponding to four consecutive runs. Same row indicates same initial state of the pseudo random number generator.

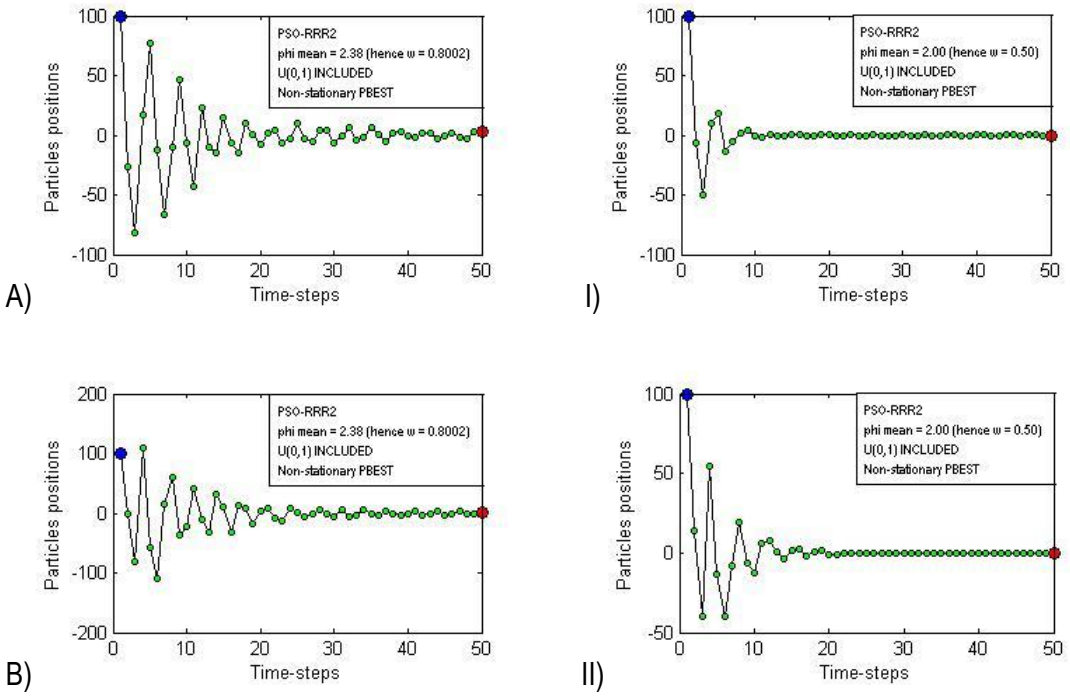


Fig. 6.28. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 2.38 and hence w = 0.80 (left column), and for ϕ_{mean} = 2.00 and hence w = 0.50 (right column), for the PSO-RRR2, corresponding to two runs consecutive to those in Fig. 6.27. Same row indicates same initial state of the pseudo random number generator.

Fig. 6.19 to Fig. 6.28 show that making **pbest** dynamic does not change the general trend of the trajectories that result from the coefficients' settings.

The first aspect to consider is the ***regularity*** of the trajectories. In general, there is a higher deviation between different runs corresponding to the same settings when compared to the trajectories with both attractors stationary. This is most noticeable for the classic PSO (e.g. compare Fig. 6.19 and Fig. 6.20 –right column– to Fig. 5.53). Thus, the order from most to least regular approaches would be as follows: PSO-RRR1; PSO-RRR2 and C-PSO; PSO-RRM; classic PSO. Within the same family, the lower the *aw* the higher the regularity –as expected–, since the influence of randomness decreases.

Another aspect is the ***amplitude*** of the oscillations. Given that **pbest** is initialized near the initial position of the particle, the initial oscillations show smaller amplitudes, although the general speed of convergence does not change noticeably. For the same both w and ϕ_{mean}, PSO-RRM exhibit higher amplitudes than the PSO-RRR1; while for the same w, the amplitudes showed by the PSO-RRR2 are in between.

Except for the completely flexible classical PSO, the lower the *w* the lower the *aw*. Thus, within the same family, the lower the *w* (or the κ) –and hence the lower the *aw*– the smaller the amplitudes and the faster the convergence.

Another important aspect of the oscillatory behaviour is the ***frequency*** with which the particle overflies **lbest**. Thus, for instance, even though the amplitudes exhibited by the PSO-RRR2 are greater than those shown by the PSO-RRR1, the (pseudo) frequency exhibited by the former is greater. The order of the approaches from greatest to smallest frequencies would be as follows: PSO-RRR2; classic PSO with *aw* beyond the convergence area[2]; PSO-RRM and PSO-RRR1; C-PSO. Notice that the pair 'ϕ_{mean}–*w*' associated to the C-PSO (*type 1"*) with popular settings (*aw* marginally greater than '4' and κ marginally smaller than '1') are located to the left of the line of the average behaviour of the PSO-RRM and the PSO-RRR1 (refer to Fig. 6.2 and Fig. 6.3), which in turn are to the left of the line of the average behaviour of the PSO-RRR2 (see Fig. 6.13). This implies a lower *aw* for a given *w*, which results in lower (pseudo) frequencies.

Fig. 6.19 to Fig. 6.28 show the trajectories of a particle optimizing the **Sphere** function, where **pbest** is updated according to the particles' experiences. Exactly the same experiments, with the same settings and initial states of the pseudo random number generator, were carried out optimizing the **Rastrigin** and the **Schaffer f6** functions. The aim was to observe whether the behaviour identified for the settings studied is maintained when the objective function is multimodal. In the case of the Rastrigin function, there exist many local optima but the trend line is the same as the Sphere function. The trajectories obtained were almost identical. The images can be found in the digital appendix. Conversely, the Schaffer f6 function presents both better and worst values as the particle gradually approaches **gbest**, making the jump from one local optimum to a better one more difficult. It is fair to remark that the difficulty increases with the dimensionality of the function, as the valley-like local optima turn into (hyper) ring-like depressions (i.e. the local optima are *isolines* rather than points). The trajectories corresponding to the PSO-RRR1 and the PSO-RRR2 optimizing the Schaffer f6 function are offered in Fig. 6.29 to Fig. 6.32 for the same settings used to optimize the Sphere function in Fig. 6.25 to Fig. 6.28.

[2] Recall that the classical PSO is completely flexible, and any behaviour can be exhibited depending on the settings of the coefficients.

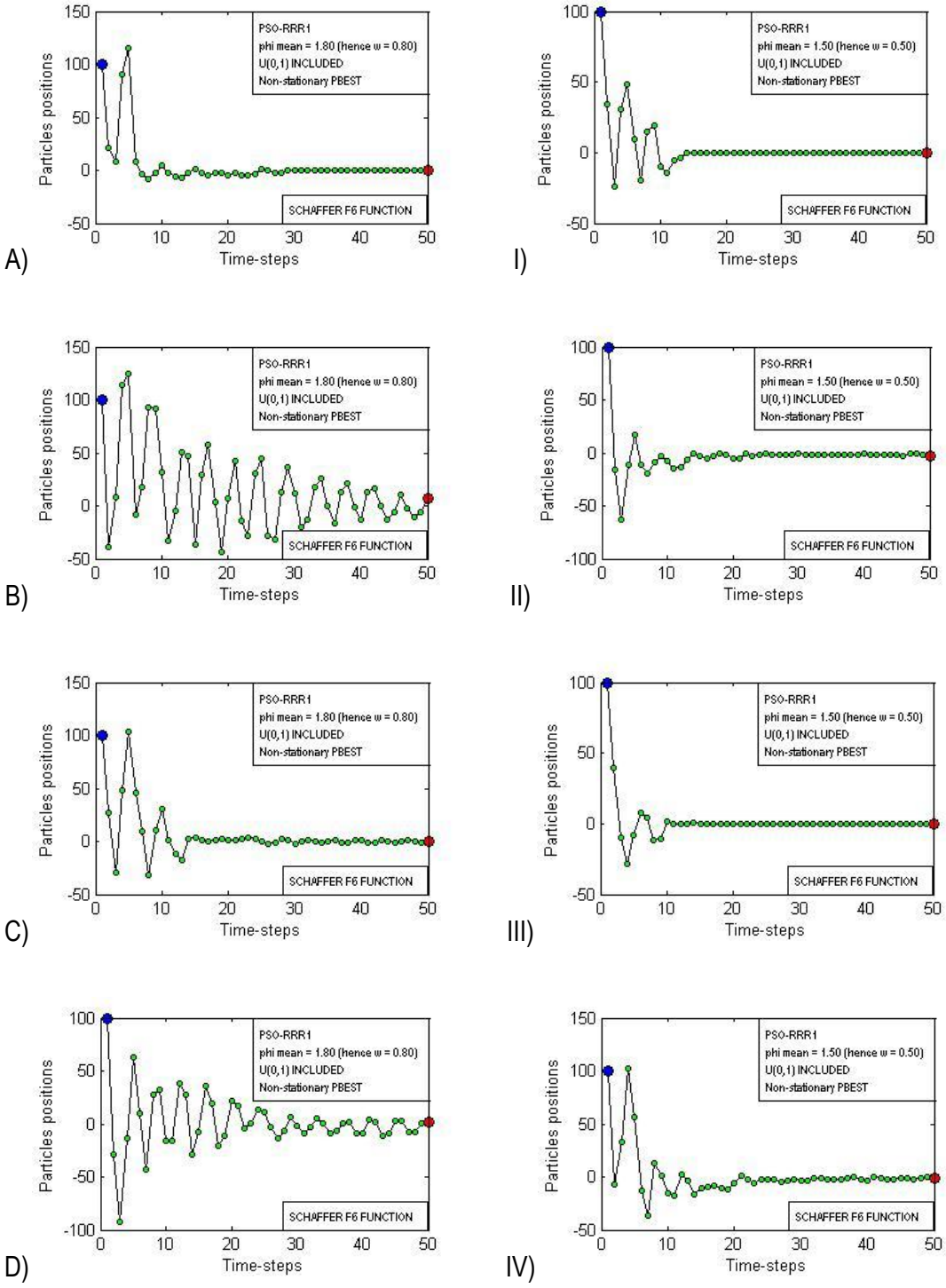


Fig. 6.29. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Schaffer f6 function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.80 and hence w = 0.80 (left column), and for ϕ_{mean} = 1.50 and hence w = 0.50 (right column), for the PSO-RRR1, corresponding to four consecutive runs. Same row indicates same initial state of the pseudo random number generator.

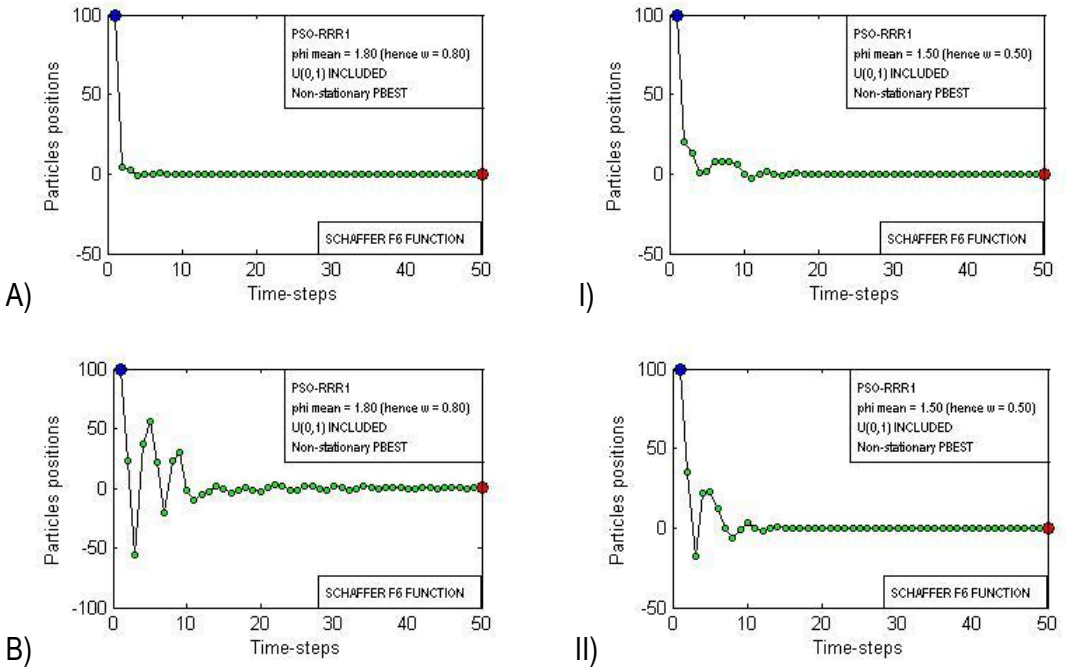


Fig. 6.30. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Schaffer f6 function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.80 and hence w = 0.80 (left column), and for ϕ_{mean} = 1.50 and hence w = 0.50 (right column), for the PSO-RRR1, corresponding to two runs consecutive to those in Fig. 6.29. Same row indicates same initial state of the pseudo random number generator.

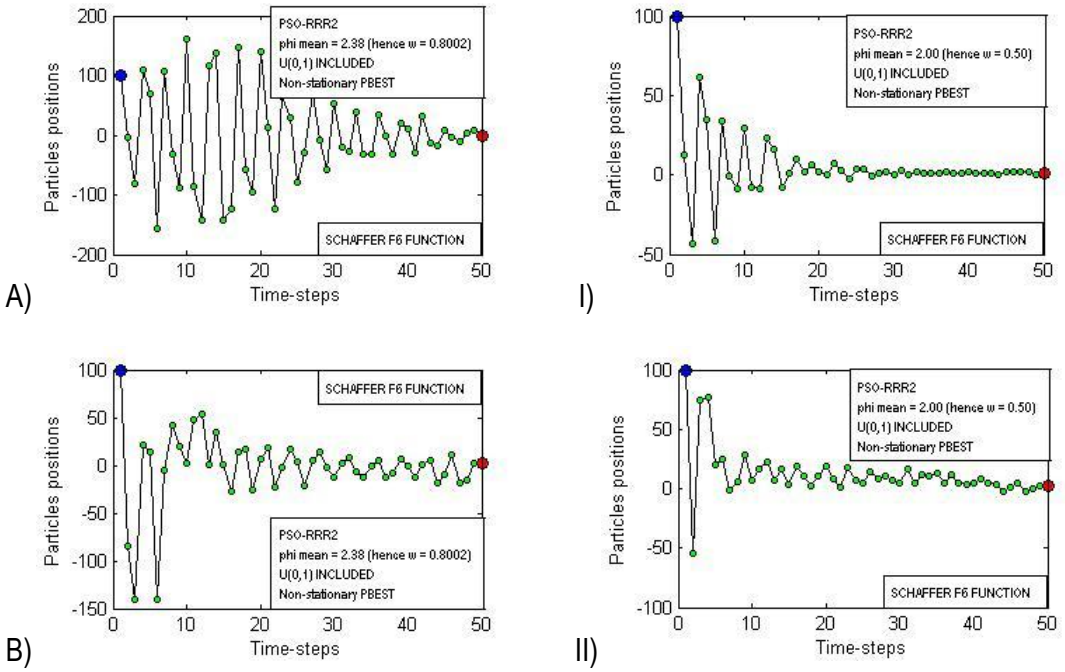


Fig. 6.31. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Schaffer f6 function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 2.38 and hence w = 0.80 (left column), and for ϕ_{mean} = 2.00 and hence w = 0.50 (right column), for the PSO-RRR2, corresponding to two consecutive runs. Same row indicates same initial state of the pseudo random number generator.

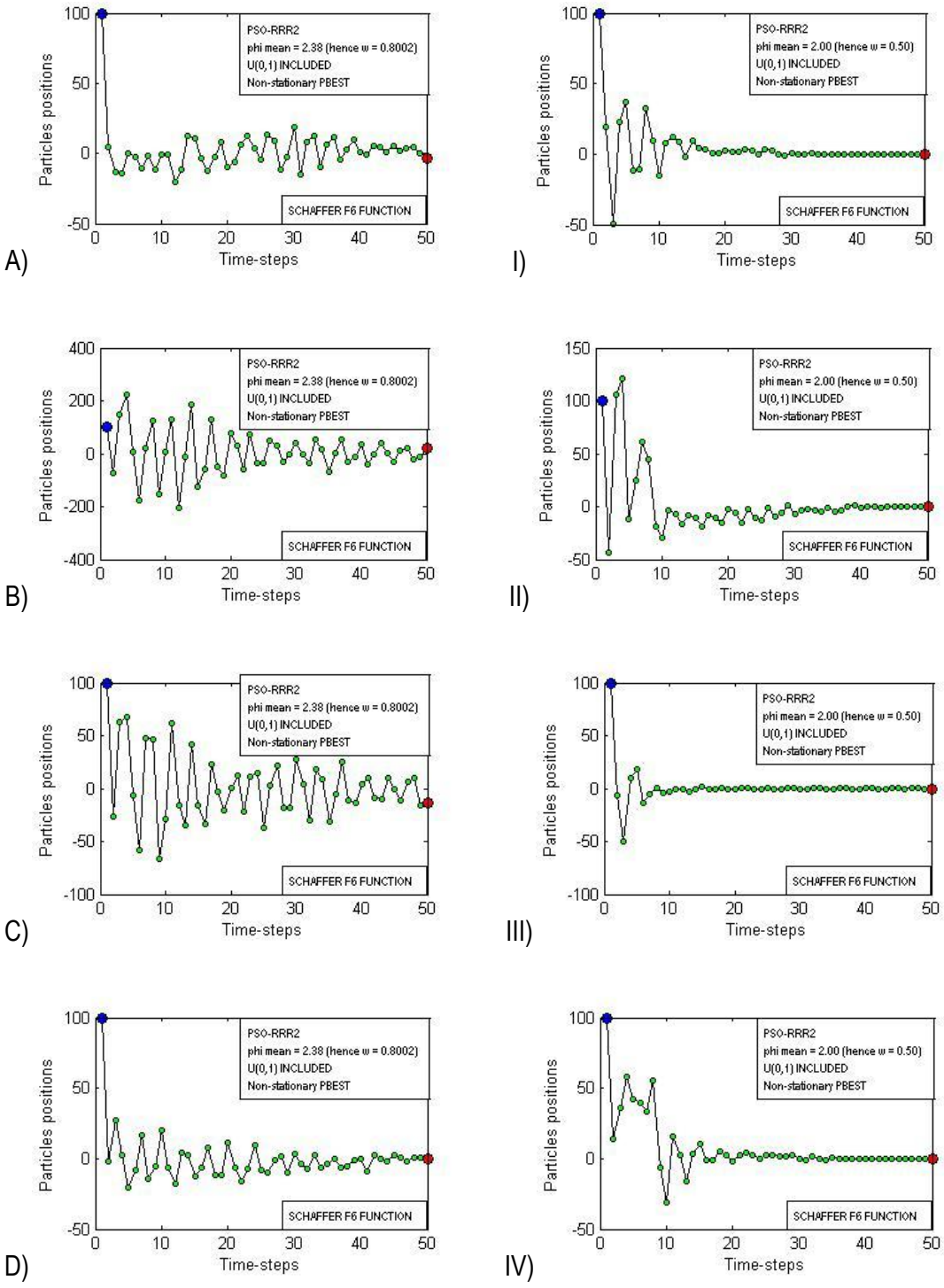


Fig. 6.32. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Schaffer f6 function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 2.38 and hence w = 0.80 (left column), and for ϕ_{mean} = 2.00 and hence w = 0.50 (right column), for the PSO-RRR2, corresponding to four runs consecutive to those in Fig. 6.31. Same row indicates same initial state of the pseudo random number generator.

As can be observed, the regularity is affected when optimizing this peculiar function, and trajectories show higher deviations for different runs using the same settings. Nevertheless, the trend is still the same and, broadly speaking, the higher the coefficients the more explorative the behaviour, while every run ends up converging. It is interesting to note that the PSO-RRR1 –which exhibit lower pseudo frequencies than the PSO-RRR2– is unable to successively overfly **gbest** is some runs. In that sense, the PSO-RRR2 shows more desirable behaviour. The trajectories obtained by the classical PSO, the C-PSO, and the PSO-RRM –with the same settings as in Fig. 6.19 to Fig. 6.24– optimizing the Schaffer f6 function can be found in the digital appendix.

6.2.4. Individuality and sociality

In all the previous studies, as well as in those that –to the best of my knowledge– can be found in the literature, the analyses of the acceleration weight is performed as a whole, without getting into the relative settings of the individuality and the sociality weights. Even when they are separated in some formulations –e.g. (Ozcan & Mohan, 1999); (van den Bergh, 2001)–, their relative settings are not studied.

In this thesis, the variables *ip* and *sp* as defined in Eqs. (6.8) and (6.10) –together with *aw*– will be used instead of *iw* and *sw*, since their meaning is more straightforward. Thus $ip + sp = 1$, where *ip* states the percentage of *aw* that is awarded to the individuality and *sp* the percentage awarded to the sociality of the particle. Therefore a particle with $ip = 1$ disregards the social experiences (completely *self-confident*) while a particle with $sp = 1$ disregards its own experiences, thus becoming completely *conformist*.

It was argued in chapter 5 that it is frequent to simply assume that the smaller the inertia weight the faster the convergence, under the argument that the search becomes more like a local search as the inertia is decreased. It has been shown in chapter 5 that the argument is not accurate, and the behaviour depends on the pair 'ϕ–w'[3].

Likewise, it was argued that one might expect that making **pbest** dynamic would delay convergence with respect to the same settings and both attractors stationary, thus result-

[3] Besides, the speed of convergence is not the only aspect that affects the performance of the algorithm. The same speed can be obtained for different amplitudes and (pseudo) frequencies.

ing in a more extensive exploration. This was shown not to be true in the previous section, as **pbest** tends to be closer to the particles' position and hence decreases the momentum, which seems to counterbalance the delay of convergence that would result from a dynamic individual attractor.

Similarly, it would be reasonable to expect that strengthening individuality would result in a more explorative behaviour of a *more self-confident* particle, as convergence would be delayed by its reluctance to conform to the social experience in detriment of its own. Again, such an intuitive statement would be wrong.

Consider, for instance, the trajectories in Fig. 6.19 and Fig. 6.20 (left columns), where $w = 0.70$, $aw = 4.00$, and $ip = sp = 0.50$. Recall that these settings lead to slow convergence. To analyze the extremal cases, the trajectories corresponding to the same settings except that $ip = 1$ (hence $sp = 0$) in the left column and $ip = 0$ (hence $sp = 1$) in the right column are offered in Fig. 6.33 and Fig. 6.34. Surprisingly, the exploration capabilities are harmed rather than improved by awarding the particle absolute self-confidence ($ip = 1$), whilst awarding it absolute conformism ($sp = 1$) results in a stochastic explosion of its trajectory. In fact, the trajectory shows premature convergence in Fig. 6.33 B), Fig. 6.33 D), and Fig. 6.34 A) despite being optimizing a very simple, unimodal function. The runs that do not show premature convergence –namely Fig. 6.33 A), Fig. 6.33 C), and Fig. 6.34 B)– are merely because the coefficients settings favour slow convergence. Therefore the same experiments are carried out for the PSO-RRR1 with $\phi_{\text{mean}} = 1.50$ ($w = 0.50$), whose settings favour fast convergence. In this case, the loss of exploration for $ip = 1$ is even more notorious, as all six runs exhibit poor exploration and premature convergence (refer to Fig. 6.35 and Fig. 6.36, left columns). This is an example of how convergence in PSO does not imply optimality in any sense but simply the complete loss of momentum. In turn, given that the PSO-RRR1 possesses fast convergence capabilities, setting $sp = 1$ does not result in any stochastic explosion. Instead, the particle behaves as it did when both attractors were stationary. In fact, the only reason why the trajectories in Fig. 6.35 and Fig. 6.36 (right columns) are not exactly the same as those in Fig. 6.9 and Fig. 6.10 (right columns) is that, although the pseudo-random number generator is set to the same initial state for the first run, the sequences of numbers generated differ. Nevertheless, the general behaviour is the same.

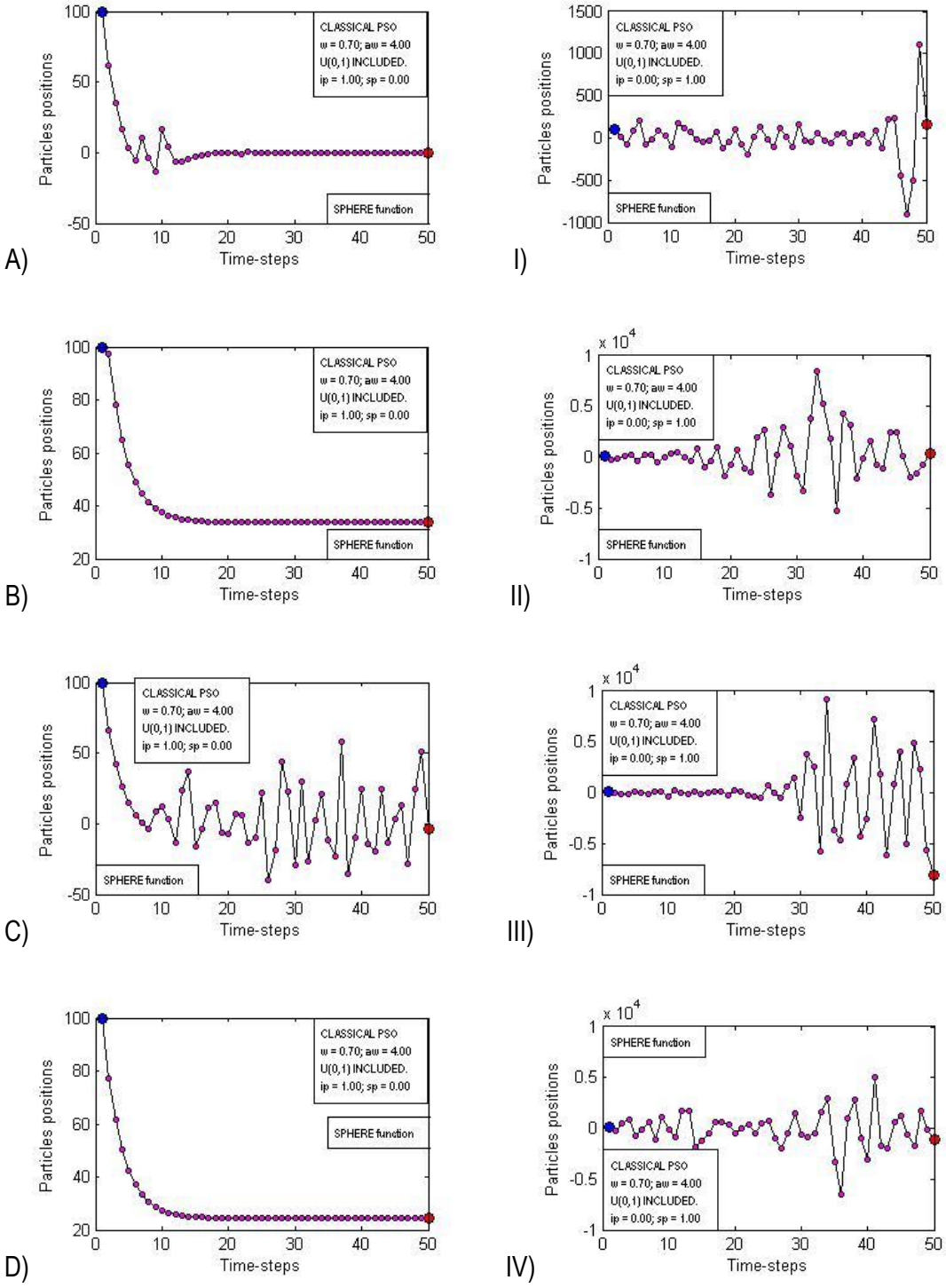


Fig. 6.33. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.70, aw = 4.00, and ip = 1.00 in the left column and ip = 0.00 in the right column, for the classical PSO, corresponding to four consecutive runs. Same row indicates same initial state of the pseudo random number generator.

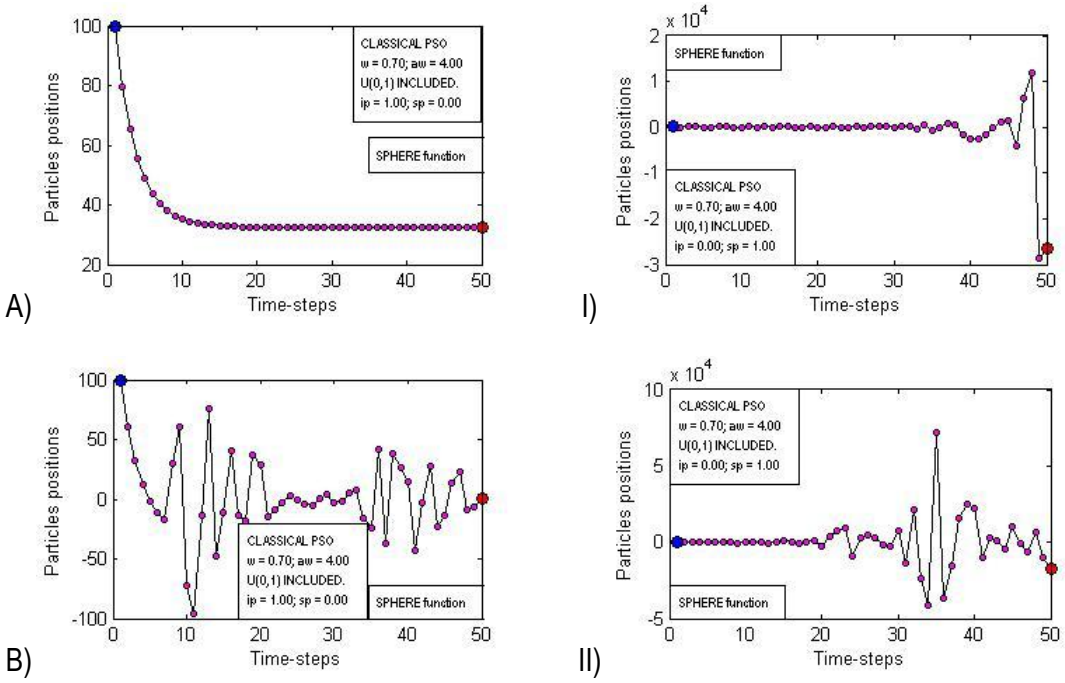


Fig. 6.34. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for w = 0.70, aw = 4.00, and ip = 1.00 in the left column and ip = 0.00 in the right column, for the classical PSO, corresponding to two runs consecutive to those in Fig. 6.33. Same row indicates same initial state of the pseudo random number generator.

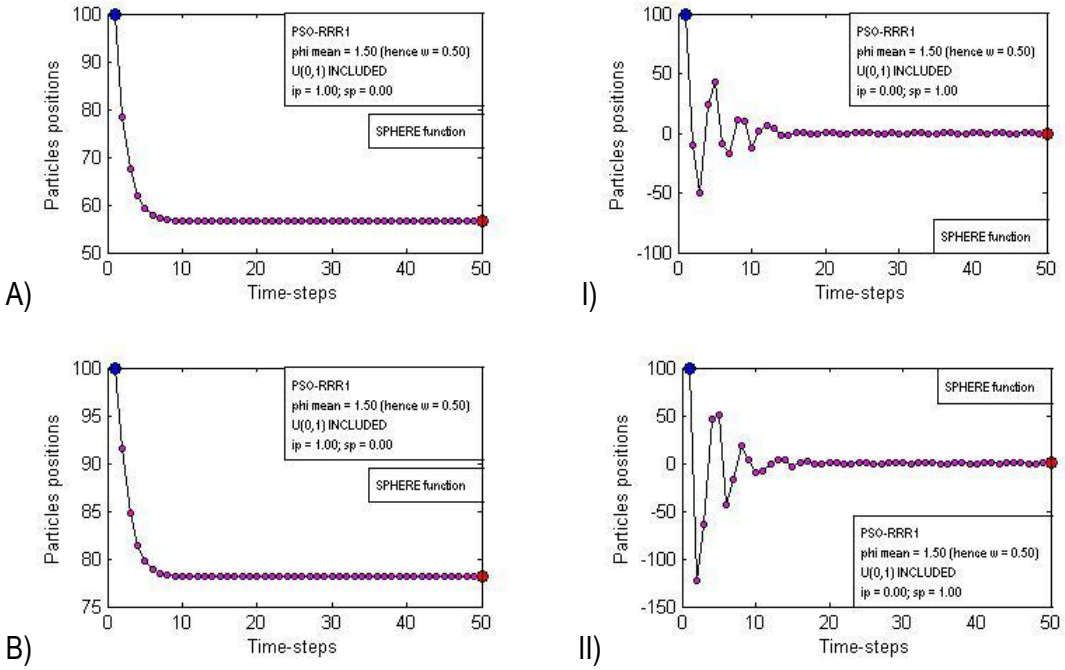


Fig. 6.35. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.50 and hence w = 0.50, and ip = 1.00 in the left column and ip = 0.00 in the right column, for the PSO-RRR1, corresponding to two consecutive runs. Same row indicates same initial state of the pseudo random number generator.

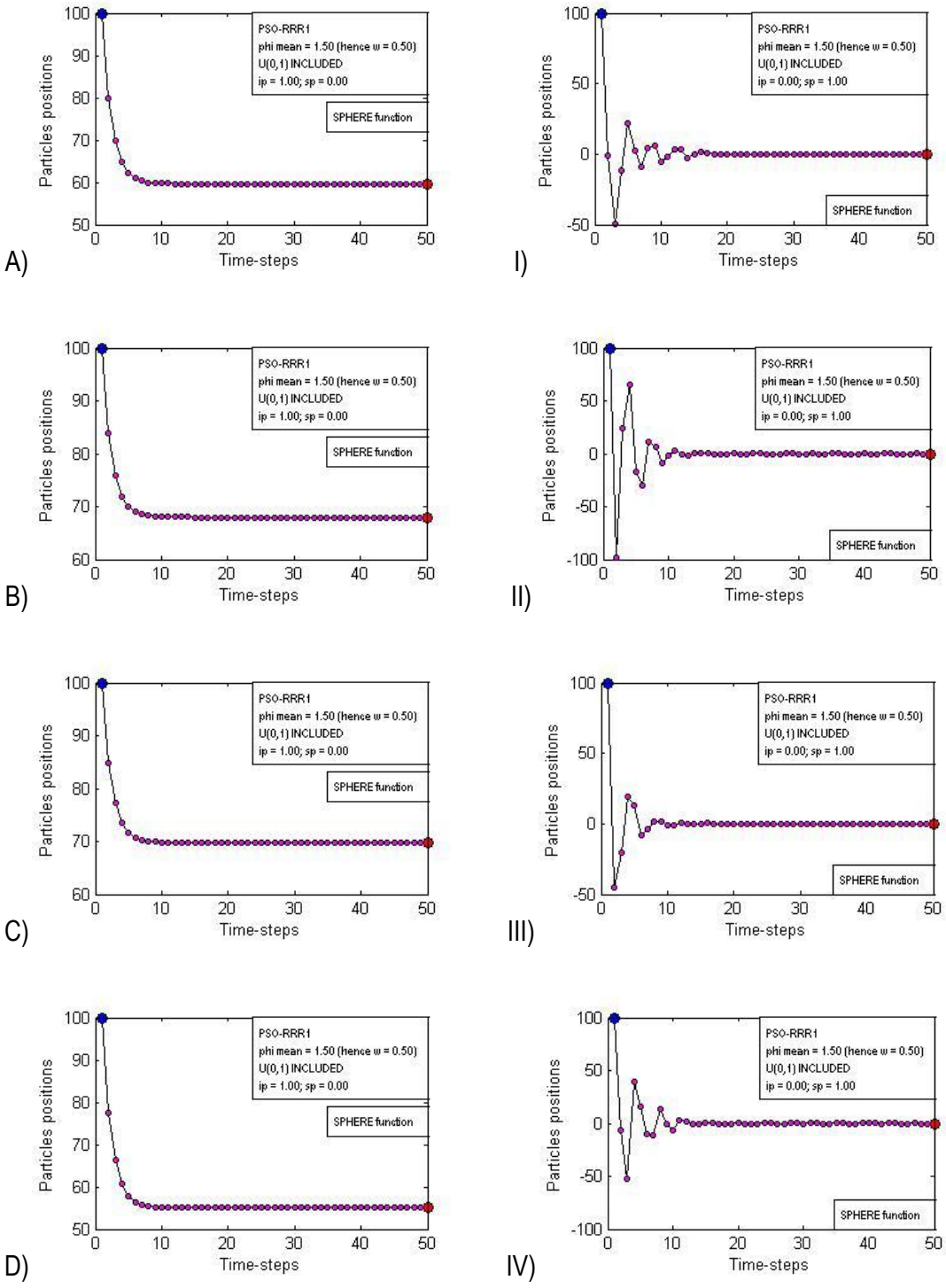


Fig. 6.36. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.50 and hence w = 0.50, and ip = 1.00 in the left column and ip = 0.00 in the right column, for the PSO-RRR1, corresponding to four runs consecutive to those in Fig. 6.35. Same row indicates same initial state of the pseudo random number generator.

If instead of comparing the trajectories in Fig. 6.35 and Fig. 6.36 (right columns) to the corresponding ones with stationary attractors they are compared to those in Fig. 6.25 and Fig. 6.26 (right columns), it can be observed that setting $sp = 1$ for the PSO-RRR1 leads to slower rather than faster convergence. Thus, the intuition that increasing the individuality of the particle would lead to more extensive exploration whilst increasing the sociality would lead to faster convergence is incorrect.

It may be a bit puzzling to see that the trajectories of the classical PSO with $w = 0.70$, $aw = 4$, and $sp = 1$ in Fig. 6.33 and Fig. 6.34 (right columns) show consistent explosion while the same settings for both attractors kept stationary –and $ip = sp = 0.50$– show either convergence or much smaller explosions, as shown in Fig. 6.37.

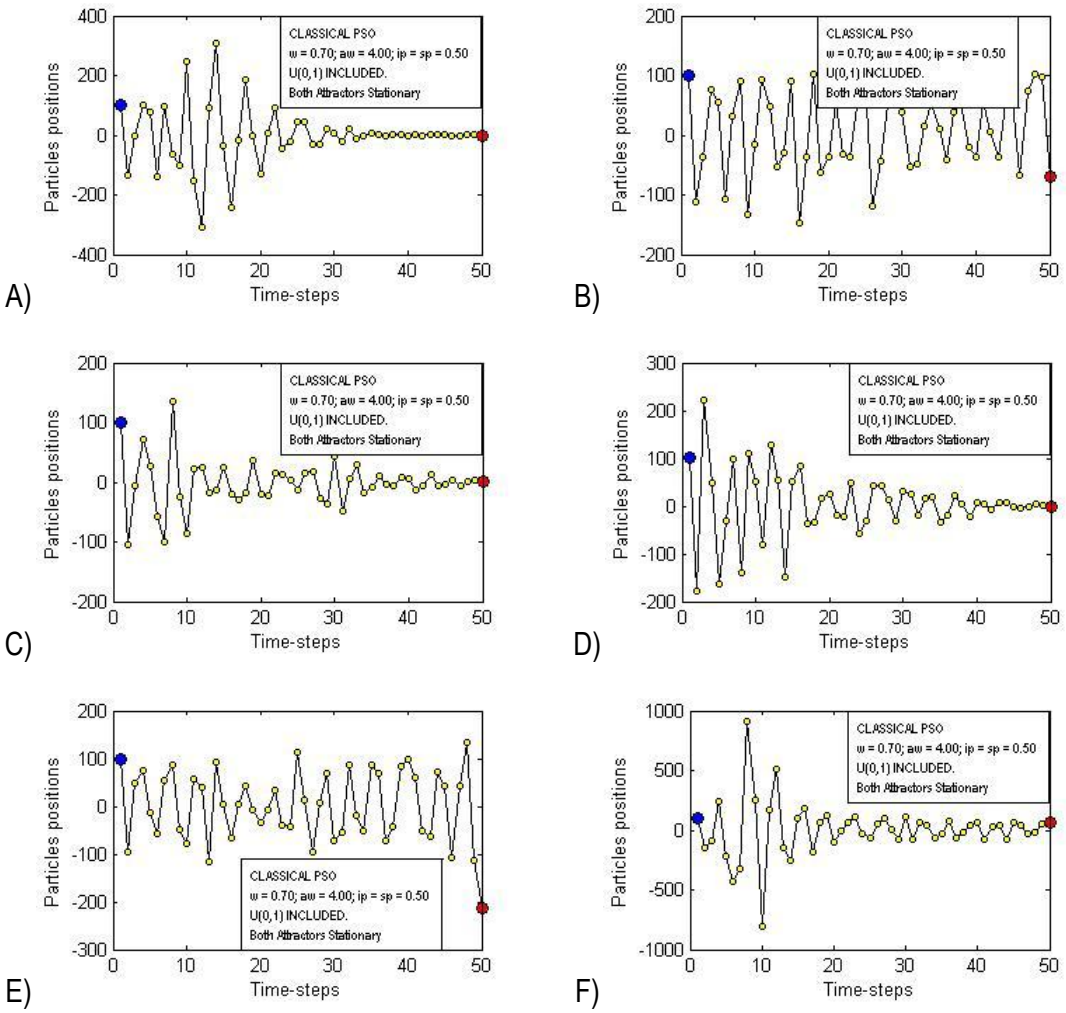


Fig. 6.37. Trajectory of a particle initialized at x = 100, with stationary attractors at x = 0 and ip = sp = 0.50, for the classical PSO with w = 0.70 and aw = 4.00, randomness included, for six consecutive runs.

The reason for this may be that there is only one random weight involved in the velocity update in Fig. 6.33 and Fig. 6.34 (right columns) whereas there are two in Fig. 6.37. Therefore, even when they have the same w and $aw = \phi_{\max}$ –and randomness is included in both cases–, the probability of ϕ taking on extremal values is smaller in the second case. For instance, $p(\phi > 2) = 0.50$ in Fig. 6.33 and Fig. 6.34 whereas $p(\phi > 2) = 0.25$ in Fig. 6.37.

In order to observe the pure influence of the relative values of the individuality and sociality, randomness is disregarded and the trajectory of the deterministic particle with $\phi = \phi_{\text{mean}}$ for the PSO-RRR1 is offered in Fig. 6.38 for increasing individuality.

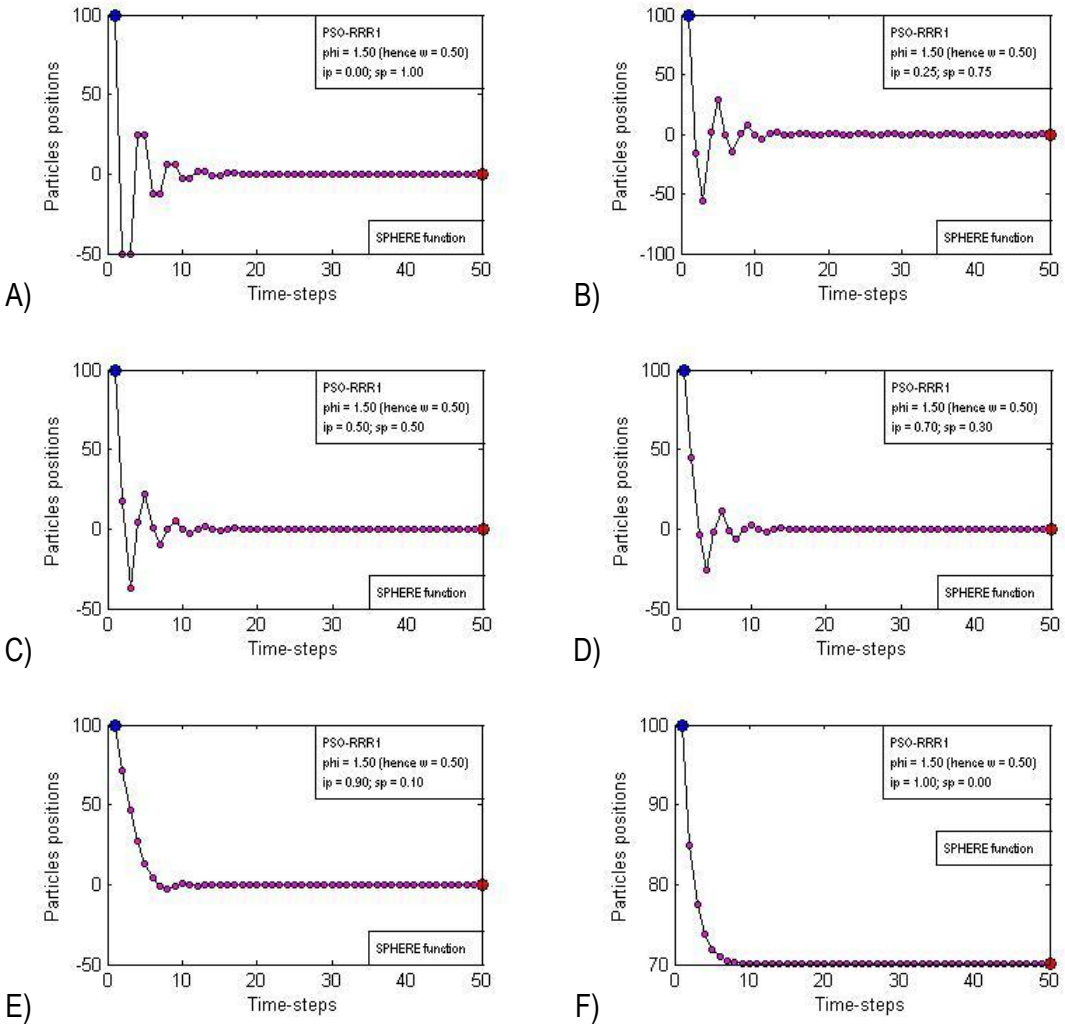


Fig. 6.38. Trajectory of a deterministic particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ = 1.50 and therefore w = 0.50, for increasing values of ip ($0 \leq ip \leq 1$), for the PSO-RRR1.

Fig. 6.38 C) shows the trajectory of the deterministic particle with strength of the individuality and the sociality equally weighted. The other trajectories in Fig. 6.38 clearly show that increasing sociality does not speed up convergence, and increasing individuality does not delay it (which would favour exploration).

The trajectories corresponding to six consecutive runs of the same optimizer –that is the PSO-RRR1 with $\phi_{\text{mean}} = 1.50$ – for $ip = 0.70$ (hence $sp = 0.30$) are shown in Fig. 6.39.

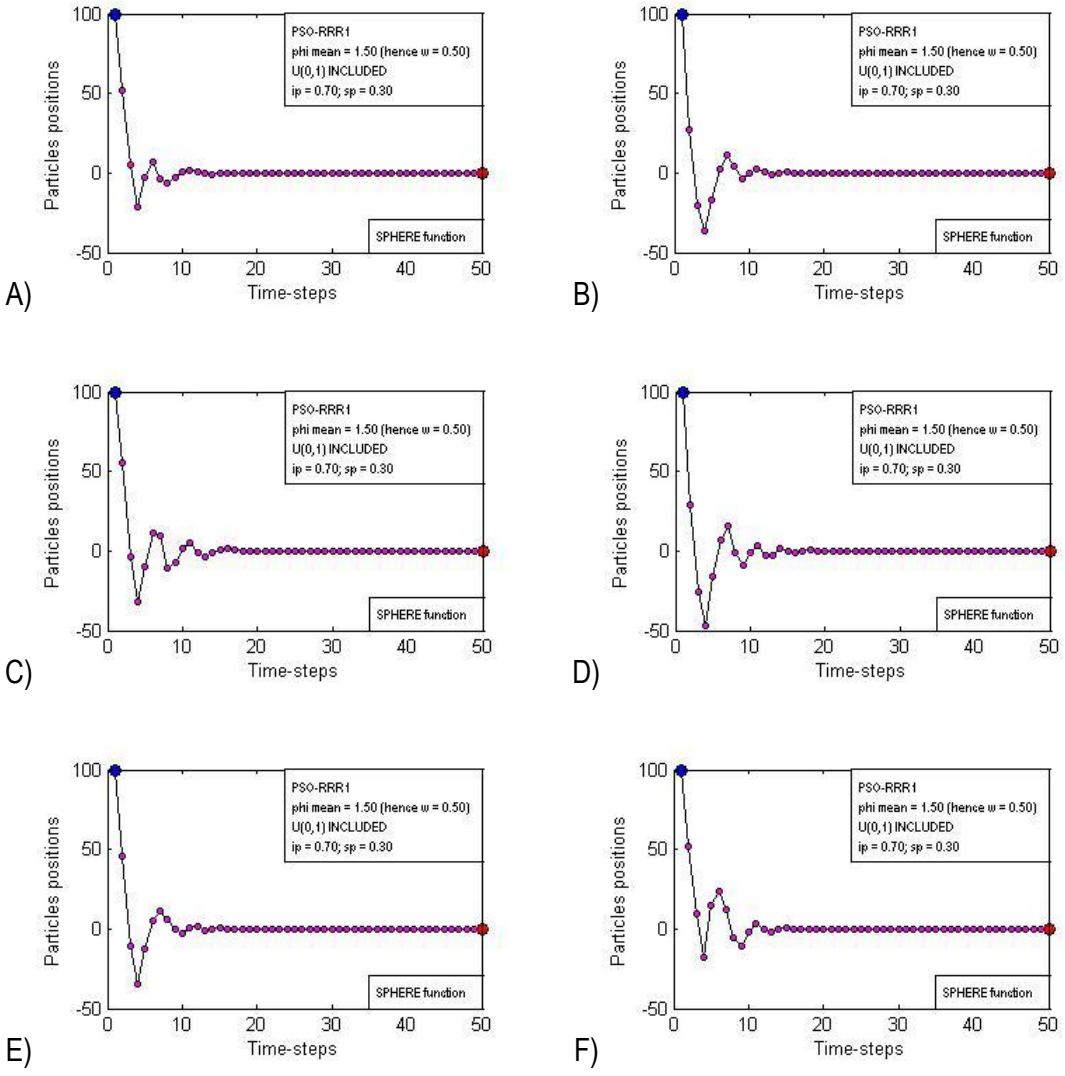


Fig. 6.39. Trajectory of a particle initialized at x = 100, optimizing the 1-dimensional Sphere function, with stationary social attractor at x = 0 and dynamic individual attractor initialized at x = 90, for ϕ_{mean} = 1.50 and hence w = 0.50, for ip = 0.70 and hence sp = 0.30, for the PSO-RRR1, corresponding to six consecutive runs.

Comparing the trajectories in Fig. 6.39 to the corresponding ones in Fig. 6.25 and Fig. 6.26 (right columns), it can be observed that the exploration is only marginally nar-

rowed by setting $ip = 0.70$ instead of $ip = 0.50$, confirming what is observed in the deterministic particle in Fig. 6.38. Therefore, while it is suggested here that the individuality and sociality be equally weighed –in the absence of additional problem-dependent information– to be on the safe side, setting different weights is not necessarily ruled out. Nonetheless, it does not appear convenient to set substantially different individuality and sociality strengths. There might be problems –probably multi-modal ones– for which an increase in the individuality appears convenient, or even unimodal problems for which a marginal increase in the individuality improves performance. But as a general setting, it would seem advisable to set them the same. A numerical study of the improvement/detriment in performance due to different settings of the individuality and sociality is beyond the scope of this thesis due to time and space constraints. Therefore such study is left for future work, and the informed guess here is to keep them equal to one another and leave the random weights in charge of altering the particle's self-confidence throughout the search.

It is to be expected that, if the social attractor is updated, the particle will take longer to approach the new attractor for higher individuality. However this is due to a momentum decrease in comparison to a particle with higher sociality. Therefore, it would seem a better choice to control the speed of convergence and the exploration by means of the 'w–ϕ_{mean}' setting and/or the neighbourhood topology.

6.2.5. Preselected sets of settings

Considering the preceding studies, a set of coefficients settings are selected in this section for the experiments to follow in this thesis. The individuality and sociality are equal to one another in all of them. Recall that the constricted PSO and the PSO-RRM can be viewed as particular cases of the classical PSO, while the acceleration weight in the PSO-RRR1 and the PSO-RRR2 stand for ϕ_{mean} rather than ϕ_{max}, as in classical PSO.

6.2.5.1. Classical PSO ($aw = \phi_{\text{max}}$)

- PSO-1: $w = 0.8000$; $aw = 4.0000$
- PSO-2: $w = 0.7000$; $aw = 4.0000$

- PSO-3: $w = 0.5000$; $aw = 4.0000$ (correlated as in Eq. (5.102))
- PSO-4: $w = 0.3000$; $aw = 4.0000$
- PSO-5: $w = 0.2882$; $aw = 3.4000$ (correlated as in Eq. (5.102))

6.2.5.2. Constricted PSO ($aw = \phi_{max}$) (*Type 1"*)

- C-PSO-1: $w = 0.7298$; $aw = 2.9922$ ($\kappa = 0.99994$; unconstricted $aw = 4.1$)
- C-PSO-2: $w = 0.6204$; $aw = 2.5435$ ($\kappa = 0.85000$; unconstricted $aw = 4.1$)

6.2.5.3. PSO-RRM ($aw = \phi_{max}$)

- PSO-RRM-1: $w = 0.8000$; hence $aw = 3.6000$
- PSO-RRM-2: $w = 0.7000$; hence $aw = 3.4000$
- PSO-RRM-3: $w = 0.5000$; hence $aw = 3.0000$
- PSO-RRM-4: $w = 0.3000$; hence $aw = 2.6000$
- PSO-RRM-5: $w = U_{(0.5,0.8)}$; $aw = 2 \cdot (w + 1)$

6.2.5.4. PSO-RRR1 ($aw = \phi_{mean}$)

- PSO-RRR1-1: $aw = 1.8000$; hence $w = 0.8000$
- PSO-RRR1-2: $aw = 1.7000$; hence $w = 0.7000$
- PSO-RRR1-3: $aw = 1.5000$; hence $w = 0.5000$
- PSO-RRR1-4: $aw = 1.3000$; hence $w = 0.3000$
- PSO-RRR1-5: $aw = U_{(1.5,1.8)}$; $w = aw - 1$

6.2.5.5. PSO-RRR2 ($aw = \phi_{mean}$)

- PSO-RRR2-1: $aw = 2.4000$; hence $w = 0.8167$
- PSO-RRR2-2: $aw = 2.2500$; hence $w = 0.6944$

- PSO-RRR2-3: $aw = 2.0000$; hence $w = 0.5000$
- PSO-RRR2-4: $aw = 1.7000$; hence $w = 0.2882$
- PSO-RRR2-5: $aw = U_{(2.0,2.4)}$; $w = 1/aw - 2 + aw$

6.2.5.6. Other authors' settings ($aw = \phi_{max}$)

- Trelea (2003): $w = 0.60$; $aw = 3.40$
- Hu et al. (2003): $w = U_{(0.5,1.0)}$; $aw = 2.9889$

The inertia weight involves randomness in the PSO-RRM-5, the PSO-RRR1-5, the PSO-RRR2-5, and in (Hu, Eberhart, & Shi, 2003). However in the latter case *w* and *aw* are unrelated whereas in the other approaches *aw* is calculated as a function of the computed *w* or vice versa. Although not specifically stated here, recall that the C-PSO is a work of Clerc and Kennedy (2002). The qualitative exploration/exploitation trade-offs that are to be expected from the settings proposed in this section are shown in Table 6.1.

Table 6.1. Qualitative exploration/exploitation trade-offs to be expected from a number of optimizers proposed in section 6.2.5.

Extreme exploration	Exploration	Balanced	Exploitation	Extreme exploitation
PSO-1	PSO-2	PSO- 3	PSO-4	PSO-5
	PSO-RRM-1	C-PSO-1	C-PSO-2	PSO-RRM-4
	PSO-RRM-2	PSO-RRM-3	PSO-RRR1-3	PSO-RRR1-4
	PSO-RRR1-1	PSO-RRM-5	PSO-RRR2-3	PSO-RRR2-4
	PSO-RRR2-1	PSO-RRR1-2		
		PSO-RRR1-5		
		PSO-RRR2-2		
		PSO-RRR2-5		
		Trelea (2003)		
		Hu et al. (2003)		

The next and final step towards the complexity of the full PSO algorithm consists of reincorporating the particles' interactions. The remainder of this thesis deals with the full algorithm. That is, a swarm of interacting particles whose **pbest** and **lbest** are non-stationary, and with random coefficients weighing the strength of the attractions.

6.3. Swarm of particles

Before conducting the experiments on a set of multidimensional benchmark problems, a small swarm of four particles optimizing the one-dimensional Sphere function is used to visualize the behaviour of interacting particles with the settings selected in section 6.2.5. Thus, by means of a visual, qualitative analyses of the trajectories, the number of selected settings is reduced for the experiments on benchmark multidimensional problems to follow in section 6.3.2.

6.3.1. Four particles and one dimension

The experiments in this section are performed using a small swarm of four particles and a global topology. The positions of the four particles and the corresponding **pbest**'s are initialized at $x = 100$, $x = 50$, $x = -50$, $x = -100$. That is to say, the initial positions and the initial individual best experiences coincide. The velocities are initialized to zero, so that there is one particle that does not move from the first to the second time-steps. As previously mentioned, the individuality and sociality are awarded equal importance in all experiments from here forth within this thesis. Hence $ip = sp = 0.50$.

It is important to note that only the simple, unimodal sphere function is optimized in these experiments. Slower and more uneven convergence is to be expected for more difficult problems, especially multimodal ones. Four consecutive runs are performed for each experiment. All the trajectories are presented in Fig. 6.40 to Fig. 6.72.

6.3.1.1. Classical PSO

Five settings within the family of the classical PSO algorithms were selected in section 6.2.5.1. Their expected explorative/exploitative trade-offs were offered in Table 6.1. The results are gathered in Fig. 6.40 to Fig. 6.46.

Fig. 6.40 shows that some particle(s) in the PSO-1 may perform stochastic explosions of considerable size, while the overall convergence is slow, uneven, and differs notably for different runs. It is evident that the PSO-1 needs the aid of the velocity constraint, and its behaviour is extremely explorative, as predicted in Table 6.1. Thus, the experiments are repeated constraining the velocities to half the feasible interval (refer to Fig. 6.41).

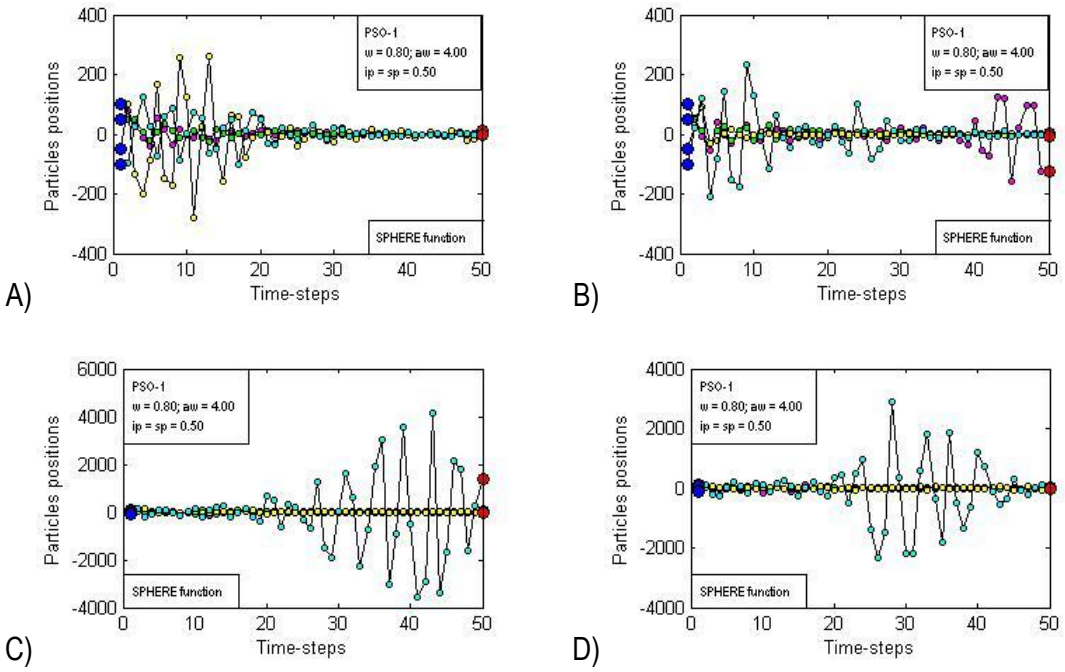


Fig. 6.40. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-1 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

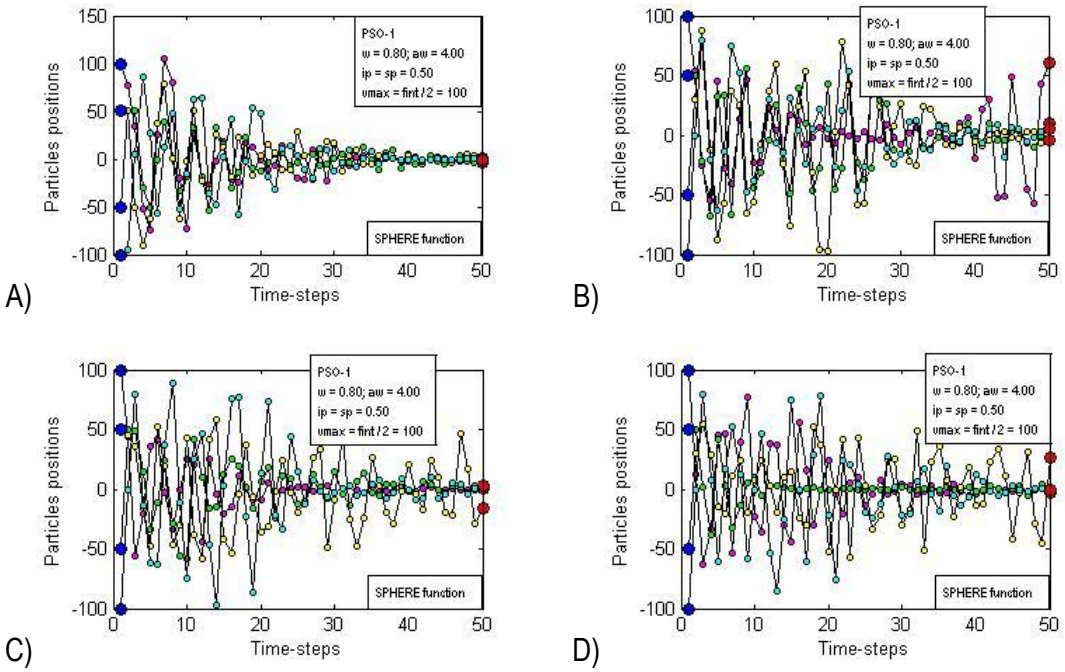


Fig. 6.41. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-1 algorithm with ip = sp = 0.50 and $v_{\max}$ = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

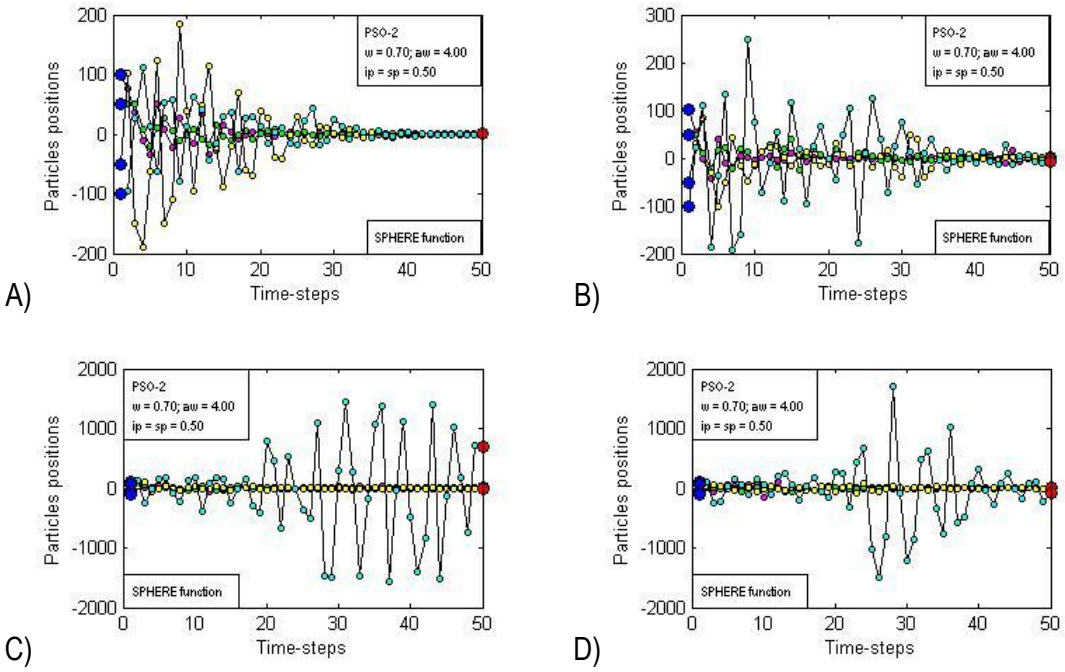


Fig. 6.42. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-2 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

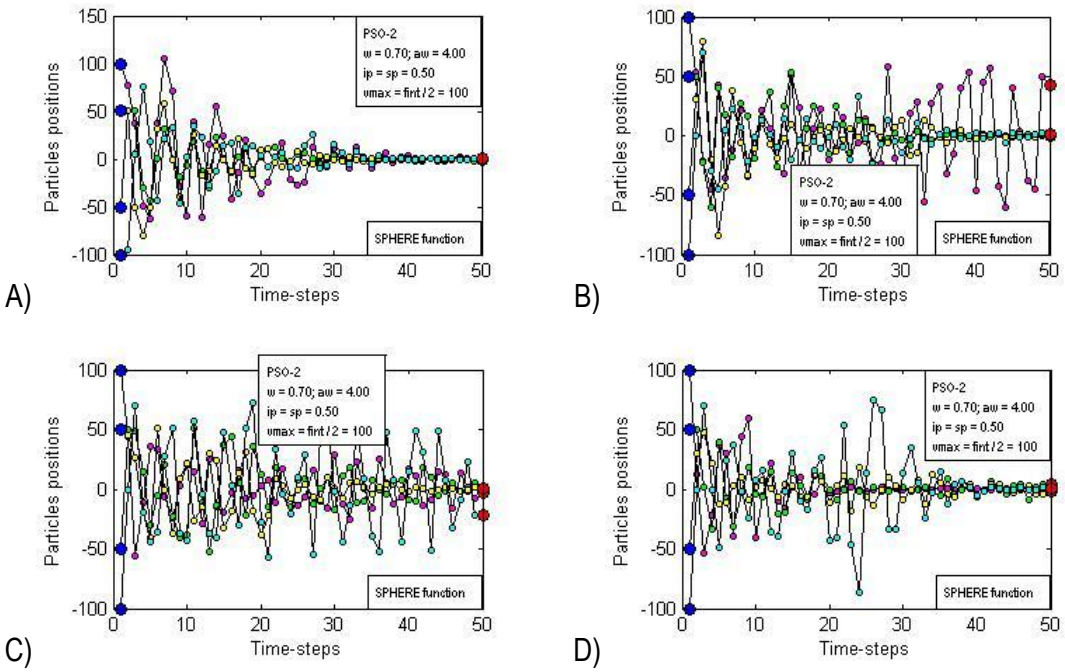


Fig. 6.43. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-2 algorithm with ip = sp = 0.50 and $v_{\max}$ = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

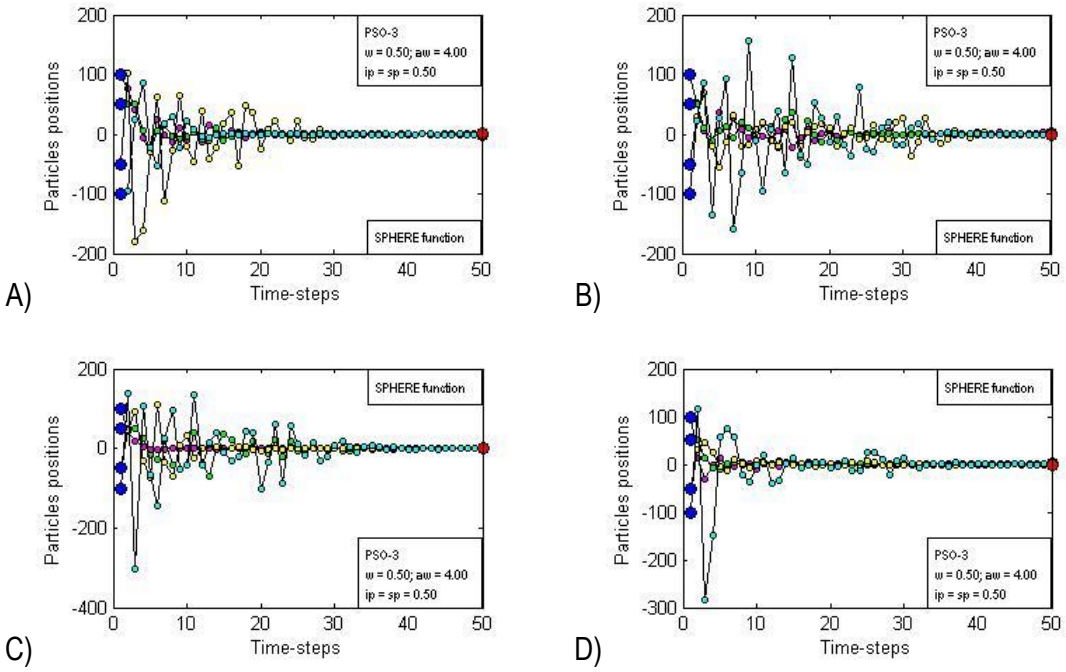


Fig. 6.44. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-3 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

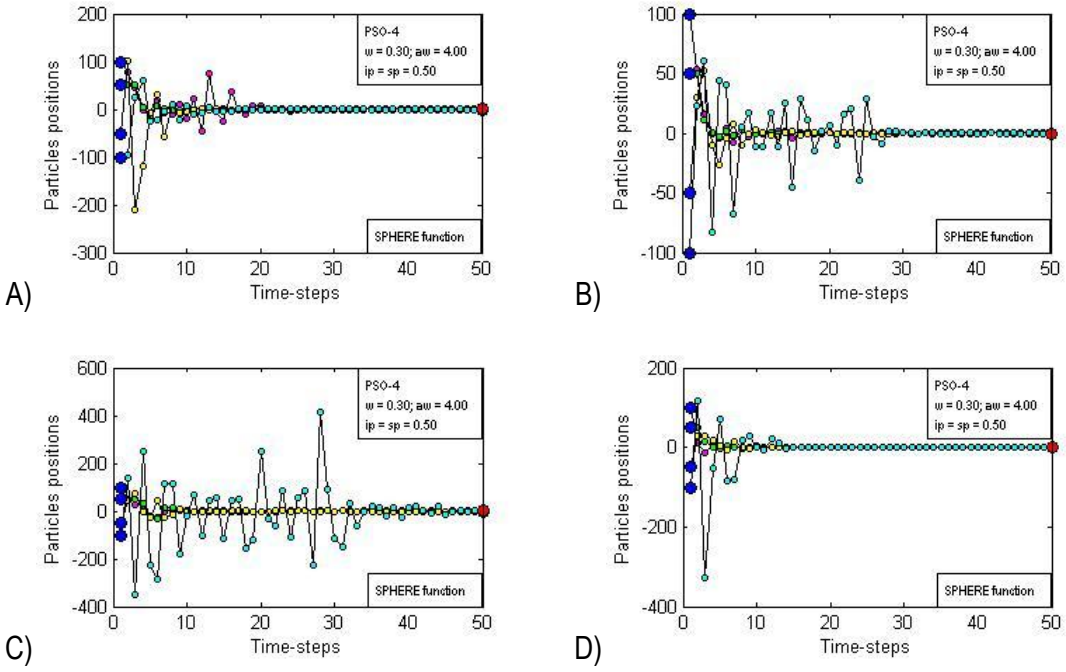


Fig. 6.45. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-4 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

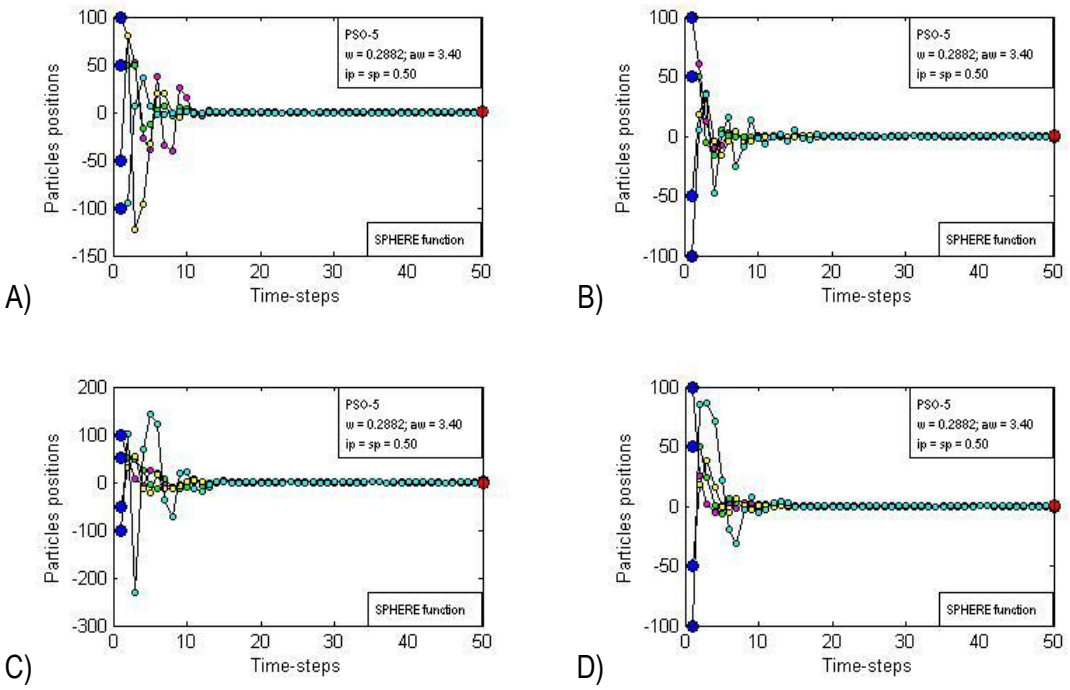


Fig. 6.46. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-5 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

It can be observed in Fig. 6.41 that the velocity constraint is successful in controlling the local, stochastic explosions. However the convergence is not consistent for all runs.

Fig. 6.42 shows the trajectories for the PSO-2. Local, stochastic explosions are still observed, although their sizes are much smaller than for the PSO-1. Nevertheless, it would seem that the implementation of the velocity constraint is also rather a must here. The trajectories with the velocity constrained to half the feasible interval [−100,100] are offered in Fig. 6.43. The constraint controls the local explosions, and there is, in general, convergence. However, the trajectories are still uneven and differ notably for different runs. The behaviour is clearly 'explorative' –as predicted in Table 6.1–, although it may be borderline 'extremely explorative'.

Fig. 6.44 shows that the PSO-3 exhibits more balanced exploration/exploitation behaviour. This would make it more appropriate for a general, stand-alone optimizer than the PSO-1 and PSO-2. Trajectories are still, nevertheless, rather uneven.

PSO-4 and PSO-5 were expected to exhibit highly exploitative trajectories, which can be confirmed by observing Fig. 6.45 and Fig. 6.46. The PSO-5 shows trajectories more

consistent for different runs, and practically no local stochastic explosions, as opposed to the PSO-4. Therefore the former exhibits a more desirable exploitative behaviour.

6.3.1.2. Constricted PSO

Only two settings within the family of the C-PSO algorithms were selected in section 6.2.5.2. Their expected explorative/exploitative trade-offs were offered in Table 6.1, while the resulting trajectories are shown in Fig. 6.47 and Fig. 6.48.

The C-PSO-1 in Fig. 6.47 confirms the expected balanced explorative/exploitative behaviour. Although the trajectories show –as expected– lower pseudo frequencies than those of the PSO-3 in Fig. 6.44, their balances of exploration and exploitation are similar, with the C-PSO-1 exhibiting slightly slower convergence. In both cases, some small local explosions can be observed, as well as some irregularity from one run to another.

The trajectories in Fig. 6.47 and Fig. 6.48 are without the v_{max} constraint. Despite the constriction factor helping prevent explosions, it still allows the particles to wander off far from their local social attractor, thus mixing up with other particles in different neighbourhoods. While this may still lead to good results, the user loses the notion and hence control of what is happening in the dynamics of the swarm. In order to show more clearly the similarly balanced exploration/exploitation trade-off of the C-PSO-1 and the PSO-3, the experiments are run again for both settings with the addition of the v_{max} constraint. The trajectories are offered in Fig. 6.49 and Fig. 6.50, where it can be clearly observed that they both present a balanced explorative/exploitative behaviour, with the PSO-3 showing somewhat faster convergence.

The C-PSO-2 in Fig. 6.48 shows –as predicted in Table 6.1– more exploitative behaviour than the C-PSO-1 and the PSO-3, with fewer and smaller local explosions. Thus, the search is narrow and convergence is fast, although not as fast as that of the PSO-5 in Fig. 6.46. Therefore the C-PSO-2 does not seem convenient as a general setting, although it might be useful to combine with some others, or when very fast convergence is sought in detriment of accuracy in the results. Consider, for instance, the case where the objective function is too expensive, and the PSO algorithm is only used to obtain the initial point for a more cost-effective local search. The same is true for the PSO-5, while the PSO-4 loses the selection against the PSO-5 for regular, fast convergence.

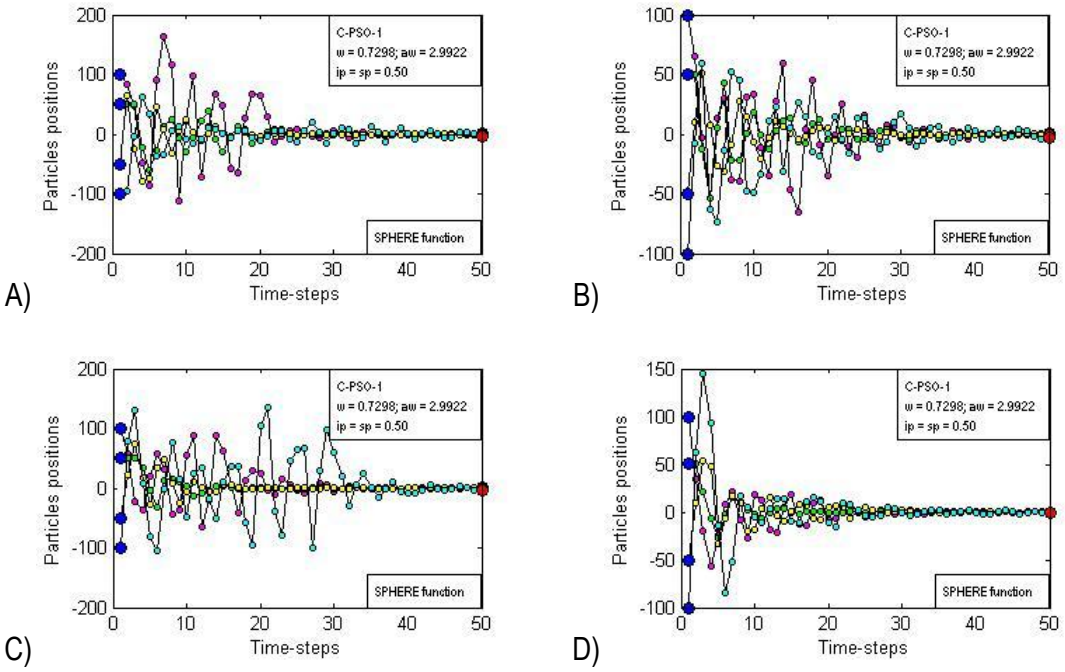


Fig. 6.47. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the C-PSO-1 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

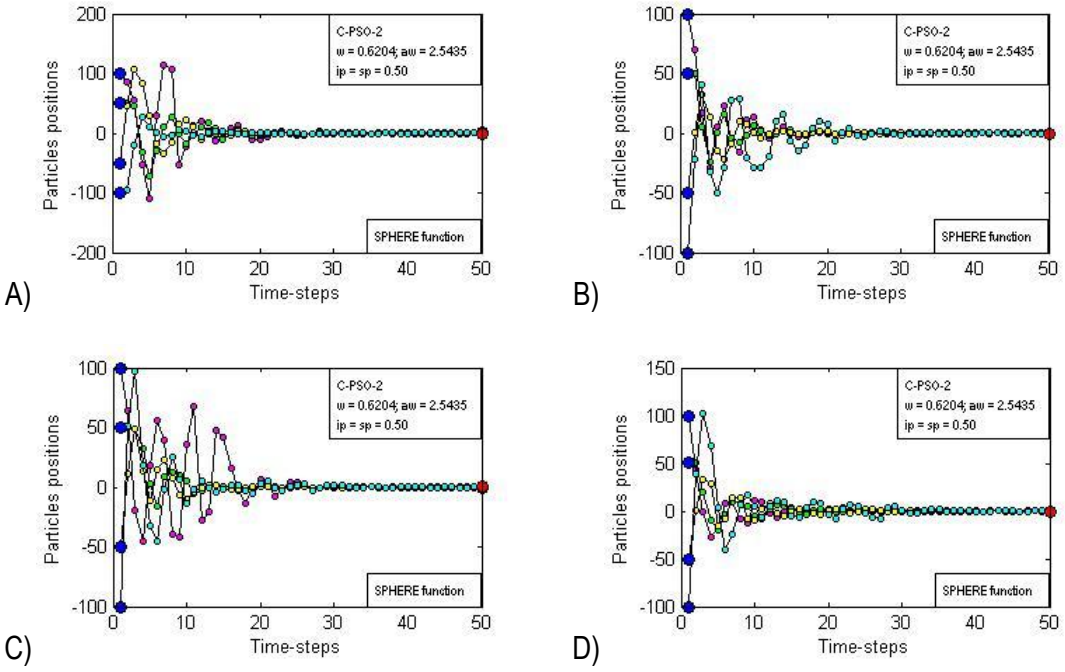


Fig. 6.48. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the C-PSO-2 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

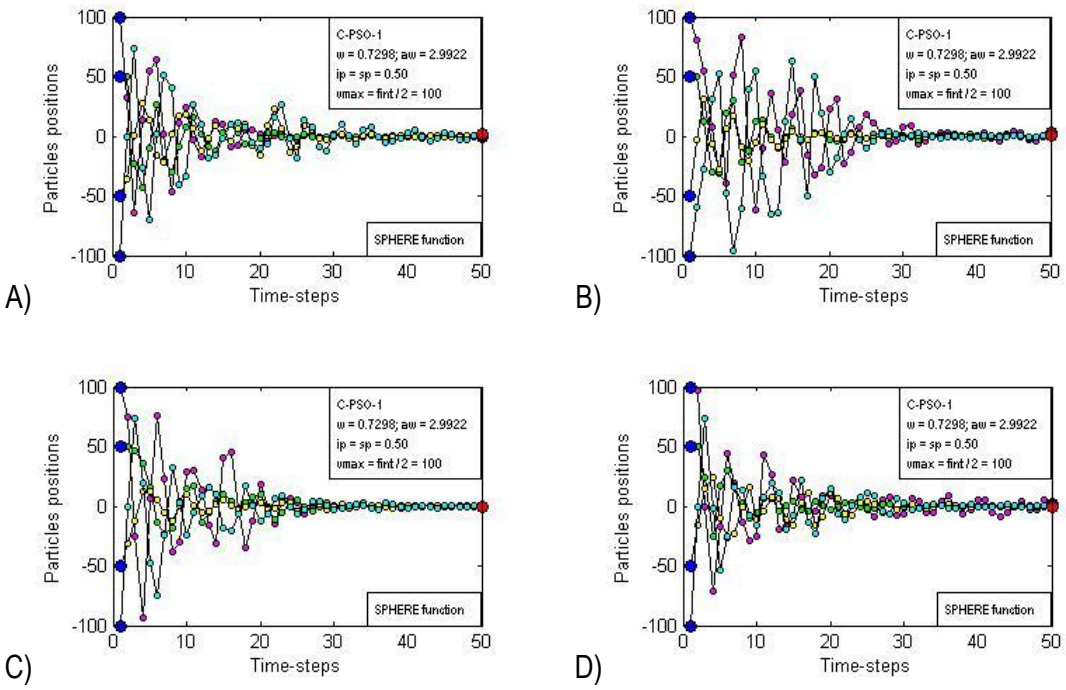


Fig. 6.49. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the C-PSO-1 algorithm with ip = sp = 0.50 and v_{max} = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

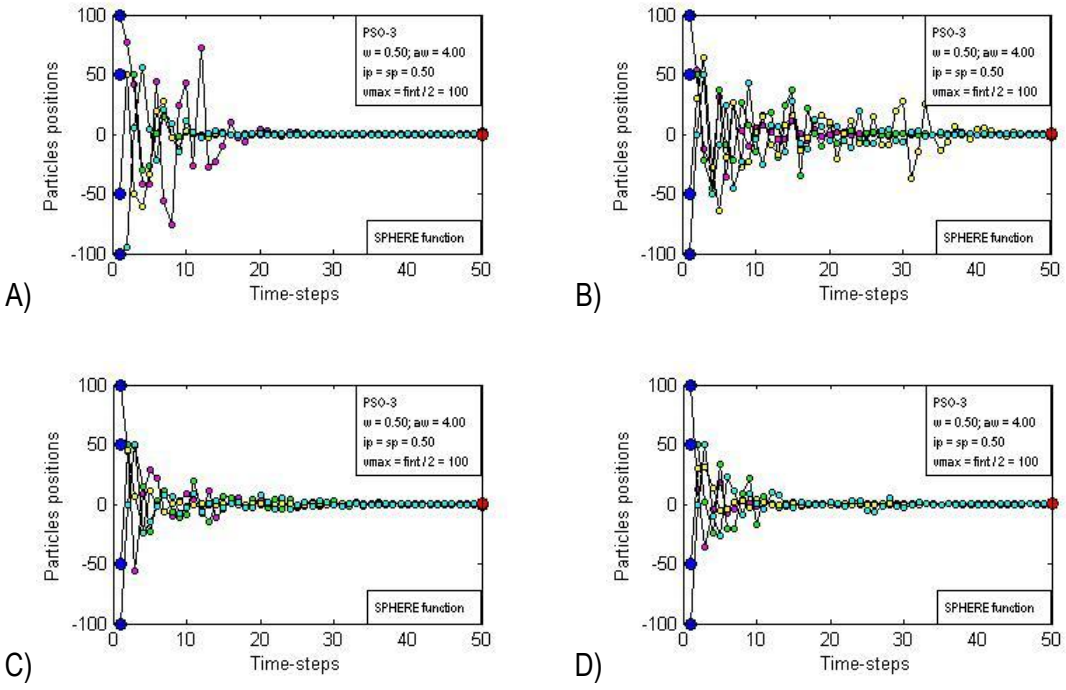


Fig. 6.50. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-3 algorithm with ip = sp = 0.50 and v_{max} = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

Bear in mind that convergence would be slowed down in problems harder than the simple, unimodal sphere function.

6.3.1.3. PSO-RRM

Five settings within the family of the PSO-RRM algorithms proposed in section 6.2.2.1 were selected in section 6.2.5.3. The fifth one is actually a variation, where w (or aw) is every time selected randomly within the arbitrary interval [0.50,0.80] (or [3.00,3.60]). Once w (or aw) is obtained, aw (or w) is calculated as in Eq. (6.6). To observe the subregion within the convergence region from where the pair 'ϕ–w' is therefore randomly chosen every time, refer to Fig. 6.2, where the lower limit should be raised to $w = 0.50$. The balance of explorative/exploitative behaviour to be expected for the five PSO-RRM settings was offered in Table 6.1, while the resulting trajectories are shown in Fig. 6.51 to Fig. 6.56.

The PSO-RRM-1 in Fig. 6.51 show very explorative behaviour, exhibiting local explosions and difficulties in the fine-clustering of the particles. However the local explosions are notably smaller than those of the PSO-1 in Fig. 6.40. Therefore, it would seem more convenient to select the PSO-RRM-1 instead of the PSO-1 if extensive exploration is desired, still with the addition of the v_{max} constraint. Thus, the PSO-1 and the PSO-4 (in the previous section) are already ruled out from the selected settings.

The PSO-RRM-2 in Fig. 6.52 still shows a reasonably explorative, borderline balanced, behaviour, practically without exhibiting local explosions. The trajectories are seemingly slightly more explorative than those of the C-PSO-1 and the PSO-3 in Fig. 6.49 and Fig. 6.44. Compared to the PSO-2 in Fig. 6.42 and Fig. 6.43 (same w), the PSO-RRM-2 shows a more balanced, less explorative behaviour and better fine-clustering abilities, even without the aid of the v_{max} constraint.

The PSO-RRM-3 in Fig. 6.53 exhibits an exploitative behaviour, with slightly faster convergence than the C-PSO-2 in Fig. 6.48. Practically no local explosion is observed, and convergence occurs in 10 to 20 time-steps. While favouring exploitation, the PSO-RRM-3 still performs some exploration. In contrast, the PSO-RRM-4 in Fig. 6.54 is almost exclusively exploitative –even more than the PSO-5 in Fig. 6.46–, consistently exhibiting a very poor exploration of the search-space and very fast convergence.

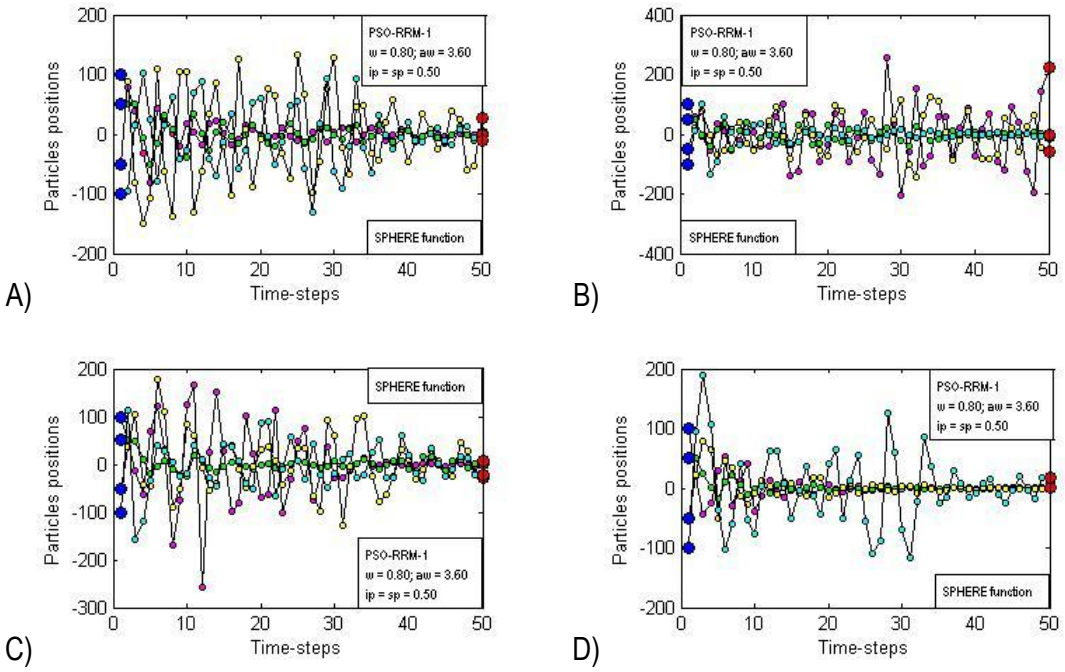


Fig. 6.51. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRM-1 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

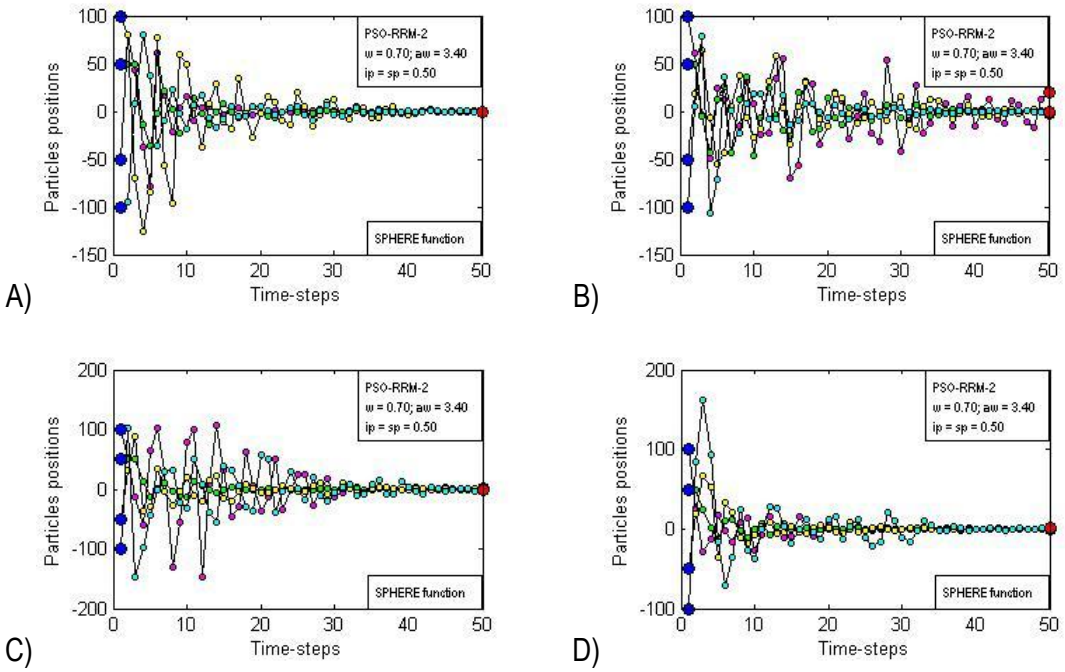


Fig. 6.52. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRM-2 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

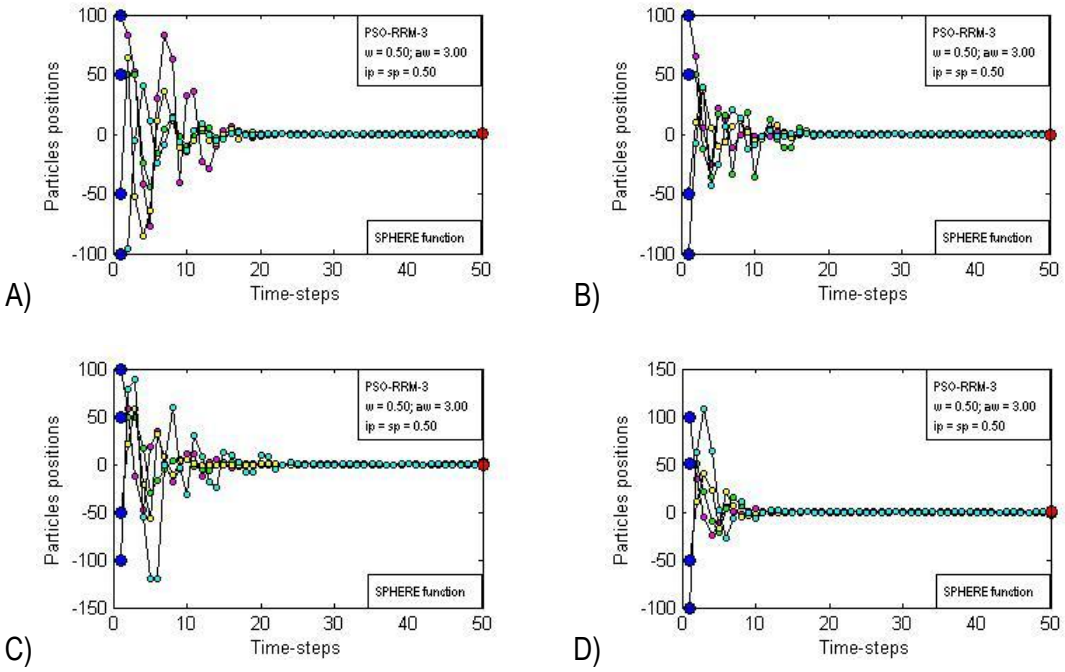


Fig. 6.53. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRM-3 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

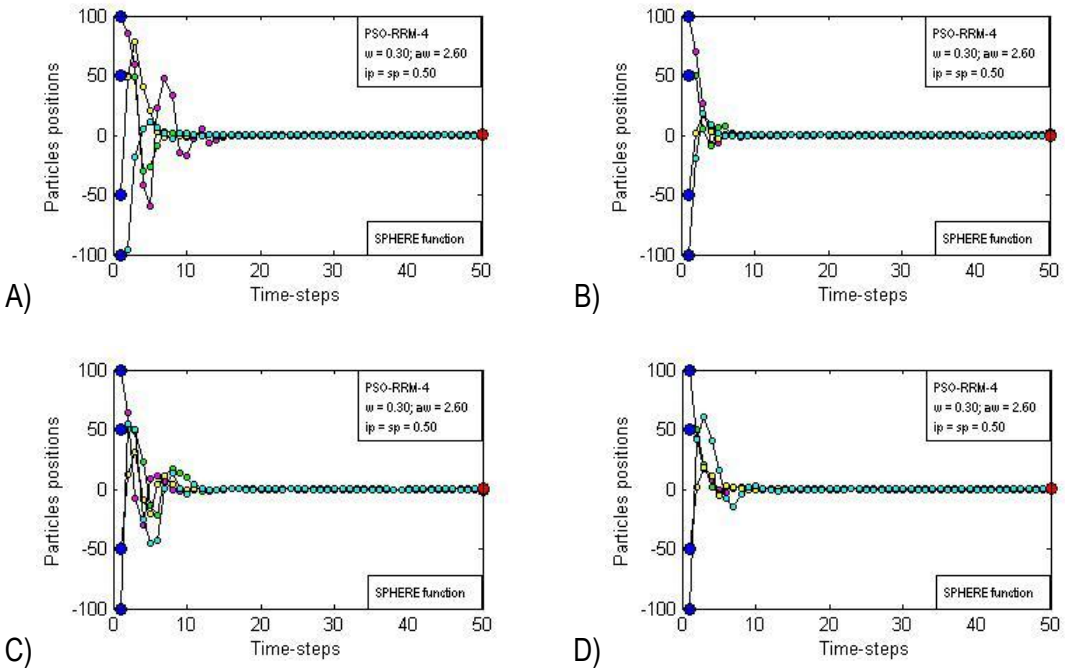


Fig. 6.54. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRM-4 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

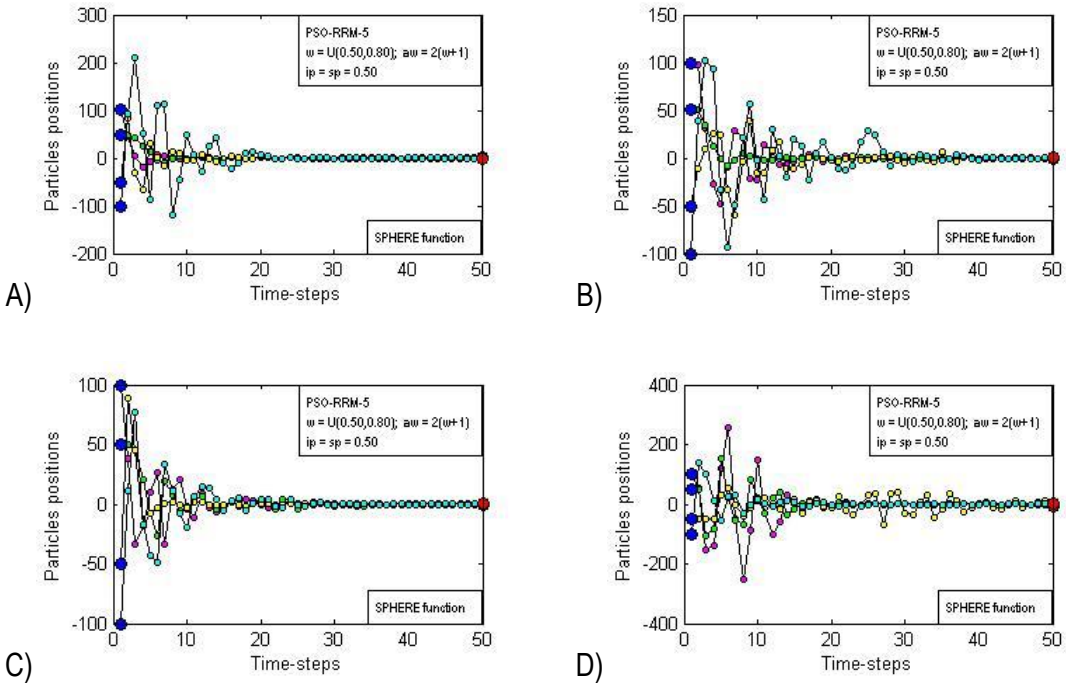


Fig. 6.55. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRM-5 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

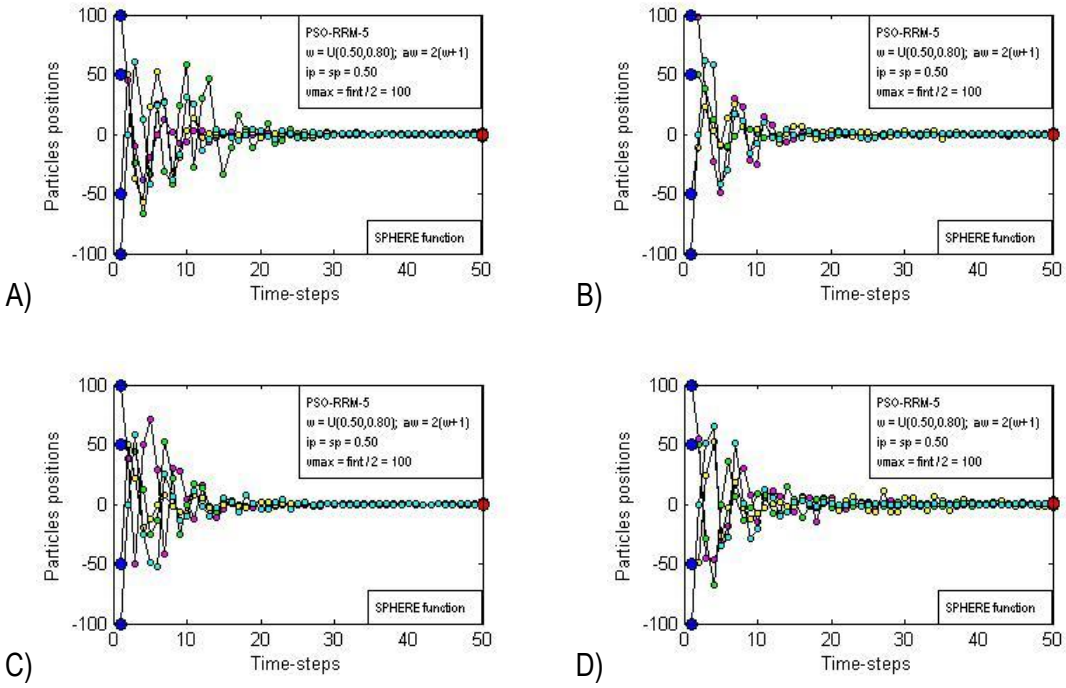


Fig. 6.56. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRM-5 algorithm with ip = sp = 0.50 and v_{max} = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

The PSO-RRM-5 in Fig. 6.55 exhibits more exploitative behaviour than anticipated. Nevertheless, the general behaviour is quite balanced, similar to that of the PSO-3 in Fig. 6.44 in terms of the exploration/exploitation trade-off.

The local explosions make it difficult to perform a more accurate comparison. Therefore, the PSO-RRM-5 is tested again with the v_{max} constraint incorporated, and the trajectories are shown in Fig. 6.56. Comparing those trajectories to those of the PSO-3 in Fig. 6.50 and to those of the C-PSO-1 in Fig. 6.49 (both with v_{max} constraint) it can be stated that the PSO-RRM-5 exhibits a balanced exploration/exploitation trade-off, showing marginally higher exploration than the PSO-3, yet lower exploration than the C-PSO-1. In addition, the trajectories are more regular for different runs than those of the PSO-3. Hence the capabilities of the PSO-RRM-5 and the PSO-3 are similar, with the former showing more desirable general behaviour. This would rule out the PSO-3 from the selected settings in favour of the PSO-RRM-5.

6.3.1.4. PSO-RRR1

Five settings within the family of the PSO-RRR1 algorithms proposed in section 6.2.2.2 were selected in section 6.2.5.4. The fifth one is a variation, where aw is chosen randomly within the arbitrarily chosen interval [1.50,1.80], and then the inertia weight is calculated as $w = aw - 1$. Hence w belongs to the interval [0.50,0.80]. To observe the sub-region within the convergence region from where the pairs 'ϕ–w' are therefore randomly chosen every time, refer to Fig. 6.3, where the lower limit should be raised to $w = 0.50$ and $aw = \phi_{mean}$. The balance of explorative/exploitative behaviour to be expected for the five PSO-RRR1 settings was offered in Table 6.1, while the resulting trajectories are presented in Fig. 6.57 to Fig. 6.61.

The PSO-RRR1-1 in Fig. 6.57 exhibits a behaviour that can be seen as explorative or balanced, with either few and small or no local explosion, while also displaying the ability to converge consistently, even without the aid of the v_{max} constraint. The exploration/exploitation trade-off is similar to those of the PSO-2 (see Fig. 6.42 and Fig. 6.43) and of the PSO-RRM-2 (see Fig. 6.52), while showing remarkably better and more consistent convergence than both, and higher regularity of the trajectories for different runs. It appears to comprise a good stand-alone optimizer.

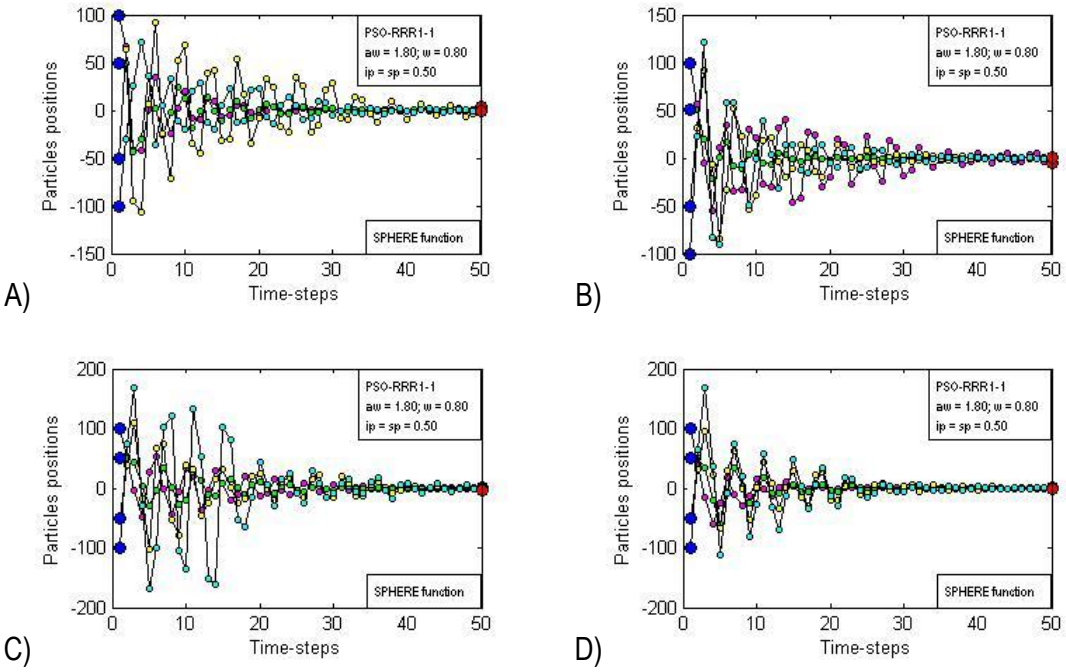


Fig. 6.57. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR1-1 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

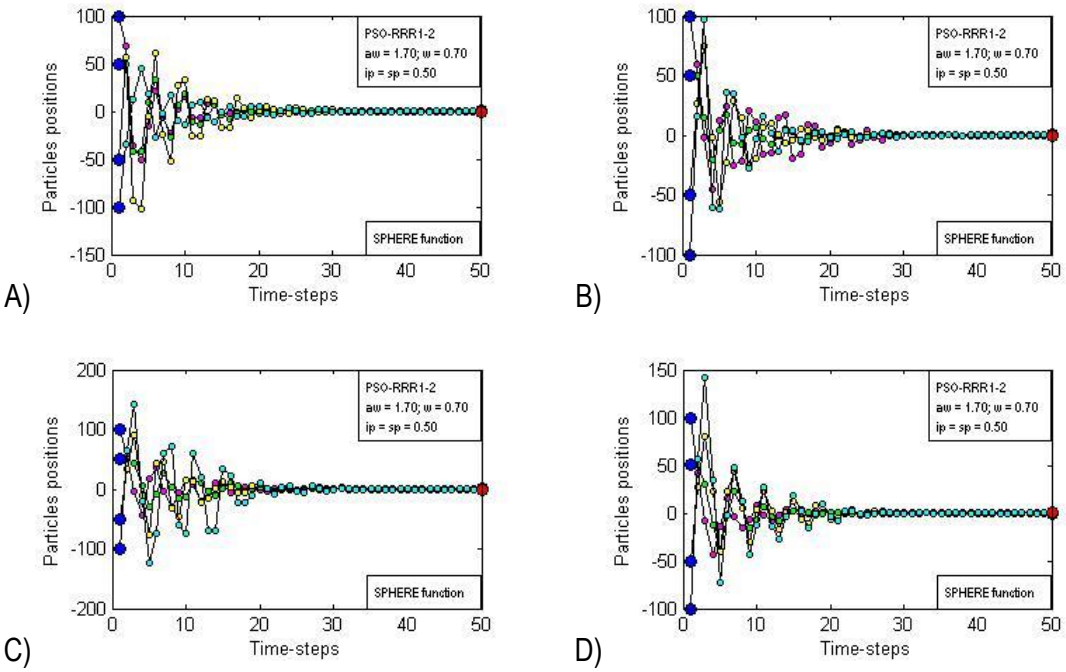


Fig. 6.58. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR1-2 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

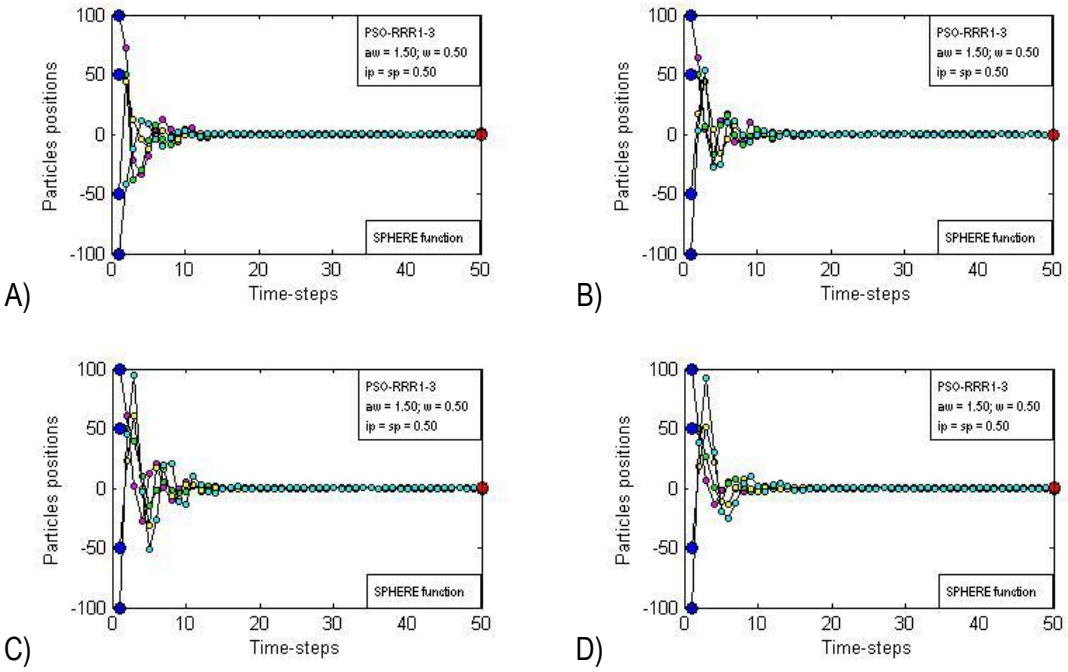


Fig. 6.59. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR1-3 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

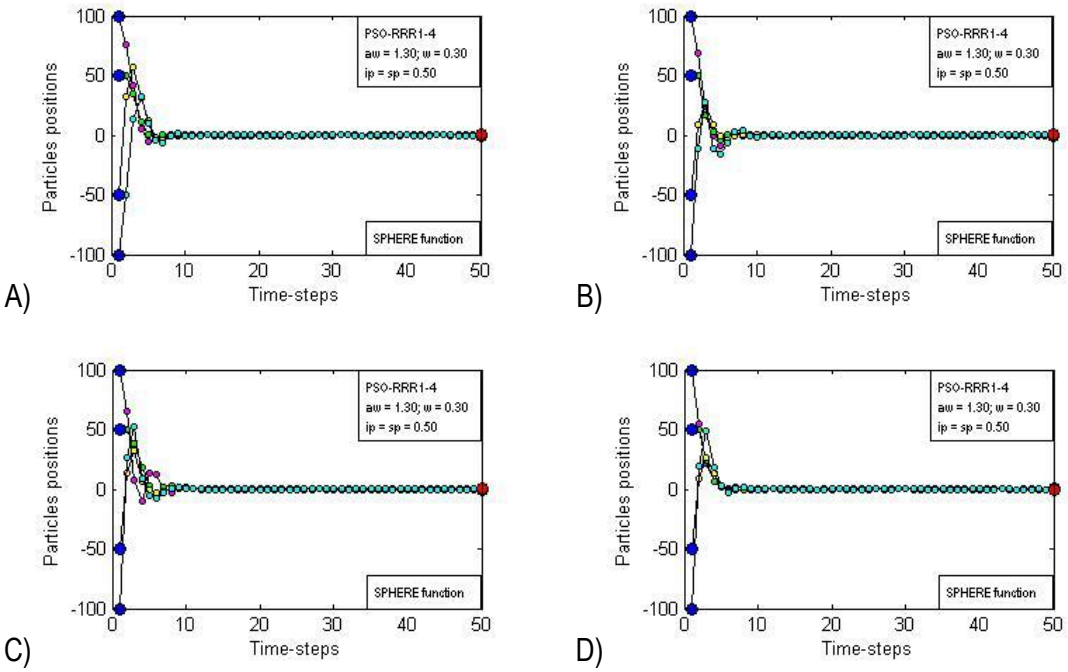


Fig. 6.60. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR1-4 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

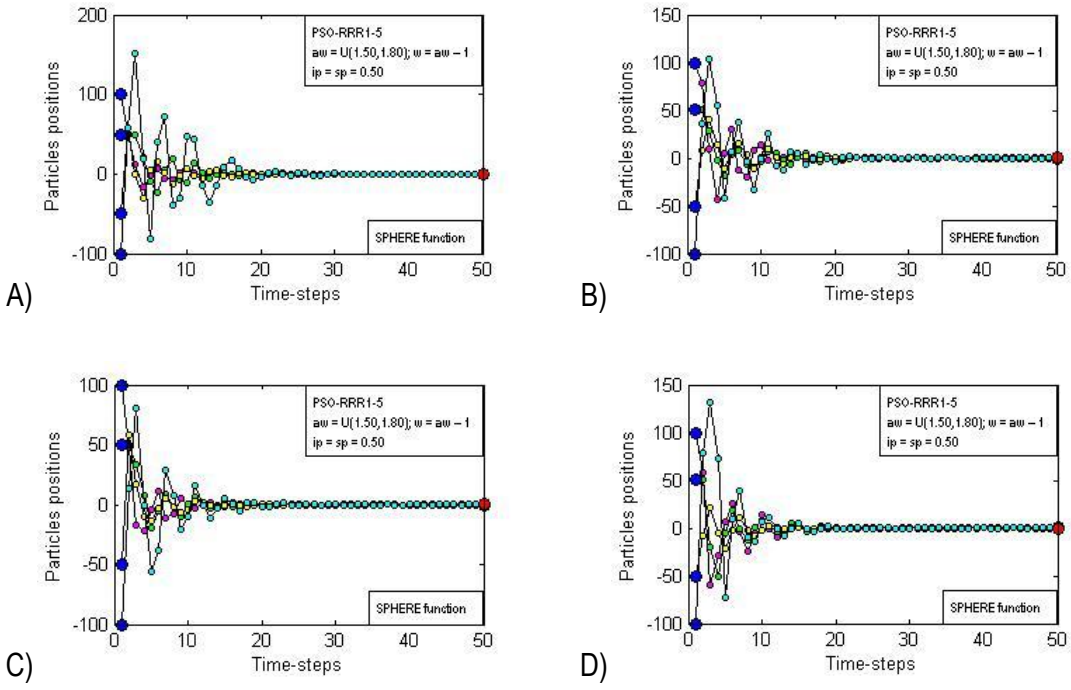


Fig. 6.61. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR1-5 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

The PSO-RRR1-2 in Fig. 6.58 shows very similar behaviour to that of the PSO-RRR1-1 in Fig. 6.57 but exhibiting lower exploration and faster convergence, as expected. It also shows lower exploration, fewer and smaller local explosions, faster convergence, and more similar trajectories between different runs than the PSO-RRM-2 in Fig. 6.52 and the C-PSO-1 in Fig. 6.47 and Fig. 6.49. The exploration/exploitation trade-off is also similar to that of the PSO-RRM-5 in Fig. 6.55 and Fig. 6.56, except that local explosions are less important and the trajectories are more regular for different runs.

The PSO-RRR1-3 in Fig. 6.59 exhibits exploitative behaviour, showing lower exploration, fewer and smaller local explosions, faster convergence, and more similar trajectories between different runs than other exploitative settings such as the C-PSO-2 in Fig. 6.48 and the PSO-RRM-3 in Fig. 6.53. In turn, it displays similar exploitation to those of the PSO-5 in Fig. 6.46 and the PSO-RRM-4 in Fig. 6.54, but showing more regularity in the trajectories for different runs. Therefore, if settings favouring very fast convergence are sought, the PSO-RRR1-3 would appear a better choice than the PSO-5 and the PSO-RRM-4 thus ruling them out from the selected settings.

The convergence for the PSO-RRR1-4 in Fig. 6.60 is so fast that not even exploitation takes place. Thus it is practically useless, and it is therefore ruled out from the selection.

The PSO-RRR1-5 in Fig. 6.61 displays clearly more exploitative behaviour than the PSO-RRM-5 in Fig. 6.56 (same ranges of w and of ϕ_{mean}), and marginally more exploitative that the C-PSO-2 in Fig. 6.48. The exploitation/exploration trade-off is similar to that of the PSO-RRM-3 in Fig. 6.53, with the PSO-RRR1-5 in Fig. 6.61 perhaps showing trajectories corresponding to different runs more similar to one another. All in all, the PSO-RRR1-5 comprises an exploitative algorithm with consistent and fast convergence, yet not as fast as the PSO-RRR1-3 in Fig. 6.59.

6.3.1.5. PSO-RRR2

Five settings within the family of the PSO-RRR2 algorithms proposed in section 6.2.2.3 were selected in section 6.2.5.5. The fifth one is a variation, where aw is chosen randomly within the arbitrarily chosen interval [2.00,2.40], and then the inertia weight is calculated as $w = aw^{-1} - 2 + aw$. Hence w belongs to the interval [0.50,0.8167]. To observe the sub-region within the convergence region from where the pairs 'ϕ–w' are therefore randomly chosen every time, refer to Fig. 6.13, where the lower limit should be raised to $w = 0.50$ and $aw = \phi_{\text{mean}}$. The explorative/exploitative trade-offs to be expected for the five PSO-RRR2 settings were offered in Table 6.1, while the resulting trajectories are presented in Fig. 6.62 to Fig. 6.69.

The PSO-RRR2-1 in Fig. 6.62 displays explorative behaviour, with occasional local explosions. It performs higher exploration than the PSO-RRR1-1 in Fig. 6.57, and similar to the PSO-RRM-1 in Fig. 6.51, while showing better convergence than the latter. Since the local explosions change the scale of the plot, the trajectories for the PSO-RRR2-1 with $v_{\max}$ constraint are offered in Fig. 6.63 for better observation of its explorative behaviour. Thus, the PSO-RRR2-1 is explorative, suitable for a stand-alone, general optimizer. The $v_{\max}$ constraint should be implemented to control the local explosions.

The PSO-RRR2-2 in Fig. 6.64 behaves very similarly to the PSO-RRR2-1 in Fig. 6.62 but performing a narrower search and faster convergence. Its exploration/exploitation trade-off is more balanced, although it also presents local explosions. The trajectories with the $v_{\max}$ constraint are offered in Fig. 6.65, where local explosions are controlled.

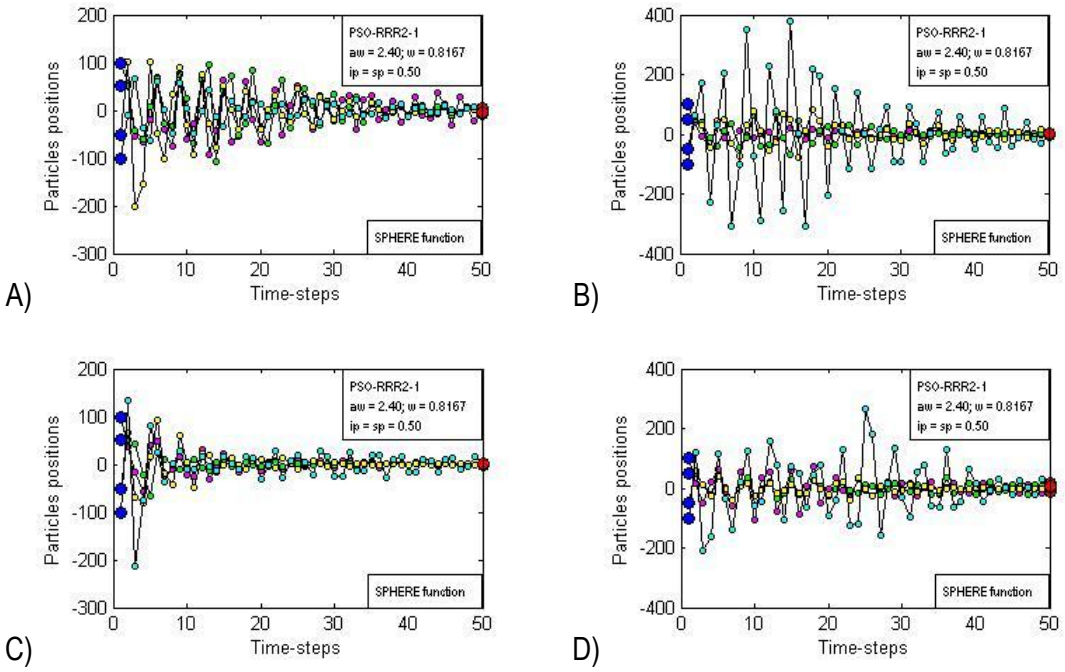


Fig. 6.62. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR2-1 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

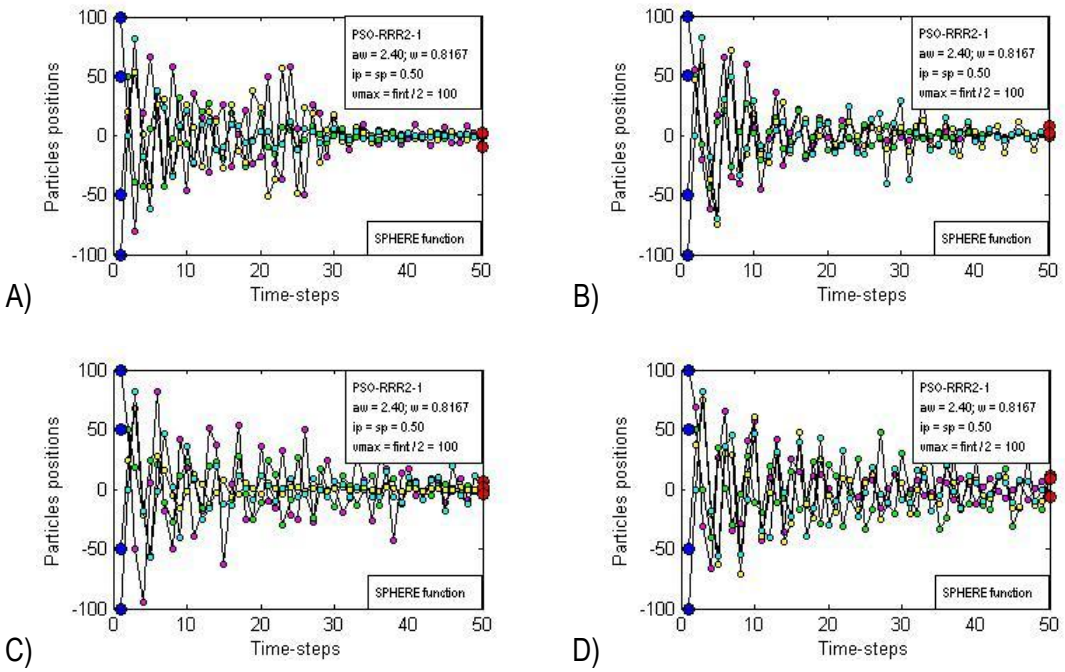


Fig. 6.63. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR2-1 algorithm with ip = sp = 0.50 and $v_{\max}$ = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

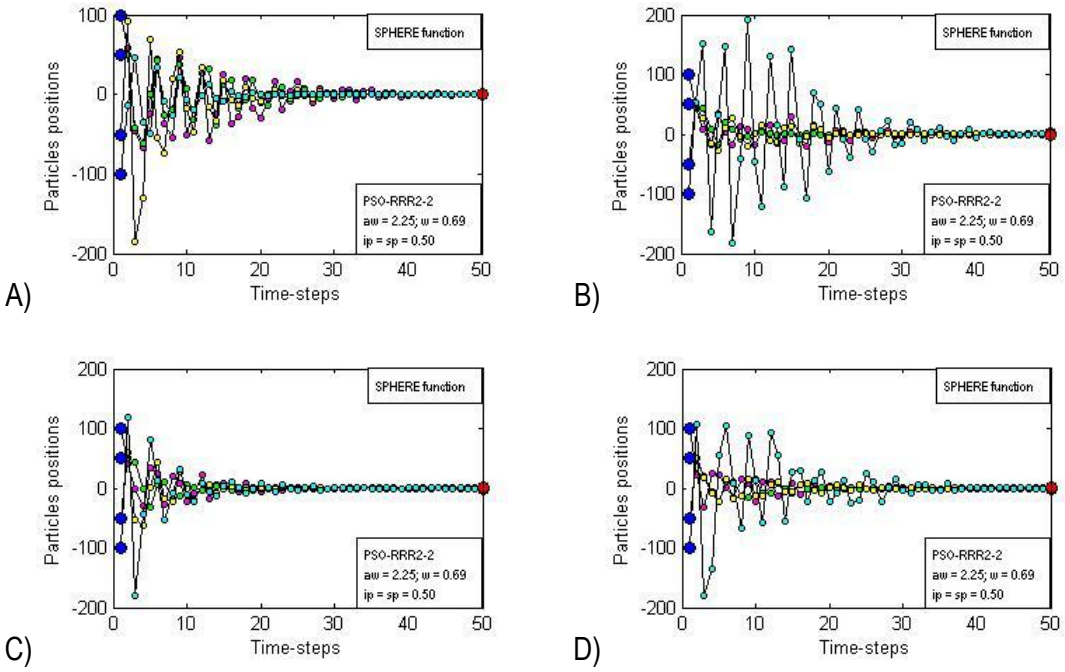


Fig. 6.64. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR2-2 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

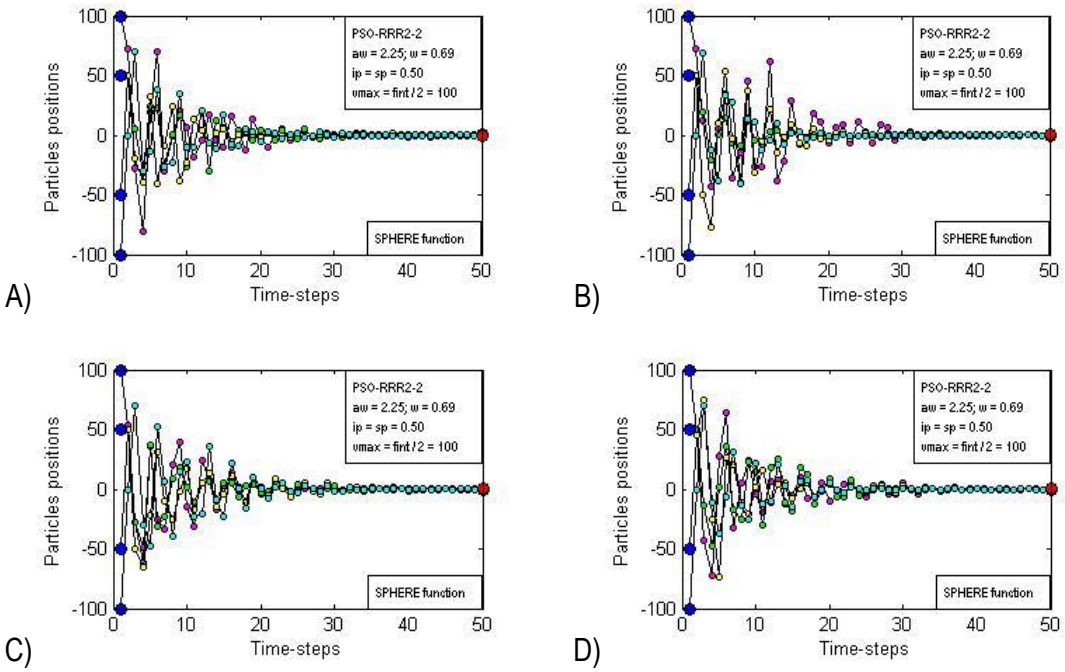


Fig. 6.65. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR2-2 algorithm with ip = sp = 0.50 and v_{max} = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

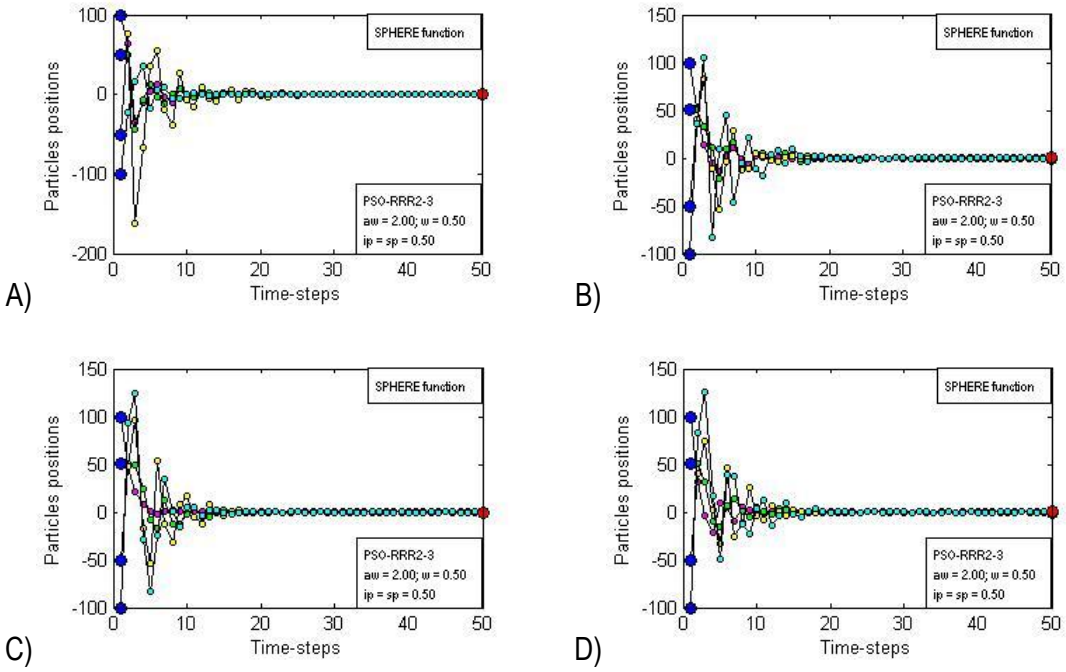


Fig. 6.66. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR2-3 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

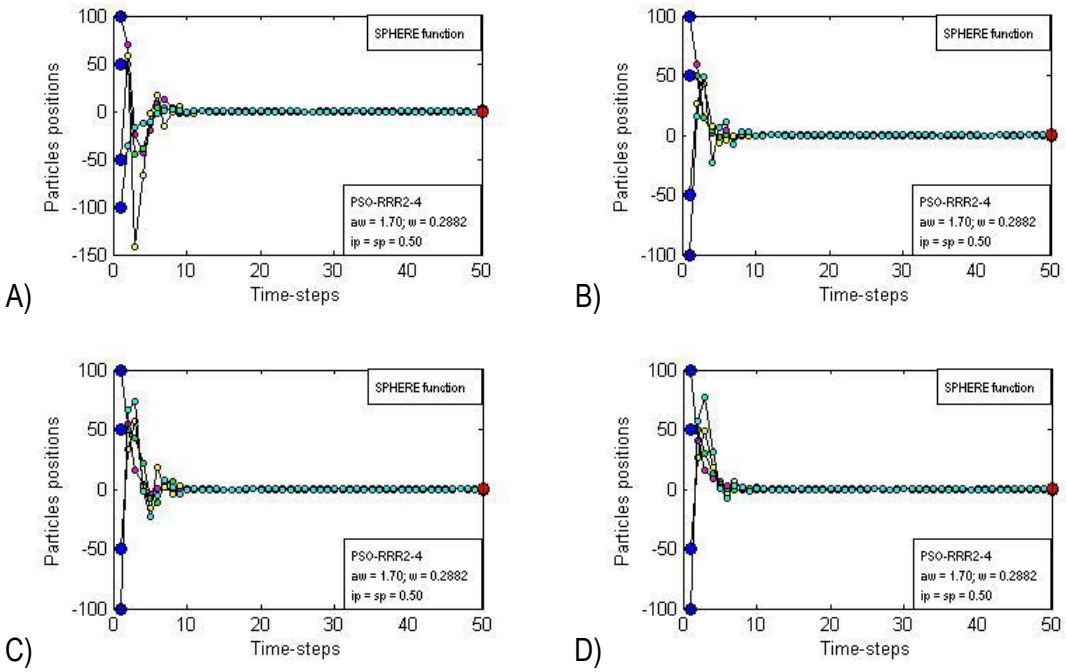


Fig. 6.67. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR2-4 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

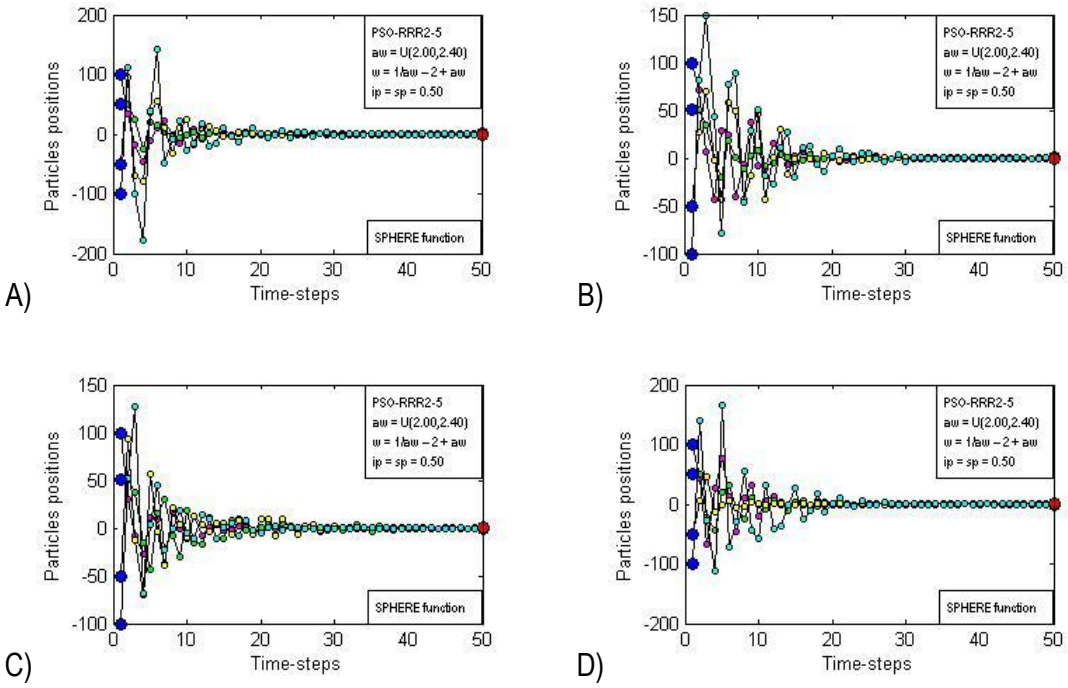


Fig. 6.68. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR2-5 algorithm with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

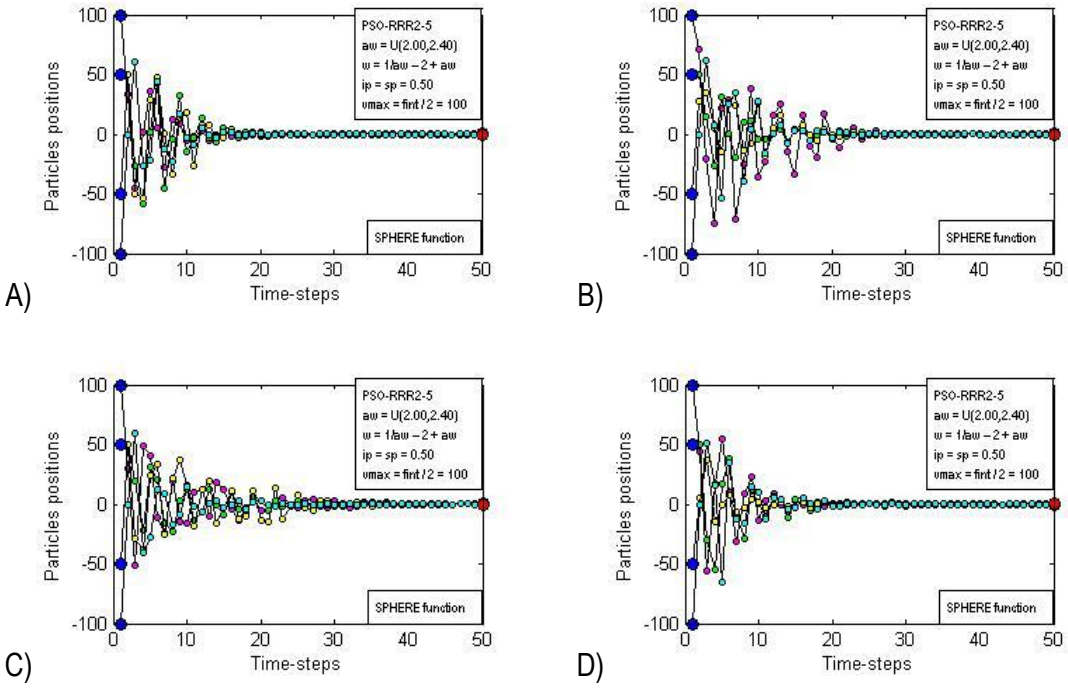


Fig. 6.69. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for the PSO-RRR2-5 algorithm with ip = sp = 0.50 and v_{max} = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

Comparing the exploration/exploitation trade-off of the PSO-RRR2-2 to those of other balanced settings, it can be observed that it is similar to those of the PSO-RRM-2 in Fig. 6.52 and the C-PSO-1 in Fig. 6.47 and Fig. 6.49, although showing more regularity for different runs, slightly less explorative behaviour, and better convergence. Conversely, it is marginally more explorative than the PSO-RRR1-2 in Fig. 6.58 and the PSO-RRM-5 in Fig. 6.55 and Fig. 6.56. Therefore it comprises a good choice if a balanced exploration/exploitation trade-off is desired. The v_{max} constraint is recommended.

The PSO-RRR2-3 in Fig. 6.66 displays exploitative behaviour. The degree of exploitation is higher than that of the C-PSO-2 in Fig. 6.48, similar to those of the PSO-RRM-3 in Fig. 6.53 and the PSO-RRR1-5 in Fig. 6.61, and lower than that of the PSO-RRR1-3 in Fig. 6.59. Therefore it comprises an exploitative optimizer, suitable for cases where fast convergence is sought while still carrying out some exploration.

The PSO-RRR2-4 in Fig. 6.67 performs too fast a convergence. Its use would be therefore limited, as almost no exploration is carried out, and even exploitation is doubtful. Hence it is discarded from the list of selected settings for the experiments to follow.

The PSO-RRR2-5 in Fig. 6.68 and Fig. 6.69 exhibits a balanced explorative/exploitative behaviour, borderline exploitative, similar to those of the PSO-RRM-5 in Fig. 6.55 and Fig. 6.56 and the PSO-RRR1-2 in Fig. 6.58. Therefore it is suitable for problems which require exploration and but at the same time fast convergence.

6.3.1.6. Other authors' settings

The settings in section 6.2.5.6 from (Trelea, 2003) –set '1'– result in a balanced explorative/exploitative behaviour –borderline exploitative– as shown in Fig. 6.70. The behaviour is similar to those of the PSO-RRM-5 in Fig. 6.55 and Fig. 6.56, the PSO-RRR1-2 in Fig. 6.58, and the PSO-RRR2-5 in Fig. 6.68 and Fig. 6.69.

The settings in section 6.2.5.6 from (Hu, Eberhart, & Shi, 2003) lead to a rather erratic explorative behaviour, which presents local explosions. The trajectories are offered in Fig. 6.71 and Fig. 6.72. The behaviour is more explorative than the PSO-RRR1-1 in Fig. 6.57, and clearly less explorative than the PSO-RRM-1 in Fig. 6.51 and the PSO-RRR2-1 in Fig. 6.62 and Fig. 6.63. Even with the aid of the v_{max} constraint, the shapes of the trajectories vary considerable for different runs.

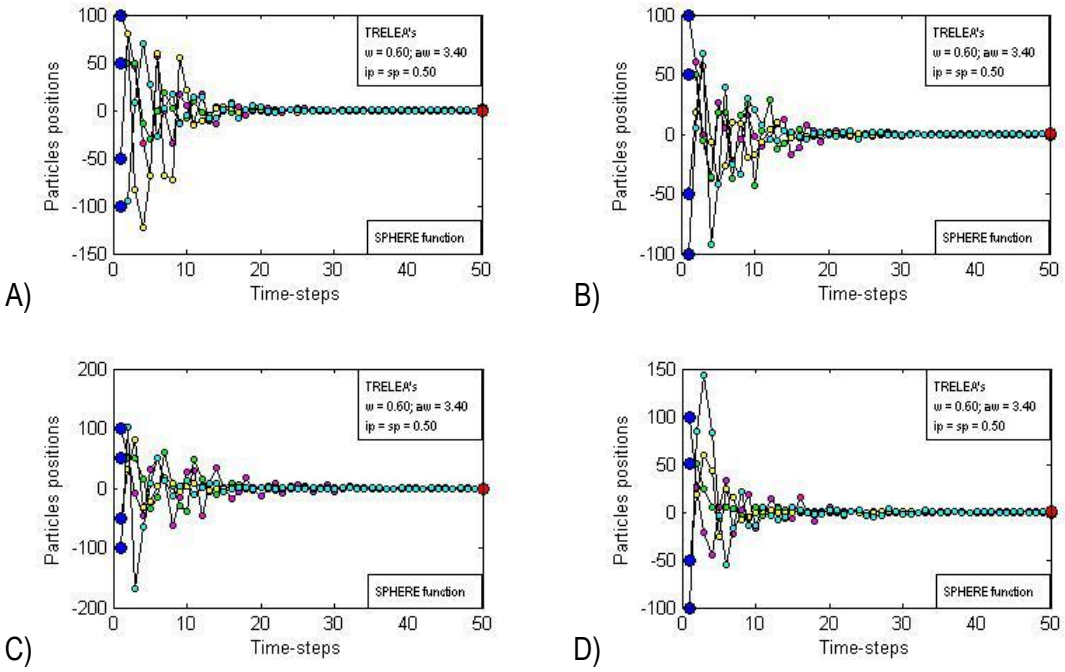


Fig. 6.70. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for (Trelea, 2003)'s settings with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

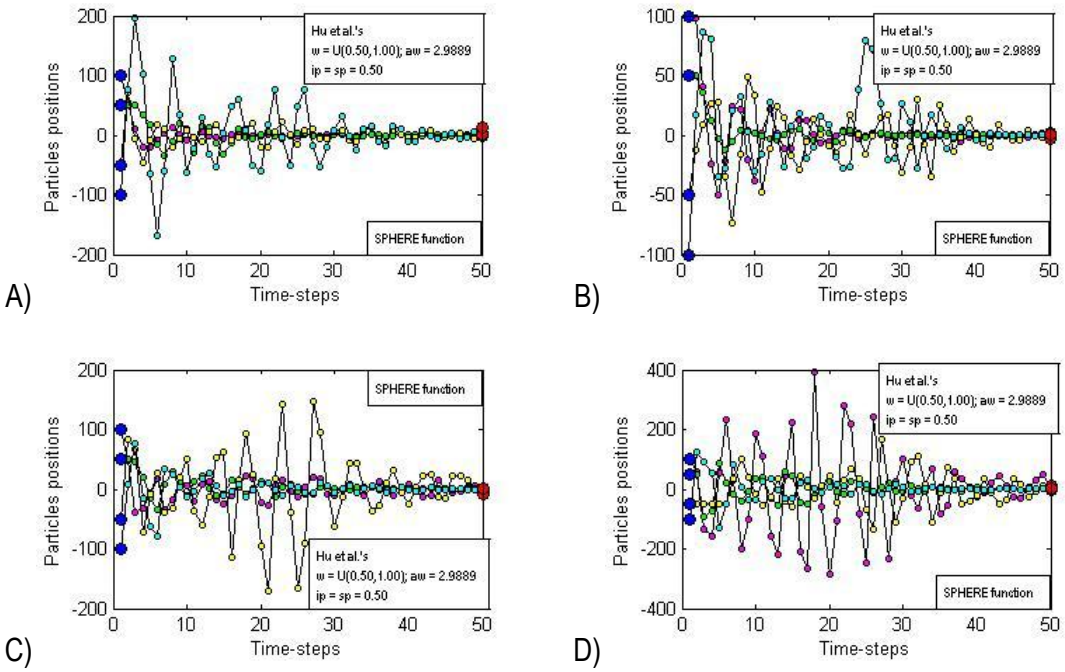


Fig. 6.71. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for (Hu, Eberhart, & Shi, 2003)'s settings with ip = sp = 0.50, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

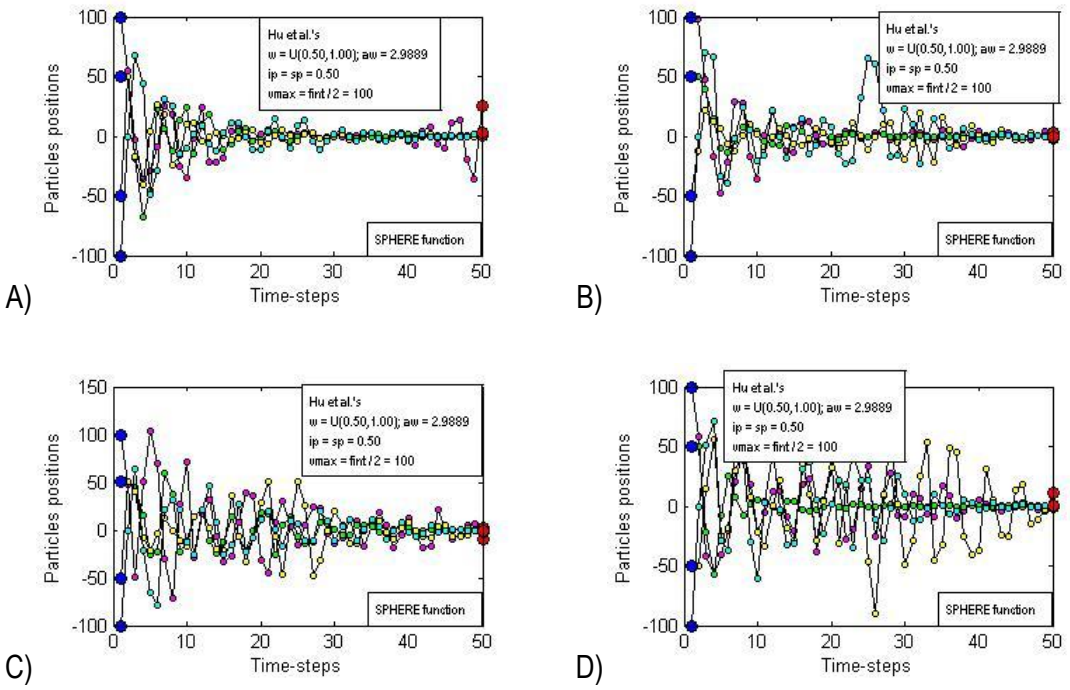


Fig. 6.72. Trajectories of four particles initialized at x = 100, x = 50, x = −50, and x = −100, for (Hu, Eberhart, & Shi, 2003)'s settings with ip = sp = 0.50 and v_{max} = $fint$ / 2 = 100, optimizing the 1-dimensional Sphere function, corresponding to four consecutive runs. The initial individual best experiences (**pbest**'s) coincide with the initial positions.

6.3.1.7. Discussion and selected sets of settings

It is important to remark that the concepts of exploration and exploitation, as discussed in section 5.6 in the previous chapter, are subjective. However, the divergent and the boundary cyclic behaviours are not. From the boundary behaviour towards convergence, it is simply stated here that the higher the speed of convergence and the narrower the part of the search-space being explored, the less explorative the behaviour. A means to quantify exploitation is proposed by Clerc (2008a), while some formal definitions of different types of exploration and exploitation with limited practical applicability are offered in (Naudts & Schippers, 1999).

Some of the preselected settings in Table 6.1 have been systematically ruled out for future experiments, as they presented similar yet somewhat less desirable behaviour than other(s) in some general sense. Thus, 8 out of the original 24 preselected settings were discarded, leaving the 16 settings in Table 6.2. Bear in mind that some of them comprise not only settings but variations to the velocity update equation. For instance, refer to the

formulations proposed here in Eqs. (6.7) to (6.10), or to that of the constricted PSO in Eqs. (5.89) and (5.90). The latter and other types of constricted PSO algorithms that vary even further from the classical PSO algorithm were proposed, studied and tested in (Clerc & Kennedy, 2002). Types other than *type 1"* invalid the notion of a position and a velocity adding up to it when moving to the next location.

Table 6.2. Qualitative exploration/exploitation trade-offs to be expected from a number of optimizers proposed in section 6.2.5. The approaches/settings are presented in a top-down fashion from more to less explorative performance.

Exploration	Balanced	Exploitation
PSO-RRM-1	PSO-RRR1-1	C-PSO-2
PSO-RRR2-1	PSO-RRM-2	PSO-RRM-3
Hu et al. (2003)	C-PSO-1	PSO-RRR2-3
	PSO-RRR2-2	PSO-RRR1-5
	PSO-RRM-5	PSO-RRR1-3
	PSO-RRR2-5	
	PSO-RRR1-2	
	Trelea (2003)	

Given that the (*type 1"*) constricted PSO can be thought of as a classical PSO by translating the coefficients' values, and that the classical PSO can be viewed as an instance of the formulations in Eqs. (6.7) to (6.10), formulations encompassing all the variations considered in Table 6.2 are presented hereafter in Eqs. (6.11) to (6.16). The user must choose the desired values for *ip* and *aw*. In the case of the C-PSO, the value of κ must also be selected by the user.

Joint formulation

User selects $ip \in [0,1)$.

$$\begin{cases} v_{ij}^{(t)} = w \cdot v_{ij}^{(t-1)} + \phi_i \cdot \left(pbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) + \phi_s \cdot \left(lbest_{ij}^{(t-1)} - x_{ij}^{(t-1)} \right) \\ \\ \phi_i = ip \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \\ \phi_s = sp \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min} \right) \cdot U_{(0,1)} \right] \\ ip \in [0,1) \quad ; \quad sp = 1 - ip \\ \\ x_{ij}^{(t)} = x_{ij}^{(t-1)} + v_{ij}^{(t)} \end{cases} \tag{6.11}$$

C-PSO

User selects aw and $\kappa \in (0,1)$, preferably $aw > 4$ (slightly) and $\kappa \to 1$.

$$cf = \begin{cases} \dfrac{2 \cdot \kappa}{aw - 2 + \sqrt{aw^2 - 4 \cdot aw}} & \text{if } aw \geq 4 \\ \kappa & \text{otherwise} \end{cases} \tag{6.12}$$

$$\begin{aligned} w &= cf \\ \phi_{\max} &= cf \cdot aw \\ \phi_{\min} &= 0 \end{aligned} \tag{6.13}$$

- C-PSO-1: κ = 0.99994; aw = 4.10; ip = 0.50.
- C-PSO-2: κ = 0.85000; aw = 4.10; ip = 0.50.

PSO-RRM

User selects $aw \in (2.00, 4.00)$, preferably $2.60 \leq aw \leq 3.60$.

$$\begin{aligned} w &= \frac{aw}{2} - 1 \\ \phi_{\max} &= aw \\ \phi_{\min} &= 0 \end{aligned} \tag{6.14}$$

- PSO-RRM-1: aw = 3.60; ip = 0.50.
- PSO-RRM-2: aw = 3.40; ip = 0.50.
- PSO-RRM-3: aw = 3.00; ip = 0.50.
- PSO-RRM-5: $aw = U_{(3.00,3.60)}$; ip = 0.50.

Therefore the PSO-RRM-5 is a variation of the general PSO-RRM, as aw is not selected by the user but randomly generated within an arbitrary interval.

PSO-RRR1

User selects $aw \in (1.00, 2.00)$, preferably $1.30 \leq aw \leq 1.80$.

$$\begin{aligned} w &= aw - 1 \\ \phi_{\max} &= \frac{3}{2} \cdot (w+1) \\ \phi_{\min} &= \frac{1}{2} \cdot (w+1) \end{aligned} \tag{6.15}$$

- PSO-RRR1-1: aw = 1.80; ip = 0.50.
- PSO-RRR1-2: aw = 1.70; ip = 0.50.
- PSO-RRR1-3: aw = 1.50; ip = 0.50.
- PSO-RRR1-5: $aw = U_{(1.50,1.80)}$; ip = 0.50.

Therefore the PSO-RRR1-5 is a variation of the general PSO-RRR1, as aw is not selected by the user but randomly generated within an arbitrary interval.

PSO-RRR2

User selects $aw \in (1.000, 2.618)$, preferably $1.70 \leq aw \leq 2.40$.

$$\begin{aligned} w &= \frac{1}{aw} - 2 + aw \\ \phi_{\max} &= 2 \cdot (w+1) \\ \phi_{\min} &= 2 \cdot aw - \phi_{\max} \end{aligned} \tag{6.16}$$

- PSO-RRR2-1: aw = 2.40; ip = 0.50.
- PSO-RRR2-2: aw = 2.25; ip = 0.50.
- PSO-RRR2-3: aw = 2.00; ip = 0.50.
- PSO-RRR2-5: $aw = U_{(2.00,2.40)}$; ip = 0.50.

Therefore the PSO-RRR2-5 is a variation of the general PSO-RRR2, as aw is not selected by the user but randomly generated within an arbitrary interval.

6.3.2. Full swarm and multidimensional space

Of course, the settings in Table 6.2 are discrete and to some extent arbitrary. The regions of the 'ϕ–w' plane from where they can be chosen –and the behaviour to be expected from the different regions– for the formulations proposed in this chapter were discussed in sections 6.2.2.1 for the PSO-RRM, 6.2.2.2 for the PSO-RRR1, and 6.2.2.3 for the PSO-RRR2. Refer to Fig. 6.2, Fig. 6.3, and Fig. 6.13 for visualization.

Since the overall speed of convergence not only depends on the rapidness of the spread of information within the swarm but also on the coefficients' settings, different combinations of neighbourhood topologies and coefficients' settings should be studied. For instance, some settings may speed up convergence while local neighbourhoods delay it. Hence raises the following question: even if a given speed of convergence could be somehow chosen and different combinations of coefficients and neighbourhoods leading to it were available, is it preferable to:

- use rather *exploitative coefficients* and *more local neighbourhoods*,
- or more *explorative coefficients* and *more global neighbourhoods*?

In the first case, the exploration is achieved due to a lower number of interconnections between particles. However, the local best experiences are exploited. That is, exploration consists of exploitation of numerous best experiences. In the second case, exploration is performed in a more general sense.

Therefore, to analyze the results obtained from different coefficients' settings, at least one rather local and one rather global neighbourhood topologies should be considered in the same way that explorative, balanced and exploitative coefficients are considered in Table 6.2. Thus, some of those settings are tested on a set of benchmark problems, both for the original global topology and for the classical ring topology with two neighbours (i.e. three-particle neighbourhoods). Due to time and space constraints, only three settings from Table 6.2 are tested hereafter: PSO-RRR2-1; PSO-RRR1-1; and C-PSO-1.

6.3.2.1. Measures of clustering and evolution

Due to the population-based nature of the PSO method, keeping track of what is happening throughout the search is not as straightforward as it may be for single-solution

algorithms. Seven so-called measures of error are proposed hereafter so as to gather useful information throughout the search. Even though they are not measures of error in a strict sense, they are given that name because they are also to be used for the development of termination conditions. They are divided in two main groups: 3 measures are computed *within the same time-step* whereas 4 measures are computed considering consecutive time-steps. In turn, there are also two sub-groups in each group: one involves information about the particles' positions; and the other involves information with regards to the particles' conflicts. The 3 measures within the same time-step are shown in Eqs. (6.17) to (6.19), and the 4 measures between consecutive time-steps are shown in Eqs. (6.20) to (6.23). The variable *tref* is a number of time-steps to be set by the user, over which the measures of error are averaged so as to smoothen the oscillations of their curves. The oscillations are due to the population-based nature and swarming behaviour of the paradigm. A trend line would probably be more elegant, but also considerably more expensive. Besides, a trend line would require continuous recalculations as the search progresses. Therefore the proposed measures are averaged in *tref*.

Note that if *tref* > 1, the measures are not really within the same time-step or between consecutive time-steps. Nevertheless, those headings are maintained because they give an idea of their meaning. That is to say, measures within the same time-step are only possible for population-based methods, whereas measures between consecutive time-steps are more traditional.

In this optimizer, the user has the option to choose whether these measures are absolute or relative. The 'position-based' measures can always be either one, whereas the 'conflict-based' measures can only be relative if a small sub-swarm is activated to search for the maximum rather than the minimum (i.e. *cgworst*). Thus, the concept of relativeness is not the standard one –where an error is related to an approximation of the correct value– but rather a sort of normalization. The expressions in Eqs. (6.17) to (6.23) are relative. For their absolute counterpart, remove $\left(cgworst^{(t)} - cgbest^{(t)}\right)$ from the 'conflict-based' measures, and/or $\left(x_{j\max} - x_{j\min}\right)$ from the 'position-based' ones. The optimizer also allows the user to choose whether to compute these measures using the 'positions swarm' (i.e. **p**) or the 'memory swarm' (i.e. **pbest**). Obviously, there are typically fewer oscillations in the latter case.

Within a single time-step

cb_me (conflict-based mean error): average in the last *tref* time-steps of the difference between the current average conflict (among all particles') and the best conflict found so far, related to the difference between the worst and the best conflicts found so far:

$$\mathrm{cb_me}^{(t)} = \frac{\sum_{i=t-tref+1}^{t} \mathrm{abs}\left(\frac{\sum_{j=1}^{m} c_j^{(i)}}{m} - cgbest^{(i)} \right)}{tref \cdot \left(cgworst^{(t)} - cgbest^{(t)} \right)} = \frac{\sum_{i=t-tref+1}^{t} \mathrm{abs}\left(\bar{c}^{(i)} - cgbest^{(i)} \right)}{tref \cdot \left(cgworst^{(t)} - cgbest^{(t)} \right)} \tag{6.17}$$

pb_me (position-based mean error): average in *tref* of the square root of the average (among all particles) of the squared normalized (with respect to the feasible intervals and to the number of dimensions) distance between each particle and the best solution:

$$\mathrm{pb_me}^{(t)} = \frac{\sum_{i=t-tref+1}^{t} \sqrt{\frac{\sum_{j=1}^{n}\sum_{k=1}^{m}\left(\frac{x_{kj}^{(i)} - gbest_j^{(i)}}{x_{j\max} - x_{j\min}} \right)^2}{m \cdot n}}}{tref} = \frac{\sum_{i=t-tref+1}^{t} \sqrt{\sum_{j=1}^{n} \frac{\sum_{k=1}^{m}\left(x_{kj}^{(i)} - gbest_j^{(i)} \right)^2}{m \cdot \left(x_{j\max} - x_{j\min} \right)^2}}}{tref \cdot \sqrt{n}} \tag{6.18}$$

pb_cge (position-based centre of gravity error): average in *tref* of the square root of the squared normalized (with respect to the feasible intervals and to the number of dimensions) distance between the centre of gravity of the swarm and the best solution:

$$\mathrm{pb_cge}^{(t)} = \frac{\sum_{i=t-tref+1}^{t} \sqrt{\frac{\sum_{j=1}^{n}\left(\frac{cg_j^{(i)} - gbest_j^{(i)}}{x_{j\max} - x_{j\min}} \right)^2}{n}}}{tref} = \frac{\sum_{i=t-tref+1}^{t} \sqrt{\sum_{j=1}^{n}\left(\frac{cg_j^{(i)} - gbest_j^{(i)}}{x_{j\max} - x_{j\min}} \right)^2}}{tref \cdot \sqrt{n}} \tag{6.19}$$

Between consecutive time-steps

cb_av (conflict-based average evolution error): average in the last *tref* time-steps of the difference between the current average conflict (among all particles') and the preceding one, related to the difference between the worst and the best conflicts found so far:

$$\text{cb_av}^{(t)} = \frac{\sum_{i=t-tref+1}^{t} \text{abs}\left(\frac{\sum_{j=1}^{m} c_j^{(i)}}{m} - \frac{\sum_{j=1}^{m} c_j^{(i-1)}}{m} \right)}{tref \cdot \left(cgworst^{(t)} - cgbest^{(t)} \right)} = \frac{\sum_{i=t-tref+1}^{t} \text{abs}\left(\bar{c}^{(i)} - \bar{c}^{(i-1)} \right)}{tref \cdot \left(cgworst^{(t)} - cgbest^{(t)} \right)} \tag{6.20}$$

cb_best (conflict-based solution evolution error): average in the last *tref* time-steps of the difference between the previous best conflict and the current one, related to the difference between the worst and the best conflicts found so far:

$$\text{cb_best}^{(t)} = \frac{\sum_{i=t-tref+1}^{t} \left(cgbest^{(i-1)} - cgbest^{(i)} \right)}{tref \cdot \left(cgworst^{(t)} - cgbest^{(t)} \right)} = \frac{\left(cgbest^{(t-tref)} - cgbest^{(t)} \right)}{tref \cdot \left(cgworst^{(t)} - cgbest^{(t)} \right)} \tag{6.21}$$

pb_cg (position-based centre of gravity evolution error): average in *tref* of the square root of the squared normalized (with respect to the feasible intervals and to the number of dimensions) distance between the current centre of gravity and the preceding one:

$$\text{pb_cg}^{(t)} = \frac{\sum_{i=t-tref+1}^{t} \sqrt{\frac{\sum_{j=1}^{n} \left(\frac{cg_j^{(i)} - cg_j^{(i-1)}}{x_{j\max} - x_{j\min}} \right)^2}{n}}}{tref} = \frac{\sum_{i=t-tref+1}^{t} \sqrt{\sum_{j=1}^{n} \left(\frac{cg_j^{(i)} - cg_j^{(i-1)}}{x_{j\max} - x_{j\min}} \right)^2}}{tref \cdot \sqrt{n}} \tag{6.22}$$

pb_gbest (position-based solution evolution error): average in *tref* of the square root of the squared normalized (with respect to the feasible intervals and to the number of dimensions) distance between the location of the current and the preceding best solutions:

$$\text{pb_gbest}^{(t)} = \frac{\sum_{i=t-tref+1}^{t} \sqrt{\frac{\sum_{j=1}^{n} \left(\frac{gbest_j^{(i)} - gbest_j^{(i-1)}}{x_{j\max} - x_{j\min}} \right)^2}{n}}}{tref} \tag{6.23}$$

$$\text{pb_gbest}^{(t)} = \frac{\sum_{i=t-tref+1}^{t} \sqrt{\sum_{j=1}^{n} \left(\frac{gbest_j^{(i)} - gbest_j^{(i-1)}}{x_{j\max} - x_{j\min}} \right)^2}}{tref \cdot \sqrt{n}}$$

As can be observed, the idea is to try to make the measures comparable regardless of dimensionality and sizes of the feasible intervals, both between and within problems.

6.3.2.2. Experimental results

The PSO-RRR2-1, PSO-RRR1-1, C-PSO-1, and a Multi-Swarm algorithm combining the three of them are tested on the suite of side-constrained benchmark problems shown in Appendix II, for 2-, 10-, and 30-dimensional spaces. A global topology and a ring topology with two neighbours are tested for each case.

Every run is performed with a swarm of 50 particles for a length of 10000 time-steps. Intermediate results at 1000th time-step are also provided.

The particles' positions are initialized by generating 1000 independent Latin Hypercube Samplings (LHSs), and selecting the one with the maximum minimum distance between particles. Velocities are initialized to zero, and the individual best experiences are initialized instead. Every best experience is initialized at exactly the same distance from its corresponding particle. Each component of this distance is calculated as the corresponding feasible interval divided by twice the number of particles in the swarm. The sign of the component, and hence the direction of the distance vector, are randomly generated. For each pair '**p**–**pbest**', a comparison is performed so that the best one becomes (or stays) **pbest** and the other becomes (or stays) **p** before the search begins. Thus, every particle starts the search with the same, moderate acceleration towards its **pbest** (the acceleration towards its **lbest** will depend on the neighbourhood structure). Therefore, in the end, the particles' initialization will most likely not be a LHS. For the Multi-Swarm algorithm, each sub-swarm is initialized independently.

This optimizer allows the user to choose whether to activate the *cut-off at the boundary* technique or the *bisection* method to handle interval constraints (see section 4.4.8.4.). Although a proper numerical analysis has not been performed, preliminary tests did not yield very good results on the use of these techniques because of the bias (in agreement with (Clerc, 2007)). Hence interval constraints are simply treated as any other inequality constraint, and infeasible particles are not evaluated (therefore those experiences are not stored in memory). This way, the normal dynamics of the swarm is least disrupted, and the particles could approach the solution from every direction without losing momen-

tum too quickly when the solution lies near the boundaries. In addition, the fact that infeasible particles are not evaluated allows tracking the number of times a particle leaves the feasible space, as the algorithm stores the number of function evaluations (FEs). The results in these experiments (refer to digital appendix) show that this number greatly depends on the coefficients' settings, neighbourhood topology, and objective function. The number of times that a particle crossed the boundaries varies from around 150 to 10000 (for 50 particles and 10000 times-steps).

The position-based measures of error are relative, whereas the conflict-based measures are absolute; *tref* = 10; the measures of error are computed on the individual best experiences (rather than on the current ones), and only 'pb_me' is provided in the tables.

A run is considered successful if the error is no greater than 0.0001, and the statistics are calculated out of 25 runs. The random number generator is reset to its initial state only before the first run of every experiment. The results are gathered in Table 6.3 to Table 6.17 and Fig. 6.74 to Fig. 6.88. In addition, the evolution of 'pb_me' and 'pb_cge' for the 'PSO-RRR1-1 Global' optimizing the 30D Sphere function is shown in Fig. 6.73. The full output files are provided in *.xlsx and *.mat formats in the digital appendix, where a great amount of information with regards to the search can be obtained.

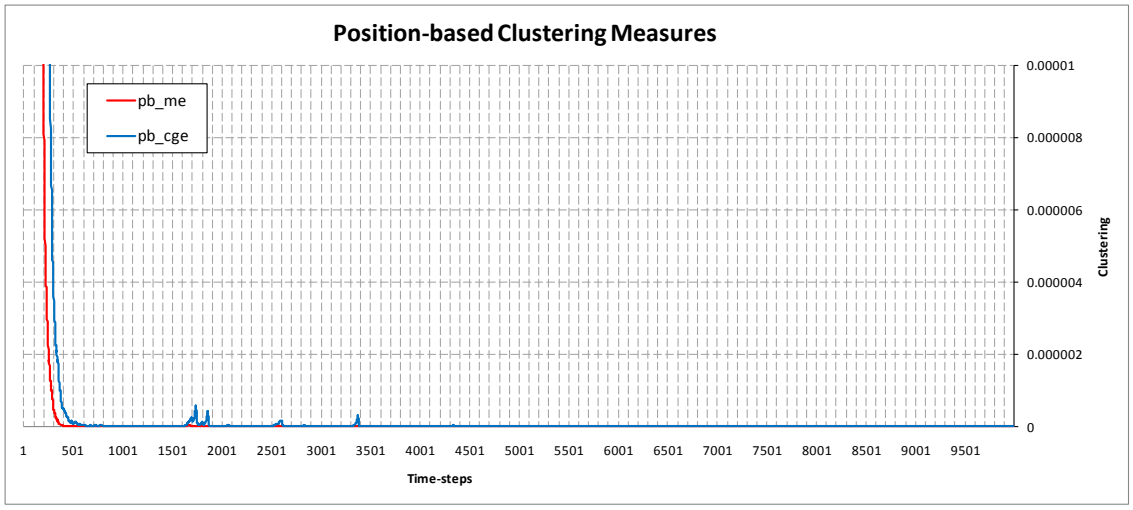


Fig. 6.73. Evolution of the mean measures of clustering 'pb_cge' and 'pb_me' for the 'PSO-RRR1-1 Global' optimizing the 30-dimensional Sphere function. It can be seen that particles imploded in less than 500 time-steps. Improvement is still possible (see Fig. 6.76) because of the simplicity of the function, but the swarm moves 'as one'.

Table 6.3. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Sphere function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.74E-57	3.42E-54	4.65E-53	5.96E-52	4.79E-18	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.61E-53	8.64E-49	7.07E-46	1.26E-44	1.55E-18	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	5.19E-88	2.30E-85	3.33E-84	4.17E-83	3.43E-37	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.09E-82	2.99E-80	2.13E-78	2.27E-77	1.52E-37	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.74E-91	5.15E-88	2.06E-84	5.14E-83	9.16E-30	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	5.58E-82	3.44E-78	5.67E-76	6.45E-75	3.58E-32	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.38E-90	4.67E-86	1.06E-83	2.16E-82	1.70E-20	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.39E-85	4.26E-78	9.85E-77	1.83E-75	2.68E-20	-

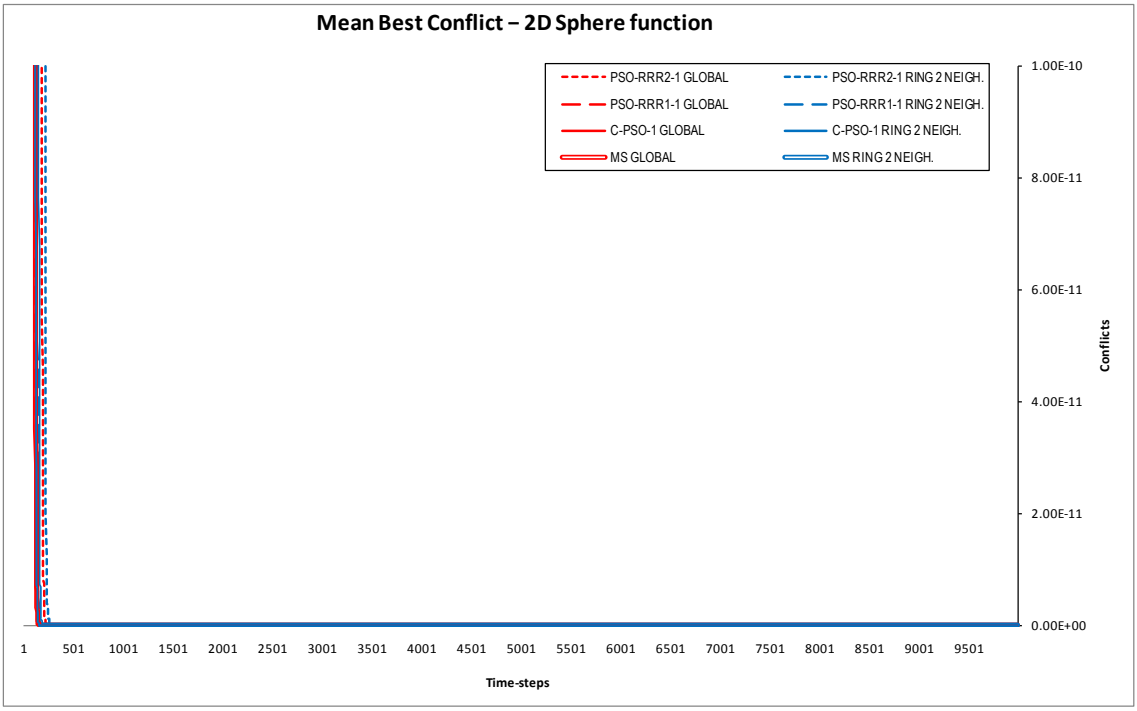


Fig. 6.74. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 6.3. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.4. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Sphere function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	2.04E-256	3.90E-250	3.93E-247	8.75E-246	2.28E-126	100
			1000	4.97E-24	5.49E-23	1.25E-22	1.06E-21	8.08E-14	-
	RING	nn = 2	10000	1.29E-145	2.82E-143	3.37E-141	5.90E-140	1.13E-73	100
			1000	1.78E-13	3.67E-12	5.33E-12	2.02E-11	8.77E-09	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.75E-67	1.61E-65	5.10E-65	3.50E-64	5.98E-35	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.86E-35	2.94E-33	4.46E-33	1.53E-32	1.01E-19	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.30E-51	6.84E-50	3.49E-49	5.15E-48	7.98E-27	-
	RING	nn = 2	10000	1.13E-280	2.23E-277	4.17E-274	5.97E-273	2.11E-140	100
			1000	7.36E-27	3.12E-25	4.91E-25	2.77E-24	1.50E-15	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.21E-58	2.00E-56	2.03E-55	1.42E-54	2.17E-18	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.20E-153	100
			1000	1.41E-32	1.17E-30	1.06E-29	9.91E-29	2.68E-11	-

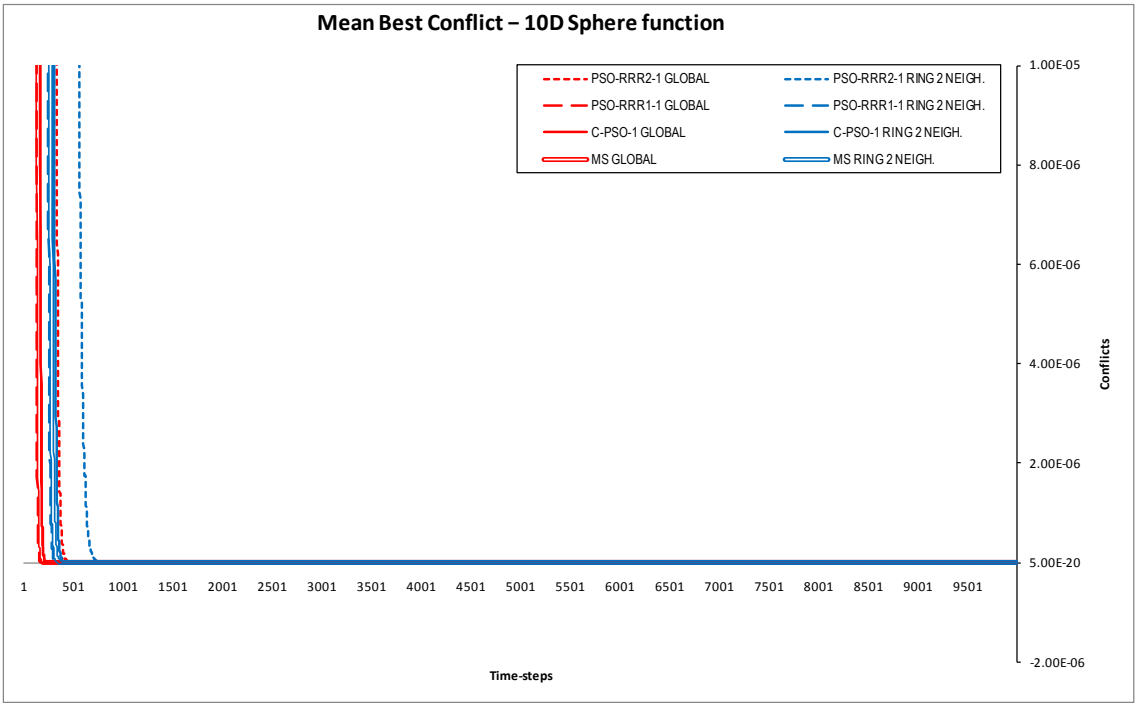


Fig. 6.75. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 6.4. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.5. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Sphere function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.22E-87	3.29E-84	3.07E-82	6.31E-81	2.12E-45	100
			1000	3.49E-06	1.85E-05	4.08E-05	2.77E-04	1.84E-06	-
	RING	nn = 2	10000	3.77E-43	1.90E-42	7.86E-42	6.78E-41	9.24E-25	100
			1000	1.23E-01	2.84E-01	3.12E-01	6.69E-01	2.14E-04	-
PSO-RRR1-1	GLOBAL		10000	4.06E-07	3.79E-04	9.89E-02	2.39E+00	4.69E-17	32
			1000	5.57E-06	1.64E-03	2.70E-01	4.03E+00	5.93E-11	-
	RING	nn = 2	10000	8.13E-144	7.26E-142	5.86E-141	6.69E-140	1.20E-74	100
			1000	1.73E-11	7.83E-11	8.50E-11	2.40E-10	2.05E-09	-
C-PSO-1	GLOBAL		10000	3.05E-220	2.21E-212	1.42E-207	3.49E-206	1.57E-108	100
			1000	8.53E-20	1.09E-17	1.04E-16	9.10E-16	1.33E-12	-
	RING	nn = 2	10000	5.68E-96	1.67E-94	1.65E-93	3.57E-92	7.60E-51	100
			1000	7.53E-07	3.07E-06	3.62E-06	1.35E-05	5.46E-07	-
Multi-Swarm	GLOBAL		10000	1.10E-181	4.68E-172	1.95E-166	4.21E-165	7.90E-88	100
			1000	4.53E-17	1.52E-14	4.02E-11	1.00E-09	3.63E-10	-
	RING	nn = 2	10000	3.13E-113	7.02E-109	2.72E-107	5.96E-106	6.93E-57	100
			1000	2.83E-08	9.73E-08	2.55E-07	3.82E-06	1.75E-06	-

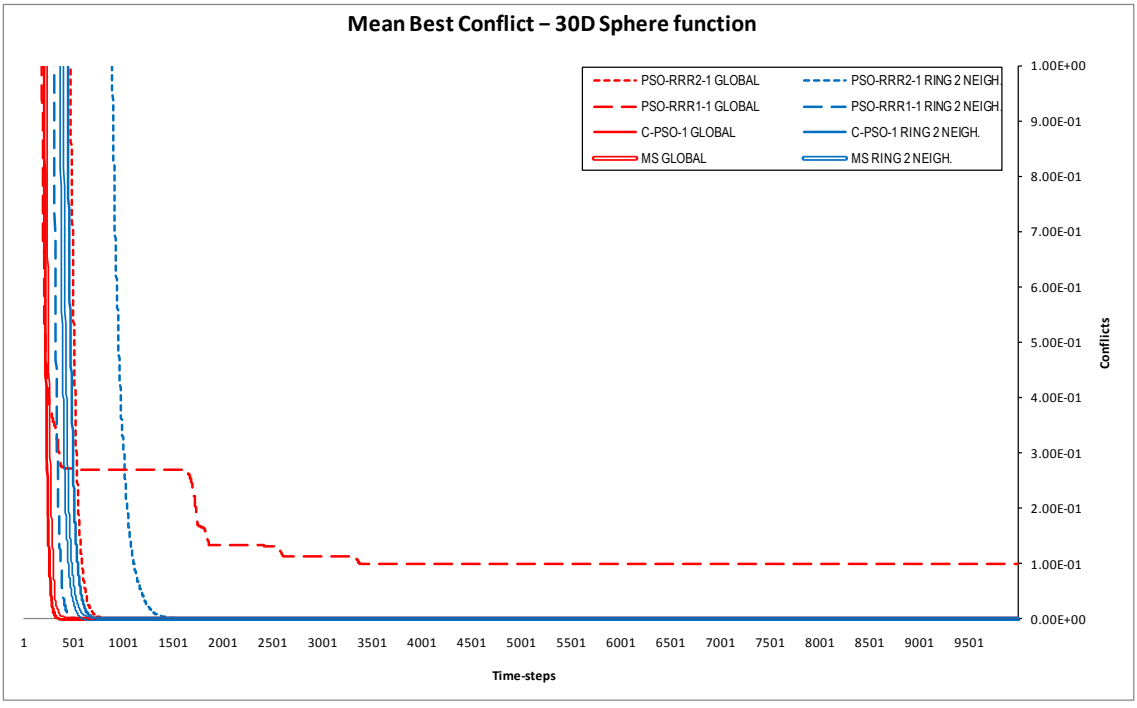


Fig. 6.76. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 6.5. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.6. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	1.54E-30	3.01E-28	4.82E-26	3.96E-25	3.72E-08	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	3.11E-20	3.03E-16	3.31E-15	3.09E-14	6.95E-06	-
PSO-RRR1-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.70E-20	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	4.93E-32	4.78E-27	6.69E-25	7.34E-24	1.27E-11	-
C-PSO-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.29E-11	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	5.77E-21	1.65E-15	6.19E-14	1.30E-12	2.76E-06	-
Multi-Swarm	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.05E-09	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	9.00E-24	2.19E-18	3.50E-17	1.78E-06	-

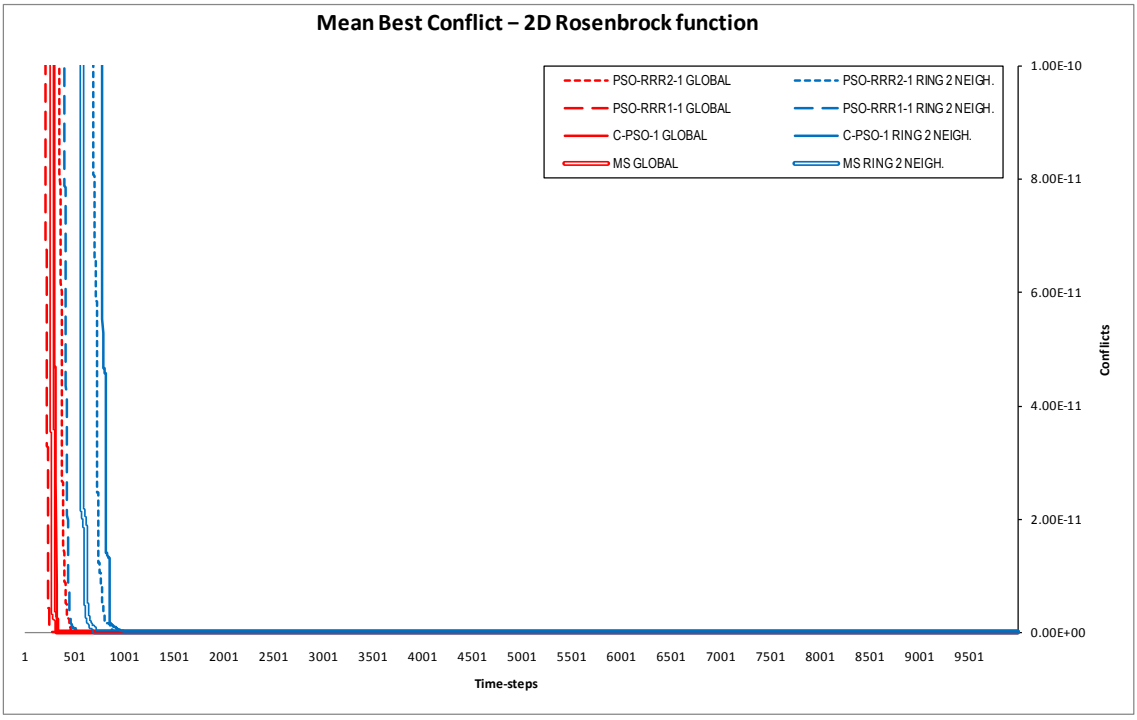


Fig. 6.77. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 6.6. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.7. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.09E-06	2.72E-04	6.38E-01	3.99E+00	6.79E-03	32
			1000	2.22E-02	2.47E+00	5.31E+00	6.85E+01	2.34E-03	-
	RING	nn = 2	10000	6.79E-05	1.64E-02	1.82E-02	7.14E-02	7.64E-03	4
			1000	9.88E-03	1.64E+00	2.03E+00	5.50E+00	1.63E-02	-
PSO-RRR1-1	GLOBAL		10000	1.13E-28	8.73E-01	1.64E+00	3.99E+00	1.08E-03	44
			1000	6.53E-06	1.46E+00	1.90E+00	4.99E+00	2.85E-03	-
	RING	nn = 2	10000	2.20E-10	3.45E-09	3.21E-08	5.63E-07	2.39E-03	100
			1000	5.10E-04	1.17E+00	1.30E+00	4.19E+00	1.11E-02	-
C-PSO-1	GLOBAL		10000	1.18E-10	4.49E-06	4.79E-01	3.99E+00	8.56E-03	76
			1000	2.73E-03	5.03E-01	7.26E+00	8.06E+01	6.76E-03	-
	RING	nn = 2	10000	1.23E-08	1.29E-03	1.61E-01	3.99E+00	7.32E-03	8
			1000	1.18E-02	3.08E+00	2.69E+00	5.08E+00	1.90E-02	-
Multi-Swarm	GLOBAL		10000	5.91E-12	2.94E-02	1.02E+00	5.59E+00	7.53E-03	16
			1000	9.37E-05	6.67E-01	1.47E+00	6.46E+00	7.93E-03	-
	RING	nn = 2	10000	4.01E-09	8.03E-07	1.61E-01	3.99E+00	8.87E-03	80
			1000	2.63E-03	1.62E+00	1.82E+00	5.11E+00	1.79E-02	-

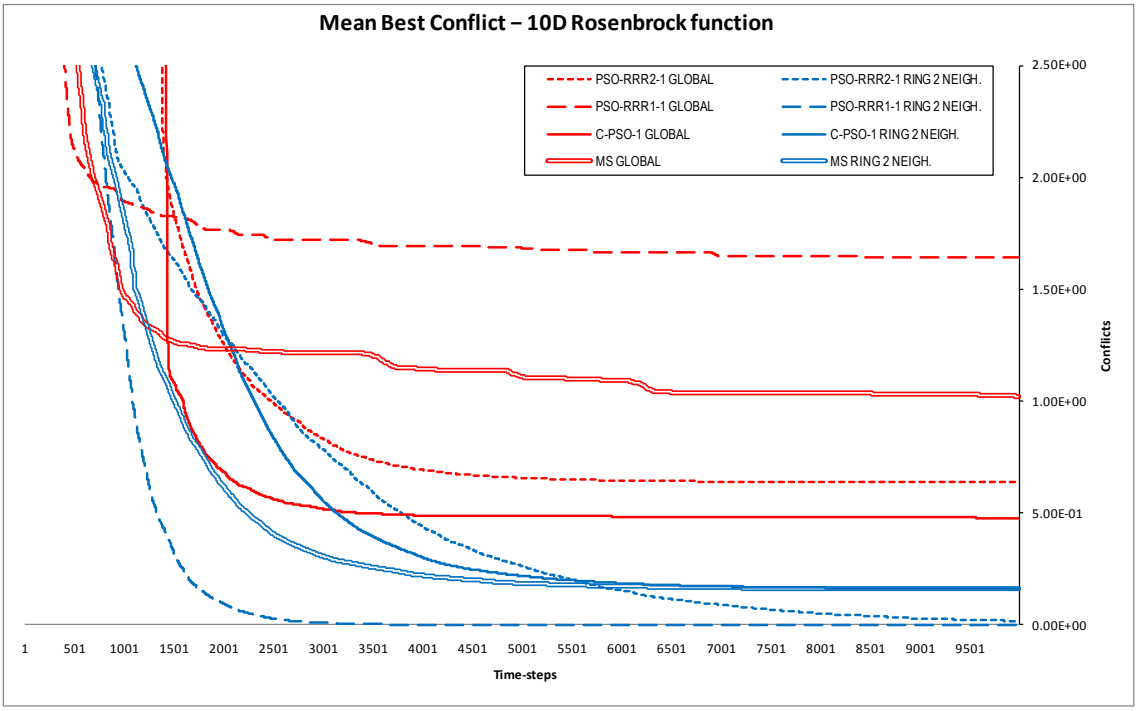


Fig. 6.78. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 6.7. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.8. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.41E-04	1.27E+01	1.03E+01	1.88E+01	1.45E-03	0
			1000	8.48E+00	2.80E+01	5.20E+01	1.24E+02	2.95E-04	-
	RING	nn = 2	10000	1.14E-01	1.00E+01	1.06E+01	2.31E+01	1.13E-02	0
			1000	4.82E+01	1.40E+02	1.48E+02	3.09E+02	1.57E-02	-
PSO-RRR1-1	GLOBAL		10000	2.27E+01	9.15E+01	1.06E+02	3.67E+02	3.98E-11	0
			1000	2.43E+01	9.38E+01	1.10E+02	3.69E+02	2.84E-08	-
	RING	nn = 2	10000	8.78E-03	7.24E+00	7.16E+00	1.91E+01	2.15E-03	0
			1000	8.28E+00	2.61E+01	4.28E+01	1.77E+02	4.38E-03	-
C-PSO-1	GLOBAL		10000	1.17E-05	3.90E-02	1.05E+00	4.02E+00	5.58E-03	8
			1000	1.55E+00	2.22E+01	3.58E+01	1.79E+02	9.02E-04	-
	RING	nn = 2	10000	2.89E-03	6.94E-01	3.39E+00	1.79E+01	7.42E-03	0
			1000	1.32E+01	2.89E+01	5.04E+01	1.45E+02	1.30E-02	-
Multi-Swarm	GLOBAL		10000	4.21E-08	1.67E+01	2.70E+01	7.68E+01	3.90E-03	4
			1000	2.33E-02	2.30E+01	4.38E+01	1.36E+02	2.59E-03	-
	RING	nn = 2	10000	9.14E-03	7.09E+00	6.59E+00	1.46E+01	3.01E-03	0
			1000	6.26E+00	7.08E+01	5.33E+01	8.71E+01	5.27E-03	-

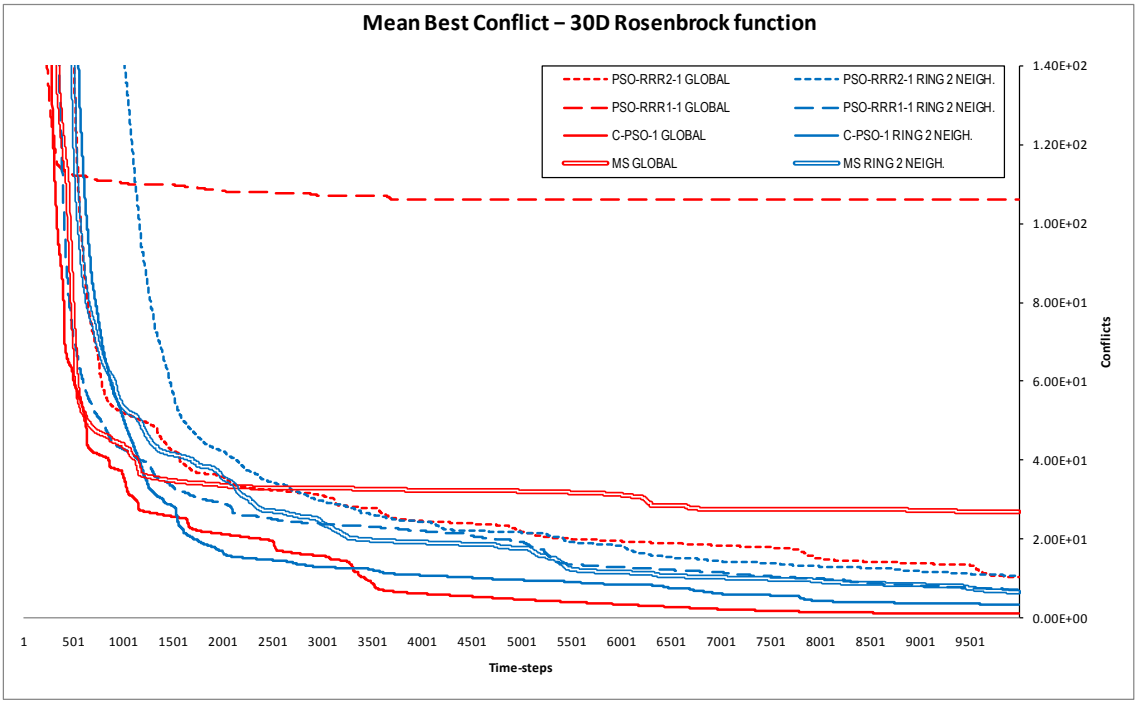


Fig. 6.79. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 6.8. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.9. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.07E-10**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.42E-11	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**9.93E-11**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	-
PSO-RRR1-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**8.09E-11**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.09E-11	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**9.05E-11**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.05E-11	-
C-PSO-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**7.79E-11**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.79E-11	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**8.52E-11**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.52E-11	-
Multi-Swarm	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**9.93E-11**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**8.73E-11**	**100**
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-

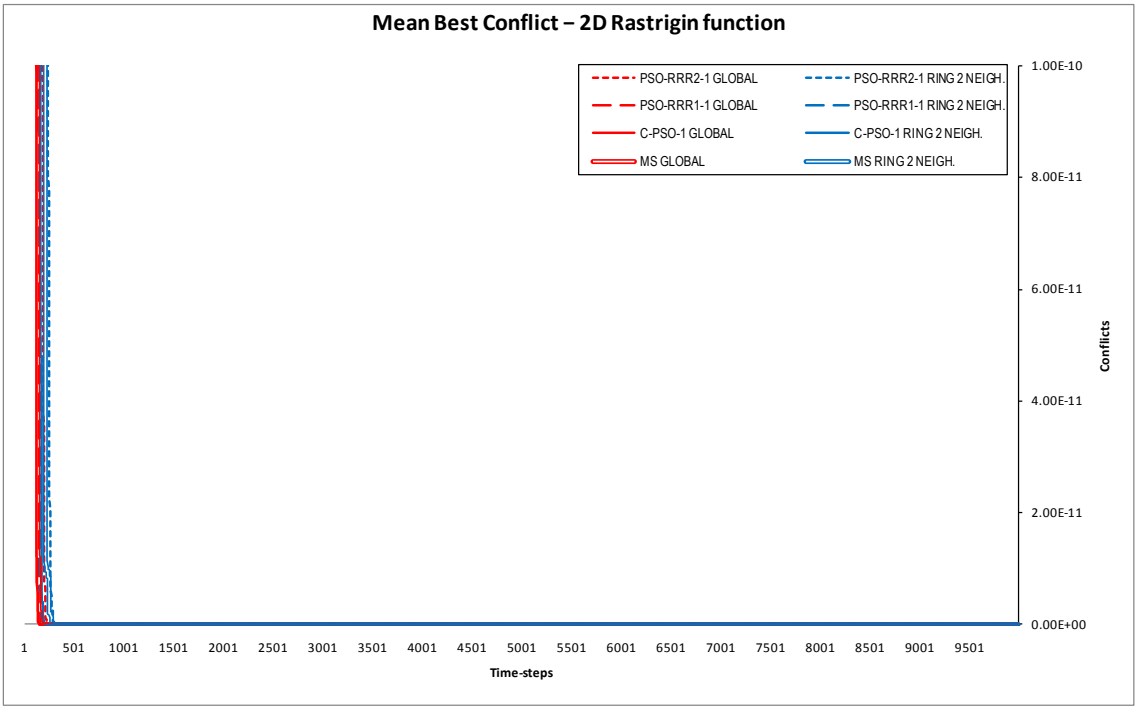


Fig. 6.80. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 6.9. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.10. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 10D				OPTIMUM =	0
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	9.95E-01	2.98E+00	2.95E+00	6.96E+00	3.48E-04	0
			1000	9.95E-01	2.98E+00	3.02E+00	6.96E+00	2.20E-03	-
	RING	nn = 2	10000	0.00E+00	1.99E+00	2.15E+00	4.97E+00	2.71E-02	20
			1000	1.99E+00	4.22E+00	4.43E+00	7.96E+00	3.54E-02	-
PSO-RRR1-1	GLOBAL		10000	5.97E+00	1.19E+01	1.35E+01	2.49E+01	1.21E-11	0
			1000	5.97E+00	1.19E+01	1.35E+01	2.49E+01	1.47E-11	-
	RING	nn = 2	10000	9.95E-01	4.97E+00	5.18E+00	1.09E+01	3.82E-02	0
			1000	2.98E+00	7.96E+00	7.94E+00	1.37E+01	4.30E-02	-
C-PSO-1	GLOBAL		10000	1.99E+00	3.98E+00	4.93E+00	1.09E+01	1.92E-11	0
			1000	1.99E+00	4.97E+00	5.17E+00	1.19E+01	8.71E-04	-
	RING	nn = 2	10000	0.00E+00	2.98E+00	2.79E+00	4.97E+00	2.93E-02	12
			1000	1.99E+00	3.98E+00	4.55E+00	7.96E+00	3.79E-02	-
Multi-Swarm	GLOBAL		10000	1.99E+00	3.98E+00	4.78E+00	1.49E+01	1.84E-11	0
			1000	1.99E+00	3.98E+00	5.13E+00	1.49E+01	5.46E-04	-
	RING	nn = 2	10000	0.00E+00	2.98E+00	2.75E+00	6.96E+00	2.90E-02	4
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-

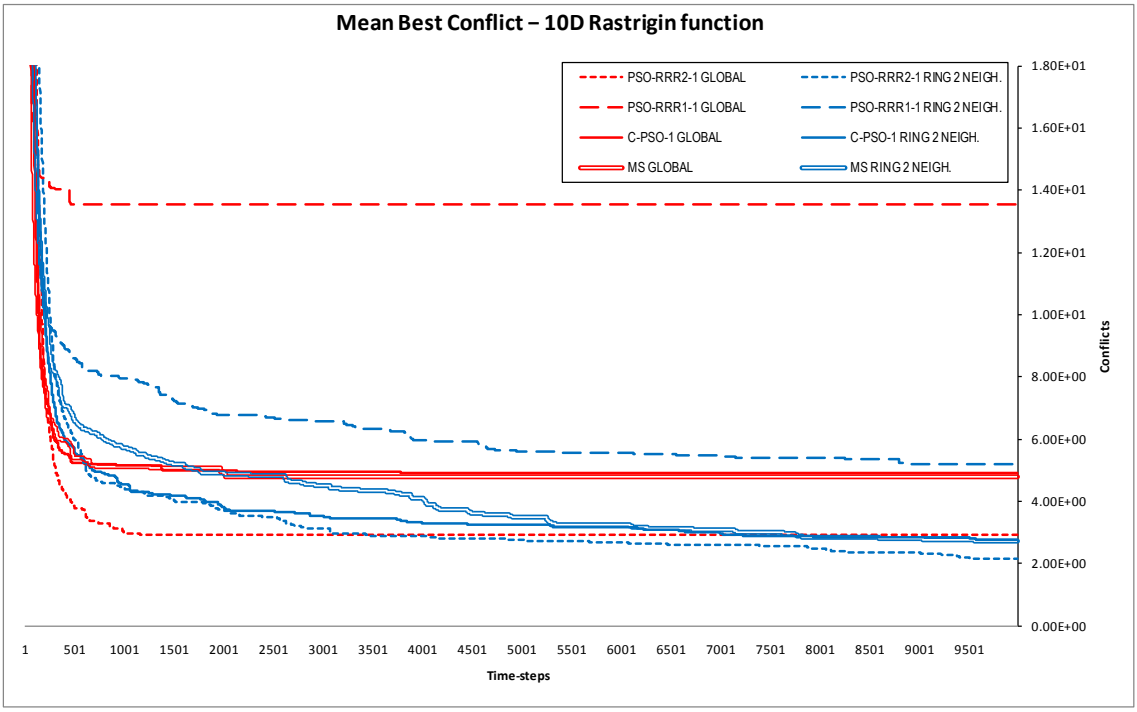


Fig. 6.81. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 6.10. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.11. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	2.69E+01	4.28E+01	4.13E+01	5.57E+01	2.64E-11	0
			1000	2.69E+01	4.28E+01	4.14E+01	5.57E+01	3.99E-05	-
	RING	nn = 2	10000	2.98E+01	4.40E+01	4.29E+01	5.29E+01	2.61E-02	0
			1000	3.46E+01	5.32E+01	5.24E+01	7.23E+01	3.00E-02	-
PSO-RRR1-1	GLOBAL		10000	2.49E+01	7.16E+01	7.41E+01	1.28E+02	6.68E-16	0
			1000	2.49E+01	7.16E+01	7.41E+01	1.28E+02	1.47E-15	-
	RING	nn = 2	10000	2.19E+01	4.68E+01	4.65E+01	6.17E+01	3.00E-02	0
			1000	2.20E+01	5.01E+01	5.03E+01	6.71E+01	3.12E-02	-
C-PSO-1	GLOBAL		10000	2.69E+01	4.88E+01	5.37E+01	9.65E+01	1.93E-11	0
			1000	2.69E+01	4.88E+01	5.37E+01	9.65E+01	1.09E-10	-
	RING	nn = 2	10000	2.89E+01	5.37E+01	5.05E+01	6.87E+01	3.31E-02	0
			1000	2.89E+01	5.88E+01	5.59E+01	7.79E+01	3.43E-02	-
Multi-Swarm	GLOBAL		10000	2.59E+01	5.27E+01	5.33E+01	8.16E+01	1.89E-11	0
			1000	2.59E+01	5.27E+01	5.33E+01	8.16E+01	5.36E-08	-
	RING	nn = 2	10000	3.28E+01	4.48E+01	4.56E+01	6.37E+01	3.03E-02	0
			1000	3.32E+01	5.21E+01	4.97E+01	6.57E+01	3.09E-02	-

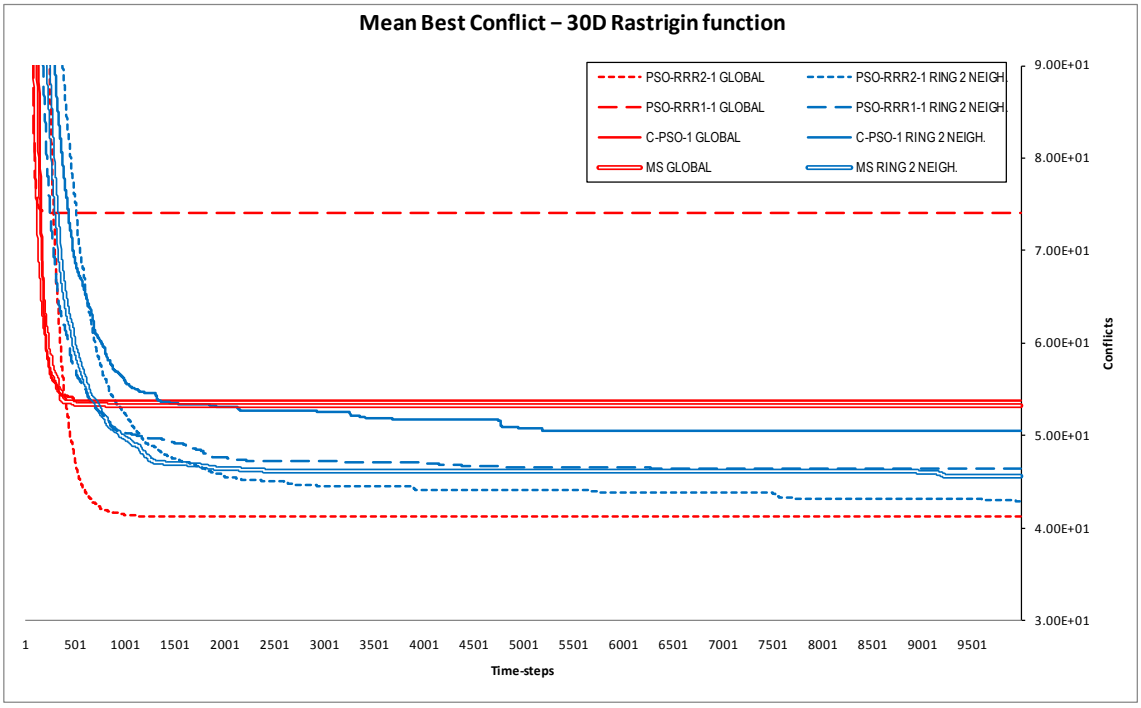


Fig. 6.82. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 6.11. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.12. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Griewank function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**2.96E-04**	**7.40E-03**	**7.41E-12**	**96**
			1000	0.00E+00	0.00E+00	5.92E-04	7.40E-03	1.10E-03	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**3.65E-04**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.10E-03	-
PSO-RRR1-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**5.92E-04**	**7.40E-03**	**5.76E-12**	**92**
			1000	0.00E+00	0.00E+00	1.18E-03	9.08E-04	9.08E-04	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**4.15E-04**	**100**
			1000	0.00E+00	0.00E+00	3.02E-13	7.54E-12	2.05E-03	-
C-PSO-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**6.40E-12**	**100**
			1000	0.00E+00	0.00E+00	1.18E-03	7.40E-03	9.46E-04	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.57E-04**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.96E-03	-
Multi-Swarm	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**6.56E-12**	**100**
			1000	0.00E+00	0.00E+00	5.95E-16	1.49E-14	8.92E-04	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**2.51E-04**	**100**
			1000	0.00E+00	0.00E+00	2.39E-05	5.88E-04	2.05E-03	-

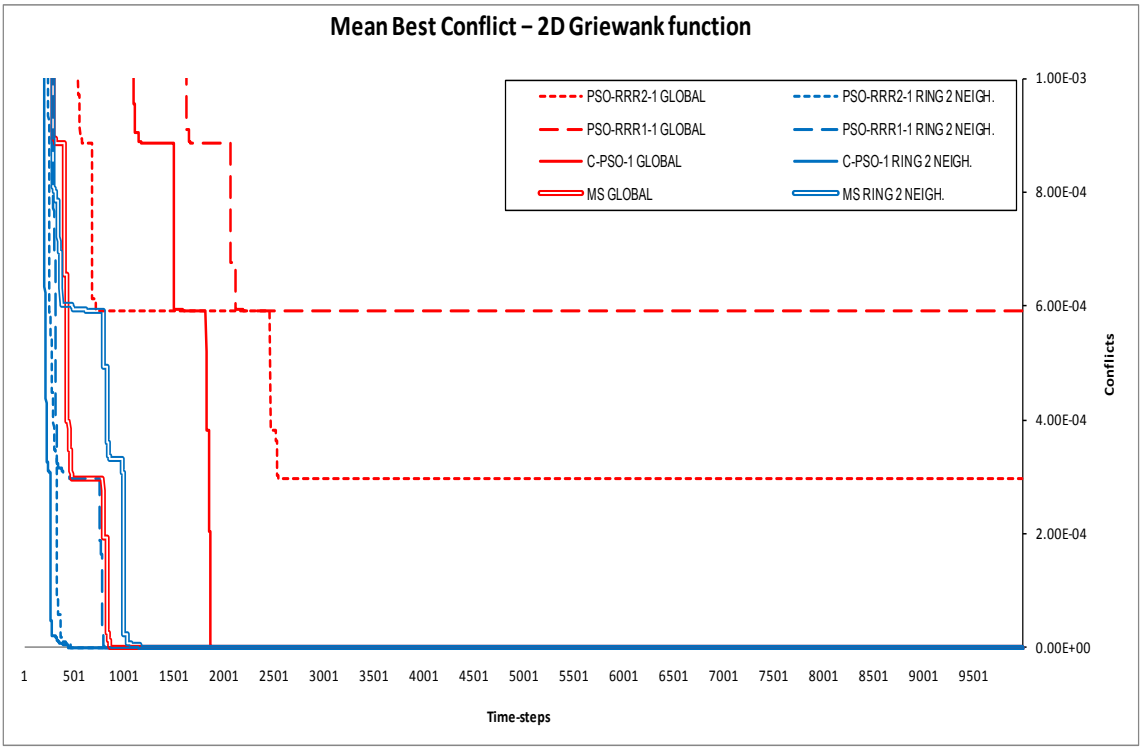


Fig. 6.83. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 6.12. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.13. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Griewank function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.97E-02	5.66E-02	6.81E-02	1.43E-01	4.94E-07	0
			1000	1.97E-02	6.16E-02	7.14E-02	1.43E-01	1.18E-04	-
	RING	nn = 2	10000	0.00E+00	2.46E-02	2.66E-02	6.15E-02	1.81E-03	4
			1000	2.96E-07	2.95E-02	3.41E-02	6.88E-02	2.01E-03	-
PSO-RRR1-1	GLOBAL		10000	2.96E-02	9.11E-02	9.27E-02	1.82E-01	1.81E-12	0
			1000	2.96E-02	9.11E-02	9.27E-02	1.82E-01	1.48E-05	-
	RING	nn = 2	10000	0.00E+00	3.19E-02	3.05E-02	7.38E-02	1.58E-03	8
			1000	0.00E+00	3.94E-02	3.64E-02	7.62E-02	1.66E-03	-
C-PSO-1	GLOBAL		10000	1.97E-02	6.64E-02	6.68E-02	1.38E-01	1.65E-06	0
			1000	2.71E-02	6.89E-02	7.21E-02	1.38E-01	1.22E-04	-
	RING	nn = 2	10000	0.00E+00	2.46E-02	2.36E-02	4.68E-02	1.55E-03	4
			1000	0.00E+00	2.71E-02	2.85E-02	8.87E-02	1.66E-03	-
Multi-Swarm	GLOBAL		10000	1.48E-02	6.64E-02	6.64E-02	1.38E-01	1.48E-05	0
			1000	2.95E-02	7.13E-02	7.85E-02	1.85E-01	1.40E-04	-
	RING	nn = 2	10000	0.00E+00	1.97E-02	2.15E-02	5.90E-02	1.60E-03	8
			1000	7.40E-03	2.22E-02	2.71E-02	5.90E-02	1.75E-03	-

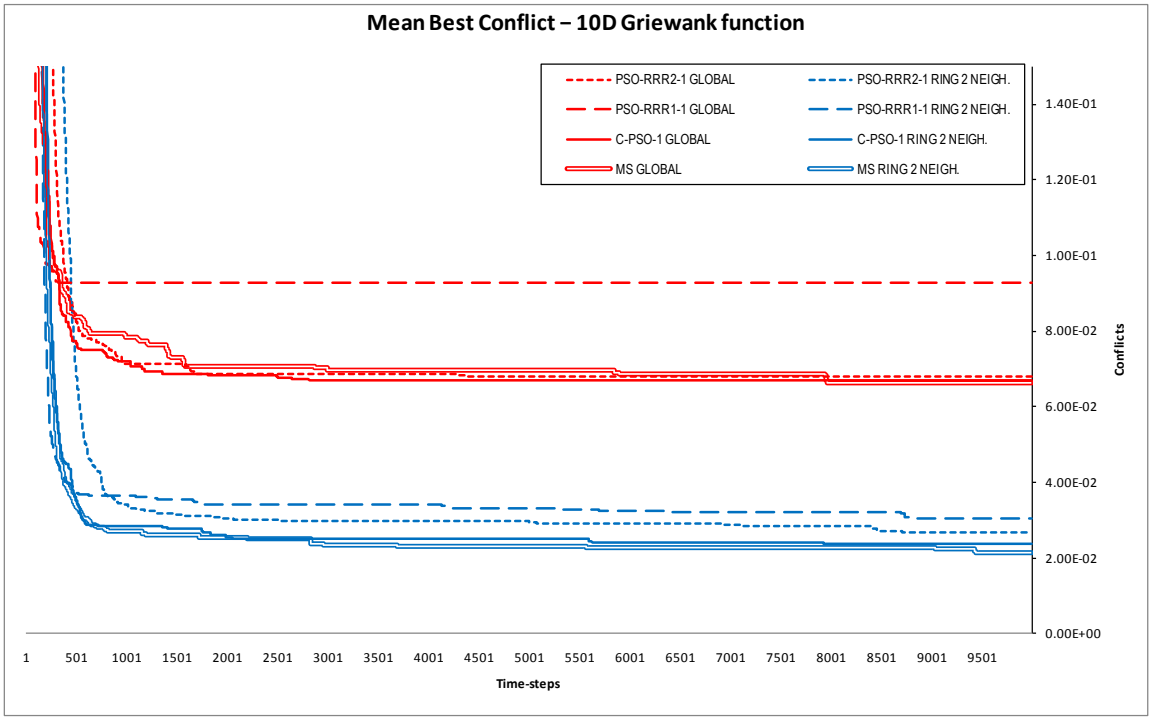


Fig. 6.84. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 6.13. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.14. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Griewank function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	7.40E-03	9.35E-03	2.96E-02	4.23E-12	44
			1000	7.95E-06	7.44E-03	9.40E-03	2.96E-02	2.93E-06	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	2.96E-04	7.40E-03	1.22E-06	96
			1000	2.54E-01	4.15E-01	4.23E-01	6.79E-01	2.52E-04	-
PSO-RRR1-1	GLOBAL		10000	3.29E-08	6.46E-02	1.02E-01	7.40E-01	3.91E-13	4
			1000	3.29E-08	7.11E-02	1.12E-01	7.40E-01	4.04E-14	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	6.90E-04	9.86E-03	6.40E-08	92
			1000	4.02E-11	7.73E-10	6.91E-04	9.86E-03	1.38E-06	-
C-PSO-1	GLOBAL		10000	0.00E+00	1.23E-02	1.79E-02	7.09E-02	2.56E-12	36
			1000	0.00E+00	1.23E-02	1.79E-02	7.09E-02	2.93E-12	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.14E-07	100
			1000	4.55E-06	3.98E-05	2.60E-03	2.22E-02	1.52E-05	-
Multi-Swarm	GLOBAL		10000	0.00E+00	4.67E-02	5.18E-02	1.41E-01	2.42E-12	4
			1000	6.66E-16	4.67E-02	5.18E-02	1.41E-01	4.06E-08	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	2.17E-03	1.23E-02	1.10E-06	76
			1000	6.02E-08	7.40E-03	6.97E-03	3.92E-02	3.08E-05	-

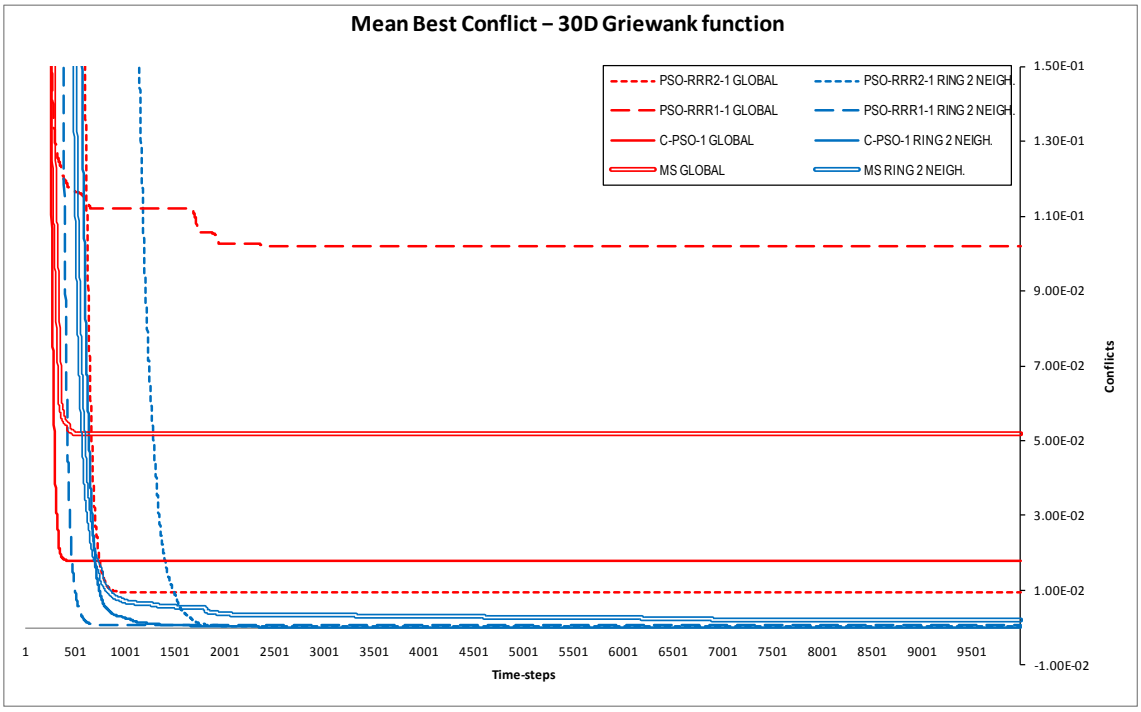


Fig. 6.85. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 6.14. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.15. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**3.89E-04**	**9.72E-03**	**2.58E-05**	**96**
			1000	0.00E+00	0.00E+00	7.77E-04	9.72E-03	2.79E-03	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**3.60E-04**	**100**
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	6.82E-03	-
PSO-RRR1-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**1.17E-03**	**9.72E-03**	**1.13E-04**	**88**
			1000	0.00E+00	0.00E+00	1.17E-03	9.72E-03	1.61E-03	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.29E-03**	**100**
			1000	0.00E+00	0.00E+00	1.96E-03	9.72E-03	7.78E-03	-
C-PSO-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**1.17E-03**	**9.72E-03**	**2.01E-04**	**88**
			1000	0.00E+00	0.00E+00	1.95E-03	9.72E-03	2.42E-03	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**3.89E-04**	**9.72E-03**	**1.38E-03**	**96**
			1000	0.00E+00	0.00E+00	2.07E-03	9.72E-03	7.76E-03	-
Multi-Swarm	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**2.33E-03**	**9.72E-03**	**1.05E-04**	**76**
			1000	0.00E+00	0.00E+00	3.11E-03	9.72E-03	2.58E-03	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**7.13E-04**	**100**
			1000	0.00E+00	0.00E+00	7.84E-04	9.72E-03	7.34E-03	-

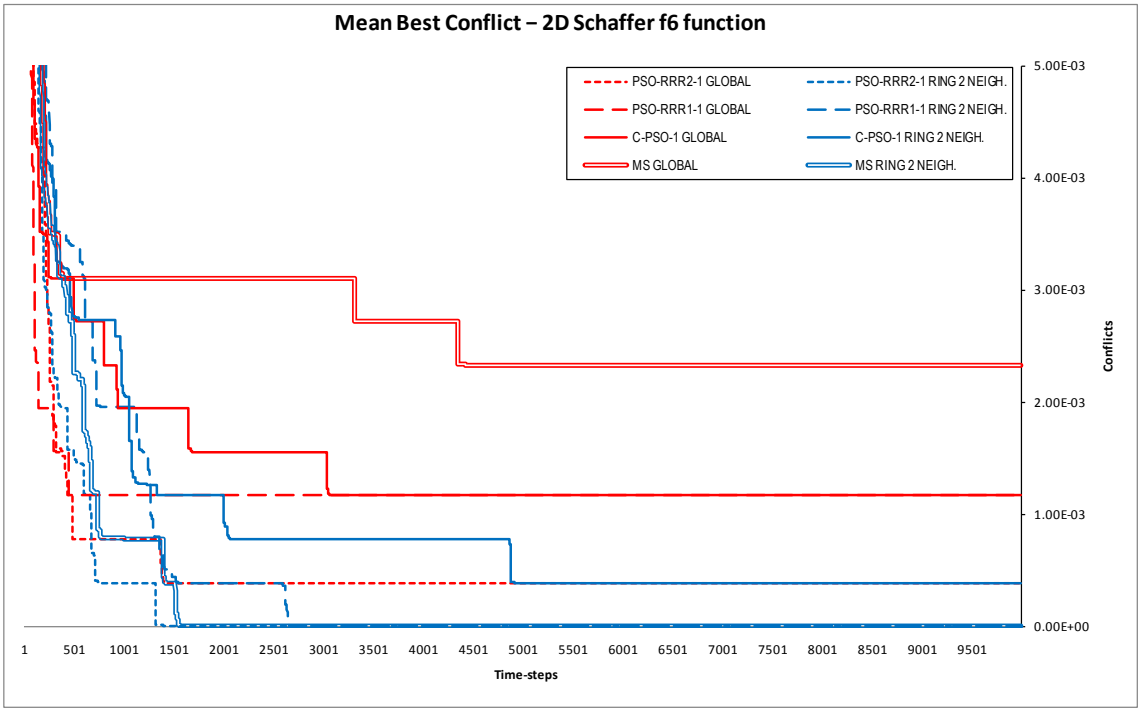


Fig. 6.86. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 6.15. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.16. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	9.72E-03	9.72E-03	1.85E-02	3.72E-02	5.10E-04	0
			1000	9.72E-03	9.72E-03	2.18E-02	3.72E-02	1.21E-03	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.94E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	3.24E-03	-
PSO-RRR1-1	GLOBAL		10000	9.72E-03	3.72E-02	3.45E-02	7.82E-02	1.29E-04	0
			1000	9.72E-03	3.72E-02	3.45E-02	7.82E-02	1.29E-04	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	2.18E-02	3.72E-02	2.43E-03	0
			1000	9.72E-03	3.72E-02	3.17E-02	3.72E-02	3.38E-03	-
C-PSO-1	GLOBAL		10000	9.72E-03	9.72E-03	1.96E-02	3.72E-02	3.01E-04	0
			1000	9.72E-03	9.72E-03	2.18E-02	3.72E-02	7.06E-04	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	2.21E-03	0
			1000	9.72E-03	9.72E-03	1.85E-02	3.72E-02	3.19E-03	-
Multi-Swarm	GLOBAL		10000	9.72E-03	3.72E-02	2.95E-02	3.72E-02	3.18E-04	0
			1000	9.72E-03	3.72E-02	3.06E-02	3.72E-02	6.67E-04	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	2.03E-03	0
			1000	9.72E-03	9.72E-03	1.74E-02	3.72E-02	3.03E-03	-

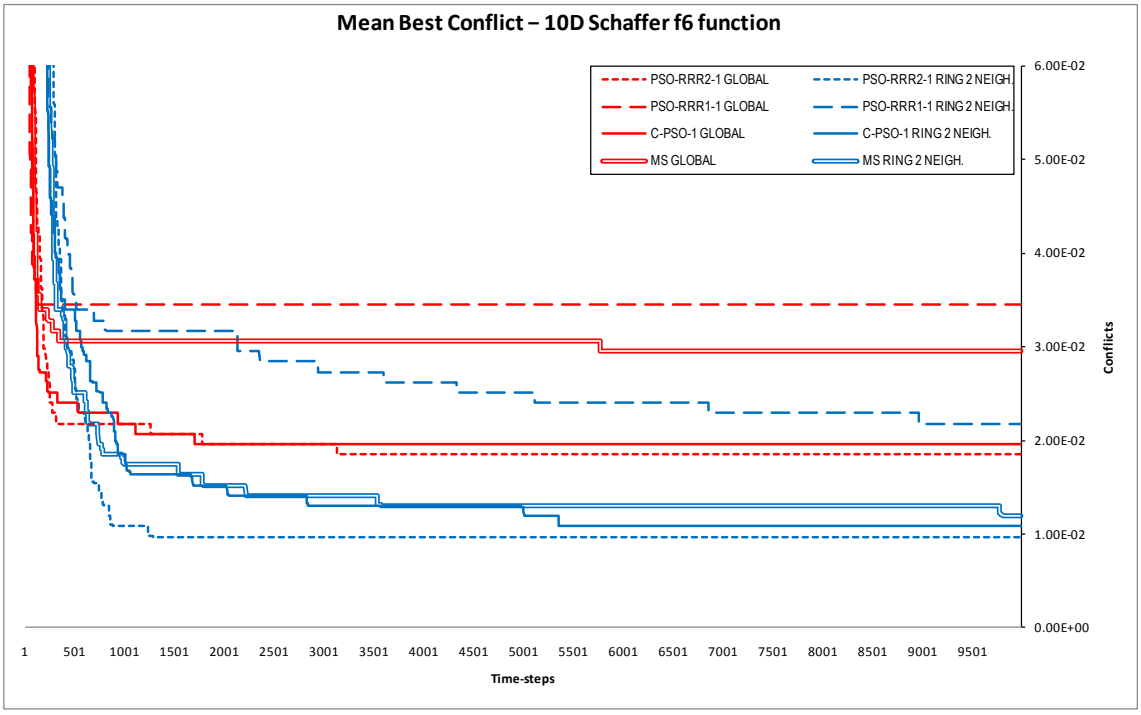


Fig. 6.87. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 6.16. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 6.17. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL and the RING with 2 neighbours. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	3.72E-02	7.82E-02	9.22E-02	1.27E-01	3.12E-04	0
			1000	7.82E-02	1.27E-01	1.08E-01	1.78E-01	8.50E-04	-
	RING	nn = 2	10000	3.72E-02	7.82E-02	6.18E-02	7.82E-02	1.77E-03	0
			1000	1.27E-01	1.96E-01	2.01E-01	2.29E-01	4.40E-03	-
PSO-RRR1-1	GLOBAL		10000	3.12E-01	4.30E-01	4.25E-01	4.85E-01	4.76E-05	0
			1000	3.12E-01	4.30E-01	4.26E-01	4.87E-01	2.01E-04	-
	RING	nn = 2	10000	7.82E-02	1.78E-01	1.67E-01	2.73E-01	1.72E-03	0
			1000	1.27E-01	2.28E-01	2.14E-01	3.12E-01	2.38E-03	-
C-PSO-1	GLOBAL		10000	7.82E-02	1.27E-01	1.31E-01	2.73E-01	1.67E-04	0
			1000	7.82E-02	1.27E-01	1.40E-01	2.73E-01	4.94E-04	-
	RING	nn = 2	10000	3.72E-02	3.72E-02	5.52E-02	7.82E-02	1.59E-03	0
			1000	1.27E-01	1.27E-01	1.52E-01	1.78E-01	3.23E-03	-
Multi-Swarm	GLOBAL		10000	7.82E-02	1.78E-01	1.86E-01	2.73E-01	2.30E-04	0
			1000	1.27E-01	1.78E-01	1.93E-01	2.73E-01	5.26E-04	-
	RING	nn = 2	10000	3.72E-02	7.82E-02	7.45E-02	1.27E-01	1.58E-03	0
			1000	1.27E-01	1.78E-01	1.68E-01	2.28E-01	3.42E-03	-

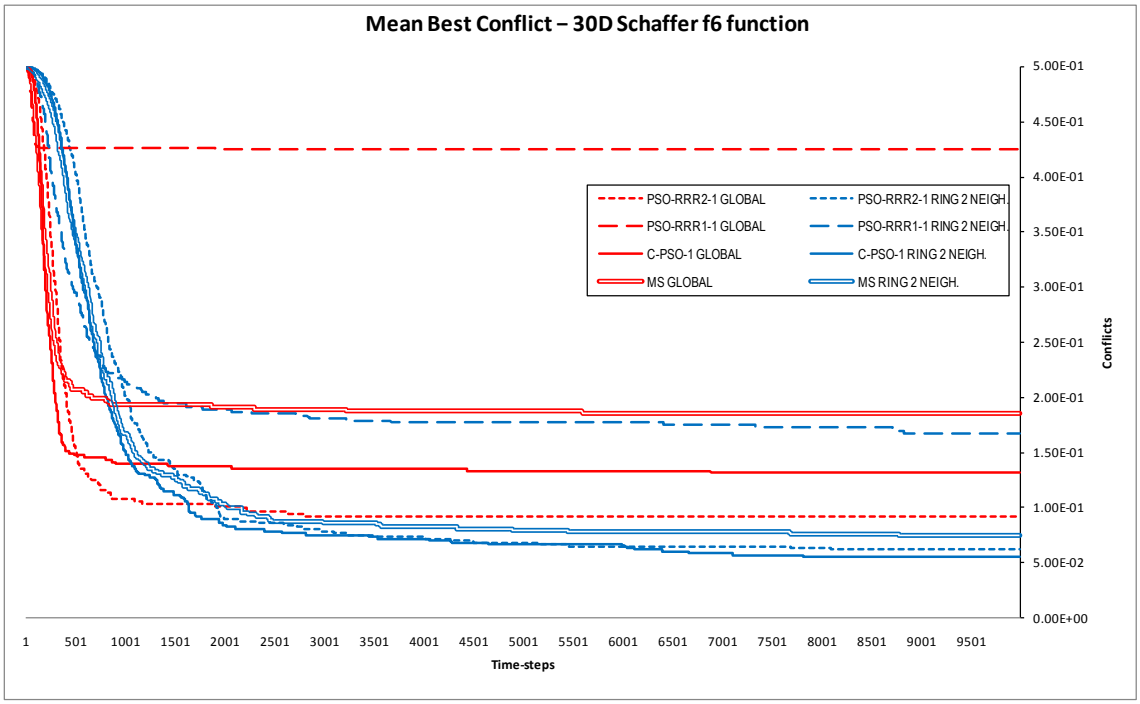


Fig. 6.88. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 6.17. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

6.3.2.3. Discussion

Sphere

The results are offered in Table 6.3 to Table 6.5, and in Fig. 6.74 to Fig. 6.76. All algorithms with both neighbourhoods perform well for all dimensions, although the accuracy of the solution decreases with the dimensionality. The only exception is the 'PSO-RRR1-1 Global', which displays premature convergence in 30D space. For all the other cases, the global versions found better results than their local counterparts both by the 10000th and by the 1000th time-steps. Except for the 2D case, they also reached a higher degree of clustering. In the 2D problem, the local versions reach a marginally higher clustering than their global counterparts because the particles move in smaller steps.

Comparing the different coefficients' settings, it appears that the PSO-RRR1-1 exhibits faster clustering than expected. Thus, the PSO-RRR1-1 is the fastest, while the PSO-RRR2-1 is the most robust. Considering the neighbourhoods as well, the 'PSO-RRR1-1 Global' is the fastest, whereas the 'PSO-RRR2-1 Ring' is the slowest. In fact, it is interesting to observe that the mean conflicts of the latter two are the only ones which do not meet the success criterion by the 1000th time-step for the 30D problem (see Table 6.5) but for two different reasons: the 'PSO-RRR1-1 Global' because of premature convergence, and the 'PSO-RRR2-1 Ring' because of slow convergence (see Fig. 6.76).

As to the premature convergence of the 'PSO-RRR1-1 Global' in 30D space, it seems that the particles practically imploded so quickly that there is premature convergence. Improvement is still possible because of the simplicity of the problem, but the swarm of particles move 'as one', resembling a hill-descending algorithm. Given that the global best keeps moving, the degree of clustering[4] reached by the 10000th time-step is the lowest despite being the algorithm with fastest clustering ability. This can be observed by comparing Fig. 6.76 and Fig. 6.73. The swarm practically imploded in less than 500 time-steps, while improvement of the mean best conflict can still be observed up until around 3500 time-steps. Also notice that the time-steps at which the mean **gbest** is updated in Fig. 6.76 coincide with the sudden (and tiny) increase of diversity in Fig. 6.73. This makes sense, as new momentum is introduced into the system.

[4] Note that a high degree of clustering implies a low pb_me, and vice versa.

Rosenbrock

The 2D Rosenbrock function is unimodal, and its global minimum is within a long, narrow, flat valley. Although it is common in the literature to simply assume that it remains unimodal when generalized for any number of dimensions, Shang and Qiu (2006) showed that it becomes multimodal for more than 3 dimensions. The results are offered in Table 6.6 to Table 6.8, and in Fig. 6.77 to Fig. 6.79.

For the 2D problem, all combinations of coefficients' settings and neighbourhood topologies tested achieve 100% success. Fig. 6.77 shows that the local topologies effectively converge more slowly than their global counterparts.

For the 10D problem, the local versions outperform their global counterparts in terms of the mean best solution. However, the 'PSO-RRR2-1 Global' and the 'C-PSO-1 Global' outperform their local counterparts in terms of the percentage of success. Note that the solution '3.99' in Table 6.7 is a local minimum of the 10D problem (refer to (Shang & Qiu, 2006)). The 'PSO-RRR1-1 Global' displays again premature convergence, while the 'PSO-RRR1-1 Ring' obtained the best results in terms of the mean and median solutions, and the percentage of success. The 'Multi-Swarm Ring' also obtained very good results in terms of the median solution and the percentage of success, although it gets stuck in a local optimum for some runs. Despite the low percentage of success, the results obtained by the 'PSO-RRR2-1 Ring' are also good. It is fair to remark that in the latter case, convergence has not been achieved by the end of the search (see Fig. 6.78).

For the 30D problem, the 'PSO-RRR1-1 Global' exhibits premature convergence once again, while the best performance is displayed by the 'C-PSO-1 Global' by a considerable margin, followed by the 'C-PSO-1 Ring', the 'Multi-Swarm Ring', and the 'PSO-RRR1-1 Ring'. It must be noted, however, that convergence has not been reached (refer to Fig. 6.79). Also notice that a high degree of clustering is achieved only by the 'PSO-RRR1-1 Global', which suffered from premature convergence (see pb_me in Table 6.8).

The percentage of success decreases considerable with the dimensionality of the problem. Some strange degrees of clustering might be found for the Rosenbrock function, as diversity typically decreases consistently until the particles reach the flat region. At that moment, they tend to scatter again as the global best changes location constantly.

Rastrigin

The results are offered in Table 6.9 to Table 6.11, and in Fig. 6.80 to Fig. 6.82.

Every algorithm finds the solution for the 2D problem by the 10000th time-step. In fact, they already do by the 1000th time-step, except for the 'Multi-Swarm Ring'.

For the 10D problem, the local versions outperform their global counterparts, as it would be expected when dealing with highly multimodal problems. The 'PSO-RRR1-1 Global' exhibits again premature convergence. The best performance in terms of the best, median and mean solutions, and percentage of success, is displayed by the 'PSO-RRR2-1 Ring' (coefficients' settings favouring exploration, and neighbourhood structure leading to slow spread of information). The 'Multi-Swarm Ring' and the 'C-PSO-1 Ring' also find good results. Note that the degree of clustering of the local neighbourhoods is notably lower than that of their global counterparts (i.e. higher pb_me in Table 6.10). Also notice that, while the 'PSO-RRR2-1 Global' exhibits the best performance *among the global neighbourhoods* –as expected–, it is also the one with the lowest degree of clustering achieved *among them* (greatest pb_me).

For the 30D problem, the 'PSO-RRR1-1 Global' also converges too quickly, while the best mean solution is found by the 'PSO-RRR2-1 Global'. The second best mean solution is found by the 'PSO-RRR2-1 Ring', and the third by the 'Multi-Swarm Ring'. However, while the global neighbourhoods completely stagnate (pb_me in the range of 10^{-11}), the local neighbourhoods do not fully converge by the time the search is terminated. Notice that their degree of clustering is much lower (pb_me in the range of 10^{-2}). That is to say that, for an extended search, further improvement would be expected. With regards to the coefficients' settings only, it is evident that the more robust setting of the PSO-RRR2-1 is more convenient for highly multimodal problems like this.

Griewank

The results are presented in Table 6.12 to Table 6.14, and in Fig. 6.83 to Fig. 6.85.

The 2D problem is highly multimodal, and therefore it is not surprising that the local neighbourhoods outperform their global counterparts, or that the 'PSO-RRR1-1 Global' finds the worst mean solution and lowest success rate, also exhibiting premature convergence. The 'PSO-RRR2-1 Global' fails to find the solution in 1 of the 25 runs.

For the 10D problem, the success rates decrease dramatically. The 'PSO-RRR1-1 Global' converges prematurely once again, while the best mean solution is found by the 'Multi-Swarm Ring'. As it is to be expected in multimodal problems (unless the search is not long enough), the local neighbourhoods clearly outperform their local counterparts. Note that full convergence has not been achieved in 10000 time-steps.

For the 30D problem, the 'PSO-RRR1-1 Global' converges early finding the worst results, while the best mean solution is found by the 'C-PSO-1 Ring' and the 'PSO-RRR2-1 ring'. In general, the local neighbourhoods outperform their local counterparts, as it is to be expected for a long enough search and a multimodal problem. Note, however, that this function does not necessarily increases its difficulty with dimensionality, as increasing dimensions decrease its multimodality. In fact, the success rates are notably higher for the 30D problem than for the 10D problem.

Schaffer f6

The results are presented in Table 6.12 to Table 6.14, and in Fig. 6.83 to Fig. 6.85.

For the 2D problem, the local neighbourhoods outperform their global counterparts. The best performances are those of the 'PSO-RRR2-1 Ring', the 'PSO-RRR1-1 Ring', and the 'Multi-Swarm Ring', which achieved a 100% success rate.

For the 10D problem, all success rates decrease to zero. The 'PSO-RRR1-1 Global' suffers from premature convergence, while the 'PSO-RRR2-1 Ring' shows the best performance. Local neighbourhoods outperform their global counterpart, as expected.

For the 30D problem, the same trends observed for the 10D problem apply, although the best performance here is displayed by the 'C-PSO-1 Ring'. Full convergence does not seem to take place either for the 10D or the 30D problems.

Overall analysis

The combined effect of a global neighbourhood with coefficients that favour fast convergence results in extremely fast convergence of the algorithm as a whole. This is very bad to cope with multimodal problems, and even with high-dimensional unimodal problems, due to the tendency of the algorithm to premature convergence. However, it might be useful for problems that require convergence in just a few time-steps, say 50 to 500.

One interesting observation from these experiments, however, is that the PSO-RRR1-1 results in fast convergence, even faster than the C-PSO-1, despite having an inertia weight as high as 0.80. Therefore, the speed of convergence may be increased simply by reducing the range of randomness (i.e. the interval of ϕ), maintaining the same average behaviour. Notice that remarkably faster convergence is to be expected for lower inertia weights in the PSO-RRR1 formulation.

Local neighbourhoods balance fast-convergent coefficients, whereas more robust coefficients balance global neighbourhoods. Thus, premature convergence may be dealt with by both means. In general, it would seem that using local neighbourhoods with fast-convergent coefficients (e.g. 'PSO-RRR1-1 Ring') works better for unimodal and moderately multimodal problems (e.g. Sphere, Rosenbrock, Griewank) whereas using global neighbourhoods with robust coefficients (e.g. 'PSO-RRR2-1 Global') works better for highly multimodal problems (e.g. Rastrigin, Schaffer f6). Nonetheless, the actual global topology is not desirable except for specific cases. Local neighbourhoods are preferable, as long as the search-length allows for convergence. For highly multimodal problems, the combination of robust coefficients and local neighbourhoods is desirable. The problem is that the speed of convergence remarkably decreases. Results might not be satisfactory if the search is not long enough.

6.4. Coefficients' settings guidelines

It is advisable that the settings lead to convergence without external mechanisms enforcing it. Thus, at least 'ϕ_{mean}–w' should be within the convergence region.

General PSO formulation (proposed in this thesis)

$$
\begin{aligned}
&v_{ij}^{(t)} = w \cdot v_{ij}^{(t-1)} + \phi_i \cdot \left(pbest_{ij}^{(t-1)} - x_{ij}^{(t-1)}\right) + \phi_s \cdot \left(lbest_{ij}^{(t-1)} - x_{ij}^{(t-1)}\right) \\
&\phi_i = ip \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min}\right) \cdot U_{(0,1)}\right] \\
&\phi_s = sp \cdot \left[\phi_{\min} + \left(\phi_{\max} - \phi_{\min}\right) \cdot U_{(0,1)}\right] \\
&ip \in [0,1) \quad ; \quad sp = 1 - ip \\
&x_{ij}^{(t)} = x_{ij}^{(t-1)} + v_{ij}^{(t)}
\end{aligned}
\tag{6.24}
$$

Choose $0.30 \le w \le 0.90$. Preferably,

$$\boxed{0.50 \le w \le 0.90} \tag{6.25}$$

Higher values increase the ability to avoid premature convergence whilst lower values speed up convergence and improve the fine-grain search (see Fig. 6.89).

Choose $\phi_{\min} \ge 0$ and $2.00 \le \phi_{\max} \le 4.00$. Advice:

$$\boxed{\begin{aligned} 0.00 &\le \phi_{\min} \le 1.00 \\ 2.00 &\le \phi_{\max} \le 2 \cdot (w+1) \end{aligned}} \tag{6.26}$$

If $\phi_{\min} \to 0$, the stochastic acceleration coefficient (ϕ) may approach zero. Hence a high inertia weight (w) with $\phi_{\min} \to 0$ may lead to greater local explosions for a sequence of low values of ϕ generated. If $\phi_{\min} \not\to 0$, the local explosions are more controlled.

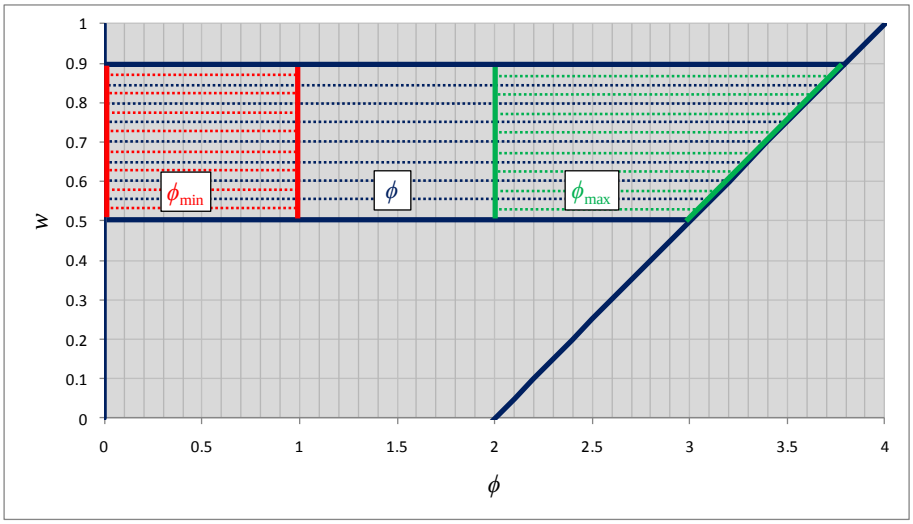


Fig. 6.89. Suggested region in the 'ϕ–w' plane from where ϕ is to be randomly sampled (blue dotted lines). The regions of suggested upper ($\phi_{\max}$) and lower ($\phi_{\min}$) limits of ϕ are shown in green and red dotted lines, respectively.

Note that $\phi_{\min}$ and $\phi_{\max}$ define the average behaviour (ϕ_{mean}) as well as the strength awarded to randomness. For the same average behaviour, a greater interval of ϕ results in higher exploration, more erratic behaviour, and slower convergence. Advice:

$$1.00 < \phi_{\text{mean}} = 0.50 \cdot (\phi_{\max} + \phi_{\min}) < 2.00 \tag{6.27}$$

Note that lower accelerations lead to higher amplitudes and lower frequencies of the oscillatory trajectories around the attractors. Higher amplitudes widen the exploration region while higher frequencies result in the particles overflying their attractors a higher number of times, and approaching them from both sides in each dimension (advisable).

Choose *ip*, *sp*, and $\mathbf{v_{max}}$. Advice:

$$\boxed{ip = sp = 0.50} \tag{6.28}$$

$$\boxed{v_{\max j} = 0.50 \cdot \left(x_{j\max} - x_{j\min}\right)} \tag{6.29}$$

Guidelines for the settings corresponding to five particular instances of this general PSO formulation are offered hereafter.

1. *Classical PSO formulation*

To translate the proposed formulation into the classical one, replace $\phi_{\min}$ in Eq. (6.26) by Eq. (6.30). Other relations between the two formulations are offered in Eq. (6.31).

$$\boxed{\phi_{\min} = 0} \tag{6.30}$$

$$\left.\begin{aligned} iw &= ip \cdot \phi_{\max} \\ sw &= sp \cdot \phi_{\max} \end{aligned}\right\} \quad \Rightarrow \quad \begin{cases} \phi_i = iw \cdot U_{(0,1)} = U_{(0,iw)} \\ \phi_s = sw \cdot U_{(0,1)} = U_{(0,sw)} \end{cases} \tag{6.31}$$

Given Eq. (6.30), higher values of $\phi_{\max}$ also have the indirect effect of increasing the effect of randomness (widen the range of ϕ). That is, the lower the $\phi_{\max}$ the more similar the actual behaviour is to the average behaviour. And therefore, higher values of $\phi_{\max}$ indirectly decrease the speed of convergence and result in more erratic behaviour.

2. *Constricted PSO formulation*

Choose *aw* and $0 < \kappa < 1$. Advice:

$$\boxed{\begin{aligned} &aw > 4\,(\text{slightly}) \\ &\kappa \to 1 \end{aligned}} \tag{6.32}$$

Replace Eqs. (6.25) and (6.26) by Eq. (6.33):

$$\begin{cases} w = \begin{cases} \dfrac{2\cdot\kappa}{aw-2+\sqrt{aw^2-4\cdot aw}} & \text{if } aw \geq 4 \\ \kappa & \text{otherwise} \end{cases} \\ \phi_{\min} = 0 \\ \phi_{\max} = w\cdot aw \end{cases} \tag{6.33}$$

3. *PSO-RRM formulation*

Choose $1.30 \leq \phi_{\text{mean}} \leq 1.90$. Advice:

$$1.50 \leq \phi_{\text{mean}} \leq 1.90 \tag{6.34}$$

Higher values favour exploration whilst lower values favour fast convergence. This average behaviour exhibits relatively low frequencies in the oscillations (see Fig. 6.90).

Replace Eqs. (6.25) and (6.26) by Eq. (6.35):

$$\begin{cases} w = \phi_{\text{mean}} - 1 \\ \phi_{\min} = 0 \\ \phi_{\max} = 2\cdot(w+1) \end{cases} \tag{6.35}$$

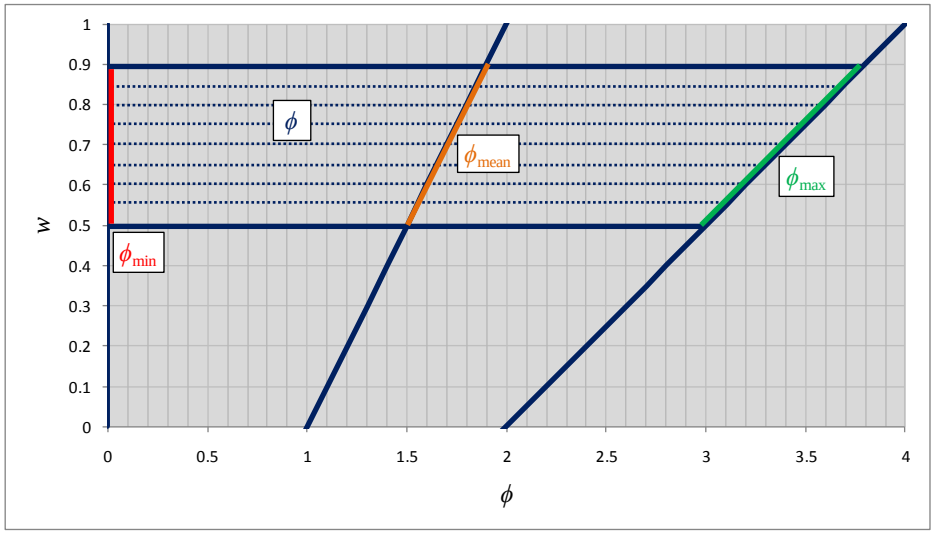


Fig. 6.90. Suggested region in the 'ϕ–w' plane from where ϕ is to be randomly sampled in the PSO-RRM. Note that $\phi_{\min}$ and $\phi_{\max}$ are no longer areas but lines. The user only chooses ϕ_{mean}.

Note that w, $\phi_{\min}$, ϕ_{mean}, and $\phi_{\max}$ comply with Eqs. (6.25) to (6.27) and with Fig. 6.89.

4. PSO-RRR1

Choose $1.30 \le \phi_{mean} \le 1.90$. Advice:

$$1.50 \le \phi_{mean} \le 1.90 \tag{6.36}$$

Higher values favour exploration whilst lower values favour fast convergence. This average behaviour exhibits relatively low frequencies in the oscillations (see Fig. 6.91).

Replace Eqs. (6.25) and (6.26) by Eq. (6.37):

$$\begin{aligned} w &= \phi_{mean} - 1 \\ \phi_{min} &= \frac{1}{2} \cdot (w+1) \\ \phi_{max} &= \frac{3}{2} \cdot (w+1) \end{aligned} \tag{6.37}$$

Because of the fact that ϕ is always within convergence region, $\phi_{min} > 0$, and the range of randomness is reduced, this approach results in fewer and smaller local explosions, and in the actual behaviour similar to the average behaviour. Therefore convergence is fast even for high values of the inertia weight (extremely fast for low values of w).

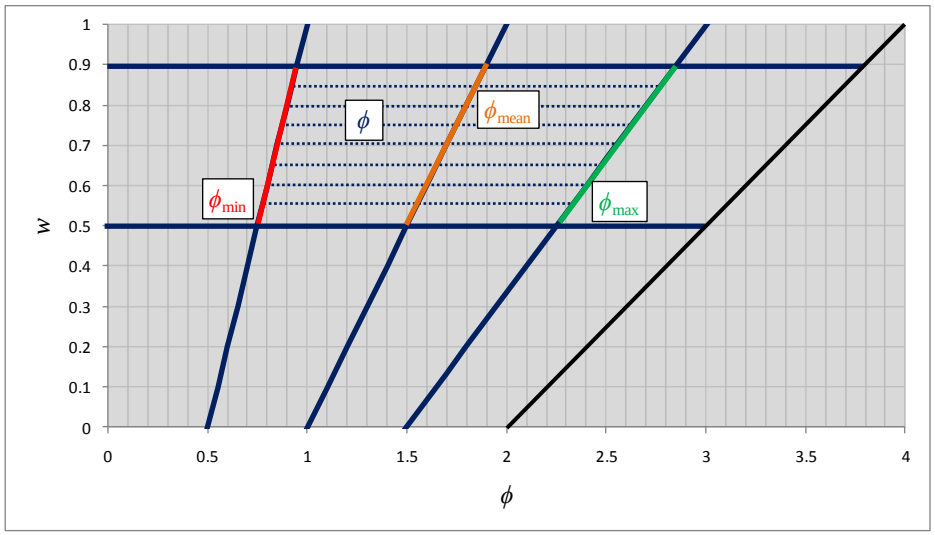


Fig. 6.91. Suggested region in the 'ϕ–w' plane from where ϕ is to be randomly sampled in the PSO-RRR1. The user only chooses ϕ_{mean}.

Note that w, ϕ_{min}, ϕ_{mean}, and ϕ_{max} comply with Eqs. (6.25) to (6.27) and with Fig. 6.89.

The influence of randomness can be increased, if desired, by enlarging the ϕ interval.

5. PSO-RRR2

Choose $1.70 \le \phi_{mean} \le 2.50$. Advice:

$$\boxed{2.00 \le \phi_{mean} \le 2.50} \tag{6.38}$$

Higher values favour exploration whilst lower values favour fast convergence. This average behaviour presents high frequencies in the oscillations (see Fig. 6.92).

Replace Eqs. (6.25) and (6.26) by Eq. (6.39):

$$\boxed{\begin{aligned} w &= \frac{1}{\phi_{mean}} - 2 + \phi_{mean} \\ \phi_{max} &= 2 \cdot (w+1) \\ \phi_{min} &= 2 \cdot \phi_{mean} - \phi_{max} \end{aligned}} \tag{6.39}$$

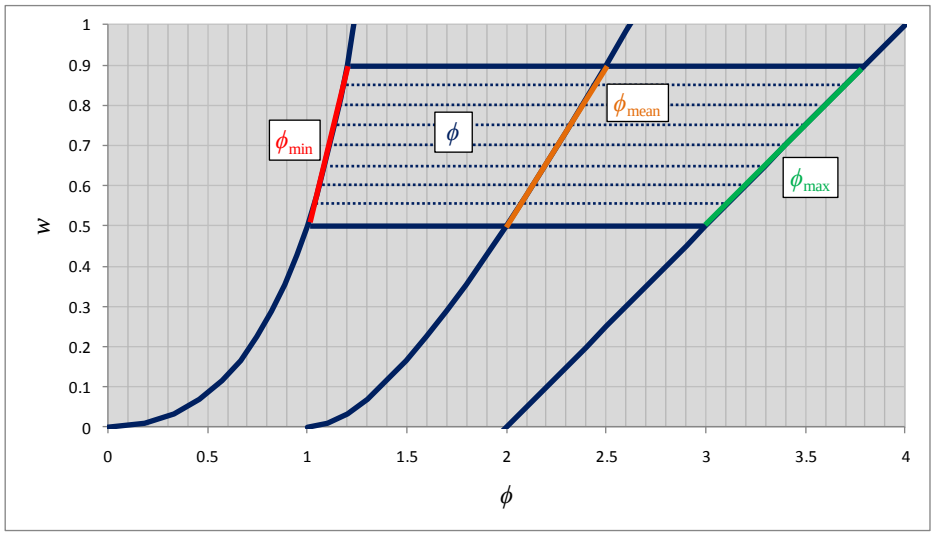


Fig. 6.92. Suggested region in the 'ϕ–w' plane from where ϕ is to be randomly sampled in the PSO-RRR2. The user only chooses ϕ_{mean}.

Given that ϕ is always within convergence region, $\phi_{min} > 0$, and the range of randomness is reduced, this approach results in fewer and smaller local explosions than classical PSO (for high values of w and ϕ_{max}). However, since both ϕ_{mean} and the ϕ interval are always greater than for the PSO-RRR1 formulation, randomness is important and

the trajectories are more erratic for the PSO-RRR2. Note that while w and ϕ_{max} do, ϕ_{min} and ϕ_{mean} do not comply with Fig. 6.89. In fact, $\phi_{mean} > 2$ takes the particle farther from rather than closer to the attractor. This approach is typically more robust.

Additional comments

These guidelines are meant to be used mainly for mathematical optimization problems, where the search-space is abstract. In applications such as 'swarm robotics', there are other aspects to take into account. For instance, high amplitudes in the oscillations and big step sizes are not desirable in these cases, even if they result in a better exploration of the search-space. Therefore much smaller values of ϕ are probably advisable.

6.5. Closure

It was shown in this chapter that it does not make sense to make the constriction factor '*adaptive with respect to* ϕ' –which would be more in line with its theoretical development– because of the discontinuity in the curve 'ϕ–constricted ϕ' for $\phi < 4$ (refer to Fig. 6.1). Therefore, it is advisable to calculate it replacing ϕ by its upper-bound ϕ_{max} (*aw*), as it is of common practice in the literature. Note that other adaptive schemes (e.g. (Chen, Lee, Liao, & Dai, 2007)) are not explored in this thesis.

It was also shown that –rather counter-intuitively– setting the individuality weight to a value perceptively higher than that of the sociality weight may not be effective in delaying convergence and improving exploration. In fact, it might turn out to be harmful for the exploration abilities of the algorithm. Conversely, handling the settings of the coefficients in the velocity update equation comprises an effective means to control the form and speed of convergence.

In addition to the study of popular, successful settings imported from the literature, three approaches were developed in this chapter by reducing randomness in the classical PSO aiming to obtain a desired average behaviour while never allowing ϕ to leave the convergence region: the PSO-RRM, PSO-RRR1, and the PSO-RRR2. This resulted in convergent trajectories at different speeds according to the setting, which show the tendency to produce remarkably fewer local explosions than traditional PSO. Thus, trajec-

tories are more consistent for different runs. In addition, the resulting speed of convergence can be controlled by a single coefficient, since w and ϕ_{mean} are correlated so as to produce the desired average behaviour.

The effect of some preselected coefficients' settings on the dynamics of a small swarm of four interacting particles flying over 1-dimensional space was visually studied. Only three settings –two of which are in fact variations of the classical PSO algorithm rather than only settings– were selected to undergo a full test on a set of multidimensional benchmark problems (see Appendix II).

The results of the full algorithm on the set of benchmark multidimensional problems confirmed the expected behaviour associated with the selected settings, except that the PSO-RRR1 exhibited remarkably faster convergence than anticipated, even for high values of w. This means that the speed of convergence can also be controlled by means of the ϕ interval without affecting the average behaviour. These experiments were performed on a global neighbourhood and on a local ring topology with only two neighbours. Results showed that the combined effect of the coefficients' settings and the neighbourhood topology controls the balance between exploration and exploitation. The coefficients control the speed of convergence of each particle to its attractor, whereas the neighbourhood topology controls the spread of information in the social network.

Finally, guidelines as to how to set the coefficients to obtain a desired behaviour of the system are provided. They apply to the classical PSO; to a more general PSO; and to the PSO-RRR1 and PSO-RRR2 formulations proposed in this chapter.

Chapter 7

NEIGHBOURHOODS

A dynamic neighbourhood combining the global topology and the ring topology (with two neighbours) from the previous chapter is proposed, where the number of neighbours increases linearly as the search progresses. This neighbourhood structure is implemented for the same three coefficients' settings, and for the multi-swarm approach combining them, tested at the end of the previous chapter. Thus the four coefficients' settings with the proposed dynamic neighbourhood are tested on the same benchmark suite of side-constrained problems. The classical wheel topology and a random topology are also tested for reference and comparison purposes, while the results corresponding to the global and the ring topology with two neighbours are imported from the previous chapter. A so-called 'forward topology' is proposed, and experiments are carried out for two neighbours and for a dynamic number of neighbours in the same fashion as previously performed for the 'ring topology'. Additional neighbourhood-related techniques are proposed, namely a 'nearest neighbourhood' heuristics suitable for the ring and for the forward topologies, and a sub-neighbourhood option suitable for any neighbourhood structure. The nearest neighbourhood heuristics and the sub-neighbourhood option are tested on the multi-swarm coefficients' settings only, due to time and space constraints.

7.1. Introduction

As discussed and shown in previous chapters, different settings of the coefficients in the velocity update equation notably affect the behaviour of the swarm, as they govern the form and speed of convergence of each particle towards a randomly weighted average of its individual and social attractors. In turn, the neighbourhoods' structure governs the form and speed of spread of individually acquired information throughout the population, thus governing the update of every particle's social attractor. Therefore, the coefficients' settings and the neighbourhood topology together control the speed and form of convergence of the algorithm as a whole.

The global topology and a ring topology with two neighbours were considered in the experiments at the end of the previous chapter. Aiming to combine 'the robustness associated with the high degree of locality of the ring topology (with 2 neighbours)' with 'the speed of convergence and ability to perform fine-grain search of the global topology', a dynamic neighbourhood is proposed and tested in this chapter, for the four aforementioned coefficients' settings, on the side-constrained benchmark suite.

A so-called ‘forward topology’ is proposed, which shares some important characteristics with the ring topology. Namely, it allows any number of neighbours and guarantees full overlapping, so that the information acquired by any particle is guaranteed to eventually reach any other particle in the swarm, given enough time. The difference is in that the interconnections are not bidirectional in the forward topology. Experiments are carried out for the same four coefficients’ settings and on the same benchmark suite as previously performed for the ring topology, thus allowing a direct numerical comparison.

A possible strategy is proposed, which is aimed at taking advantage of proximity in the physical space, without renouncing the overlapping of the ring and forward topologies. This strategy is tested on the full benchmark suite, but for the multi-swarm coefficients’ settings only, due to time and space restrictions.

Finally, aiming to better exploit the strengths of different coefficients’ settings without interfering with one another, the strategy of dividing the swarm in sub-neighbourhoods, each with its own coefficients’ settings and topology, is proposed. Two alternative interconnections to pass information through sub-neighbourhoods are investigated. The multi-swarm coefficients’ setting is used to test this technique. Experiments are carried out using three sub-neighbourhoods, each with different coefficients’ settings.

7.2. Dynamic neighbourhood

If a given problem space is to be searched, for which there is no information available with regards to its landscape, it makes sense to scout the space first to identify potential good areas before spending too much effort in exploiting. This way the resources are expected to be better spent. With this in mind, and considering the previous experimental results, it is immediate to think of starting the search with a very local topology to identify good areas, gradually increasing the ‘globality’ of the social network so as to improve convergence and exploitation as the search progresses. The original idea was to start with the neighbourhoods composed by each particle itself only (i.e. number of neighbours equal to zero) and linearly increase the number of neighbours so that the neighbourhood topology becomes global (i.e. full cooperation) at the end of the search. Given that the experiments in the previous chapter were performed for the global topol-

ogy and the ring topology with two neighbours, the dynamic neighbourhood tested here linearly varies the number of neighbours from two to the swarm-size, so as to visualize the results obtained by a linear variation between them.

This neighbourhood structure is implemented for the same three coefficients' settings (PSO-RRR2-1, PSO-RRR1-1, and C-PSO-1), and for the multi-swarm (MS) approach combining them, which were tested at the end of the previous chapter. For comparison, and as a frame of reference, the experiments are also carried out for the classical wheel and a random topology, while the results obtained for the global topology and the ring topology with two neighbours are imported from the previous chapter. For the random topology, the number of neighbours for each particle at each time-step is randomly generated (*nn*), and then *nn* particles from the swarm are randomly chosen. The results are presented in Table 7.1 to Table 7.15, and in Fig. 7.1 to Fig. 7.27.

It is fair to note that the first dynamic topology in PSO –to the best of my knowledge– was proposed by Suganthan (1999), where the social attractor varied from the local to the swarm's best experience. An empirical formula to calculate a threshold is proposed. If the threshold is passed, the neighbourhood is global. Otherwise the neighbourhood is composed of the particles within a given normalized distance.

Another dynamic topology is the 'Stochastic Star' topology proposed in (Miranda, Keko, & Duque, 2008), which is a sort of generalization of the 'global topology'. At each time-step, and for each dimension of the search-space, there is a probability that a particle will not access the global best information, and therefore would move only under the influence of the inertia and the attraction towards its individual best experience.

A similar approach to the dynamic neighbourhood proposed here is that in (Richards & Ventura, 2003), where each particle starts accessing only one other particles' best experience, and the number of neighbours is increased at regular intervals until it becomes global by the time 80% of the search length has elapsed.

Other classical topologies that have not been tested in this thesis are the 'von Neumann' and the 'Pyramid' topologies (see (Engelbrecht, 2005, pp. 107-109) and (Kennedy & Mendes, 2006)). For further studies on neighbourhood topologies, refer to (Kennedy, 1998), (Kennedy, 1999), (Mendes, 2004), (Li, 2004), (Clerc, 2006a, pp. 87-101), (Abraham, Liu, & Chang, 2006), (Mohais, 2007), and (Akat & Gazi, 2008).

Table 7.1. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Sphere function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.74E-57	3.42E-54	4.65E-53	5.96E-52	4.79E-18	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.61E-53	8.64E-49	7.07E-46	1.26E-44	1.55E-18	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.76E-54	1.58E-50	1.80E-48	2.05E-47	5.87E-19	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.96E-54	7.33E-48	1.69E-39	4.23E-38	5.71E-19	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.74E-51	3.89E-49	2.60E-48	1.45E-47	2.63E-19	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	5.19E-88	2.30E-85	3.33E-84	4.17E-83	3.43E-37	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.09E-82	2.99E-80	2.13E-78	2.27E-77	1.52E-37	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.65E-84	8.87E-82	6.58E-81	8.62E-80	3.99E-38	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.81E-86	3.57E-80	1.30E-76	3.14E-75	1.59E-37	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.01E-86	4.41E-83	1.33E-82	7.29E-82	6.36E-37	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.74E-91	5.15E-88	2.06E-84	5.14E-83	9.16E-30	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	5.58E-82	3.44E-78	5.67E-76	6.45E-75	3.58E-32	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	7.89E-85	1.89E-81	2.99E-79	4.49E-78	4.59E-33	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.41E-87	2.54E-79	1.85E-67	4.63E-66	3.51E-33	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.41E-87	2.03E-84	1.54E-82	2.99E-81	3.73E-34	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.38E-90	4.67E-86	1.06E-83	2.16E-82	1.70E-20	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.39E-85	4.26E-78	9.85E-77	1.83E-75	2.68E-20	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.99E-86	1.12E-80	2.13E-79	2.33E-78	4.47E-20	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.69E-64	2.13E-48	8.29E-44	2.07E-42	1.92E-19	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.59E-85	2.47E-82	2.84E-80	5.68E-79	2.37E-20	-

Table 7.2. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Sphere function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	2.04E-256	3.90E-250	3.93E-247	8.75E-246	2.28E-126	100
			1000	4.97E-24	5.49E-23	1.25E-22	1.06E-21	8.08E-14	-
	RING	nn = 2	10000	1.29E-145	2.82E-143	3.37E-141	5.90E-140	1.13E-73	100
			1000	1.78E-13	3.67E-12	5.33E-12	2.02E-11	8.77E-09	-
	RING	nni = 2 nnf = (m – 1)	10000	1.46E-226	5.19E-223	5.65E-221	6.34E-220	1.32E-113	100
			1000	2.26E-16	7.12E-15	1.30E-14	6.38E-14	2.46E-10	-
	WHEEL		10000	9.84E-173	5.18E-161	2.52E-154	5.85E-153	4.05E-81	100
			1000	1.44E-15	2.61E-14	1.09E-12	1.14E-11	6.59E-10	-
	RANDOM		10000	1.31E-72	2.36E-67	6.47E-65	1.05E-63	6.16E-36	100
			1000	2.31E-06	6.63E-05	3.40E-04	4.98E-03	1.98E-05	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.75E-67	1.61E-65	5.10E-65	3.50E-64	5.98E-35	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.86E-35	2.94E-33	4.46E-33	1.53E-32	1.01E-19	-
	RING	nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.00E-44	7.30E-43	2.52E-42	1.96E-41	1.93E-24	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.62E-48	1.39E-43	1.02E-41	1.74E-40	9.63E-25	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.74E-54	8.67E-53	2.50E-52	1.48E-51	2.26E-29	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.30E-51	6.84E-50	3.49E-49	5.15E-48	7.98E-27	-
	RING	nn = 2	10000	1.13E-280	2.23E-277	4.17E-274	5.97E-273	2.11E-140	100
			1000	7.36E-27	3.12E-25	4.91E-25	2.77E-24	1.50E-15	-
	RING	nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	7.32E-33	1.59E-31	3.09E-31	2.69E-30	1.18E-18	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	6.57E-35	2.83E-30	2.16E-29	3.11E-28	3.41E-18	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.27E-36	3.86E-34	9.19E-33	1.87E-31	1.24E-19	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.21E-58	2.00E-56	2.03E-55	1.42E-54	2.17E-18	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.20E-153	100
			1000	1.41E-32	1.17E-30	1.06E-29	9.91E-29	2.68E-11	-
	RING	nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.18E-43	2.00E-39	5.49E-38	1.09E-36	1.35E-14	-
	WHEEL		10000	2.19E-202	5.77E-193	1.51E-186	2.67E-185	5.31E-97	100
			1000	6.55E-22	4.21E-17	2.35E-15	3.32E-14	2.55E-11	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.71E-44	2.13E-42	4.52E-41	4.89E-40	4.41E-17	-

Table 7.3. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Sphere function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.22E-87	3.29E-84	3.07E-82	6.31E-81	2.12E-45	100
			1000	3.49E-06	1.85E-05	4.08E-05	2.77E-04	1.84E-06	-
	RING	nn = 2	10000	3.77E-43	1.90E-42	7.86E-42	6.78E-41	9.24E-25	100
			1000	1.23E-01	2.84E-01	3.12E-01	6.69E-01	2.14E-04	-
		nni = 2 nnf = (m – 1)	10000	2.21E-74	8.42E-73	6.49E-72	6.44E-71	3.87E-40	100
			1000	9.91E-03	2.85E-02	2.98E-02	7.82E-02	4.65E-05	-
	WHEEL		10000	9.29E-50	8.45E-47	7.72E-45	1.17E-43	4.78E-27	100
			1000	3.04E-02	9.25E-02	1.34E-01	3.69E-01	5.88E-05	-
	RANDOM		10000	6.28E-07	5.97E-04	2.90E-03	2.00E-02	1.56E-05	20
			1000	2.75E+02	1.02E+03	1.25E+03	3.38E+03	1.01E-02	-
PSO-RRR1-1	GLOBAL		10000	4.06E-07	3.79E-04	9.89E-02	2.39E+00	4.69E-17	32
			1000	5.57E-06	1.64E-03	2.70E-01	4.03E+00	5.93E-11	-
	RING	nn = 2	10000	8.13E-144	7.26E-142	5.86E-141	6.69E-140	1.20E-74	100
			1000	1.73E-11	7.83E-11	8.50E-11	2.40E-10	2.05E-09	-
		nni = 2 nnf = (m – 1)	10000	2.04E-268	4.97E-257	6.23E-249	8.26E-248	1.16E-134	100
			1000	5.25E-17	2.48E-16	3.07E-16	1.05E-15	3.12E-12	-
	WHEEL		10000	4.05E-48	3.99E-40	3.64E-29	9.08E-28	1.20E-22	100
			1000	2.86E-05	1.73E-03	1.91E-02	2.02E-01	2.62E-06	-
	RANDOM		10000	1.19E-286	7.00E-282	1.74E-276	4.28E-275	7.38E-143	100
			1000	7.92E-27	7.54E-25	1.36E-23	2.88E-22	2.67E-16	-
C-PSO-1	GLOBAL		10000	3.05E-220	2.21E-212	1.42E-207	3.49E-206	1.57E-108	100
			1000	8.53E-20	1.09E-17	1.04E-16	9.10E-16	1.33E-12	-
	RING	nn = 2	10000	5.68E-96	1.67E-94	1.65E-93	3.57E-92	7.60E-51	100
			1000	7.53E-07	3.07E-06	3.62E-06	1.35E-05	5.46E-07	-
		nni = 2 nnf = (m – 1)	10000	3.06E-182	6.62E-179	1.98E-177	3.84E-176	3.81E-93	100
			1000	7.02E-10	4.53E-09	4.77E-09	1.48E-08	1.77E-08	-
	WHEEL		10000	4.49E-100	1.31E-95	8.45E-91	2.09E-89	2.29E-50	100
			1000	1.74E-08	9.80E-07	4.16E-06	2.23E-05	1.49E-07	-
	RANDOM		10000	4.50E-109	1.09E-103	7.90E-102	5.27E-101	4.94E-55	100
			1000	1.49E-08	1.67E-07	3.64E-07	3.06E-06	1.30E-07	-
Multi-Swarm	GLOBAL		10000	1.10E-181	4.68E-172	1.95E-166	4.21E-165	7.90E-88	100
			1000	4.53E-17	1.52E-14	4.02E-11	1.00E-09	3.63E-10	-
	RING	nn = 2	10000	3.13E-113	7.02E-109	2.72E-107	5.96E-106	6.93E-57	100
			1000	2.83E-08	9.73E-08	2.55E-07	3.82E-06	1.75E-06	-
		nni = 2 nnf = (m – 1)	10000	1.28E-185	6.46E-180	6.80E-173	1.68E-171	2.58E-91	100
			1000	7.07E-12	1.43E-10	3.51E-10	3.97E-09	1.57E-08	-
	WHEEL		10000	4.31E-56	1.21E-53	6.22E-52	6.12E-51	1.42E-30	100
			1000	8.25E-04	1.66E-02	3.24E-02	2.36E-01	2.49E-05	-
	RANDOM		10000	9.07E-180	1.28E-175	2.08E-172	3.32E-171	1.66E-90	100
			1000	1.80E-16	1.41E-14	1.28E-13	2.27E-12	1.40E-10	-

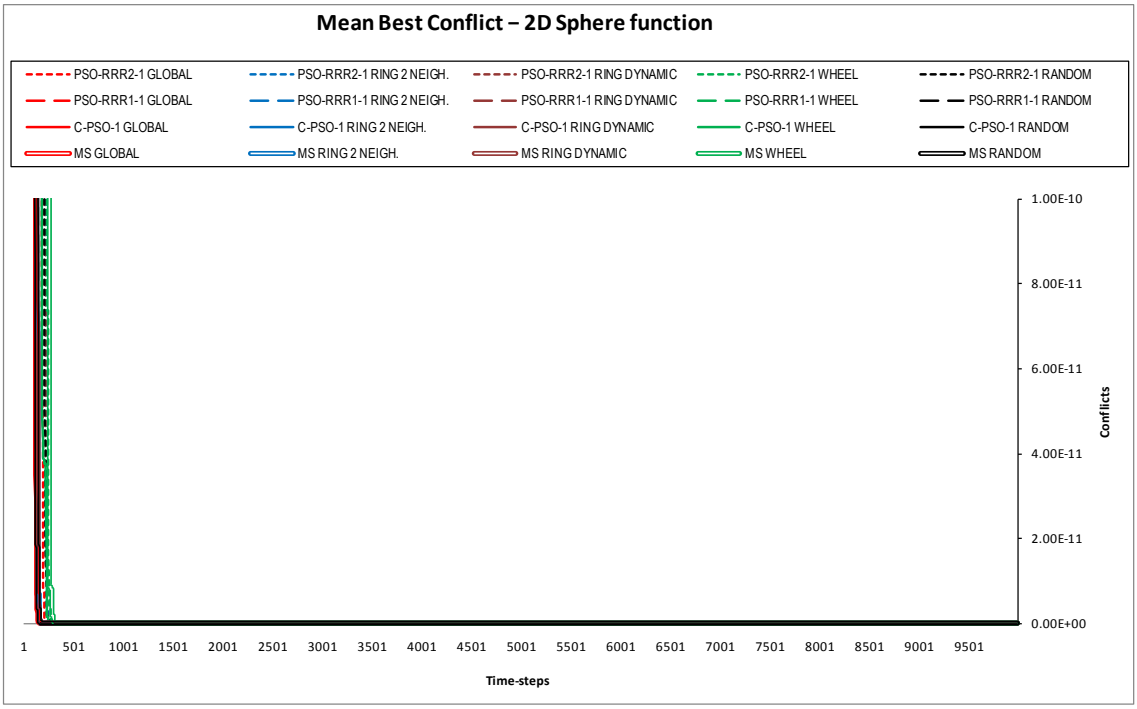


Fig. 7.1. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 7.1. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

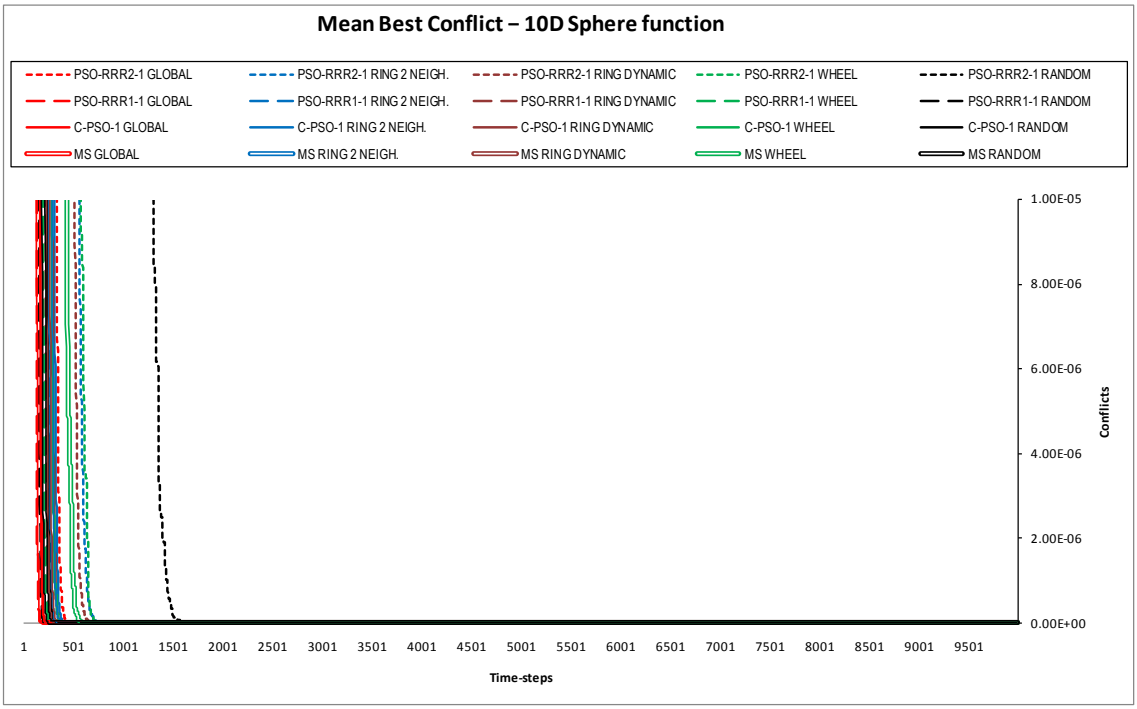


Fig. 7.2. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 7.2. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

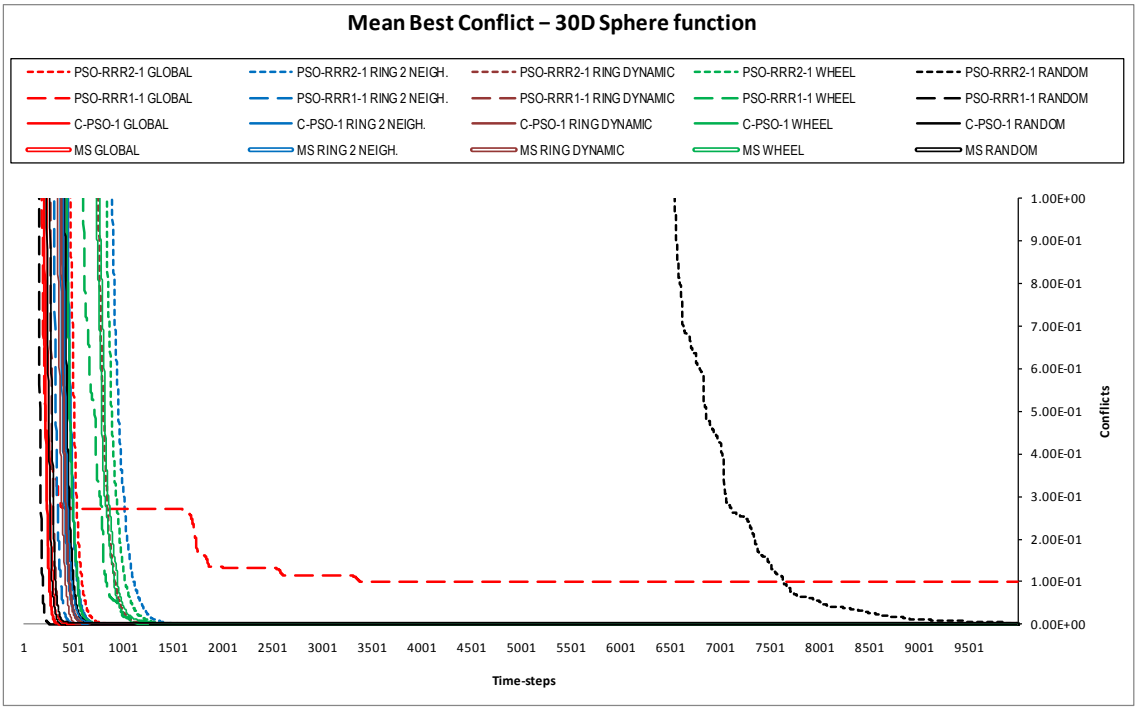


Fig. 7.3. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.3. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

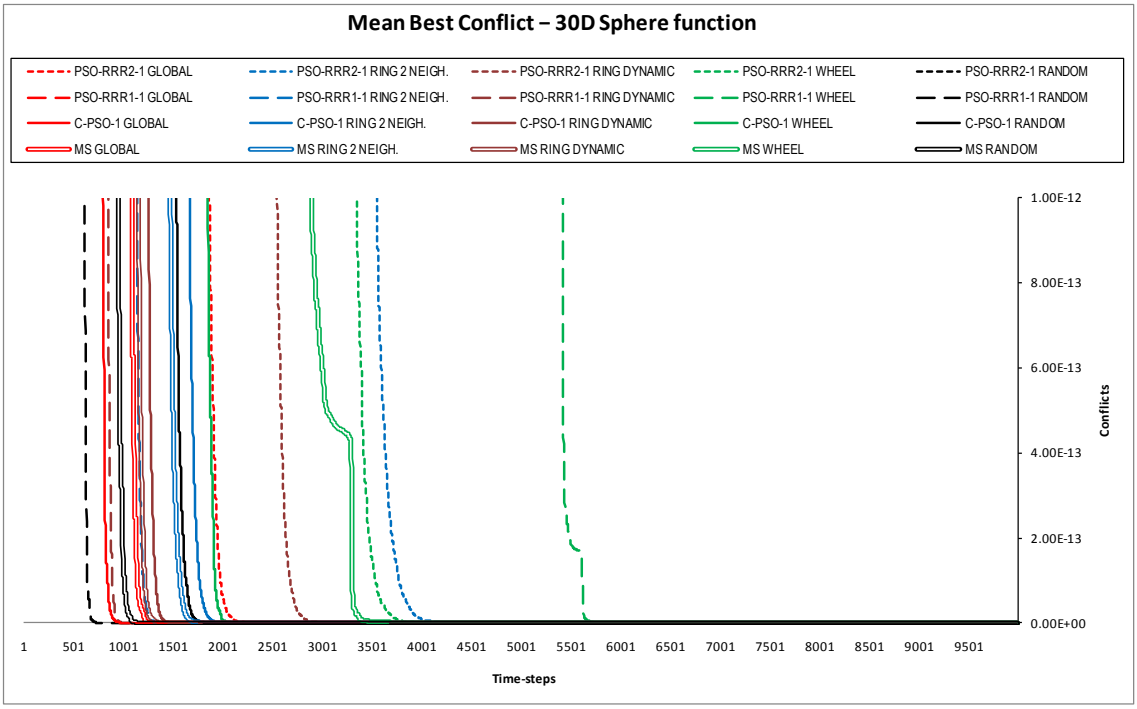


Fig. 7.4. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.3. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.4. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	1.54E-30	3.01E-28	4.82E-26	3.96E-25	3.72E-08	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	3.11E-20	3.03E-16	3.31E-15	3.09E-14	6.95E-06	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	3.71E-23	1.08E-18	2.30E-17	4.07E-16	4.46E-06	-
	WHEEL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	6.68E-25	1.00E-19	1.96E-15	4.47E-14	8.76E-07	-
	RANDOM		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	4.13E-20	5.56E-17	2.58E-15	5.86E-14	7.67E-07	-
PSO-RRR1-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.70E-20	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	4.93E-32	4.78E-27	6.69E-25	7.34E-24	1.27E-11	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	7.89E-33	1.97E-31	6.36E-15	-
	WHEEL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	3.20E-32	7.89E-31	3.08E-16	-
	RANDOM		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.39E-18	-
C-PSO-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.29E-11	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	5.77E-21	1.65E-15	6.19E-14	1.30E-12	2.76E-06	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	6.36E-24	2.02E-20	4.32E-19	5.48E-18	1.46E-07	-
	WHEEL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	4.44E-31	7.89E-26	2.30E-18	3.61E-17	6.35E-09	-
	RANDOM		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	1.11E-29	8.77E-28	1.56E-26	3.12E-09	-
Multi-Swarm	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.05E-09	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	9.00E-24	2.19E-18	3.50E-17	1.78E-06	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	1.77E-30	1.67E-26	3.95E-25	1.36E-06	-
	WHEEL		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	2.61E-29	7.67E-24	1.99E-16	4.89E-15	4.17E-07	-
	RANDOM		**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**100**
			1000	0.00E+00	0.00E+00	1.41E-28	3.49E-27	5.78E-09	-

Table 7.5. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.09E-06	2.72E-04	6.38E-01	3.99E+00	6.79E-03	32
			1000	2.22E-02	2.47E+00	5.31E+00	6.85E+01	2.34E-03	-
	RING	nn = 2	10000	6.79E-05	1.64E-02	1.82E-02	7.14E-02	7.64E-03	4
			1000	9.88E-03	1.64E+00	2.03E+00	5.50E+00	1.63E-02	-
		nni = 2 nnf = (m – 1)	10000	1.74E-06	3.60E-04	3.93E-04	1.15E-03	1.20E-02	24
			1000	1.72E-02	4.02E+00	3.24E+00	5.10E+00	1.71E-02	-
	WHEEL		10000	7.23E-04	1.57E-03	3.21E-01	3.99E+00	4.79E-03	0
			1000	3.35E+00	4.59E+00	4.70E+00	7.49E+00	6.32E-04	-
	RANDOM		10000	1.14E-01	1.57E+00	1.95E+00	5.93E+00	1.03E-03	0
			1000	6.91E-01	1.06E+01	3.96E+01	2.06E+02	8.98E-03	-
PSO-RRR1-1	GLOBAL		10000	1.13E-28	8.73E-01	1.64E+00	3.99E+00	1.08E-03	44
			1000	6.53E-06	1.46E+00	1.90E+00	4.99E+00	2.85E-03	-
	RING	nn = 2	10000	2.20E-10	3.45E-09	3.21E-08	5.63E-07	2.39E-03	100
			1000	5.10E-04	1.17E+00	1.30E+00	4.19E+00	1.11E-02	-
		nni = 2 nnf = (m – 1)	10000	5.92E-29	1.37E-28	1.59E-01	3.99E+00	1.36E-03	96
			1000	1.51E-04	1.16E-01	4.36E-01	4.17E+00	7.96E-03	-
	WHEEL		10000	5.31E-22	1.26E-17	7.97E-01	3.99E+00	2.60E-04	80
			1000	9.02E-03	3.45E-01	1.11E+00	4.46E+00	6.77E-04	-
	RANDOM		10000	1.44E-26	1.74E-22	3.19E-01	3.99E+00	2.59E-03	92
			1000	3.29E-03	9.70E-03	5.37E-01	4.98E+00	4.31E-03	-
C-PSO-1	GLOBAL		10000	1.18E-10	4.49E-06	4.79E-01	3.99E+00	8.56E-03	76
			1000	2.73E-03	5.03E-01	7.26E+00	8.06E+01	6.76E-03	-
	RING	nn = 2	10000	1.23E-08	1.29E-03	1.61E-01	3.99E+00	7.32E-03	8
			1000	1.18E-02	3.08E+00	2.69E+00	5.08E+00	1.90E-02	-
		nni = 2 nnf = (m – 1)	10000	3.97E-08	8.83E-07	6.94E-06	1.28E-04	8.35E-03	96
			1000	1.33E-03	2.28E+00	2.11E+00	3.79E+00	1.63E-02	-
	WHEEL		10000	6.82E-07	1.55E-05	7.97E-01	3.99E+00	5.04E-03	72
			1000	4.74E-02	2.79E+00	8.92E+00	7.60E+01	1.54E-03	-
	RANDOM		10000	2.90E-05	6.25E-04	4.79E-01	3.99E+00	8.31E-03	16
			1000	2.01E-01	2.51E+00	2.75E+00	6.57E+00	3.84E-03	-
Multi-Swarm	GLOBAL		10000	5.91E-12	2.94E-02	1.02E+00	5.59E+00	7.53E-03	16
			1000	9.37E-05	6.67E-01	1.47E+00	6.46E+00	7.93E-03	-
	RING	nn = 2	10000	4.01E-09	8.03E-07	1.61E-01	3.99E+00	8.87E-03	80
			1000	2.63E-03	1.62E+00	1.82E+00	5.11E+00	1.79E-02	-
		nni = 2 nnf = (m – 1)	10000	1.88E-15	1.16E-09	1.42E-05	3.49E-04	1.08E-02	96
			1000	5.39E-04	6.18E-01	8.49E-01	4.07E+00	1.45E-02	-
	WHEEL		10000	8.63E-07	1.79E-04	9.58E-01	4.00E+00	5.16E-03	32
			1000	2.41E-02	4.69E+00	5.62E+00	1.35E+01	3.19E-03	-
	RANDOM		10000	6.03E-16	6.41E-15	6.38E-01	3.99E+00	5.68E-03	84
			1000	3.40E-02	2.42E-01	8.58E-01	4.27E+00	4.87E-03	-

Table 7.6. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rosenbrock function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.41E-04	1.27E+01	1.03E+01	1.88E+01	1.45E-03	0
			1000	8.48E+00	2.80E+01	5.20E+01	1.24E+02	2.95E-04	-
	RING	nn = 2	10000	1.14E-01	1.00E+01	1.06E+01	2.31E+01	1.13E-02	0
			1000	4.82E+01	1.40E+02	1.48E+02	3.09E+02	1.57E-02	-
	RING	nni = 2 nnf = (m – 1)	10000	2.91E-07	1.46E+01	1.29E+01	2.28E+01	7.38E-03	4
			1000	2.84E+01	6.81E+01	7.44E+01	1.43E+02	1.16E-02	-
	WHEEL		10000	3.17E+00	1.89E+01	2.63E+01	7.94E+01	6.73E-04	0
			1000	3.52E+01	1.11E+02	1.60E+02	5.59E+02	3.21E-04	-
	RANDOM		10000	1.33E+01	9.97E+01	1.63E+02	5.42E+02	1.54E-03	0
			1000	8.75E+04	2.48E+05	3.90E+05	1.94E+06	2.56E-02	-
PSO-RRR1-1	GLOBAL		10000	2.27E+01	9.15E+01	1.06E+02	3.67E+02	3.98E-11	0
			1000	2.43E+01	9.38E+01	1.10E+02	3.69E+02	2.84E-08	-
	RING	nn = 2	10000	8.78E-03	7.24E+00	7.16E+00	1.91E+01	2.15E-03	0
			1000	8.28E+00	2.61E+01	4.28E+01	1.77E+02	4.38E-03	-
	RING	nni = 2 nnf = (m – 1)	10000	2.56E-17	9.97E-13	1.35E+00	9.73E+00	2.25E-03	72
			1000	2.21E+01	2.48E+01	3.96E+01	8.33E+01	2.28E-03	-
	WHEEL		10000	5.13E-04	2.33E+00	8.76E+00	8.10E+01	4.75E-06	0
			1000	2.84E+01	9.14E+01	1.16E+02	2.21E+02	1.97E-04	-
	RANDOM		10000	2.12E-14	7.86E-10	1.12E+00	3.99E+00	6.76E-04	72
			1000	4.23E+00	2.08E+01	3.14E+01	1.22E+02	1.54E-04	-
C-PSO-1	GLOBAL		10000	1.17E-05	3.90E-02	1.05E+00	4.02E+00	5.58E-03	8
			1000	1.55E+00	2.22E+01	3.58E+01	1.79E+02	9.02E-04	-
	RING	nn = 2	10000	2.89E-03	6.94E-01	3.39E+00	1.79E+01	7.42E-03	0
			1000	1.32E+01	2.89E+01	5.04E+01	1.45E+02	1.30E-02	-
	RING	nni = 2 nnf = (m – 1)	10000	3.41E-06	3.73E+00	3.16E+00	1.00E+01	6.55E-03	4
			1000	1.13E+01	2.63E+01	3.40E+01	8.12E+01	8.11E-03	-
	WHEEL		10000	1.02E-03	4.82E+00	4.80E+00	1.07E+01	1.33E-03	0
			1000	2.04E+01	7.57E+01	6.24E+01	1.77E+02	2.16E-05	-
	RANDOM		10000	3.07E-03	1.32E+01	1.09E+01	7.22E+01	1.63E-03	0
			1000	1.68E+01	6.96E+01	6.14E+01	2.16E+02	1.10E-04	-
Multi-Swarm	GLOBAL		10000	4.21E-08	1.67E+01	2.70E+01	7.68E+01	3.90E-03	4
			1000	2.33E-02	2.30E+01	4.38E+01	1.36E+02	2.59E-03	-
	RING	nn = 2	10000	9.14E-03	7.09E+00	6.59E+00	1.46E+01	3.01E-03	0
			1000	6.26E+00	7.08E+01	5.33E+01	8.71E+01	5.27E-03	-
	RING	nni = 2 nnf = (m – 1)	10000	1.56E-05	5.30E+00	6.36E+00	1.98E+01	2.76E-03	4
			1000	4.72E+00	2.73E+01	4.05E+01	1.34E+02	3.79E-03	-
	WHEEL		10000	2.16E-02	1.33E+01	1.98E+01	7.13E+01	6.47E-04	0
			1000	2.67E+01	1.29E+02	1.30E+02	2.98E+02	1.17E-04	-
	RANDOM		10000	2.60E-04	1.16E+00	2.11E+00	1.15E+01	5.07E-03	0
			1000	7.09E+00	2.55E+01	4.35E+01	1.14E+02	2.25E-03	-

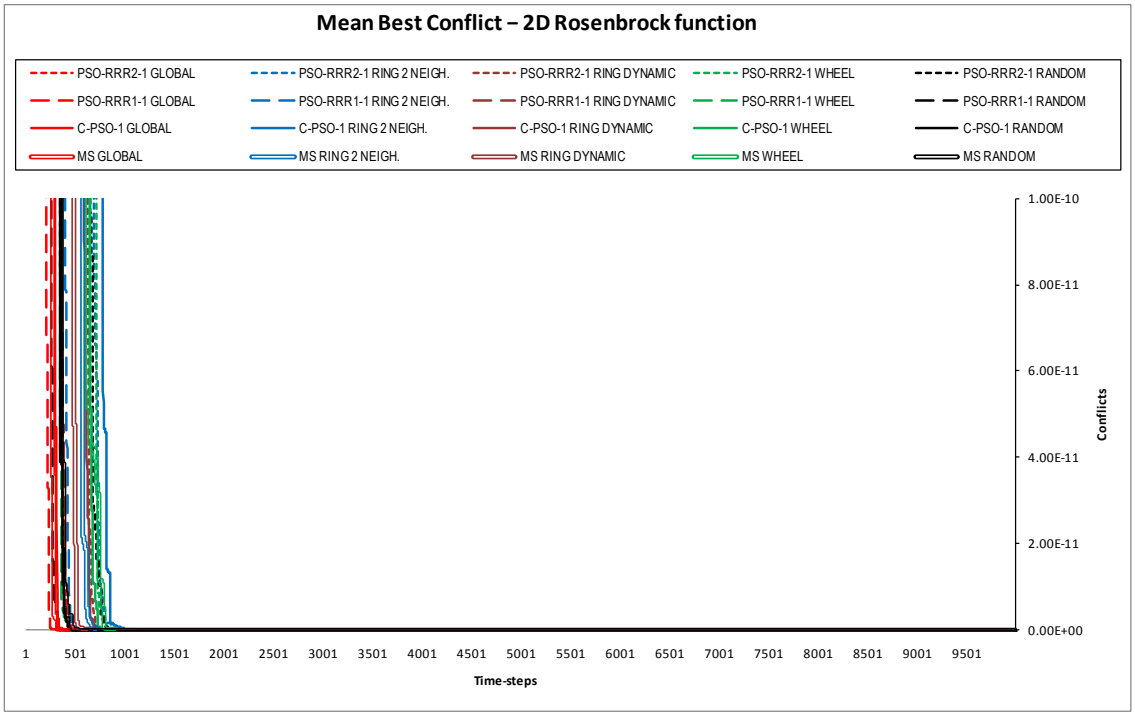


Fig. 7.5. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 7.4. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

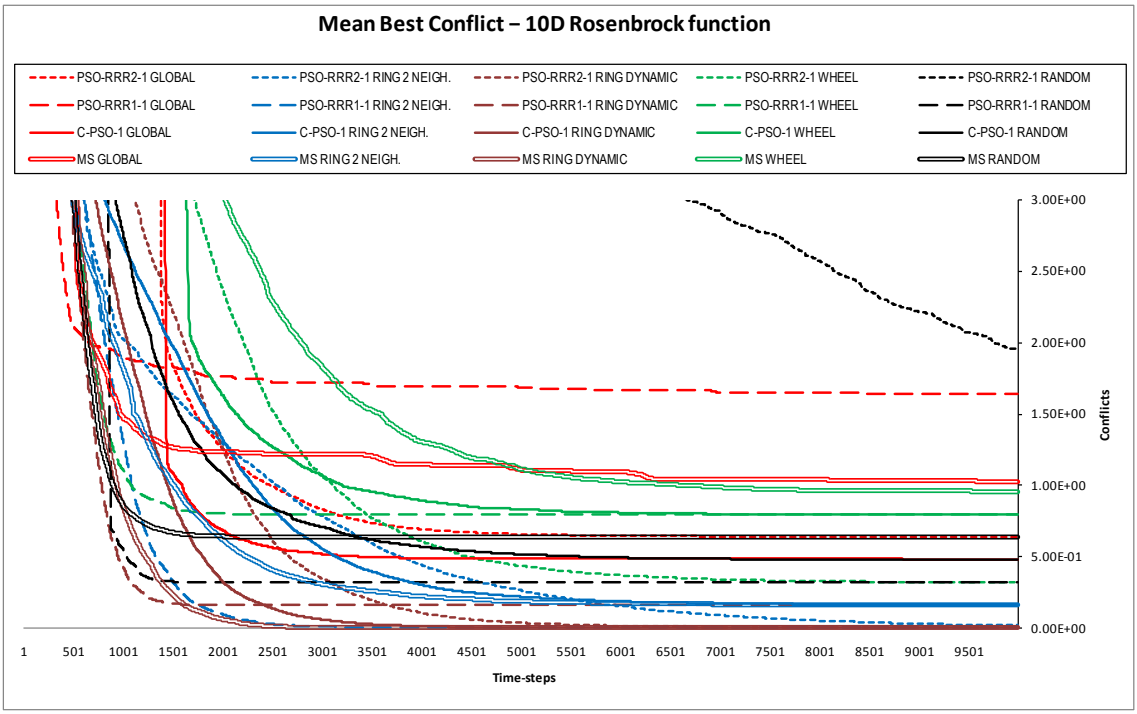


Fig. 7.6. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.5. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

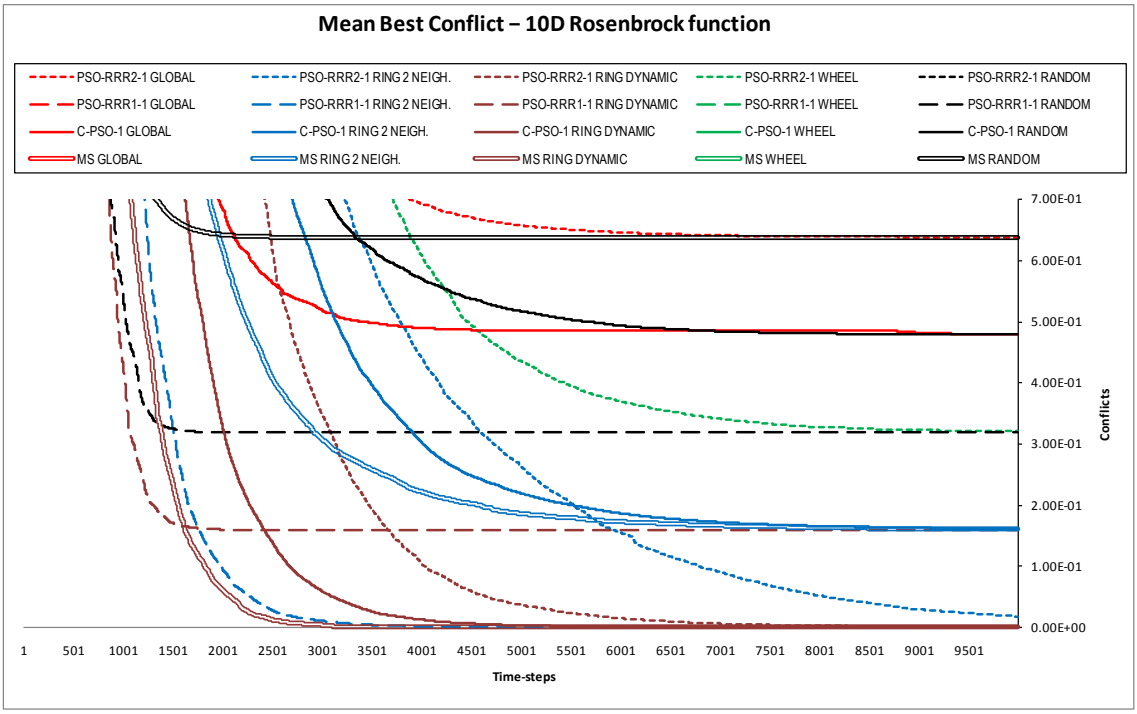


Fig. 7.7. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.5. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

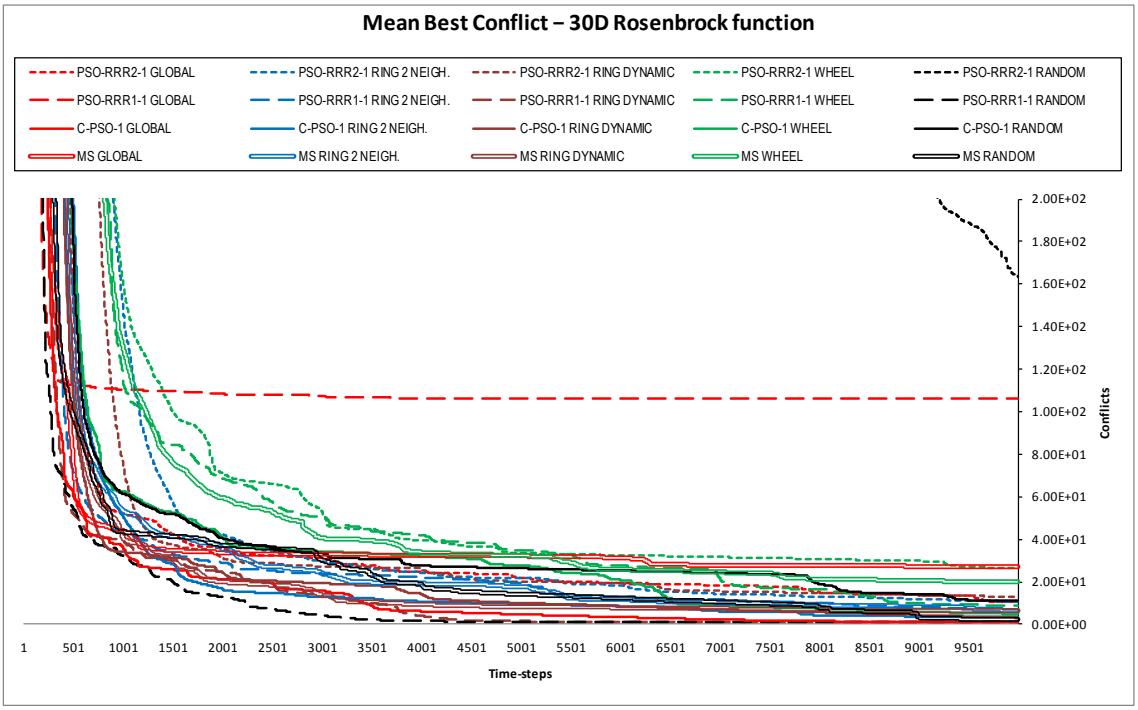


Fig. 7.8. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.6. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

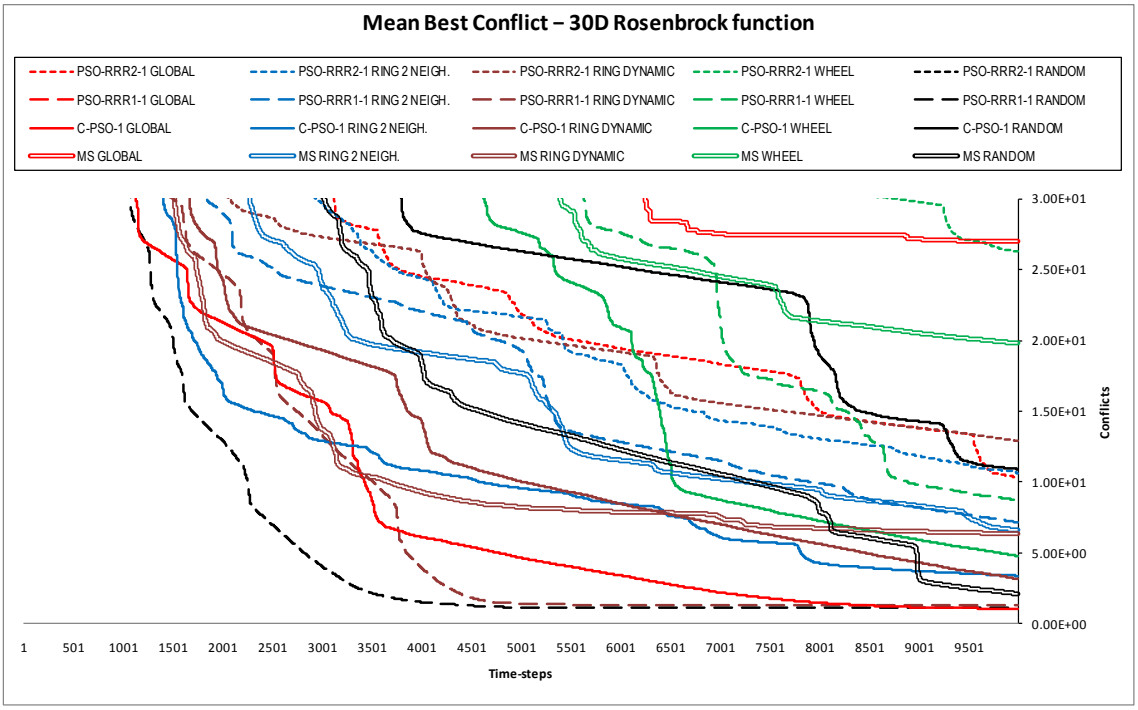


Fig. 7.9. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.6. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

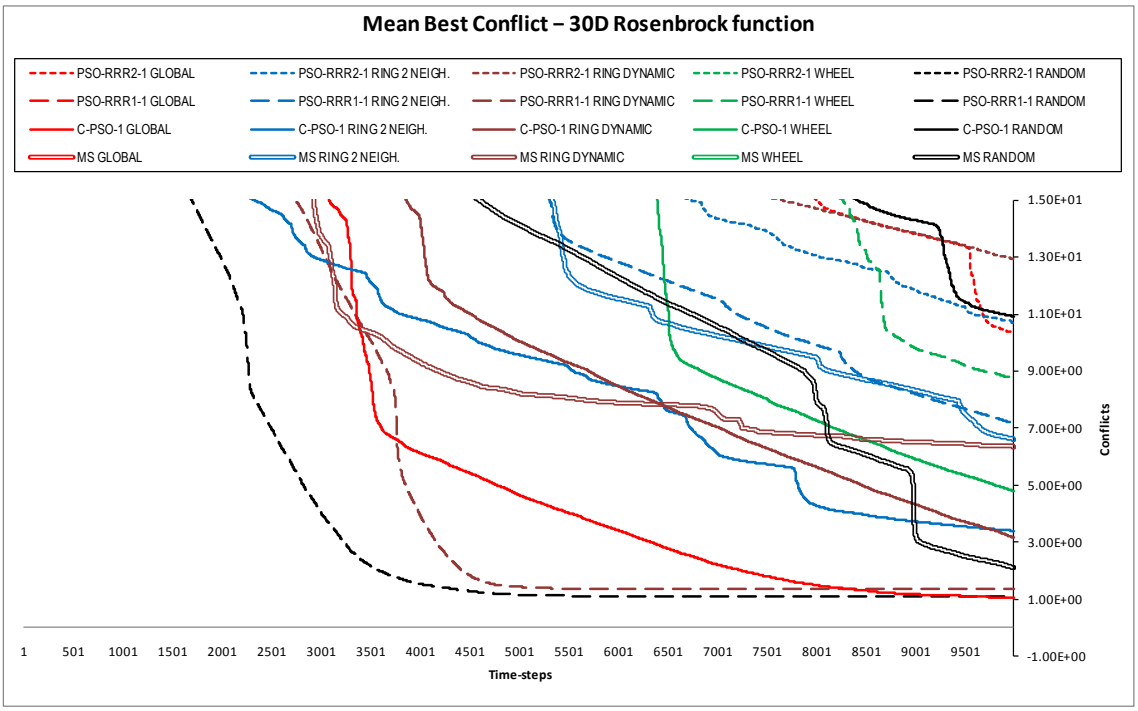


Fig. 7.10. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.6. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.7. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.07E-10	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.42E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.96E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.96E-11	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.01E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.01E-11	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.33E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.33E-11	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.09E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.09E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.05E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.05E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.60E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.60E-11	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.45E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.45E-11	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.35E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.35E-11	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.79E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.79E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.52E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.52E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.26E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.26E-11	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.03E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.03E-11	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.73E-11	100
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.81E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.81E-11	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.49E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.49E-11	-

Table 7.8. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	9.95E-01	2.98E+00	2.95E+00	6.96E+00	3.48E-04	0
			1000	9.95E-01	2.98E+00	3.02E+00	6.96E+00	2.20E-03	-
	RING	nn = 2	10000	0.00E+00	1.99E+00	2.15E+00	4.97E+00	2.71E-02	20
			1000	1.99E+00	4.22E+00	4.43E+00	7.96E+00	3.54E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	7.16E-01	3.98E+00	1.65E-02	60
			1000	9.95E-01	3.98E+00	4.04E+00	1.09E+01	3.51E-02	-
	WHEEL		10000	9.95E-01	1.99E+00	2.47E+00	5.97E+00	8.48E-04	0
			1000	9.97E-01	2.99E+00	3.09E+00	5.97E+00	7.73E-03	-
	RANDOM		10000	0.00E+00	1.19E+01	1.18E+01	2.87E+01	3.01E-02	12
			1000	7.30E+00	2.51E+01	2.45E+01	3.65E+01	5.00E-02	-
PSO-RRR1-1	GLOBAL		10000	5.97E+00	1.19E+01	1.35E+01	2.49E+01	1.21E-11	0
			1000	5.97E+00	1.19E+01	1.35E+01	2.49E+01	1.47E-11	-
	RING	nn = 2	10000	9.95E-01	4.97E+00	5.18E+00	1.09E+01	3.82E-02	0
			1000	2.98E+00	7.96E+00	7.94E+00	1.37E+01	4.30E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.98E+00	3.02E+00	5.97E+00	2.79E-02	8
			1000	2.98E+00	5.97E+00	7.44E+00	1.59E+01	4.40E-02	-
	WHEEL		10000	4.97E+00	8.95E+00	9.47E+00	1.99E+01	1.80E-04	0
			1000	4.97E+00	8.95E+00	9.83E+00	1.99E+01	1.91E-03	-
	RANDOM		10000	9.95E-01	4.97E+00	5.21E+00	8.95E+00	2.78E-11	0
			1000	9.95E-01	6.96E+00	7.01E+00	2.63E+01	4.96E-03	-
C-PSO-1	GLOBAL		10000	1.99E+00	3.98E+00	4.93E+00	1.09E+01	1.92E-11	0
			1000	1.99E+00	4.97E+00	5.17E+00	1.19E+01	8.71E-04	-
	RING	nn = 2	10000	0.00E+00	2.98E+00	2.79E+00	4.97E+00	2.93E-02	12
			1000	1.99E+00	3.98E+00	4.55E+00	7.96E+00	3.79E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.23E+00	5.97E+00	1.98E-02	48
			1000	1.99E+00	3.98E+00	4.66E+00	7.96E+00	3.65E-02	-
	WHEEL		10000	0.00E+00	2.98E+00	3.26E+00	7.96E+00	4.96E-04	4
			1000	0.00E+00	2.99E+00	3.54E+00	7.96E+00	2.41E-03	-
	RANDOM		10000	0.00E+00	1.99E+00	2.71E+00	2.21E+01	2.64E-03	12
			1000	9.95E-01	5.97E+00	1.12E+01	3.50E+01	2.43E-02	-
Multi-Swarm	GLOBAL		10000	1.99E+00	3.98E+00	4.78E+00	1.49E+01	1.84E-11	0
			1000	1.99E+00	3.98E+00	5.13E+00	1.49E+01	5.46E-04	-
	RING	nn = 2	10000	0.00E+00	2.98E+00	2.75E+00	6.96E+00	2.90E-02	4
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.68E+00	5.97E+00	1.94E-02	32
			1000	2.18E+00	4.97E+00	5.26E+00	1.09E+01	3.66E-02	-
	WHEEL		10000	0.00E+00	3.98E+00	4.42E+00	8.95E+00	7.48E-04	4
			1000	2.42E-12	4.97E+00	4.82E+00	8.95E+00	4.10E-03	-
	RANDOM		10000	9.95E-01	2.98E+00	2.91E+00	4.97E+00	1.93E-03	0
			1000	9.95E-01	4.98E+00	9.76E+00	3.80E+01	2.37E-02	-

Table 7.9. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rastrigin function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	2.69E+01	4.28E+01	4.13E+01	5.57E+01	2.64E-11	0
			1000	2.69E+01	4.28E+01	4.14E+01	5.57E+01	3.99E-05	-
	RING	nn = 2	10000	2.98E+01	4.40E+01	4.29E+01	5.29E+01	2.61E-02	0
			1000	3.46E+01	5.32E+01	5.24E+01	7.23E+01	3.00E-02	-
		nni = 2 nnf = (m – 1)	10000	2.69E+01	4.28E+01	4.31E+01	6.96E+01	2.97E-02	0
			1000	3.84E+01	4.88E+01	5.19E+01	7.99E+01	2.99E-02	-
	WHEEL		10000	2.09E+01	3.68E+01	3.77E+01	6.77E+01	2.23E-04	0
			1000	2.82E+01	4.41E+01	4.54E+01	6.98E+01	5.67E-03	-
	RANDOM		10000	2.50E+01	1.77E+02	1.42E+02	2.19E+02	2.81E-02	0
			1000	1.23E+02	2.01E+02	1.99E+02	2.48E+02	3.14E-02	-
PSO-RRR1-1	GLOBAL		10000	2.49E+01	7.16E+01	7.41E+01	1.28E+02	6.68E-16	0
			1000	2.49E+01	7.16E+01	7.41E+01	1.28E+02	1.47E-15	-
	RING	nn = 2	10000	2.19E+01	4.68E+01	4.65E+01	6.17E+01	3.00E-02	0
			1000	2.20E+01	5.01E+01	5.03E+01	6.71E+01	3.12E-02	-
		nni = 2 nnf = (m – 1)	10000	2.29E+01	4.88E+01	4.91E+01	7.46E+01	3.15E-02	0
			1000	3.48E+01	4.88E+01	5.13E+01	7.98E+01	3.27E-02	-
	WHEEL		10000	4.58E+01	7.36E+01	6.90E+01	9.35E+01	4.60E-12	0
			1000	4.58E+01	7.36E+01	6.90E+01	9.35E+01	6.62E-04	-
	RANDOM		10000	2.49E+01	4.78E+01	4.94E+01	7.46E+01	2.99E-11	0
			1000	2.49E+01	4.88E+01	5.98E+01	2.10E+02	1.58E-03	-
C-PSO-1	GLOBAL		10000	2.69E+01	4.88E+01	5.37E+01	9.65E+01	1.93E-11	0
			1000	2.69E+01	4.88E+01	5.37E+01	9.65E+01	1.09E-10	-
	RING	nn = 2	10000	2.89E+01	5.37E+01	5.05E+01	6.87E+01	3.31E-02	0
			1000	2.89E+01	5.88E+01	5.59E+01	7.79E+01	3.43E-02	-
		nni = 2 nnf = (m – 1)	10000	2.19E+01	5.27E+01	5.11E+01	7.36E+01	3.15E-02	0
			1000	2.69E+01	5.77E+01	5.59E+01	8.28E+01	3.36E-02	-
	WHEEL		10000	2.98E+01	4.88E+01	5.13E+01	8.36E+01	3.01E-04	0
			1000	2.98E+01	4.88E+01	5.27E+01	8.36E+01	6.88E-04	-
	RANDOM		10000	2.29E+01	3.48E+01	3.73E+01	6.57E+01	1.07E-04	0
			1000	2.30E+01	1.36E+02	1.28E+02	2.25E+02	1.91E-02	-
Multi-Swarm	GLOBAL		10000	2.59E+01	5.27E+01	5.33E+01	8.16E+01	1.89E-11	0
			1000	2.59E+01	5.27E+01	5.33E+01	8.16E+01	5.36E-08	-
	RING	nn = 2	10000	3.28E+01	4.48E+01	4.56E+01	6.37E+01	3.03E-02	0
			1000	3.32E+01	5.21E+01	4.97E+01	6.57E+01	3.09E-02	-
		nni = 2 nnf = (m – 1)	10000	2.59E+01	3.98E+01	4.32E+01	6.67E+01	2.77E-02	0
			1000	2.72E+01	4.48E+01	4.77E+01	6.83E+01	2.96E-02	-
	WHEEL		10000	1.69E+01	4.48E+01	4.61E+01	6.96E+01	1.04E-04	0
			1000	1.71E+01	4.68E+01	4.77E+01	7.77E+01	2.40E-03	-
	RANDOM		10000	1.99E+01	3.78E+01	3.86E+01	6.67E+01	3.08E-11	0
			1000	2.19E+01	4.18E+01	6.60E+01	2.00E+02	5.51E-03	-

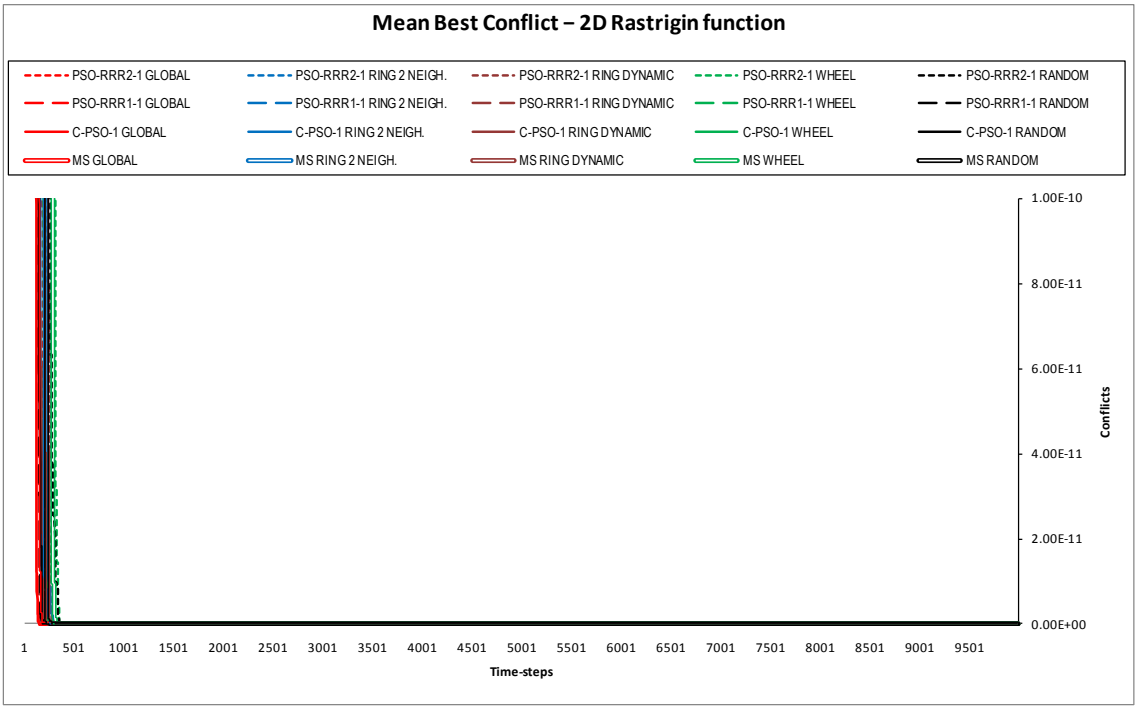


Fig. 7.11. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 7.7. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

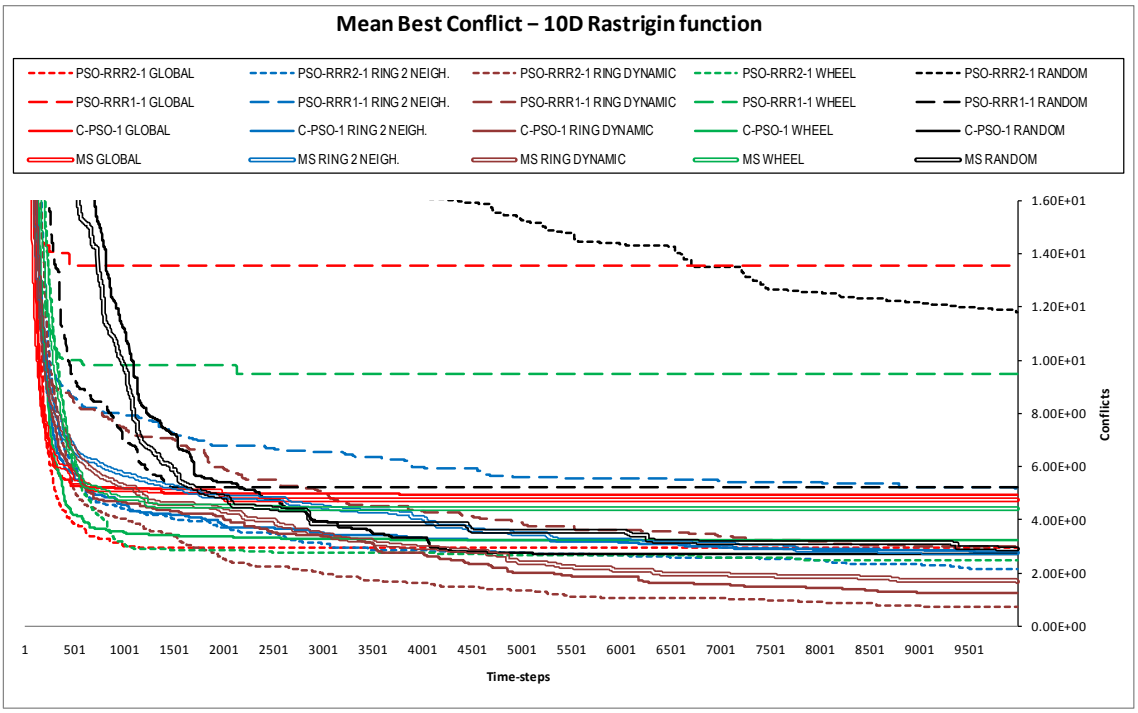


Fig. 7.12. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.8. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

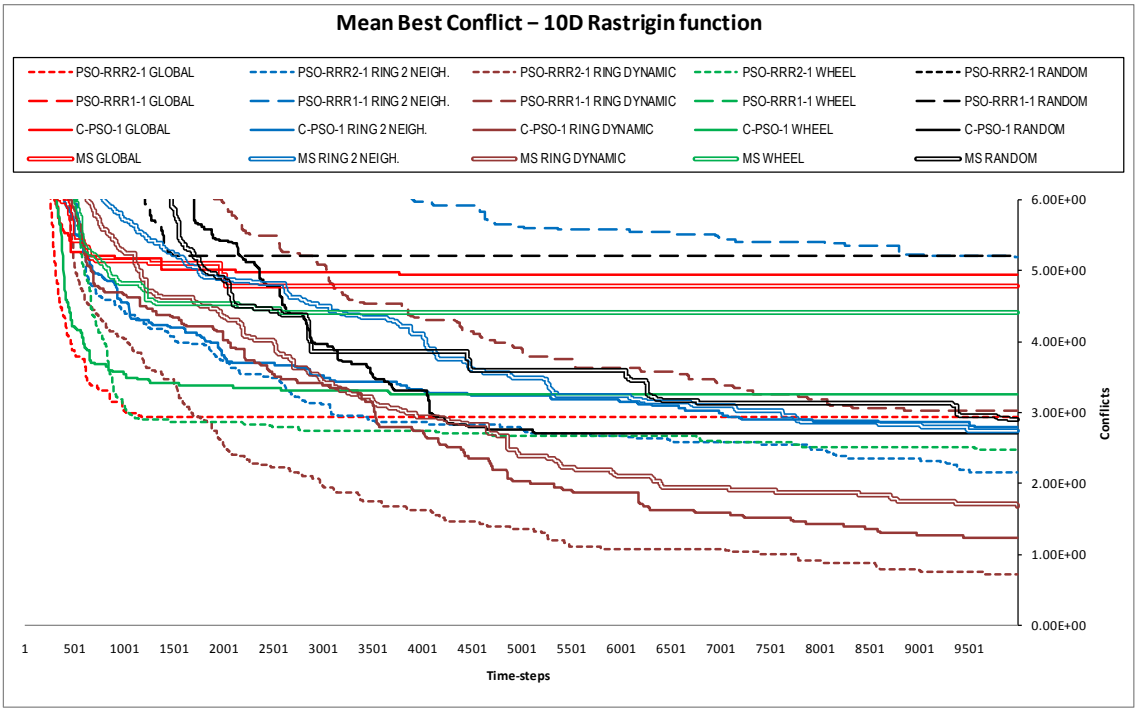


Fig. 7.13. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.8. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

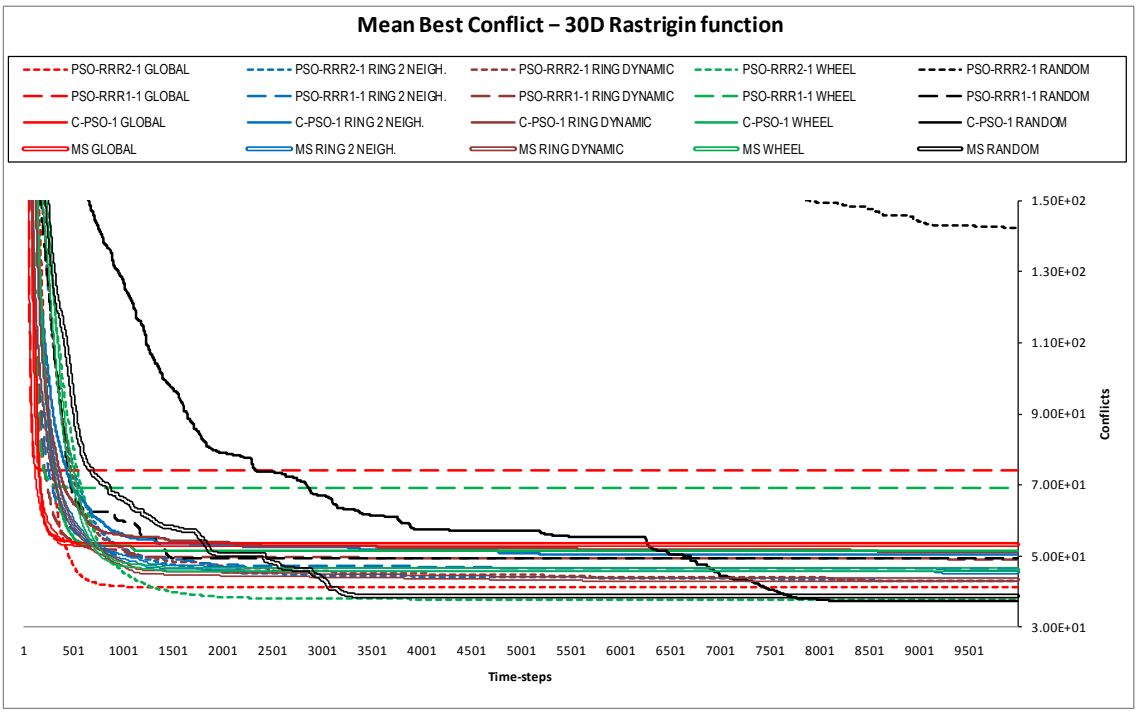


Fig. 7.14. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.9. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

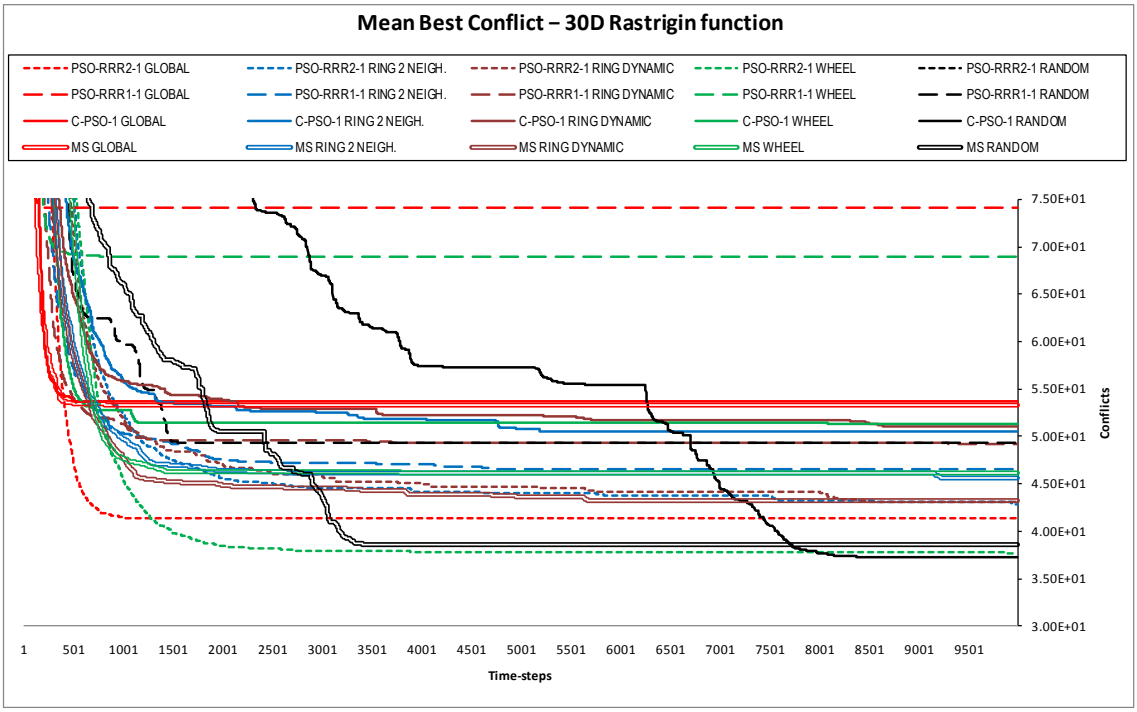


Fig. 7.15. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.9. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

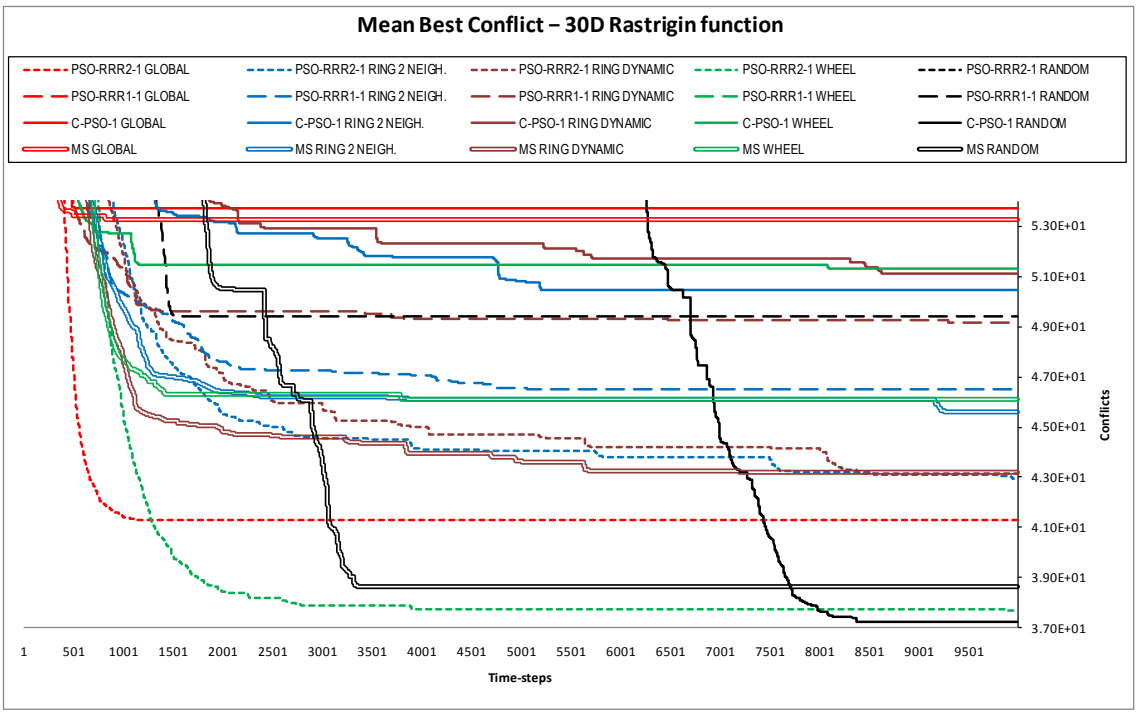


Fig. 7.16. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.9. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.10. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Griewank function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	0.00E+00	2.96E-04	7.40E-03	7.41E-12	96
			1000	0.00E+00	0.00E+00	5.92E-04	7.40E-03	1.10E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.65E-04	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.10E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.57E-12	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.64E-03	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.57E-05	100
			1000	0.00E+00	0.00E+00	2.10E-04	4.65E-03	1.56E-03	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.05E-122	100
			1000	0.00E+00	0.00E+00	1.70E-08	3.18E-07	1.84E-03	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	5.92E-04	7.40E-03	5.76E-12	92
			1000	0.00E+00	0.00E+00	1.18E-03	9.08E-04	9.08E-04	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.15E-04	100
			1000	0.00E+00	0.00E+00	3.02E-13	7.54E-12	2.05E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.45E-12	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.67E-03	-
	WHEEL		10000	0.00E+00	0.00E+00	8.88E-04	7.40E-03	2.96E-05	88
			1000	0.00E+00	0.00E+00	9.21E-04	7.96E-03	9.56E-04	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.32E-12	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.54E-04	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.40E-12	100
			1000	0.00E+00	0.00E+00	1.18E-03	7.40E-03	9.46E-04	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.57E-04	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.96E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.81E-12	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.67E-03	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.55E-09	100
			1000	0.00E+00	0.00E+00	3.99E-04	7.40E-03	1.37E-03	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.50E-12	100
			1000	0.00E+00	0.00E+00	3.60E-12	9.00E-11	1.01E-03	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.56E-12	100
			1000	0.00E+00	0.00E+00	5.95E-16	1.49E-14	8.92E-04	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.51E-04	100
			1000	0.00E+00	0.00E+00	2.39E-05	5.88E-04	2.05E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.74E-12	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.59E-03	-
	WHEEL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.51E-12	100
			1000	0.00E+00	0.00E+00	2.99E-04	7.40E-03	1.25E-03	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.64E-12	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.16E-03	-

Table 7.11. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Griewank function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.97E-02	5.66E-02	6.81E-02	1.43E-01	4.94E-07	0
			1000	1.97E-02	6.16E-02	7.14E-02	1.43E-01	1.18E-04	-
	RING	nn = 2	10000	0.00E+00	2.46E-02	2.66E-02	6.15E-02	1.81E-03	4
			1000	2.96E-07	2.95E-02	3.41E-02	6.88E-02	2.01E-03	-
		nni = 2 nnf = (m – 1)	10000	7.40E-03	2.96E-02	3.02E-02	5.65E-02	1.35E-03	0
			1000	9.86E-03	3.94E-02	3.68E-02	5.66E-02	1.54E-03	-
	WHEEL		10000	7.40E-03	5.66E-02	5.98E-02	1.48E-01	6.64E-05	0
			1000	7.40E-03	6.64E-02	7.03E-02	1.52E-01	4.81E-04	-
	RANDOM		10000	3.20E-02	4.48E-01	3.97E-01	7.91E-01	2.89E-03	0
			1000	1.23E-01	6.21E-01	6.02E-01	2.99E-03	2.99E-03	-
PSO-RRR1-1	GLOBAL		10000	2.96E-02	9.11E-02	9.27E-02	1.82E-01	1.81E-12	0
			1000	2.96E-02	9.11E-02	9.27E-02	1.82E-01	1.48E-05	-
	RING	nn = 2	10000	0.00E+00	3.19E-02	3.05E-02	7.38E-02	1.58E-03	8
			1000	0.00E+00	3.94E-02	3.64E-02	7.62E-02	1.66E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.22E-02	2.95E-02	6.64E-02	1.27E-03	12
			1000	9.86E-03	4.18E-02	4.26E-02	1.11E-01	1.61E-03	-
	WHEEL		10000	1.97E-02	9.11E-02	1.24E-01	3.84E-01	1.24E-12	0
			1000	1.97E-02	9.11E-02	1.27E-01	3.84E-01	6.60E-05	-
	RANDOM		10000	1.72E-02	5.66E-02	5.60E-02	9.35E-02	1.07E-04	0
			1000	2.95E-02	6.89E-02	1.09E-01	5.28E-01	4.53E-04	-
C-PSO-1	GLOBAL		10000	1.97E-02	6.64E-02	6.68E-02	1.38E-01	1.65E-06	0
			1000	2.71E-02	6.89E-02	7.21E-02	1.38E-01	1.22E-04	-
	RING	nn = 2	10000	0.00E+00	2.46E-02	2.36E-02	4.68E-02	1.55E-03	4
			1000	0.00E+00	2.71E-02	2.85E-02	8.87E-02	1.66E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.71E-02	2.49E-02	5.91E-02	1.19E-03	4
			1000	7.40E-03	3.69E-02	3.52E-02	7.38E-02	1.52E-03	-
	WHEEL		10000	0.00E+00	6.65E-02	6.65E-02	1.30E-01	2.23E-05	4
			1000	1.97E-02	7.13E-02	7.34E-02	1.30E-01	4.00E-04	-
	RANDOM		10000	9.86E-03	7.13E-02	1.58E-01	4.75E-01	6.70E-04	0
			1000	2.21E-02	4.25E-01	3.69E-01	7.91E-01	1.47E-03	-
Multi-Swarm	GLOBAL		10000	1.48E-02	6.64E-02	6.64E-02	1.38E-01	1.48E-05	0
			1000	2.95E-02	7.13E-02	7.85E-02	1.85E-01	1.40E-04	-
	RING	nn = 2	10000	0.00E+00	1.97E-02	2.15E-02	5.90E-02	1.60E-03	8
			1000	7.40E-03	2.22E-02	2.71E-02	5.90E-02	1.75E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.46E-02	2.77E-02	6.89E-02	1.47E-03	8
			1000	0.00E+00	3.45E-02	3.59E-02	7.38E-02	1.74E-03	-
	WHEEL		10000	1.97E-02	6.15E-02	6.50E-02	1.45E-01	1.70E-05	0
			1000	4.55E-02	7.38E-02	8.35E-02	1.45E-01	3.40E-04	-
	RANDOM		10000	1.72E-02	3.94E-02	5.85E-02	2.49E-01	7.33E-05	0
			1000	1.72E-02	6.90E-02	1.28E-01	6.89E-01	6.48E-04	-

Table 7.12. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Griewank function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	7.40E-03	9.35E-03	2.96E-02	4.23E-12	44
			1000	7.95E-06	7.44E-03	9.40E-03	2.96E-02	2.93E-06	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	2.96E-04	7.40E-03	1.22E-06	96
			1000	2.54E-01	4.15E-01	4.23E-01	6.79E-01	2.52E-04	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	4.05E-03	1.72E-02	3.77E-06	64
			1000	3.65E-02	7.18E-02	8.05E-02	1.70E-01	1.13E-04	-
	WHEEL		10000	0.00E+00	7.40E-03	1.03E-02	3.94E-02	2.15E-12	40
			1000	6.00E-02	2.26E-01	2.86E-01	7.71E-01	9.81E-05	-
	RANDOM		10000	6.05E-04	3.52E-02	1.63E-01	1.02E+00	1.38E-04	0
			1000	3.47E+00	1.02E+01	1.23E+01	3.14E+01	1.01E-02	-
PSO-RRR1-1	GLOBAL		10000	3.29E-08	6.46E-02	1.02E-01	7.40E-01	3.91E-13	4
			1000	3.29E-08	7.11E-02	1.12E-01	7.40E-01	4.04E-14	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	6.90E-04	9.86E-03	6.40E-08	92
			1000	4.02E-11	7.73E-10	6.91E-04	9.86E-03	1.38E-06	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	1.58E-03	9.86E-03	8.49E-09	80
			1000	0.00E+00	4.76E-08	3.15E-03	1.48E-02	1.11E-06	-
	WHEEL		10000	4.58E-13	8.57E-02	1.46E-01	8.89E-01	2.96E-13	4
			1000	7.63E-03	9.45E-02	1.77E-01	9.38E-01	1.72E-06	-
	RANDOM		10000	0.00E+00	9.86E-03	1.47E-02	7.36E-02	4.13E-12	28
			1000	0.00E+00	9.86E-03	1.47E-02	7.36E-02	3.71E-12	-
C-PSO-1	GLOBAL		10000	0.00E+00	1.23E-02	1.79E-02	7.09E-02	2.56E-12	36
			1000	0.00E+00	1.23E-02	1.79E-02	7.09E-02	2.93E-12	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.14E-07	100
			1000	4.55E-06	3.98E-05	2.60E-03	2.22E-02	1.52E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	1.97E-03	1.72E-02	4.73E-07	84
			1000	3.22E-09	3.49E-08	2.37E-03	1.72E-02	8.37E-06	-
	WHEEL		10000	0.00E+00	9.86E-03	3.55E-02	2.37E-01	1.68E-12	24
			1000	7.76E-08	9.87E-03	3.55E-02	2.37E-01	1.62E-07	-
	RANDOM		10000	0.00E+00	7.40E-03	9.15E-03	5.15E-02	5.03E-12	44
			1000	3.95E-08	7.40E-03	9.15E-03	5.16E-02	4.29E-07	-
Multi-Swarm	GLOBAL		10000	0.00E+00	4.67E-02	5.18E-02	1.41E-01	2.42E-12	4
			1000	6.66E-16	4.67E-02	5.18E-02	1.41E-01	4.06E-08	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	2.17E-03	1.23E-02	1.10E-06	76
			1000	6.02E-08	7.40E-03	6.97E-03	3.92E-02	3.08E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	6.39E-03	3.92E-02	4.12E-12	64
			1000	9.47E-11	7.40E-03	9.54E-03	3.92E-02	1.26E-05	-
	WHEEL		10000	0.00E+00	1.23E-02	1.75E-02	1.30E-01	1.66E-12	44
			1000	4.12E-03	5.41E-02	1.10E-01	4.16E-01	3.36E-05	-
	RANDOM		10000	0.00E+00	7.40E-03	1.43E-02	7.09E-02	5.23E-12	44
			1000	2.66E-15	7.40E-03	1.43E-02	7.09E-02	2.14E-07	-

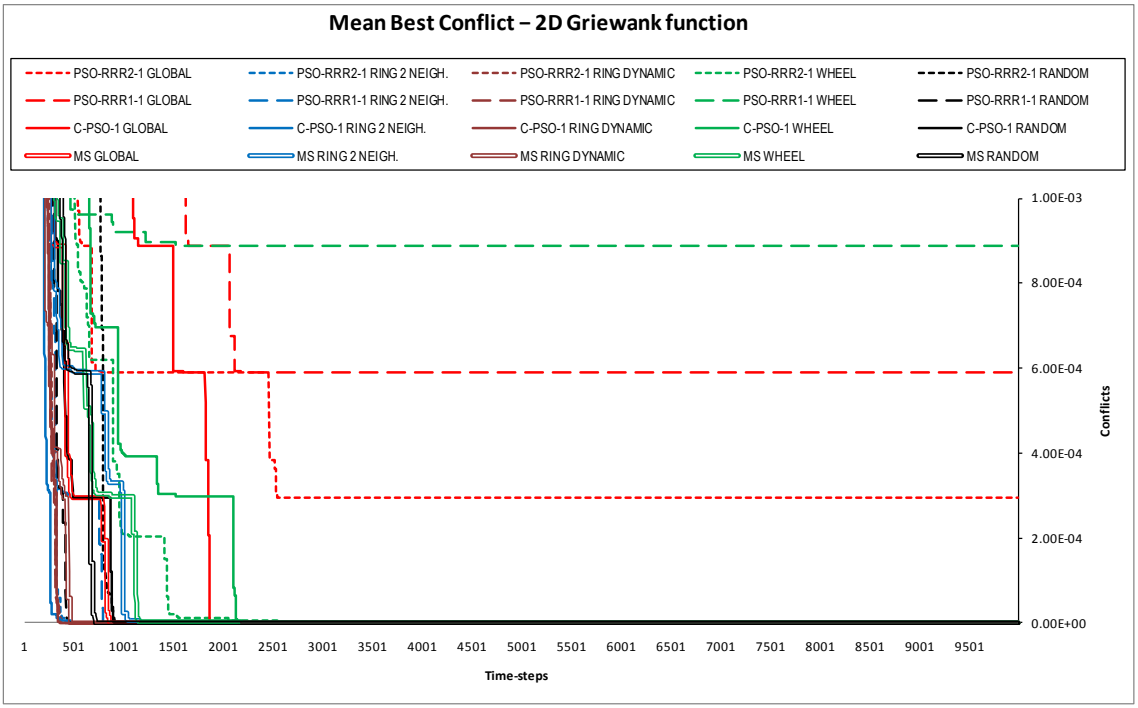


Fig. 7.17. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 7.10. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

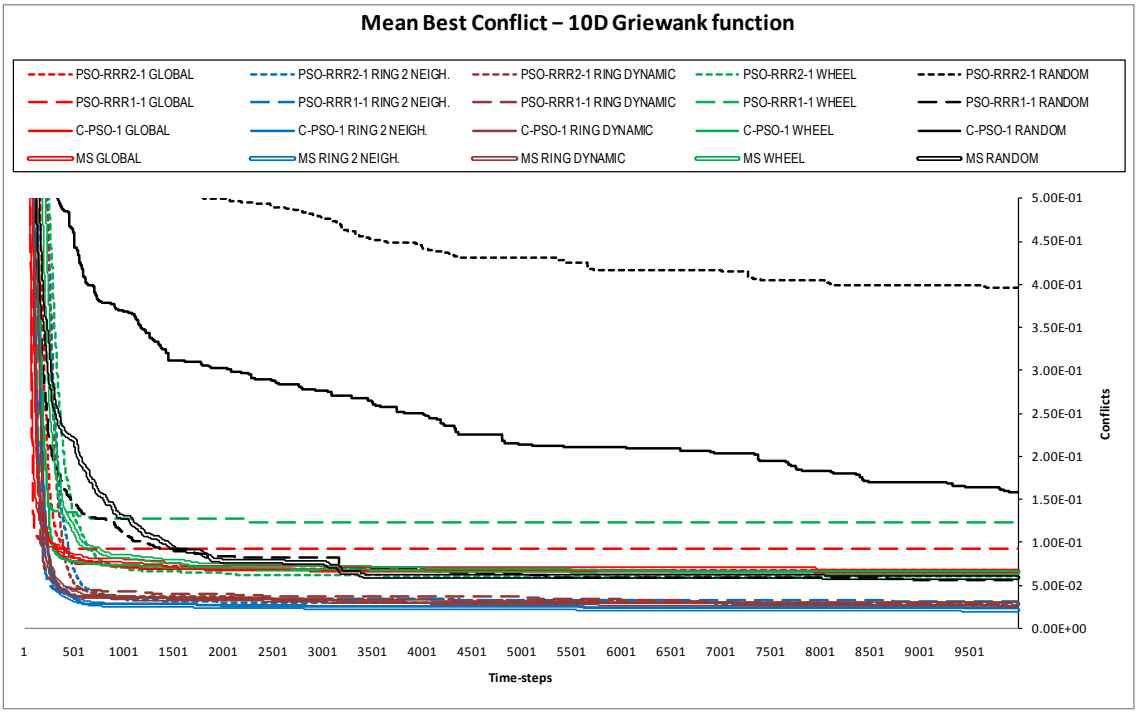


Fig. 7.18. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.11. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

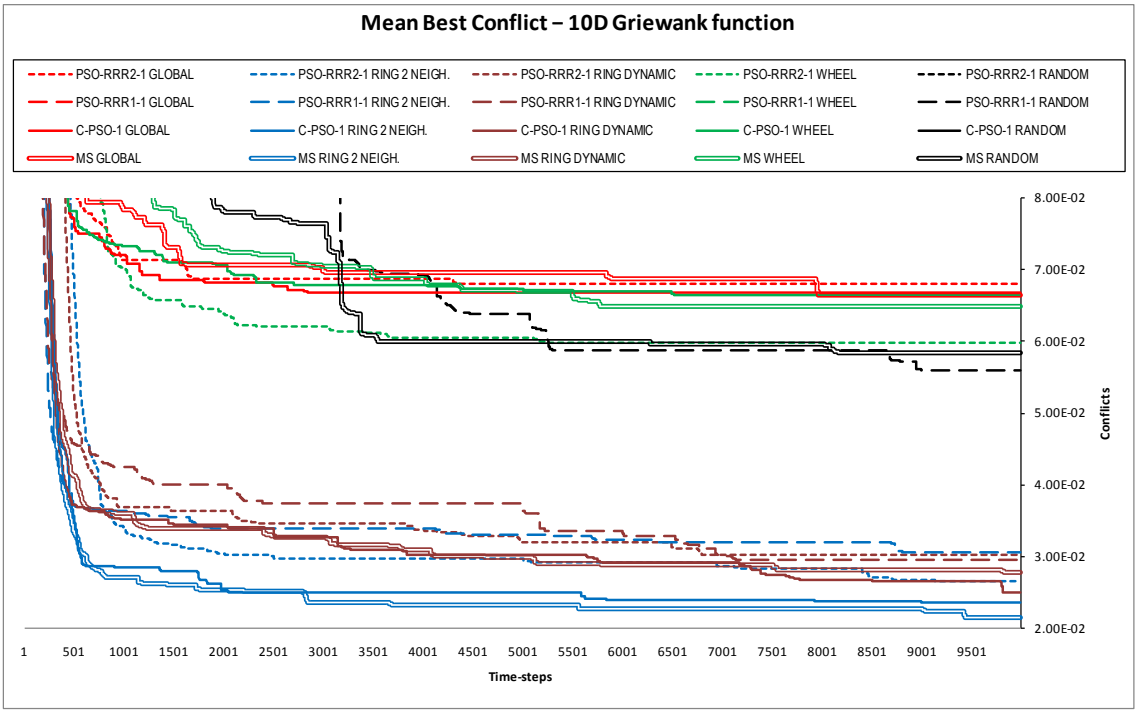


Fig. 7.19. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.11.The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

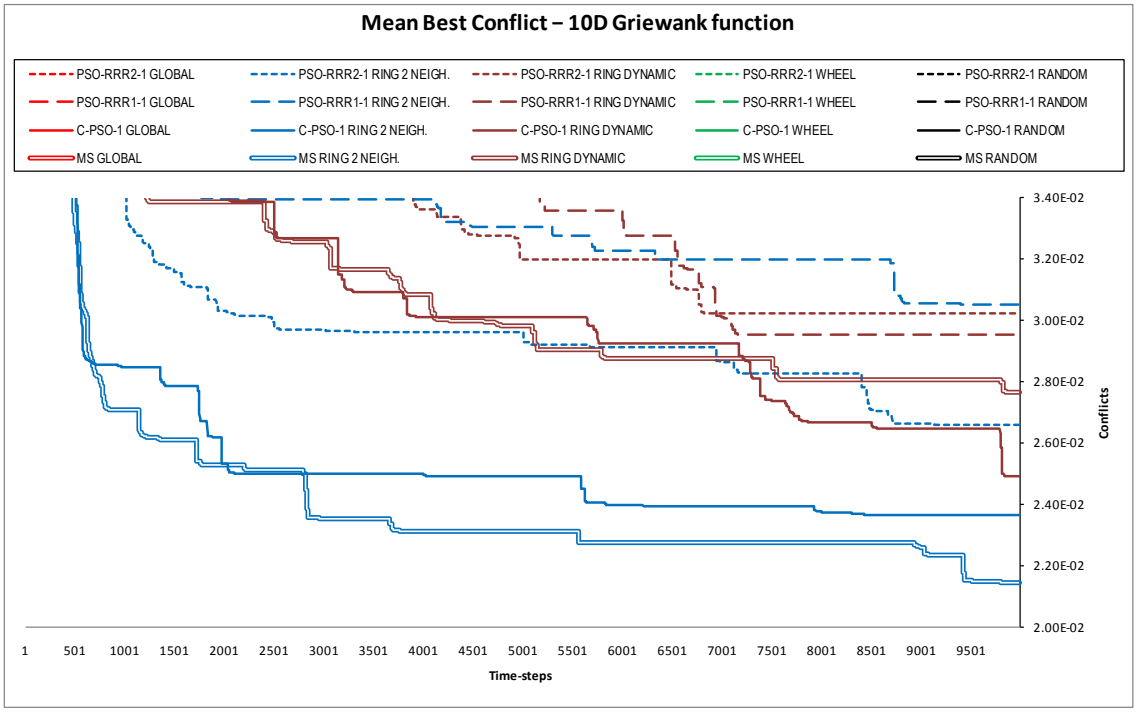


Fig. 7.20. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.11. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

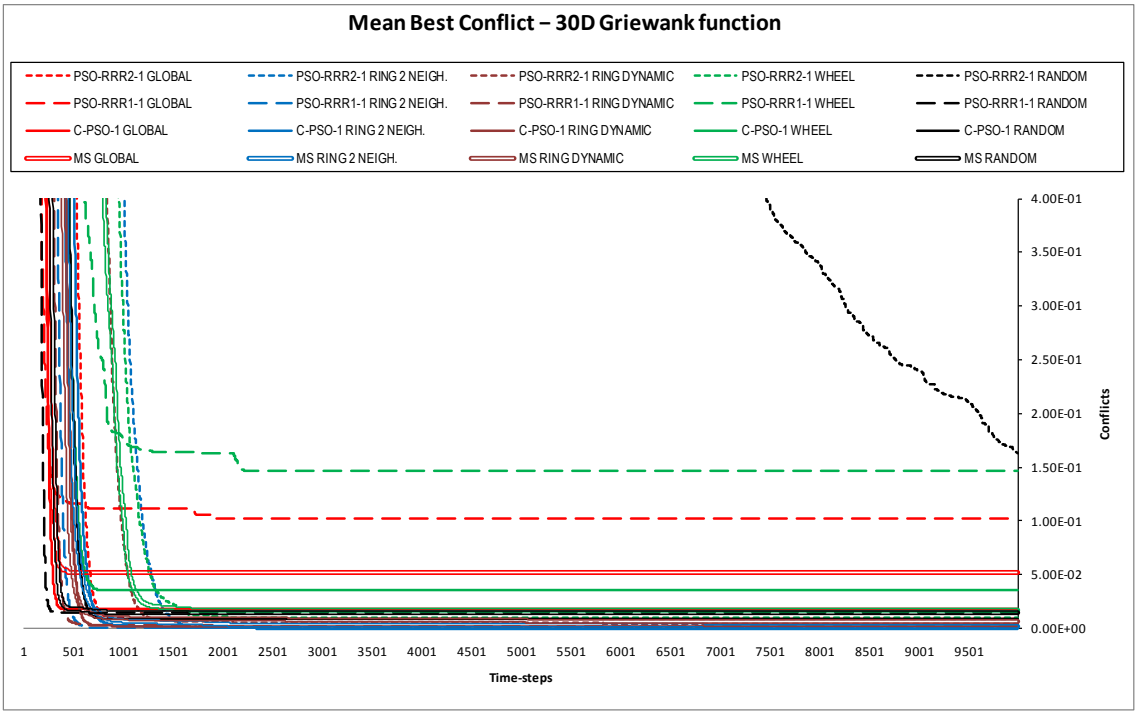


Fig. 7.21. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.12. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

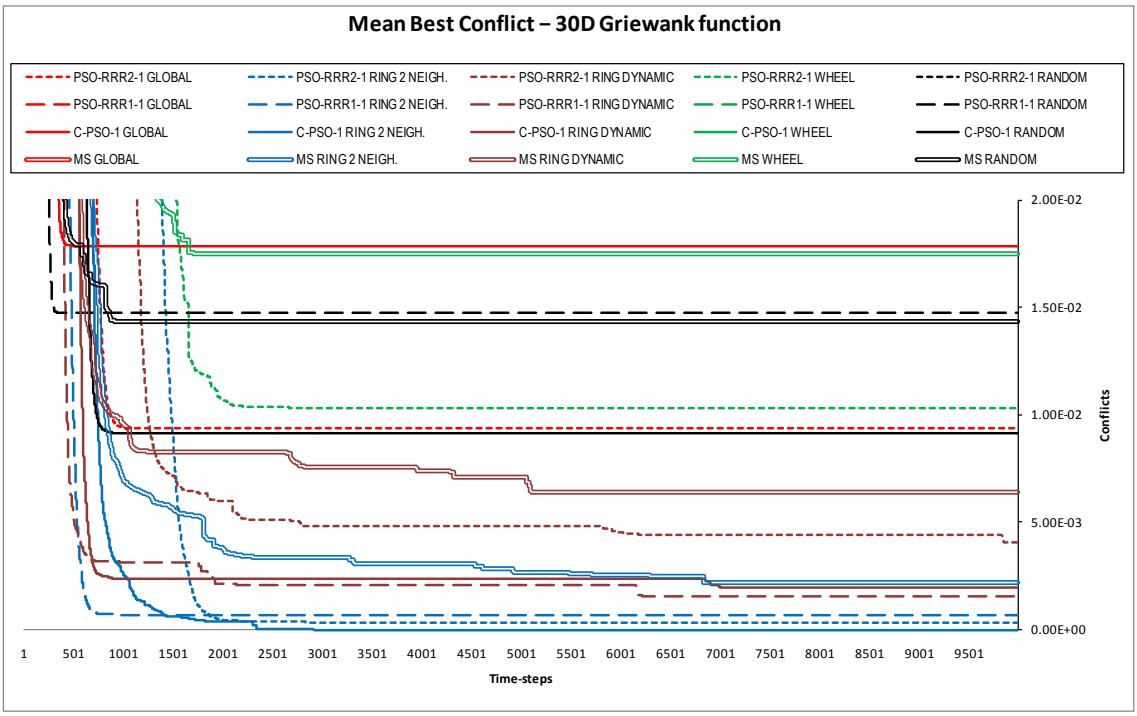


Fig. 7.22. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.12. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.13. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	2.58E-05	96
			1000	0.00E+00	0.00E+00	7.77E-04	9.72E-03	2.79E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.60E-04	100
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	6.82E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.25E-11	100
			1000	0.00E+00	0.00E+00	5.43E-04	9.72E-03	5.57E-03	-
	WHEEL		10000	0.00E+00	0.00E+00	7.77E-04	9.72E-03	7.57E-04	92
			1000	0.00E+00	0.00E+00	3.11E-03	9.72E-03	5.30E-03	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.18E-11	100
			1000	0.00E+00	0.00E+00	2.75E-16	5.88E-15	5.00E-03	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	1.17E-03	9.72E-03	1.13E-04	88
			1000	0.00E+00	0.00E+00	1.17E-03	9.72E-03	1.61E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.29E-03	100
			1000	0.00E+00	0.00E+00	1.96E-03	9.72E-03	7.78E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.26E-11	100
			1000	0.00E+00	0.00E+00	7.03E-07	1.76E-05	5.62E-03	-
	WHEEL		10000	0.00E+00	0.00E+00	2.72E-03	9.72E-03	9.53E-04	72
			1000	0.00E+00	0.00E+00	3.50E-03	9.72E-03	3.96E-03	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.14E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.79E-03	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	1.17E-03	9.72E-03	2.01E-04	88
			1000	0.00E+00	0.00E+00	1.95E-03	9.72E-03	2.42E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	1.38E-03	96
			1000	0.00E+00	0.00E+00	2.07E-03	9.72E-03	7.76E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.23E-11	100
			1000	0.00E+00	0.00E+00	7.58E-06	1.21E-04	6.07E-03	-
	WHEEL		10000	0.00E+00	0.00E+00	7.77E-04	9.72E-03	3.75E-04	92
			1000	0.00E+00	0.00E+00	2.76E-03	9.72E-03	4.62E-03	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.19E-11	100
			1000	0.00E+00	0.00E+00	6.26E-04	9.72E-03	3.26E-03	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	2.33E-03	9.72E-03	1.05E-04	76
			1000	0.00E+00	0.00E+00	3.11E-03	9.72E-03	2.58E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.13E-04	100
			1000	0.00E+00	0.00E+00	7.84E-04	9.72E-03	7.34E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.36E-11	100
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	6.25E-03	-
	WHEEL		10000	0.00E+00	0.00E+00	7.77E-04	9.72E-03	3.08E-04	92
			1000	0.00E+00	0.00E+00	1.24E-03	9.72E-03	4.38E-03	-
	RANDOM		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.13E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.56E-03	-

Table 7.14. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	**9.72E-03**	**9.72E-03**	**1.85E-02**	**3.72E-02**	**5.10E-04**	**0**
			1000	9.72E-03	9.72E-03	2.18E-02	3.72E-02	1.21E-03	-
	RING	nn = 2	**10000**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**1.94E-03**	**0**
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	3.24E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**1.65E-03**	**0**
			1000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	2.13E-03	-
	WHEEL		**10000**	**9.72E-03**	**9.72E-03**	**1.52E-02**	**3.72E-02**	**1.40E-03**	**0**
			1000	9.72E-03	9.72E-03	1.85E-02	3.72E-02	1.94E-03	-
	RANDOM		**10000**	**9.72E-03**	**9.72E-03**	**1.19E-02**	**3.72E-02**	**2.57E-03**	**0**
			1000	3.72E-02	7.82E-02	7.24E-02	2.28E-01	8.20E-03	-
PSO-RRR1-1	GLOBAL		**10000**	**9.72E-03**	**3.72E-02**	**3.45E-02**	**7.82E-02**	**1.29E-04**	**0**
			1000	9.72E-03	3.72E-02	3.45E-02	7.82E-02	1.29E-04	-
	RING	nn = 2	**10000**	**9.72E-03**	**9.72E-03**	**2.18E-02**	**3.72E-02**	**2.43E-03**	**0**
			1000	9.72E-03	3.72E-02	3.17E-02	3.72E-02	3.38E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**1.39E-03**	**0**
			1000	9.72E-03	9.72E-03	1.74E-02	3.72E-02	2.35E-03	-
	WHEEL		**10000**	**3.72E-02**	**1.27E-01**	**1.22E-01**	**2.73E-01**	**1.04E-03**	**0**
			1000	3.72E-02	1.27E-01	1.26E-01	2.73E-01	1.23E-03	-
	RANDOM		**10000**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**1.29E-03**	**0**
			1000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.44E-03	-
C-PSO-1	GLOBAL		**10000**	**9.72E-03**	**9.72E-03**	**1.96E-02**	**3.72E-02**	**3.01E-04**	**0**
			1000	9.72E-03	9.72E-03	2.18E-02	3.72E-02	7.06E-04	-
	RING	nn = 2	**10000**	**9.72E-03**	**9.72E-03**	**1.08E-02**	**3.72E-02**	**2.21E-03**	**0**
			1000	9.72E-03	9.72E-03	1.85E-02	3.72E-02	3.19E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**1.92E-03**	**0**
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	2.11E-03	-
	WHEEL		**10000**	**9.72E-03**	**3.72E-02**	**2.79E-02**	**7.82E-02**	**1.09E-03**	**0**
			1000	9.72E-03	3.72E-02	3.01E-02	7.82E-02	1.36E-03	-
	RANDOM		**10000**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**1.68E-03**	**0**
			1000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.90E-03	-
Multi-Swarm	GLOBAL		**10000**	**9.72E-03**	**3.72E-02**	**2.95E-02**	**3.72E-02**	**3.18E-04**	**0**
			1000	9.72E-03	3.72E-02	3.06E-02	3.72E-02	6.67E-04	-
	RING	nn = 2	**10000**	**9.72E-03**	**9.72E-03**	**1.19E-02**	**3.72E-02**	**2.03E-03**	**0**
			1000	9.72E-03	9.72E-03	1.74E-02	3.72E-02	3.03E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**1.76E-03**	**0**
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	1.95E-03	-
	WHEEL		**10000**	**9.72E-03**	**9.72E-03**	**1.96E-02**	**3.72E-02**	**8.52E-04**	**0**
			1000	9.72E-03	9.72E-03	2.18E-02	3.72E-02	1.17E-03	-
	RANDOM		**10000**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**9.72E-03**	**1.63E-03**	**0**
			1000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.81E-03	-

Table 7.15. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Schaffer f6 function. The neighbourhoods tested are the GLOBAL; the RING with 2 neighbours; the RING with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'); the WHEEL; and a RANDOM topology. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	3.72E-02	7.82E-02	9.22E-02	1.27E-01	3.12E-04	0
			1000	7.82E-02	1.27E-01	1.08E-01	1.78E-01	8.50E-04	-
	RING	nn = 2	10000	3.72E-02	7.82E-02	6.18E-02	7.82E-02	1.77E-03	0
			1000	1.27E-01	1.96E-01	2.01E-01	2.29E-01	4.40E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	3.72E-02	3.72E-02	7.48E-04	0
			1000	7.82E-02	1.27E-01	1.26E-01	2.04E-01	2.76E-03	-
	WHEEL		10000	7.82E-02	1.27E-01	1.24E-01	2.28E-01	7.45E-04	0
			1000	1.27E-01	1.78E-01	1.90E-01	3.46E-01	1.40E-03	-
	RANDOM		10000	2.73E-01	3.96E-01	3.74E-01	4.42E-01	8.30E-03	0
			1000	4.72E-01	4.90E-01	4.88E-01	4.97E-01	1.95E-02	-
PSO-RRR1-1	GLOBAL		10000	3.12E-01	4.30E-01	4.25E-01	4.85E-01	4.76E-05	0
			1000	3.12E-01	4.30E-01	4.26E-01	4.87E-01	2.01E-04	-
	RING	nn = 2	10000	7.82E-02	1.78E-01	1.67E-01	2.73E-01	1.72E-03	0
			1000	1.27E-01	2.28E-01	2.14E-01	3.12E-01	2.38E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	1.27E-01	1.12E-01	2.28E-01	6.68E-04	0
			1000	7.82E-02	1.27E-01	1.49E-01	2.73E-01	1.23E-03	-
	WHEEL		10000	4.52E-01	4.89E-01	4.87E-01	4.96E-01	4.22E-04	0
			1000	4.52E-01	4.89E-01	4.87E-01	4.96E-01	3.34E-04	-
	RANDOM		10000	3.72E-02	3.72E-02	5.63E-02	1.78E-01	6.69E-04	0
			1000	3.72E-02	7.82E-02	8.58E-02	1.78E-01	9.84E-04	-
C-PSO-1	GLOBAL		10000	7.82E-02	1.27E-01	1.31E-01	2.73E-01	1.67E-04	0
			1000	7.82E-02	1.27E-01	1.40E-01	2.73E-01	4.94E-04	-
	RING	nn = 2	10000	3.72E-02	3.72E-02	5.52E-02	7.82E-02	1.59E-03	0
			1000	1.27E-01	1.27E-01	1.52E-01	1.78E-01	3.23E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	3.72E-02	3.72E-02	6.49E-04	0
			1000	7.82E-02	7.82E-02	9.19E-02	1.27E-01	2.00E-03	-
	WHEEL		10000	1.27E-01	2.73E-01	2.63E-01	4.30E-01	8.05E-04	0
			1000	1.78E-01	2.73E-01	2.76E-01	4.30E-01	9.49E-04	-
	RANDOM		10000	3.72E-02	3.72E-02	5.36E-02	7.82E-02	1.32E-03	0
			1000	1.78E-01	2.29E-01	2.54E-01	3.73E-01	4.54E-03	-
Multi-Swarm	GLOBAL		10000	7.82E-02	1.78E-01	1.86E-01	2.73E-01	2.30E-04	0
			1000	1.27E-01	1.78E-01	1.93E-01	2.73E-01	5.26E-04	-
	RING	nn = 2	10000	3.72E-02	7.82E-02	7.45E-02	1.27E-01	1.58E-03	0
			1000	1.27E-01	1.78E-01	1.68E-01	2.28E-01	3.42E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.38E-02	7.82E-02	5.48E-04	0
			1000	3.74E-02	1.27E-01	1.08E-01	1.78E-01	1.89E-03	-
	WHEEL		10000	7.82E-02	2.28E-01	2.13E-01	3.46E-01	6.95E-04	0
			1000	7.82E-02	2.73E-01	2.46E-01	3.46E-01	1.07E-03	-
	RANDOM		10000	3.72E-02	3.72E-02	4.05E-02	7.82E-02	1.12E-03	0
			1000	7.82E-02	1.27E-01	1.10E-01	1.78E-01	2.18E-03	-

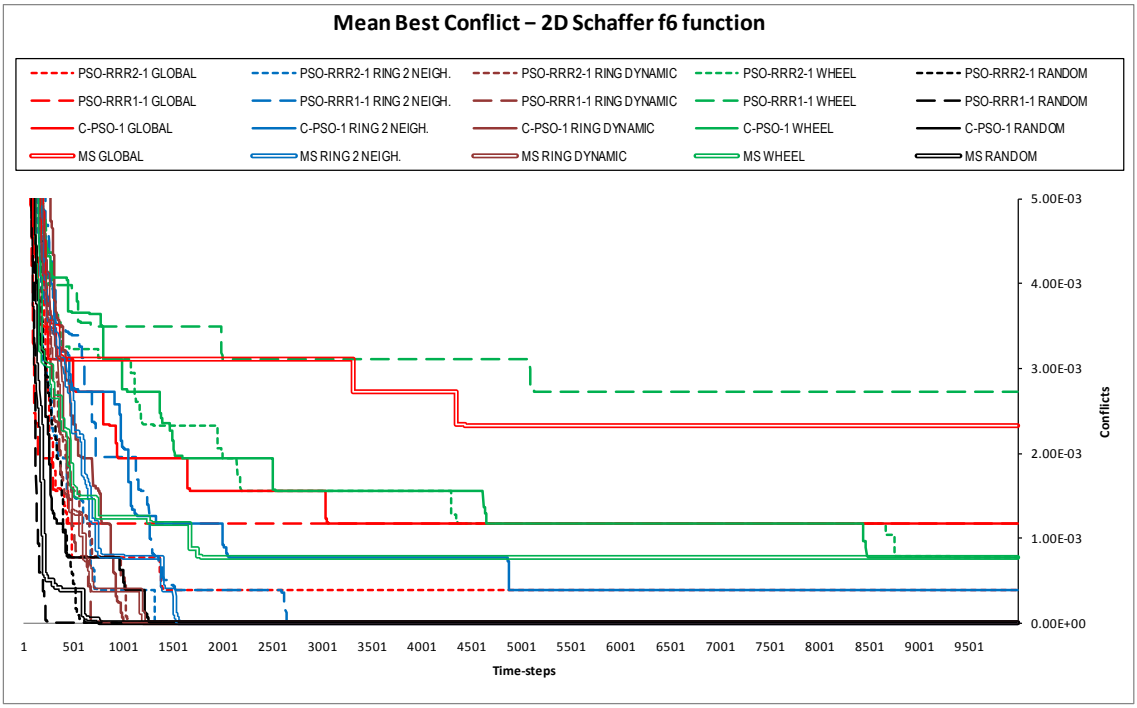


Fig. 7.23. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 7.13. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

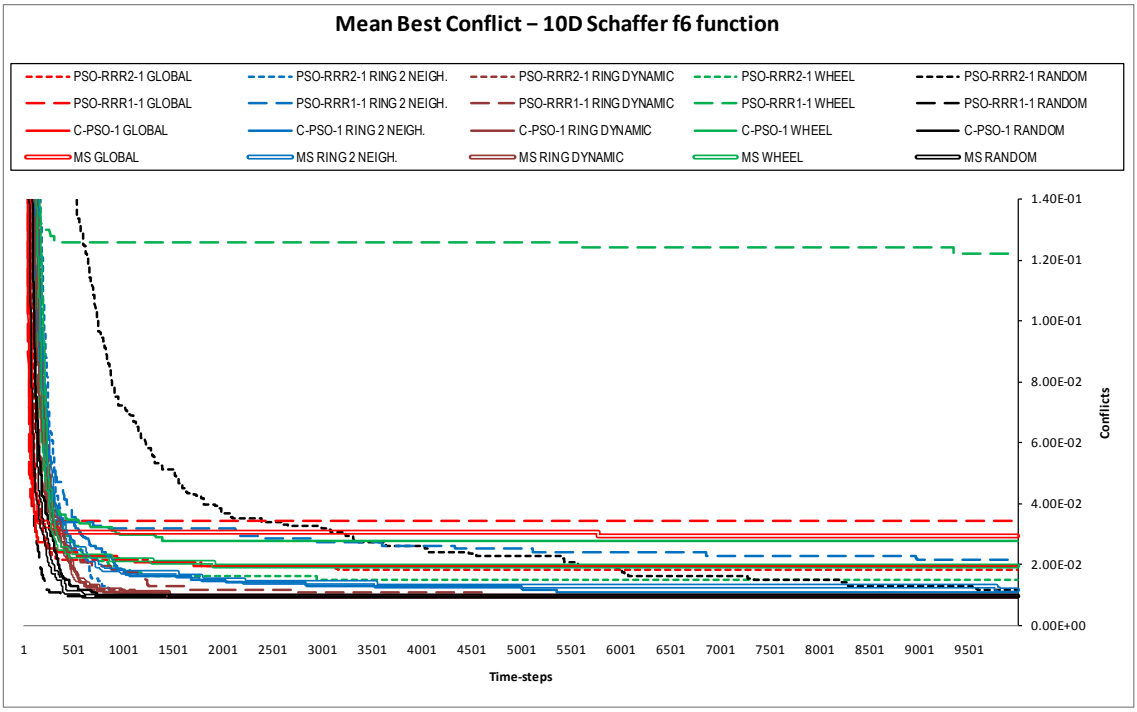


Fig. 7.24. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.14. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

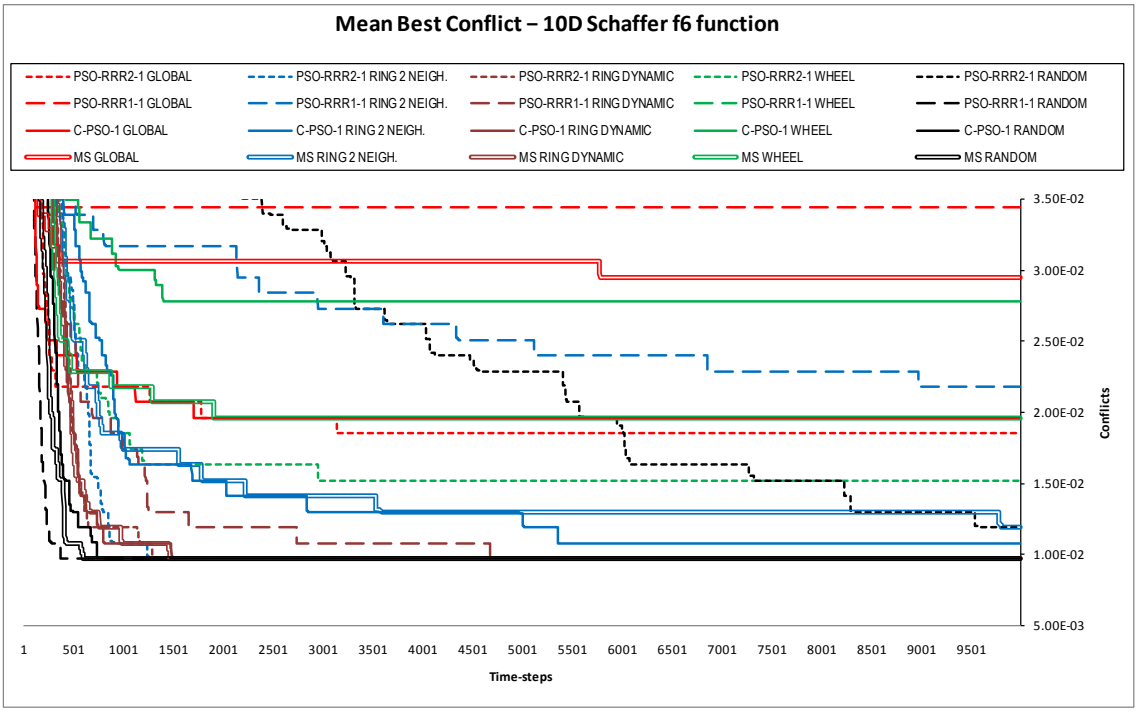


Fig. 7.25. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.14. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

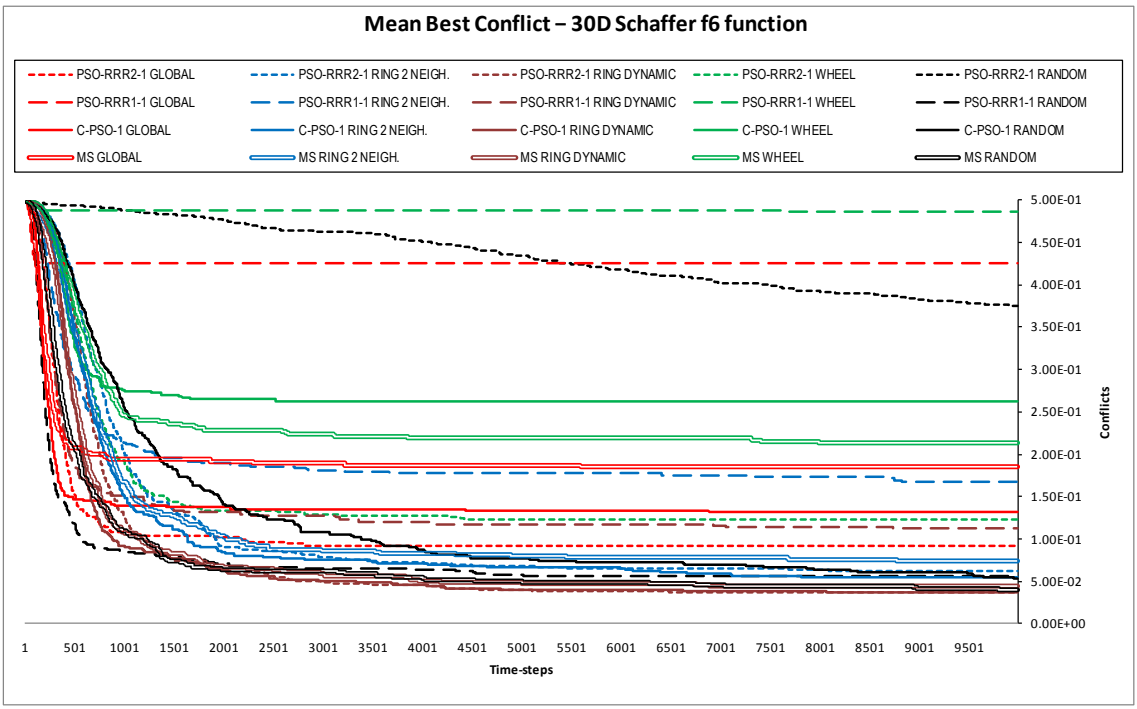


Fig. 7.26. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.15. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

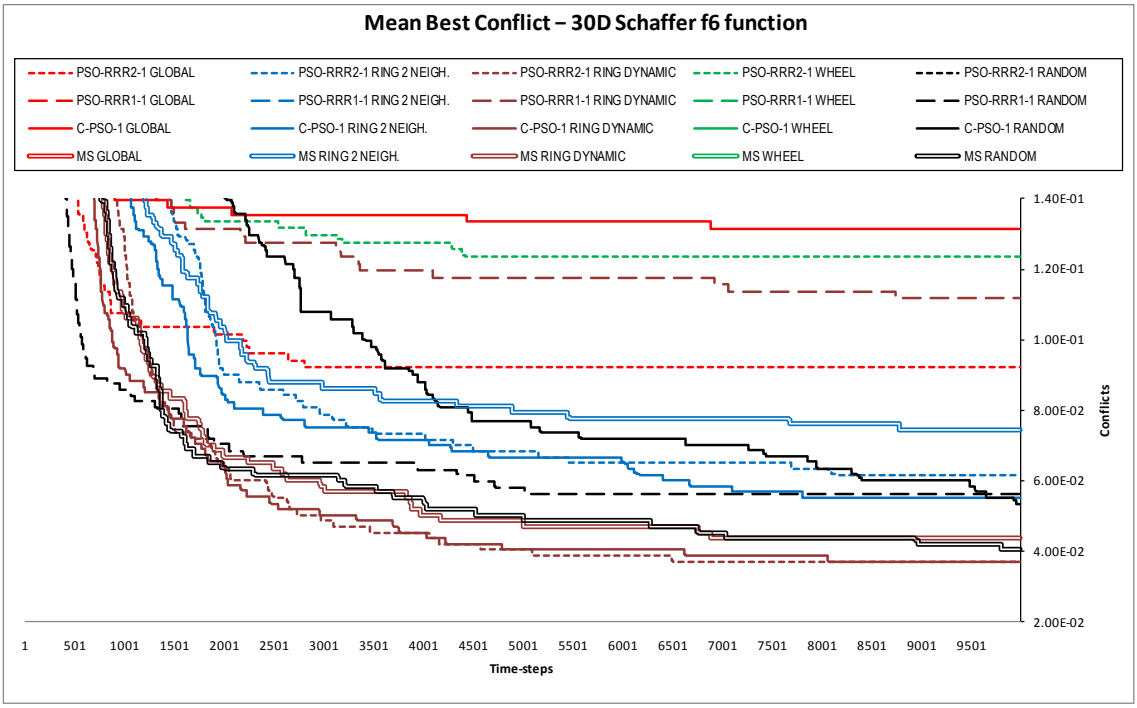


Fig. 7.27. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.15. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Discussion

The settings of the experiments are the same as those described in the previous chapter (see section 6.3.2.2.) unless specifically stated otherwise.

Sphere

In the 2D problem, every combination of coefficients' settings and neighbourhood topology finds the exact solution for every run. It can also be observed that the implosion of the particles is virtually complete in every case by the end of the search, while the PSO-RRR1 approach exhibits the highest degree of clustering by the 1,000th time-step regardless of the neighbourhood topology (see pb_me in Table 7.1).

In the 10D problem, again every algorithm achieves a 100% success rate (SR) by the end of the search (10,000th time-step). It can also be observed that the 'PSO-RRR2-1 Random' is the only one whose mean solution does not meet the success criterion (i.e. error below 10^{-4}) by the 1,000th time-step. It is also the one with the lowest degree of clustering by the end of the search. The values of pb_me in Table 7.2 show that the

PSO-RRR1-1 reaches the highest degree of clustering, the PSO-RRR2-1 the lowest, and the C-PSO-1 and the Multi-Swarm (MS) are in between.

In the 30D problem, the 'PSO-RRR2-1 Random' and the 'PSO-RRR1-1 Global' are the only ones which do not achieve a SR of 100% by the end of the search, but for two very different reasons: the former because convergence is too slow, and the latter because of premature convergence (Fig. 7.3). The proposed dynamic neighbourhood appears successful: the median and mean solutions found are either between those obtained by the global and the ring (*nn*=2) topologies (PSO-RRR2-1, C-PSO-1), or they are better than both (PSO-RRR1-1, MS). Making the neighbourhood dynamic avoids the premature convergence observed in the 'PSO-RRR1-1 Global', while achieving better solutions and higher degree of clustering than the 'PSO-RRR1-1 Ring *nn*=2' (see Table 7.3).

Rosenbrock

In the 2D problem, every algorithm finds the exact solution for every run. It can also be observed that the implosion of the particles is virtually complete in every case by the end of the search, while the PSO-RRR1 approach presents the highest degree of clustering by the 1,000th time-step (refer to pb_me in Table 7.4).

In the 10D problem, the success rates (SRs) of most algorithms fall dramatically. In the same fashion as when optimizing the 30D Sphere, the 'PSO-RRR2-1 Random' and the 'PSO-RRR1-1 Global' obtained the worst results, the former due to slow convergence whereas the latter due to premature convergence (Fig. 7.6). Although the 'PSO-RRR1-1 Global' achieved a SR of 44%, it converged to a local optimum (3.99) 36% of the times (refer to digital appendix). The best performance is exhibited by the 'PSO-RRR1-1 Ring *nn*=2', which is the only algorithm that achieved 100% success. As to the proposed dynamic topology, results are very promising. For the PSO-RRR2-1, the SR and the median solution are between those of the global and of the ring (*nn*=2) topologies (closer to the better one), as expected, while the mean solution is better than both (never falling into the local optimum). For the PSO-RRR1-1, the mean solution and the SR are between those of the global and the ring (*nn*=2) topologies, while the median solution is better than both. The SR is 96%, falling into a local optimum only once in 25 runs. For the C-PSO-1, the median, the mean, and the SR are remarkably better than those obtained by the global and the ring (*nn*=2) topologies, achieving a SR of 96% and never

falling into a local optimum (both the global and ring (*nn*=2) do). The same is true for the MS. As shown in Fig. 7.6 and Fig. 7.7, the best mean solutions are exhibited by the 'Ring Dynamic' and the 'Ring *nn*=2' topologies (brown and blue curves, respectively).

In the 30D problem, achieving success becomes notably harder. The same as before, the 'PSO-RRR2-1 Random' exhibits extremely slow convergence (never achieved) while the 'PSO-RRR1-1 Global' shows premature convergence (refer to Fig. 7.8). It is interesting to observe that, after 10,000 time-steps, only 4 algorithms reach convergence (see Fig. 7.8 to Fig. 7.10): 'the PSO-RRR1-1 Global', which shows premature convergence, the 'PSO-RRR1-1 Ring Dynamic', the 'PSO-RRR1-1 Random', and the 'C-PSO-1 Global'. The latter three obtain the best results. The 'PSO-RRR2-1 Ring Dynamic' shows marginally worse results than its global and ring (*nn*=2) counterparts (none of which converges). The 'PSO-RRR1-1 Ring Dynamic' converges and shows remarkably better performance than its global and ring (*nn*=2) counterparts. The 'C-PSO-1 Ring Dynamic' obtains a median solution worse than, and a mean solution in between, those of its global and ring (*nn*=2) counterparts. Note that this algorithm is still far from converging (brown solid line in Fig. 7.9 and Fig. 7.10). Finally, the 'MS Ring Dynamic' finds results in between those of its global and ring (*nn*=2) counterparts.

Rastrigin

In the 2D problem, every algorithm finds the exact solution for every run by the end of the search. In fact, they all do by the 1,000th time-step already, except for the 'MS Ring *nn*=2' (which also shows a remarkable lower degree of clustering). It is not clear why convergence is so delayed in this case (see Table 7.7).

In the 10D problem, the 'Ring Dynamic' topologies result in a remarkable increase in the success rate (SR) when compared to their global and ring (*nn*=2) counterparts for every coefficients' settings (see Table 7.8). Since this is a highly multimodal function, several cases of (early) stagnation can be observed in Fig. 7.12 and Fig. 7.13. The best performance overall is exhibited by the 'PSO-RRR2-1 Ring Dynamic', while the 'C-PSO-1 Ring Dynamic' and the 'MS Ring Dynamic' also show very good performance.

In the 30D problem, no algorithm is able to meet the success criterion in any run. The best performance is exhibited by the 'C-PSO-1 Random' (see Table 7.9) although con-

vergence is very slow (see Fig. 7.14 to Fig. 7.16). The 'PSO-RRR2-1 Wheel' and 'MS Random' also find very good solutions, while exhibiting faster convergence and a much earlier stagnation. Notice that all the global topologies, the 'PSO-RRR1-1 Wheel', the 'PSO-RRR1-1 Random', and the 'MS Random', show a complete loss of diversity (see the values of pb_me in Table 7.9) and stagnation (see convergence curves in Fig. 7.14 to Fig. 7.16). All the 'Ring *nn*=2' and the 'Ring Dynamic' topologies still present some diversity by the end of the search, so that improvement is to be expected for an extended search-length. Nonetheless, the 'PSO-RRR2-1 Ring Dynamic' and the 'MS Ring Dynamic' exhibit some of the best performances. For the 'PSO-RRR1-1' and the 'C-PSO-1', the 'Ring Dynamic' topology finds solutions in between their global and ring (*nn*=2) counterparts, while the 'MS Ring Dynamic' exhibits better performance than both. Instead, the 'PSO-RRR2-1 Ring Dynamic' shows similar performance to that of the 'PSO-RRR2-1 Ring *nn*=2', which, surprisingly, are worse than that of the global topology (in a highly multimodal problem!). The only conclusion is that the combination of PSO-RRR2-1 with local neighbours for a high-dimensional and highly multimodal problem might just result in too slow a convergence for a search this long.

Griewank

In the 2D problem, 3 algorithms exhibit premature convergence: the 'PSO-RRR2-1 Global' (in 1 out of 25 runs); the 'PSO-RRR1-1 Global' (in 2 out of 25 runs); and the 'PSO-RRR1-1 Wheel' (in 3 out of 25 runs). In the 2 global cases, making them dynamic resolves the problem, while at the end of the search they end up showing similar degrees of clustering to those of their global counterparts, and much higher degrees of clustering than those of their ring (*nn*=2) counterparts (see pb_me in Table 7.10). The other algorithms achieve a 100% success rate (SR).

In the 10D problem, the SRs decrease dramatically. By a large margin, the best performances are exhibited by all the 'Ring *nn*=2' and the 'Ring Dynamic' topologies, as can be clearly seen in Table 7.11, and in Fig. 7.18 to Fig. 7.20. For the PSO-RRR1-1, the 'Ring Dynamic' topology obtains better results than both its global and ring (*nn*=2) counterparts, while in the other cases results are in between, as it would be expected. The best performance overall is exhibited by the 'MS Ring *nn*=2', followed by the 'C-PSO-1 Ring *nn*=2', and the 'C-PSO-1 Ring Dynamic'.

In the 30D problem, the SRs increase, as the difficulty of this particular problem decreases with dimensionality. By a large margin again, the best performances are exhibited by all the 'Ring *nn*=2' and by the 'Ring Dynamic' topologies, as can be clearly seen in Table 7.12, and in Fig. 7.21 and Fig. 7.22. The performances of the 'Ring Dynamic' neighbourhoods fall between those of their global and ring (*nn*=2) counterparts, as expected, close to the better one (i.e. the ring topology). The best performance overall is exhibited by the 'C-PSO-1 Ring *nn*=2', followed by the 'PSO-RRR2-1 Ring *nn*=2'.

Schaffer f6

In the 2D problem, 9 algorithms, namely all the global and wheel topologies plus the 'C-PSO-1 Ring *nn*=2', present some few failures to achieve the success criterion. The remaining 11 algorithms find the solution in every run (see Table 7.13 and Fig. 7.23).

In the 10D problem, no algorithm is able to find the solution in any run (see Table 7.14). The best performances are exhibited by all the 'Ring Dynamic' neighbourhoods, the 'PSO-RRR2-1 Ring *nn*=2', the 'PSO-RRR1-1 Random', the 'C-PSO-1 Random', and the 'MS Random' (see Table 7.14, and Fig. 7.24 and Fig. 7.25).

In the 30D problem, no algorithm is able to find the solution in any run (see Table 7.15). The best performances are exhibited by the 'PSO-RRR2-1 Ring Dynamic' and the 'C-PSO-1 Ring Dynamic', followed by the 'MS Random' and the 'MS Ring Dynamic' (refer to Table 7.15, Fig. 7.26, and Fig. 7.27).

Overall analysis

The 'Ring Dynamic' topology proposed is successful and appears desirable, as the robustness gained by the reduced number of neighbours at the early stages of the search does not seem to affect the fine-grain search at the end. The performance exhibited is most of the time either between that of its global and ring (*nn*=2) counterparts –and closer to the better one– or better than both. Only a few times it happens to be worse than both, and only marginally. The global, wheel and random topologies are able to find very good results in some isolated problems, but the overall performance is inferior and remarkably less robust. The ring topology with 2 neighbours is more stable, but its performance is also less robust than that of the proposed dynamic ring topology.

7.3. Forward topology

A so-called ‘forward topology’ is proposed, which shares some important characteristics with the ring topology. Namely, it allows any number of neighbours, from ‘0’ to ‘swarm-size – 1’, and guarantees full overlapping so that the information acquired by any particle is guaranteed to eventually spread throughout the whole swarm, given enough time. In both cases the overlapping is given by ‘neighbourhood-size – 1’ particles. The only difference is in that the interconnections are not bidirectional, so that –in general– a particle is not informed by the same particles it informs. Experiments are carried out for the same four coefficients’ settings and on the same benchmark suite as previously performed for the ring topology to allow numerical comparisons.

A graphical comparison between the ‘Ring topology’ and the ‘Forward topology’ is offered in Fig. 7.28. Notice that the number of particles that a given particle needs to go through to access the information gathered by the topologically farthest particle are the same. For instance, the farthest particle whose information to access for particle ‘1’ in Fig. 7.28 is particle ‘4’ for the ring topology (going through particles ‘2’ and ‘3’) and particle ‘6’ for the forward topology (going through particles ‘3’ and ‘5’). The reverse is only true for the ring topology: the farthest particle whose information to access for particle ‘4’ is particle ‘1’ for the ring topology, whereas the one for particle ‘6’ in the forward topology is not particle ‘1’ –which is actually the closest– but particle ‘5’.

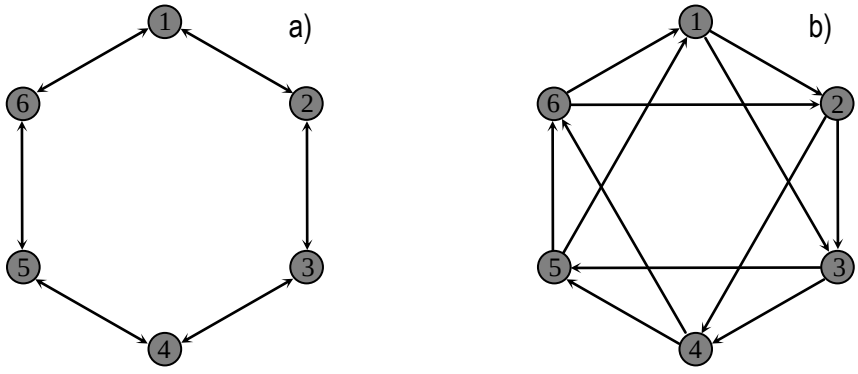


Fig. 7.28. a) Ring topology with neighbourhood-size equal to three; b) Forward topology with neighbourhood-size equal to three. The farthest particle for particle ‘1’ to extract information from is particle ‘4’ for the ring topology (going through particles ‘2’ and ‘3’) and particle ‘6’ for the forward topology (going through particles ‘3’ and ‘5’).

Table 7.16. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Sphere function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.74E-57	3.42E-54	4.65E-53	5.96E-52	4.79E-18	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.61E-53	8.64E-49	7.07E-46	1.26E-44	1.55E-18	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.76E-54	1.58E-50	1.80E-48	2.05E-47	5.87E-19	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	5.38E-49	6.51E-46	3.70E-44	5.22E-43	4.19E-19	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.51E-51	3.68E-49	1.90E-47	2.31E-46	1.93E-19	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	5.19E-88	2.30E-85	3.33E-84	4.17E-83	3.43E-37	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.09E-82	2.99E-80	2.13E-78	2.27E-77	1.52E-37	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.65E-84	8.87E-82	6.58E-81	8.62E-80	3.99E-38	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	8.31E-79	6.76E-77	2.38E-76	2.21E-75	2.83E-37	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.18E-83	1.16E-80	8.47E-80	1.04E-78	4.13E-38	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.74E-91	5.15E-88	2.06E-84	5.14E-83	9.16E-30	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	5.58E-82	3.44E-78	5.67E-76	6.45E-75	3.58E-32	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	7.89E-85	1.89E-81	2.99E-79	4.49E-78	4.59E-33	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	6.17E-76	7.64E-73	2.57E-69	6.40E-68	1.66E-31	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.45E-80	8.77E-78	1.06E-76	1.18E-75	4.89E-31	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.38E-90	4.67E-86	1.06E-83	2.16E-82	1.70E-20	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.39E-85	4.26E-78	9.85E-77	1.83E-75	2.68E-20	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.99E-86	1.12E-80	2.13E-79	2.33E-78	4.47E-20	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.24E-75	8.38E-73	4.72E-69	1.16E-67	9.31E-20	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.17E-79	3.23E-74	3.17E-70	7.36E-69	8.94E-22	-

Table 7.17. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Sphere function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	2.04E-256	3.90E-250	3.93E-247	8.75E-246	2.28E-126	100
			1000	4.97E-24	5.49E-23	1.25E-22	1.06E-21	8.08E-14	-
	RING	nn = 2	10000	1.29E-145	2.82E-143	3.37E-141	5.90E-140	1.13E-73	100
			1000	1.78E-13	3.67E-12	5.33E-12	2.02E-11	8.77E-09	-
		nni = 2 nnf = (m – 1)	10000	1.46E-226	5.19E-223	5.65E-221	6.34E-220	1.32E-113	100
			1000	2.26E-16	7.12E-15	1.30E-14	6.38E-14	2.46E-10	-
	FWD	nn = 2	10000	1.34E-134	1.93E-133	2.03E-132	3.80E-131	1.73E-69	100
			1000	6.90E-12	1.96E-11	2.90E-11	1.08E-10	1.00E-08	-
		nni = 2 nnf = (m – 1)	10000	3.44E-221	1.41E-217	1.92E-211	4.79E-210	1.61E-109	100
			1000	5.91E-15	4.03E-14	4.15E-14	1.14E-13	3.98E-10	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.75E-67	1.61E-65	5.10E-65	3.50E-64	5.98E-35	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.86E-35	2.94E-33	4.46E-33	1.53E-32	1.01E-19	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.00E-44	7.30E-43	2.52E-42	1.96E-41	1.93E-24	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.03E-35	3.52E-34	6.82E-34	5.66E-33	2.49E-20	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.68E-44	2.91E-43	4.91E-43	2.18E-42	9.41E-25	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.30E-51	6.84E-50	3.49E-49	5.15E-48	7.98E-27	-
	RING	nn = 2	10000	1.13E-280	2.23E-277	4.17E-274	5.97E-273	2.11E-140	100
			1000	7.36E-27	3.12E-25	4.91E-25	2.77E-24	1.50E-15	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	7.32E-33	1.59E-31	3.09E-31	2.69E-30	1.18E-18	-
	FWD	nn = 2	10000	1.49E-264	1.82E-260	1.53E-259	7.23E-259	4.92E-133	100
			1000	1.13E-24	8.57E-24	1.71E-23	1.77E-22	7.11E-15	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.53E-31	3.31E-30	5.88E-30	3.32E-29	4.34E-18	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.21E-58	2.00E-56	2.03E-55	1.42E-54	2.17E-18	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.20E-153	100
			1000	1.41E-32	1.17E-30	1.06E-29	9.91E-29	2.68E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.18E-43	2.00E-39	5.49E-38	1.09E-36	1.35E-14	-
	FWD	nn = 2	10000	6.03E-186	1.07E-182	4.73E-180	1.07E-178	2.39E-93	100
			1000	4.92E-18	6.77E-17	1.20E-16	5.25E-16	7.03E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.97E-23	1.46E-22	3.15E-22	1.75E-21	1.37E-13	-

Table 7.18. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Sphere function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.22E-87	3.29E-84	3.07E-82	6.31E-81	2.12E-45	100
			1000	3.49E-06	1.85E-05	4.08E-05	2.77E-04	1.84E-06	-
	RING	nn = 2	10000	3.77E-43	1.90E-42	7.86E-42	6.78E-41	9.24E-25	100
			1000	1.23E-01	2.84E-01	3.12E-01	6.69E-01	2.14E-04	-
		nni = 2 nnf = (m – 1)	10000	2.21E-74	8.42E-73	6.49E-72	6.44E-71	3.87E-40	100
			1000	9.91E-03	2.85E-02	2.98E-02	7.82E-02	4.65E-05	-
	FWD	nn = 2	10000	3.87E-42	4.68E-41	6.23E-41	1.97E-40	2.41E-24	100
			1000	1.45E-01	3.27E-01	3.22E-01	4.72E-01	1.60E-04	-
		nni = 2 nnf = (m – 1)	10000	7.23E-73	7.88E-71	4.68E-70	3.46E-69	3.16E-39	100
			1000	1.05E-02	2.69E-02	3.38E-02	8.56E-02	5.21E-05	-
PSO-RRR1-1	GLOBAL		10000	4.06E-07	3.79E-04	9.89E-02	2.39E+00	4.69E-17	32
			1000	5.57E-06	1.64E-03	2.70E-01	4.03E+00	5.93E-11	-
	RING	nn = 2	10000	8.13E-144	7.26E-142	5.86E-141	6.69E-140	1.20E-74	100
			1000	1.73E-11	7.83E-11	8.50E-11	2.40E-10	2.05E-09	-
		nni = 2 nnf = (m – 1)	10000	2.04E-268	4.97E-257	6.23E-249	8.26E-248	1.16E-134	100
			1000	5.25E-17	2.48E-16	3.07E-16	1.05E-15	3.12E-12	-
	FWD	nn = 2	10000	3.36E-160	3.85E-158	2.90E-157	4.10E-156	8.61E-83	100
			1000	3.15E-13	1.10E-12	1.59E-12	1.06E-11	2.49E-10	-
		nni = 2 nnf = (m – 1)	10000	4.62E-266	1.69E-250	2.13E-239	5.33E-238	1.05E-127	100
			1000	9.21E-19	6.91E-18	9.95E-18	3.43E-17	6.01E-13	-
C-PSO-1	GLOBAL		10000	3.05E-220	2.21E-212	1.42E-207	3.49E-206	1.57E-108	100
			1000	8.53E-20	1.09E-17	1.04E-16	9.10E-16	1.33E-12	-
	RING	nn = 2	10000	5.68E-96	1.67E-94	1.65E-93	3.57E-92	7.60E-51	100
			1000	7.53E-07	3.07E-06	3.62E-06	1.35E-05	5.46E-07	-
		nni = 2 nnf = (m – 1)	10000	3.06E-182	6.62E-179	1.98E-177	3.84E-176	3.81E-93	100
			1000	7.02E-10	4.53E-09	4.77E-09	1.48E-08	1.77E-08	-
	FWD	nn = 2	10000	6.49E-95	2.99E-94	9.02E-94	9.82E-93	6.28E-51	100
			1000	8.78E-07	2.88E-06	2.91E-06	5.14E-06	4.55E-07	-
		nni = 2 nnf = (m – 1)	10000	3.56E-177	7.09E-174	3.38E-171	3.38E-171	1.74E-90	100
			1000	1.89E-09	5.93E-09	6.95E-09	1.46E-08	2.20E-08	-
Multi-Swarm	GLOBAL		10000	1.10E-181	4.68E-172	1.95E-166	4.21E-165	7.90E-88	100
			1000	4.53E-17	1.52E-14	4.02E-11	1.00E-09	3.63E-10	-
	RING	nn = 2	10000	3.13E-113	7.02E-109	2.72E-107	5.96E-106	6.93E-57	100
			1000	2.83E-08	9.73E-08	2.55E-07	3.82E-06	1.75E-06	-
		nni = 2 nnf = (m – 1)	10000	1.28E-185	6.46E-180	6.80E-173	1.68E-171	2.58E-91	100
			1000	7.07E-12	1.43E-10	3.51E-10	3.97E-09	1.57E-08	-
	FWD	nn = 2	10000	9.63E-63	2.09E-61	1.04E-60	6.46E-60	3.03E-34	100
			1000	4.04E-04	1.74E-03	1.79E-03	3.91E-03	1.73E-05	-
		nni = 2 nnf = (m – 1)	10000	1.16E-174	4.86E-169	2.11E-165	2.09E-164	3.76E-87	100
			1000	1.09E-06	4.95E-06	5.29E-06	1.20E-05	7.73E-07	-

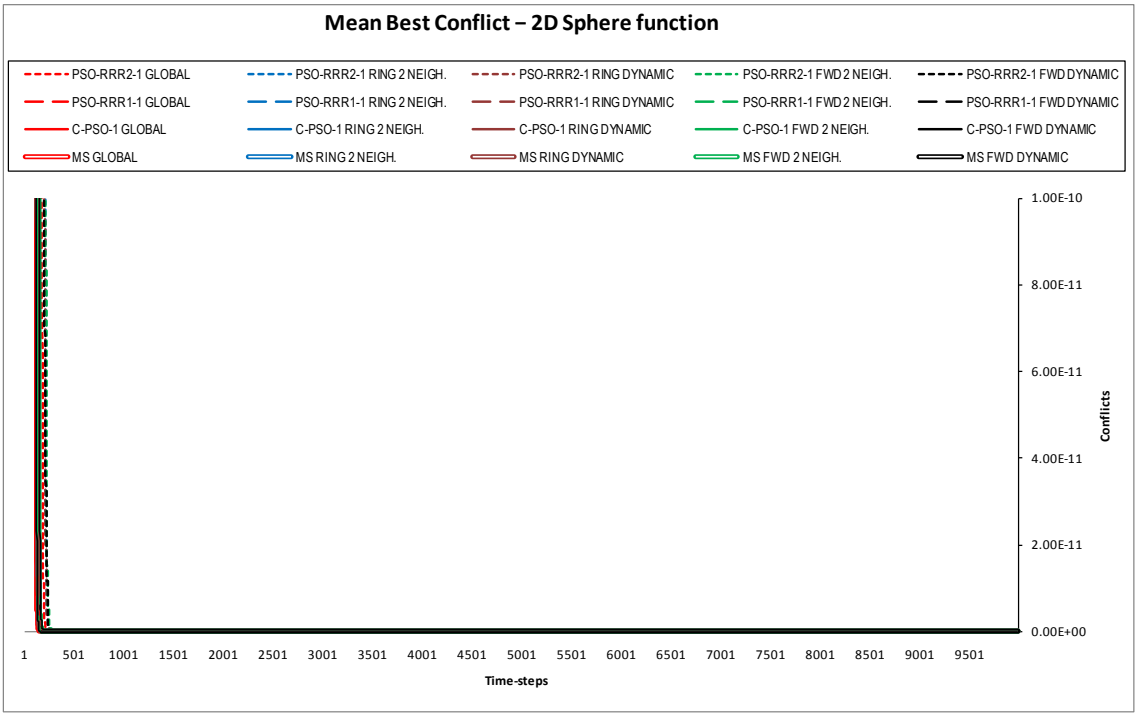


Fig. 7.29. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 7.16. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

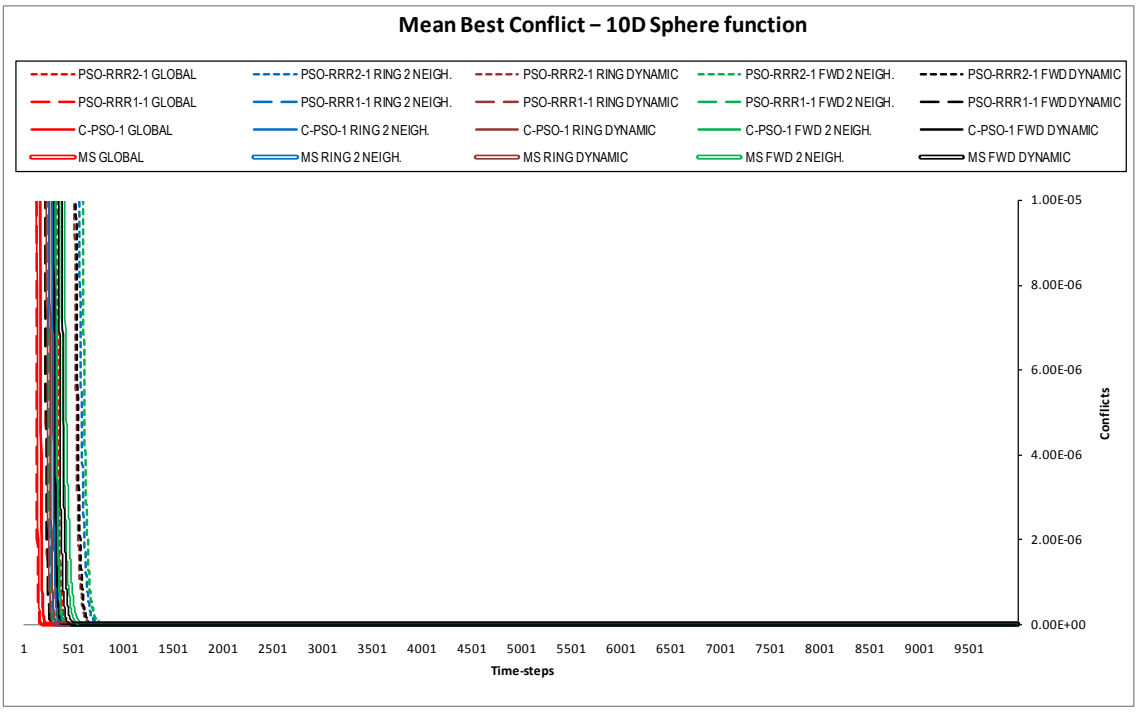


Fig. 7.30. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 7.17. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

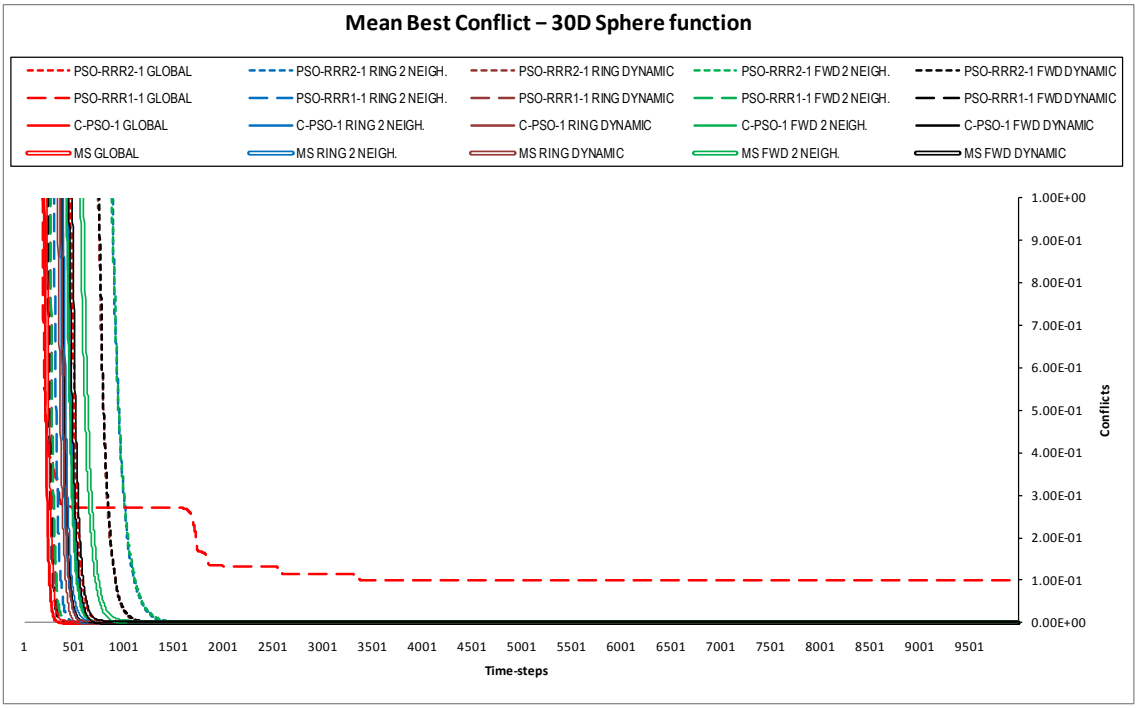


Fig. 7.31. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.18. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

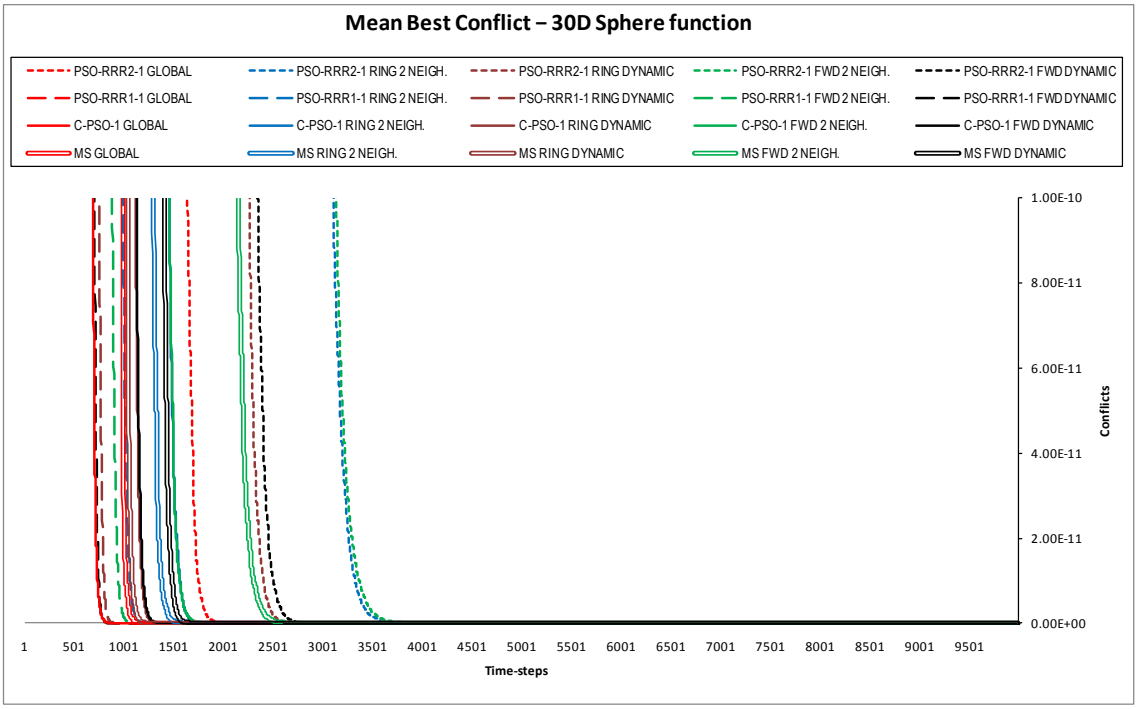


Fig. 7.32. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.18. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.19. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rosenbrock function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.54E-30	3.01E-28	4.82E-26	3.96E-25	3.72E-08	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.11E-20	3.03E-16	3.31E-15	3.09E-14	6.95E-06	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.71E-23	1.08E-18	2.30E-17	4.07E-16	4.46E-06	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.32E-15	9.58E-14	3.19E-12	5.72E-11	3.09E-04	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.20E-20	3.93E-17	3.75E-16	4.30E-15	7.13E-07	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.70E-20	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.93E-32	4.78E-27	6.69E-25	7.34E-24	1.27E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	0.00E+00	7.89E-33	1.97E-31	6.36E-15	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.81E-27	1.61E-23	1.37E-21	1.40E-20	9.64E-09	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	0.00E+00	1.48E-29	3.28E-28	1.06E-13	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.29E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	5.77E-21	1.65E-15	6.19E-14	1.30E-12	2.76E-06	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	6.36E-24	2.02E-20	4.32E-19	5.48E-18	1.46E-07	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.44E-17	2.44E-17	3.93E-12	6.53E-11	1.74E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	6.64E-25	1.69E-20	6.64E-19	1.13E-17	1.73E-07	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.05E-09	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	9.00E-24	2.19E-18	3.50E-17	1.78E-06	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	1.77E-30	1.67E-26	3.95E-25	1.36E-06	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.90E-22	1.03E-16	4.09E-14	5.40E-13	7.85E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.34E-26	1.05E-21	5.04E-19	1.24E-17	1.10E-07	-

Table 7.20. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rosenbrock function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.09E-06	2.72E-04	6.38E-01	3.99E+00	6.79E-03	32
			1000	2.22E-02	2.47E+00	5.31E+00	6.85E+01	2.34E-03	-
	RING	nn = 2	10000	6.79E-05	1.64E-02	1.82E-02	7.14E-02	7.64E-03	4
			1000	9.88E-03	1.64E+00	2.03E+00	5.50E+00	1.63E-02	-
		nni = 2 nnf = (m – 1)	10000	1.74E-06	3.60E-04	3.93E-04	1.15E-03	1.20E-02	24
			1000	1.72E-02	4.02E+00	3.24E+00	5.10E+00	1.71E-02	-
	FWD	nn = 2	10000	1.14E+00	4.14E+00	4.12E+00	8.35E+00	1.02E-02	0
			1000	1.36E+00	5.34E+00	5.22E+00	9.58E+00	1.74E-02	-
		nni = 2 nnf = (m – 1)	10000	5.98E-05	9.99E-04	4.82E-01	4.01E+00	5.80E-03	4
			1000	4.87E-01	5.02E+00	4.72E+00	7.39E+00	1.03E-02	-
PSO-RRR1-1	GLOBAL		10000	1.13E-28	8.73E-01	1.64E+00	3.99E+00	1.08E-03	44
			1000	6.53E-06	1.46E+00	1.90E+00	4.99E+00	2.85E-03	-
	RING	nn = 2	10000	2.20E-10	3.45E-09	3.21E-08	5.63E-07	2.39E-03	100
			1000	5.10E-04	1.17E+00	1.30E+00	4.19E+00	1.11E-02	-
		nni = 2 nnf = (m – 1)	10000	5.92E-29	1.37E-28	1.59E-01	3.99E+00	1.36E-03	96
			1000	1.51E-04	1.16E-01	4.36E-01	4.17E+00	7.96E-03	-
	FWD	nn = 2	10000	8.06E-08	1.87E-06	7.89E-01	7.26E+00	2.41E-03	80
			1000	8.48E-01	2.32E+00	2.90E+00	7.33E+00	3.54E-03	-
		nni = 2 nnf = (m – 1)	10000	9.70E-29	2.18E-28	1.43E-26	1.51E-25	1.53E-03	100
			1000	4.62E-02	1.55E-01	7.58E-01	8.46E+00	3.46E-03	-
C-PSO-1	GLOBAL		10000	1.18E-10	4.49E-06	4.79E-01	3.99E+00	8.56E-03	76
			1000	2.73E-03	5.03E-01	7.26E+00	8.06E+01	6.76E-03	-
	RING	nn = 2	10000	1.23E-08	1.29E-03	1.61E-01	3.99E+00	7.32E-03	8
			1000	1.18E-02	3.08E+00	2.69E+00	5.08E+00	1.90E-02	-
		nni = 2 nnf = (m – 1)	10000	3.97E-08	8.83E-07	6.94E-06	1.28E-04	8.35E-03	96
			1000	1.33E-03	2.28E+00	2.11E+00	3.79E+00	1.63E-02	-
	FWD	nn = 2	10000	3.51E-01	4.05E+00	3.83E+00	7.43E+00	7.11E-03	0
			1000	4.88E-01	5.50E+00	5.87E+00	1.82E+01	1.13E-02	-
		nni = 2 nnf = (m – 1)	10000	4.49E-07	2.83E-06	6.57E-04	1.39E-02	6.88E-03	92
			1000	2.32E+00	4.72E+00	4.86E+00	8.56E+00	9.37E-03	-
Multi-Swarm	GLOBAL		10000	5.91E-12	2.94E-02	1.02E+00	5.59E+00	7.53E-03	16
			1000	9.37E-05	6.67E-01	1.47E+00	6.46E+00	7.93E-03	-
	RING	nn = 2	10000	4.01E-09	8.03E-07	1.61E-01	3.99E+00	8.87E-03	80
			1000	2.63E-03	1.62E+00	1.82E+00	5.11E+00	1.79E-02	-
		nni = 2 nnf = (m – 1)	10000	1.88E-15	1.16E-09	1.42E-05	3.49E-04	1.08E-02	96
			1000	5.39E-04	6.18E-01	8.49E-01	4.07E+00	1.45E-02	-
	FWD	nn = 2	10000	9.26E-03	3.47E+00	3.47E+00	7.03E+00	4.92E-03	0
			1000	1.09E-02	5.46E+00	5.35E+00	9.55E+00	9.53E-03	-
		nni = 2 nnf = (m – 1)	10000	3.40E-12	4.82E-09	4.80E-01	4.01E+00	4.57E-03	76
			1000	8.74E-01	4.48E+00	4.63E+00	9.59E+00	6.54E-03	-

Table 7.21. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rosenbrock function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.41E-04	1.27E+01	1.03E+01	1.88E+01	1.45E-03	0
			1000	8.48E+00	2.80E+01	5.20E+01	1.24E+02	2.95E-04	-
	RING	nn = 2	10000	1.14E-01	1.00E+01	1.06E+01	2.31E+01	1.13E-02	0
			1000	4.82E+01	1.40E+02	1.48E+02	3.09E+02	1.57E-02	-
		nni = 2 nnf = (m – 1)	10000	2.91E-07	1.46E+01	1.29E+01	2.28E+01	7.38E-03	4
			1000	2.84E+01	6.81E+01	7.44E+01	1.43E+02	1.16E-02	-
	FWD	nn = 2	10000	2.42E+01	2.95E+01	4.12E+01	9.17E+01	4.89E-03	0
			1000	8.59E+01	1.64E+02	1.91E+02	6.58E+02	9.83E-03	-
		nni = 2 nnf = (m – 1)	10000	4.49E-02	1.66E+01	2.22E+01	7.25E+01	3.38E-03	0
			1000	3.31E+01	9.89E+01	1.03E+02	1.77E+02	5.95E-03	-
PSO-RRR1-1	GLOBAL		10000	2.27E+01	9.15E+01	1.06E+02	3.67E+02	3.98E-11	0
			1000	2.43E+01	9.38E+01	1.10E+02	3.69E+02	2.84E-08	-
	RING	nn = 2	10000	8.78E-03	7.24E+00	7.16E+00	1.91E+01	2.15E-03	0
			1000	8.28E+00	2.61E+01	4.28E+01	1.77E+02	4.38E-03	-
		nni = 2 nnf = (m – 1)	10000	2.56E-17	9.97E-13	1.35E+00	9.73E+00	2.25E-03	72
			1000	2.21E+01	2.48E+01	3.96E+01	8.33E+01	2.28E-03	-
	FWD	nn = 2	10000	4.36E-03	6.77E+00	1.36E+01	7.96E+01	8.13E-05	0
			1000	1.40E+01	2.55E+01	3.74E+01	1.50E+02	3.63E-05	-
		nni = 2 nnf = (m – 1)	10000	9.54E-16	2.05E-11	1.12E+00	3.99E+00	4.13E-04	72
			1000	8.21E+00	2.42E+01	3.30E+01	9.27E+01	1.52E-05	-
C-PSO-1	GLOBAL		10000	1.17E-05	3.90E-02	1.05E+00	4.02E+00	5.58E-03	8
			1000	1.55E+00	2.22E+01	3.58E+01	1.79E+02	9.02E-04	-
	RING	nn = 2	10000	2.89E-03	6.94E-01	3.39E+00	1.79E+01	7.42E-03	0
			1000	1.32E+01	2.89E+01	5.04E+01	1.45E+02	1.30E-02	-
		nni = 2 nnf = (m – 1)	10000	3.41E-06	3.73E+00	3.16E+00	1.00E+01	6.55E-03	4
			1000	1.13E+01	2.63E+01	3.40E+01	8.12E+01	8.11E-03	-
	FWD	nn = 2	10000	1.46E+01	2.99E+01	4.02E+01	1.02E+02	2.26E-03	0
			1000	2.70E+01	4.75E+01	6.34E+01	1.55E+02	4.32E-03	-
		nni = 2 nnf = (m – 1)	10000	3.68E-05	3.85E+00	3.23E+00	5.59E+00	2.15E-03	4
			1000	2.00E+01	2.74E+01	4.45E+01	1.19E+02	3.65E-03	-
Multi-Swarm	GLOBAL		10000	4.21E-08	1.67E+01	2.70E+01	7.68E+01	3.90E-03	4
			1000	2.33E-02	2.30E+01	4.38E+01	1.36E+02	2.59E-03	-
	RING	nn = 2	10000	9.14E-03	7.09E+00	6.59E+00	1.46E+01	3.01E-03	0
			1000	6.26E+00	7.08E+01	5.33E+01	8.71E+01	5.27E-03	-
		nni = 2 nnf = (m – 1)	10000	1.56E-05	5.30E+00	6.36E+00	1.98E+01	2.76E-03	4
			1000	4.72E+00	2.73E+01	4.05E+01	1.34E+02	3.79E-03	-
	FWD	nn = 2	10000	1.95E+01	3.25E+01	5.09E+01	1.77E+02	3.71E-03	0
			1000	3.27E+01	7.61E+01	9.11E+01	2.12E+02	5.89E-03	-
		nni = 2 nnf = (m – 1)	10000	2.04E-04	1.10E+00	4.08E+00	7.00E+01	2.35E-03	0
			1000	2.25E+01	2.71E+01	5.27E+01	2.17E+02	1.53E-03	-

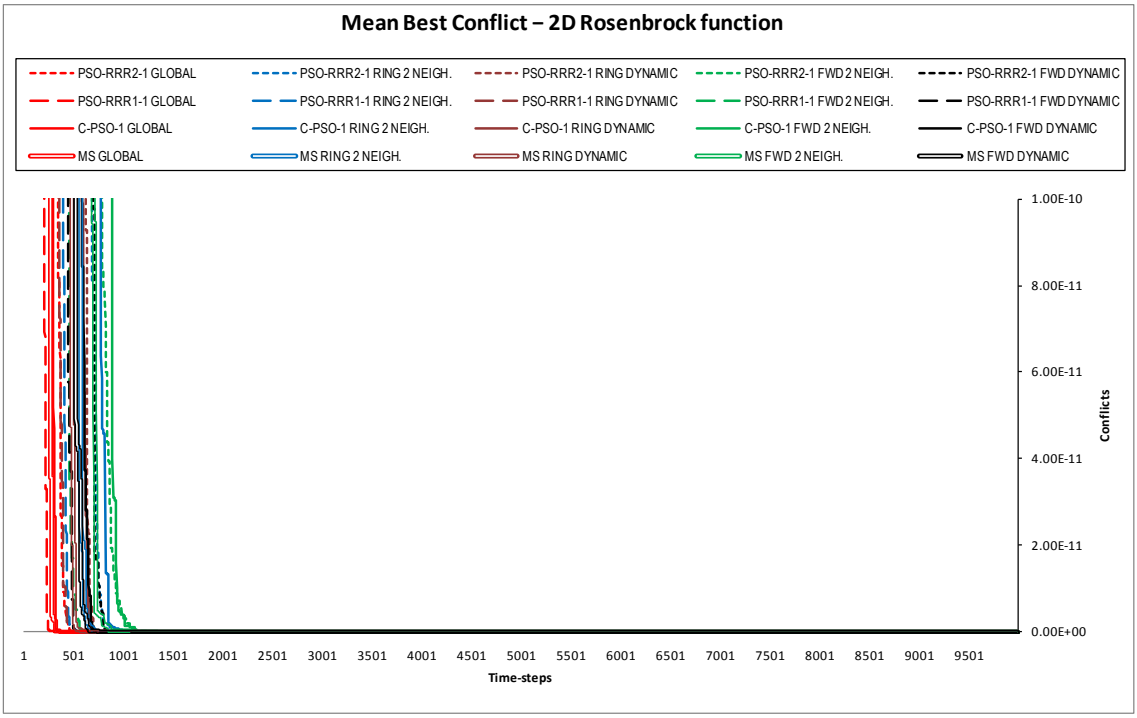


Fig. 7.33. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 7.19. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

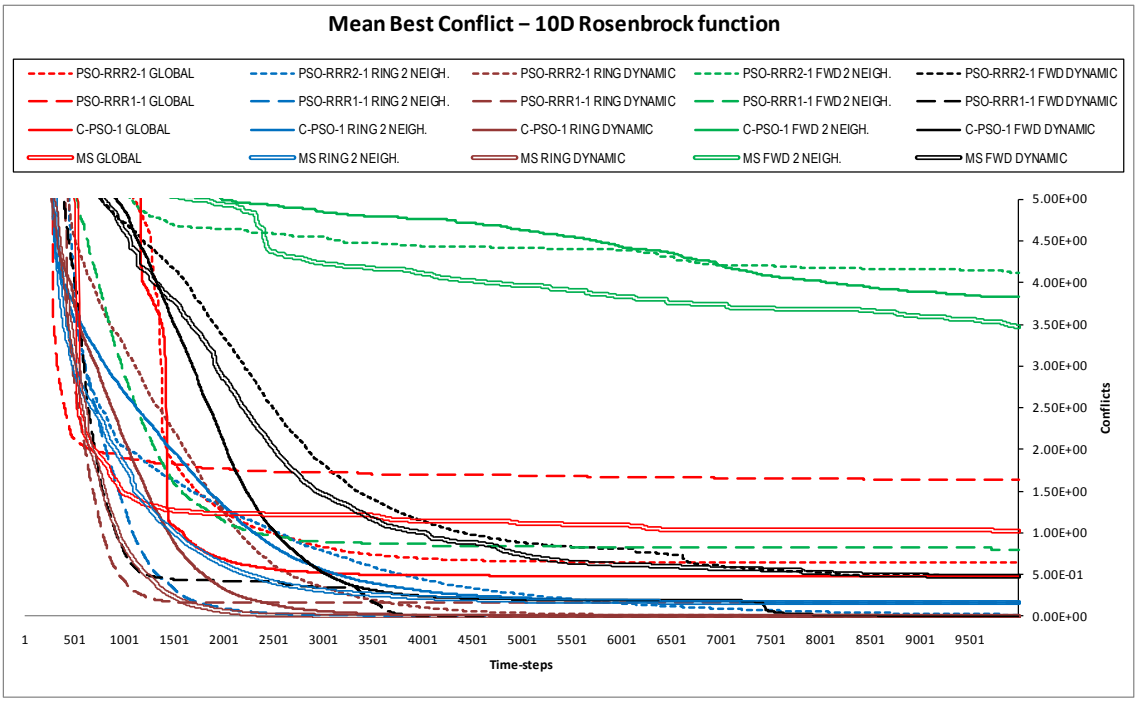


Fig. 7.34. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.20. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

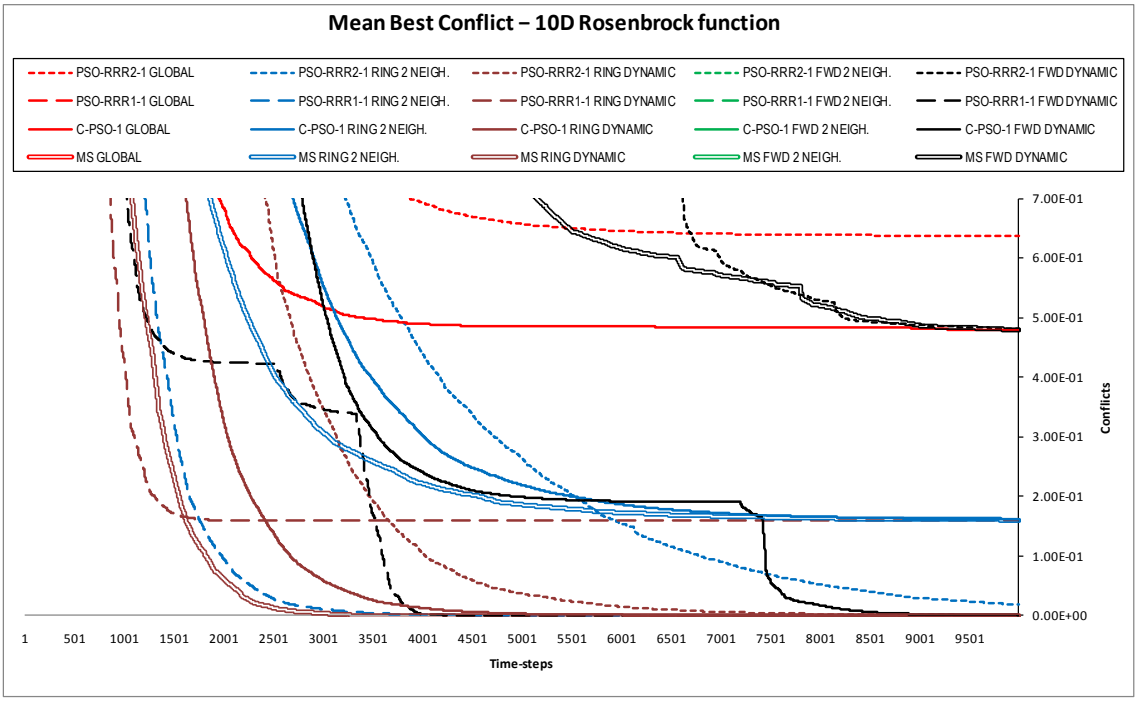


Fig. 7.35. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.20. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

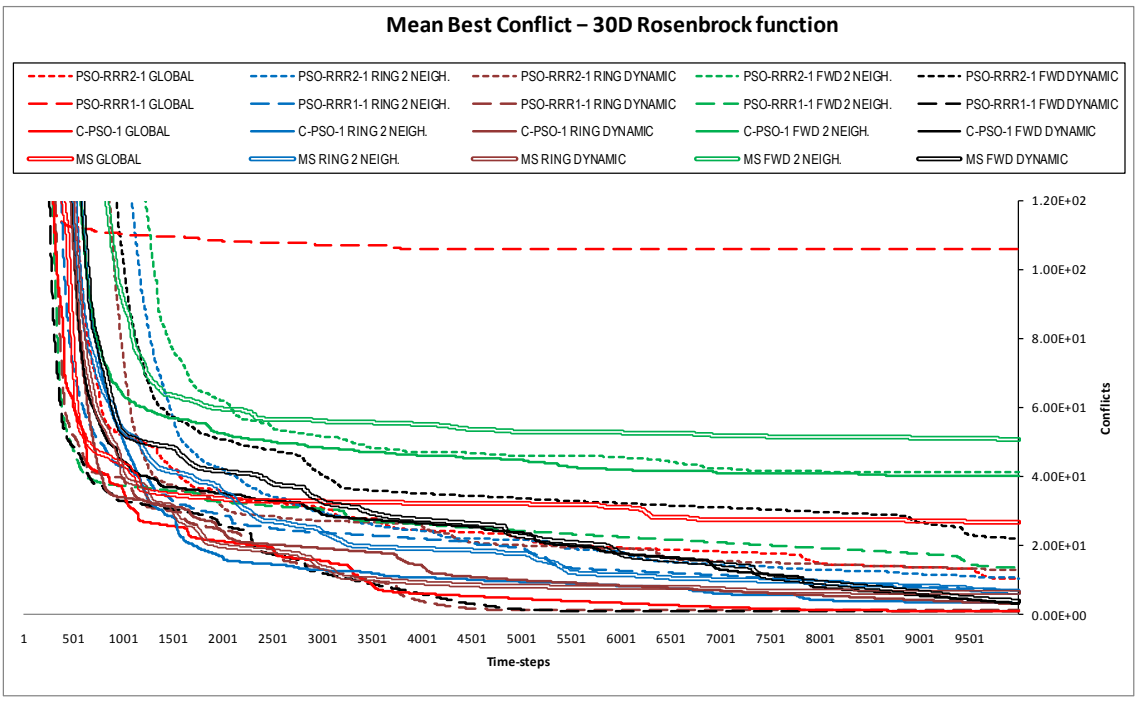


Fig. 7.36. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.21. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

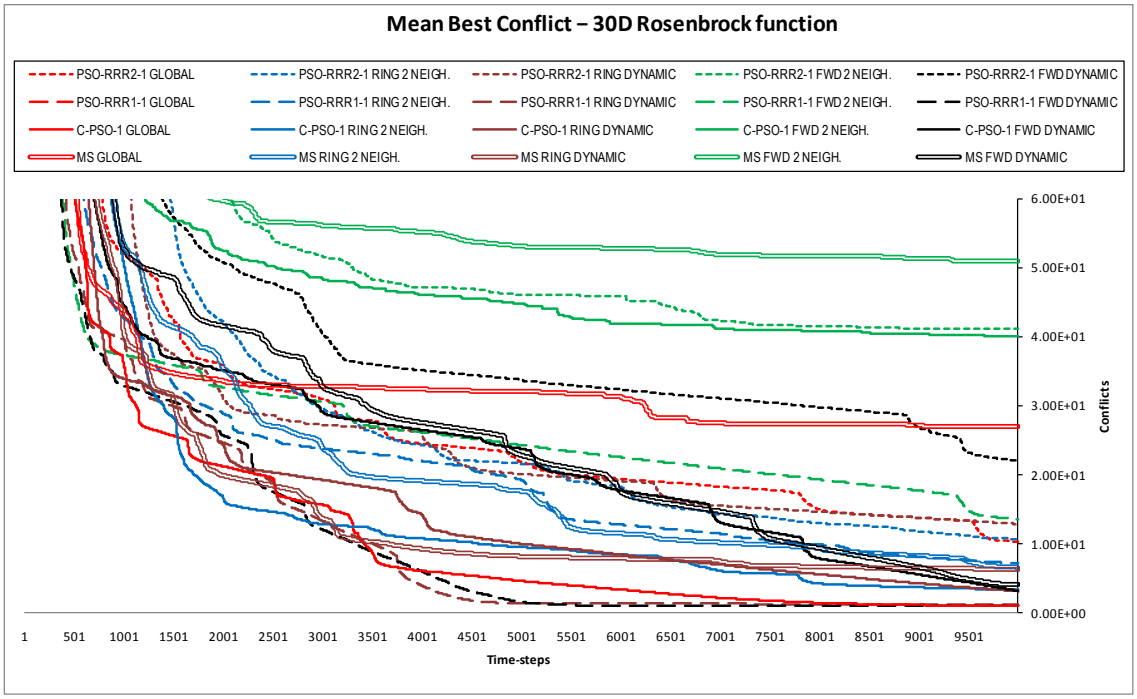


Fig. 7.37. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.21. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

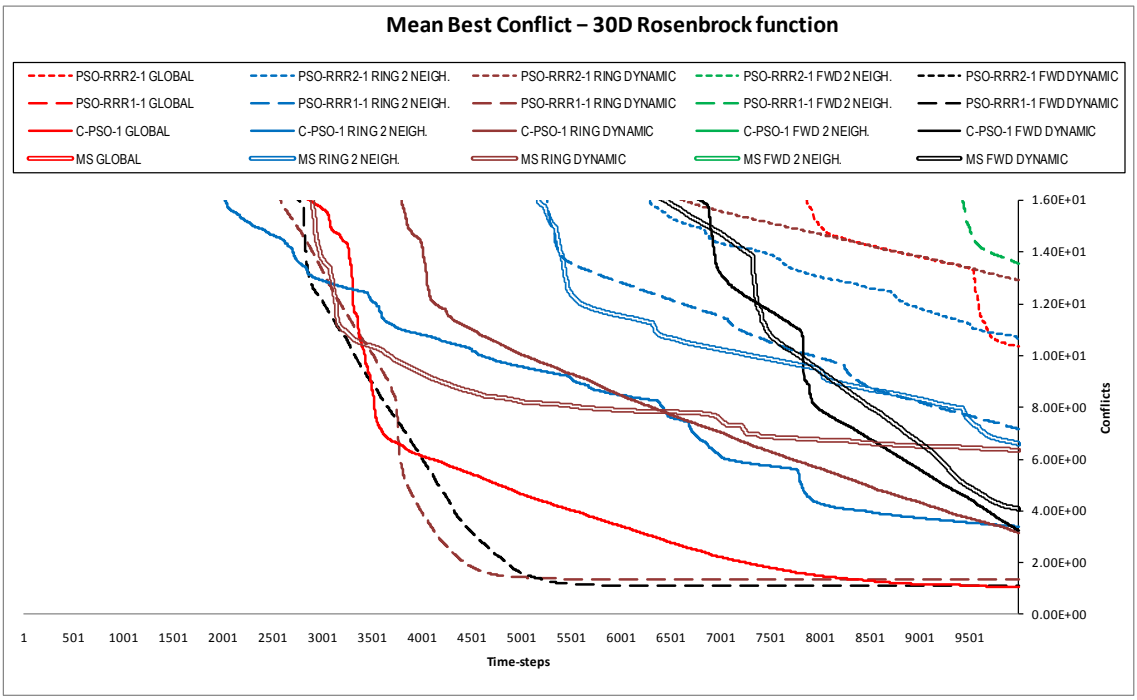


Fig. 7.38. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.21. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.22. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Rastrigin function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.07E-10	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.42E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.96E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.96E-11	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.34E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.34E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.18E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.18E-11	-
PSO-RRR1-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.09E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.09E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.05E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.05E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.60E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.60E-11	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.99E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.99E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.09E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.09E-11	-
C-PSO-1	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.79E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.79E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.52E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.52E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.26E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.26E-11	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.24E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.24E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.95E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.95E-11	-
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.73E-11	100
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.95E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.95E-11	-

Table 7.23. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Rastrigin function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	9.95E-01	2.98E+00	2.95E+00	6.96E+00	3.48E-04	0
			1000	9.95E-01	2.98E+00	3.02E+00	6.96E+00	2.20E-03	-
	RING	nn = 2	10000	0.00E+00	1.99E+00	2.15E+00	4.97E+00	2.71E-02	20
			1000	1.99E+00	4.22E+00	4.43E+00	7.96E+00	3.54E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	7.16E-01	3.98E+00	1.65E-02	60
			1000	9.95E-01	3.98E+00	4.04E+00	1.09E+01	3.51E-02	-
	FWD	nn = 2	10000	5.27E-03	2.06E+00	2.26E+00	4.58E+00	2.67E-02	0
			1000	2.54E+00	5.30E+00	5.57E+00	8.53E+00	3.63E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	4.43E-01	2.98E+00	1.41E-02	68
			1000	6.01E-01	4.00E+00	4.28E+00	7.88E+00	3.33E-02	-
PSO-RRR1-1	GLOBAL		10000	5.97E+00	1.19E+01	1.35E+01	2.49E+01	1.21E-11	0
			1000	5.97E+00	1.19E+01	1.35E+01	2.49E+01	1.47E-11	-
	RING	nn = 2	10000	9.95E-01	4.97E+00	5.18E+00	1.09E+01	3.82E-02	0
			1000	2.98E+00	7.96E+00	7.94E+00	1.37E+01	4.30E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.98E+00	3.02E+00	5.97E+00	2.79E-02	8
			1000	2.98E+00	5.97E+00	7.44E+00	1.59E+01	4.40E-02	-
	FWD	nn = 2	10000	4.36E-03	2.48E+00	3.02E+00	8.87E+00	2.80E-02	0
			1000	3.53E+00	6.57E+00	6.85E+00	1.19E+01	3.84E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.51E+00	3.98E+00	1.48E-02	20
			1000	1.50E+00	5.85E+00	5.92E+00	1.17E+01	3.48E-02	-
C-PSO-1	GLOBAL		10000	1.99E+00	3.98E+00	4.93E+00	1.09E+01	1.92E-11	0
			1000	1.99E+00	4.97E+00	5.17E+00	1.19E+01	8.71E-04	-
	RING	nn = 2	10000	0.00E+00	2.98E+00	2.79E+00	4.97E+00	2.93E-02	12
			1000	1.99E+00	3.98E+00	4.55E+00	7.96E+00	3.79E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.23E+00	5.97E+00	1.98E-02	48
			1000	1.99E+00	3.98E+00	4.66E+00	7.96E+00	3.65E-02	-
	FWD	nn = 2	10000	3.80E-03	2.04E+00	2.38E+00	5.49E+00	2.44E-02	0
			1000	1.57E+00	5.94E+00	5.99E+00	9.35E+00	3.56E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	5.57E-01	1.99E+00	1.22E-02	52
			1000	9.96E-01	3.84E+00	3.96E+00	6.88E+00	3.35E-02	-
Multi-Swarm	GLOBAL		10000	1.99E+00	3.98E+00	4.78E+00	1.49E+01	1.84E-11	0
			1000	1.99E+00	3.98E+00	5.13E+00	1.49E+01	5.46E-04	-
	RING	nn = 2	10000	0.00E+00	2.98E+00	2.75E+00	6.96E+00	2.90E-02	4
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.68E+00	5.97E+00	1.94E-02	32
			1000	2.18E+00	4.97E+00	5.26E+00	1.09E+01	3.66E-02	-
	FWD	nn = 2	10000	6.41E-05	2.12E+00	2.34E+00	6.20E+00	2.45E-02	4
			1000	2.80E+00	6.48E+00	6.42E+00	1.03E+01	3.42E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.07E+00	2.98E+00	1.66E-02	28
			1000	2.25E+00	5.07E+00	5.03E+00	9.50E+00	3.34E-02	-

Table 7.24. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Rastrigin function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	**2.69E+01**	**4.28E+01**	**4.13E+01**	**5.57E+01**	**2.64E-11**	**0**
			1000	2.69E+01	4.28E+01	4.14E+01	5.57E+01	3.99E-05	-
	RING	nn = 2	**10000**	**2.98E+01**	**4.40E+01**	**4.29E+01**	**5.29E+01**	**2.61E-02**	**0**
			1000	3.46E+01	5.32E+01	5.24E+01	7.23E+01	3.00E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**2.69E+01**	**4.28E+01**	**4.31E+01**	**6.96E+01**	**2.97E-02**	**0**
			1000	3.84E+01	4.88E+01	5.19E+01	7.99E+01	2.99E-02	-
	FWD	nn = 2	**10000**	**4.09E+01**	**6.23E+01**	**6.35E+01**	**8.78E+01**	**2.87E-02**	**0**
			1000	5.77E+01	8.37E+01	8.38E+01	1.05E+02	3.23E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**1.29E+01**	**3.28E+01**	**3.25E+01**	**4.48E+01**	**2.29E-02**	**0**
			1000	5.06E+01	6.92E+01	6.97E+01	9.26E+01	3.13E-02	-
PSO-RRR1-1	GLOBAL		**10000**	**2.49E+01**	**7.16E+01**	**7.41E+01**	**1.28E+02**	**6.68E-16**	**0**
			1000	2.49E+01	7.16E+01	7.41E+01	1.28E+02	1.47E-15	-
	RING	nn = 2	**10000**	**2.19E+01**	**4.68E+01**	**4.65E+01**	**6.17E+01**	**3.00E-02**	**0**
			1000	2.20E+01	5.01E+01	5.03E+01	6.71E+01	3.12E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**2.29E+01**	**4.88E+01**	**4.91E+01**	**7.46E+01**	**3.15E-02**	**0**
			1000	3.48E+01	4.88E+01	5.13E+01	7.98E+01	3.27E-02	-
	FWD	nn = 2	**10000**	**4.15E+01**	**7.21E+01**	**7.14E+01**	**1.03E+02**	**2.36E-02**	**0**
			1000	5.63E+01	8.37E+01	8.69E+01	1.16E+02	2.58E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**1.99E+01**	**3.98E+01**	**4.14E+01**	**6.57E+01**	**2.03E-02**	**0**
			1000	4.38E+01	7.73E+01	7.87E+01	1.13E+02	2.57E-02	-
C-PSO-1	GLOBAL		**10000**	**2.69E+01**	**4.88E+01**	**5.37E+01**	**9.65E+01**	**1.93E-11**	**0**
			1000	2.69E+01	4.88E+01	5.37E+01	9.65E+01	1.09E-10	-
	RING	nn = 2	**10000**	**2.89E+01**	**5.37E+01**	**5.05E+01**	**6.87E+01**	**3.31E-02**	**0**
			1000	2.89E+01	5.88E+01	5.59E+01	7.79E+01	3.43E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**2.19E+01**	**5.27E+01**	**5.11E+01**	**7.36E+01**	**3.15E-02**	**0**
			1000	2.69E+01	5.77E+01	5.59E+01	8.28E+01	3.36E-02	-
	FWD	nn = 2	**10000**	**5.32E+01**	**7.29E+01**	**7.32E+01**	**9.73E+01**	**2.97E-02**	**0**
			1000	6.33E+01	9.06E+01	9.20E+01	1.29E+02	2.99E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**1.69E+01**	**3.68E+01**	**3.81E+01**	**5.57E+01**	**2.53E-02**	**0**
			1000	5.01E+01	8.45E+01	8.26E+01	1.11E+02	3.20E-02	-
Multi-Swarm	GLOBAL		**10000**	**2.59E+01**	**5.27E+01**	**5.33E+01**	**8.16E+01**	**1.89E-11**	**0**
			1000	2.59E+01	5.27E+01	5.33E+01	8.16E+01	5.36E-08	-
	RING	nn = 2	**10000**	**3.28E+01**	**4.48E+01**	**4.56E+01**	**6.37E+01**	**3.03E-02**	**0**
			1000	3.32E+01	5.21E+01	4.97E+01	6.57E+01	3.09E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**2.59E+01**	**3.98E+01**	**4.32E+01**	**6.67E+01**	**2.77E-02**	**0**
			1000	2.72E+01	4.48E+01	4.77E+01	6.83E+01	2.96E-02	-
	FWD	nn = 2	**10000**	**2.96E+01**	**6.81E+01**	**6.71E+01**	**8.76E+01**	**3.07E-02**	**0**
			1000	4.94E+01	8.76E+01	8.55E+01	1.21E+02	3.31E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**1.69E+01**	**3.38E+01**	**3.34E+01**	**5.67E+01**	**2.18E-02**	**0**
			1000	3.73E+01	7.11E+01	7.11E+01	9.42E+01	3.10E-02	-

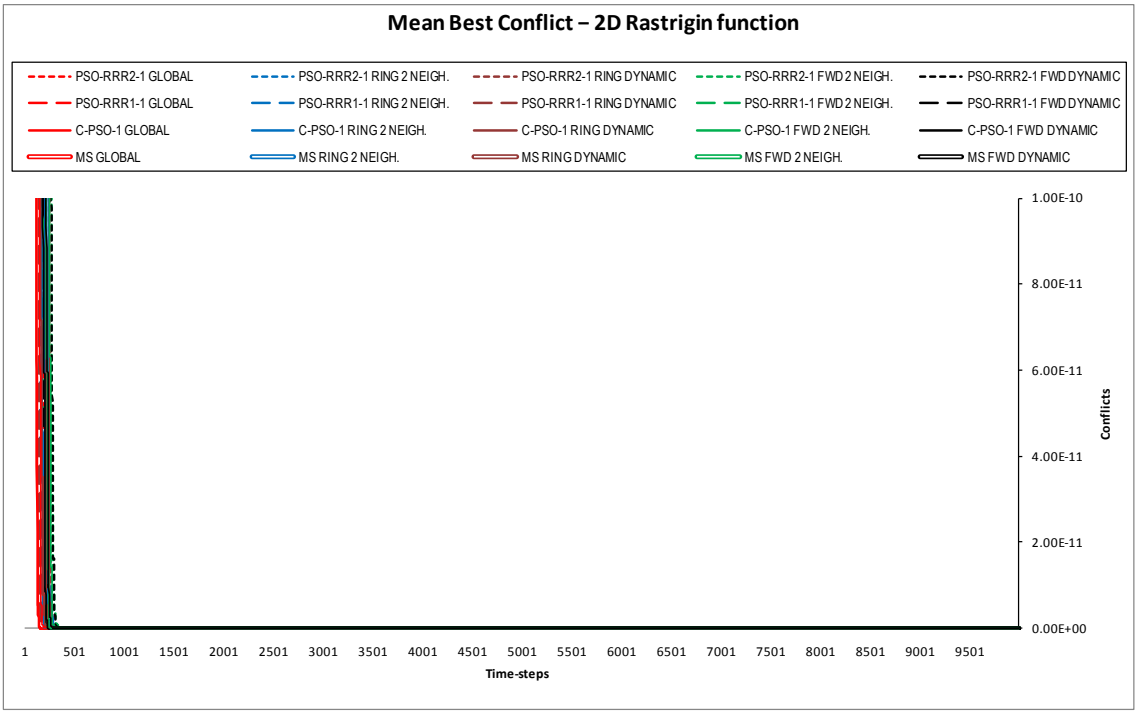


Fig. 7.39. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 7.22. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

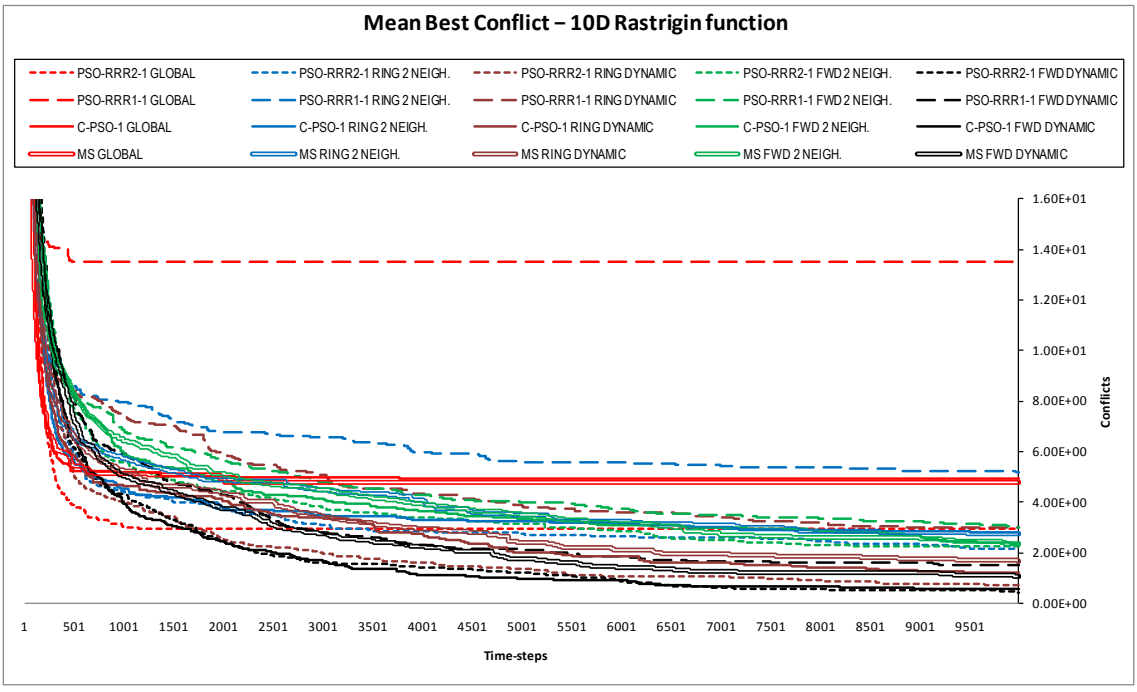


Fig. 7.40. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.23. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

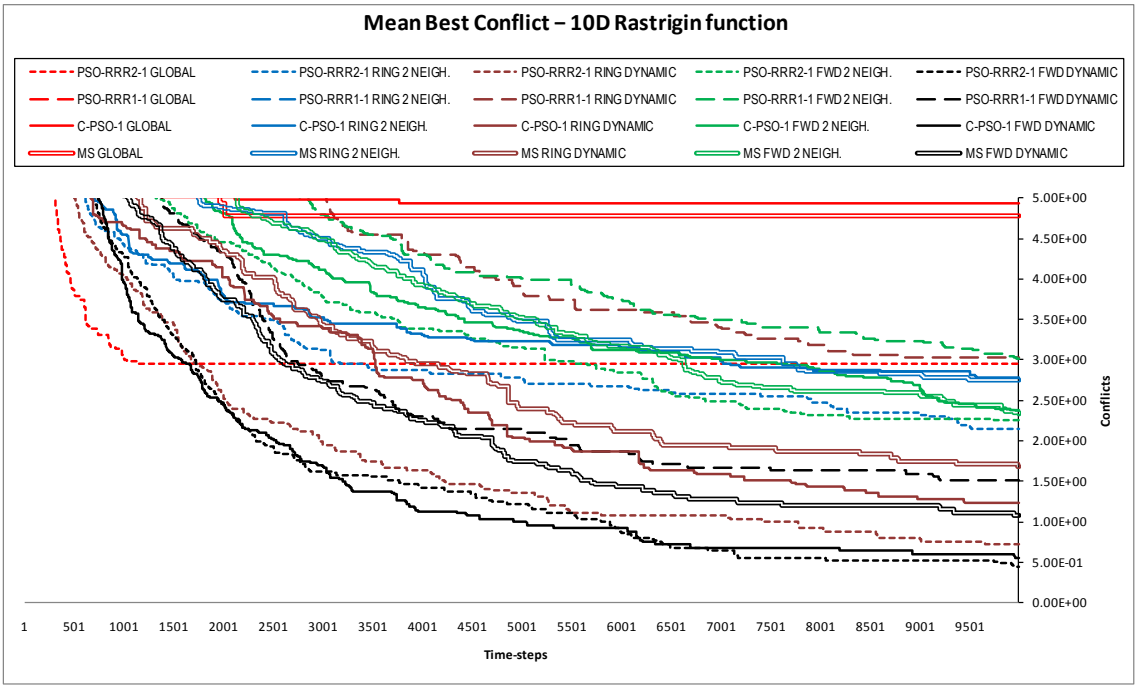


Fig. 7.41. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.23. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

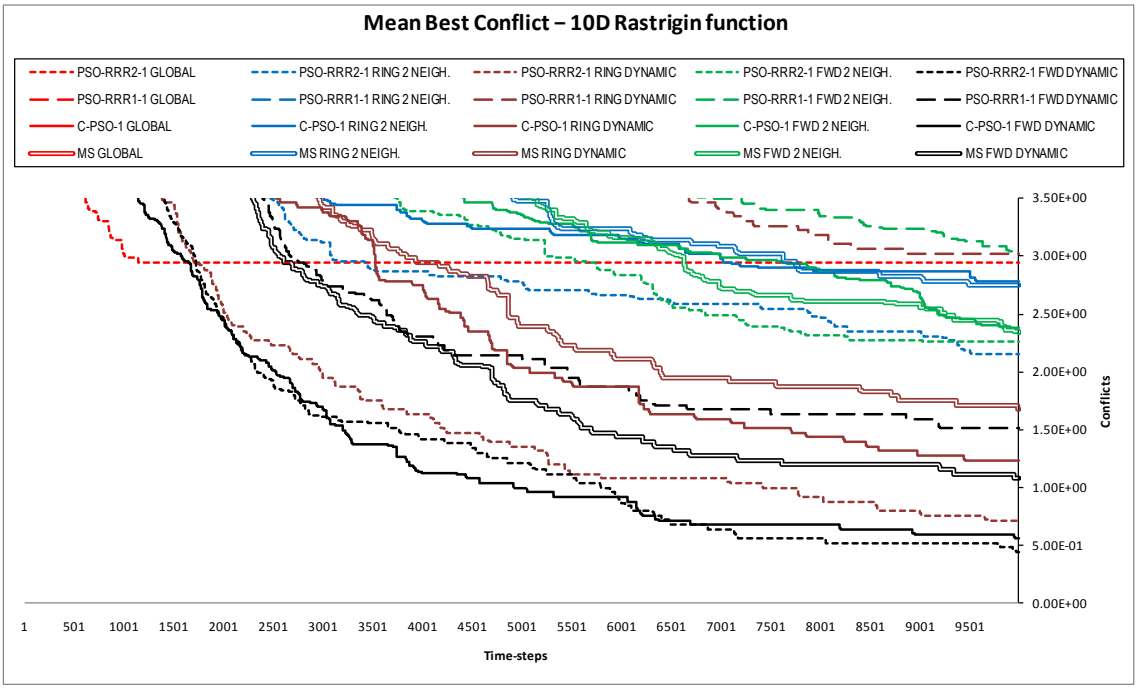


Fig. 7.42. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.23. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

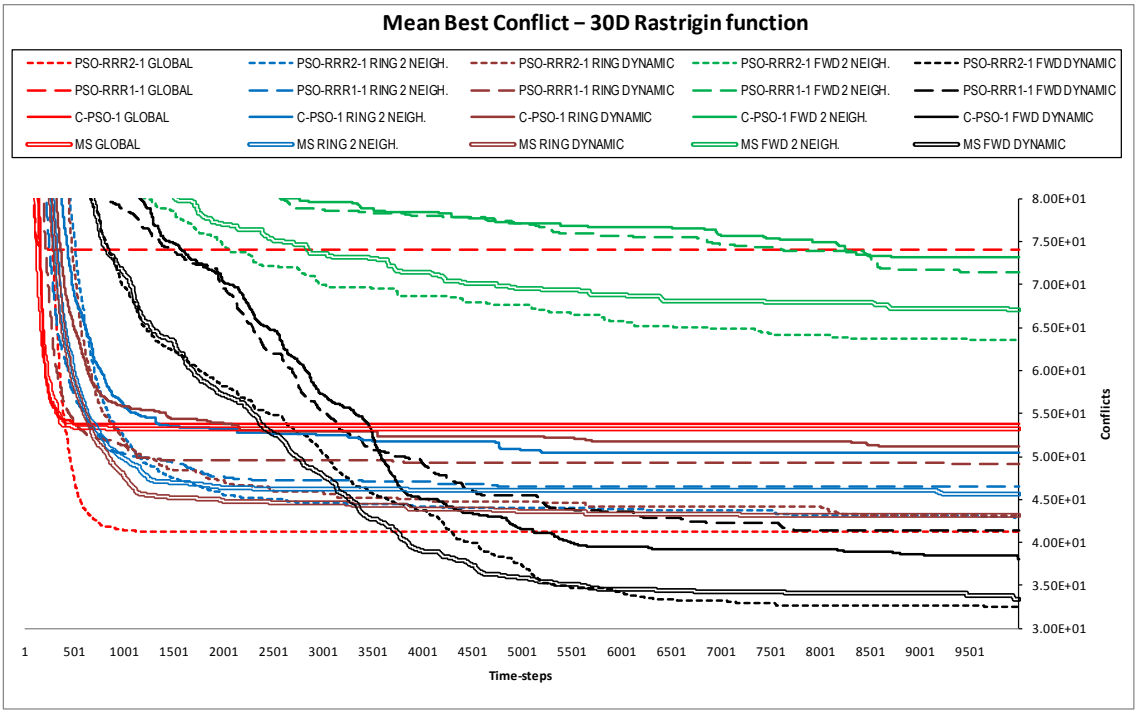


Fig. 7.43. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.24. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

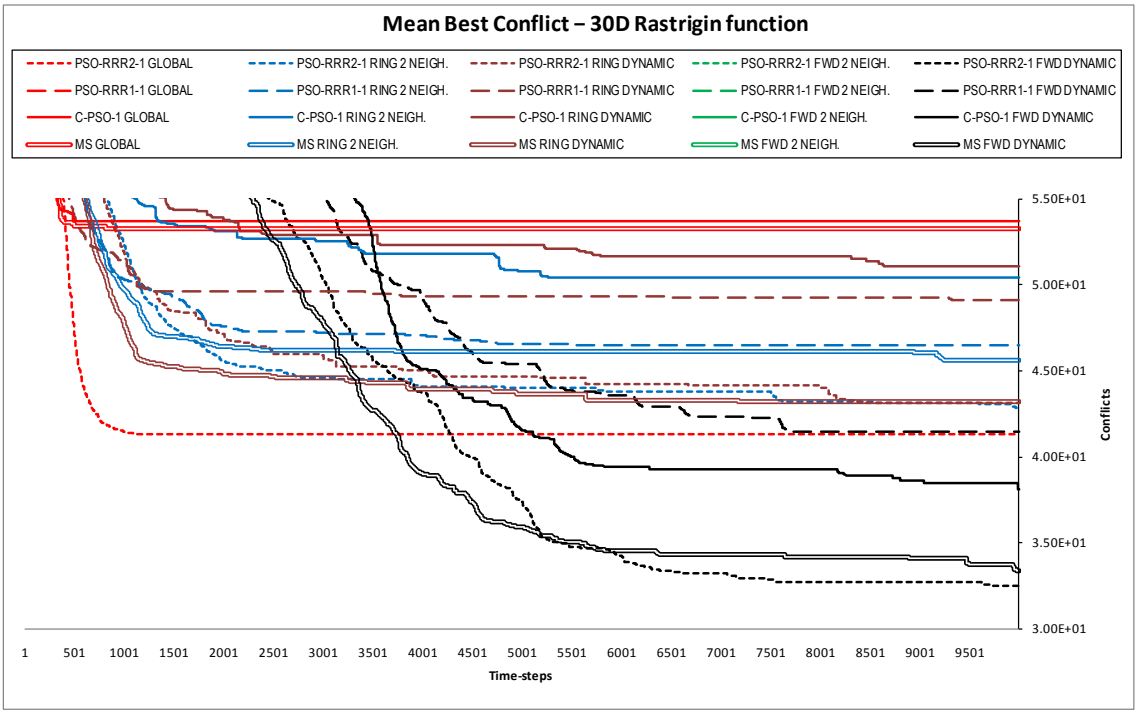


Fig. 7.44. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.24. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.25. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Griewank function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	0.00E+00	0.00E+00	2.96E-04	7.40E-03	**7.41E-12**	**96**
			1000	0.00E+00	0.00E+00	5.92E-04	7.40E-03	1.10E-03	-
	RING	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**3.65E-04**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.10E-03	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.57E-12**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.64E-03	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**1.48E-04**	**100**
			1000	0.00E+00	0.00E+00	8.90E-15	2.22E-13	2.02E-03	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.59E-12**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.67E-03	-
PSO-RRR1-1	GLOBAL		**10000**	0.00E+00	0.00E+00	5.92E-04	7.40E-03	**5.76E-12**	**92**
			1000	0.00E+00	0.00E+00	1.18E-03	9.08E-04	9.08E-04	-
	RING	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**4.15E-04**	**100**
			1000	0.00E+00	0.00E+00	3.02E-13	7.54E-12	2.05E-03	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.45E-12**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.67E-03	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**1.37E-04**	**100**
			1000	0.00E+00	0.00E+00	3.58E-09	8.74E-08	2.03E-03	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.87E-12**	**100**
			1000	0.00E+00	0.00E+00	2.18E-16	5.44E-15	1.54E-03	-
C-PSO-1	GLOBAL		**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.40E-12**	**100**
			1000	0.00E+00	0.00E+00	1.18E-03	7.40E-03	9.46E-04	-
	RING	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**1.57E-04**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.96E-03	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.81E-12**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.67E-03	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.13E-05**	**100**
			1000	0.00E+00	0.00E+00	2.05E-10	4.60E-09	1.89E-03	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.87E-12**	**100**
			1000	0.00E+00	0.00E+00	6.39E-12	1.60E-10	1.58E-03	-
Multi-Swarm	GLOBAL		**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.56E-12**	**100**
			1000	0.00E+00	0.00E+00	5.95E-16	1.49E-14	8.92E-04	-
	RING	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**2.51E-04**	**100**
			1000	0.00E+00	0.00E+00	2.39E-05	5.88E-04	2.05E-03	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.74E-12**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.59E-03	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**1.67E-04**	**100**
			1000	0.00E+00	0.00E+00	4.02E-08	8.51E-07	1.92E-03	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**6.88E-12**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.53E-03	-

Table 7.26. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Griewank function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	1.97E-02	5.66E-02	6.81E-02	1.43E-01	4.94E-07	0
			1000	1.97E-02	6.16E-02	7.14E-02	1.43E-01	1.18E-04	-
	RING	nn = 2	10000	0.00E+00	2.46E-02	2.66E-02	6.15E-02	1.81E-03	4
			1000	2.96E-07	2.95E-02	3.41E-02	6.88E-02	2.01E-03	-
		nni = 2 nnf = (m – 1)	10000	7.40E-03	2.96E-02	3.02E-02	5.65E-02	1.35E-03	0
			1000	9.86E-03	3.94E-02	3.68E-02	5.66E-02	1.54E-03	-
	FWD	nn = 2	10000	1.30E-02	3.15E-02	3.53E-02	8.10E-02	1.34E-03	0
			1000	4.99E-02	8.36E-02	8.31E-02	1.27E-01	1.65E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.71E-02	2.45E-02	5.90E-02	1.04E-03	16
			1000	1.02E-02	5.65E-02	5.98E-02	1.21E-01	1.54E-03	-
PSO-RRR1-1	GLOBAL		10000	2.96E-02	9.11E-02	9.27E-02	1.82E-01	1.81E-12	0
			1000	2.96E-02	9.11E-02	9.27E-02	1.82E-01	1.48E-05	-
	RING	nn = 2	10000	0.00E+00	3.19E-02	3.05E-02	7.38E-02	1.58E-03	8
			1000	0.00E+00	3.94E-02	3.64E-02	7.62E-02	1.66E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.22E-02	2.95E-02	6.64E-02	1.27E-03	12
			1000	9.86E-03	4.18E-02	4.26E-02	1.11E-01	1.61E-03	-
	FWD	nn = 2	10000	1.11E-08	1.96E-02	2.34E-02	6.66E-02	1.03E-03	4
			1000	1.07E-03	5.21E-02	4.87E-02	8.87E-02	1.26E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	1.23E-02	1.75E-02	5.41E-02	6.64E-04	4
			1000	4.97E-03	3.35E-02	4.19E-02	9.49E-02	1.25E-03	-
C-PSO-1	GLOBAL		10000	1.97E-02	6.64E-02	6.68E-02	1.38E-01	1.65E-06	0
			1000	2.71E-02	6.89E-02	7.21E-02	1.38E-01	1.22E-04	-
	RING	nn = 2	10000	0.00E+00	2.46E-02	2.36E-02	4.68E-02	1.55E-03	4
			1000	0.00E+00	2.71E-02	2.85E-02	8.87E-02	1.66E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.71E-02	2.49E-02	5.91E-02	1.19E-03	4
			1000	7.40E-03	3.69E-02	3.52E-02	7.38E-02	1.52E-03	-
	FWD	nn = 2	10000	2.19E-05	1.59E-02	2.16E-02	5.97E-02	1.23E-03	4
			1000	2.26E-02	6.27E-02	6.45E-02	2.06E-01	1.57E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	1.72E-02	1.94E-02	5.66E-02	8.92E-04	12
			1000	8.60E-03	4.60E-02	5.14E-02	9.34E-02	1.43E-03	-
Multi-Swarm	GLOBAL		10000	1.48E-02	6.64E-02	6.64E-02	1.38E-01	1.48E-05	0
			1000	2.95E-02	7.13E-02	7.85E-02	1.85E-01	1.40E-04	-
	RING	nn = 2	10000	0.00E+00	1.97E-02	2.15E-02	5.90E-02	1.60E-03	8
			1000	7.40E-03	2.22E-02	2.71E-02	5.90E-02	1.75E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.46E-02	2.77E-02	6.89E-02	1.47E-03	8
			1000	0.00E+00	3.45E-02	3.59E-02	7.38E-02	1.74E-03	-
	FWD	nn = 2	10000	4.64E-05	3.60E-02	3.64E-02	9.76E-02	1.30E-03	4
			1000	2.16E-02	6.01E-02	6.36E-02	1.49E-01	1.53E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	1.72E-02	2.33E-02	5.90E-02	1.08E-03	12
			1000	7.94E-03	5.59E-02	6.33E-02	1.25E-01	1.73E-03	-

Table 7.27. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Griewank function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	0.00E+00	7.40E-03	9.35E-03	2.96E-02	**4.23E-12**	**44**
			1000	7.95E-06	7.44E-03	9.40E-03	2.96E-02	2.93E-06	-
	RING	nn = 2	**10000**	0.00E+00	0.00E+00	2.96E-04	7.40E-03	**1.22E-06**	**96**
			1000	2.54E-01	4.15E-01	4.23E-01	6.79E-01	2.52E-04	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	4.05E-03	1.72E-02	**3.77E-06**	**64**
			1000	3.65E-02	7.18E-02	8.05E-02	1.70E-01	1.13E-04	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	3.40E-06	8.14E-05	**1.88E-06**	**100**
			1000	2.79E-01	4.89E-01	4.85E-01	7.33E-01	2.52E-04	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	1.19E-03	1.23E-02	**1.25E-07**	**84**
			1000	2.30E-02	1.37E-01	1.42E-01	2.65E-01	1.44E-04	-
PSO-RRR1-1	GLOBAL		**10000**	3.29E-08	6.46E-02	1.02E-01	7.40E-01	**3.91E-13**	**4**
			1000	3.29E-08	7.11E-02	1.12E-01	7.40E-01	4.04E-14	-
	RING	nn = 2	**10000**	0.00E+00	0.00E+00	6.90E-04	9.86E-03	**6.40E-08**	**92**
			1000	4.02E-11	7.73E-10	6.91E-04	9.86E-03	1.38E-06	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	1.58E-03	9.86E-03	**8.49E-09**	**80**
			1000	0.00E+00	4.76E-08	3.15E-03	1.48E-02	1.11E-06	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	8.17E-04	1.26E-02	**2.62E-07**	**88**
			1000	3.02E-12	1.61E-03	3.88E-03	2.08E-02	1.43E-05	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	2.56E-03	1.72E-02	**3.65E-12**	**80**
			1000	0.00E+00	0.00E+00	3.68E-03	1.73E-02	5.83E-06	-
C-PSO-1	GLOBAL		**10000**	0.00E+00	1.23E-02	1.79E-02	7.09E-02	**2.56E-12**	**36**
			1000	0.00E+00	1.23E-02	1.79E-02	7.09E-02	2.93E-12	-
	RING	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**1.14E-07**	**100**
			1000	4.55E-06	3.98E-05	2.60E-03	2.22E-02	1.52E-05	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	1.97E-03	1.72E-02	**4.73E-07**	**84**
			1000	3.22E-09	3.49E-08	2.37E-03	1.72E-02	8.37E-06	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	2.96E-04	7.40E-03	**1.72E-07**	**96**
			1000	1.75E-05	5.85E-03	1.41E-02	7.88E-02	6.27E-05	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	1.28E-03	9.86E-03	**3.91E-11**	**84**
			1000	1.16E-08	2.73E-07	6.22E-03	6.23E-02	1.84E-05	-
Multi-Swarm	GLOBAL		**10000**	0.00E+00	4.67E-02	5.18E-02	1.41E-01	**2.42E-12**	**4**
			1000	6.66E-16	4.67E-02	5.18E-02	1.41E-01	4.06E-08	-
	RING	nn = 2	**10000**	0.00E+00	0.00E+00	2.17E-03	1.23E-02	**1.10E-06**	**76**
			1000	6.02E-08	7.40E-03	6.97E-03	3.92E-02	3.08E-05	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	6.39E-03	3.92E-02	**4.12E-12**	**64**
			1000	9.47E-11	7.40E-03	9.54E-03	3.92E-02	1.26E-05	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	3.07E-04	7.40E-03	**2.58E-06**	**92**
			1000	6.00E-03	4.20E-02	5.62E-02	2.12E-01	1.19E-04	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	2.07E-03	2.71E-02	**3.36E-12**	**84**
			1000	9.39E-06	1.49E-03	7.82E-03	3.60E-02	2.34E-05	-

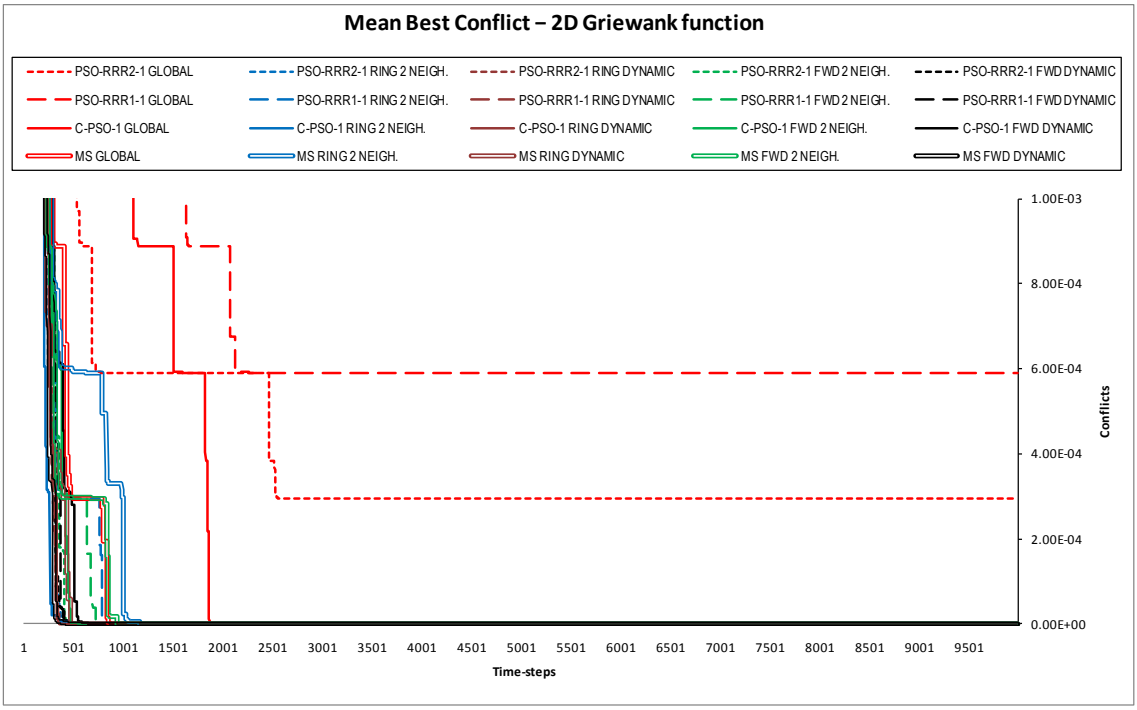


Fig. 7.45. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 7.25. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

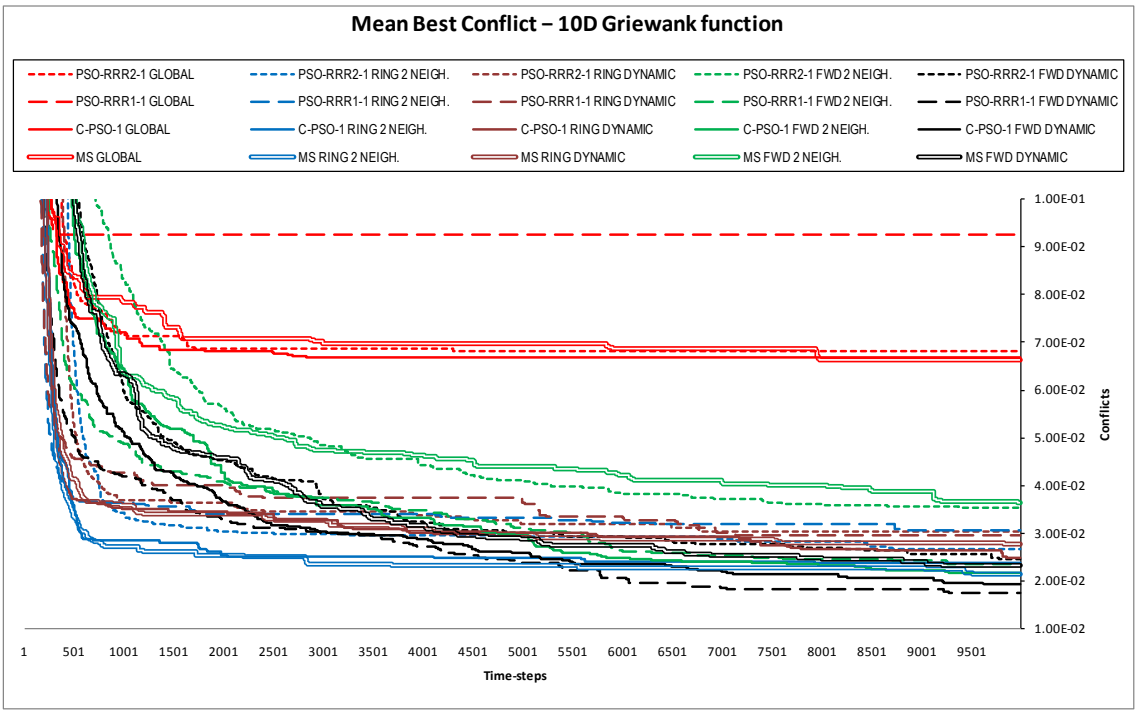


Fig. 7.46. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.26. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

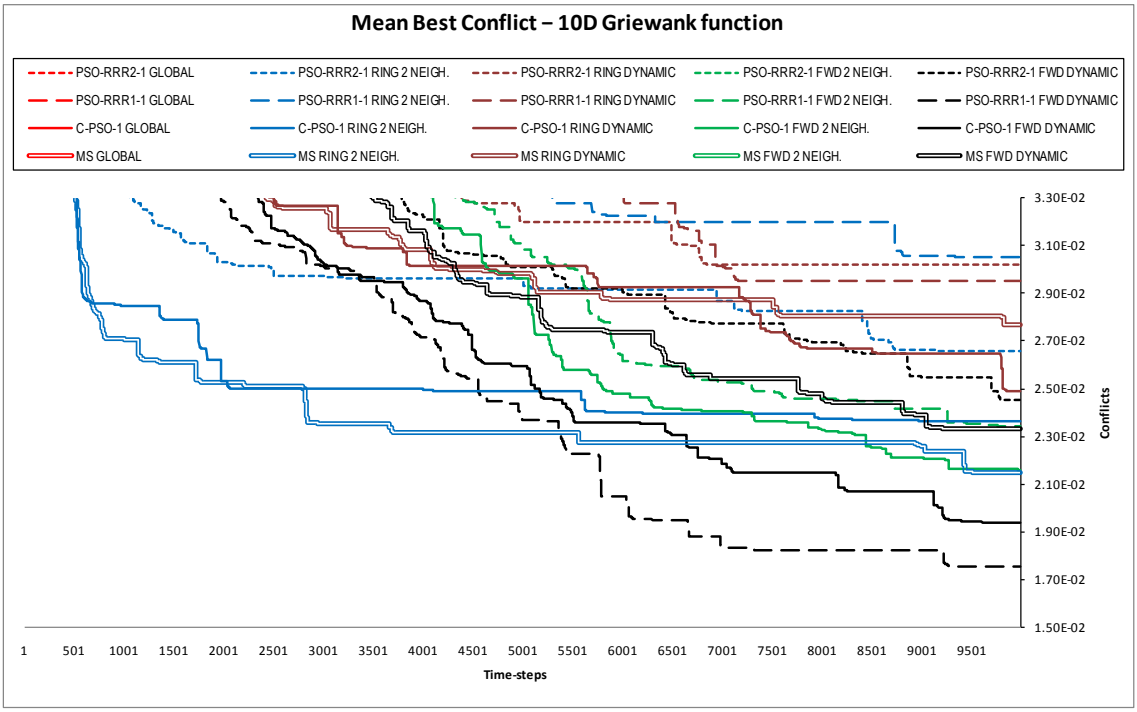


Fig. 7.47. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.26. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

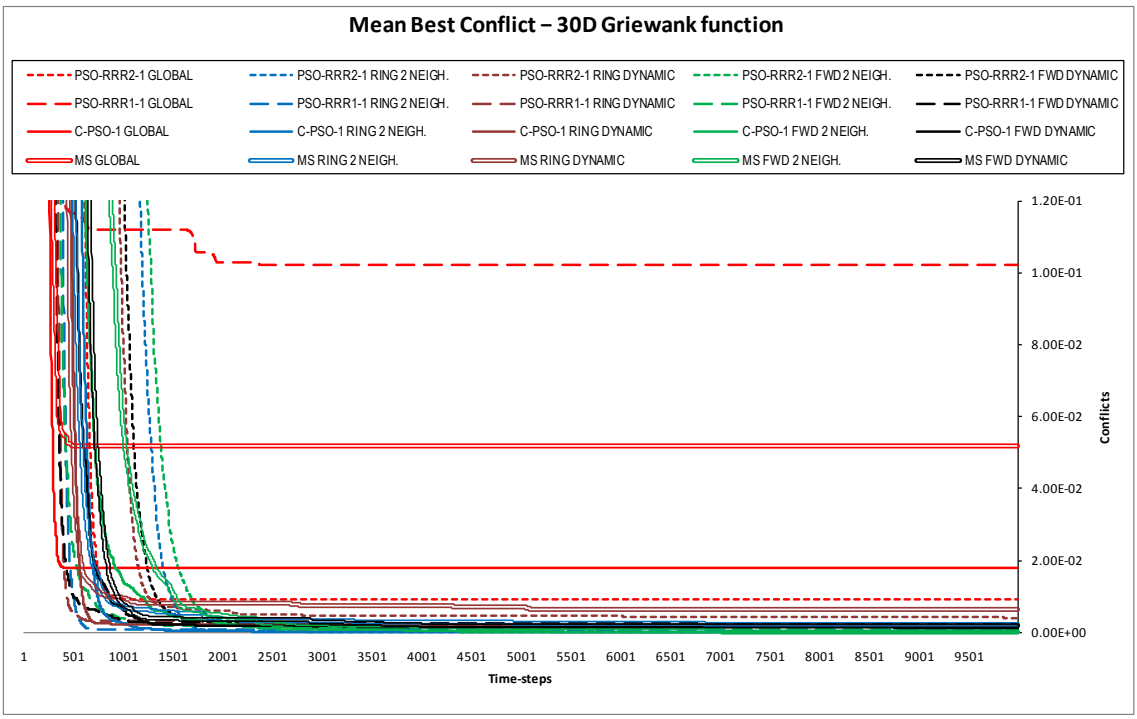


Fig. 7.48. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.27. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

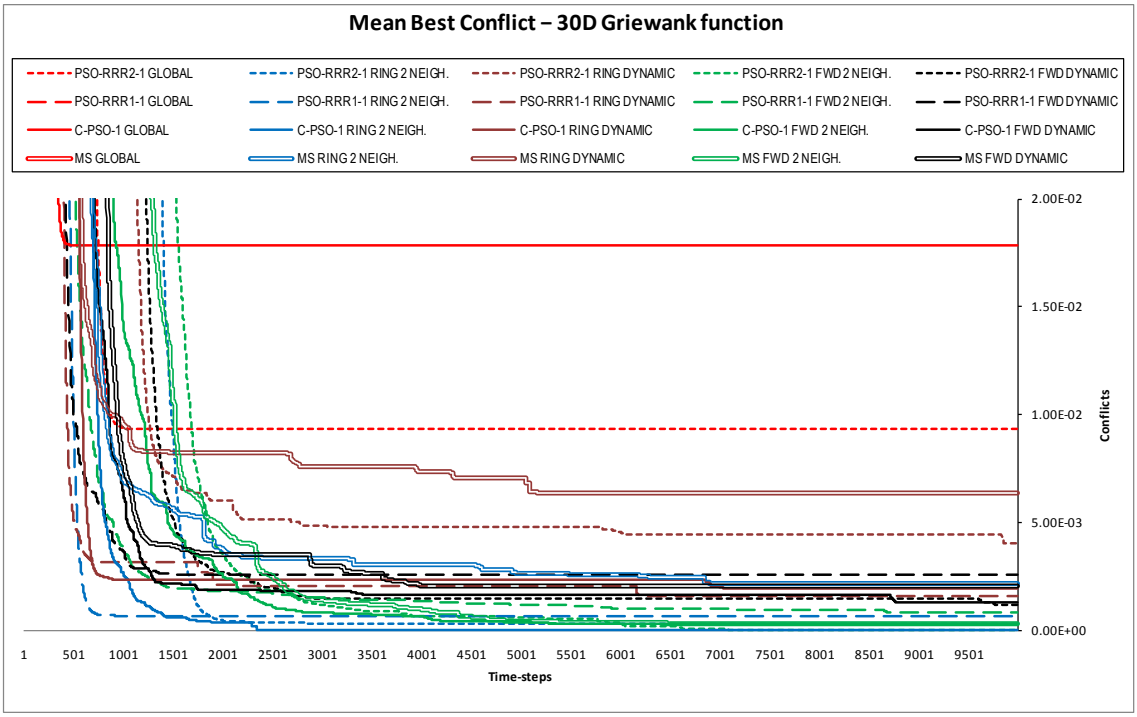


Fig. 7.49. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.27. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

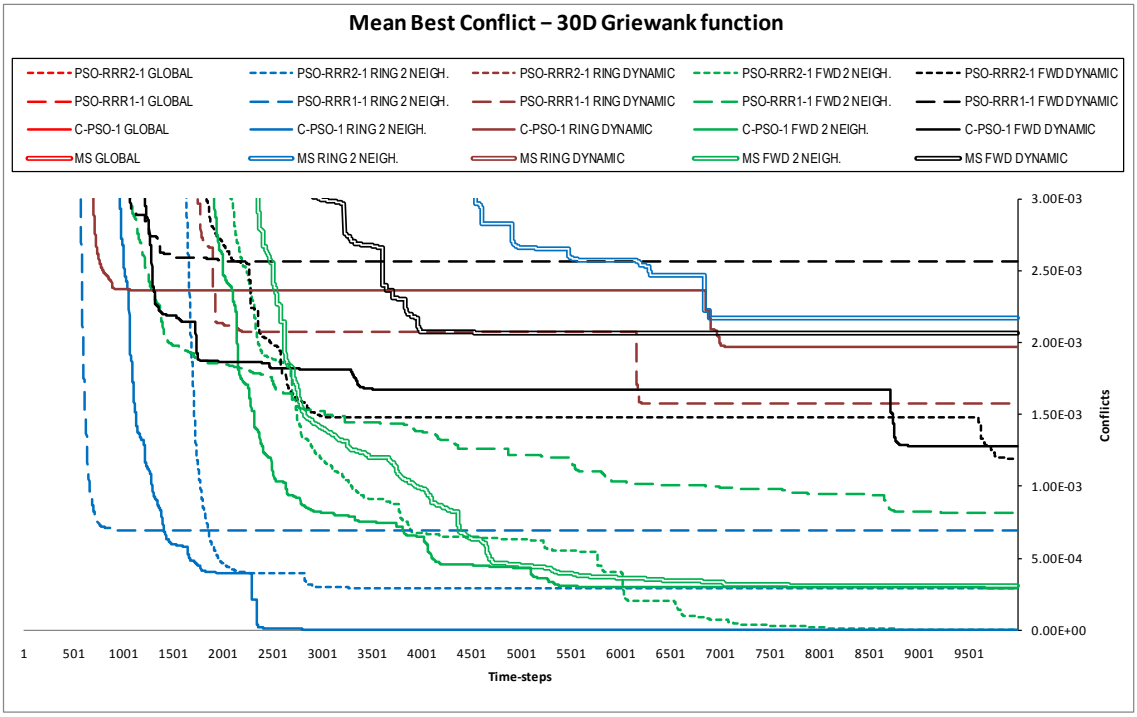


Fig. 7.50. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.27. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.28. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 2-dimensional Schaffer f6 function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**3.89E-04**	**9.72E-03**	**2.58E-05**	**96**
			1000	0.00E+00	0.00E+00	7.77E-04	9.72E-03	2.79E-03	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**3.60E-04**	**100**
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	6.82E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.25E-11**	**100**
			1000	0.00E+00	0.00E+00	5.43E-04	9.72E-03	5.57E-03	-
	FWD	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**2.45E-04**	**100**
			1000	0.00E+00	3.16E-15	3.96E-04	9.72E-03	6.84E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.23E-11**	**100**
			1000	0.00E+00	0.00E+00	3.98E-13	9.16E-12	5.53E-03	-
PSO-RRR1-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**1.17E-03**	**9.72E-03**	**1.13E-04**	**88**
			1000	0.00E+00	0.00E+00	1.17E-03	9.72E-03	1.61E-03	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.29E-03**	**100**
			1000	0.00E+00	0.00E+00	1.96E-03	9.72E-03	7.78E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.26E-11**	**100**
			1000	0.00E+00	0.00E+00	7.03E-07	1.76E-05	5.62E-03	-
	FWD	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**3.01E-05**	**100**
			1000	0.00E+00	5.93E-14	6.42E-04	9.72E-03	6.97E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.12E-11**	**100**
			1000	0.00E+00	0.00E+00	7.53E-12	1.88E-10	5.19E-03	-
C-PSO-1	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**1.17E-03**	**9.72E-03**	**2.01E-04**	**88**
			1000	0.00E+00	0.00E+00	1.95E-03	9.72E-03	2.42E-03	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**3.89E-04**	**9.72E-03**	**1.38E-03**	**96**
			1000	0.00E+00	0.00E+00	2.07E-03	9.72E-03	7.76E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.23E-11**	**100**
			1000	0.00E+00	0.00E+00	7.58E-06	1.21E-04	6.07E-03	-
	FWD	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**3.09E-04**	**100**
			1000	0.00E+00	1.45E-12	1.12E-03	9.72E-03	7.38E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.28E-11**	**100**
			1000	0.00E+00	0.00E+00	1.46E-05	3.63E-04	5.62E-03	-
Multi-Swarm	GLOBAL		**10000**	**0.00E+00**	**0.00E+00**	**2.33E-03**	**9.72E-03**	**1.05E-04**	**76**
			1000	0.00E+00	0.00E+00	3.11E-03	9.72E-03	2.58E-03	-
	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**7.13E-04**	**100**
			1000	0.00E+00	0.00E+00	7.84E-04	9.72E-03	7.34E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.36E-11**	**100**
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	6.25E-03	-
	FWD	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**3.80E-04**	**100**
			1000	0.00E+00	5.94E-14	8.08E-04	9.72E-03	6.89E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.13E-11**	**100**
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	5.17E-03	-

Table 7.29. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 10-dimensional Schaffer f6 function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topology are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	9.72E-03	9.72E-03	1.85E-02	3.72E-02	5.10E-04	0
			1000	9.72E-03	9.72E-03	2.18E-02	3.72E-02	1.21E-03	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.94E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	3.24E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.65E-03	0
			1000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	2.13E-03	-
	FWD	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.78E-03	0
			1000	9.72E-03	9.72E-03	1.41E-02	3.72E-02	2.87E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	2.01E-03	0
			1000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	2.16E-03	-
PSO-RRR1-1	GLOBAL		10000	9.72E-03	3.72E-02	3.45E-02	7.82E-02	1.29E-04	0
			1000	9.72E-03	3.72E-02	3.45E-02	7.82E-02	1.29E-04	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	2.18E-02	3.72E-02	2.43E-03	0
			1000	9.72E-03	3.72E-02	3.17E-02	3.72E-02	3.38E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.39E-03	0
			1000	9.72E-03	9.72E-03	1.74E-02	3.72E-02	2.35E-03	-
	FWD	nn = 2	10000	9.72E-03	9.72E-03	1.85E-02	3.72E-02	1.85E-03	0
			1000	9.72E-03	3.72E-02	2.40E-02	3.72E-02	2.80E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	1.05E-03	0
			1000	9.72E-03	3.72E-02	2.40E-02	3.72E-02	2.14E-03	-
C-PSO-1	GLOBAL		10000	9.72E-03	9.72E-03	1.96E-02	3.72E-02	3.01E-04	0
			1000	9.72E-03	9.72E-03	2.18E-02	3.72E-02	7.06E-04	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	2.21E-03	0
			1000	9.72E-03	9.72E-03	1.85E-02	3.72E-02	3.19E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.92E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	2.11E-03	-
	FWD	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.81E-03	0
			1000	9.72E-03	9.72E-03	1.30E-02	3.72E-02	2.81E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.79E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	2.02E-03	-
Multi-Swarm	GLOBAL		10000	9.72E-03	3.72E-02	2.95E-02	3.72E-02	3.18E-04	0
			1000	9.72E-03	3.72E-02	3.06E-02	3.72E-02	6.67E-04	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	2.03E-03	0
			1000	9.72E-03	9.72E-03	1.74E-02	3.72E-02	3.03E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.76E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	1.95E-03	-
	FWD	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.70E-03	0
			1000	9.72E-03	9.72E-03	1.41E-02	3.72E-02	2.41E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.59E-03	0
			1000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	1.79E-03	-

Table 7.30. Statistical results out of 25 runs for the PSO-RRR2-1, the PSO-RRR1-1, the C-PSO-1, and a Multi-Swarm algorithm optimizing the 30-dimensional Schaffer f6 function. The neighbourhoods tested are the FOWARD topology with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). The results for the GLOBAL and RING topologies are imported from the previous section for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
PSO-RRR2-1	GLOBAL		10000	3.72E-02	7.82E-02	9.22E-02	1.27E-01	3.12E-04	0
			1000	7.82E-02	1.27E-01	1.08E-01	1.78E-01	8.50E-04	-
	RING	nn = 2	10000	3.72E-02	7.82E-02	6.18E-02	7.82E-02	1.77E-03	0
			1000	1.27E-01	1.96E-01	2.01E-01	2.29E-01	4.40E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	3.72E-02	3.72E-02	7.48E-04	0
			1000	7.82E-02	1.27E-01	1.26E-01	2.04E-01	2.76E-03	-
	FWD	nn = 2	10000	3.72E-02	7.82E-02	6.18E-02	7.82E-02	1.57E-03	0
			1000	1.27E-01	1.78E-01	1.82E-01	2.28E-01	3.86E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.54E-02	7.82E-02	6.99E-04	0
			1000	7.82E-02	1.27E-01	1.21E-01	1.55E-01	2.86E-03	-
PSO-RRR1-1	GLOBAL		10000	3.12E-01	4.30E-01	4.25E-01	4.85E-01	4.76E-05	0
			1000	3.12E-01	4.30E-01	4.26E-01	4.87E-01	2.01E-04	-
	RING	nn = 2	10000	7.82E-02	1.78E-01	1.67E-01	2.73E-01	1.72E-03	0
			1000	1.27E-01	2.28E-01	2.14E-01	3.12E-01	2.38E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	1.27E-01	1.12E-01	2.28E-01	6.68E-04	0
			1000	7.82E-02	1.27E-01	1.49E-01	2.73E-01	1.23E-03	-
	FWD	nn = 2	10000	7.82E-02	1.78E-01	1.58E-01	2.28E-01	1.59E-03	0
			1000	1.27E-01	1.78E-01	1.72E-01	2.73E-01	1.70E-03	-
		nni = 2 nnf = (m – 1)	10000	7.82E-02	1.27E-01	1.20E-01	2.28E-01	6.60E-04	0
			1000	7.82E-02	1.27E-01	1.55E-01	2.73E-01	1.14E-03	-
C-PSO-1	GLOBAL		10000	7.82E-02	1.27E-01	1.31E-01	2.73E-01	1.67E-04	0
			1000	7.82E-02	1.27E-01	1.40E-01	2.73E-01	4.94E-04	-
	RING	nn = 2	10000	3.72E-02	3.72E-02	5.52E-02	7.82E-02	1.59E-03	0
			1000	1.27E-01	1.27E-01	1.52E-01	1.78E-01	3.23E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	3.72E-02	3.72E-02	6.49E-04	0
			1000	7.82E-02	7.82E-02	9.19E-02	1.27E-01	2.00E-03	-
	FWD	nn = 2	10000	3.72E-02	7.82E-02	5.85E-02	7.82E-02	1.55E-03	0
			1000	7.82E-02	1.27E-01	1.22E-01	1.78E-01	2.59E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	3.72E-02	4.10E-02	7.82E-02	1.06E-03	0
			1000	4.78E-02	7.82E-02	8.09E-02	1.27E-01	1.87E-03	-
Multi-Swarm	GLOBAL		10000	7.82E-02	1.78E-01	1.86E-01	2.73E-01	2.30E-04	0
			1000	1.27E-01	1.78E-01	1.93E-01	2.73E-01	5.26E-04	-
	RING	nn = 2	10000	3.72E-02	7.82E-02	7.45E-02	1.27E-01	1.58E-03	0
			1000	1.27E-01	1.78E-01	1.68E-01	2.28E-01	3.42E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.38E-02	7.82E-02	5.48E-04	0
			1000	3.74E-02	1.27E-01	1.08E-01	1.78E-01	1.89E-03	-
	FWD	nn = 2	10000	3.72E-02	7.82E-02	7.29E-02	1.27E-01	1.70E-03	0
			1000	7.82E-02	1.27E-01	1.44E-01	1.78E-01	2.83E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.38E-02	7.82E-02	6.55E-04	0
			1000	3.72E-02	7.82E-02	9.05E-02	1.27E-01	1.79E-03	-

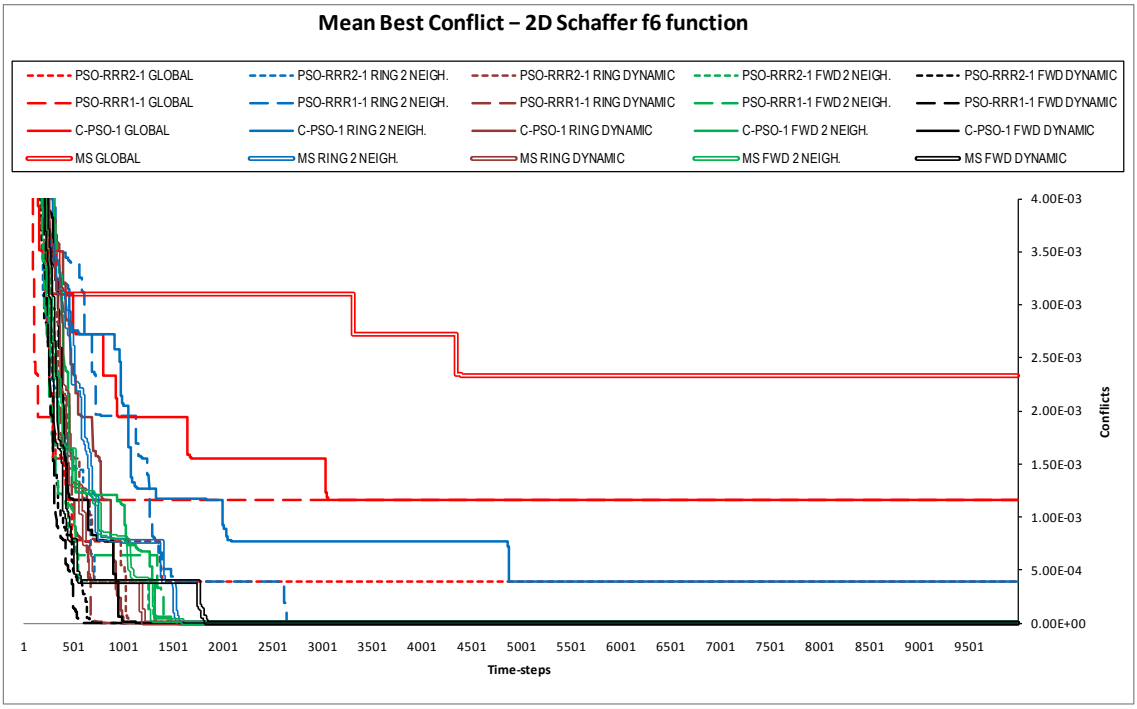


Fig. 7.51. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 7.28. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

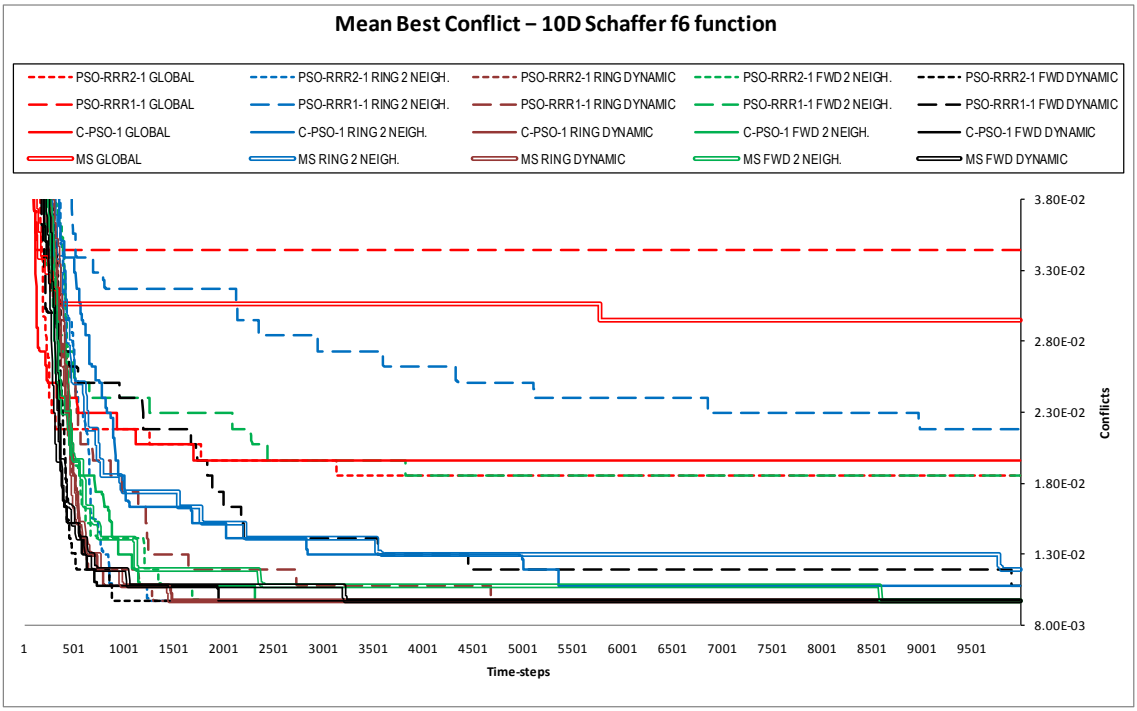


Fig. 7.52. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.29. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

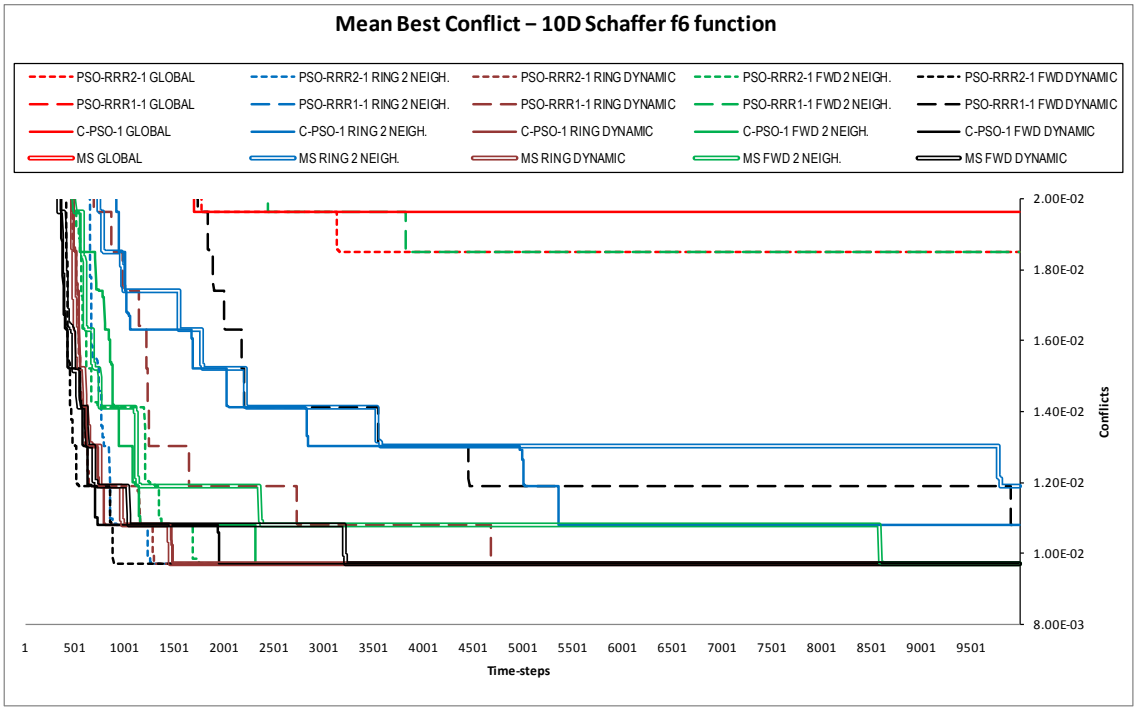


Fig. 7.53. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.29. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

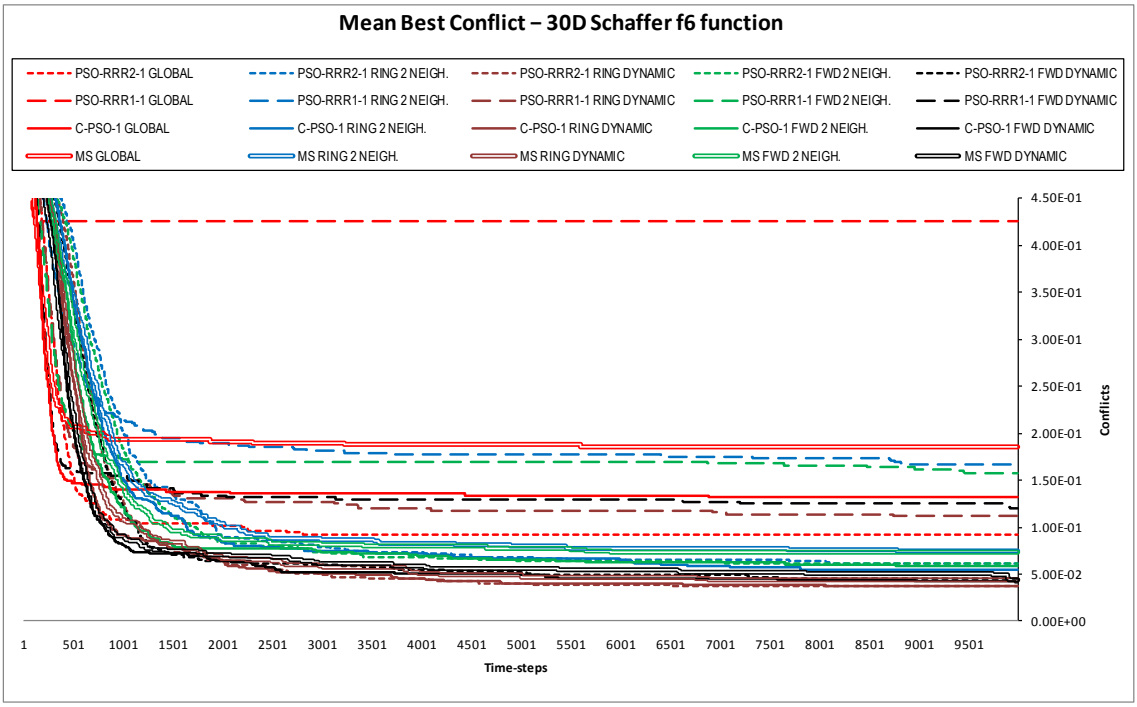


Fig. 7.54. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.30. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

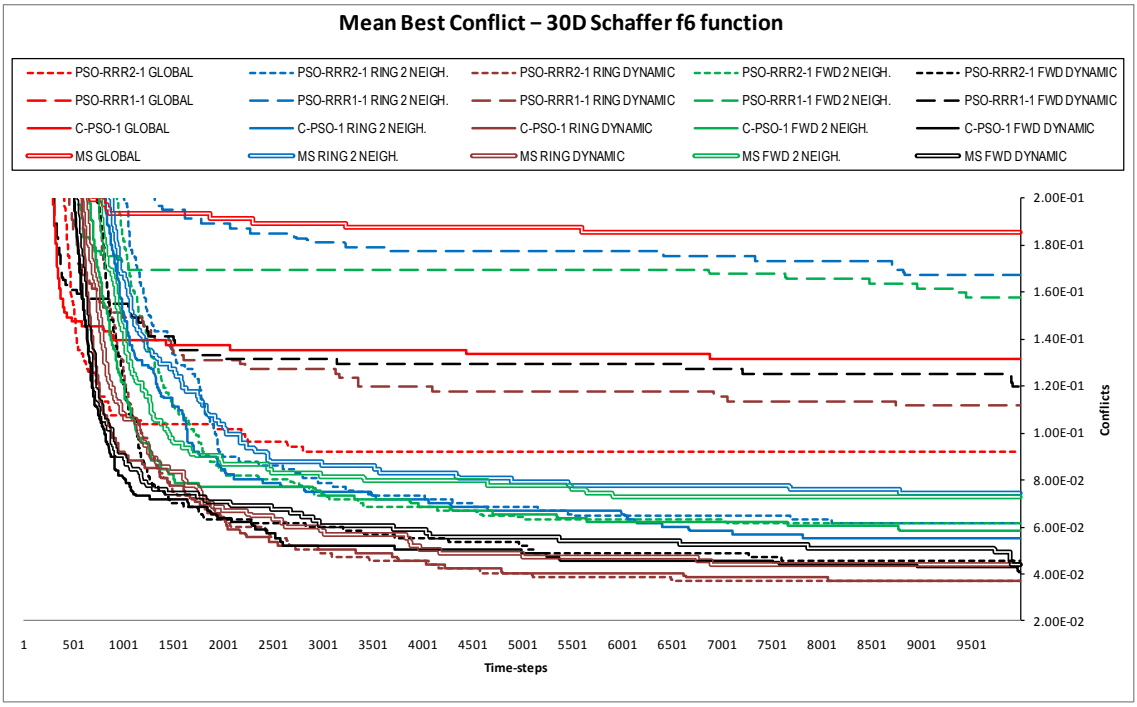


Fig. 7.55. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.30. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Discussion

The settings of the experiments are the same as those described in the previous chapter (see section 6.3.2.2.) unless specifically stated otherwise. Note that although the global neighbourhood is included here, the analyses focus on the comparison between the ring and the forward topologies, especially their dynamic versions.

Sphere

In the 2D problem, every algorithm finds the exact solution for every run. It can also be observed that the implosion of the particles is virtually complete in every case by the end of the search, while the PSO-RRR1 approach exhibits the highest degree of clustering by the 1,000th time-step (see pb_me in Table 7.16).

In the 10D problem, every algorithm achieves a success rate (SR) of 100%. In fact, they all meet the success criterion by the 1,000th time-step. The 'Ring Dynamic' and the 'Fwd Dynamic' topologies find the exact solution and perform a complete implosion for all coefficients' settings except for the more robust PSO-RRR2-1 (see Table 7.17).

In the 30D problem, every algorithm meets the success criterion by the end of the search. Considering only the 'Ring Dynamic' and the 'Fwd Dynamic' topologies, the success criterion is met by the 1,000th time-steps for all coefficients' settings except for the more robust PSO-RRR2-1 (see Table 7.18).

Rosenbrock

In the 2D problem, every algorithm finds the exact solution for every run, already meeting the success criterion by the 1,000th time-step. The implosion is virtually complete in every case by the end of the search, while the PSO-RRR1 approach exhibits the highest degree of clustering by the 1,000th time-step (see pb_me in Table 7.19).

In the 10D problem, it appears that the forward topology leads to slower convergence than the ring topology. The PSO-RRR2-1, C-PSO-1 and MS with 'Fwd *nn*=2' topology find poor results by the end of the search. Better results are found by the PSO-RRR1-1, which favours faster convergence (see Fig. 7.34). As to the 'Fwd Dynamic' topology, it returns very good results if coupled with the PSO-RRR1-1 and with the C-PSO-1. Coupling it with the robust PSO-RRR2-1 does not return good results because it leads to very slow a convergence. The mean solution of the 'MS Fwd Dynamic' is a bit misleading, as it achieves a SR of 76% and a good median solution (see Table 7.20).

In the 30D problem, the trend is the same. The 'Fwd *nn*=2' topology coupled with any of the coefficients' settings and the 'PSO-RRR2-1 Fwd Dynamic' return poor results. In contrast, the 'PSO-RRR1-1 Fwd Dynamic', the 'C-PSO-1 Fwd Dynamic', and the 'MS Fwd Dynamic' find some of the best results. Note that the latter two do not achieve convergence by the end of the search, and further improvement is to be expected for an extended search-length (see Table 7.21, Fig. 7.37, and Fig. 7.38).

The performances of the 'Ring Dynamic' and the 'Fwd Dynamic' topologies are very similar to each other for the PSO-RRR1-1, the C-PSO-1, and the MS coefficients.

Rastrigin

In the 2D problem, every algorithm finds the exact solution by the end of the search. In fact, they do so by the 1,000th time-step, except for the 'MS Ring *nn*=2'.

In the 10D problem, the best performances are clearly exhibited by all the 'Ring Dynamic' and the 'Fwd Dynamic' topologies, except for the 'PSO-RRR1-1 Ring Dynamic' (see Fig. 7.41 and Fig. 7.42). Comparing the two dynamic topologies for the same coefficients' settings, each 'Fwd Dynamic' algorithm outperforms its 'Ring Dynamic' counterpart. The very best performance is exhibited by the 'PSO-RRR2-1 Fwd Dynamic', followed by the 'C-PSO-1 Fwd Dynamic', the 'PSO-RRR2-1 Ring Dynamic', and the 'MS Fwd Dynamic' (see Table 7.23 and Fig. 7.40 to Fig. 7.42).

In the 30D problem, the best performances are exhibited by the 'Fwd Dynamic' topology by a considerable margin. This seems to confirm that the forward topology tends to maintain diversity for longer than the ring topology for the same neighbourhood-size. Thus, the very best results are again found by the 'PSO-RRR2-1 Fwd Dynamic', now followed by the 'MS Fwd Dynamic', the 'C-PSO-1 Fwd Dynamic', and the 'PSO-RRR1-1 Fwd Dynamic' (see Table 7.24, Fig. 7.43, and Fig. 7.44).

Griewank

In the 2D problem, all the non-global topologies find the exact solution in every run.

In the 10D problem, all the algorithms with 'Fwd Dynamic' topology outperform all those with 'Ring Dynamic' topology (see Table 7.26 and Fig. 7.47). The best performance is shown by the 'PSO-RRR1-1 Fwd Dynamic', followed by the 'C-PSO-1 Fwd Dynamic', the 'MS Ring *nn*=2', the 'C-PSO-1 Fwd *nn*=2, and the 'MS Fwd Dynamic'.

In the 30D problem, the best performances are obtained by the 'Ring *nn*=2' and the 'Fwd *nn*=2' topologies. Considering only the dynamic ones, the best performances are shown by the 'PSO-RRR2-1 Fwd Dynamic', followed by the 'C-PSO-1 Fwd Dynamic', the 'PSO-RRR1-1 Ring Dynamic', the 'C-PSO-1 Ring Dynamic', and the 'MS Fwd Dynamic'. The results are in Table 7.27, Fig. 7.49, and Fig. 7.50. Note that although the fixed ring and forward topologies perform better in this particular problem, the aforementioned dynamic topologies obtained success rates of 80% or more.

Schaffer f6

In the 2D problem, only the global topologies and the 'C-PSO-1 Ring *nn*=2' exhibit premature convergence. The other algorithms find the exact solution in every run.

In the 10D problem, the best performances are exhibited by the 'Ring Dynamic' and the 'Fwd Dynamic' topologies, except for the 'PSO-RRR1-1 Fwd Dynamic'.

In the 30D problem, the best performances are exhibited by the 'Ring Dynamic' topologies (except for the 'PSO-RRR1-1 Ring Dynamic'), followed by the 'Fwd Dynamic' ones (except for the 'PSO-RRR1-1 Fwd Dynamic'). Refer to Table 7.30 and Fig. 7.55.

Overall analyses

The proposed 'Forward Dynamic' topology seems promising according to the results from the experiments carried out in this section.

Broadly speaking, it is marginally outperformed by the 'Ring Dynamic' topology in the 10D Rosenbrock function (Fig. 7.35) and in the 30D Schaffer f6 function (Fig. 7.55). Conversely, it outperforms the 'Ring Dynamic' topology in the 10D and 30D Rastrigin function (Fig. 7.42 and Fig. 7.44), and in the 10D and 30D Griewank function (Fig. 7.47, Fig. 7.49, and Fig. 7.50). They are competitive in the remaining problems. Regarding the coefficients' settings, the Multi-Swarm approach (MS) appears to be the most stable, as it does reasonably well on all problems tested. It seems that the 'MS Fwd Dynamic' is the algorithm that performs the best as a general-purpose optimizer.

7.4. Nearest neighbour

The overlapping of the ring topology allows the spread of information throughout the whole swarm at every instance of the neighbourhood-size. That is to say, every particle is connected to every other either directly or indirectly.

Aiming to keep this feature while also introducing information on the actual distance between particles in the 'physical' space so that nearer particles have more chances to become neighbours, a nearest neighbour procedure is proposed. The latter consists of reordering the list of particles: the first particle is kept, and the remaining particles are re-ordered so that the ith particle is the one that is nearest to the $(i-1)^{th}$ amongst the remaining (swarm-size $- i + 1$) particles. That is, the well-known 'nearest neighbour' algorithm for approximately solving a travelling salesman problem (TSP). A strict nearest neighbour strategy would not guarantee an overlapping between neighbourhoods. The

distances used for the nearest neighbour re-ordering are normalized to the corresponding feasible interval. The normalization is illustrated in Fig. 7.56.

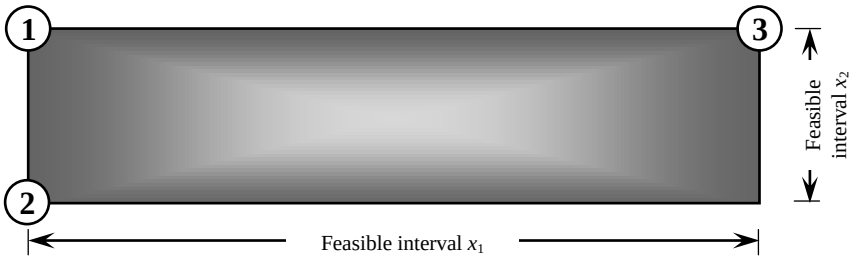


Fig. 7.56. Two-dimensional search-space with feasible intervals of different size; three particles are allocated to illustrate the normalization of the distances for the nearest neighbour algorithm: particles '2' and '3' are at the same normalized distance from particle '1'.

This heuristics can be applied to the particles' positions (NN), or to the particles' best experiences (NNB). This optimizer allows the user to choose between the two.

In order to illustrate how this technique takes advantage of the proximity in the physical space while maintaining the neighbourhood topologies discussed before, the technique is applied to a swarm of particles, which is then simply split in three groups. Fig. 7.57 shows that the particles in the group are reasonably close to each other.

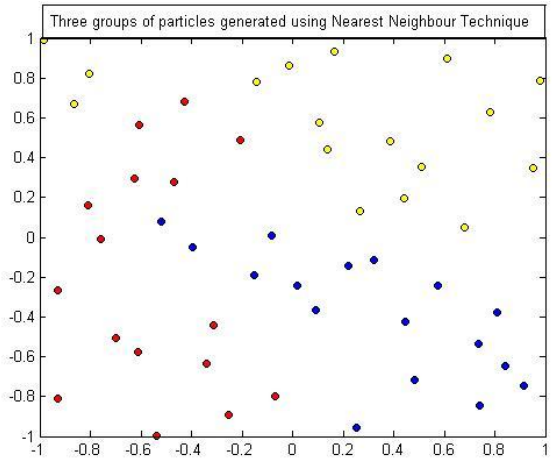


Fig. 7.57. Example of generating three groups of particles by means of the Nearest Neighbour heuristics.

The NNB technique is applied to the Multi-Swarm algorithm with ring and forward topologies, each with *nn*=2 and dynamic. The results obtained without the technique are imported from previous sections for comparison. The experimental results are presented in Table 7.31 to Table 7.45, and Fig. 7.58 to Fig. 7.72.

Table 7.31. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Sphere function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	3.39E-85	4.26E-78	9.85E-77	1.83E-75	2.68E-20	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	2.99E-86	1.12E-80	2.13E-79	2.33E-78	4.47E-20	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	1.24E-75	8.38E-73	4.72E-69	1.16E-67	9.31E-20	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	2.17E-79	3.23E-74	3.17E-70	7.36E-69	8.94E-22	-
	RING NNB	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	7.16E-81	5.09E-75	4.18E-73	3.98E-72	2.55E-20	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	7.65E-89	3.19E-84	1.06E-82	8.95E-82	4.66E-20	-
	FWD NNB	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	1.11E-69	4.70E-65	6.80E-62	9.70E-61	2.27E-21	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	6.75E-84	8.27E-80	1.30E-76	2.83E-75	3.80E-20	-

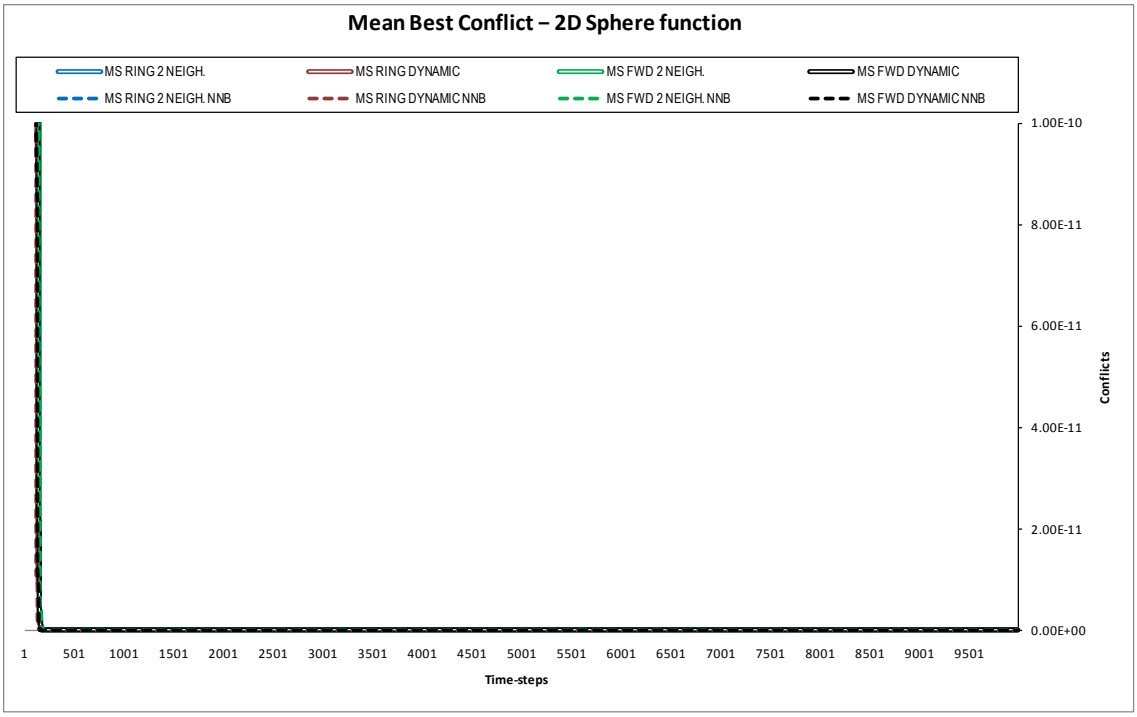


Fig. 7.58. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 7.31. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.32. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Sphere function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.20E-153	100
			1000	1.41E-32	1.17E-30	1.06E-29	9.91E-29	2.68E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.18E-43	2.00E-39	5.49E-38	1.09E-36	1.35E-14	-
	FWD	nn = 2	10000	6.03E-186	1.07E-182	4.73E-180	1.07E-178	2.39E-93	100
			1000	4.92E-18	6.77E-17	1.20E-16	5.25E-16	7.03E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.97E-23	1.46E-22	3.15E-22	1.75E-21	1.37E-13	-
	RING NNB	nn = 2	10000	1.82E-260	4.77E-222	3.85E-212	9.58E-211	7.01E-109	100
			1000	3.02E-24	2.16E-20	3.22E-19	2.68E-18	2.85E-12	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.61E-36	3.81E-34	2.09E-32	4.39E-31	1.72E-10	-
	FWD NNB	nn = 2	10000	2.63E-233	4.12E-226	9.01E-215	1.45E-213	5.02E-111	100
			1000	1.33E-21	1.42E-20	1.08E-19	1.88E-18	4.83E-13	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.44E-135	100
			1000	1.10E-27	3.95E-17	3.09E-14	7.21E-13	3.39E-07	-

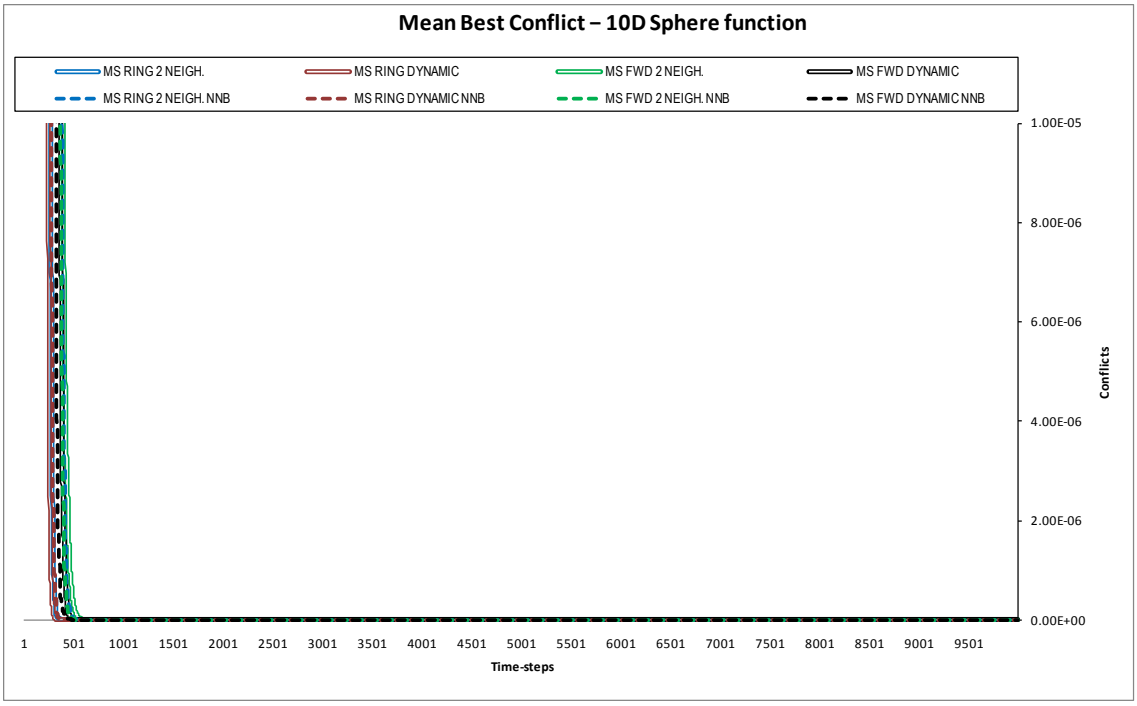


Fig. 7.59. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 7.32. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.33. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Sphere function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	10000	3.13E-113	7.02E-109	2.72E-107	5.96E-106	6.93E-57	100
			1000	2.83E-08	9.73E-08	2.55E-07	3.82E-06	1.75E-06	-
		nni = 2 nnf = (m – 1)	10000	1.28E-185	6.46E-180	6.80E-173	1.68E-171	2.58E-91	100
			1000	7.07E-12	1.43E-10	3.51E-10	3.97E-09	1.57E-08	-
	FWD	nn = 2	10000	9.63E-63	2.09E-61	1.04E-60	6.46E-60	3.03E-34	100
			1000	4.04E-04	1.74E-03	1.79E-03	3.91E-03	1.73E-05	-
		nni = 2 nnf = (m – 1)	10000	1.16E-174	4.86E-169	2.11E-165	2.09E-164	3.76E-87	100
			1000	1.09E-06	4.95E-06	5.29E-06	1.20E-05	7.73E-07	-
	RING NNB	nn = 2	10000	6.03E-80	2.15E-74	4.07E-72	7.20E-71	2.44E-40	100
			1000	1.13E-05	2.86E-04	3.29E-04	8.99E-04	6.26E-06	-
		nni = 2 nnf = (m – 1)	10000	2.51E-171	6.64E-166	2.04E-158	3.84E-157	5.35E-84	100
			1000	1.07E-09	1.06E-08	7.90E-08	5.10E-07	1.41E-05	-
	FWD NNB	nn = 2	10000	8.69E-89	2.95E-85	8.20E-81	1.47E-79	8.38E-45	100
			1000	4.22E-06	1.75E-05	2.60E-05	1.08E-04	1.38E-06	-
		nni = 2 nnf = (m – 1)	10000	1.29E-133	7.59E-128	2.06E-125	2.68E-124	4.66E-67	100
			1000	2.73E-05	1.12E-03	1.78E-03	7.83E-03	1.75E-04	-

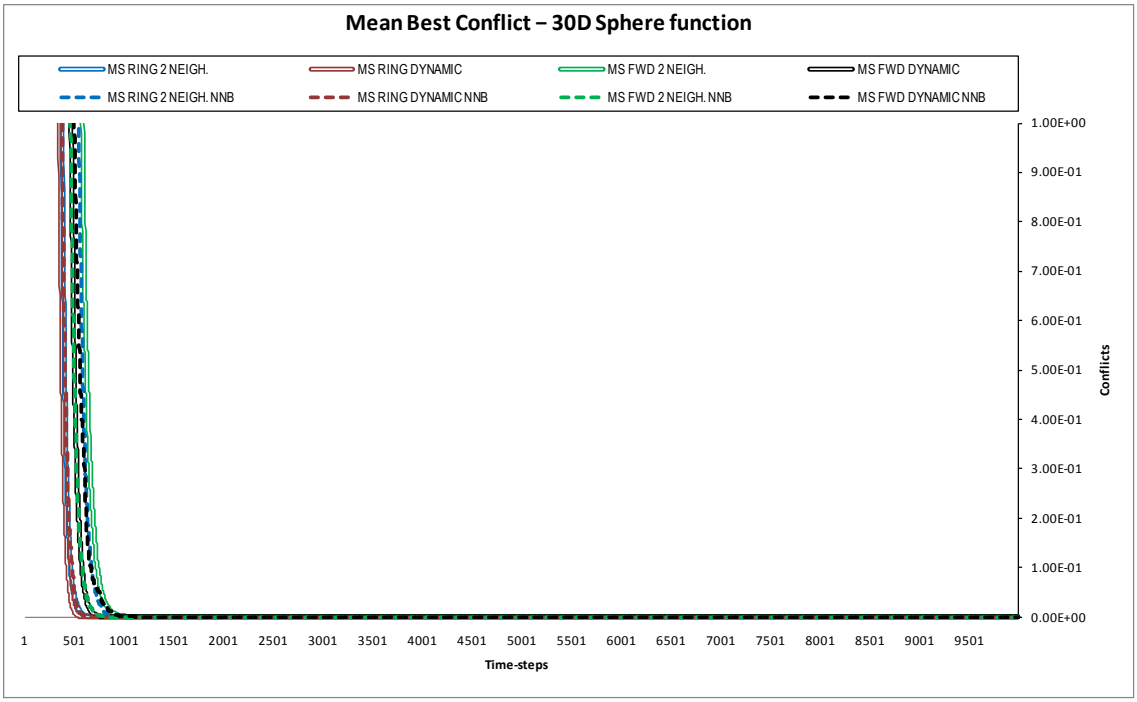


Fig. 7.60. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.33. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.34. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Rosenbrock function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	0.00E+00	9.00E-24	2.19E-18	3.50E-17	1.78E-06	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	0.00E+00	1.77E-30	1.67E-26	3.95E-25	1.36E-06	-
	FWD	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	2.90E-22	1.03E-16	4.09E-14	5.40E-13	7.85E-05	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	4.34E-26	1.05E-21	5.04E-19	1.24E-17	1.10E-07	-
	RING NNB	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	3.83E-24	3.41E-19	2.47E-16	4.47E-15	4.01E-06	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	0.00E+00	1.18E-28	6.54E-25	1.25E-23	1.66E-06	-
	FWD NNB	nn = 2	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	5.85E-21	4.05E-16	1.26E-15	8.21E-15	1.12E-05	-
		nni = 2 nnf = (m – 1)	**10000**	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**100**
			1000	2.94E-24	1.79E-14	1.59E-10	3.31E-09	2.51E-04	-

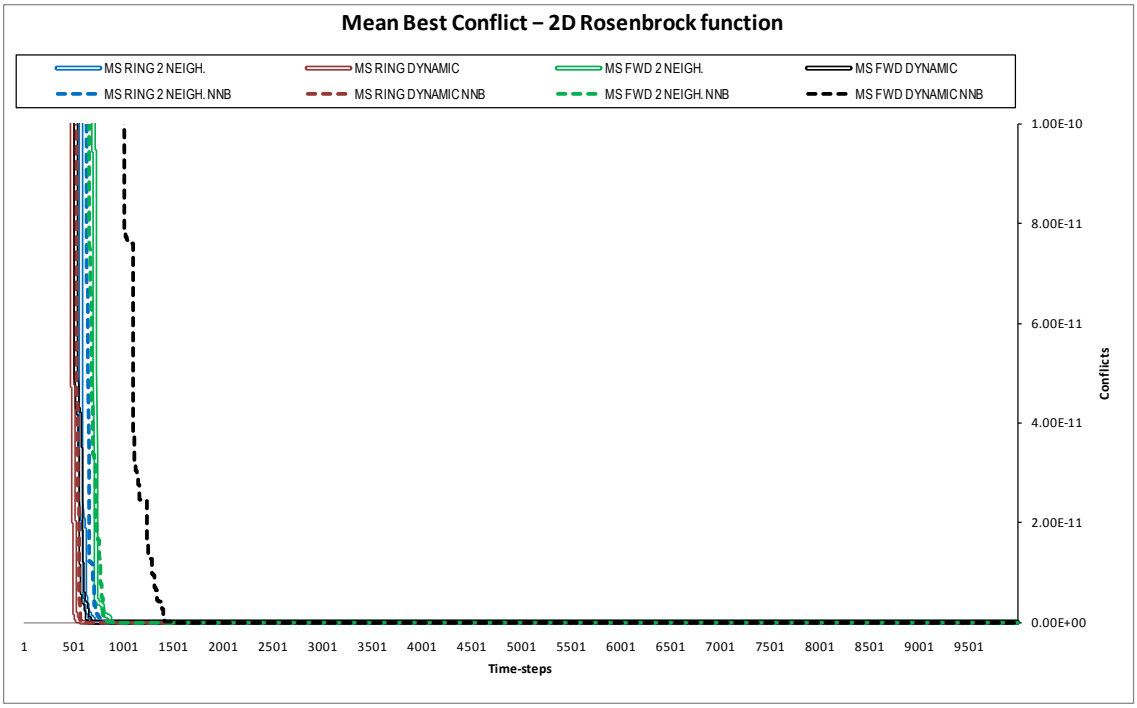


Fig. 7.61. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 7.34. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.35. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Rosenbrock function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	**10000**	**4.01E-09**	**8.03E-07**	**1.61E-01**	**3.99E+00**	**8.87E-03**	**80**
			1000	2.63E-03	1.62E+00	1.82E+00	5.11E+00	1.79E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**1.88E-15**	**1.16E-09**	**1.42E-05**	**3.49E-04**	**1.08E-02**	**96**
			1000	5.39E-04	6.18E-01	8.49E-01	4.07E+00	1.45E-02	-
	FWD	nn = 2	**10000**	**9.26E-03**	**3.47E+00**	**3.47E+00**	**7.03E+00**	**4.92E-03**	**0**
			1000	1.09E-02	5.46E+00	5.35E+00	9.55E+00	9.53E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**3.40E-12**	**4.82E-09**	**4.80E-01**	**4.01E+00**	**4.57E-03**	**76**
			1000	8.74E-01	4.48E+00	4.63E+00	9.59E+00	6.54E-03	-
	RING NNB	nn = 2	**10000**	**1.04E-04**	**1.13E-03**	**2.27E-03**	**1.06E-02**	**6.41E-03**	**0**
			1000	3.67E-02	1.41E+00	1.76E+00	4.70E+00	1.70E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**5.12E-12**	**5.25E-08**	**3.19E-01**	**3.99E+00**	**1.65E-02**	**84**
			1000	6.33E-03	3.31E-01	7.48E-01	4.25E+00	2.31E-02	-
	FWD NNB	nn = 2	**10000**	**1.00E-01**	**2.48E+00**	**3.25E+00**	**7.84E+00**	**8.55E-03**	**0**
			1000	3.96E-01	5.81E+00	5.52E+00	9.19E+00	1.62E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**7.11E-18**	**2.22E-07**	**6.60E-01**	**3.99E+00**	**1.94E-02**	**68**
			1000	1.13E-01	1.88E+00	2.97E+00	7.32E+00	3.38E-02	-

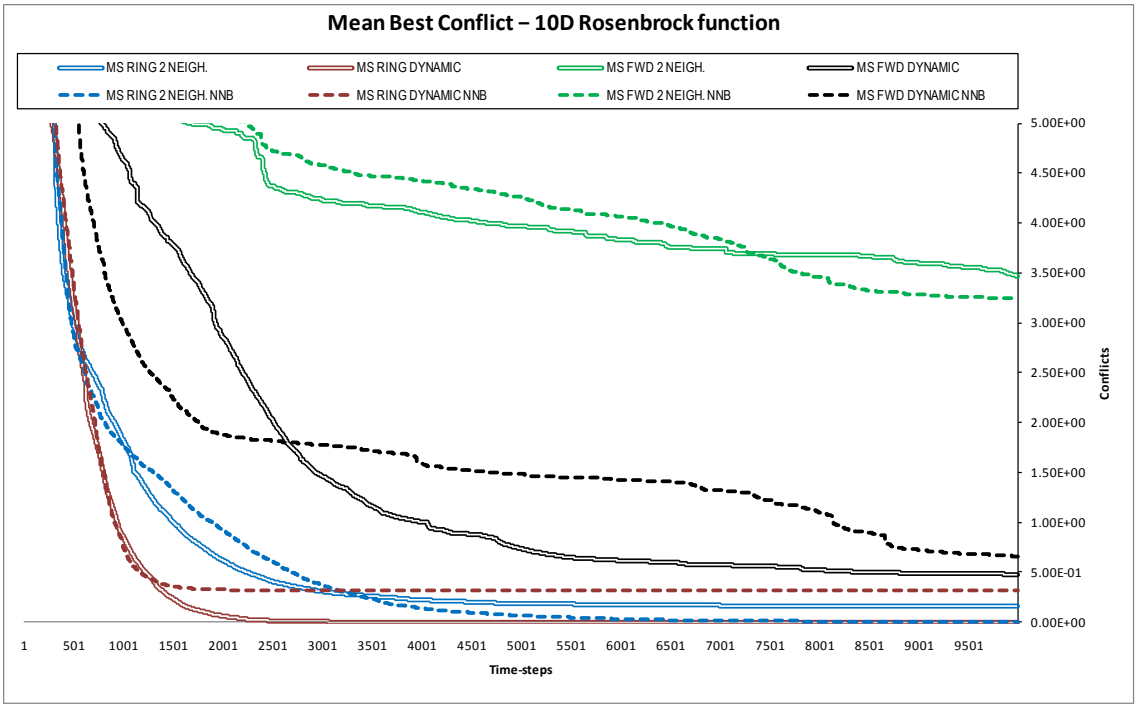


Fig. 7.62. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.35. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.36. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Rosenbrock function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	10000	9.14E-03	7.09E+00	6.59E+00	1.46E+01	3.01E-03	0
			1000	6.26E+00	7.08E+01	5.33E+01	8.71E+01	5.27E-03	-
		nni = 2 nnf = (m – 1)	10000	1.56E-05	5.30E+00	6.36E+00	1.98E+01	2.76E-03	4
			1000	4.72E+00	2.73E+01	4.05E+01	1.34E+02	3.79E-03	-
	FWD	nn = 2	10000	1.95E+01	3.25E+01	5.09E+01	1.77E+02	3.71E-03	0
			1000	3.27E+01	7.61E+01	9.11E+01	2.12E+02	5.89E-03	-
		nni = 2 nnf = (m – 1)	10000	2.04E-04	1.10E+00	4.08E+00	7.00E+01	2.35E-03	0
			1000	2.25E+01	2.71E+01	5.27E+01	2.17E+02	1.53E-03	-
	RING NNB	nn = 2	10000	1.61E-02	4.03E+00	7.39E+00	1.96E+01	7.11E-03	0
			1000	2.72E+01	6.94E+01	6.88E+01	1.60E+02	1.44E-02	-
		nni = 2 nnf = (m – 1)	10000	3.35E-04	4.05E+00	1.31E+01	7.13E+01	6.09E-03	0
			1000	1.41E+01	7.62E+01	6.64E+01	1.36E+02	7.35E-03	-
	FWD NNB	nn = 2	10000	8.32E+00	2.79E+01	4.95E+01	1.52E+02	3.28E-03	0
			1000	2.62E+01	7.41E+01	7.47E+01	1.63E+02	4.59E-03	-
		nni = 2 nnf = (m – 1)	10000	6.69E-05	3.99E+00	1.05E+01	8.14E+01	1.61E-02	4
			1000	1.88E+01	7.94E+01	7.17E+01	1.28E+02	1.64E-02	-

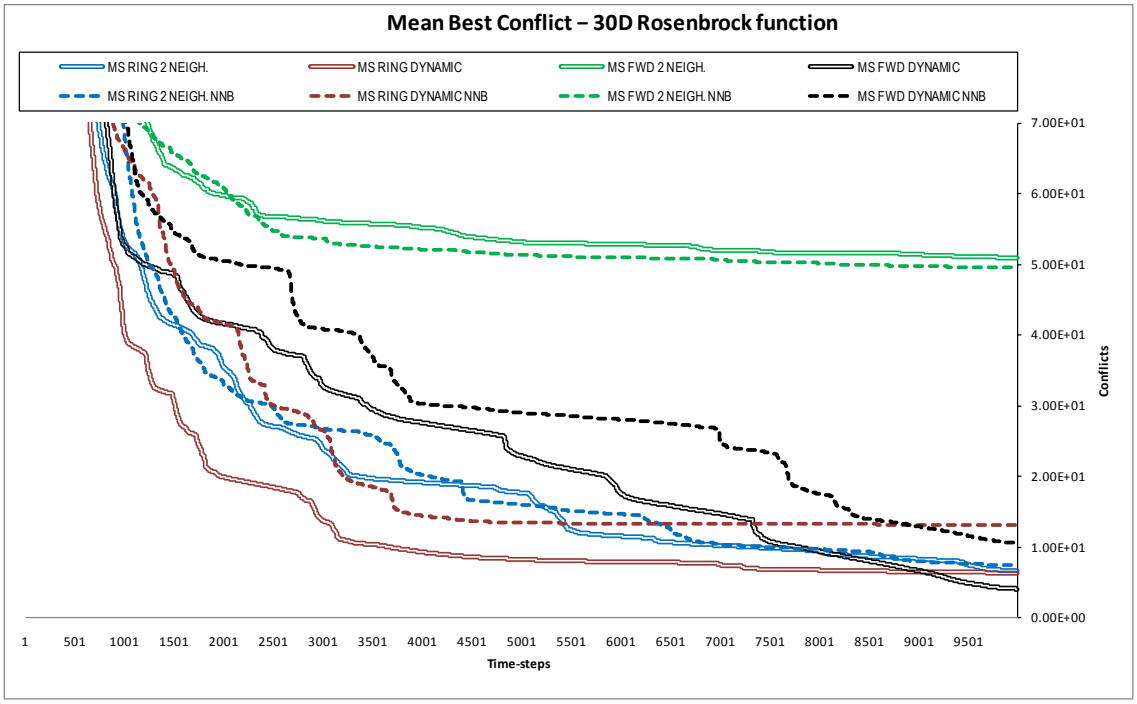


Fig. 7.63. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.36. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.37. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Rastrigin function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.73E-11	100
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.95E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.95E-11	-
	RING NNB	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.59E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.59E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	-
	FWD NNB	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.78E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.78E-11	-

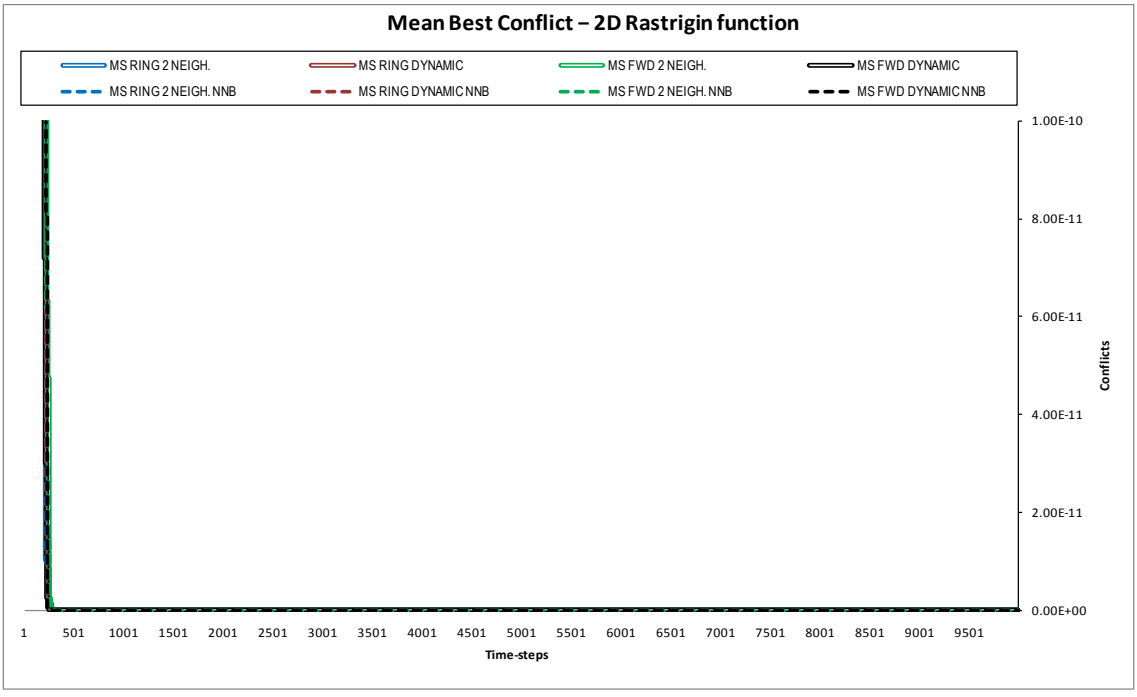


Fig. 7.64. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 7.37. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.38. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Rastrigin function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	**10000**	**0.00E+00**	**2.98E+00**	**2.75E+00**	**6.96E+00**	**2.90E-02**	**4**
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**9.95E-01**	**1.68E+00**	**5.97E+00**	**1.94E-02**	**32**
			1000	2.18E+00	4.97E+00	5.26E+00	1.09E+01	3.66E-02	-
	FWD	nn = 2	**10000**	**6.41E-05**	**2.12E+00**	**2.34E+00**	**6.20E+00**	**2.45E-02**	**4**
			1000	2.80E+00	6.48E+00	6.42E+00	1.03E+01	3.42E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**9.95E-01**	**1.07E+00**	**2.98E+00**	**1.66E-02**	**28**
			1000	2.25E+00	5.07E+00	5.03E+00	9.50E+00	3.34E-02	-
	RING NNB	nn = 2	**10000**	**0.00E+00**	**2.98E+00**	**3.42E+00**	**7.96E+00**	**3.10E-02**	**8**
			1000	2.98E+00	5.97E+00	6.27E+00	1.49E+01	3.94E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**9.95E-01**	**1.43E+00**	**5.97E+00**	**1.81E-02**	**40**
			1000	1.99E+00	4.97E+00	5.68E+00	1.29E+01	4.25E-02	-
	FWD NNB	nn = 2	**10000**	**0.00E+00**	**2.26E+00**	**2.47E+00**	**5.41E+00**	**2.58E-02**	**8**
			1000	2.45E+00	5.20E+00	5.42E+00	9.79E+00	3.63E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**6.37E-01**	**1.99E+00**	**1.37E-02**	**56**
			1000	1.99E+00	6.32E+00	6.03E+00	9.52E+00	3.75E-02	-

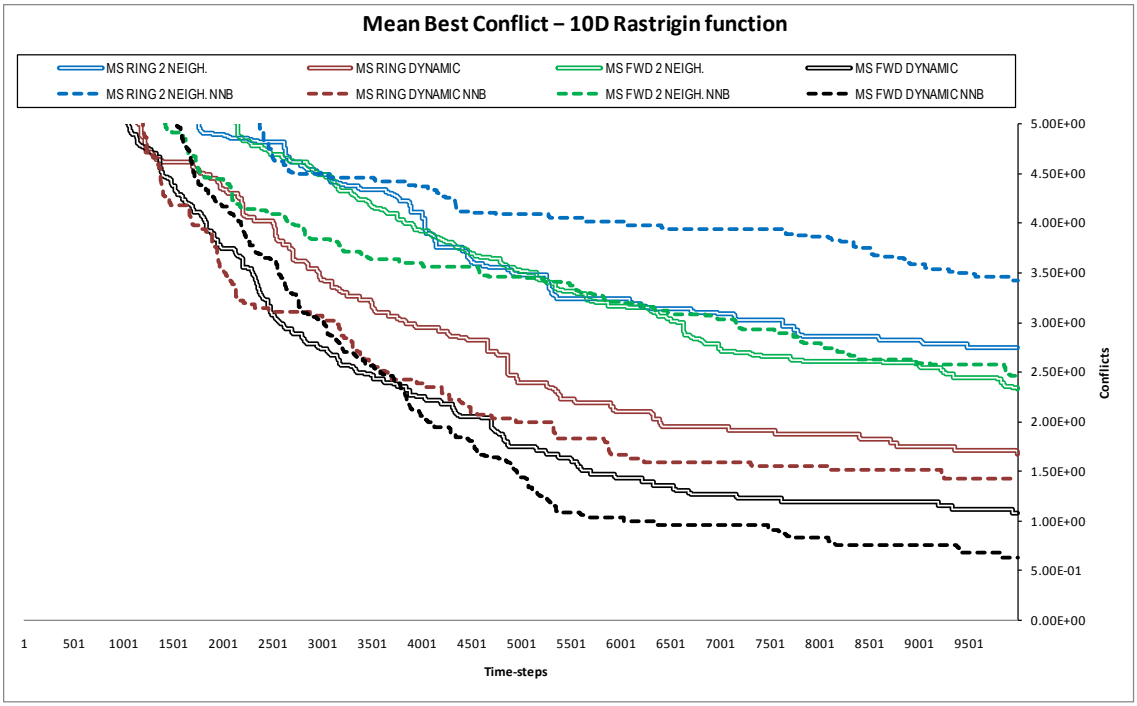


Fig. 7.65. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.38. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.39. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Rastrigin function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	10000	3.28E+01	4.48E+01	4.56E+01	6.37E+01	3.03E-02	0
			1000	3.32E+01	5.21E+01	4.97E+01	6.57E+01	3.09E-02	-
		nni = 2 nnf = (m – 1)	10000	2.59E+01	3.98E+01	4.32E+01	6.67E+01	2.77E-02	0
			1000	2.72E+01	4.48E+01	4.77E+01	6.83E+01	2.96E-02	-
	FWD	nn = 2	10000	2.96E+01	6.81E+01	6.71E+01	8.76E+01	3.07E-02	0
			1000	4.94E+01	8.76E+01	8.55E+01	1.21E+02	3.31E-02	-
		nni = 2 nnf = (m – 1)	10000	1.69E+01	3.38E+01	3.34E+01	5.67E+01	2.18E-02	0
			1000	3.73E+01	7.11E+01	7.11E+01	9.42E+01	3.10E-02	-
	RING NNB	nn = 2	10000	3.48E+01	5.27E+01	5.39E+01	7.36E+01	3.62E-02	0
			1000	4.66E+01	5.90E+01	6.11E+01	8.16E+01	3.67E-02	-
		nni = 2 nnf = (m – 1)	10000	1.89E+01	4.28E+01	4.40E+01	6.87E+01	2.85E-02	0
			1000	2.49E+01	4.88E+01	4.99E+01	7.60E+01	2.85E-02	-
	FWD NNB	nn = 2	10000	4.45E+01	6.32E+01	6.49E+01	9.70E+01	3.20E-02	0
			1000	5.76E+01	8.46E+01	8.65E+01	1.20E+02	3.30E-02	-
		nni = 2 nnf = (m – 1)	10000	1.39E+01	2.98E+01	3.13E+01	5.47E+01	2.31E-02	0
			1000	3.38E+01	5.61E+01	5.65E+01	8.22E+01	2.83E-02	-

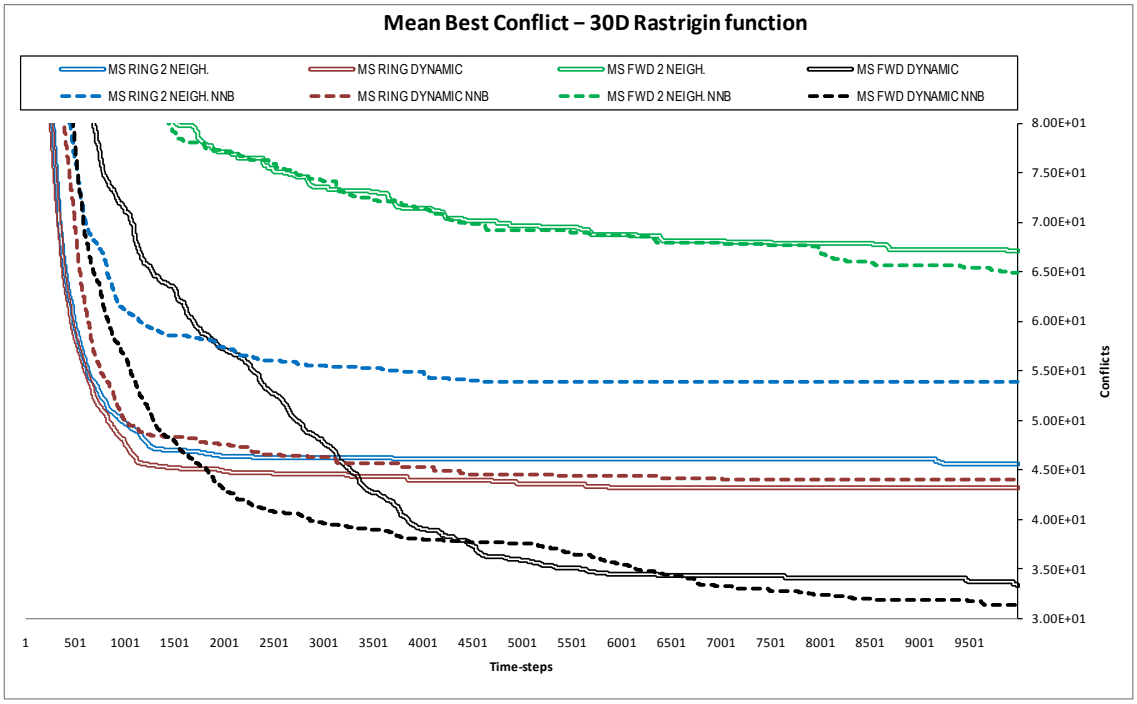


Fig. 7.66. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.39. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.40. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Griewank function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**2.51E-04**	**100**
			1000	0.00E+00	0.00E+00	2.39E-05	5.88E-04	2.05E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**6.74E-12**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.59E-03	-
	FWD	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.67E-04**	**100**
			1000	0.00E+00	0.00E+00	4.02E-08	8.51E-07	1.92E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**6.88E-12**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.53E-03	-
	RING NNB	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**2.90E-04**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.08E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**7.69E-12**	**100**
			1000	0.00E+00	0.00E+00	8.88E-04	7.40E-03	2.38E-03	-
	FWD NNB	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**8.62E-05**	**100**
			1000	0.00E+00	0.00E+00	1.89E-06	4.70E-05	1.94E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.48E-05**	**100**
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.23E-03	-

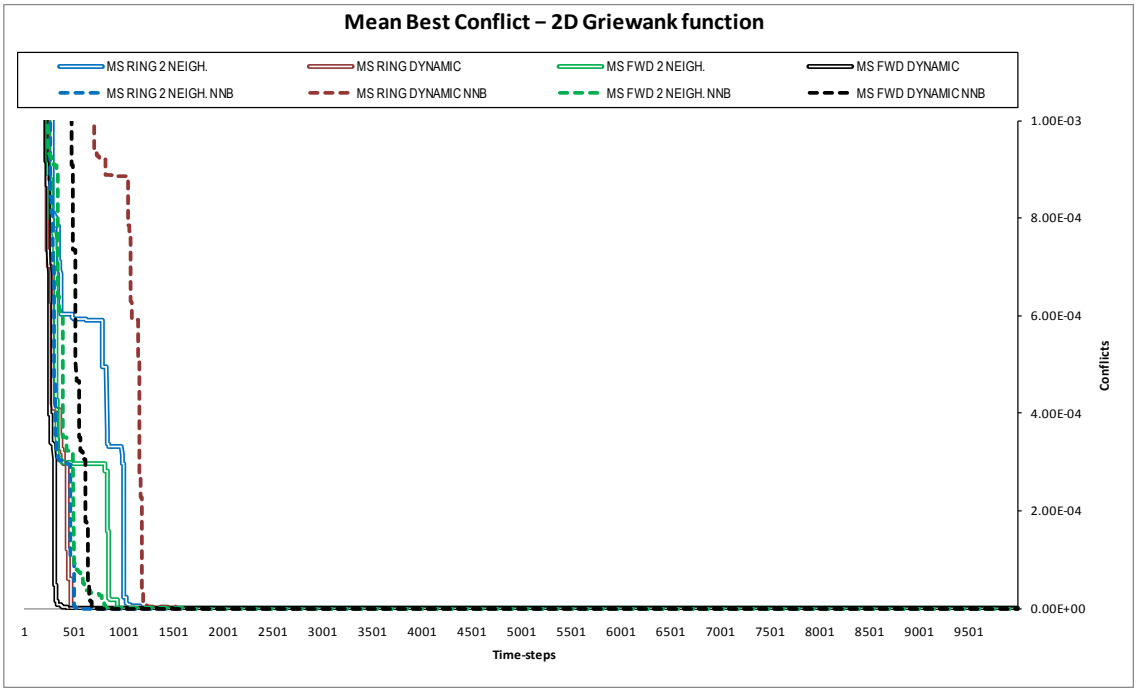


Fig. 7.67. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 7.40. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.41. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Griewank function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	**10000**	**0.00E+00**	**1.97E-02**	**2.15E-02**	**5.90E-02**	**1.60E-03**	**8**
			1000	7.40E-03	2.22E-02	2.71E-02	5.90E-02	1.75E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**2.46E-02**	**2.77E-02**	**6.89E-02**	**1.47E-03**	**8**
			1000	0.00E+00	3.45E-02	3.59E-02	7.38E-02	1.74E-03	-
	FWD	nn = 2	**10000**	**4.64E-05**	**3.60E-02**	**3.64E-02**	**9.76E-02**	**1.30E-03**	**4**
			1000	2.16E-02	6.01E-02	6.36E-02	1.49E-01	1.53E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**1.72E-02**	**2.33E-02**	**5.90E-02**	**1.08E-03**	**12**
			1000	7.94E-03	5.59E-02	6.33E-02	1.25E-01	1.73E-03	-
	RING NNB	nn = 2	**10000**	**1.11E-16**	**3.20E-02**	**3.31E-02**	**7.62E-02**	**1.67E-03**	**4**
			1000	2.60E-08	3.69E-02	4.00E-02	7.65E-02	1.72E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**2.46E-02**	**2.94E-02**	**6.89E-02**	**9.51E-04**	**16**
			1000	1.97E-02	4.43E-02	4.73E-02	9.10E-02	1.43E-03	-
	FWD NNB	nn = 2	**10000**	**7.38E-05**	**3.44E-02**	**3.81E-02**	**9.58E-02**	**1.21E-03**	**4**
			1000	2.56E-02	7.46E-02	7.77E-02	1.23E-01	1.61E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**1.48E-02**	**1.41E-02**	**5.17E-02**	**7.92E-04**	**32**
			1000	2.59E-03	4.67E-02	4.77E-02	9.92E-02	1.64E-03	-

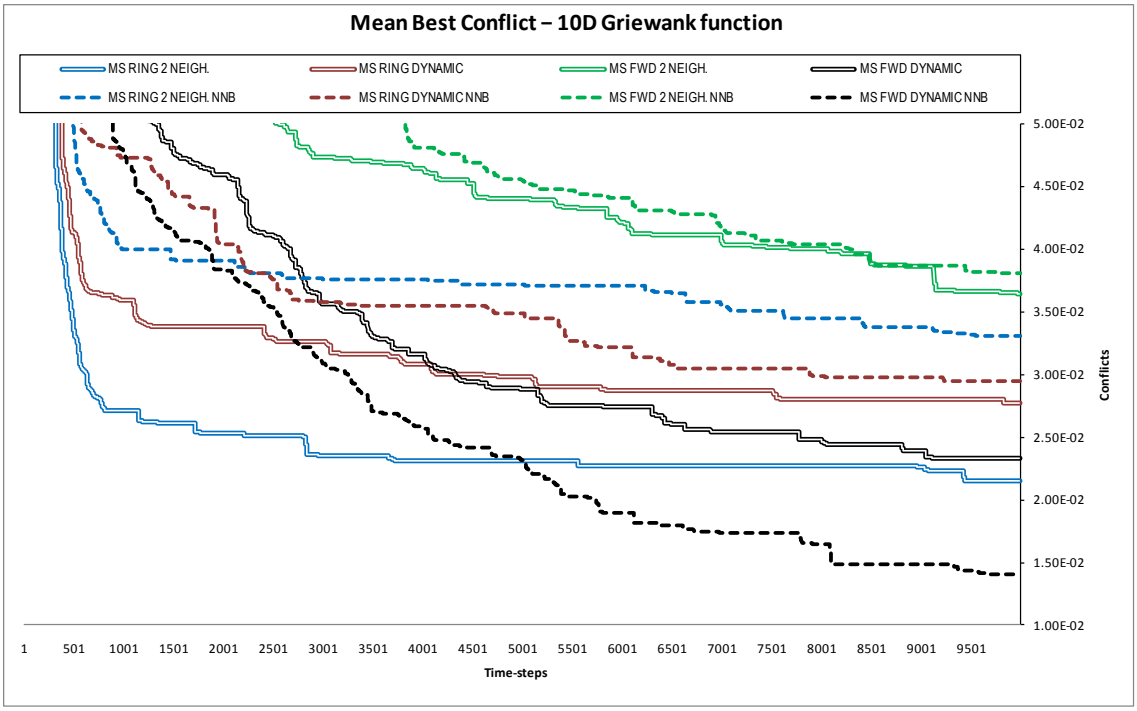


Fig. 7.68. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.41. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.42. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Griewank function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	10000	0.00E+00	0.00E+00	2.17E-03	1.23E-02	1.10E-06	76
			1000	6.02E-08	7.40E-03	6.97E-03	3.92E-02	3.08E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	6.39E-03	3.92E-02	4.12E-12	64
			1000	9.47E-11	7.40E-03	9.54E-03	3.92E-02	1.26E-05	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	3.07E-04	7.40E-03	2.58E-06	92
			1000	6.00E-03	4.20E-02	5.62E-02	2.12E-01	1.19E-04	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	2.07E-03	2.71E-02	3.36E-12	84
			1000	9.39E-06	1.49E-03	7.82E-03	3.60E-02	2.34E-05	-
	RING NNB	nn = 2	10000	0.00E+00	0.00E+00	5.92E-04	7.40E-03	3.54E-07	92
			1000	4.41E-04	2.61E-03	6.09E-03	6.06E-02	3.84E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	7.40E-03	1.10E-02	3.44E-02	7.28E-09	36
			1000	1.53E-08	1.48E-02	1.46E-02	4.44E-02	7.13E-05	-
	FWD NNB	nn = 2	10000	0.00E+00	0.00E+00	7.48E-04	1.04E-02	2.64E-06	88
			1000	7.33E-05	1.31E-02	1.74E-02	5.68E-02	6.44E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	6.40E-03	4.67E-02	3.10E-06	68
			1000	1.82E-04	2.59E-02	3.12E-02	8.07E-02	2.24E-04	-

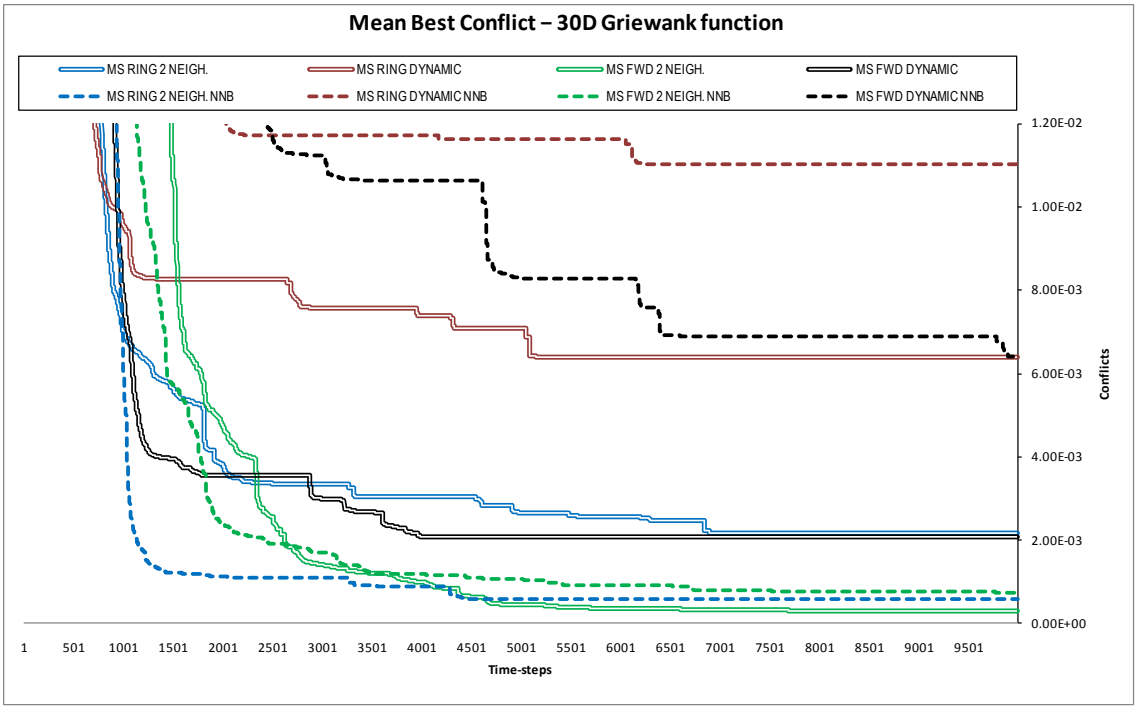


Fig. 7.69. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.42. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.43. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 2-dimensional Schaffer f6 function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**7.13E-04**	**100**
			1000	0.00E+00	0.00E+00	7.84E-04	9.72E-03	7.34E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.36E-11**	**100**
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	6.25E-03	-
	FWD	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**3.80E-04**	**100**
			1000	0.00E+00	5.94E-14	8.08E-04	9.72E-03	6.89E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.13E-11**	**100**
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	5.17E-03	-
	RING NNB	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**3.71E-04**	**100**
			1000	0.00E+00	0.00E+00	1.28E-03	9.72E-03	7.36E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.35E-11**	**100**
			1000	0.00E+00	0.00E+00	4.58E-03	9.72E-03	8.44E-03	-
	FWD NNB	nn = 2	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**9.85E-05**	**100**
			1000	0.00E+00	2.22E-16	5.84E-08	1.45E-06	6.60E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.25E-11**	**100**
			1000	0.00E+00	0.00E+00	7.97E-04	9.72E-03	7.46E-03	-

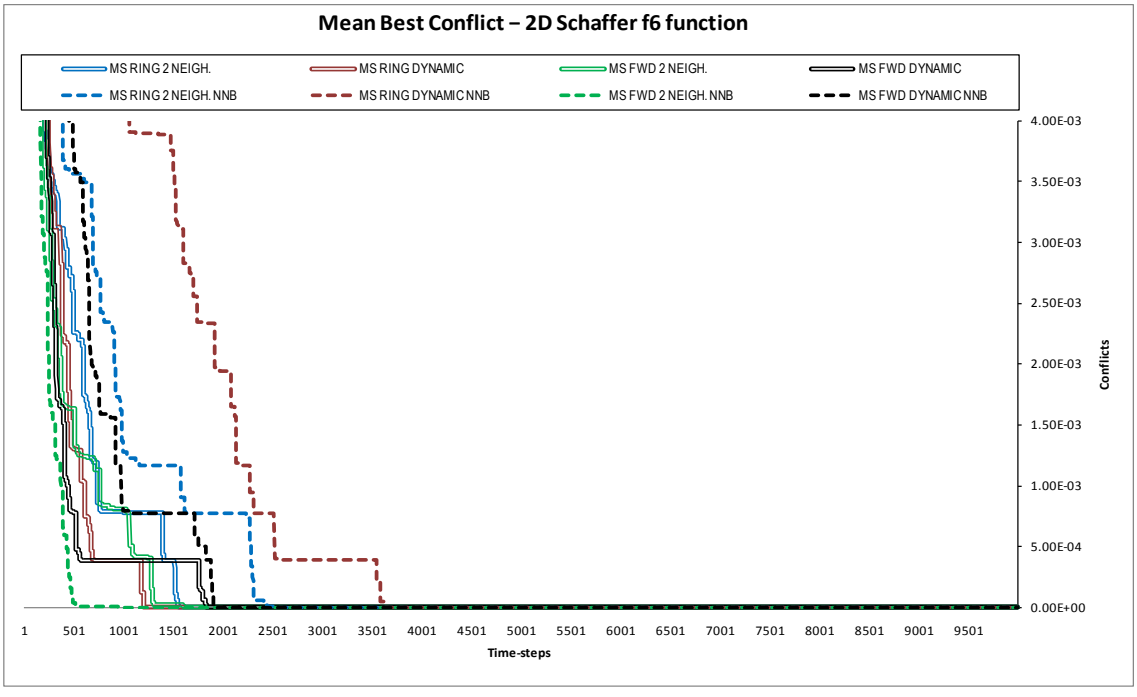


Fig. 7.70. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 7.43. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.44. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 10-dimensional Schaffer f6 function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	10000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	2.03E-03	0
			1000	9.72E-03	9.72E-03	1.74E-02	3.72E-02	3.03E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.76E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	1.95E-03	-
	FWD	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.70E-03	0
			1000	9.72E-03	9.72E-03	1.41E-02	3.72E-02	2.41E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.59E-03	0
			1000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	1.79E-03	-
	RING NNB	nn = 2	10000	9.72E-03	9.72E-03	1.30E-02	3.72E-02	2.01E-03	0
			1000	9.72E-03	9.72E-03	2.07E-02	3.72E-02	3.18E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.71E-03	0
			1000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.96E-03	-
	FWD NNB	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.91E-03	0
			1000	9.72E-03	9.72E-03	1.41E-02	3.72E-02	2.72E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.42E-03	0
			1000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	2.29E-03	-

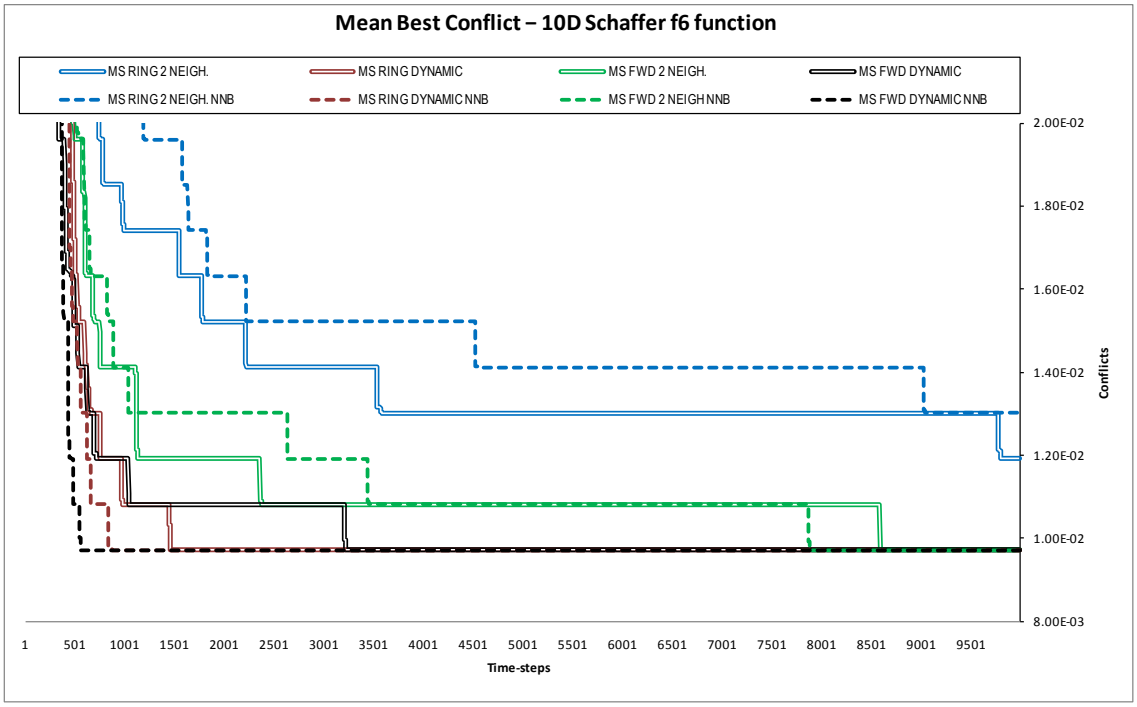


Fig. 7.71. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.44. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.45. Statistical results out of 25 runs for a Multi-Swarm algorithm optimizing the 30-dimensional Schaffer f6 function. The neighbourhoods tested are the RING and the FOWARD topologies, both with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'), combined with a nearest neighbour heuristics (NNB). The results for the same neighbourhood topologies without such heuristics are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	RING	nn = 2	10000	3.72E-02	7.82E-02	7.45E-02	1.27E-01	1.58E-03	0
			1000	1.27E-01	1.78E-01	1.68E-01	2.28E-01	3.42E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.38E-02	7.82E-02	5.48E-04	0
			1000	3.74E-02	1.27E-01	1.08E-01	1.78E-01	1.89E-03	-
	FWD	nn = 2	10000	3.72E-02	7.82E-02	7.29E-02	1.27E-01	1.70E-03	0
			1000	7.82E-02	1.27E-01	1.44E-01	1.78E-01	2.83E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.38E-02	7.82E-02	6.55E-04	0
			1000	3.72E-02	7.82E-02	9.05E-02	1.27E-01	1.79E-03	-
	RING NNB	nn = 2	10000	3.72E-02	7.82E-02	7.19E-02	1.27E-01	1.85E-03	0
			1000	1.27E-01	1.78E-01	1.84E-01	2.28E-01	3.51E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	3.89E-02	7.82E-02	4.33E-04	0
			1000	6.66E-02	1.27E-01	1.09E-01	1.78E-01	4.33E-04	-
	FWD NNB	nn = 2	10000	3.72E-02	7.82E-02	7.59E-02	1.27E-01	1.71E-03	0
			1000	7.82E-02	1.27E-01	1.14E-01	1.78E-01	2.46E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	3.72E-02	3.56E-02	7.82E-02	7.96E-04	0
			1000	7.82E-02	7.82E-02	1.02E-01	1.78E-01	3.44E-03	-

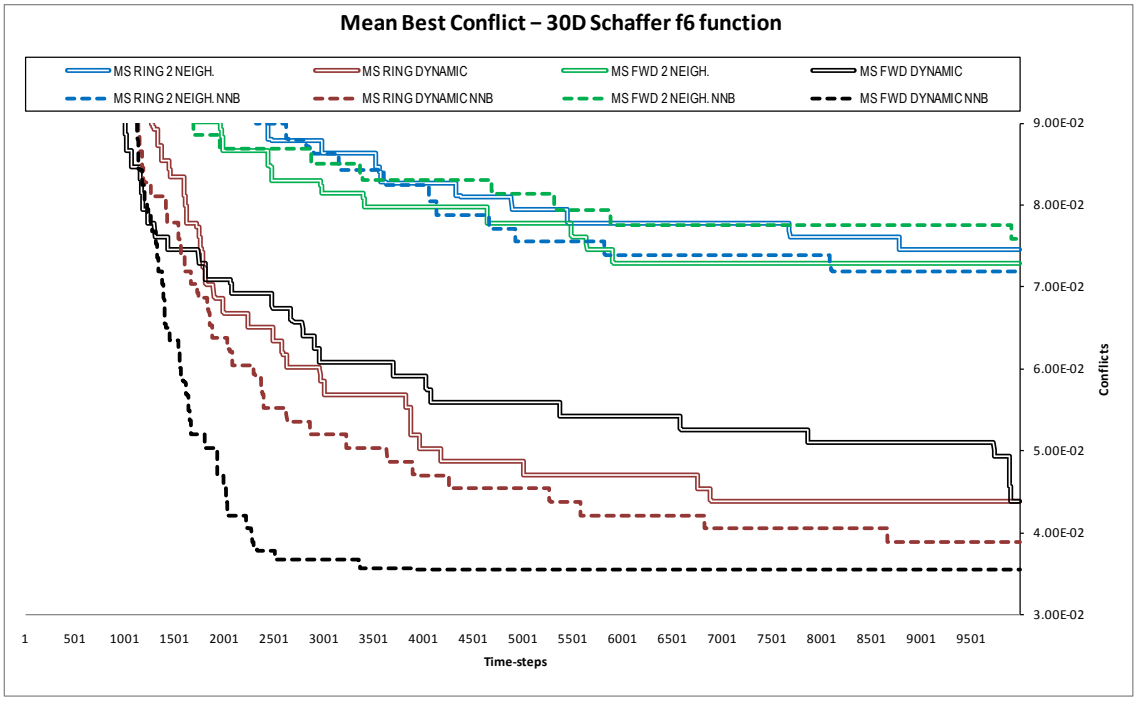


Fig. 7.72. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.45. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Discussion

The settings of the experiments are the same as those described in the previous chapter (see section 6.3.2.2.) unless specifically stated otherwise. Although the 'Fwd *nn*=2' and the 'Ring *nn*=2' topologies are included in the experiments, the analyses focus on the 'Fwd Dynamic' and the 'Ring Dynamic' topologies.

Sphere

All the algorithms, with and without NNB, find the exact solutions in 2D, 10D and 30D, so that the effect of the NNB technique cannot be appreciated (Table 7.31 to Table 7.33, Fig. 7.58 to Fig. 7.60).

Rosenbrock

In the 2D problem, all algorithms, with and without NNB, find the exact solutions so that the effect of the NNB technique cannot be appreciated (Table 7.34, Fig. 7.61). In the 10D problem, no clear trend can be observed in the forward topologies. However, the 'Ring Dynamic' topology marginally deteriorates whereas the 'Ring *nn*=2' topology improves (Table 7.35, Fig. 7.62). In the 30D problem, both the 'Fwd Dynamic' and the 'Ring Dynamic' topologies deteriorate, whereas no clear trend is observed for the others (Table 7.36, Fig. 7.63).

Rastrigin

In the 2D problem, all algorithms, with and without NNB, find the exact solutions so that the effect of the NNB technique cannot be appreciated (Table 7.37, Fig. 7.64). In the 10D problem, both the 'Fwd Dynamic' and the 'Ring Dynamic' improve noticeably with the NNB technique, while the 'Ring *nn*=2' deteriorates (Table 7.38, Fig. 7.65). In the 30D problem, no noticeable effect can be observed except for the deterioration of the results obtained by the 'Ring *nn*=2' (Table 7.39, Fig. 7.66).

Griewank

In the 2D problem, all algorithms, with and without NNB, find the exact solutions so that the effect of the NNB technique cannot be appreciated (Table 7.40, Fig. 7.67). In the 10D problem, the 'Fwd Dynamic' topology improves with the NNB whereas all the

others deteriorate (Table 7.41, Fig. 7.68). In the 30D problem, both the 'Fwd Dynamic' and the 'Ring Dynamic' topologies deteriorate with the NNB whereas the 'Ring *nn*=2' improves (Table 7.42, Fig. 7.69).

Schaffer f6

In the 2D problem, all algorithms, with and without NNB, find the exact solutions so that the effect of the NNB technique cannot be appreciated (Table 7.43, Fig. 7.70). In the 10D problem, there is no noticeable effect of the NNB (Table 7.44, Fig. 7.71). In the 30D problem, the NNB results again in improvement for both the 'Fwd Dynamic' and the 'Ring Dynamic' (Table 7.45, Fig. 7.72).

Overall analyses

The application of the proposed Nearest Neighbour technique (based on the memory swarm) on the problems of this benchmark resulted sometimes in no noticeable effect, sometimes in deterioration, and sometimes in improvement. Given that the reasons and mechanisms for these either improvements or deteriorations were not identified, the technique is not really useful at this stage for mathematical optimization. It may be helpful for physical systems, where the neighbourhoods are generated using distance information. This technique would allow that while still keeping the full overlapping, thus guaranteeing the spread of information at any time throughout the search.

7.5. Sub-neighbourhoods

In order to clarify the nomenclature, let us call a *sub-swarm* to a subset of the swarm composed of all the particles sharing the same setting of the coefficients in the velocity update equation. In turn, we call a sub-neighbourhood to a subset of the swarm composed of a number of particles subject to an independent neighbourhood structure. Hence there can be more than one sub-swarm in one neighbourhood, and different sub-neighbourhoods can share the same coefficients' settings. The implemented PSO code allows one sub-swarm and multiple sub-neighbourhoods; one sub-neighbourhood and multiple sub-swarms; and if there are more than one sub-neighbourhood and more than one sub-swarm, they are forced to coincide.

In the algorithm with three sub-swarms in the previous sections, the particles equipped with different coefficients –i.e. comprising different sub-swarms– end up scrambled. Although the question of whether this is beneficial or harmful has not been addressed yet, it was decided here to set three sub-neighbourhoods coinciding with the three sub-swarms in the previous section so that they do not mix and each sub-neighbourhood behaves as dictated by one set of coefficients. Thus, the forward dynamic neighbourhood from section 7.3 is implemented on each sub-neighbourhood independently. Since the update of the number of neighbours is linear, each particle spends a longer interval of time between updates of the number of neighbours when the sub-neighbourhoods option is activated.

The question is how to pass information through between sub-neighbourhoods without excessively disrupting their independence. Two techniques to interconnect the sub-neighbourhoods are investigated. In the first one, aiming to keep them as independent from each other as possible, the neighbourhood of the first particle in each sub-neighbourhood is extended by adding the individual best experience of the first particles in the other sub-neighbourhoods. Thus, information can be passed through very mildly. This is called here 'individual overlapping'. The other technique, called 'local overlapping', consists of extending the neighbourhood of the first particle in each sub-neighbourhood with the neighbourhood (rather than the individual best) of the first particles of the other sub-neighbourhoods. In other words, their neighbourhoods are aggregated and shared. If the sub-neighbourhoods' topology is dynamic, the overlapping using the second technique will be dynamic as well. The individual overlapping is illustrated in Fig. 7.73, where the sub-neighbourhoods consist of a forward topology with two neighbours. The local overlapping is illustrated in Fig. 7.74 for the same sub-neighbourhoods' structure.

The sub-neighbourhood option is tested on the same suite of side-constrained problems used before. The Multi-Swarm coefficients' settings is used, and five neighbourhood topologies are tested, namely the global, the forward dynamic, the forward (*nn*=2), the ring dynamic, and the ring (*nn*=2). Each one of this is coupled with three options: no sub-neighbourhood; 3 Sub-Neighbourhoods with Individual overlapping (SNI); and 3 Sub-Neighbourhoods with Local overlapping (SNL). The results are presented in Table 7.46 to Table 7.60, and Fig. 7.75 to Fig. 7.101.

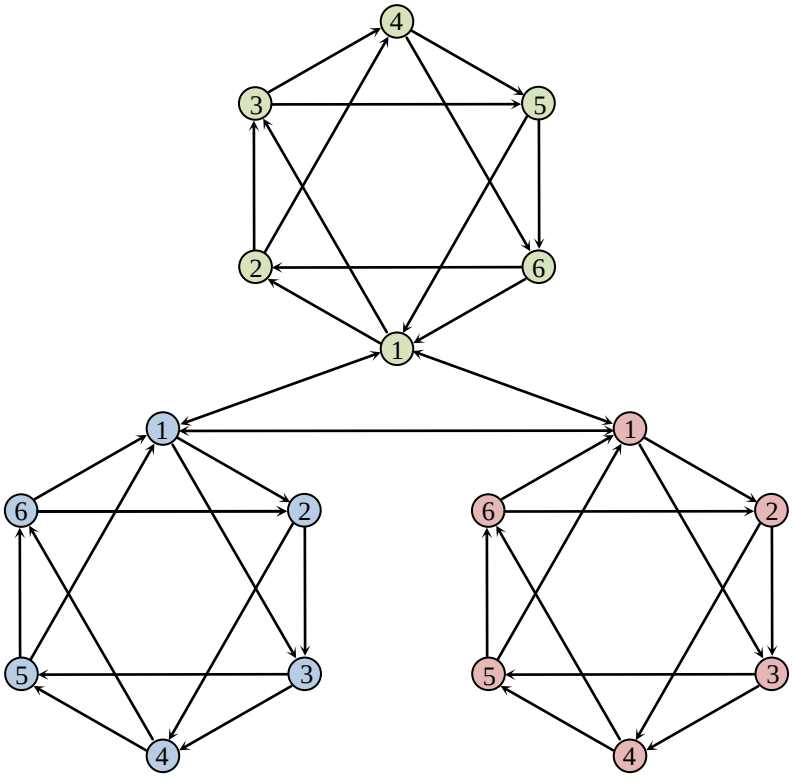


Fig. 7.73. Sub-neighbourhoods with INDIVIDUAL overlapping.

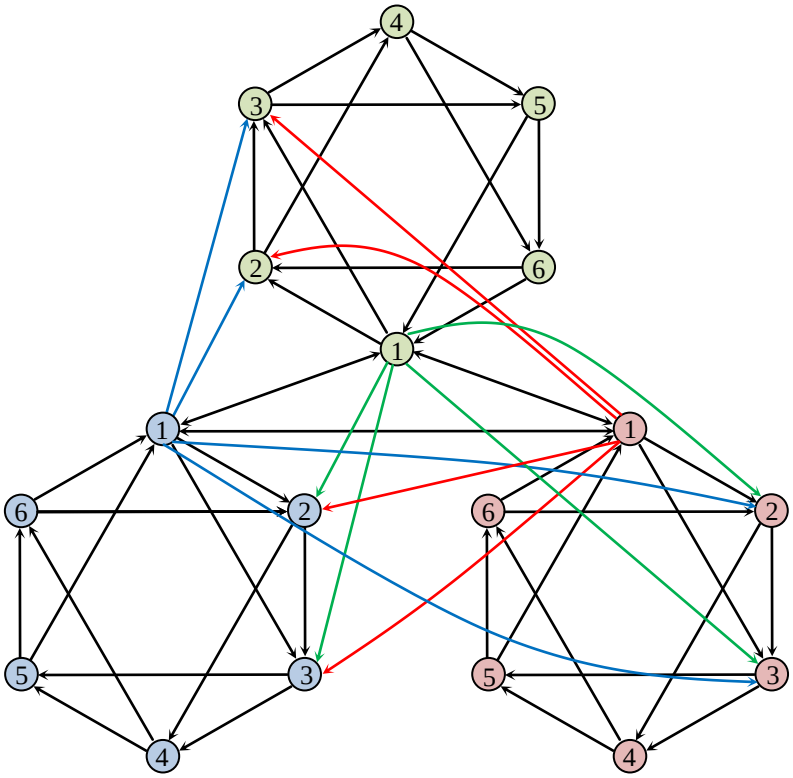


Fig. 7.74. Sub-neighbourhoods with LOCAL overlapping.

Table 7.46. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Sphere function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.38E-90	4.67E-86	1.06E-83	2.16E-82	1.70E-20	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.39E-85	4.26E-78	9.85E-77	1.83E-75	2.68E-20	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.99E-86	1.12E-80	2.13E-79	2.33E-78	4.47E-20	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.24E-75	8.38E-73	4.72E-69	1.16E-67	9.31E-20	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.17E-79	3.23E-74	3.17E-70	7.36E-69	8.94E-22	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.69E-89	1.05E-83	9.99E-83	1.80E-81	2.11E-20	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.39E-85	1.51E-79	3.18E-76	5.09E-75	5.04E-21	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	7.06E-82	3.59E-79	1.77E-76	2.80E-75	8.89E-21	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.26E-80	3.24E-76	6.26E-75	7.36E-74	7.51E-21	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.30E-79	2.09E-77	2.47E-76	2.50E-75	5.46E-20	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.27E-87	8.43E-84	1.61E-81	3.74E-80	1.31E-21	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	9.83E-82	1.20E-78	3.67E-76	7.10E-75	4.16E-21	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.25E-85	2.16E-79	2.48E-76	6.10E-75	2.84E-21	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	7.37E-79	7.28E-77	2.33E-74	4.64E-73	7.66E-21	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.78E-81	6.51E-78	7.10E-77	5.31E-76	1.00E-20	-

Table 7.47. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Sphere function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.21E-58	2.00E-56	2.03E-55	1.42E-54	2.17E-18	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.20E-153	100
			1000	1.41E-32	1.17E-30	1.06E-29	9.91E-29	2.68E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.18E-43	2.00E-39	5.49E-38	1.09E-36	1.35E-14	-
	FWD	nn = 2	10000	6.03E-186	1.07E-182	4.73E-180	1.07E-178	2.39E-93	100
			1000	4.92E-18	6.77E-17	1.20E-16	5.25E-16	7.03E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.97E-23	1.46E-22	3.15E-22	1.75E-21	1.37E-13	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.81E-55	3.16E-51	1.28E-48	2.42E-47	1.25E-15	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.40E-159	100
			1000	3.39E-35	3.35E-32	1.44E-31	1.09E-30	1.86E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	3.13E-37	5.93E-36	4.45E-35	6.25E-34	1.46E-12	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.46E-160	100
			1000	1.63E-35	9.93E-34	5.03E-33	3.98E-32	2.01E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.92E-38	5.64E-37	1.19E-35	1.40E-34	2.68E-13	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.45E-55	5.60E-53	1.60E-49	2.50E-48	1.81E-15	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	5.83E-158	100
			1000	2.12E-34	4.30E-33	1.02E-32	5.53E-32	1.56E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.07E-38	2.95E-36	5.43E-35	9.58E-34	4.97E-13	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.68E-161	100
			1000	9.70E-36	7.67E-34	8.62E-33	1.23E-31	4.24E-12	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.52E-39	1.64E-36	1.06E-35	1.49E-34	6.11E-13	-

Table 7.48. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Sphere function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SPHERE 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	1.10E-181	4.68E-172	1.95E-166	4.21E-165	7.90E-88	100
			1000	4.53E-17	1.52E-14	4.02E-11	1.00E-09	3.63E-10	-
	RING	nn = 2	10000	3.13E-113	7.02E-109	2.72E-107	5.96E-106	6.93E-57	100
			1000	2.83E-08	9.73E-08	2.55E-07	3.82E-06	1.75E-06	-
		nni = 2 nnf = (m – 1)	10000	1.28E-185	6.46E-180	6.80E-173	1.68E-171	2.58E-91	100
			1000	7.07E-12	1.43E-10	3.51E-10	3.97E-09	1.57E-08	-
	FWD	nn = 2	10000	9.63E-63	2.09E-61	1.04E-60	6.46E-60	3.03E-34	100
			1000	4.04E-04	1.74E-03	1.79E-03	3.91E-03	1.73E-05	-
		nni = 2 nnf = (m – 1)	10000	1.16E-174	4.86E-169	2.11E-165	2.09E-164	3.76E-87	100
			1000	1.09E-06	4.95E-06	5.29E-06	1.20E-05	7.73E-07	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	3.05E-173	6.28E-170	1.98E-165	4.84E-164	3.29E-87	100
			1000	5.76E-15	9.75E-14	2.81E-13	1.72E-12	3.29E-87	-
	RING	nn = 2	10000	1.26E-123	2.66E-120	2.71E-118	5.65E-117	3.84E-62	100
			1000	5.76E-10	5.67E-09	1.11E-08	8.51E-08	6.80E-07	-
		nni = 2 nnf = (m – 1)	10000	1.42E-162	3.34E-157	4.42E-156	3.95E-155	1.34E-81	100
			1000	1.34E-10	1.05E-09	1.49E-09	7.63E-09	1.19E-07	-
	FWD	nn = 2	10000	4.21E-120	4.99E-117	2.05E-115	3.97E-114	8.97E-61	100
			1000	7.61E-10	1.01E-08	1.80E-08	1.26E-07	4.70E-07	-
		nni = 2 nnf = (m – 1)	10000	1.02E-158	2.75E-154	2.95E-150	3.79E-149	6.89E-79	100
			1000	7.57E-11	1.33E-09	2.51E-09	1.36E-08	1.07E-07	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	1.36E-181	2.61E-176	1.66E-171	4.00E-170	1.29E-89	100
			1000	6.80E-17	8.27E-15	3.59E-14	2.53E-13	1.03E-10	-
	RING	nn = 2	10000	2.30E-126	3.53E-124	3.24E-122	2.99E-121	8.01E-64	100
			1000	3.32E-10	1.92E-09	3.60E-09	2.29E-08	6.13E-07	-
		nni = 2 nnf = (m – 1)	10000	2.22E-170	2.66E-164	2.21E-160	5.45E-159	4.63E-84	100
			1000	3.58E-11	2.15E-10	8.94E-10	4.37E-09	8.74E-08	-
	FWD	nn = 2	10000	3.22E-124	1.61E-120	4.42E-118	7.32E-117	5.58E-62	100
			1000	7.20E-10	3.64E-09	9.23E-09	7.65E-08	5.05E-07	-
		nni = 2 nnf = (m – 1)	10000	1.15E-169	4.78E-162	8.43E-158	2.06E-156	4.78E-83	100
			1000	3.46E-11	6.45E-10	9.43E-10	2.80E-09	1.27E-07	-

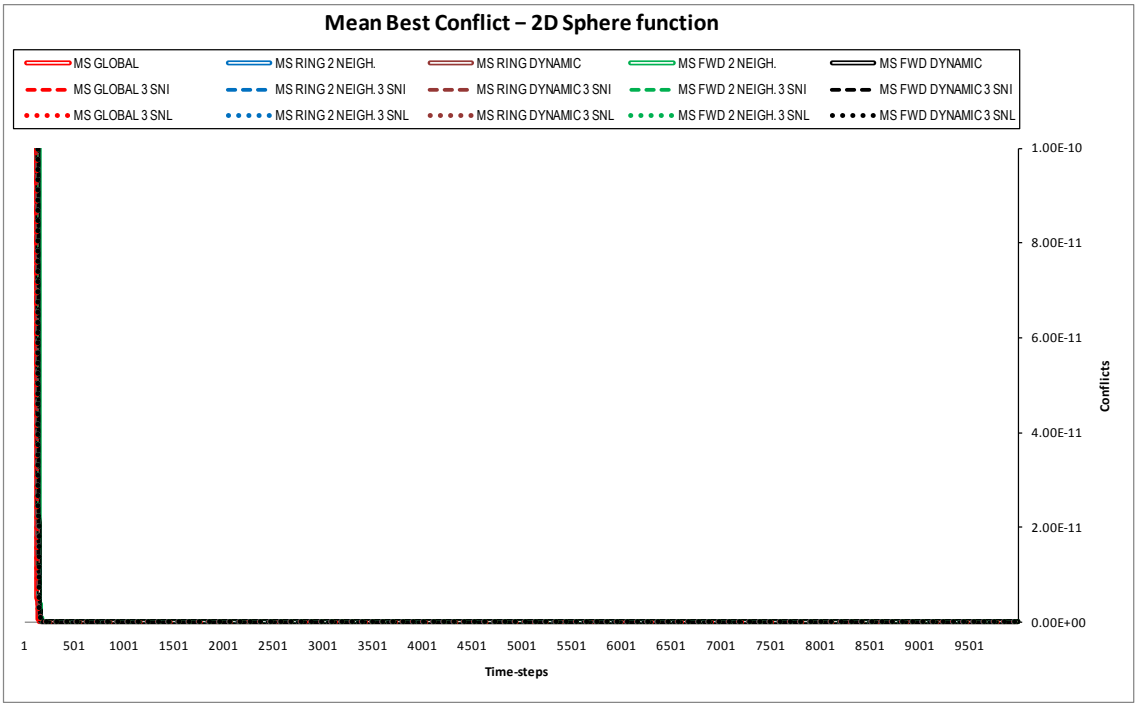


Fig. 7.75. Convergence curves of the mean best conflict for the 2D Sphere function, associated to Table 7.46. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

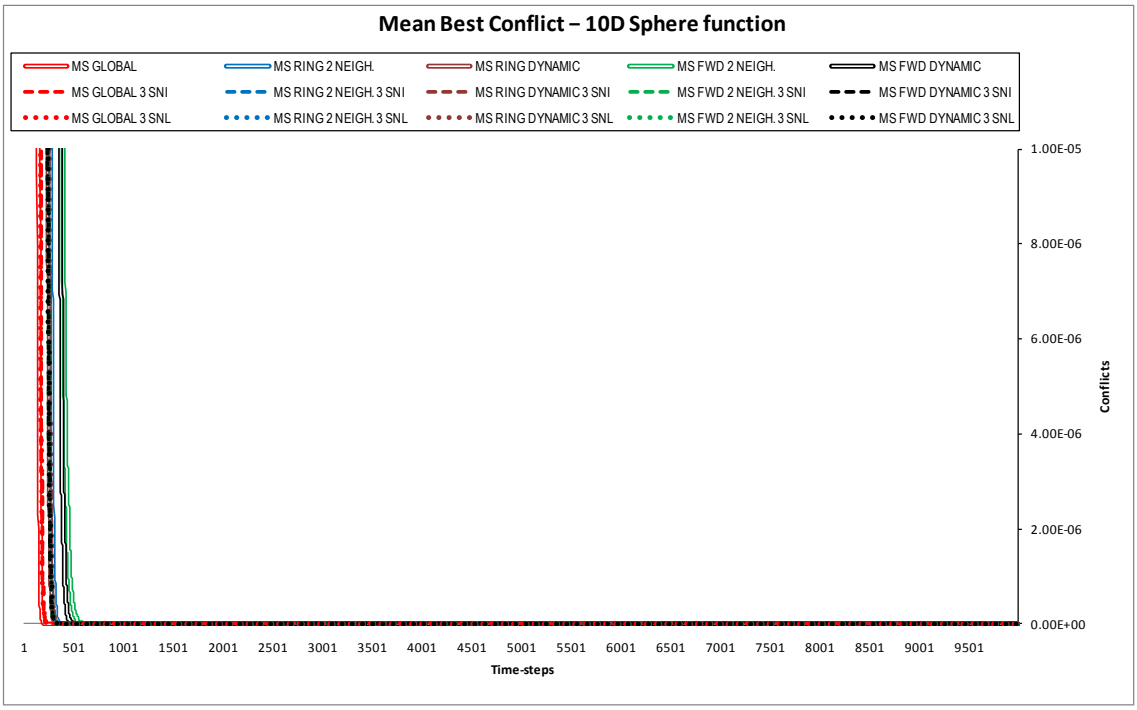


Fig. 7.76. Convergence curves of the mean best conflict for the 10D Sphere function, associated to Table 7.47. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

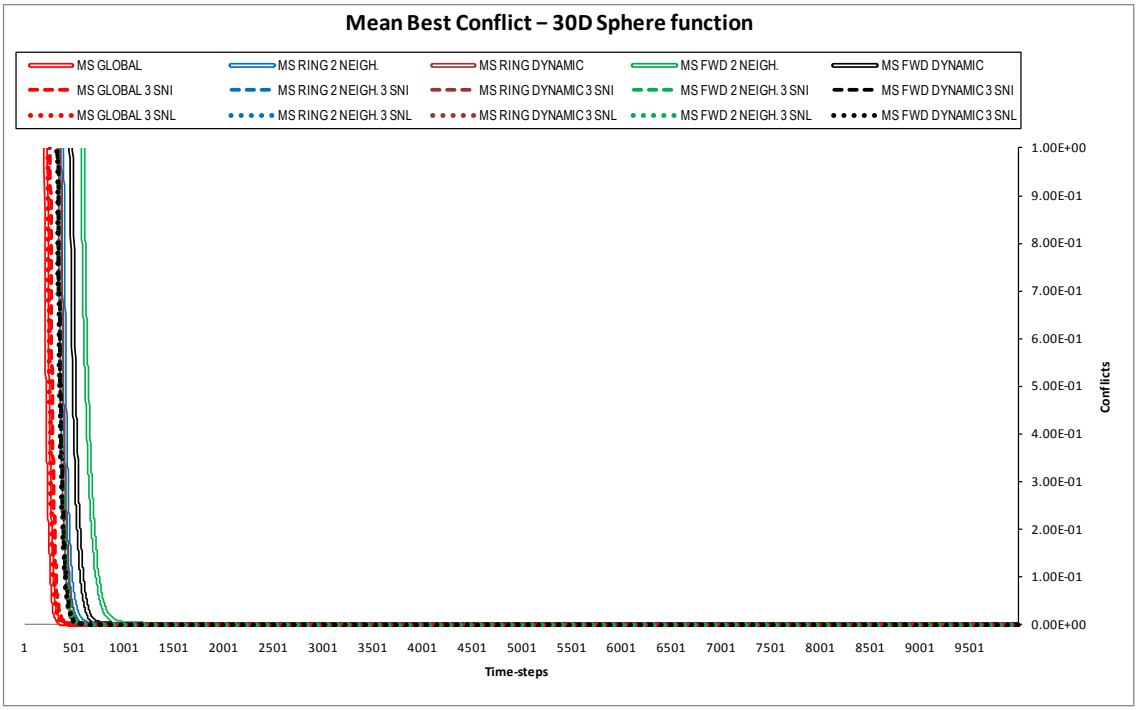


Fig. 7.77. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.48. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

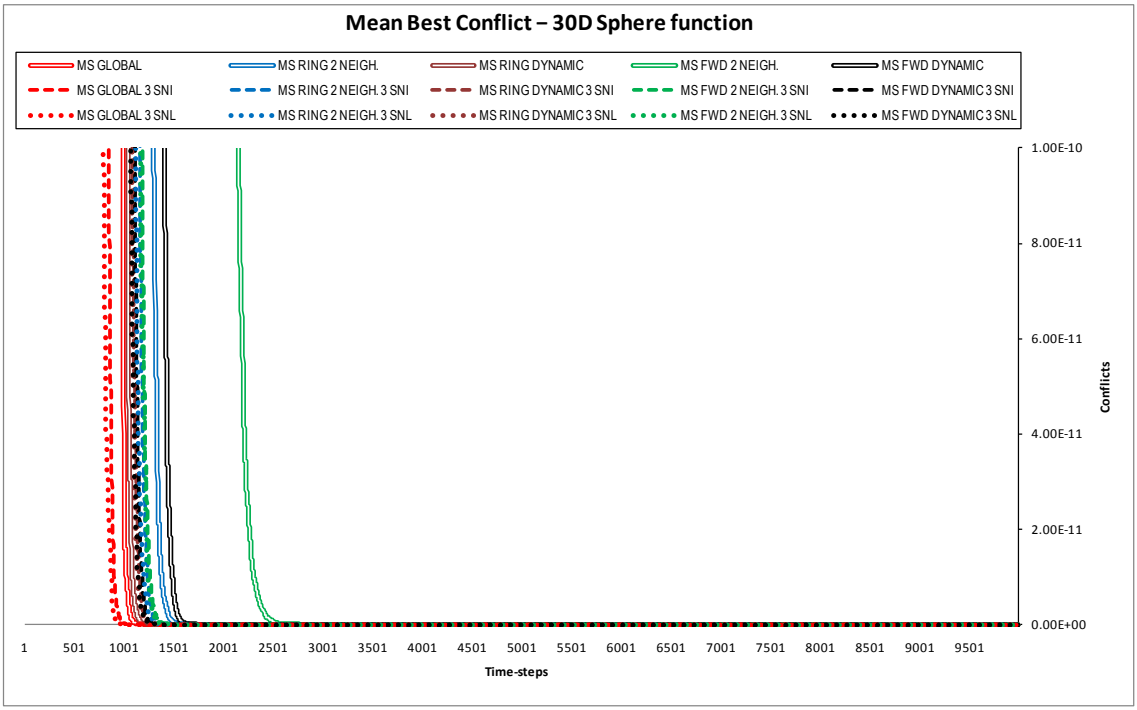


Fig. 7.78. Convergence curves of the mean best conflict for the 30D Sphere function, associated to Table 7.48. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.49. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Rosenbrock function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.05E-09	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	9.00E-24	2.19E-18	3.50E-17	1.78E-06	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	1.77E-30	1.67E-26	3.95E-25	1.36E-06	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.90E-22	1.03E-16	4.09E-14	5.40E-13	7.85E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.34E-26	1.05E-21	5.04E-19	1.24E-17	1.10E-07	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.33E-08	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	3.45E-25	1.16E-22	2.04E-21	6.95E-07	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	6.15E-28	1.01E-23	1.53E-22	1.87E-04	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.08E-27	8.07E-23	1.92E-20	2.02E-19	1.54E-06	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.23E-30	1.31E-26	4.77E-24	8.59E-23	2.01E-07	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	0.00E+00	7.89E-33	1.97E-31	1.14E-08	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	1.97E-29	2.53E-25	7.47E-23	1.36E-21	2.26E-06	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	0.00E+00	6.88E-27	9.48E-26	1.05E-24	1.87E-04	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	4.79E-30	1.87E-23	5.45E-20	9.54E-19	4.92E-06	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	100
			1000	2.42E-30	2.72E-27	3.91E-23	6.78E-22	6.13E-07	-

Table 7.50. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Rosenbrock function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		**10000**	**5.91E-12**	**2.94E-02**	**1.02E+00**	**5.59E+00**	**7.53E-03**	**16**
			1000	9.37E-05	6.67E-01	1.47E+00	6.46E+00	7.93E-03	-
	RING	nn = 2	**10000**	**4.01E-09**	**8.03E-07**	**1.61E-01**	**3.99E+00**	**8.87E-03**	**80**
			1000	2.63E-03	1.62E+00	1.82E+00	5.11E+00	1.79E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**1.88E-15**	**1.16E-09**	**1.42E-05**	**3.49E-04**	**1.08E-02**	**96**
			1000	5.39E-04	6.18E-01	8.49E-01	4.07E+00	1.45E-02	-
	FWD	nn = 2	**10000**	**9.26E-03**	**3.47E+00**	**3.47E+00**	**7.03E+00**	**4.92E-03**	**0**
			1000	1.09E-02	5.46E+00	5.35E+00	9.55E+00	9.53E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**3.40E-12**	**4.82E-09**	**4.80E-01**	**4.01E+00**	**4.57E-03**	**76**
			1000	8.74E-01	4.48E+00	4.63E+00	9.59E+00	6.54E-03	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		**10000**	**1.29E-07**	**2.50E-04**	**2.32E-03**	**1.96E-02**	**4.79E-03**	**28**
			1000	4.85E-02	1.54E+00	6.27E+00	7.17E+01	5.49E-03	-
	RING	nn = 2	**10000**	**1.74E-08**	**1.58E-07**	**1.78E-06**	**3.58E-05**	**5.54E-03**	**100**
			1000	5.35E-03	2.24E+00	1.88E+00	4.24E+00	1.88E+00	-
		nni = 2 nnf = (m – 1)	**10000**	**3.24E-22**	**1.54E-17**	**6.16E-11**	**1.47E-09**	**5.86E-03**	**100**
			1000	5.00E-03	1.80E+00	1.86E+00	4.42E+00	9.82E-03	-
	FWD	nn = 2	**10000**	**4.78E-09**	**5.90E-08**	**1.60E-01**	**3.99E+00**	**5.73E-03**	**92**
			1000	6.92E-02	2.10E+00	2.18E+00	7.01E+00	5.98E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**4.37E-20**	**1.28E-17**	**1.59E-01**	**3.99E+00**	**4.86E-03**	**96**
			1000	6.28E-02	1.18E+00	1.41E+00	4.07E+00	5.82E-03	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		**10000**	**4.76E-08**	**1.50E-04**	**6.42E-01**	**3.99E+00**	**5.48E-03**	**44**
			1000	2.34E-02	1.21E+00	1.85E+00	6.36E+00	5.24E-03	-
	RING	nn = 2	**10000**	**2.66E-10**	**1.81E-07**	**1.66E-01**	**3.99E+00**	**5.43E-03**	**80**
			1000	3.36E-02	1.72E+00	1.74E+00	5.28E+00	1.05E-02	-
		nni = 2 nnf = (m – 1)	**10000**	**1.26E-20**	**2.03E-17**	**3.19E-01**	**3.99E+00**	**6.26E-03**	**88**
			1000	3.82E-02	1.51E+00	1.62E+00	4.19E+00	1.00E-02	-
	FWD	nn = 2	**10000**	**7.73E-09**	**3.22E-08**	**3.19E-01**	**3.99E+00**	**4.11E-03**	**92**
			1000	9.07E-02	2.22E+00	2.24E+00	6.76E+00	2.37E-03	-
		nni = 2 nnf = (m – 1)	**10000**	**1.88E-21**	**4.15E-18**	**1.59E-01**	**3.99E+00**	**3.49E-03**	**96**
			1000	5.13E-02	1.43E+00	1.46E+00	4.13E+00	2.34E-03	-

Table 7.51. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Rosenbrock function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	ROSENBROCK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	4.21E-08	1.67E+01	2.70E+01	7.68E+01	3.90E-03	4
			1000	2.33E-02	2.30E+01	4.38E+01	1.36E+02	2.59E-03	-
	RING	nn = 2	10000	9.14E-03	7.09E+00	6.59E+00	1.46E+01	3.01E-03	0
			1000	6.26E+00	7.08E+01	5.33E+01	8.71E+01	5.27E-03	-
		nni = 2 nnf = (m – 1)	10000	1.56E-05	5.30E+00	6.36E+00	1.98E+01	2.76E-03	4
			1000	4.72E+00	2.73E+01	4.05E+01	1.34E+02	3.79E-03	-
	FWD	nn = 2	10000	1.95E+01	3.25E+01	5.09E+01	1.77E+02	3.71E-03	0
			1000	3.27E+01	7.61E+01	9.11E+01	2.12E+02	5.89E-03	-
		nni = 2 nnf = (m – 1)	10000	2.04E-04	1.10E+00	4.08E+00	7.00E+01	2.35E-03	0
			1000	2.25E+01	2.71E+01	5.27E+01	2.17E+02	1.53E-03	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	5.43E-05	4.55E+00	1.23E+01	7.94E+01	3.02E-03	4
			1000	1.29E+01	7.13E+01	5.58E+01	1.45E+02	1.46E-03	-
	RING	nn = 2	10000	1.98E-04	4.13E+00	6.49E+00	1.45E+01	4.08E-03	0
			1000	1.71E+00	7.79E+01	6.58E+01	1.37E+02	3.53E-03	-
		nni = 2 nnf = (m – 1)	10000	2.75E-06	3.28E+00	3.02E+00	1.31E+01	4.09E-03	8
			1000	1.70E+00	7.46E+01	5.71E+01	1.35E+02	1.96E-03	-
	FWD	nn = 2	10000	2.16E-04	1.01E+01	1.67E+01	7.54E+01	2.51E-03	0
			1000	1.62E+01	7.90E+01	7.78E+01	1.51E+02	1.79E-03	-
		nni = 2 nnf = (m – 1)	10000	2.52E-05	3.58E+00	3.16E+00	1.00E+01	4.32E-03	16
			1000	8.72E+00	7.92E+01	7.24E+01	1.54E+02	1.58E-03	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	5.92E-01	1.21E+01	1.48E+01	7.03E+01	2.94E-03	0
			1000	5.68E+00	2.59E+01	4.51E+01	1.07E+02	1.11E-03	-
	RING	nn = 2	10000	1.96E-02	4.00E+00	7.56E+00	6.84E+01	4.63E-03	0
			1000	1.24E+01	7.46E+01	6.69E+01	3.32E+02	3.00E-03	-
		nni = 2 nnf = (m – 1)	10000	6.97E-05	3.14E+00	3.03E+00	1.11E+01	5.78E-03	4
			1000	1.18E+01	7.24E+01	5.83E+01	1.34E+02	2.68E-03	-
	FWD	nn = 2	10000	8.40E-03	9.38E+00	1.14E+01	7.81E+01	1.32E-03	0
			1000	4.19E+00	7.99E+01	7.68E+01	2.58E+02	1.62E-03	-
		nni = 2 nnf = (m – 1)	10000	1.58E-05	1.92E+00	5.05E+00	7.09E+01	2.90E-03	4
			1000	3.76E+00	7.31E+01	6.69E+01	2.15E+02	1.16E-03	-

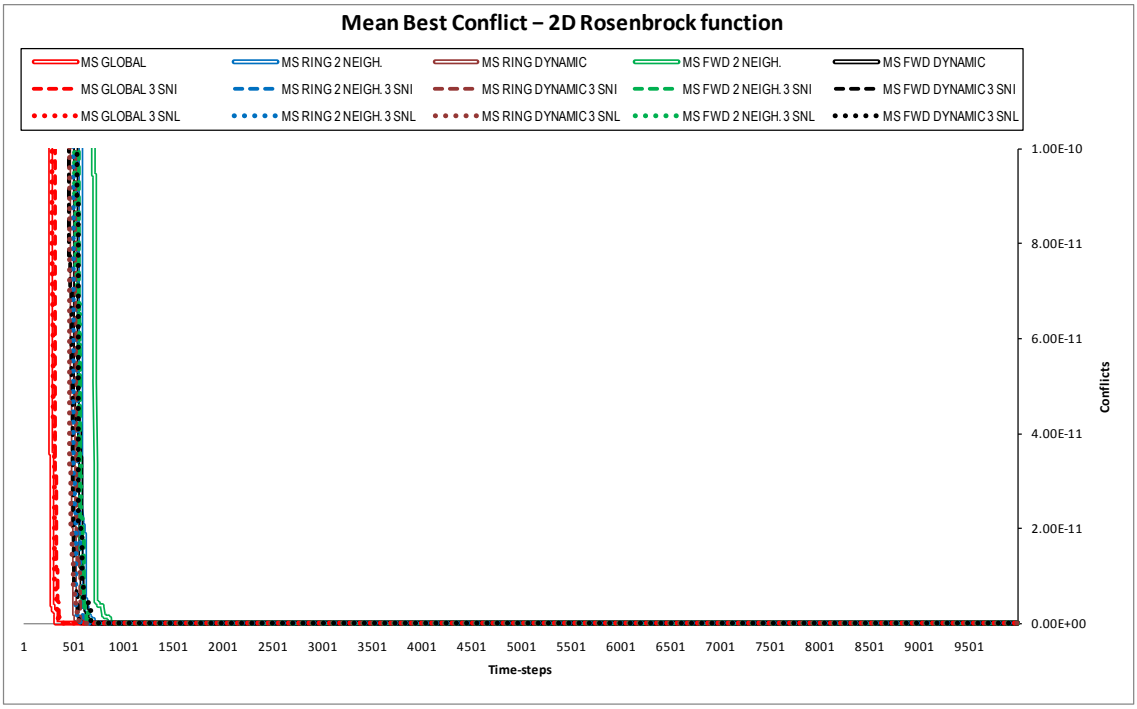


Fig. 7.79. Convergence curves of the mean best conflict for the 2D Rosenbrock function, associated to Table 7.49. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

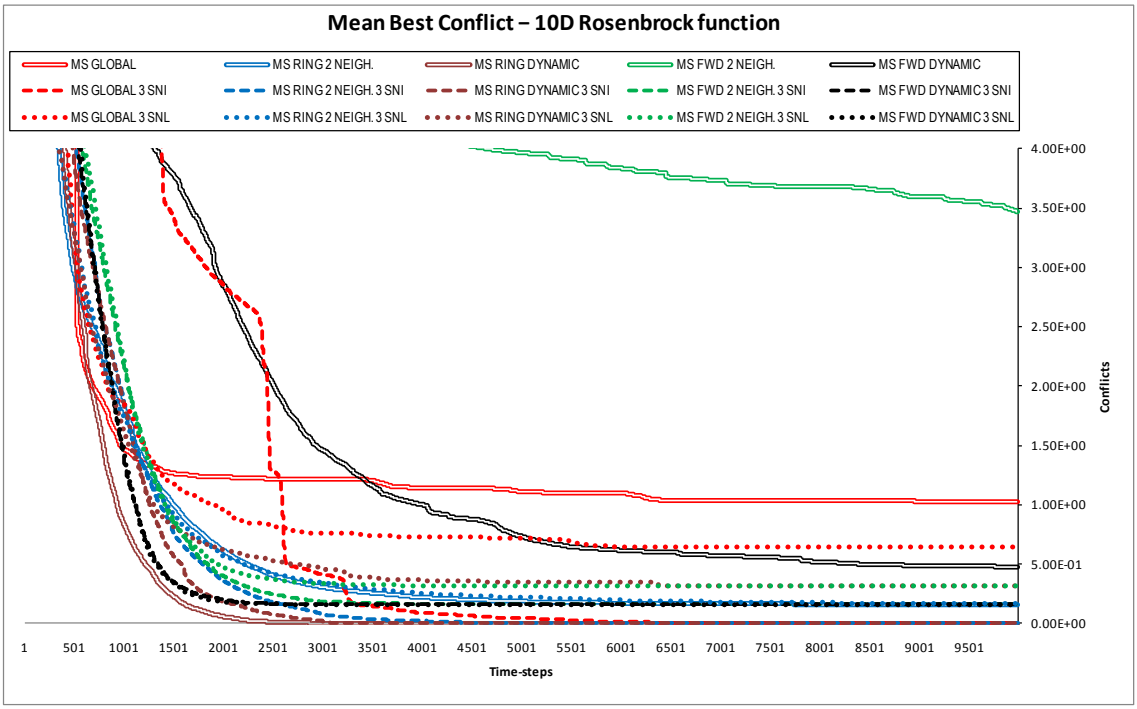


Fig. 7.80. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.50. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

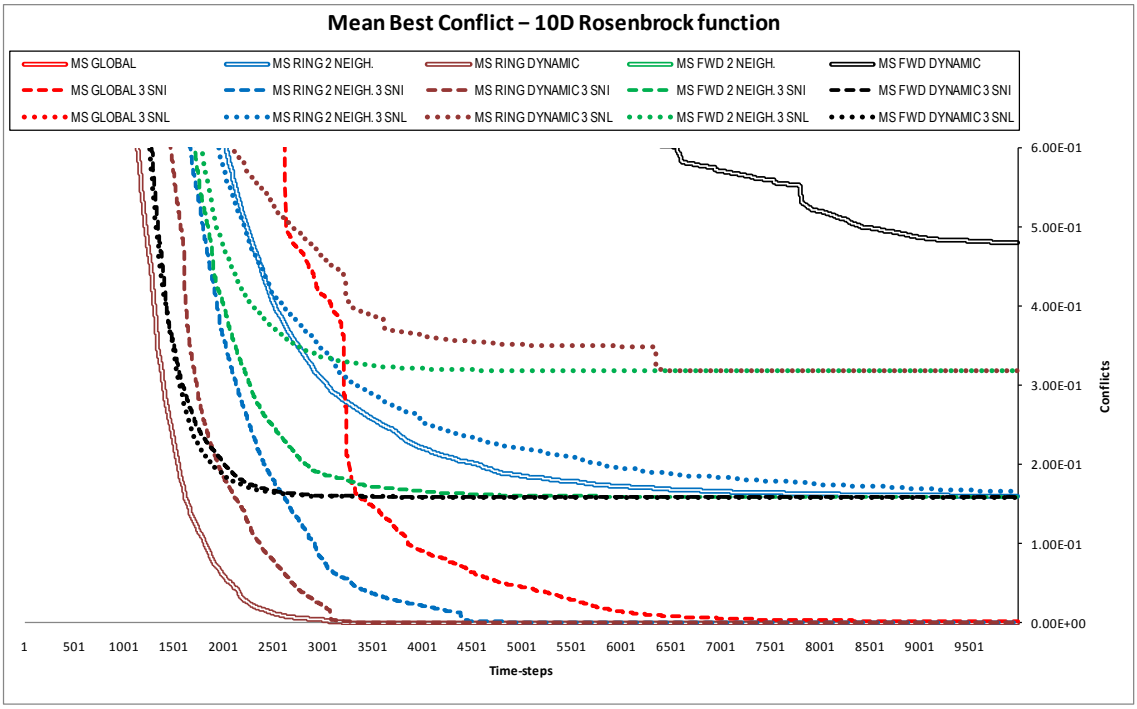


Fig. 7.81. Convergence curves of the mean best conflict for the 10D Rosenbrock function, associated to Table 7.50. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

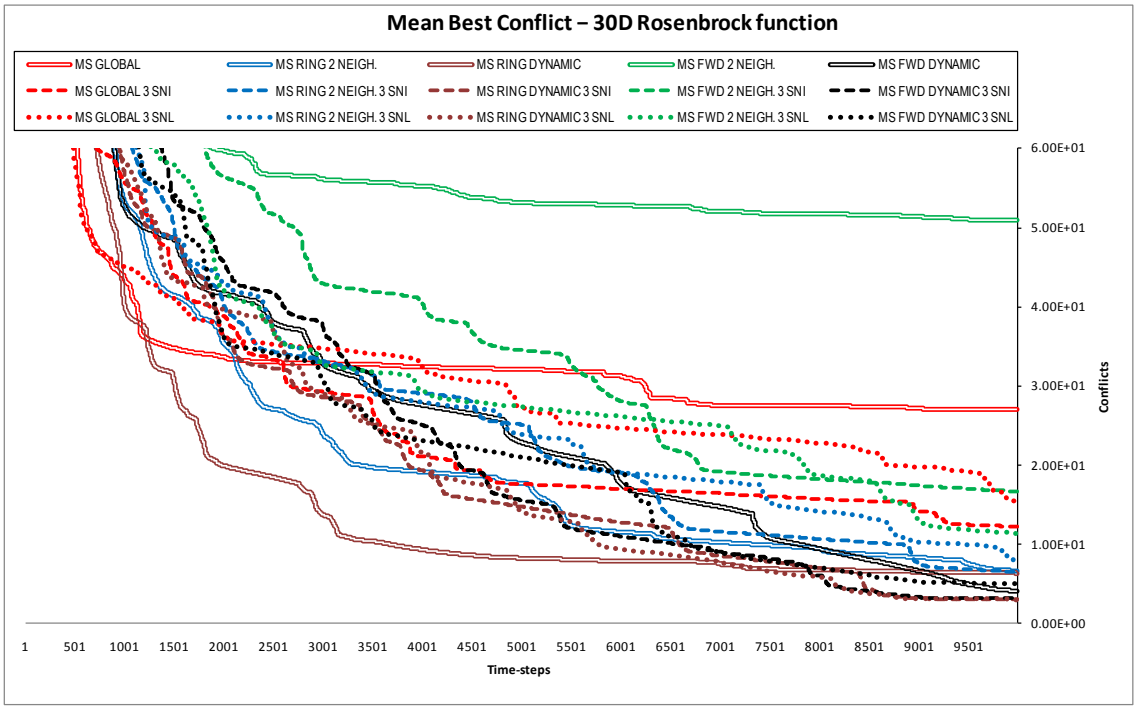


Fig. 7.82. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.51. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

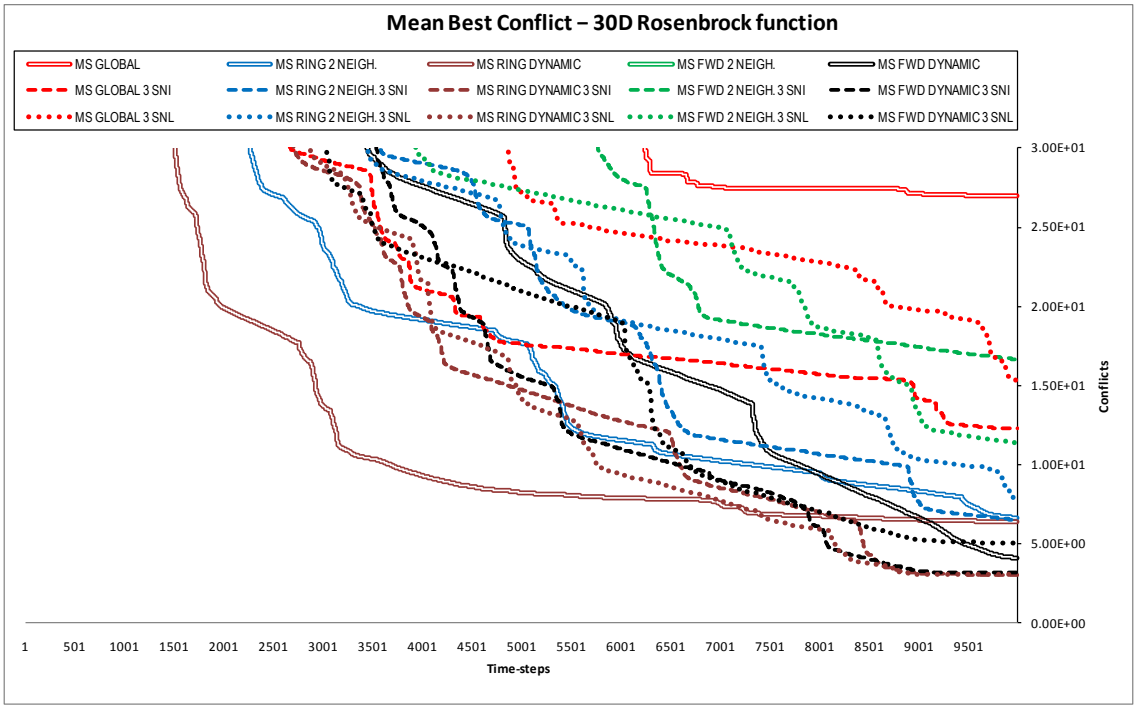


Fig. 7.83. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.51. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

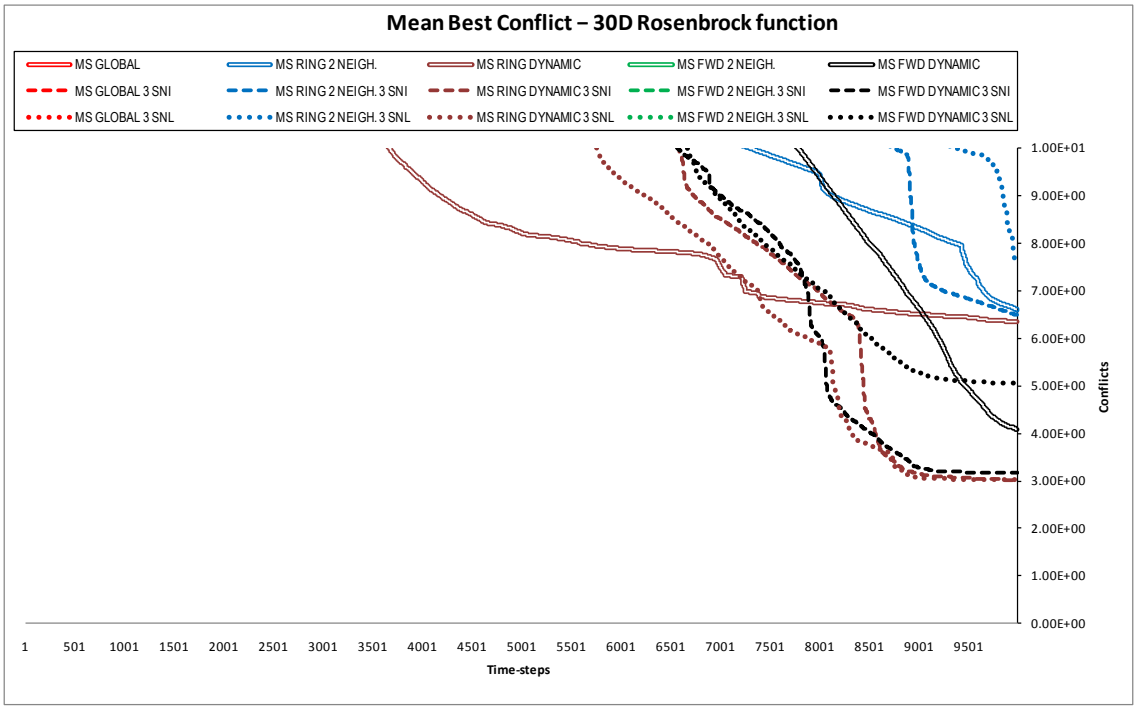


Fig. 7.84. Convergence curves of the mean best conflict for the 30D Rosenbrock function, associated to Table 7.51. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.52. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Rastrigin function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.93E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.73E-11	100
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E-11	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.07E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.95E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.95E-11	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.91E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.91E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.15E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.15E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.95E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.95E-11	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.91E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.91E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.85E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.85E-11	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.42E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.42E-11	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.58E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.58E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.30E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	8.30E-11	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.56E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.56E-11	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.48E-11	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.48E-11	-

Table 7.53. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Rastrigin function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	1.99E+00	3.98E+00	4.78E+00	1.49E+01	1.84E-11	0
			1000	1.99E+00	3.98E+00	5.13E+00	1.49E+01	5.46E-04	-
	RING	nn = 2	10000	0.00E+00	2.98E+00	2.75E+00	6.96E+00	2.90E-02	4
			1000	2.98E+00	5.97E+00	5.72E+00	1.00E+01	3.74E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.68E+00	5.97E+00	1.94E-02	32
			1000	2.18E+00	4.97E+00	5.26E+00	1.09E+01	3.66E-02	-
	FWD	nn = 2	10000	6.41E-05	2.12E+00	2.34E+00	6.20E+00	2.45E-02	4
			1000	2.80E+00	6.48E+00	6.42E+00	1.03E+01	3.42E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.07E+00	2.98E+00	1.66E-02	28
			1000	2.25E+00	5.07E+00	5.03E+00	9.50E+00	3.34E-02	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	0.00E+00	2.98E+00	3.54E+00	8.95E+00	1.57E-02	4
			1000	9.95E-01	4.97E+00	4.71E+00	8.95E+00	2.85E-02	-
	RING	nn = 2	10000	0.00E+00	2.98E+00	2.71E+00	5.97E+00	2.36E-02	12
			1000	2.04E+00	5.97E+00	5.65E+00	7.96E+00	3.25E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	1.99E+00	2.02E+00	4.97E+00	2.42E-02	20
			1000	2.98E+00	4.99E+00	5.35E+00	8.95E+00	3.39E-02	-
	FWD	nn = 2	10000	0.00E+00	1.09E+00	1.64E+00	4.09E+00	2.22E-02	4
			1000	1.99E+00	4.86E+00	4.89E+00	9.14E+00	3.25E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.08E+00	2.98E+00	1.57E-02	28
			1000	1.99E+00	4.86E+00	4.67E+00	8.03E+00	3.32E-02	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	9.95E-01	2.98E+00	3.10E+00	7.96E+00	9.38E-03	0
			1000	1.99E+00	3.98E+00	4.29E+00	7.96E+00	1.92E-02	-
	RING	nn = 2	10000	0.00E+00	1.99E+00	2.40E+00	7.96E+00	2.60E-02	12
			1000	9.95E-01	4.98E+00	5.17E+00	7.96E+00	3.63E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	1.99E+00	1.75E+00	3.98E+00	2.10E-02	20
			1000	9.95E-01	4.97E+00	5.14E+00	8.95E+00	3.54E-02	-
	FWD	nn = 2	10000	0.00E+00	2.21E+00	2.37E+00	6.31E+00	2.40E-02	4
			1000	3.05E+00	4.83E+00	5.42E+00	1.05E+01	3.26E-02	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.95E-01	1.47E+00	4.97E+00	1.47E-02	28
			1000	3.30E+00	5.01E+00	5.62E+00	1.05E+01	3.17E-02	-

Table 7.54. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Rastrigin function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	RASTRIGIN 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	2.59E+01	5.27E+01	5.33E+01	8.16E+01	1.89E-11	0
			1000	2.59E+01	5.27E+01	5.33E+01	8.16E+01	5.36E-08	-
	RING	nn = 2	10000	3.28E+01	4.48E+01	4.56E+01	6.37E+01	3.03E-02	0
			1000	3.32E+01	5.21E+01	4.97E+01	6.57E+01	3.09E-02	-
		nni = 2 nnf = (m – 1)	10000	2.59E+01	3.98E+01	4.32E+01	6.67E+01	2.77E-02	0
			1000	2.72E+01	4.48E+01	4.77E+01	6.83E+01	2.96E-02	-
	FWD	nn = 2	10000	2.96E+01	6.81E+01	6.71E+01	8.76E+01	3.07E-02	0
			1000	4.94E+01	8.76E+01	8.55E+01	1.21E+02	3.31E-02	-
		nni = 2 nnf = (m – 1)	10000	1.69E+01	3.38E+01	3.34E+01	5.67E+01	2.18E-02	0
			1000	3.73E+01	7.11E+01	7.11E+01	9.42E+01	3.10E-02	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	1.79E+01	4.38E+01	4.23E+01	6.37E+01	1.20E-02	0
			1000	1.99E+01	4.88E+01	4.81E+01	7.16E+01	1.68E-02	-
	RING	nn = 2	10000	1.89E+01	4.28E+01	4.32E+01	6.27E+01	3.02E-02	0
			1000	2.46E+01	4.88E+01	4.93E+01	6.47E+01	3.09E-02	-
		nni = 2 nnf = (m – 1)	10000	1.89E+01	3.98E+01	4.25E+01	6.17E+01	3.05E-02	0
			1000	2.29E+01	4.54E+01	4.70E+01	6.47E+01	3.15E-02	-
	FWD	nn = 2	10000	3.30E+01	5.91E+01	6.07E+01	9.46E+01	2.77E-02	0
			1000	3.48E+01	7.70E+01	7.36E+01	1.01E+02	3.18E-02	-
		nni = 2 nnf = (m – 1)	10000	1.89E+01	3.58E+01	3.65E+01	6.17E+01	2.42E-02	0
			1000	3.36E+01	7.00E+01	6.54E+01	9.64E+01	3.02E-02	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	2.29E+01	3.88E+01	4.06E+01	6.67E+01	3.97E-03	0
			1000	2.39E+01	4.08E+01	4.44E+01	7.06E+01	5.67E-03	-
	RING	nn = 2	10000	2.19E+01	4.18E+01	4.38E+01	6.77E+01	2.90E-02	0
			1000	2.20E+01	4.74E+01	4.75E+01	7.21E+01	3.04E-02	-
		nni = 2 nnf = (m – 1)	10000	1.79E+01	4.18E+01	4.14E+01	6.47E+01	2.88E-02	0
			1000	1.97E+01	4.58E+01	4.55E+01	7.36E+01	3.07E-02	-
	FWD	nn = 2	10000	2.26E+01	4.87E+01	5.12E+01	9.34E+01	2.21E-02	0
			1000	3.02E+01	6.91E+01	6.58E+01	1.02E+02	2.56E-02	-
		nni = 2 nnf = (m – 1)	10000	1.79E+01	3.38E+01	3.47E+01	6.17E+01	1.92E-02	0
			1000	2.49E+01	5.88E+01	6.00E+01	1.02E+02	2.43E-02	-

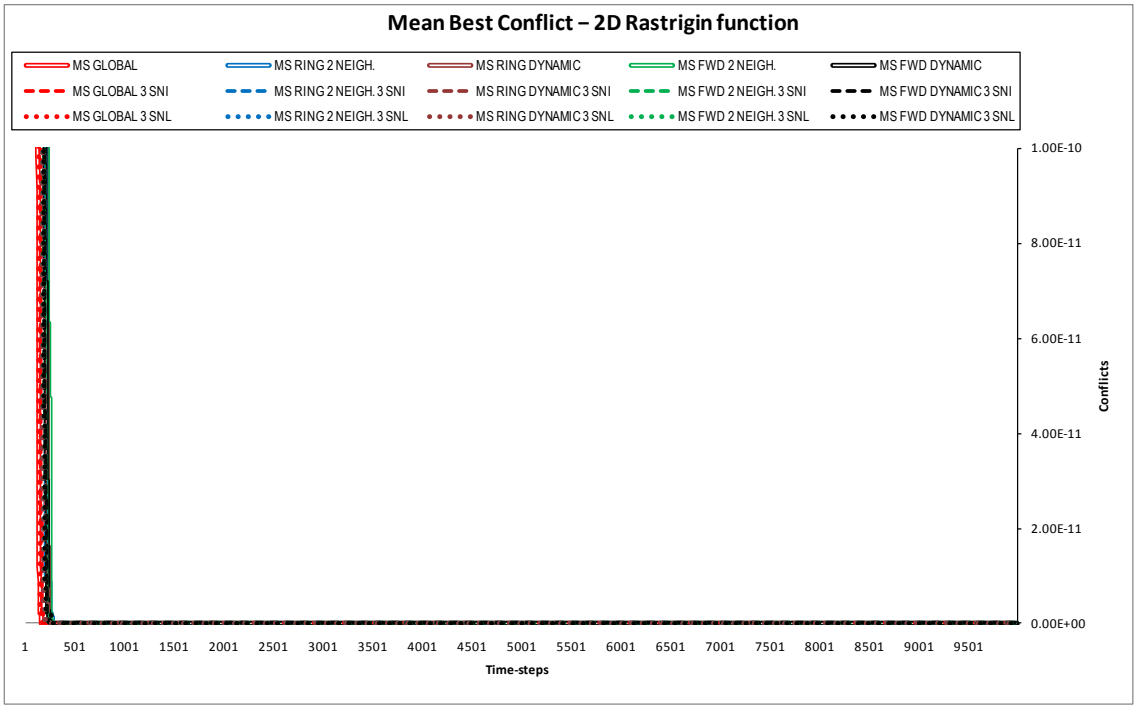


Fig. 7.85. Convergence curves of the mean best conflict for the 2D Rastrigin function, associated to Table 7.52. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

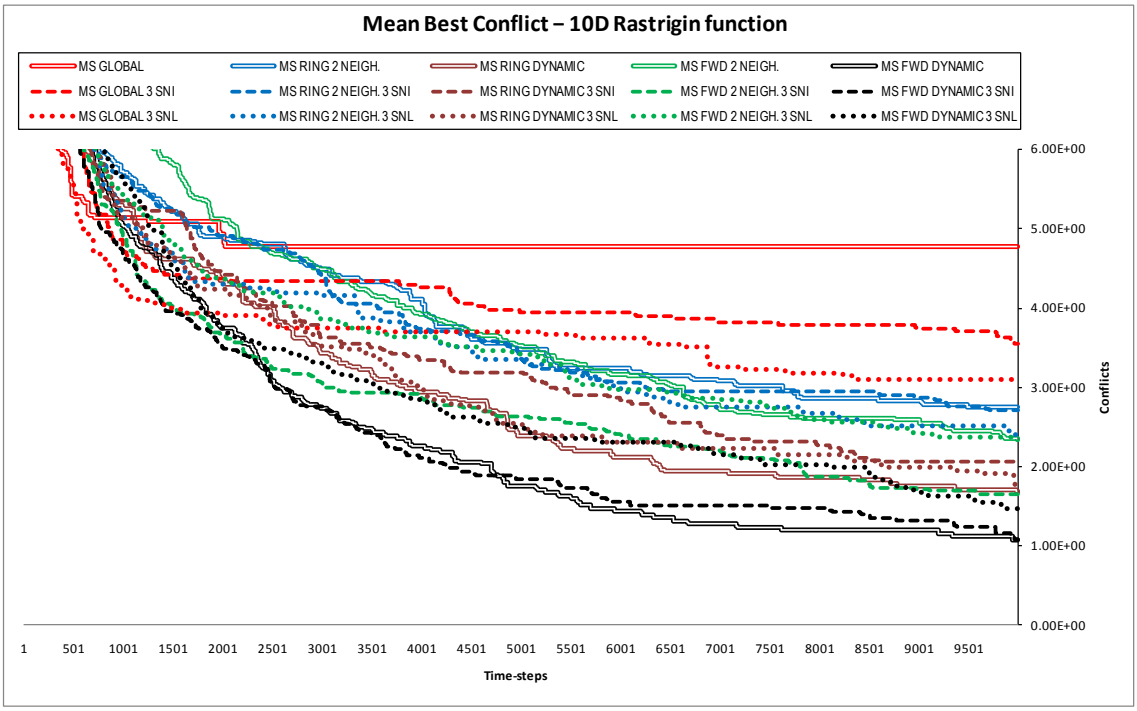


Fig. 7.86. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.53. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

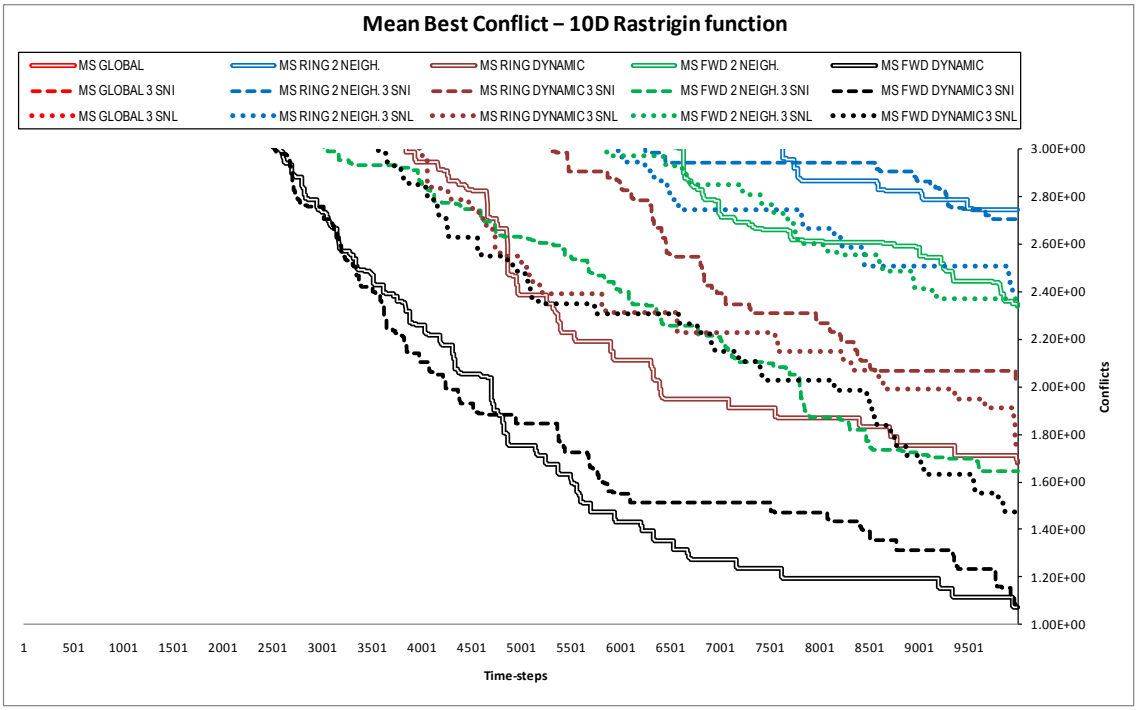


Fig. 7.87. Convergence curves of the mean best conflict for the 10D Rastrigin function, associated to Table 7.53. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

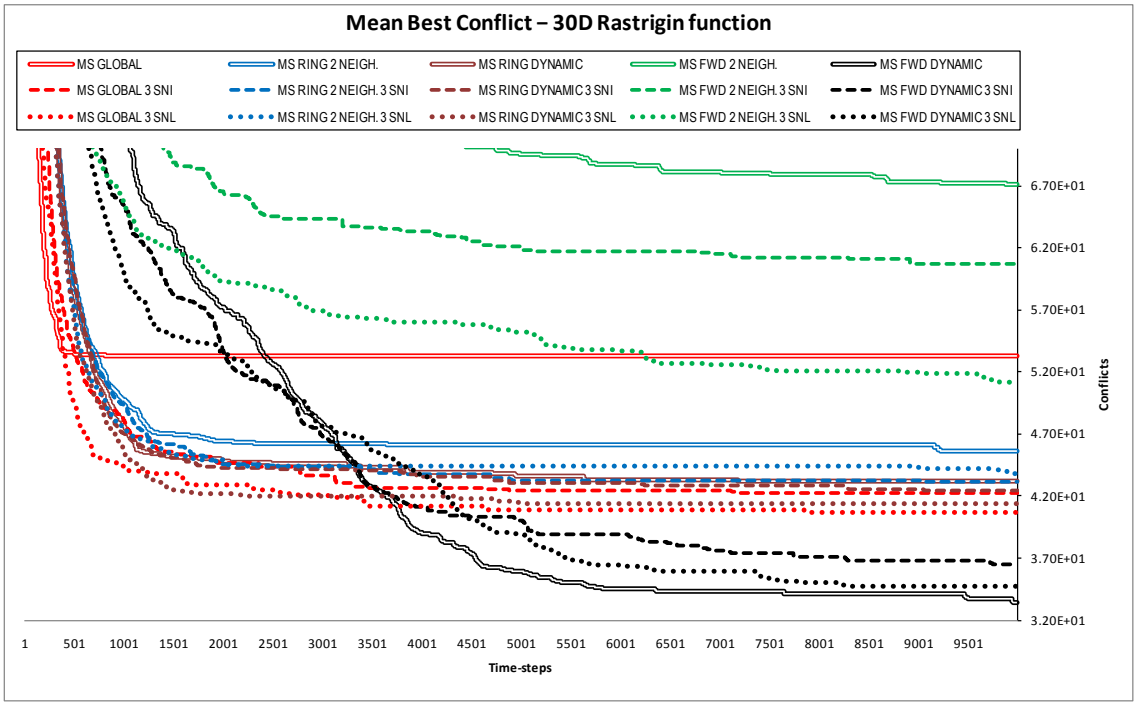


Fig. 7.88. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.54. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

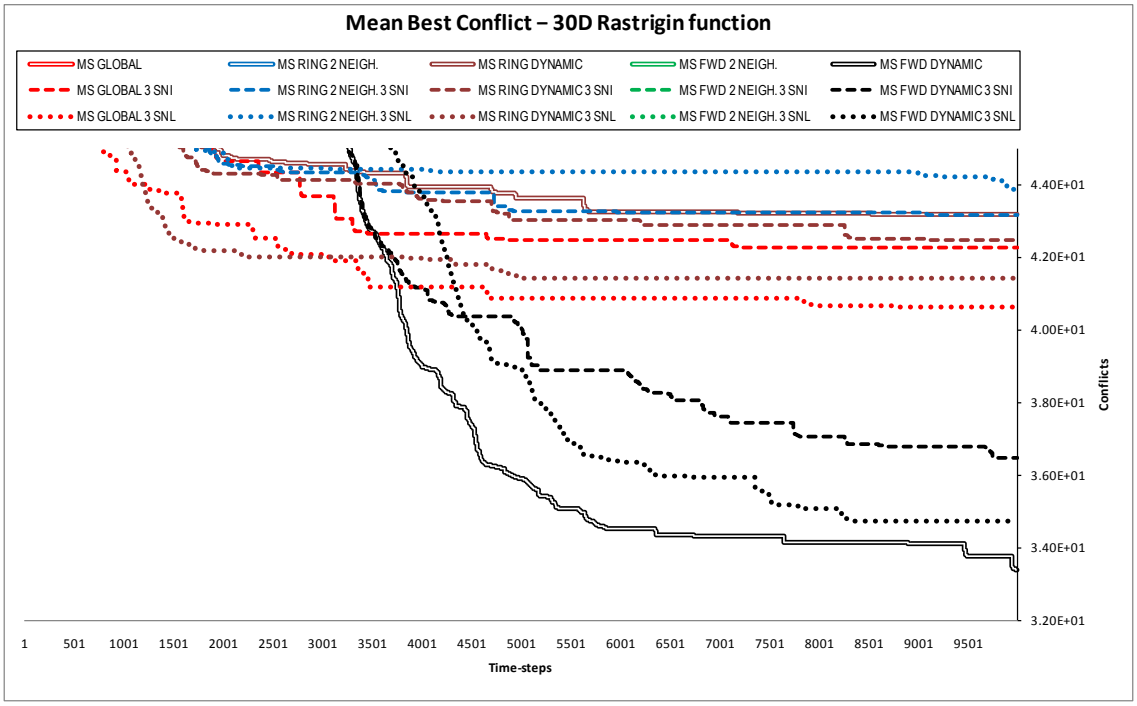


Fig. 7.89. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.54. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

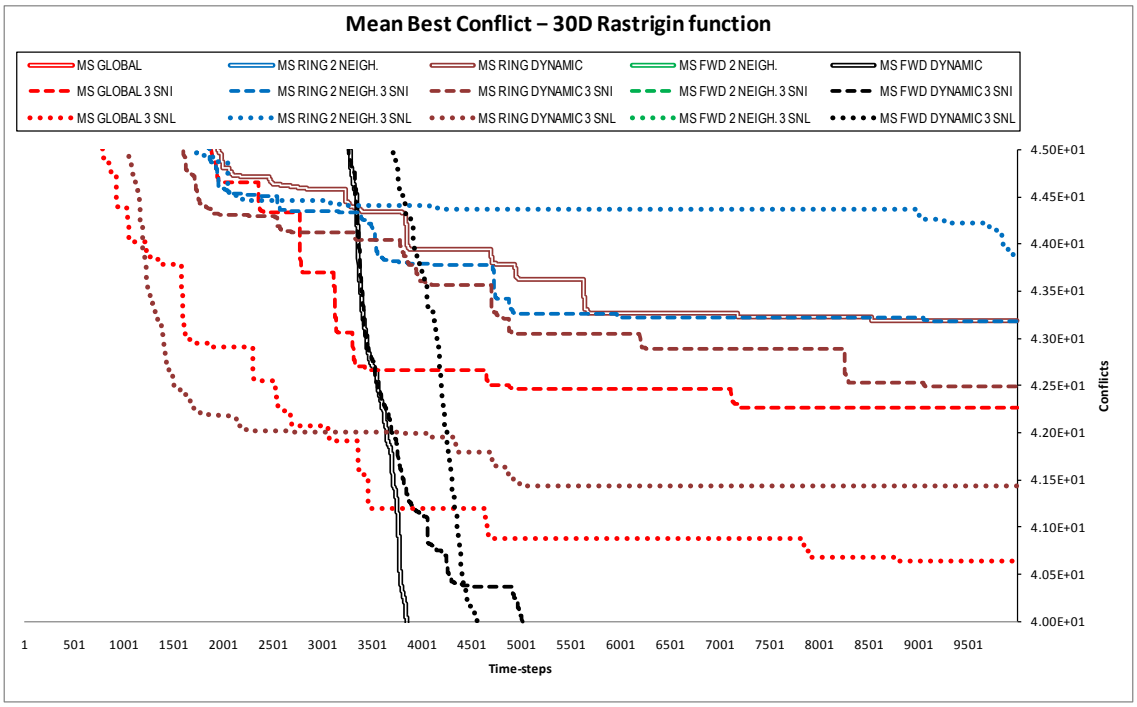


Fig. 7.90. Convergence curves of the mean best conflict for the 30D Rastrigin function, associated to Table 7.54. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.55. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Griewank function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.56E-12	100
			1000	0.00E+00	0.00E+00	5.95E-16	1.49E-14	8.92E-04	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.51E-04	100
			1000	0.00E+00	0.00E+00	2.39E-05	5.88E-04	2.05E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.74E-12	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.59E-03	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.67E-04	100
			1000	0.00E+00	0.00E+00	4.02E-08	8.51E-07	1.92E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.88E-12	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.53E-03	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	2.96E-04	7.40E-03	6.52E-12	96
			1000	0.00E+00	0.00E+00	4.49E-04	7.40E-03	1.36E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	5.52E-05	100
			1000	0.00E+00	0.00E+00	5.86E-07	1.46E-05	1.83E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.92E-12	100
			1000	0.00E+00	0.00E+00	4.18E-09	1.05E-07	1.70E-03	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.44E-05	100
			1000	0.00E+00	0.00E+00	1.14E-14	2.71E-13	1.80E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.10E-12	100
			1000	0.00E+00	0.00E+00	5.81E-11	1.10E-09	1.71E-03	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	2.96E-04	7.40E-03	6.17E-12	96
			1000	0.00E+00	0.00E+00	2.96E-04	7.40E-03	1.20E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.13E-05	100
			1000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.94E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.15E-12	100
			1000	0.00E+00	0.00E+00	2.91E-08	7.27E-07	1.78E-03	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	5.05E-05	100
			1000	0.00E+00	0.00E+00	3.45E-08	7.30E-07	1.81E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.37E-12	100
			1000	0.00E+00	0.00E+00	4.15E-09	1.04E-07	1.72E-03	-

Table 7.56. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Griewank function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	1.48E-02	6.64E-02	6.64E-02	1.38E-01	1.48E-05	0
			1000	2.95E-02	7.13E-02	7.85E-02	1.85E-01	1.40E-04	-
	RING	nn = 2	10000	0.00E+00	1.97E-02	2.15E-02	5.90E-02	1.60E-03	8
			1000	7.40E-03	2.22E-02	2.71E-02	5.90E-02	1.75E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.46E-02	2.77E-02	6.89E-02	1.47E-03	8
			1000	0.00E+00	3.45E-02	3.59E-02	7.38E-02	1.74E-03	-
	FWD	nn = 2	10000	4.64E-05	3.60E-02	3.64E-02	9.76E-02	1.30E-03	4
			1000	2.16E-02	6.01E-02	6.36E-02	1.49E-01	1.53E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	1.72E-02	2.33E-02	5.90E-02	1.08E-03	12
			1000	7.94E-03	5.59E-02	6.33E-02	1.25E-01	1.73E-03	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	7.40E-03	4.92E-02	4.83E-02	1.23E-01	7.45E-04	0
			1000	9.86E-03	5.66E-02	5.90E-02	1.23E-01	9.86E-04	-
	RING	nn = 2	10000	0.00E+00	3.20E-02	2.98E-02	7.63E-02	1.66E-03	12
			1000	7.40E-03	3.69E-02	3.80E-02	7.87E-02	1.84E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	2.46E-02	2.84E-02	7.87E-02	1.54E-03	4
			1000	7.40E-03	3.54E-02	3.90E-02	9.60E-02	1.82E-03	-
	FWD	nn = 2	10000	7.73E-10	2.72E-02	3.12E-02	1.12E-01	1.28E-03	4
			1000	7.43E-03	5.00E-02	6.08E-02	1.30E-01	1.50E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	1.97E-02	2.59E-02	1.01E-01	1.05E-03	12
			1000	4.03E-03	5.25E-02	5.21E-02	1.30E-01	1.53E-03	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	1.72E-02	4.43E-02	4.99E-02	9.59E-02	6.21E-04	0
			1000	3.20E-02	5.65E-02	6.13E-02	1.50E-01	9.17E-04	-
	RING	nn = 2	10000	0.00E+00	2.46E-02	2.40E-02	6.15E-02	1.23E-03	16
			1000	3.44E-15	2.46E-02	2.97E-02	7.36E-02	1.43E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	1.97E-02	2.38E-02	8.12E-02	1.13E-03	20
			1000	7.77E-16	2.46E-02	3.04E-02	8.12E-02	1.39E-03	-
	FWD	nn = 2	10000	1.78E-04	2.56E-02	2.78E-02	6.54E-02	1.34E-03	0
			1000	1.91E-02	6.20E-02	5.89E-02	9.95E-02	1.69E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	1.97E-02	1.80E-02	5.20E-02	9.81E-04	20
			1000	2.30E-02	6.02E-02	6.21E-02	1.57E-01	1.61E-03	-

Table 7.57. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Griewank function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	GRIEWANK 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	0.00E+00	4.67E-02	5.18E-02	1.41E-01	2.42E-12	4
			1000	6.66E-16	4.67E-02	5.18E-02	1.41E-01	4.06E-08	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	2.17E-03	1.23E-02	1.10E-06	76
			1000	6.02E-08	7.40E-03	6.97E-03	3.92E-02	3.08E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	6.39E-03	3.92E-02	4.12E-12	64
			1000	9.47E-11	7.40E-03	9.54E-03	3.92E-02	1.26E-05	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	3.07E-04	7.40E-03	2.58E-06	92
			1000	6.00E-03	4.20E-02	5.62E-02	2.12E-01	1.19E-04	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	2.07E-03	2.71E-02	3.36E-12	84
			1000	9.39E-06	1.49E-03	7.82E-03	3.60E-02	2.34E-05	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	0.00E+00	9.86E-03	1.37E-02	4.92E-02	5.43E-07	28
			1000	4.00E-15	9.86E-03	1.44E-02	4.92E-02	1.06E-06	-
	RING	nn = 2	10000	0.00E+00	4.44E-16	9.04E-03	4.17E-02	5.37E-07	52
			1000	4.78E-09	7.40E-03	9.93E-03	4.17E-02	1.41E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	7.40E-03	1.01E-02	4.17E-02	4.12E-12	44
			1000	2.22E-10	7.40E-03	1.07E-02	4.17E-02	9.69E-06	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	7.77E-03	5.65E-02	1.31E-09	56
			1000	1.46E-09	1.23E-02	1.79E-02	7.09E-02	1.38E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	7.40E-03	1.13E-02	6.34E-02	3.60E-12	48
			1000	1.75E-10	1.23E-02	1.53E-02	6.82E-02	8.68E-06	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	0.00E+00	1.72E-02	3.40E-02	2.12E-01	3.03E-12	24
			1000	4.44E-16	1.72E-02	3.62E-02	2.12E-01	2.84E-07	-
	RING	nn = 2	10000	0.00E+00	7.40E-03	1.13E-02	5.66E-02	8.88E-07	40
			1000	1.17E-09	9.86E-03	1.58E-02	6.40E-02	2.00E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	9.86E-03	1.26E-02	5.66E-02	5.95E-08	28
			1000	8.58E-11	9.87E-03	1.51E-02	5.66E-02	5.53E-06	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	1.04E-02	6.34E-02	3.31E-08	64
			1000	1.62E-09	7.14E-05	1.40E-02	6.34E-02	1.78E-05	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	7.40E-03	1.43E-02	6.34E-02	3.93E-12	48
			1000	1.23E-10	7.40E-03	1.43E-02	6.34E-02	9.04E-07	-

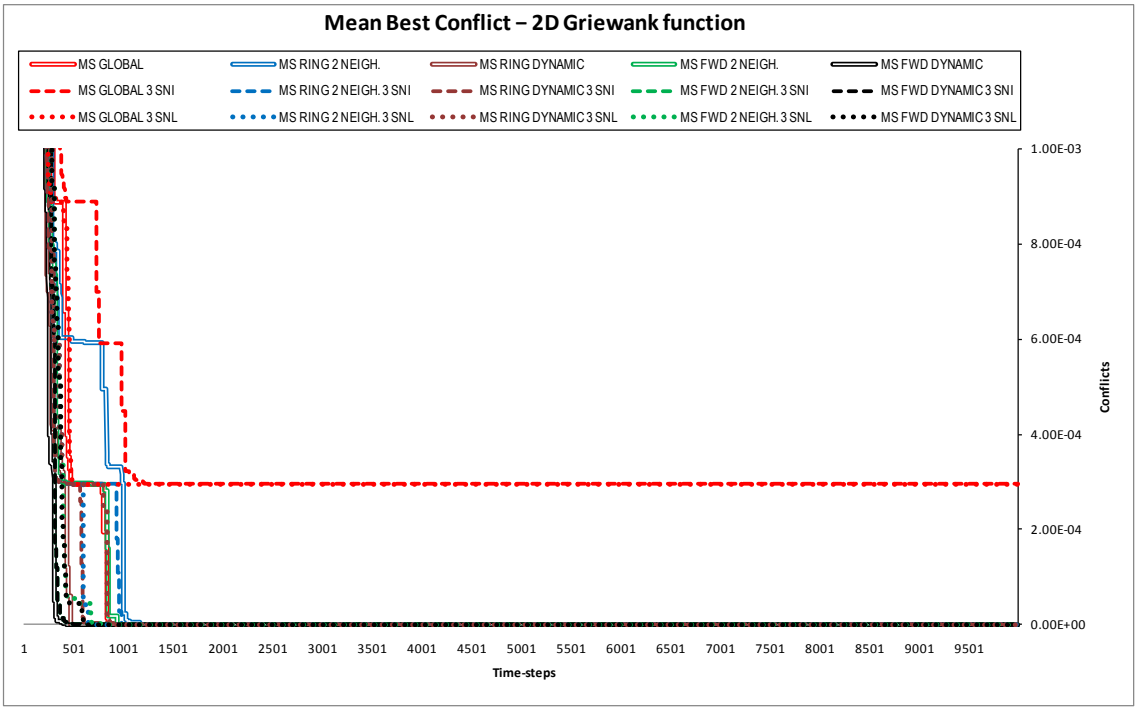


Fig. 7.91. Convergence curves of the mean best conflict for the 2D Griewank function, associated to Table 7.55. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

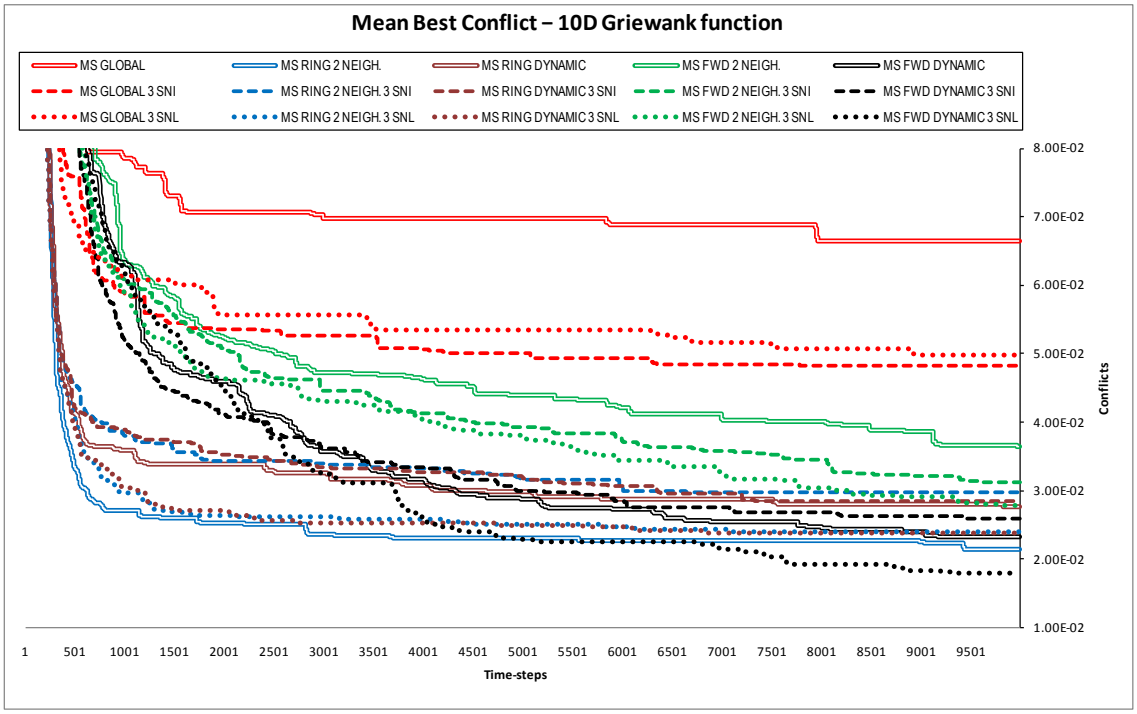


Fig. 7.92. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.56. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

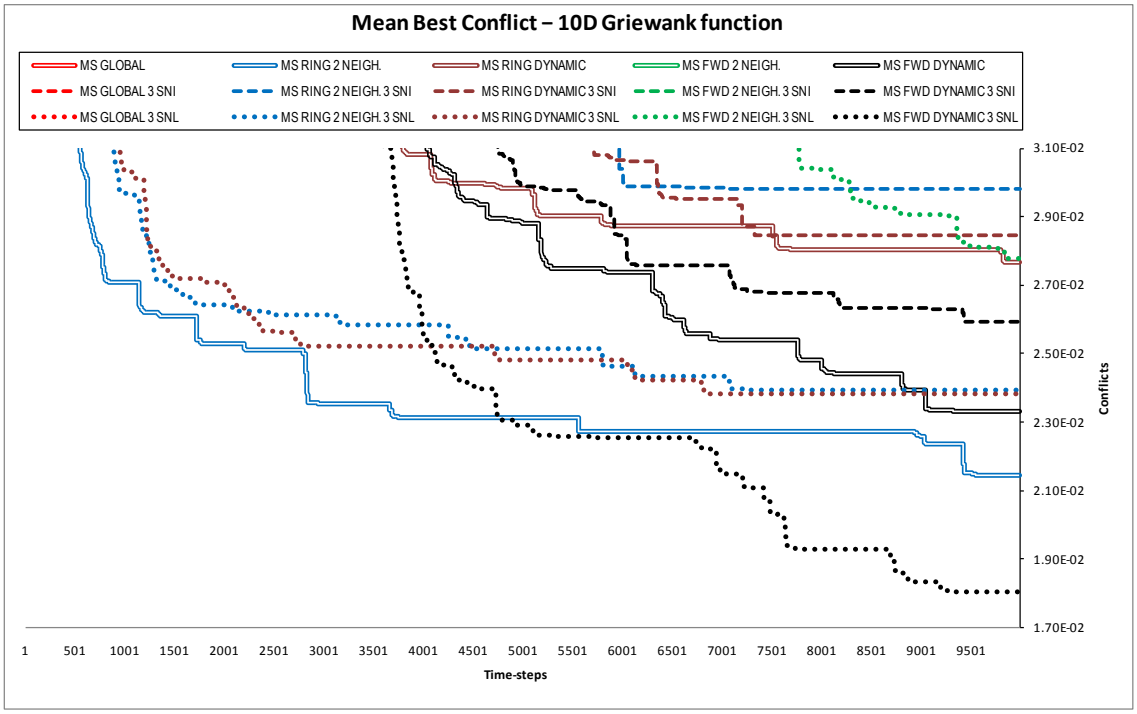


Fig. 7.93. Convergence curves of the mean best conflict for the 10D Griewank function, associated to Table 7.56. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

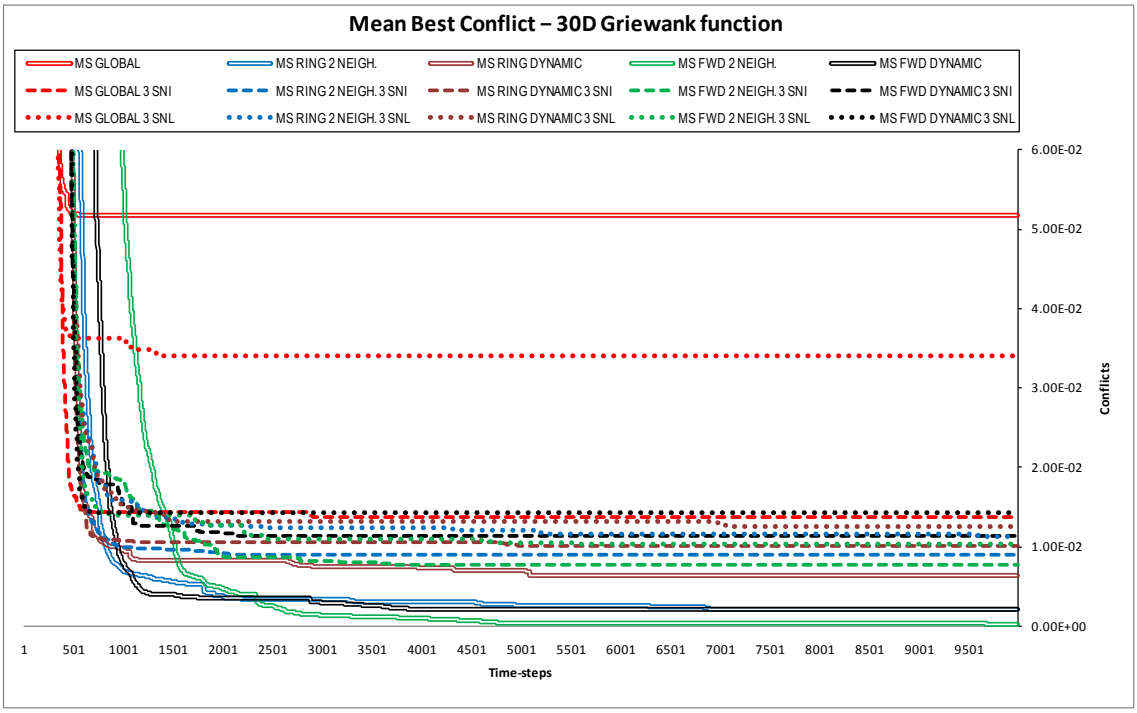


Fig. 7.94. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.57. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

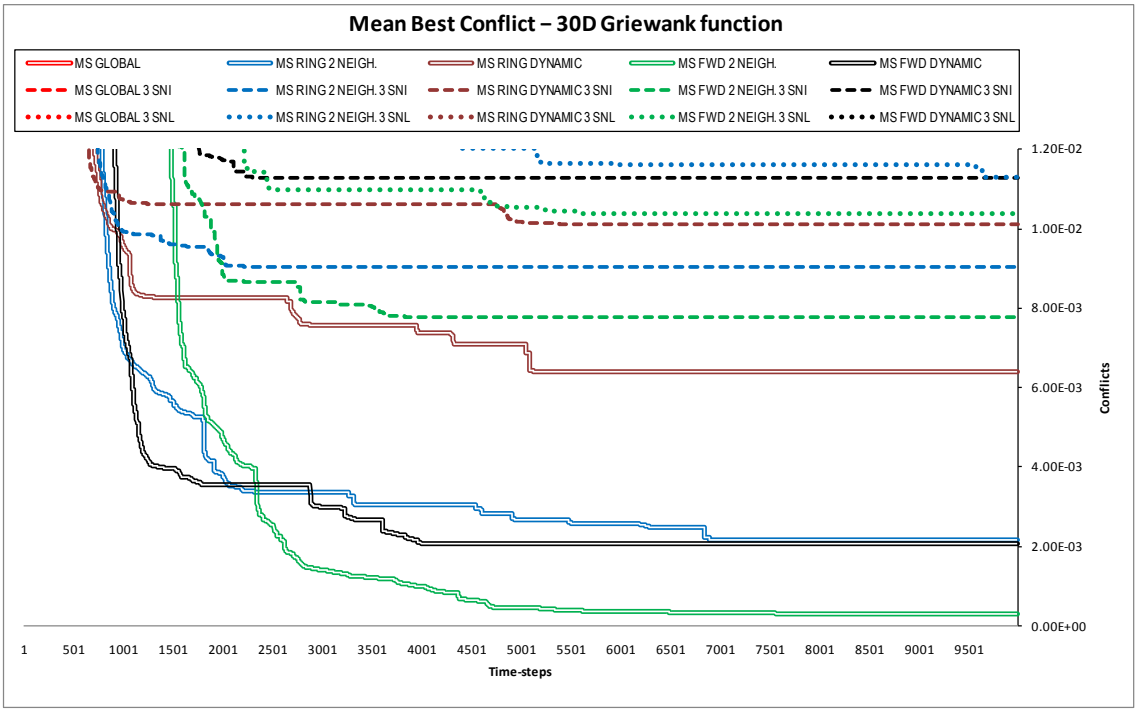


Fig. 7.95. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.57. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

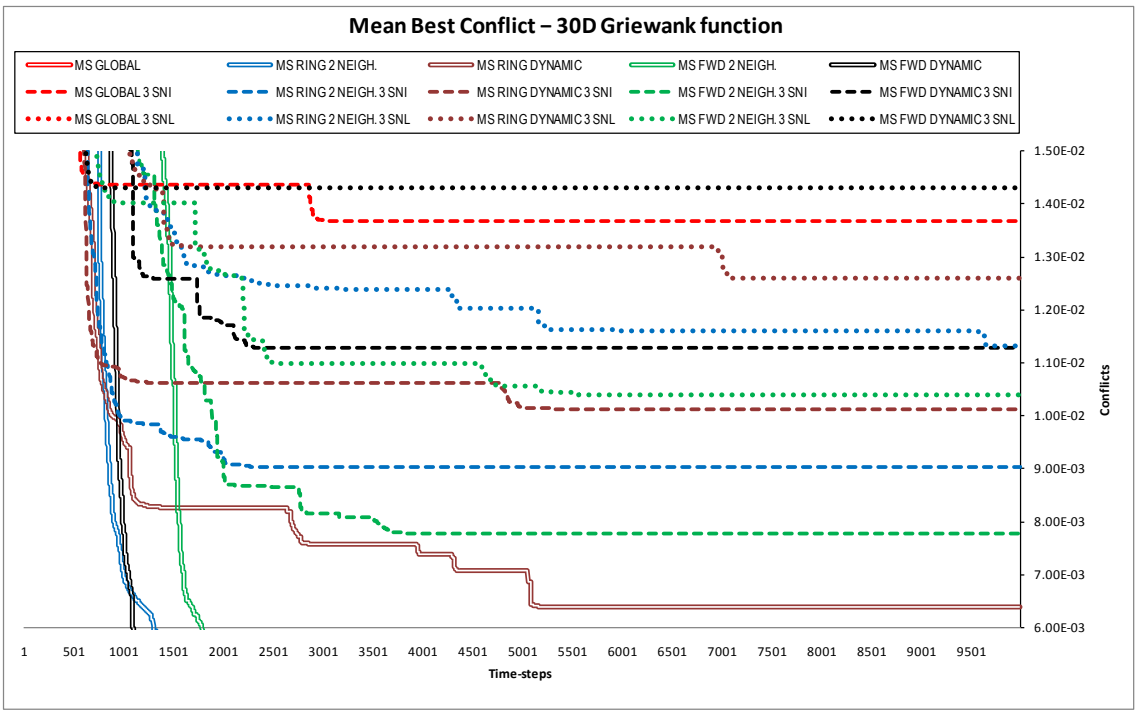


Fig. 7.96. Convergence curves of the mean best conflict for the 30D Griewank function, associated to Table 7.57. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Table 7.58. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 2-dimensional Schaffer f6 function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 2D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	0.00E+00	0.00E+00	2.33E-03	9.72E-03	1.05E-04	76
			1000	0.00E+00	0.00E+00	3.11E-03	9.72E-03	2.58E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.13E-04	100
			1000	0.00E+00	0.00E+00	7.84E-04	9.72E-03	7.34E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.36E-11	100
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	6.25E-03	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.80E-04	100
			1000	0.00E+00	5.94E-14	8.08E-04	9.72E-03	6.89E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.13E-11	100
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	5.17E-03	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	7.77E-04	9.72E-03	1.12E-03	92
			1000	0.00E+00	0.00E+00	1.94E-03	9.72E-03	5.80E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.23E-11	100
			1000	0.00E+00	0.00E+00	2.01E-04	4.70E-03	6.79E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.21E-11	100
			1000	0.00E+00	0.00E+00	2.01E-04	4.70E-03	6.44E-03	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.56E-10	100
			1000	0.00E+00	5.77E-15	7.79E-04	9.72E-03	6.67E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.25E-11	100
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	5.99E-03	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.31E-11	100
			1000	0.00E+00	0.00E+00	1.25E-03	9.72E-03	4.62E-03	-
	RING	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.53E-05	100
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	6.97E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.29E-11	100
			1000	0.00E+00	0.00E+00	3.89E-04	9.72E-03	6.55E-03	-
	FWD	nn = 2	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.30E-11	100
			1000	0.00E+00	0.00E+00	7.86E-04	9.72E-03	6.37E-03	-
		nni = 2 nnf = (m – 1)	10000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.31E-11	100
			1000	0.00E+00	0.00E+00	4.65E-04	9.72E-03	5.98E-03	-

Table 7.59. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 10-dimensional Schaffer f6 function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 10D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	9.72E-03	3.72E-02	2.95E-02	3.72E-02	3.18E-04	0
			1000	9.72E-03	3.72E-02	3.06E-02	3.72E-02	6.67E-04	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	2.03E-03	0
			1000	9.72E-03	9.72E-03	1.74E-02	3.72E-02	3.03E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.76E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	1.95E-03	-
	FWD	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.70E-03	0
			1000	9.72E-03	9.72E-03	1.41E-02	3.72E-02	2.41E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.59E-03	0
			1000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	1.79E-03	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	9.72E-03	9.72E-03	1.52E-02	3.72E-02	1.73E-03	0
			1000	9.72E-03	9.72E-03	1.74E-02	3.72E-02	2.22E-03	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.81E-03	0
			1000	9.72E-03	9.72E-03	1.96E-02	3.72E-02	2.60E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.70E-03	0
			1000	9.72E-03	9.72E-03	1.30E-02	3.72E-02	2.21E-03	-
	FWD	nn = 2	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.69E-03	0
			1000	9.72E-03	9.72E-03	1.30E-02	3.72E-02	2.30E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.82E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	1.85E-03	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	9.72E-03	9.72E-03	1.30E-02	3.72E-02	1.35E-03	0
			1000	9.72E-03	9.72E-03	1.52E-02	3.72E-02	1.49E-03	-
	RING	nn = 2	10000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	1.81E-03	0
			1000	9.72E-03	9.72E-03	1.52E-02	3.72E-02	2.52E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.62E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	2.05E-03	-
	FWD	nn = 2	10000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	1.74E-03	0
			1000	9.72E-03	9.72E-03	1.08E-02	3.72E-02	2.26E-03	-
		nni = 2 nnf = (m – 1)	10000	9.72E-03	9.72E-03	9.72E-03	9.72E-03	1.80E-03	0
			1000	9.72E-03	9.72E-03	1.19E-02	3.72E-02	2.13E-03	-

Table 7.60. Statistical results out of 25 runs for a Multi-Swarm algorithm with three sub-neighbourhoods (one per sub-swarm) optimizing the 30-dimensional Schaffer f6 function. The sub-neighbourhoods tested are the GLOBAL, the RING, and the FOWARD structures with 2 neighbours and with linearly increasing number of neighbours (from 2 to 'swarm-size – 1'). Two types of interconnections between sub-neighbourhoods are tested, the 'individual' and the 'local' overlapping. The results for the case without sub-neighbourhoods are imported from previous sections for reference and comparison purposes. A run with an error no greater than 0.0001 is regarded as successful.

OPTIMIZER	NEIGHBOURHOOD STRUCTURE		Time-steps	SCHAFFER F6 30D				OPTIMUM = 0	
				BEST	MEDIAN	MEAN	WORST	MEAN PB_ME	[%] Success
Multi-Swarm	GLOBAL		10000	7.82E-02	1.78E-01	1.86E-01	2.73E-01	2.30E-04	0
			1000	1.27E-01	1.78E-01	1.93E-01	2.73E-01	5.26E-04	-
	RING	nn = 2	10000	3.72E-02	7.82E-02	7.45E-02	1.27E-01	1.58E-03	0
			1000	1.27E-01	1.78E-01	1.68E-01	2.28E-01	3.42E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.38E-02	7.82E-02	5.48E-04	0
			1000	3.74E-02	1.27E-01	1.08E-01	1.78E-01	1.89E-03	-
	FWD	nn = 2	10000	3.72E-02	7.82E-02	7.29E-02	1.27E-01	1.70E-03	0
			1000	7.82E-02	1.27E-01	1.44E-01	1.78E-01	2.83E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.38E-02	7.82E-02	6.55E-04	0
			1000	3.72E-02	7.82E-02	9.05E-02	1.27E-01	1.79E-03	-
Multi-Swarm 3 Sub-Neigh. Indiv. Overlap.	GLOBAL		10000	7.82E-02	7.82E-02	9.98E-02	1.78E-01	1.21E-03	0
			1000	7.82E-02	1.27E-01	1.22E-01	1.78E-01	1.56E-03	-
	RING	nn = 2	10000	3.72E-02	7.82E-02	6.87E-02	1.27E-01	1.40E-03	0
			1000	8.05E-02	1.27E-01	1.50E-01	2.28E-01	2.67E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.05E-02	7.82E-02	7.57E-04	0
			1000	7.82E-02	1.27E-01	1.20E-01	1.78E-01	2.31E-03	-
	FWD	nn = 2	10000	3.72E-02	7.82E-02	6.54E-02	1.27E-01	1.58E-03	0
			1000	7.82E-02	1.27E-01	1.23E-01	1.78E-01	1.99E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	5.52E-02	7.82E-02	7.79E-04	0
			1000	7.82E-02	1.27E-01	1.10E-01	1.27E-01	1.71E-03	-
Multi-Swarm 3 Sub-Neigh. Local Overlap.	GLOBAL		10000	3.72E-02	7.82E-02	9.35E-02	1.78E-01	7.73E-04	0
			1000	7.82E-02	1.27E-01	1.11E-01	1.78E-01	1.17E-03	-
	RING	nn = 2	10000	3.72E-02	7.82E-02	6.67E-02	7.82E-02	1.25E-03	0
			1000	7.84E-02	1.78E-01	1.52E-01	1.78E-01	2.65E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	4.21E-02	7.82E-02	7.46E-04	0
			1000	7.82E-02	1.27E-01	1.30E-01	1.78E-01	1.68E-03	-
	FWD	nn = 2	10000	3.72E-02	3.72E-02	5.03E-02	7.82E-02	1.20E-03	0
			1000	7.82E-02	1.27E-01	1.23E-01	1.78E-01	1.99E-03	-
		nni = 2 nnf = (m – 1)	10000	3.72E-02	3.72E-02	5.23E-02	1.27E-01	7.84E-04	0
			1000	7.82E-02	1.27E-01	1.12E-01	1.78E-01	2.07E-03	-

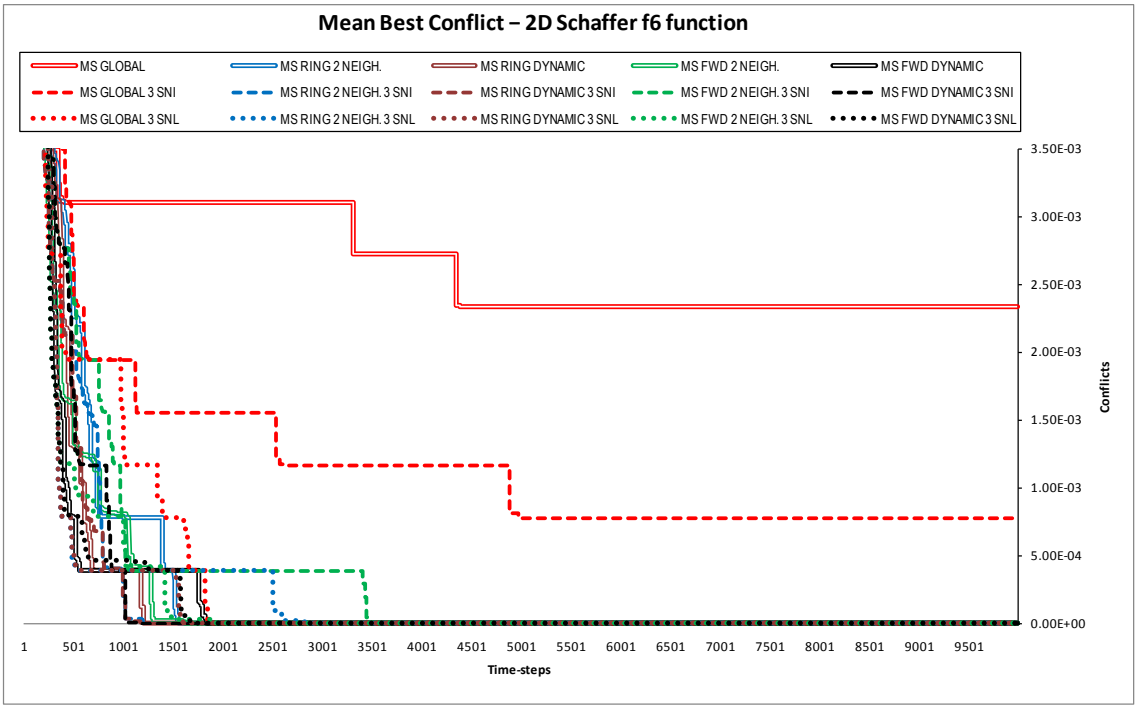


Fig. 7.97. Convergence curves of the mean best conflict for the 2D Schaffer f6 function, associated to Table 7.58. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

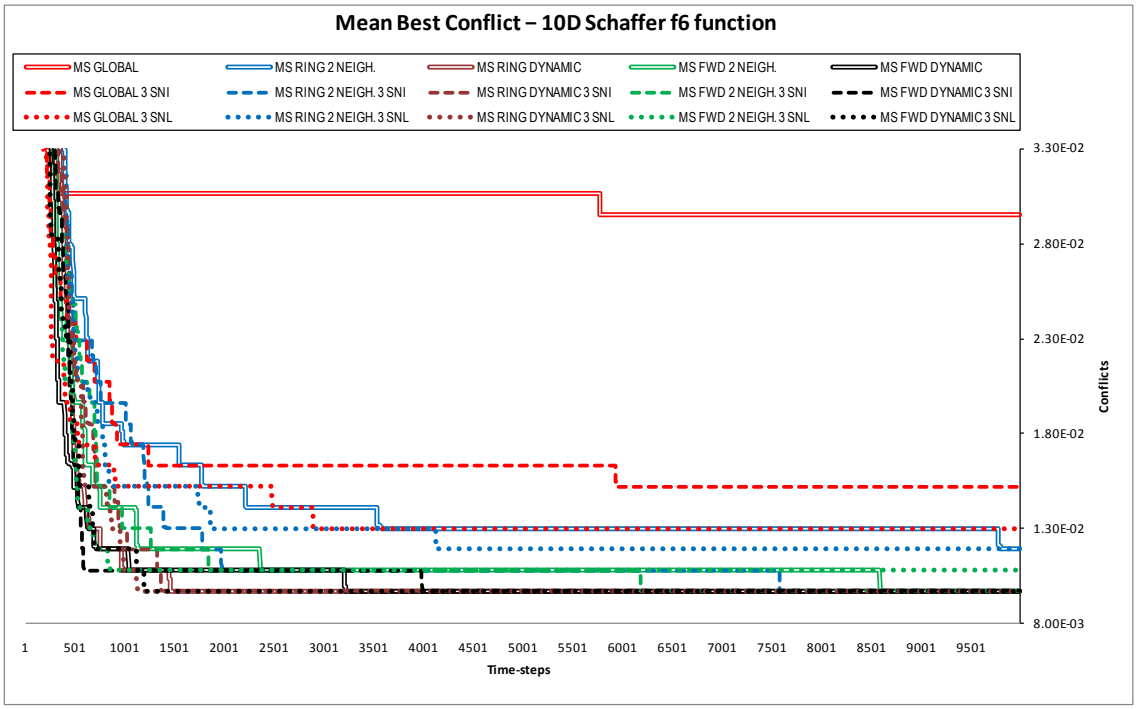


Fig. 7.98. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.59. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

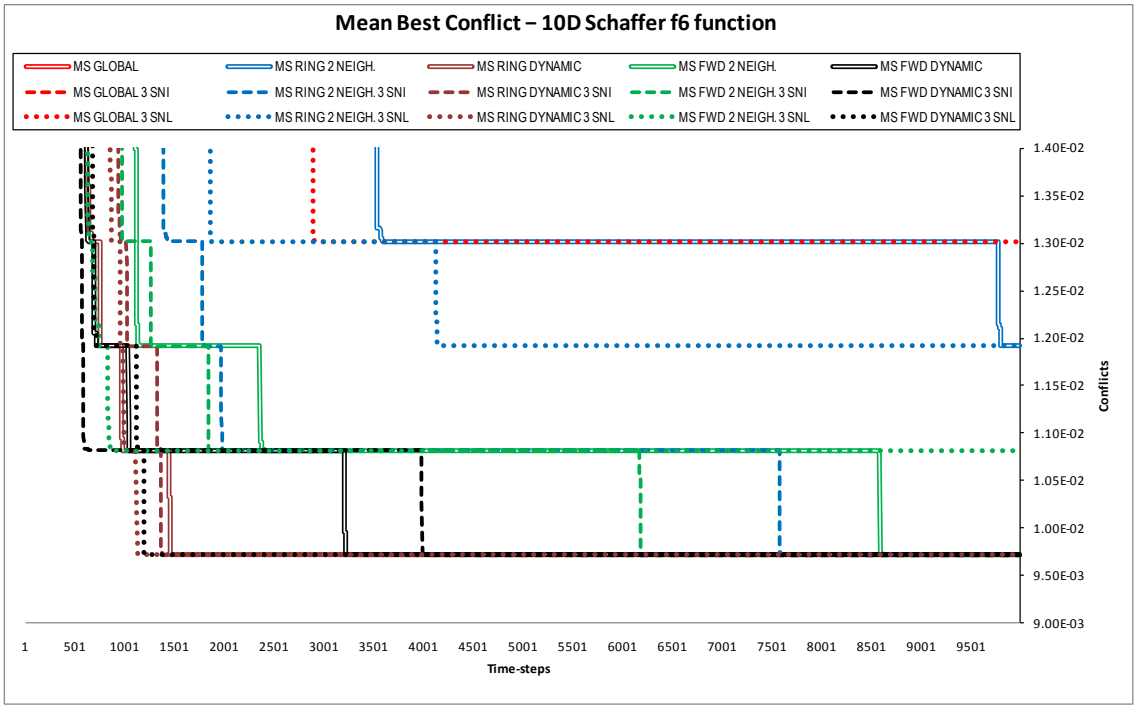


Fig. 7.99. Convergence curves of the mean best conflict for the 10D Schaffer f6 function, associated to Table 7.59. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

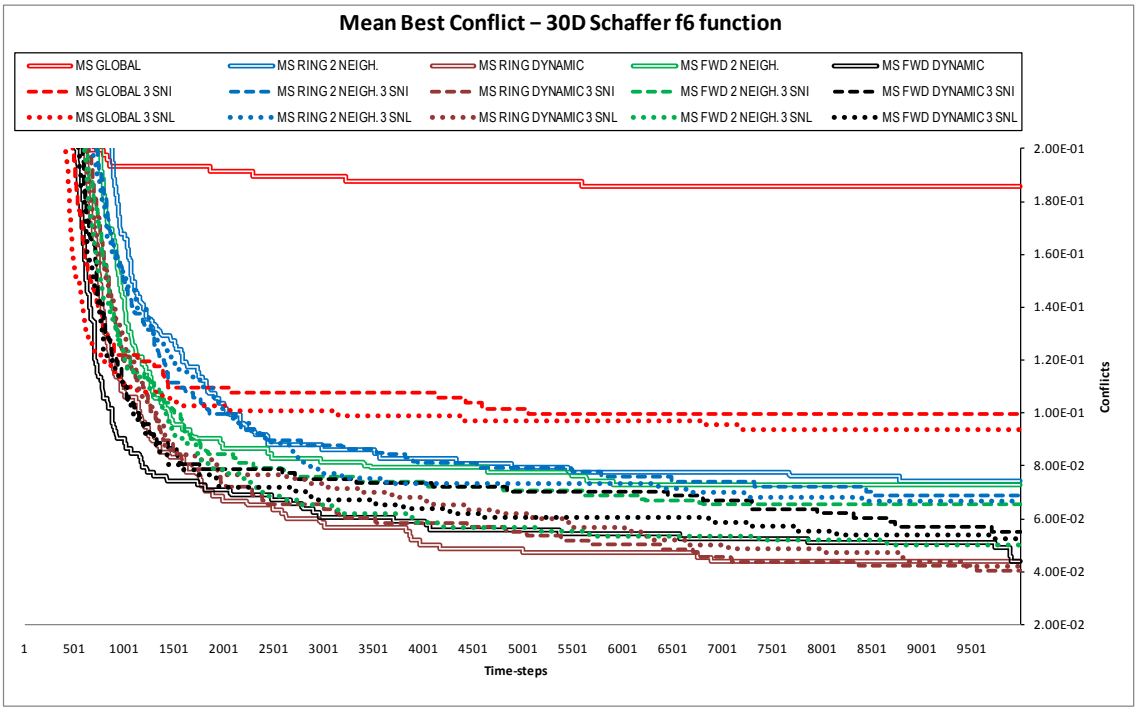


Fig. 7.100. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.60. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

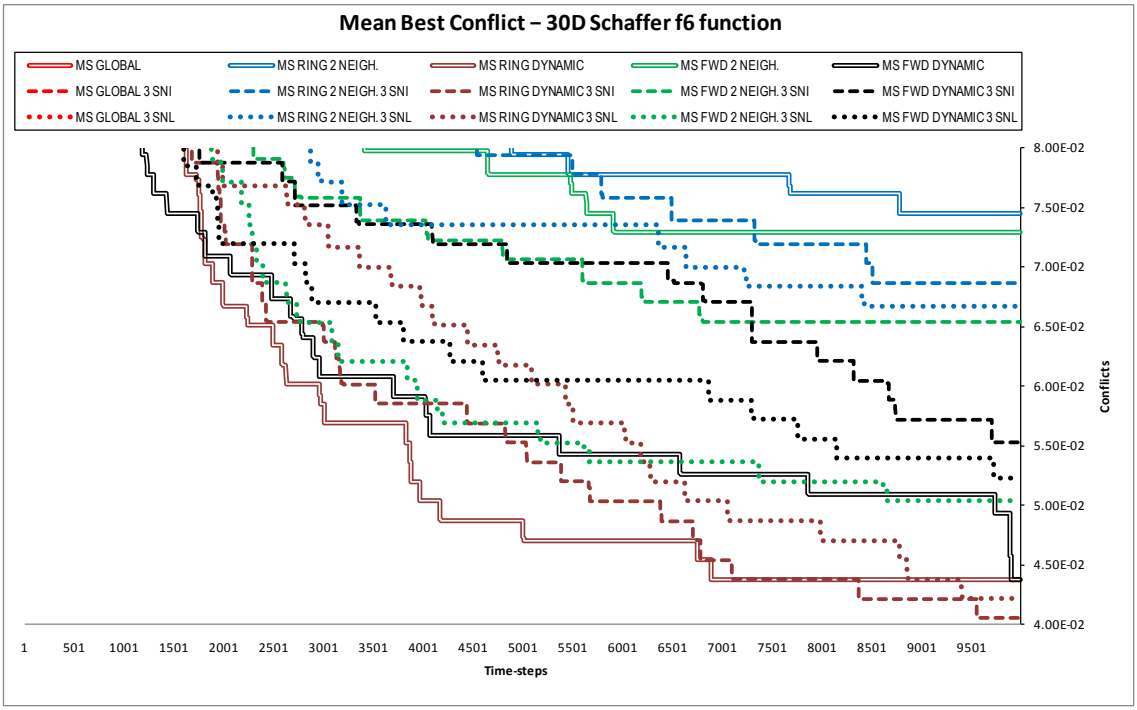


Fig. 7.101. Convergence curves of the mean best conflict for the 30D Schaffer f6 function, associated to Table 7.60. The colour-codes used to identify the neighbourhood structures are the same in the table and figure associated.

Discussion

The settings of the experiments are the same as those described in the previous chapter (see section 6.3.2.2.) unless specifically stated otherwise.

Sphere

In the 2D problem, all the algorithms find the exact solution in every run, with or without sub-neighbourhoods. Hence their effect on the performance cannot be appreciated (refer to Table 7.46 and Fig. 7.75).

In the 10D problem, all the algorithms achieve a success rate (SR) of 100%. Only the 'MS Fwd (*nn*=2)' does not obtain the exact solution in every run. The use of sub-neighbourhoods resolves this issue (Table 7.47, Fig. 7.76).

In the 30D problem, all the algorithms achieve a success rate (SR) of 100%. No noticeable effect of the sub-neighbourhoods can be appreciated here (Table 7.48, Fig. 7.78).

Rosenbrock

In the 2D problem, all the algorithms find the exact solution in every run, already meeting the success criterion by the 1,000th time-step (see Table 7.49 and Fig. 7.79).

In the 10D problem, the implementation of the SNI and SNL techniques are highly beneficial. For the SNI, all success rates (SRs) increase, and all median and mean values are improved for the five topologies tested. For the SNL, the performance of both forward and the global topologies improve, while the performance of the 'Ring (*nn*=2)' topology remains more or less the same. The SNL technique appears detrimental for the 'Ring Dynamic' topology. The best performances are exhibited by the 'Ring Dynamic 3 SNI', the 'Ring Dynamic', and the 'Ring (*nn*=2) 3 SNI'. The 'Global 3 SNI' also exhibits good performance. The experimental results and curves of convergence can be found in Table 7.50, Fig. 7.80, and Fig. 7.81.

In the 30D problem, the sub-neighbourhoods are also beneficial. For the SNI, the 'Fwd Dynamic' topology maintains more or less the same performance, whereas the other four topologies improve theirs. The SNL appears to be slightly detrimental for the 'Ring (*nn*=2)' and for the 'Fwd Dynamic' topologies, while beneficial for the other three. The best performances are exhibited by the three 'Fwd Dynamic', and by the three 'Ring Dynamic' topologies. Refer to Table 7.51; and to Fig. 7.82 to Fig. 7.84.

Rastrigin

In the 2D problem, all algorithms find the exact solution by the end of the search. And only the 'Ring (*nn*=2)' does not meet the success criterion by the 1,000th time-step (see Table 7.52). Notice that both the SNI and the SNL resolve this minor issue.

In the 10D problem, the SNI is also beneficial, improving the performances of the 'MS Global', the 'MS Ring (*nn*=2)', and the 'MS Fwd (*nn*=2)'. The performance of the 'MS Fwd Dynamic' is virtually the same with and without SNI, while the 'MS Ring Dynamic' shows some minor deterioration in its performance. The SNL improves the performances of the 'MS Global' and of the 'MS Ring (*nn*=2)', while it deteriorates those of the 'Fwd Dynamic' and of the 'Ring Dynamic'. The performance of the 'MS Fwd (*nn*=2)' remains virtually unchanged. The best results are obtained by the 'MS Fwd Dy-

namic', followed by the 'MS Fwd Dynamic 3 SNI' and by the 'MS Fwd Dynamic 3 SNL'. Refer to Table 7.53 and to Fig. 7.87 for the experimental results.

In the 30D problem, both the SNI and SNL lead to improvement in every topology except for the 'Fwd Dynamic'. There is a minor deterioration in the performance of the 'MS Fwd Dynamic' if the SNI or SNL are implemented. Nonetheless, the 'MS Fwd Dynamic', the 'MS Fwd Dynamic 3 SNL' and the 'MS Fwd Dynamic 3 SNI' are still the three best performers by a large margin (see Table 7.54; and Fig. 7.88 to Fig. 7.90).

Griewank

In the 2D problem, all algorithms find the exact solution in every run, except for the 'MS Global 3 SNI' and the 'MS Global 3 SNL', which fail in 1 out of 25 runs (refer to Table 7.55 and to Fig. 7.91).

In the 10D problem, both the SNI and SNL lead to improvement in the performance of the 'MS Global' and of the 'MS Fwd (*nn*=2)'. The SNL also improves considerably the performance of the 'MS Fwd Dynamic' and of the 'MS Ring Dynamic', whereas the SNI deteriorates them. Thus, the SNL remarkably improve performance of the dynamic neighbourhoods for this problem. For visualization of the results, refer to Table 7.56, Fig. 7.92, and to Fig. 7.93.

In the 30D problem, the SNI and SNL techniques appear to be detrimental, as every topology decreases its performance except for the global ones (refer to Table 7.57, and Fig. 7.94 and Fig. 7.95).

Schaffer f6

In the 2D problem, all algorithms find the exact solution in every run, except for the 'MS Global' and the 'MS Global 3 SNI' (see Table 7.58 and Fig. 7.97). The implementation of the sub-neighbourhood strategies helps resolve this premature convergence.

In the 10D problem, the implementation of the SNI improves the performance of the 'Global' and of the 'Ring (*nn*=2)' topologies (see Table 7.59). The performances of the remaining topologies stay the same. In turn, the SNL improves the performances of the 'Global' topology, marginally deteriorates the performance of the 'Fwd (*nn*=2)' topology, while the performances of the other topologies remain the same.

In the 30D problem, the implementation of either the SNI or the SNL is beneficial for all topologies except for the 'Fwd Dynamic' whose performance is slightly deteriorated.

Overall analyses

Although a global topology is hardly ever chosen, it might still be convenient when fast convergence is desired. The SNI and SNL techniques improve the performance of the global topology in every problem and dimensionality tested. As to the other topologies, it appears that the convenience or not of applying the techniques is problem-dependent. It seems that they are highly beneficial in unimodal and moderately multimodal problems (e.g. Sphere and Rosenbrock functions), while detrimental in noisy problems such as the 30-dimensional Griewank[1]. While they seem to be, in general, beneficial for the forward topology with two neighbours, the techniques seem to be detrimental for the robust forward dynamic topology in highly multimodal problems (e.g. Rastrigin, Schaffer f6, low-dimensional Griewank). A first glance at the results obtained seems to suggest that the most robust performers on this small test suite are the 'Fwd Dynamic' and the 'Fwd Dynamic 3 SNI' topologies.

It is important to note that the optimizer allows breaking and reconnecting the links between different sub-neighbourhoods very easily. Therefore this sets the ground for first approaches on multi-solution PSO, as well as the use of multi-solutions to improve reluctance to sub-optimal attractors.

7.6. Closure

A dynamic neighbourhood was proposed, consisting of linearly increasing the number of neighbours of a local ring topology as the search progresses, until the topology becomes global. The combined effect of different neighbourhood topologies and coefficients' settings was observed through numerical testing on a benchmark suite of side-constrained problems, showing that the use of the proposed dynamic topology together with different, complementary coefficients' settings within the swarm result in a more robust, general-purpose optimizer.

[1] Note that the Griewank function is highly multimodal in low dimensions, while it becomes more like a Sphere function with noise as dimensionality is increased.

A so-called 'forward topology' was proposed and tested on the same benchmark suite of problems. The combination of this topology with a linearly time-increasing number of neighbours, and a multi-swarm approach using complementary coefficients' settings appear to comprise a robust optimizer. Experimental results are promising.

An additional neighbourhood-related heuristics, namely the 'nearest neighbour' technique, was proposed and tested, aiming to consider actual distances in the physical space in the formation of neighbourhoods while still keeping the full overlapping of the social network. That is to say, guaranteeing that any particle has direct or indirect access to the information acquired by any other particle in the swarm at any stage of the search. The experimental results are inconclusive with regards to its use in mathematical optimization. The technique may be especially useful for applications like swarm robotics, where the actual distance travelled by the particles matter.

Finally, the possibility of splitting the neighbourhood in sub-neighbourhoods is investigated, together with two different ways of exchanging information between sub-neighbourhoods. The original aim was to keep the different coefficients of a multi-swarm approach associated with relatively independent sub-swarms of particles. However the approach can also be applied to sub-neighbourhoods where particles share the same coefficients' settings. Besides, it can be easily adapted for multi-solution search.

It is well-known and widely accepted that the optimal neighbourhood topology is problem-dependent. However, the extensive testing on the forward dynamic topology –with and without sub-neighbourhoods– shows promising results for a general-purpose optimizer. Recall that 'general-purpose' here means that it would perform reasonably well on a wide range of problems presenting different difficulties, and not that it would outperform all other optimizers in all problems (see comments on the 'No Free Lunch Theorem' in Chapter 2).

Chapter 8

CONSTRAINT-HANDLING

The constrained optimization problem is conveniently posed to be handled by a Particle Swarm Optimizer, and two constraint-handling techniques are described: the 'Preserving Feasibility with Priority Rules', and the 'Penalization Method'. A pseudo adaptive scheme to deal with constraints is proposed, and later coupled with the two constraint-handling techniques described. The two resulting pseudo adaptive constraint-handling methods are tested on a traditional benchmark suite composed of 13 constrained optimization problems. Finally, some well-known engineering problems are solved by the proposed approach.

8.1. Introduction

The PSO method is inherently suitable for unconstrained problems. Therefore some external mechanism needs to be incorporated to deal with constraints. Thus, the constrained optimization problem is conveniently formulated so that it can be tackled by a Particle Swarm Optimizer. Two constraint-handling techniques, namely the 'Preserving Feasibility with Priority Rules (PFPR)' and the 'Penalization Method (PM)', are briefly introduced. The proposed 'pseudo adaptive scheme' is described in details, and later coupled with the PFPR. The resulting 'Pseudo Adaptive PFPR' is tested on a classical benchmark suite of 13 constrained optimization problems, whose formulations can be found in appendix II. The same pseudo adaptive scheme is viewed as a means of introducing adaptiveness to the penalization method (PM) without making the penalization coefficients adaptive. Hence a simple static PM is coupled with the pseudo adaptive scheme, and the resulting 'Pseudo Adaptive PM' is tested on the same suite of 13 constrained problems as the PFPR. Finally, a number of classical, well-known engineering problems are tackled using the proposed optimizer.

8.2. Constrained optimization

Since the PSO method cannot handle equality constraints directly, each of them must be turned into a pair of inequality constraints, while some tolerance for their violations

must be set. In order to be tackled by a PSO algorithm, the optimization problem is conveniently formulated as shown in Eq. (8.1):

$$\begin{aligned} &\text{Minimize } f(\mathbf{x}) \\ &\text{subject to } \begin{cases} g_j(\mathbf{x}) \le Tol_{ineq} & ; \quad j = 1, \ldots, q \\ \text{abs}\left(g_j(\mathbf{x})\right) \le Tol_{eq} & ; \quad j = q+1, \ldots, q+r \\ \max\left(0, x_i - u_i\right) + \max\left(0, -x_i + l_i\right) \le 0 & ; \quad i = 1, \ldots, n \end{cases} \end{aligned} \tag{8.1}$$

where:

$g_j(\mathbf{x})$: j^{th} constraint function.

Tol_{ineq} : Tolerance for inequality constraint violations. Typically, $Tol_{ineq} = 0$.

Tol_{eq} : Tolerance for equality constraint violations.

q, r, : Number of inequality and of equality constraints, respectively.

u_i, l_i : Upper and lower bound of the variables in i^{th} Dimension.

Hence there are three types of constraints: *inequality*, *equality*, and *interval constraints* (also *side* or *boundary constraints*), where the latter are a special case of inequality constraints. The amount of constraint violations (*cv*) is calculated as shown in Eq. (8.2):

$$cv = \sum_{j=1}^{q} \max\left(0, g_j(\mathbf{x})\right) + \sum_{j=q+1}^{q+r} \text{abs}\left(g_j(\mathbf{x})\right) + \sum_{i=1}^{n} \left[\max(0, x_i - u_i) + \max(0, -x_i + l_i)\right] \tag{8.2}$$

Thus, the constrained single-objective problem is in fact a multi-objective (MO) unconstrained problem. At the very least, there are two objectives: minimization of the objective function, and minimization of the constraint violations (*cv*). However, *cv* is already an aggregation of objectives, since the minimization of the violation of each constraint is one objective in itself.

8.3. Constraint-Handling Techniques

There are numerous constraint-handling techniques (CHTs) in the literature, according to the way the objective function and the constraints are handled (refer to chapter 4, sec-

tion 4.4.8). Only the so-called 'Preserving Feasibility with Priority Rules (PFPR)' technique and the 'Penalization Method (PM)' are briefly discussed hereafter.

Similar CHTs and some others (e.g. involving levels of comparison, or MO-based techniques) can be found in the literature. Refer, for instance, to (Coello Coello, 1999), (Coello Coello, 2000), (Hu & Eberhart, 2002), (Farmani & Wright, 2003), (Xie, Zhang, & Bi, 2004), (Takahama & Sakai, 2005), (de Freitas Vaz & da Graça Pinto Fernandes, 2006), (Takahama, Sakai, & Iwane, 2006), (Fuentes Cabrera & Coello Coello, 2007), (Helwig & Wanka, 2007), (Innocente & Sienz, 2008), and (Venter & Haftka, 2008).

8.3.1. Preserving Feasibility with Priority Rules

Hu and Eberhart (2002), (2003) proposed the so-called 'preserving feasibility' (PF) technique, which consisted of simply ignoring infeasible particles. Thus the PSO algorithm remains the same, with the addition of feasibility as a condition for a potential solution to be stored in the particles' memories. The other difference is that such a CHT requires the (random) initialization of a feasible swarm to start the search, which happens to spend considerable resources for highly constrained problems.

Hence the Preserving Feasibility with Priority Rules (PFPR) is proposed here, which consists of the original PF technique proposed by Hu & Eberhart (2002) with the incorporation of priority rules (PR) in the comparisons. Thus, priority is given to:

- the lower conflict value between feasible solutions;
- the lower constraint violation (*cv*) between infeasible solutions;
- the feasibility between a feasible solution and an infeasible one;
- the lower conflict value if the constraint violations (*cv*) are the same when two infeasible solutions are compared.

The concept is similar to solving a bi-objective unconstrained problem, where the objectives are the minimization of the objective function and of the constraint violations, and full priority is given to the compliance of the second objective (hence the name PFPR).

This method allows avoiding the evaluation of the objective function for most infeasible particles. The exception is when two infeasible particles with the same *cv* are compared.

This CHT is popular in the literature under different names, with few or no modifications. For instance, refer to (Toscano Pulido & Coello Coello, 2004); (Takahama & Sakai, 2005); (Takahama & Sakai, 2006b); and (He & Wang, 2007); to name a few.

8.3.2. Penalization Method

One of the most popular CHTs in Evolutionary Computation (EC) is the penalization method (PM) introduced in chapter 4, whose main drawback is the need to find suitable penalization coefficients. Performance is very sensitive to these coefficients, whose optimum settings are problem-dependent.

Penalization methods can be viewed as optimizing two objectives: minimizing the objective function $f(\mathbf{x})$ and minimizing the constraint violations $cv(\mathbf{x})$, where the second objective is already an aggregation of objectives. These methods combine all the objectives into a single scalar function to be optimized, thus turning the constrained problem into an unconstrained one where the relative priority awarded to the different objectives is somewhat weighed. The main concept is that this new function must coincide with the original one when every constraint is satisfied. Hence the objective function is penalized for infeasible solutions only. Different kinds of penalization methods can be found in the literature according to the way the penalization is calculated.

The advantage is that they use both objective and constraint functions information within the infeasible space to smoothly guide the search towards more promising areas. The drawback is that they are sensitive to the tuning of at least a couple of problem-dependent penalization coefficients. High penalizations might lead to infeasible regions not being explored converging to non-optimal but feasible solutions, whereas low penalizations might lead to the system evolving solutions that are violating constraints but present themselves as better than feasible solutions. However, research on adaptive coefficients is extensive in the literature (e.g. (Parsopoulos & Vrahatis, 2002), (Coello Coello, 2000)). A classical penalization scheme is shown in Equations (8.3) and (8.4):

$$f_p(\mathbf{x}) = f(\mathbf{x}) + \sum_{j=1}^{m} \left[k_j \cdot \left(f_j(\mathbf{x}) \right)^{\alpha_j} \right] \tag{8.3}$$

$$f_j(\mathbf{x}) = \begin{cases} \max\{0, g_j(\mathbf{x})\} & ; \quad 1 \le j \le q \\ \text{abs}(g_j(\mathbf{x})) & ; \quad q < j \le m \end{cases} \tag{8.4}$$

where $f(\mathbf{x})$ is the conflict function; $f_p(\mathbf{x})$ is the penalized conflict function; $f_j(\mathbf{x})$ is the amount of violation of jth constraint; and k_j and α_j are penalization coefficients.

These coefficients may be constant, dynamic or adaptive, and they can be the same or different for different constraints. Typically, k_j is set to high and α_j to small values. It is not recommendable to use different penalization coefficients for different constraints because that makes the coefficients' tuning more difficult for every problem. An alternative to account for the different sensitivity of the penalized objective function to the different constraint violations which may be of different orders of magnitude consists of normalizing the constraint violations. Even further, the original conflict function may also be normalized so that the latter and the overall measure of constraint violations are of the same order of magnitude. These normalizations are not dealt with in this thesis.

8.4. Pseudo Adaptive Scheme

The optimization problem to be dealt with by a PSO algorithm is formulated as shown in Eq. (8.1). Although the tolerance for inequality constraint violations is typically set to zero, it is set to the variable Tol_{ineq} here, both for generality and to allow its relaxation.

The pseudo adaptive scheme proposed comprises two main parts: 1) a self-tuned initial relaxation of the tolerances for inequality and equality constraint violations; and 2) some means to progressively decrease the tolerance as the search progresses to meet a final desired value.

8.4.1. Self-tuned initial relaxation

The use of tolerances for equality constraint violations is of common practice in population-based methods. It is also not uncommon to relax such tolerances to an arbitrary initial value, where the decrease of such relaxations is typically deterministic. The aim of these relaxations is to temporarily increase the feasible region of the search-space.

However, the impact of a given relaxation on the feasibility ratio (FR) of the search-space is problem-dependent, and can vary greatly. For instance, to obtain a feasibility ratio in the range $\mathrm{FR} \in [20\%, 25\%]$, a tolerance for equality constraint violations of around '0.26' is required for problem g11 (see formulation in digital appendix II). Instead, a tolerance of around '6.63' is required for problem g13, both problems involving equality constraints only.

In addition, since there are problems involving only inequality constraints that present very small FRs, the same concept can be applied. That is, the tolerance for inequality constraint violations can also be dynamically relaxed. For instance, the tolerance required for problem g10 to present a $\mathrm{FR} \in [20\%, 25\%]$ is around '10.83' whereas it is around '2790' for problem g06 (both involving inequality constraints only).

These examples clearly illustrate how problem-dependent the effect of a given tolerance may be. Hence an initial 'self-tuned tolerance relaxation' is proposed, aiming for a user-defined 'target FR'. Thus, the self-tuning procedure consists of starting with a small, minimum value for the tolerance, and evaluating the constraint functions of 1000 randomly selected solutions[1]. The FR is evaluated, and the tolerance is adequately increased or decreased. For problems involving inequality and equality constraints, the tolerance for the violations of equality constraints are arbitrarily kept 10 times greater than that of inequality constraint violations.

The aim is to find a tolerance such that $(\text{target FR}) \leq \mathrm{FR} \leq (\text{target FR} + \delta\mathrm{FR})$, where the 'target FR' and the 'δFR' are defined by the user. The procedure is as follows:

The tolerances are arbitrarily set as $tol(1,1) = 0.01$ for inequality and $tol(1,2) = 0.1$ for equality constraints, and the approximate FR is calculated by randomly generating 1000 solutions.

Set $k_{\min} = 1.10$ and $k_{\max} = 10$.

If $\mathrm{FR} < \text{target FR}$, the tolerances are increased k times within a while loop as shown in Eqs. (8.5) and (8.6), until $\mathrm{FR} \geq \text{target FR}$. Therefore $k \in [1.1, 10]$, constant within the loop. Once the latter is terminated, it may be that $\mathrm{FR} > \text{target FR} + \delta\mathrm{FR}$.

[1] Of course this number is arbitrary, and can be adapted as the user sees convenient for available resources.

$$k = \max\left(k_{\min}, \min\left(k_{\max}, \left[\text{target FR} - \text{FR}\right]\right)\right) \tag{8.5}$$

$$tol(1, j) \leftarrow k \cdot tol(1, j) \tag{8.6}$$

If $\text{FR} > \text{target FR} + \delta\text{FR}$, the tolerances must be decreased. This is performed by storing the previous tolerance $\overline{tol}$ (when $\text{FR} < \text{target FR}$) and calculating the difference δtol as shown in Eq. (8.7). The values of $k_{\min}$ and $k_{\max}$ are updated as in Eq. (8.8), where *counter* is the number of times that $\text{FR} > \left(\text{target FR} + \delta\text{FR}\right)$.

$$\delta tol = tol - \overline{tol} \tag{8.7}$$

$$k_{\min} = k_{\max} = 1 + \frac{1}{10 \cdot counter} \tag{8.8}$$

The new tolerance to be considered is then calculated within a while loop as shown in Eq. (8.9) until $\text{FR} \leq \left(\text{target FR} + \delta\text{FR}\right)$. Note that $\overline{tol}$ is outside this while loop.

$$\begin{cases} \delta tol \leftarrow 0.90 \cdot \delta tol \\ tol = \overline{tol} + \delta tol \end{cases} \tag{8.9}$$

After the while loop in Eq. (8.9) is terminated, it may happen that $\text{FR} \leq \text{target FR}$, in which case the tolerance must be increased again as in Eq. (8.6), but now *k* has been reduced as in Eq. (8.8).

Note that the cost of this self-tuning procedure counts in constraint function evaluations (CEs), involving no objective function evaluation (FE).

Although it is unusual to relax inequality constraints, there are highly constrained problems with inequality constraints only, which present a very low FR. In addition, even when the FR is not low, relaxing the tolerances may help approach the optimum from every direction when it is located on active constraints or nearby.

When the FR of the problem is unknown, the program calculates it in the same fashion as it calculates an approximate one. If the FR is already higher than the target FR chosen by the user, the program re-computes the target FR as 1.1 times the FR of the problem, of course upper-bounded by 100%. An example of the self-tuned initial tolerances

obtained by this procedure for the 13 constrained problems in the benchmark suite in appendix II is shown in Table 8.1, where $\text{target FR} = 20\%$, and the $\delta\text{FR} = 5\%$. Once the procedure has finished, the FR is calculated for the obtained tolerance by randomly initializing 10^6 solutions (for higher precision). Those values are provided in the Table 8.1 so as to confirm that using 'only' 10^3 random solutions to calculate the FR during the self-tuning procedure is accurate enough. Note that all problems with a FR lower than the target (20%) present a FR with initial tolerance $(20\%) \leq \text{FR} \leq (20\% + 5\%)$. In turn, if a problem has a FR greater than the target, it is increased by a factor of 1.1 by means of the initial tolerance (see problem g04 in Table 8.1).

Table 8.1. Features of the problems in the test suite: number of dimensions, inequality and equality constraints; feasibility ratios (FRs) of the problem with no tolerance, desired tolerance, and initial tolerance; and the mean self-tuned initial inequality and equality tolerances. FRs are calculated by randomly generating 10^6 solutions, where final (desired) equality constraint violations tolerance equals 10^{-4}.

Problem	Optimum	Dim.	IC	EC	FR [%]	FR [%] for desired tolerance	FR [%] for Initial relaxed tolerance	Mean initial inequality tolerance	Mean initial equality tolerance
g01	-15.000000	13	9	0	0.0003	0.0003	23.4617	89.92	N/A
g02	-0.803619	20	2	0	99.9971	99.9971	99.9971	0.01	N/A
g03	-1.000500	10	0	1	< 0.0001	0.0002	24.5335	N/A	1.66
g04	-30665.538672	5	3 (#)	0	26.9887	26.9887	30.2026	0.11	N/A
g05	5126.496714	4	1 (*)	3	< 0.0001	< 0.0001	23.3053	68.88	688.79
g06	-6961.813876	2	2	0	0.0074	0.0074	24.3050	2790.51	N/A
g07	24.306209	10	8	0	0.0001	0.0001	23.8399	383.89	N/A
g08	-0.095825	2	2	0	0.8610	0.8610	23.4371	9.88	N/A
g09	680.630057	7	4	0	0.5232	0.5232	24.0533	421.13	N/A
g10	7049.248021	8	6	0	0.0005	0.0005	21.1715	10.83	N/A
g11	0.749900	2	0	1	<0.0001	0.0108	24.8914	N/A	0.26
g12	-1.000000	3	1 (@)	0	4.7713	4.7713	22.0256	0.11	N/A
g13	0.053942	5	0	3	<0.0001	<0.0001	22.8845	N/A	6.63

(#) Other authors claim there are 6 inequality constraints, but each one defines an interval, therefore no more than 3 constraints can be violated simultaneously.

(*) Other authors claim there are two inequality constraints, but they define an interval, so that no more than 1 constraint can be violated simultaneously.

(@) Most authors claim there are 9^3 inequality constraints, but in reality it is one constraint that splits the feasible space in 9^3 (729) disjointed spheres. The solution needs to be inside one sphere to be feasible, so that membership to all 729 spheres is not possible. If the constraint is viewed as 729 constraints, then at least 728 of them will be always violated.

8.4.2. Pseudo Adaptive Tolerance Update

The aim is to make the tolerance update adaptive so that updates are performed when they would have a less disruptive or harmful effect in the dynamics of the swarm and in maintaining potentially good solutions.

Thus, updates are performed when a given percentage (*per*) of the particles' best experiences (**pbest**) are located within feasible space (i.e. within the current tolerances). The coefficient for the exponential update ($ktol^{(t)}$) is also pseudo adaptive, as shown in Eq. (8.10), while the exponential update of the tolerances is as posed in Eq. (8.11).

$$ktol^{(t)} = \frac{0.99 - ktol_{\min}}{100 - per_{\min}} \cdot (100 - per) + ktol_{\min} \tag{8.10}$$

$$tol^{(t)} = ktol^{(t)} \cdot tol^{(t-1)} \tag{8.11}$$

Thus, $ktol^{(t)} = 0.99$ for $per = per_{\min}$, $ktol^{(t)} = ktol_{\min}$ for $per = 100$, and the variation in between is linear. Therefore, the greater the percentage above a minimum established, the greater the size of the tolerance decrease.

Given that the tolerance for inequality constraint violations is typically set to '0' and here it is kept ten times smaller than the tolerance for equality constraints when both are present and relaxed, whenever the tolerance for inequality constraints goes below 10^{-5}, it is automatically reset to '0'.

Aiming to avoid too many time-steps without a tolerance update, a 'safety mechanism' is implemented by enforcing a tolerance update if:

$$\frac{t}{n^o\ tol.\ updates} \geq tol.\ timespan \tag{8.12}$$

where *t* is the number of time-step, n^o *tol. updates* is the number of tolerance updates, and *tol. timespan* is the average number of time-steps between tolerance updates that is not allowed to be surpassed. When the update is enforced due to Eq. (8.12), the coefficient used in Eq. (8.11) equals $ktol^{(t)} = 0.99$.

In order to give some time for the particles to find feasible solutions once the tolerances have reached their desired value, it is arbitrarily set that such values must be reached by the time 80% of the search has been carried out. That is, the desired tolerances must be reached by $t_{\min} = 0.8 \cdot t_{\max}$. If the desired tolerance was not reached by $t = 0.9 \cdot t_{\min}$, a coefficient is calculated so that final tolerances are reached by $t = t_{\min}$ performing an update at every remaining time-step. Thus, the coefficient *ktol* is calculated as shown in

Eqs. (8.13) and (8.14), and is fixed for the whole remaining $0.1 \cdot t_{\min}$ time-steps (i.e. from $t = 0.9 \cdot t_{\min} + 1$ to $t = t_{\min}$).

$$ktol^{0.1 \cdot t_{\min}} \cdot tol^{(0.9 \cdot t_{\min})} = tol^{(t_{\min})} \tag{8.13}$$

$$ktol = \left(\frac{tol^{(t_{\min})}}{tol^{(0.9 \cdot t_{\min})}} \right)^{\frac{1}{0.1 \cdot t_{\min}}} \tag{8.14}$$

where *ktol* is calculated independently for inequality and equality constraints. For inequality constraints –where typically $tol^{(t_{\min})} = 0$–, the latter is replaced by 10^{-5} for the calculation of *ktol* in Eq. (8.14), and the tolerance is set to '0' as soon as it reaches a value equal to or below 10^{-5}.

A different adaptive relaxing rule of the tolerance for equality constraint violations is proposed in (Xie, Zhang, & Bi, 2004).

8.5. Pseudo Adaptive PFPR

When using the PFPR technique in highly constrained problems, most of the search is driven by constraint satisfaction, disregarding the conflict function information. Thus, by the time a particle finds a feasible location, it might be anywhere with respect to the optimal solution. Hence the pseudo adaptive scheme previously proposed is coupled with the PFPR technique to help in handling constraints, especially in problems with low FRs. By working with the pseudo adaptive scheme, the PFPR technique 'is fooled' into using objective function information while searching the infeasible space.

8.5.1. Experimental results

A Multi-Swarm (MS) PSO algorithm is used, which is composed of 50 particles split in three sub-swarms. The first sub-swarm is composed of 17 particles under the PSO-RRR2-1 settings; the second sub-swarm is composed of 16 particles under the PSO-RRR1-1 settings; and the third is composed of 17 particles under the C-PSO-1 settings. Three sub-neighbourhoods are used, coinciding with the sub-swarms, with the individ-

ual overlapping (SNI) discussed in the previous chapter. Every run is performed for a length of 10000 time-steps.

The neighbourhood is the 'dynamic forward topology' with linearly increasing number of neighbours, from 2 at the first time-step until it becomes global at the end of the search. The nearest neighbour heuristics is activated and acting on the individual best experiences (NNB) rather than on the particles' positions (refer to chapter 7).

The particles' positions are initialized by generating 1000 independent Latin Hypercube Samplings (LHSs), and selecting the one with the maximum minimum distance between particles. Each sub-swarm is initialized independent from the others. Velocities are initialized to zero. Every best experience is initialized at exactly the same distance from its corresponding particle. Each component of this distance is calculated as the corresponding feasible interval divided by twice the number of particles in the swarm. The sign of the component, and hence the direction of the distance vector, are randomly generated. For each pair '**p**–**pbest**', a comparison is performed so that the best one becomes (or stays) **pbest** and the other becomes (or stays) **p** before the search begins. Thus, every particle starts the search with the same, moderate acceleration towards its **pbest** (the acceleration towards its **lbest** depends on the neighbourhood structure).

The position-based measures of error are relative, whereas the conflict-based measures are absolute; *tref* = 10; the measures of error are computed on the individual best experiences (rather than on the current ones).

The CHT is the PFPR with Pseudo Adaptive tolerance relaxation. Target $\mathrm{FR} = 20\%$ $(\delta\mathrm{FR} = 5\%)$. For the tolerance decrease, $ktol_{\min} = 0.90$ and $per_{\min} = 80\%$ in Eq. (8.10), while $t_{\min} = 0.80 \cdot t_{\max} = 8000$ in Eqs. (8.13) and (8.14). $Tol_{ineq}^{final} = 0$ and $Tol_{eq}^{final} = 10^{-4}$.

A run is considered successful if the error is no greater than 0.0001, and the statistics are calculated out of 25 runs. The random number generator is reset to its initial state only before the first run of every experiment. The results are provided in Table 8.2 and some repeated in Table 8.3 for comparison.

For reference, the experiments are also run without relaxing the constraints, and also with self-tuned initial relaxation but with a classic deterministic, exponential decrease. In the latter case, $ktol = 0.98$. These results are also presented in Table 8.2.

The full output files are provided in *.xlsx and *.mat formats in the digital appendix, from where a great amount of information with regards to the search can be obtained. It is fair to note that the settings of the experiments did not respond to any kind of tuning.

Table 8.2. Statistical results obtained for the 13 problems in the test suite for the PFPR technique and 3 types of tolerance relaxation: none, initially self-tuned with exponential decrease, and initially self-tuned with Pseudo Adaptive decrease. The percentages of feasible solutions and successful solutions (error not greater than 10^{-4}); the mean numbers of FEs and CEs; and the mean percentage of feasible **pbest**s at the end of the search are also provided.

Problem	OPTIMUM	Tolerance Relaxation (PFPR)	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Mean CEs	Mean [%] Feasible pbests	Runs
g01	-15.0000	NONE	-15.0000	-15.0000	-15.0000	-15.0000	100.00	100.00	2.24E+05	5.00E+05	100.00	25
		EXP.	-15.0000	-15.0000	-15.0000	-15.0000	100.00	100.00	2.25E+05	5.86E+05	100.00	25
		P.AD.	-15.0000	-15.0000	-14.8981	-12.4531	100.00	96.00	1.16E+05	6.42E+05	100.00	25
g02	-0.8036	NONE	-0.8036	-0.7949	-0.7926	-0.6879	100.00	40.00	2.51E+05	5.00E+05	100.00	25
		EXP	0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	2.52E+05	5.35E+05	100.00	25
		P.AD.	-0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	2.52E+05	5.08E+05	100.00	25
g03	-1.0005	NONE	-0.9996	-0.9875	-0.9713	-0.8227	100.00	0.00	7.76E+04	5.00E+05	98.64	25
		EXP.	-1.0005	-1.0005	-1.0005	-1.0003	100.00	92.00	1.21E+05	5.77E+05	100.00	25
		P.AD.	-1.0005	-1.0005	-1.0005	-1.0005	100.00	100.00	1.25E+05	5.84E+05	99.92	25
g04	-30665.5387	NONE	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	3.24E+05	5.00E+05	100.00	25
		EXP.	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	3.19E+05	5.50E+05	100.00	25
		P.AD.	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	3.22E+05	5.14E+05	100.00	25
g05	5126.4967	NONE	5126.4983	5183.5743	5262.5931	5197.5450	92.00	0.00	8.82E+04	5.00E+05	20.48	25
		EXP	5126.6759	5149.4887	5226.5105	5211.9654	96.00	0.00	1.09E+05	5.87E+05	26.16	25
		P.AD.	5126.5060	5129.6245	5135.0431	5205.0354	100.00	0.00	1.21E+05	6.28E+05	66.80	25
g06	-6961.8139	NONE	-6961.8139	-6961.8139	-6961.8139	-6961.8139	100.00	100.00	2.87E+05	5.00E+05	99.76	25
		EXP.	-6961.8139	-6961.8139	-6961.8139	-6961.8139	100.00	100.00	2.98E+05	6.18E+05	100.00	25
		P.AD.	-6961.8139	-6961.8139	-6961.8139	-6961.8139	100.00	100.00	2.41E+05	6.18E+05	100.00	25
g07	24.3062	NONE	24.3120	24.3494	24.4183	24.9295	100.00	0.00	1.59E+05	5.00E+05	100.00	25
		EXP.	24.3149	24.4884	24.5306	24.9681	100.00	0.00	1.54E+05	6.03E+05	100.00	25
		P.AD.	24.3246	24.4429	24.4863	25.1151	100.00	0.00	1.60E+05	5.57E+05	100.00	25
g08	-0.0958	NONE	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	4.98E+05	5.00E+05	100.00	25
		EXP.	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	4.96E+05	5.72E+05	100.00	25
		P.AD.	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	4.98E+05	5.20E+05	100.00	25
g09	680.6301	NONE	680.6306	680.6341	680.6346	680.6444	100.00	0.00	1.80E+05	5.00E+05	100.00	25
		EXP	680.6306	680.6322	680.6330	680.6427	100.00	0.00	1.82E+05	6.03E+05	100.00	25
		P.AD.	680.6305	680.6334	680.6339	680.6406	100.00	0.00	1.81E+05	5.36E+05	100.00	25
g10	7049.2480	NONE	7057.5044	7107.2362	7146.9957	7292.5136	100.00	0.00	1.69E+05	5.00E+05	100.00	25
		EXP.	7053.8366	7193.8963	7199.2320	7454.0316	100.00	0.00	1.64E+05	5.73E+05	100.00	25
		P.AD.	7049.5448	7145.6473	7145.1506	7437.8998	100.00	0.00	8.61E+04	6.25E+05	98.00	25
g11	0.7499	NONE	0.7499	0.7499	0.7499	0.7499	100.00	100.00	1.09E+05	5.00E+05	98.96	25
		EXP.	0.7499	0.7499	0.7499	0.7499	100.00	100.00	1.36E+05	5.57E+05	97.28	25
		P.AD.	0.7499	0.7499	0.7499	0.7499	100.00	100.00	1.20E+05	5.98E+05	88.88	25
g12	-1.0000	NONE	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	4.93E+05	5.00E+05	100.00	25
		EXP.	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	4.96E+05	5.48E+05	100.00	25
		P.AD.	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	4.95E+05	5.16E+05	100.00	25
g13	0.0539	NONE	0.0915	0.6562	0.5911	0.9910	100.00	0.00	8.79E+04	5.00E+05	40.80	25
		EXP.	0.0545	0.2172	0.2825	0.8696	100.00	0.00	9.99E+04	5.63E+05	40.80	25
		P.AD.	0.0539	0.0540	0.1030	0.4391	100.00	64.00	1.11E+05	6.35E+05	76.88	25

Table 8.3. Statistical results obtained for the 13 problems in the test suite for the PFPR constraint-handling technique coupled with the proposed Pseudo Adaptive Scheme. The percentages of feasible solutions and successful solutions (error not greater than 10^{-4}); the mean numbers of FEs and CEs; and the mean percentage of feasible **pbest**s at the end of the search are also provided. Results from (Toscano Pulido & Coello Coello, 2004) and (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005) (PESO) are provided for reference.

Problem	OPTIMUM	OPTIMIZER	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Mean CEs	Mean [%] Feasible pbests	Runs
g01	-15.0000	P.AD. PFPR	-15.0000	-15.0000	-14.8981	-12.4531	100.00	96.00	1.16E+05	6.42E+05	100.00	25
		Toscano Pulido et al.	-15.0000	-	-15.0000	-15.0000	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-15.0000	-15.0000	-15.0000	-15.0000	-	-	3.40E+05	-	-	30
g02	-0.8036	P.AD. PFPR	-0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	2.52E+05	5.08E+05	100.00	25
		Toscano Pulido et al.	-0.8034	-	-0.7904	-0.7504	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-0.7926	-0.7317	-0.7217	0.6141	-	-	3.40E+05	-	-	30
g03	-1.0005	P.AD. PFPR	-1.0005	-1.0005	-1.0005	-1.0005	100.00	100.00	1.25E+05	5.84E+05	99.92	25
		Toscano Pulido et al.	-1.0047	-	-1.0038	-1.0025	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-1.0050	-1.0050	-1.0050	-1.0050	-	-	3.40E+05	-	-	30
g04	-30665.5387	P.AD. PFPR	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	3.22E+05	5.14E+05	100.00	25
		Toscano Pulido et al.	-30665.5000	-30665.5000	-30665.5000	-30665.5000	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-30665.5387	-30665.5387	-30665.5387	-30665.5387	-	-	3.40E+05	-	-	30
g05	5126.4967	P.AD. PFPR	5126.5060	5129.6245	5135.0431	5205.0354	100.00	0.00	1.21E+05	6.28E+05	66.80	25
		Toscano Pulido et al.	5126.6400	-	5461.0813	6104.7500	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	5126.4842	5126.5383	5129.1783	5148.8594	-	-	3.40E+05	-	-	30
g06	-6961.8139	P.AD. PFPR	-6961.8139	-6961.8139	-6961.8139	-6961.8139	100.00	100.00	2.41E+05	6.18E+05	100.00	25
		Toscano Pulido et al.	-6961.8100	-	-6961.8100	-6961.8100	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-6961.8139	-6961.8139	-6961.8139	-6961.8139	-	-	3.40E+05	-	-	30
g07	24.3062	P.AD. PFPR	24.3246	24.4429	24.4863	25.1151	100.00	0.00	1.60E+05	5.57E+05	100.00	25
		Toscano Pulido et al.	24.3511	-	25.3558	27.3168	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	24.3069	24.3713	24.3713	24.5935	-	-	3.40E+05	-	-	30
g08	-0.0958	P.AD. PFPR	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	4.98E+05	5.20E+05	100.00	25
		Toscano Pulido et al.	-0.0958	-	-0.0958	-0.0958	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-0.0958	-0.0958	0.0958	-0.0958	-	-	3.40E+05	-	-	30
g09	680.6301	P.AD. PFPR	680.6305	680.6334	680.6339	680.6406	100.00	0.00	1.81E+05	5.36E+05	100.00	25
		Toscano Pulido et al.	680.6380	-	680.8524	681.5530	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	680.6301	680.6301	680.6301	680.6301	-	-	3.40E+05	-	-	30
g10	7049.2480	P.AD. PFPR	7049.5448	7145.6473	7145.1506	7437.8998	100.00	0.00	8.61E+04	6.25E+05	98.00	25
		Toscano Pulido et al.	7057.5900	-	7560.0479	8104.3100	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	7049.4595	7069.9262	7099.1014	7251.3962	-	-	3.40E+05	-	-	30
g11	0.7499	P.AD. PFPR	0.7499	0.7499	0.7499	0.7499	100.00	100.00	1.20E+05	5.98E+05	88.88	25
		Toscano Pulido et al.	0.7500	-	0.7501	0.7529	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	0.7490	0.7490	0.7490	0.7490	-	-	3.40E+05	-	-	30
g12	-1.0000	P.AD. PFPR	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	4.95E+05	5.16E+05	100.00	25
		Toscano Pulido et al.	-1.0000	-	-1.0000	-1.0000	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-1.0000	-1.0000	-1.0000	-1.0000	-	-	3.40E+05	-	-	30
g13	0.0539	P.AD. PFPR	0.0539	0.0540	0.1030	0.4391	100.00	64.00	1.11E+05	6.35E+05	76.88	25
		Toscano Pulido et al.	0.0687	-	1.7164	13.6695	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	0.0815	0.6319	0.6269	0.9976	-	-	3.40E+05	-	-	30

An example of the results of the initialization procedure is shown in Fig. 8.1. This example is intended to illustrate both the effect of the nearest neighbour procedure and the initialization of each **pbest** at the same distance from its corresponding **p**. However, in

the settings for the experiments there are three sub-swarms rather than one, and hence their initializations are independent. That is to say, the particles corresponding to each sub-neighbourhood would be allocated by the LHS so as to cover the whole search-space, regardless of the location of the particles of other sub-neighbourhoods.

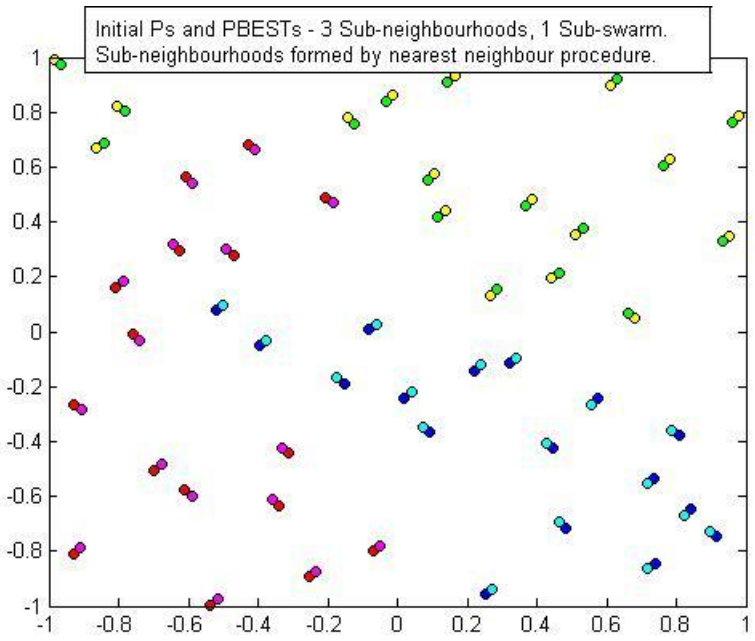


Fig. 8.1. Example of the initialization procedure for three sub-neighbourhoods, one sub-swarm, and nearest neighbourhood procedure for the generation of the sub-neighbourhoods. If there were three sub-swarms as well, each sub-neighbourhood/sub-swarm would attempt to cover the whole search-space independently from the others.

8.5.2. Discussion

From Table 8.2, it can be observed that the adaptive scheme improves the performance of the algorithm in several aspects. In problem g05, it can be observed that it raised the percentage of feasible solutions to 100% while the median and mean solutions show great improvement despite being unable to meet the demanding success condition. In problem g02, it can be observed that the self-tuned initial relaxation is beneficial (irrespective of the form of its posterior decrease), since the percentage of successful solutions increase when this initial relaxation is present. Note that this problem presents a very high feasibility ratio, and yet relaxing the constraints is beneficial. The percentage of successful solutions is also notably increased by means of the proposed pseudo adap-

tive scheme in problems g03 and g13. The drawback is that one failure (in 25 runs) to find the solution in g01 can be now observed.

From Table 8.3, it can be observed that the results obtained are clearly better than those in (Toscano Pulido & Coello Coello, 2004); and competitive with the optimizer PESO in (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005). Note that they both use a higher tolerance for equality constraint violations ($Tol_{eq.} = 10^{-3}$).

8.6. Pseudo Adaptive Penalization

The aim here is not to develop an adaptive penalization method, but simply to explore the possibility of introducing the pseudo adaptive scheme proposed in this thesis into a penalization method, so that it becomes adaptive by penalizing adaptive constraint violations rather than by using adaptive penalization coefficients. Therefore, a simple penalization method is implemented with arbitrarily set –not tuned– constant coefficients.

The penalization scheme proposed is shown in Eqs. (8.15) to (8.17). Notice how the adaptiveness in the formulation lies in the adaptiveness of the tolerances for constraint violations, despite keeping the penalization coefficients constant.

$$f_p(\mathbf{x}) = f(\mathbf{x}) + k \cdot \sum_{j=1}^{m} \left(f_j(\mathbf{x})\right)^{\alpha} \tag{8.15}$$

$$f_j(\mathbf{x}) = \begin{cases} \max\left\{0, \left(g_j(\mathbf{x}) - Tol_{ineq}\right)\right\} & ; \quad 1 \le j \le q \\ \max\left\{0, \left(\text{abs}\left(g_j(\mathbf{x})\right) - Tol_{eq}\right)\right\} & ; \quad q < j \le m \end{cases} \tag{8.16}$$

$$k = 10^6$$
$$\alpha = \begin{cases} 2 & \text{if} \quad f_j(\mathbf{x}) \ge 1 \\ 1 & \text{if} \quad f_j(\mathbf{x}) < 1 \end{cases} \tag{8.17}$$

8.6.1. Experimental results

The same settings used in the experiments in section 8.5.1 are used here, but now the CHT is a penalization method (PM) with the static coefficients shown in Eq. (8.17).

The results are provided in Table 8.4 and some repeated in Table 8.5 for comparison. Some interesting selected curves of the evolution of the pseudo adaptive tolerances and of the mean best solution are offered in Fig. 8.2 to Fig. 8.6. For reference, the experiments are also run without relaxing the constraints, and also with self-tuned initial relaxation but with a classic deterministic, exponential decrease. In the latter case, $ktol = 0.98$. These results are also presented in Table 8.4.

The full output files are provided in *.xlsx and *.mat formats in the digital appendix, from where a great amount of information with regards to the search can be obtained.

8.6.2. Discussion

From Table 8.4, it can be observed that the adaptive scheme improves the performance of the algorithm in several aspects, while here there is no failure for g01 (like in the PFPR). In problem g05, the median and mean solutions also show great improvement. In problem g02, the percentage of successful solutions increases with the proposed pseudo adaptive scheme. The percentage of successful solutions is also notably increased with the proposed technique in problems g03 and g13, while here also in g09. However it seems to be detrimental in g10 as it is. Further studies are needed to investigate the reason for this, and to improve even more its performance.

From Table 8.5, it can be observed that it generally outperforms results in (Toscano Pulido & Coello Coello, 2004), while it is competitive with PESO in (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005), outperforming each other on some problems. Namely, the algorithm tested here outperforms PESO in problems g02 and g13, whereas PESO outperforms it in problems g05 and g10. They are competitive in the rest of the problems in the test suite, with PESO performing marginally better overall.

Since the values of the tolerances are pseudo adaptive, it is interesting to observe the form of the curves and their evolution throughout the search. Due to the stochastic nature of the paradigm, those curves vary from one run to the next for a given problem. The curves showing the evolution of the tolerances corresponding to four selected problems are offered in Fig. 8.2 to Fig. 8.5. Thus, Fig. 8.2 corresponds to problem g01 involving 9 inequality constraints; Fig. 8.3 corresponds to problem g03 involving 1 equality constraint; Fig. 8.4 corresponds to problem g05 involving 3 equality and 1 inequality

constraints (in the form of an interval); and Fig. 8.5 corresponds to problem g13 involving 3 equality constraints. In those figures, the tolerance curves correspond to one single arbitrary run, whereas the average curves correspond to the average among all 25 runs.

Table 8.4. Statistical results obtained for the 13 problems in the test suite for the PM technique and 3 types of tolerance relaxation: none, initially self-tuned with exponential decrease, and initially self-tuned with Pseudo Adaptive decrease. The percentages of feasible solutions and successful solutions (error not greater than 10^{-4}); the mean numbers of FEs and CEs; and the mean percentage of feasible **pbest**s at the end of the search are also provided.

Problem	OPTIMUM	Tolerance Relaxation (PM)	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Mean CEs	Mean [%] Feasible pbests	Runs
g01	-15.0000	NONE	-15.0000	-15.0000	-15.0000	-15.0000	100.00	100.00	5.00E+05	5.00E+05	99.92	25
		EXP.	-15.0000	-15.0000	-15.0000	-15.0000	100.00	100.00	5.00E+05	5.86E+05	99.76	25
		P.AD.	-15.0000	-15.0000	-15.0000	-15.0000	100.00	100.00	5.00E+05	5.75E+05	99.36	25
g02	-0.8036	NONE	-0.8036	-0.7949	-0.7926	-0.6879	100.00	40.00	5.00E+05	5.00E+05	100.00	25
		EXP.	-0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	5.00E+05	5.35E+05	100.00	25
		P.AD.	-0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	5.00E+05	5.08E+05	100.00	25
g03	-1.0005	NONE	-0.9992	-0.9832	-0.9726	-0.8949	100.00	0.00	5.00E+05	5.00E+05	98.80	25
		EXP.	-1.0005	-1.0005	-1.0005	-1.0003	100.00	96.00	5.00E+05	5.77E+05	100.00	25
		P.AD.	-1.0005	-1.0005	-1.0005	-1.0005	100.00	100.00	5.00E+05	5.84E+05	99.92	25
g04	-30665.5387	NONE	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	5.00E+05	5.00E+05	100.00	25
		EXP.	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	5.00E+05	5.50E+05	100.00	25
		ADAPTIVE	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	5.00E+05	5.14E+05	100.00	25
g05	5126.4967	NONE	5126.4984	5158.4660	5242.6720	5708.2809	100.00	0.00	5.00E+05	5.00E+05	25.20	25
		EXP.	5126.5160	5155.0725	5235.5661	5885.9125	100.00	0.00	5.00E+05	5.87E+05	24.80	25
		P.AD.	5126.5938	5130.1227	5142.2653	5318.2998	100.00	0.00	5.00E+05	6.31E+05	57.52	25
g06	-6961.8139	NONE	-6961.8139	6961.8139	-6961.8139	-6961.8139	100.00	100.00	5.00E+05	5.00E+05	100.00	25
		EXP.	-6961.8139	-6961.8139	-6961.8139	-6961.8139	100.00	100.00	5.00E+05	6.18E+05	100.00	25
		P.AD.	-6961.8139	-6961.8139	-6961.8139	-6961.8139	100.00	100.00	5.00E+05	6.38E+05	100.00	25
g07	24.3062	NONE	24.3086	24.3974	24.4474	25.1851	100.00	0.00	5.00E+05	5.00E+05	100.00	25
		EXP.	24.3223	24.4382	24.4473	24.7962	100.00	0.00	5.00E+05	6.03E+05	100.00	25
		P.AD.	24.3222	24.4835	24.5153	24.9482	100.00	0.00	5.00E+05	5.58E+05	100.00	25
g08	-0.0958	NONE	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	5.00E+05	5.00E+05	100.00	25
		EXP.	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	5.00E+05	5.72E+05	100.00	25
		P.AD.	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	5.00E+05	5.20E+05	100.00	25
g09	680.6301	NONE	680.6305	680.6326	680.6330	680.6388	100.00	0.00	5.00E+05	5.00E+05	100.00	25
		EXP.	680.6308	680.6320	680.6325	680.6366	100.00	0.00	5.00E+05	6.03E+05	100.00	25
		P.AD.	680.6301	680.6324	680.6329	680.6380	100.00	8.00	5.00E+05	5.36E+05	100.00	25
g10	7049.2480	NONE	7059.9290	7154.0361	7169.1473	7440.0397	100.00	0.00	5.00E+05	5.00E+05	98.72	25
		EXP.	7049.7290	7105.7654	7140.7933	7348.4473	100.00	0.00	5.00E+05	5.73E+05	97.04	25
		P.AD.	7118.8730	7489.9450	7570.7811	8155.6550	96.00	0.00	5.00E+05	6.20E+05	82.88	25
g11	0.7499	NONE	0.7499	0.7499	0.7499	0.7499	100.00	100.00	5.00E+05	5.00E+05	99.20	25
		EXP.	0.7499	0.7499	0.7499	0.7500	100.00	100.00	5.00E+05	5.19E+05	99.68	25
		P.AD.	0.7499	0.7499	0.7499	0.7499	100.00	100.00	5.00E+05	5.95E+05	90.24	25
g12	-1.0000	NONE	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	5.00E+05	5.00E+05	100.00	25
		EXP.	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	5.00E+05	5.48E+05	100.00	25
		P.AD.	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	5.00E+05	5.16E+05	100.00	25
g13	0.0539	NONE	0.1701	0.6003	0.6324	0.9837	100.00	0.00	5.00E+05	5.00E+05	38.80	25
		EXP.	0.2780	0.7206	0.7063	0.9969	100.00	0.00	5.00E+05	5.08E+05	40.96	25
		P.AD.	0.0539	0.0541	0.1312	0.4397	100.00	36.00	5.00E+05	6.29E+05	79.92	25

Table 8.5. Statistical results obtained for the 13 problems in the test suite for the PM constraint-handling technique coupled with the proposed pseudo adaptive scheme. The percentages of feasible solutions and successful solutions (error not greater than 10^{-4}); the mean numbers of FEs and CEs; and the mean percentage of feasible **pbest**s at the end of the search are also provided. Results from (Toscano Pulido & Coello Coello, 2004) and (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005) (PESO) are provided for reference.

Problem	OPTIMUM	OPTIMIZER	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Mean CEs	Mean [%] Feasible pbests	Runs
g01	-15.0000	P.AD. PM	-15.0000	-15.0000	-15.0000	-15.0000	100.00	100.00	5.00E+05	5.75E+05	25	-15.0000
		Toscano Pulido et al.	-15.0000	-	-15.0000	-15.0000	-	-	3.40E+05	-	30	-15.0000
		Muñoz Zavala et al.	-15.0000	-15.0000	-15.0000	-15.0000	-	-	3.40E+05	-	30	-15.0000
g02	-0.8036	P.AD. PM	-0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	5.00E+05	5.08E+05	25	-0.8036
		Toscano Pulido et al.	-0.8034	-	-0.7904	-0.7504	-	-	3.40E+05	-	30	-0.8034
		Muñoz Zavala et al.	-0.7926	-0.7317	-0.7217	0.6141	-	-	3.40E+05	-	30	-0.7926
g03	-1.0005	P.AD. PM	-1.0005	-1.0005	-1.0005	-1.0005	100.00	100.00	5.00E+05	5.84E+05	25	-1.0005
		Toscano Pulido et al.	-1.0047	-	-1.0038	-1.0025	-	-	3.40E+05	-	30	-1.0047
		Muñoz Zavala et al.	-1.0050	-1.0050	-1.0050	-1.0050	-	-	3.40E+05	-	30	-1.0050
g04	-30665.5387	P.AD. PM	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	5.00E+05	5.14E+05	25	-30665.5387
		Toscano Pulido et al.	-30665.5000	-30665.5000	-30665.5000	-30665.5000	-	-	3.40E+05	-	30	-30665.5000
		Muñoz Zavala et al.	-30665.5387	-30665.5387	-30665.5387	-30665.5387	-	-	3.40E+05	-	30	-30665.5387
g05	5126.4967	P.AD. PM	5126.5938	5130.1227	5142.2653	5318.2998	100.00	0.00	5.00E+05	6.31E+05	25	5126.5938
		Toscano Pulido et al.	5126.6400	-	5461.0813	6104.7500	-	-	3.40E+05	-	30	5126.6400
		Muñoz Zavala et al.	5126.4842	5126.5383	5129.1783	5148.8594	-	-	3.40E+05	-	30	5126.4842
g06	-6961.8139	P.AD. PM	-6961.8139	-6961.8139	-6961.8139	-6961.8139	100.00	100.00	5.00E+05	6.38E+05	25	-6961.8139
		Toscano Pulido et al.	-6961.8100	-	-6961.8100	-6961.8100	-	-	3.40E+05	-	30	-6961.8100
		Muñoz Zavala et al.	-6961.8139	-6961.8139	-6961.8139	-6961.8139	-	-	3.40E+05	-	30	-6961.8139
g07	24.3062	P.AD. PM	24.3222	24.4835	24.5153	24.9482	100.00	0.00	5.00E+05	5.58E+05	25	24.3222
		Toscano Pulido et al.	24.3511	-	25.3558	27.3168	-	-	3.40E+05	-	30	24.3511
		Muñoz Zavala et al.	24.3069	24.3713	24.3713	24.5935	-	-	3.40E+05	-	30	24.3069
g08	-0.0958	P.AD. PM	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	5.00E+05	5.20E+05	25	-0.0958
		Toscano Pulido et al.	-0.0958	-	-0.0958	-0.0958	-	-	3.40E+05	-	30	-0.0958
		Muñoz Zavala et al.	-0.0958	-0.0958	0.0958	-0.0958	-	-	3.40E+05	-	30	-0.0958
g09	680.6301	P.AD. PM	680.6301	680.6324	680.6329	680.6380	100.00	8.00	5.00E+05	5.36E+05	25	680.6301
		Toscano Pulido et al.	680.6380	-	680.8524	681.5530	-	-	3.40E+05	-	30	680.6380
		Muñoz Zavala et al.	680.6301	680.6301	680.6301	680.6301	-	-	3.40E+05	-	30	680.6301
g10	7049.2480	P.AD. PM	7118.8730	7489.9450	7570.7811	8155.6550	96.00	0.00	5.00E+05	6.20E+05	25	7118.8730
		Toscano Pulido et al.	7057.5900	-	7560.0479	8104.3100	-	-	3.40E+05	-	30	7057.5900
		Muñoz Zavala et al.	7049.4595	7069.9262	7099.1014	7251.3962	-	-	3.40E+05	-	30	7049.4595
g11	0.7499	P.AD. PM	0.7499	0.7499	0.7499	0.7499	100.00	100.00	5.00E+05	5.95E+05	25	0.7499
		Toscano Pulido et al.	0.7500	-	0.7501	0.7529	-	-	3.40E+05	-	30	0.7500
		Muñoz Zavala et al.	0.7490	0.7490	0.7490	0.7490	-	-	3.40E+05	-	30	0.7490
g12	-1.0000	P.AD. PM	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	5.00E+05	5.16E+05	25	-1.0000
		Toscano Pulido et al.	-1.0000	-	-1.0000	-1.0000	-	-	3.40E+05	-	30	-1.0000
		Muñoz Zavala et al.	-1.0000	-1.0000	-1.0000	-1.0000	-	-	3.40E+05	-	30	-1.0000
g13	0.0539	P.AD. PM	0.0539	0.0541	0.1312	0.4397	100.00	36.00	5.00E+05	6.29E+05	25	0.0539
		Toscano Pulido et al.	0.0687	-	1.7164	13.6695	-	-	3.40E+05	-	30	0.0687
		Muñoz Zavala et al.	0.0815	0.6319	0.6269	0.9976	-	-	3.40E+05	-	30	0.0815

Due to the tolerance relaxations, intermediate solutions that are temporarily regarded as feasible are smaller than the actual feasible minimum. Hence the best solution might increase rather than decrease as the search progresses (see Fig. 8.6 for problem g05).

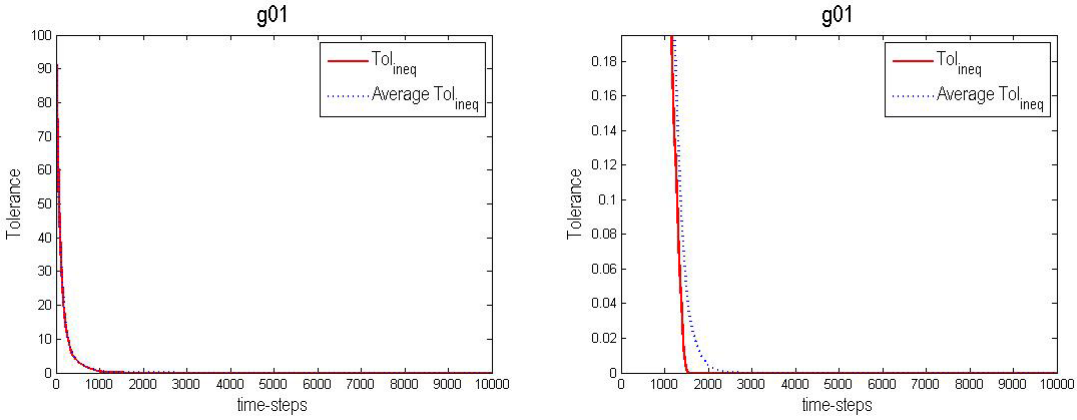


Fig. 8.2. Pseudo Adaptive tolerance (for the PM) for inequality constraint violations in problem g01. The average is among the 25 runs. The figure on the right is just a zoom of the one on the left.

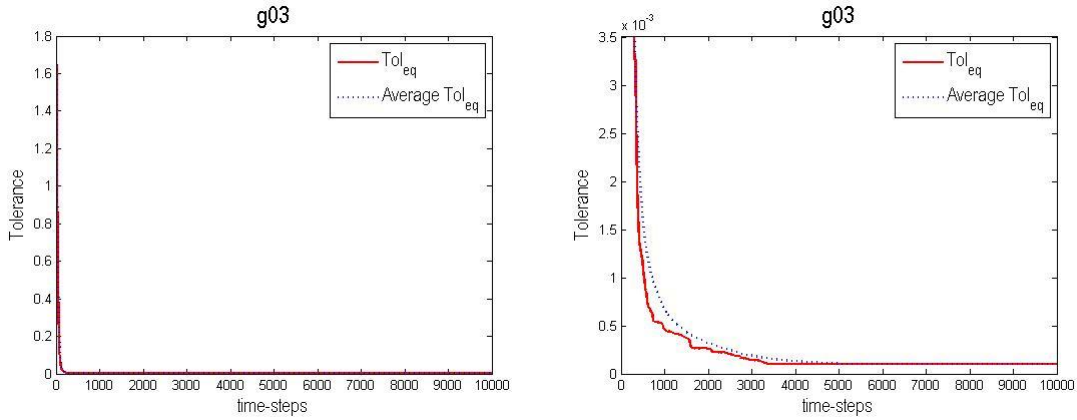


Fig. 8.3. Pseudo Adaptive tolerance (for the PM) for equality constraint violations in problem g03. The average is among the 25 runs. The figure on the right is just a zoom of the one on the left.

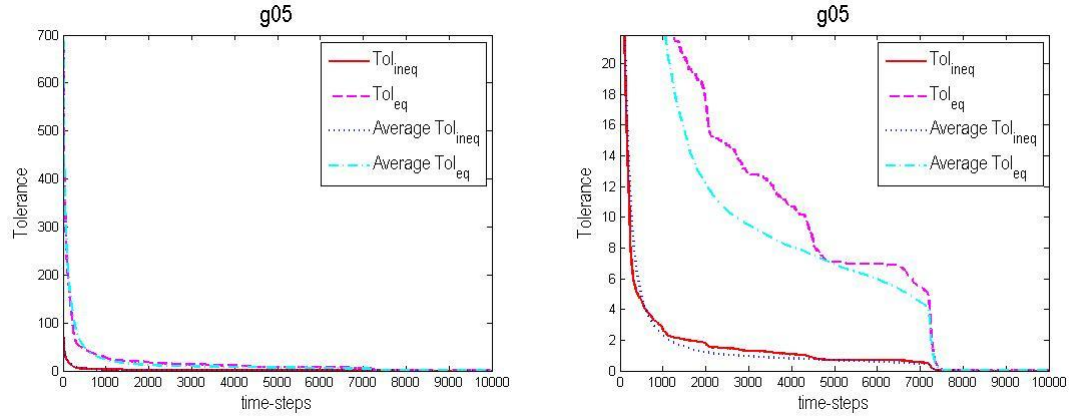


Fig. 8.4. Pseudo Adaptive tolerance (for the PM) for inequality and equality constraint violations in problem g05. The average is among the 25 runs. The figure on the right is just a zoom of the one on the left.

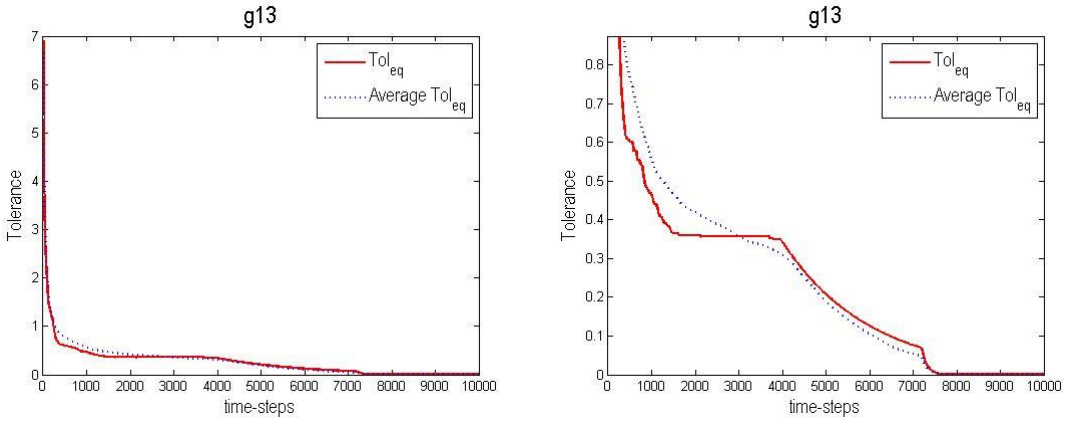


Fig. 8.5. Pseudo Adaptive tolerance (for the PM) for equality constraint violations in problem g13. The average is among the 25 runs. The figure on the right is just a zoom of the one on the left.

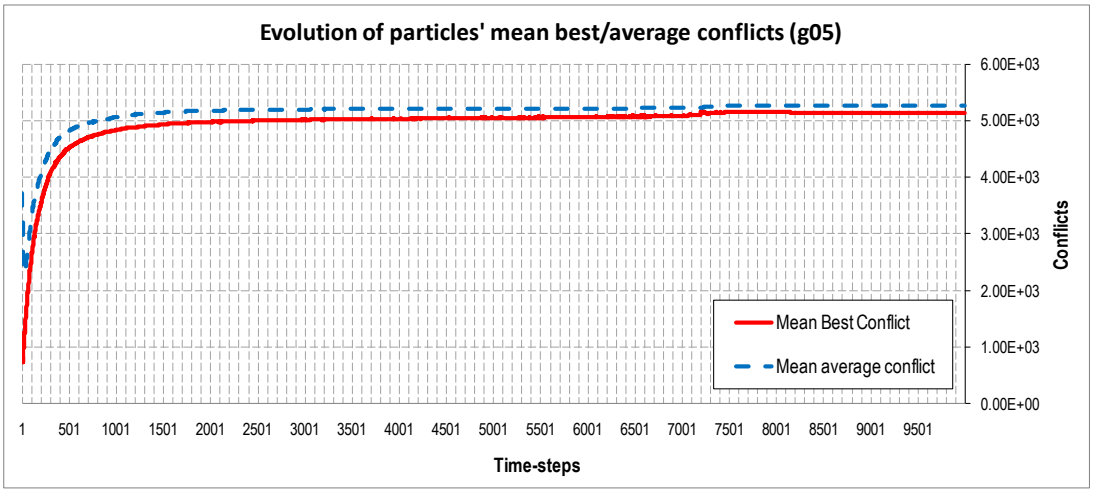


Fig. 8.6. Evolution of the mean best and average conflict values among 25 runs for problem g05 (for the PM). Recall that this is a minimization problem. The shape of the curve is due to the tolerance relaxations. The average conflict is among the conflicts of **pbest**s (best individual positions) rather than among the conflicts of **p**s (current positions).

8.7. Pseudo Adaptive PFPR with Repair Operator

By tracking the evolution of the optimization with the 'pseudo adaptive PFPR' technique in some of the most difficult problems in the test suite, it was realized that a potentially good solution was often lost during a tolerance update. One of the main reasons for this is that the CHT implemented gives absolute priority to a feasible solution over an infeasible one. Hence whenever a global best (**gbest**) becomes infeasible due to a

tolerance update, any individual experience (**pbest**) which remains feasible is regarded as a better solution regardless of its location and conflict.

A rudimentary repair operator was implemented to attempt to find a feasible solution near **gbest** whenever the latter becomes infeasible due to a tolerance update. Thus the operator only acts on **gbest** and not at every time-step but between irregular intervals. The operator consists of calculating an approximate gradient of the constraint violations function (*cv*) defined in Eq. (8.2). However, *cv* is typically highly nonlinear, so that it is not always straightforward to repair **gbest** by following ($-\nabla cv$). In some cases, *cv* is decreased but **gbest** is still infeasible, so that it might still be lost. Besides, an appropriate step-size is also not straightforward. The computation of the scaling factor used to calculate the step-size at each iteration of the repair algorithm is shown in Eq. (8.18), and the generation of the next position (following the direction of $-\nabla cv$) is shown in Eq. (8.19). If the components of ∇cv are all zeros, Eq. (8.19) is replaced by Eq. (8.20).

$$sf = \min\left(\frac{x_{\max 1,j} - x_{\min 1,j}}{100} \cdot \frac{1}{abs\left(\nabla cv_{1,j}\right)}\right) \tag{8.18}$$

$$x_{1,j}^{(iteration)} = x_{1,j}^{(iteration-1)} - sf \cdot \nabla cv_{1,j} \tag{8.19}$$

$$x_{1,j}^{(iteration)} = x_{1,j}^{(iteration-1)} + \left(2 \cdot U_{(0,1)} - 1\right) \cdot \frac{x_{\max 1,j} - x_{\min 1,j}}{100} \tag{8.20}$$

If $x_{1,j}^{(iteration)}$ goes out of boundaries, the 'cut-off at the boundary' mechanism is activated, in which case the new location will not be following the actual $-\nabla cv$.

Once a new location has been generated, the function *cv* is evaluated. If the new location is feasible, the procedure is terminated. If the new solution is not feasible but the *cv* was reduced, the procedure is repeated from this new location (calculating the new gradient) for a maximum of 1000 iterations (it_1). If the new location is not feasible and the *cv* was not reduced, then the new location is discarded, *sf* is reduced to a 50%, and the procedure is repeated for a maximum of 100 iterations (it_2) for the same gradient. Finally, if the *cv* cannot be reduced after 100 iterations by following $-\nabla cv$, the same procedure is repeated but the new location is randomly generated as shown in Eq. (8.21).

$$x_{1,j}^{(iteration)} = x_{1,j}^{(iteration-1)} + \left(2 \cdot U_{(0,1)} - 1\right) \cdot \frac{x_{\max 1,j} - x_{\min 1,j}}{1000} \tag{8.21}$$

This operator is certainly poor and inefficient. The aim here was not to propose or develop a specific repair operator, but simply to confirm the suspicion that repairing **gbest** would lead to noticeable improvement of the solutions found using this Pseudo Adaptive scheme. Coupling a better repairing local search should be tested in the future.

8.7.1. Experimental results

The same settings used in the experiments in section 8.5.1 are used here, but now the CHT is the 'Pseudo Adaptive PFPR with Repair operator'. The results obtained are offered in Table 8.6, together with those obtained with the plain 'Pseudo Adaptive PFPR' and with the 'Pseudo Adaptive PM'.

For reference, results are also presented together with those in (Toscano Pulido & Coello Coello, 2004) and in (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005) (PESO) in Table 8.7; with those in (Fuentes Cabrera & Coello Coello, 2007) and in (Muñoz Zavala, Hernández Aguirre, Villa Diharce, & Botello Rionda, 2006) in Table 8.8; and the success rates (SRs) are compared to those in (Zielinski & Laur, 2006) and in (Muñoz Zavala, Hernández Aguirre, Villa Diharce, & Botello Rionda, 2006) in Table 8.9. Finally, results reported by numerous other authors using different types of optimizers are offered in Table 8.10 to Table 8.12.

The full output files are provided in *.xlsx and *.mat formats in the digital appendix, from where a great amount of information with regards to the search can be obtained.

8.7.2. Discussion

From Table 8.6, it can be observed that the repair operator improves the performance of the 'Pesudo-Adaptive PFPR' technique. The failure in problem g01 is resolved; solutions for problem g05 are improved; while the mean solution and the percentage of successful solutions for problem g13 are improved by the repair operator.

From Table 8.7, the 'Pesudo Adaptive PFPR with Repair operator' clearly outperforms (Toscano Pulido & Coello Coello, 2004), while its performance is competitive with that of the PESO in (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005). From Table 8.8, the proposed algorithm finds better results than (Fuentes Cabrera & Coello Coello, 2007), while its performance is not as good as that of the PESO+ in (Muñoz Zavala, Hernández Aguirre, Villa Diharce, & Botello Rionda, 2006). It is fair to note, however, that the PESO and PESO+ are equipped with a diversity operator similar to the differential mutation in Differential Evolution (DE). Table 8.9 also shows better performance of the PESO+ in terms of the percentage of successful solutions. In turn, the proposed algorithm performs better than the PSO in (Zielinski & Laur, 2006), in particular for problems g01, g02, g03, g05 and g13. Table 8.10 to Table 8.12 show that the results obtained by the proposed algorithm are competitive with the results obtained by a number of different approaches in the literature, outperforming several of them. Arguably, the best performer in those tables is the αNSM in (Takahama & Sakai, 2005).

Top-notch PSO algorithms tested not only on these 13 benchmark problems but also on 11 additional ones extended in (Liang, et al., 2006) for the '2006 IEEE Congress on Evolutionary Computation' (CEC'06) are the aforementioned PESO+ in (Muñoz Zavala, Hernández Aguirre, Villa Diharce, & Botello Rionda, 2006) and the 'Dynamic Multi-Swarm PSO' in (Liang & Suganthan, 2006).

8.8. Engineering applications

A few classical engineering optimization problems are dealt with hereafter with the proposed optimizer. Given that the aim is a general-purpose optimizer in the sense that it is able to cope reasonably well with different problems, exactly the same settings as before are used in these experiments (without repair operator).

The problems to be solved are the Mixed-Discrete Pressure Vessel Design (MDPVD) in (Coello Coello, 2000) and its continuous version (PVD); the Welded Beam Design (WBD) and the Tension/Compression Spring Design (TCSD) in (Coello Coello, 2000); the Himmelblau Nonlinear Problem (HNP) in (Hu, Eberhart, & Shi, 2003); three variations of the 10-bar truss problem in (Fleury & Schmit, 1980) and in (Haftka &

Gürdal, 1992); and the 25-bar truss problem in (Fleury & Schmit, 1980) as well as one variation found in (Park & Ryu, 2004).

Table 8.6. Statistical results for the proposed 'Pseudo Adaptive PM', 'Pseudo Adaptive PFPR' and 'Pseudo Adaptive PFPR with Repair operator' obtained for the 13 problems in the test suite. The percentages of feasible solutions and successful solutions (error not greater than 10^{-4}); mean numbers of FEs and CEs; and mean percentage of feasible **pbest**s at the end of the search are also provided.

Problem	OPTIMUM	Pseudo Adaptive Tolerance Relaxation	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Mean CEs	Mean [%] Feasible PBESTs	Runs
g01	-15.0000	P.AD. PM	-15.0000	-15.0000	-15.0000	-15.0000	100.00	100.00	5.00E+05	5.75E+05	99.36	25
		P.AD. PFPR	-15.0000	-15.0000	-14.8981	-12.4531	100.00	96.00	1.16E+05	6.42E+05	100.00	25
		P.AD. PFPR + R	-15.0000	-15.0000	-15.0000	-15.0000	100.00	100.00	9.19E+04	2.36E+06	100.00	25
g02	-0.8036	P.AD. PM	-0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	5.00E+05	5.08E+05	100.00	25
		P.AD. PFPR	-0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	2.52E+05	5.08E+05	100.00	25
		P.AD. PFPR + R	-0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	2.52E+05	5.08E+05	100.00	25
g03	-1.0005	P.AD. PM	-1.0005	-1.0005	-1.0005	-1.0005	100.00	100.00	5.00E+05	5.84E+05	99.92	25
		P.AD. PFPR	-1.0005	-1.0005	-1.0005	-1.0005	100.00	100.00	1.25E+05	5.84E+05	99.92	25
		P.AD. PFPR + R	-1.0005	-1.0005	-1.0005	-1.0005	100.00	100.00	1.27E+05	5.91E+05	99.84	25
g04	-30665.5387	P.AD. PM	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	5.00E+05	5.14E+05	100.00	25
		P.AD. PFPR	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	3.22E+05	5.14E+05	100.00	25
		P.AD. PFPR + R	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	3.21E+05	5.13E+05	100.00	25
g05	5126.4967	P.AD. PM	5126.5938	5130.1227	5142.2653	5318.2998	100.00	0.00	5.00E+05	6.31E+05	57.52	25
		P.AD. PFPR	5126.5060	5129.6245	5135.0431	5205.0354	100.00	0.00	1.21E+05	6.28E+05	66.80	25
		P.AD. PFPR + R	5126.4985	5128.9372	5130.8828	5149.1589	100.00	0.00	1.09E+05	9.78E+06	46.32	25
g06	-6961.8139	P.AD. PM	-6961.8139	-6961.8139	-6961.8139	-6961.8139	100.00	100.00	5.00E+05	6.38E+05	100.00	25
		P.AD. PFPR	-6961.8139	-6961.8139	-6961.8139	-6961.8139	100.00	100.00	2.41E+05	6.18E+05	100.00	25
		P.AD. PFPR + R	6961.8139	6961.8139	6961.8139	6961.8139	100.00	100.00	2.99E+05	7.26E+05	100.00	25
g07	24.3062	P.AD. PM	24.3222	24.4835	24.5153	24.9482	100.00	0.00	5.00E+05	5.58E+05	100.00	25
		P.AD. PFPR	24.3246	24.4429	24.4863	25.1151	100.00	0.00	1.60E+05	5.57E+05	100.00	25
		P.AD. PFPR + R	24.3532	24.4868	24.5000	24.8745	100.00	0.00	1.56E+05	5.61E+05	100.00	25
g08	-0.0958	P.AD. PM	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	5.00E+05	5.20E+05	100.00	25
		P.AD. PFPR	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	4.98E+05	5.20E+05	100.00	25
		P.AD. PFPR + R	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	4.98E+05	5.20E+05	100.00	25
g09	680.6301	P.AD. PM	680.6301	680.6324	680.6329	680.6380	100.00	8.00	5.00E+05	5.36E+05	100.00	25
		P.AD. PFPR	680.6305	680.6334	680.6339	680.6406	100.00	0.00	1.81E+05	5.36E+05	100.00	25
		P.AD. PFPR + R	680.6303	680.6323	680.6325	680.6363	100.00	0.00	1.83E+05	5.36E+05	100.00	25
g10	7049.2480	P.AD. PM	7118.8730	7489.9450	7570.7811	8155.6550	96.00	0.00	5.00E+05	6.20E+05	82.88	25
		P.AD. PFPR	7049.5448	7145.6473	7145.1506	7437.8998	100.00	0.00	8.61E+04	6.25E+05	98.00	25
		P.AD. PFPR + R	7049.7951	7142.9934	7167.2404	7480.5126	100.00	0.00	8.77E+04	3.29E+06	99.12	25
g11	0.7499	P.AD. PM	0.7499	0.7499	0.7499	0.7499	100.00	100.00	5.00E+05	5.95E+05	90.24	25
		P.AD. PFPR	0.7499	0.7499	0.7499	0.7499	100.00	100.00	1.20E+05	5.98E+05	88.88	25
		P.AD. PFPR + R	0.7499	0.7499	0.7499	0.7499	100.00	100.00	9.77E+04	6.26E+05	80.88	25
g12	-1.0000	P.AD. PM	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	5.00E+05	5.16E+05	100.00	25
		P.AD. PFPR	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	4.95E+05	5.16E+05	100.00	25
		P.AD. PFPR + R	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	4.95E+05	5.16E+05	100.00	25
g13	0.0539	P.AD. PM	0.0539	0.0541	0.1312	0.4397	100.00	36.00	5.00E+05	6.29E+05	79.92	25
		P.AD. PFPR	0.0539	0.0540	0.1030	0.4391	100.00	64.00	1.11E+05	6.35E+05	76.88	25
		P.AD. PFPR + R	0.0539	0.0540	0.0545	0.0607	100.00	68.00	8.64E+04	7.12E+05	30.56	25

Table 8.7. Statistical results obtained for the 13 problems in the test suite for the 'Pseudo Adaptive PFPR with Repair operator'. The percentages of feasible solutions and successful solutions (error not greater than 10^{-4}); the mean numbers of FEs and CEs; and the mean percentage of feasible **pbest**s at the end of the search are also provided. Results from (Toscano Pulido & Coello Coello, 2004) and (Muñoz Zavala, Hernández Aguirre, & Villa Diharce, 2005) (PESO) are provided for reference.

Problem	OPTIMUM	OPTIMIZER	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Mean CEs	Mean [%] Feasible pbests	Runs
g01	-15.0000	P.AD. PFPR + R	-15.0000	-15.0000	-15.0000	-15.0000	100.00	100.00	9.19E+04	2.36E+06	100.00	25
		Toscano Pulido et al.	-15.0000	-	-15.0000	-15.0000	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-15.0000	-15.0000	-15.0000	-15.0000	-	-	3.40E+05	-	-	30
g02	-0.8036	P.AD. PFPR + R	-0.8036	-0.8034	-0.7949	-0.7581	100.00	48.00	2.52E+05	5.08E+05	100.00	25
		Toscano Pulido et al.	-0.8034	-	-0.7904	-0.7504	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-0.7926	-0.7317	-0.7217	0.6141	-	-	3.40E+05	-	-	30
g03	-1.0005	P.AD. PFPR + R	-1.0005	-1.0005	-1.0005	-1.0005	100.00	100.00	1.27E+05	5.91E+05	99.84	25
		Toscano Pulido et al.	-1.0047	-	-1.0038	-1.0025	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-1.0050	-1.0050	-1.0050	-1.0050	-	-	3.40E+05	-	-	30
g04	-30665.5387	P.AD. PFPR + R	-30665.5387	-30665.5387	-30665.5387	-30665.5387	100.00	100.00	3.21E+05	5.13E+05	100.00	25
		Toscano Pulido et al.	-30665.5000	-30665.5000	-30665.5000	-30665.5000	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-30665.5387	-30665.5387	-30665.5387	-30665.5387	-	-	3.40E+05	-	-	30
g05	5126.4967	P.AD. PFPR + R	5126.4985	5128.9372	5130.8828	5149.1589	100.00	0.00	1.09E+05	9.78E+06	46.32	25
		Toscano Pulido et al.	5126.6400	-	5461.0813	6104.7500	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	5126.4842	5126.5383	5129.1783	5148.8594	-	-	3.40E+05	-	-	30
g06	-6961.8139	P.AD. PFPR + R	6961.8139	6961.8139	6961.8139	6961.8139	100.00	100.00	2.99E+05	7.26E+05	100.00	25
		Toscano Pulido et al.	-6961.8100	-	-6961.8100	-6961.8100	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-6961.8139	-6961.8139	-6961.8139	-6961.8139	-	-	3.40E+05	-	-	30
g07	24.3062	P.AD. PFPR + R	24.3532	24.4868	24.5000	24.8745	100.00	0.00	1.56E+05	5.61E+05	100.00	25
		Toscano Pulido et al.	24.3511	-	25.3558	27.3168	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	24.3069	24.3713	24.3713	24.5935	-	-	3.40E+05	-	-	30
g08	-0.0958	P.AD. PFPR + R	-0.0958	-0.0958	-0.0958	-0.0958	100.00	100.00	4.98E+05	5.20E+05	100.00	25
		Toscano Pulido et al.	-0.0958	-	-0.0958	-0.0958	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-0.0958	-0.0958	0.0958	-0.0958	-	-	3.40E+05	-	-	30
g09	680.6301	P.AD. PFPR + R	680.6303	680.6323	680.6325	680.6363	100.00	0.00	1.83E+05	5.36E+05	100.00	25
		Toscano Pulido et al.	680.6380	-	680.8524	681.5530	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	680.6301	680.6301	680.6301	680.6301	-	-	3.40E+05	-	-	30
g10	7049.2480	P.AD. PFPR + R	7049.7951	7142.9934	7167.2404	7480.5126	100.00	0.00	8.77E+04	3.29E+06	99.12	25
		Toscano Pulido et al.	7057.5900	-	7560.0479	8104.3100	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	7049.4595	7069.9262	7099.1014	7251.3962	-	-	3.40E+05	-	-	30
g11	0.7499	P.AD. PFPR + R	0.7499	0.7499	0.7499	0.7499	100.00	100.00	9.77E+04	6.26E+05	80.88	25
		Toscano Pulido et al.	0.7500	-	0.7501	0.7529	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	0.7490	0.7490	0.7490	0.7490	-	-	3.40E+05	-	-	30
g12	-1.0000	P.AD. PFPR + R	-1.0000	-1.0000	-1.0000	-1.0000	100.00	100.00	4.95E+05	5.16E+05	100.00	25
		Toscano Pulido et al.	-1.0000	-	-1.0000	-1.0000	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	-1.0000	-1.0000	-1.0000	-1.0000	-	-	3.40E+05	-	-	30
g13	0.0539	P.AD. PFPR + R	0.0539	0.0540	0.0545	0.0607	100.00	68.00	8.64E+04	7.12E+05	30.56	25
		Toscano Pulido et al.	0.0687	-	1.7164	13.6695	-	-	3.40E+05	-	-	30
		Muñoz Zavala et al.	0.0815	0.6319	0.6269	0.9976	-	-	3.40E+05	-	-	30

Table 8.8. Statistical results obtained for the 13 problems in the test suite for the 'Pseudo Adaptive PFPR with Repair operator'. The percentages of feasible solutions and successful solutions (error not greater than 10^{-4}); the mean numbers of FEs and CEs; and the mean percentage of feasible **pbest**s at the end of the search are also provided. Results from (Fuentes Cabrera & Coello Coello, 2007) and (Muñoz Zavala, Hernández Aguirre, Villa Diharce, & Botello Rionda, 2006) (PESO+) are provided for reference.

Problem	OPTIMUM	OPTIMIZER	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Mean CEs	Mean [%] Feasible pbests	Runs
g01	-15.0000	P.AD. PFPR + R	-15.000000	-15.000000	-15.000000	-15.000000	100.00	100.00	9.19E+04	2.36E+06	100.00	25
		Fuentes Cabr.	-15.000100	-13.000100	-13.273400	-9.701200	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	-15.000000	-15.000000	-15.000000	-15.000000	100	100	5.00E+05	-	-	25
g02	-0.8036	P.AD. PFPR + R	-0.803618	-0.803429	-0.794852	-0.758093	100.00	48.00	2.52E+05	5.08E+05	100.00	25
		Fuentes Cabr.	0.803620	0.778481	0.777143	0.711603	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	-0.803619	-0.803616	-0.800062	-0.785266	100	56	5.00E+05	-	-	25
g03	-1.0005	P.AD. PFPR + R	-1.000500	-1.000497	-1.000495	-1.000478	100.00	100.00	1.27E+05	5.91E+05	99.84	25
		Fuentes Cabr.	-1.000400	-1.000400	-0.993600	-0.667400	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	-1.000500	-1.000500	-1.000500	-1.000499	100	100	5.00E+05	-	-	25
g04	-30665.5387	P.AD. PFPR + R	-30665.538672	-30665.538672	-30665.538672	-30665.538672	100.00	100.00	3.21E+05	5.13E+05	100.00	25
		Fuentes Cabr.	-30665.539800	-30665.539800	-30665.539700	-30665.533800	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	-30665.538672	-30665.538672	-30665.538672	-30665.538672	100	100	5.00E+05	-	-	25
g05	5126.4967	P.AD. PFPR + R	5126.498527	5128.937242	5130.882764	5149.158908	100.00	0.00	1.09E+05	9.78E+06	46.32	25
		Fuentes Cabr.	5126.646700	5261.767500	5495.238900	6272.742300	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	5126.496714	5126.496714	5126.496714	5126.496714	100	100	5.00E+05	-	-	25
g06	-6961.8139	P.AD. PFPR + R	6961.813876	6961.813876	6961.813876	6961.813876	100.00	100.00	2.99E+05	7.26E+05	100.00	25
		Fuentes Cabr.	-6961.837100	-6961.837100	-6961.837000	-6961.835500	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	-6961.813876	-6961.813876	-6961.813876	-6961.813876	100	100	5.00E+05	-	-	25
g07	24.3062	P.AD. PFPR + R	24.353160	24.486815	24.500021	24.874481	100.00	0.00	1.56E+05	5.61E+05	100.00	25
		Fuentes Cabr.	24.327800	24.645500	24.699600	25.296200	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	24.306209	24.306214	24.306223	24.306301	100	16	5.00E+05	-	-	25
g08	-0.0958	P.AD. PFPR + R	-0.095825	-0.095825	-0.095825	-0.095825	100.00	100.00	4.98E+05	5.20E+05	100.00	25
		Fuentes Cabr.	-0.095825	-0.095825	-0.095825	-0.095825	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	-0.095825	-0.095825	-0.095825	-0.095825	100	100	5.00E+05	-	-	25
g09	680.6301	P.AD. PFPR + R	680.630324	680.632332	680.632495	680.636292	100.00	0.00	1.83E+05	5.36E+05	100.00	25
		Fuentes Cabr.	680.630700	680.637800	680.639100	680.667100	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	680.630057	680.630057	680.630057	680.630057	100	100	5.00E+05	-	-	25
g10	7049.2480	P.AD. PFPR + R	7049.795137	7142.993354	7167.240386	7480.512601	100.00	0.00	8.77E+04	3.29E+06	99.12	25
		Fuentes Cabr.	7090.452400	7557.431400	7747.629800	10533.665800	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	7049.248027	7049.250013	7049.262277	7049.349764	100	16	5.00E+05	-	-	25
g11	0.7499	P.AD. PFPR + R	0.749900	0.749900	0.749900	0.749901	100.00	100.00	9.77E+04	6.26E+05	80.88	25
		Fuentes Cabr.	0.749900	0.749900	0.767300	0.992500	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	0.749999	0.749999	0.749999	0.749999	100	100	5.00E+05	-	-	25
g12	-1.0000	P.AD. PFPR + R	-1.000000	-1.000000	-1.000000	-1.000000	100.00	100.00	4.95E+05	5.16E+05	100.00	25
		Fuentes Cabr.	-1.000000	-1.000000	-1.000000	-1.000000	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	-1.000000	-1.000000	-1.000000	-1.000000	100	100	5.00E+05	-	-	25
g13	0.0539	P.AD. PFPR + R	0.053945	0.053985	0.054494	0.060726	100.00	68.00	8.64E+04	7.12E+05	30.56	25
		Fuentes Cabr.	0.059410	0.909530	0.813350	2.444150	-	-	2.40E+05	-	-	50
		Muñoz Zavala +	0.053942	0.053942	0.053946	0.054022	100	100	5.00E+05	-	-	25

Table 8.9. Percentage of successful solutions (error not greater than 10^{-4}) obtained for the 13 problems in the test suite by the 'Pseudo Adaptive PFPR with Repair operator'. The percentages of feasible solutions and the mean numbers of FEs are also provided. Results from (Zielinski & Laur, 2006) and (Muñoz Zavala, Hernández Aguirre, Villa Diharce, & Botello Rionda, 2006) (PESO+) are provided for reference.

Problem	OPTIMUM	OPTIMIZER	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Runs
g01	-15.000000	P.AD. PFPR + R	100.00	100.00	9.19E+04	25
		Zielinski & Laur	100.00	52.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25
g02	-0.803619	P.AD. PFPR + R	100.00	48.00	2.52E+05	25
		Zielinski & Laur	100.00	0.00	5.00E+05	25
		Muñoz Zavala +	100.00	56.00	5.00E+05	25
g03	-1.000500	P.AD. PFPR + R	100.00	100.00	1.27E+05	25
		Zielinski & Laur	100.00	0.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25
g04	-30665.538672	P.AD. PFPR + R	100.00	100.00	3.21E+05	25
		Zielinski & Laur	100.00	100.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25
g05	5126.496714	P.AD. PFPR + R	100.00	0.00	1.09E+05	25
		Zielinski & Laur	100.00	16.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25
g06	-6961.813876	P.AD. PFPR + R	100.00	100.00	2.99E+05	25
		Zielinski & Laur	100.00	100.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25
g07	24.306209	P.AD. PFPR + R	100.00	0.00	1.56E+05	25
		Zielinski & Laur	100.00	8.00	5.00E+05	25
		Muñoz Zavala +	100.00	96.00	5.00E+05	25
g08	-0.095825	P.AD. PFPR + R	100.00	100.00	4.98E+05	25
		Zielinski & Laur	100.00	100.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25
g09	680.630057	P.AD. PFPR + R	100.00	0.00	1.83E+05	25
		Zielinski & Laur	100.00	100.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25
g10	7049.248021	P.AD. PFPR + R	100.00	0.00	8.77E+04	25
		Zielinski & Laur	100.00	32.00	5.00E+05	25
		Muñoz Zavala +	100.00	16.00	5.00E+05	25
g11	0.749900	P.AD. PFPR + R	100.00	100.00	9.77E+04	25
		Zielinski & Laur	100.00	100.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25
g12	-1.000000	P.AD. PFPR + R	100.00	100.00	4.95E+05	25
		Zielinski & Laur	100.00	100.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25
g13	0.053942	P.AD. PFPR + R	100.00	68.00	8.64E+04	25
		Zielinski & Laur	100.00	0.00	5.00E+05	25
		Muñoz Zavala +	100.00	100.00	5.00E+05	25

Table 8.10. Best and mean solutions obtained by the 'Pseudo Adaptive PFPR with Repair operator' and by different authors and different approaches in the literature for problems g01 to g06 from the benchmark suite of constrained problems in appendix II. The authors included are: (Runarsson & Yao, 2000); (Hamida & Schoenauer, 2002); (Farmani & Wright, 2003); (Takahama & Sakai, 2004); (Takahama & Sakai, 2005); (Landa Becerra & Coello Coello, 2005); (Zheng, Wu, & Song, 2007); (Hernández Aguirre, Villa Diharce, & Coello Coello, 2007); (Parsopoulos & Vrahatis, 2002); (de Freitas Vaz & da Graça Pinto Fernandes, 2006); and (He & Wang, 2007).

OPTIMIZER	g01				g02				Runs
	OPTIMUM	BEST	MEAN	Mean FEs	OPTIMUM	BEST	MEAN	Mean FEs	
P.AD. PFPR + R	-15.0000	-15.0000	-15.0000	9.19E+04	-0.8036	-0.8036	-0.7949	2.52E+05	25
Runarsson et al. (ES + SR)		-15.0000	-15.0000	3.50E+05		-0.8035	-0.7820	3.50E+05	30
Hamida et al. (ES + ASCHEA)		-15.0000	-14.8400	1.50E+06		-0.7850	-0.5900	1.50E+06	31
Farmani et al. (GA + SAFF)		-15.0000	-15.0000	1.40E+06		-0.8030	-0.7901	1.40E+06	20
Takahama et al. (αGA)		-15.0000	-15.0000	3.50E+05		-	-	-	100
Takahama et al. (αNSM)		-15.0000	-15.0000	8.34E+04		-0.8036	-0.7842	8.34E+04	30
Landa Becerra et al. (CDE)		-15.0000	-15.0000	1.00E+05		-0.8036	-0.7249	1.00E+05	30
Zheng et al. (IPSO)		-	-	-		-0.8035	-0.7899	3.40E+05	30
Hernández Aguirre et al. (NOPREDA + SR)		-15.0000	-14.8400	3.50E+05		-0.8036	-0.7806	3.50E+05	50

OPTIMIZER	g03				g04				Runs
	OPTIMUM	BEST	MEAN	Mean FEs	OPTIMUM	BEST	MEAN	Mean FEs	
P.AD. PFPR + R	-1.0005	-1.0005	-1.0005	1.27E+05	-30665.5387	-30665.5387	-30665.5387	3.21E+05	25
Runarsson et al. (ES + SR)		-1.0000	-1.0000	3.50E+05		-30665.5390	-30665.5390	3.50E+05	30
Hamida et al. (ES + ASCHEA)		-1.0000	-0.9999	1.50E+06		-30665.5000	-30665.5000	1.50E+06	31
Parsopoulos et al. (global PSO-Co + Pena)		-	-	-		-31542.5780	-31528.2890	1.00E+05	10
Farmani et al. (GA + SAFF)		-1.0000	-0.9999	1.40E+06		-30665.5000	-30665.2000	1.40E+06	20
Takahama et al. (αNSM)		-1.0005	-1.0005	8.34E+04		-30665.5387	-30665.5387	8.34E+04	30
Landa Becerra et al. (CDE)		-0.9954	-0.7886	1.00E+05		-30665.5387	-30665.5387	1.00E+05	30
de Freitas Vaz et al. (global PSO + MO)		-	-	-		-30665.5000	-	9.75E+05	-
He et al. (PSO + SA)		-	-	-		-30665.5390	-30665.5390	8.10E+04	30
Hernández Aguirre et al. (NOPREDA + SR)		-0.9999	-0.9979	3.50E+05		-30665.5387	-30664.3957	3.50E+05	50

OPTIMIZER	g05				g06				Runs
	OPTIMUM	BEST	MEAN	Mean FEs	OPTIMUM	BEST	MEAN	Mean FEs	
P.AD. PFPR + R	5126.4967	5126.4985	5130.8828	1.09E+05	-6961.8139	6961.8139	6961.8139	2.99E+05	25
Runarsson et al. (ES + SR)		5126.4970	5128.8810	3.50E+05		-6961.8140	-6875.9400	3.50E+05	30
Hamida et al. (ES + ASCHEA)		5126.5000	5141.6500	1.50E+06		-6961.8100	-6961.8100	1.50E+06	31
Parsopoulos et al. (global PSO-Co + Pena)		-	-	-		-6961.837 (*)	-6961.8360	1.00E+05	10
Farmani et al. (GA + SAFF)		5126.9890	5432.0800	1.40E+06		-6961.8000	-6961.8000	1.40E+06	20
Takahama et al. (αNSM)		5126.4967	5126.4967	8.34E+04		-6961.8139	-6961.8139	8.34E+04	30
Landa Becerra et al. (CDE)		5126.5709	5207.4107	1.00E+05		-6961.8139	-6961.8139	1.00E+05	30
de Freitas Vaz et al. (global PSO + MO)		-	-	-		-6961.8100	-	1.46E+06	-
Zheng et al. (IPSO)		-	-	-		-6961.8140	-6961.8140	3.40E+05	30
Hernández Aguirre et al. (NOPREDA + SR)		No feasible solution found in 50 runs		3.50E+05		-6961.8139	-6961.8139	3.50E+05	50

(*) Probably a typo (value below optimum).

Table 8.11. Best and mean solutions obtained by the 'Pseudo Adaptive PFPR with Repair operator' and by different authors and different approaches in the literature for problems g07 to g12 from the benchmark suite of constrained problems in appendix II. The authors included are: (Runarsson & Yao, 2000); (Hamida & Schoenauer, 2002); (Farmani & Wright, 2003); (Takahama & Sakai, 2004); (Takahama & Sakai, 2005); (Landa Becerra & Coello Coello, 2005); (Zheng, Wu, & Song, 2007); (Hernández Aguirre, Villa Diharce, & Coello Coello, 2007); (Parsopoulos & Vrahatis, 2002); (de Freitas Vaz & da Graça Pinto Fernandes, 2006); and (He & Wang, 2007).

OPTIMIZER	g07				g08				Runs
	OPTIMUM	BEST	MEAN	Mean FEs	OPTIMUM	BEST	MEAN	Mean FEs	
P.AD. PFPR + R	24.3062	24.3532	24.5000	1.56E+05	-0.0958	-0.0958	-0.0958	4.98E+05	25
Runarsson et al. (ES + SR)		24.3070	24.3740	3.50E+05		-0.0958	-0.0958	3.50E+05	30
Hamida et al. (ES + ASCHEA)		24.3323	24.6636	1.50E+06		-0.0958	-0.0958	1.50E+06	31
Farmani et al. (GA + SAFF)		24.4800	26.5800	1.40E+06		-0.0958	-0.0958	1.40E+06	20
Takahama et al. (αGA)		24.4010	24.5420	3.50E+05		-	-	-	100
Takahama et al. (αNSM)		24.3062	24.3062	8.34E+04		-0.0958	-0.0958	8.34E+04	30
Landa Becerra et al. (CDE)		24.3062	24.3062	1.00E+05		-0.0958	-0.0958	1.00E+05	30
He et al. (PSO + SA)		-	-	-		-0.0958	-0.0958	8.10E+04	30
Zheng et al. (IPSO)		-	-	-		-0.0958	-0.0958	3.40E+05	30
Hernández Aguirre et al. (NOPREDA + SR)		24.3118	24.5243	3.50E+05		-0.0958	-0.0945	3.50E+05	50
OPTIMIZER	g09				g10				Runs
	OPTIMUM	BEST	MEAN	Mean FEs	OPTIMUM	BEST	MEAN	Mean FEs	
P.AD. PFPR + R	680.6301	680.6303	680.6325	1.83E+05	7049.2480	7049.7951	7167.2404	8.77E+04	25
Runarsson et al. (ES + SR)		680.6300	680.6560	3.50E+05		7054.3160	7559.1920	3.50E+05	30
Hamida et al. (ES + ASCHEA)		680.6300	680.6410	1.50E+06		7061.1300	7497.4340	1.50E+06	31
Parsopoulos et al. (global PSO-Co + Pena)		680.6350	680.6630	1.00E+05		-	-	-	20
Farmani et al. (GA + SAFF)		680.6400	680.7200	1.40E+06		7061.3400	7627.8900	1.40E+06	20
Takahama et al. (αGA)		680.6460	680.6870	3.50E+05		7053.9510	7514.2330	3.50E+05	100
Takahama et al. (αNSM)		680.6301	680.6301	8.34E+04		7049.2480	7049.2480	8.34E+04	30
Landa Becerra et al. (CDE)		680.6301	680.6301	1.00E+05		7049.2481	7049.2483	1.00E+05	30
de Freitas Vaz et al. (global PSO + MO)		680.6300	-	1.69E+06		-	-	-	-
Hernández Aguirre et al. (NOPREDA + SR)		680.6305	680.6415	3.50E+05		7142.8357	9891.6745	3.50E+05	50
OPTIMIZER	g11				g12				Runs
	OPTIMUM	BEST	MEAN	Mean FEs	OPTIMUM	BEST	MEAN	Mean FEs	
P.AD. PFPR + R	0.7499	0.7499	0.7499	9.77E+04	-1.0000	-1.0000	-1.0000	4.95E+05	25
Runarsson et al. (ES + SR)		0.7500	0.7500	3.50E+05		-1.0000	-1.0000	3.50E+05	30
Hamida et al. (ES + ASCHEA)		0.7500	0.7500	1.50E+06		-	-	-	31
Farmani et al. (GA + SAFF)		0.7500	0.7500	1.40E+06		-	-	-	-
Takahama et al. (αNSM)		0.7499	0.7499	8.34E+04	-1.0000	-1.0000	-1.0000	8.34E+04	30
Landa Becerra et al. (CDE)		0.7499	0.7580	1.00E+05		-1.0000	-1.0000	1.00E+05	30
He et al. (PSO + SA)		-	-	-		-1.0000	-1.0000	8.10E+04	30
Zheng et al. (IPSO)		0.7500	0.7500	3.40E+05		-	-	-	30
Hernández Aguirre et al. (NOPREDA + SR)		0.7499	0.7527	3.50E+05		-1.0000	-1.0000	3.50E+05	50

Table 8.12. Best and mean solutions obtained by the 'Pseudo Adaptive PFPR with Repair operator' and by different authors and different approaches in the literature for problem g13 from the benchmark suite of constrained problems in appendix II. The authors included are: (Runarsson & Yao, 2000); (Takahama & Sakai, 2004); (Takahama & Sakai, 2005); (Landa Becerra & Coello Coello, 2005); (Zheng, Wu, & Song, 2007); (Hernández Aguirre, Villa Diharce, & Coello Coello, 2007); and (Wang & Yin, 2008).

OPTIMIZER	g13				Runs
	OPTIMUM	BEST	MEAN	FEs	
P.AD. PFPR + R	0.0539	0.0539	0.0545	8.64E+04	25
Runarsson et al. (ES + SR)		0.0540	0.0675	3.50E+05	30
Takahama et al. (αGA)		0.0540	0.0558	3.50E+05	100
Takahama et al. (αNSM)		0.0539	0.0668	8.34E+04	30
Landa Becerra et al. (CDE)		0.0562	0.2883	1.00E+05	30
Zheng et al. (IPSO)		0.1709	0.8257	3.40E+05	30
Hernández Aguirre et al. (NOPREDA + SR)		0.0540	0.053985 (*)	3.50E+05	50
Wang et al. (RSPSO)		0.0540	0.0565	3.50E+05	30

(*) Only 7 out of 50 runs could find a feasible solution for this problem.

8.8.1. Pressure Vessel Design (PVD)

The problem consists of minimizing the total cost of the material, forming, and welding of a cylindrical vessel capped at both ends by hemispherical heads. There are 4 design variables: x_1 is the thickness of the shell ($\mathbf{T_s}$), x_2 is the thickness of the head ($\mathbf{T_h}$), x_3 is the inner radius (**R**), and x_4 is the length of the cylindrical section of the vessel (**L**). Refer to Fig. 8.7 for a graphical visualization of the problem.

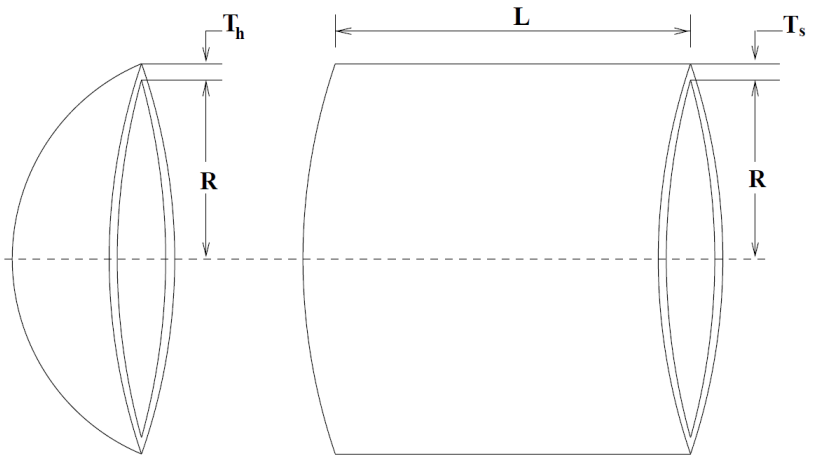


Fig. 8.7. Illustration of the Pressure Vessel Design (PVD) problem, taken from (Coello Coello, 2000).

The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = 0.6224 \cdot x_1 \cdot x_3 \cdot x_4 + 1.7781 \cdot x_2 \cdot x_3^2 + 3.1661 \cdot x_1^2 \cdot x_4 + 19.84 \cdot x_1^2 \cdot x_3 \tag{8.22}$$

Subject to:

$$\begin{aligned} g_1(\mathbf{x}) &= -x_1 + 0.0193 \cdot x_3 \le 0 \\ g_2(\mathbf{x}) &= -x_2 + 0.00954 \cdot x_3 \le 0 \\ g_3(\mathbf{x}) &= -\pi \cdot x_3^2 \cdot x_4 - \frac{4}{3} \cdot \pi \cdot x_3^3 + 1296000 \le 0 \\ x_j - 99 \le 0 \quad &; \quad -x_j + 0 \le 0 \quad ; \quad j = 1, 2 \\ x_j - 200 \le 0 \quad &; \quad -x_j + 10 \le 0 \quad ; \quad j = 3, 4 \end{aligned} \tag{8.23}$$

x_1, x_2 are constrained to integer multiples of 0.0625 (mixed-discrete version).

Best known solution (to the best of my knowledge) for the mixed-discrete version:

$$\begin{aligned} \mathbf{x}^* &= (0.812500, 0.437500, 42.098446, 176.636596) \\ f(\mathbf{x}^*) &= 6059.714335 \end{aligned} \tag{8.24}$$

Best solution for the continuous version (obtained by (Auger, et al., 2007)):

$$\begin{aligned} \mathbf{x}^* &= (0.7781686414, 0.3846491626, 40.3196187241, 200.0000000000) \\ f(\mathbf{x}^*) &= 5885.332774 \end{aligned} \tag{8.25}$$

Coello Coello (2000) and Hu, Eberhart, & Shi (2003) include a 4th constraint in their formulations, but it is redundant with the interval constraints. They both also set the lower limit of the first two variables above the actual solution they report. Hence such limit is relaxed here from '1' to '0'.

It appears that the third coefficient in Eq. (8.22) is replaced by some authors by '3.1611', in which case the mixed-discrete optimum is $f(\mathbf{x}^*) = 6059.131296$, as reported in (Hu, Eberhart, & Shi, 2003) (hence they solve a problem different from the formulation they reported). Here the problem solved is as posed in Eq. (8.22). The results are shown in Table 8.13, in Table 8.14 and in Fig. 8.8 for the mixed-discrete version, and in Table 8.15, in Table 8.16, and in Fig. 8.9 for the continuous version of the problem. The results obtained are competitive with the best results in the literature.

Table 8.13. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Mixed Discrete Pressure Vessel Design (MDPVD) problem. The authors included are: (Coello Coello, 2000); (He & Wang, 2007); (Takahama & Sakai, 2006a); (Takahama, Sakai, & Iwane, 2006); (Worasucheep, 2008); and (Wang & Yin, 2008).

MDPVD	OPTIIMUM	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Runs
MS-SN NNB P.AD. PFPR		**6059.714335**	**6059.714335**	**6060.972012**	**6090.526202**	**100.00**	**80.00**	**1.42E+05**	**25**
Coello Coello (GA)		6288.744500	6290.018736	6293.843232	6308.149652	-	-	-	11
He et al. (PSO + SA)		6059.7143	-	6099.9323	6288.6770	-	-	8.10E+04	30
Takahama et al. (εPSO)	**6059.714335**	6059.7143		6136.7744	6410.0868	-	-	5.00E+04	30
Takahama et al. (εDE)		6059.7143	-	6065.8767	6090.5262	-	-	1.00E+04	30
Worasucheep (CPSO-DD - NP=20)		6059.714334	-	6059.714353	6059.714726	-	-	2.00E+05	30
Wang et al. (RSPSO)		6059.7143	-	6066.2032	6100.3196	-	-	3.00E+04	30

Table 8.14. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the MDPVD problem.

Coordinates of best solution found by the 'MS-SN NNB P.AD. PFPR'			
x_1	x_2	x_3	x_4
0.81250000	**0.43750000**	**42.09844560**	**176.63659584**

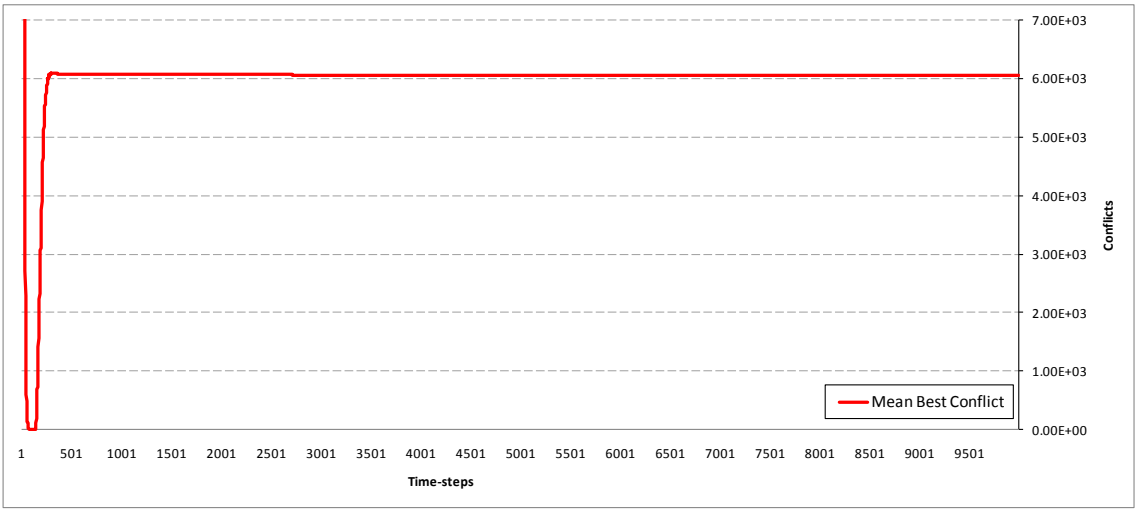


Fig. 8.8. Convergence of the mean best solution for the Mixed-Discrete Pressure Vessel Design (MDPVD) problem.

Table 8.15. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Pressure Vessel Design (PVD) problem. The authors included are: (de Freitas Vaz & da Graça Pinto Fernandes, 2006) and (Foryś & Bochenek, 2004).

PVD	OPTIMUM	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Runs
MS-SN NNB P.AD. PFPR		**5885.333005**	**5890.160720**	**5896.662569**	**5930.479983**	**100.00**	**0.00**	**1.89E+05**	**25**
de Freitas Vaz et al. (global PSO + MO)	**5885.332774**	5885.33	-	-	-	-	-	8.79E+05	-
Forys (PSO with 2-level sociality)		5885.49	-	-	-	-	-	-	-

Table 8.16. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the PVD problem.

Coordinates of best solution found by the 'MS-SN NNB P.AD. PFPR'			
x_1	x_2	x_3	x_4
0.77816878	0.38464923	40.31962573	199.99990247

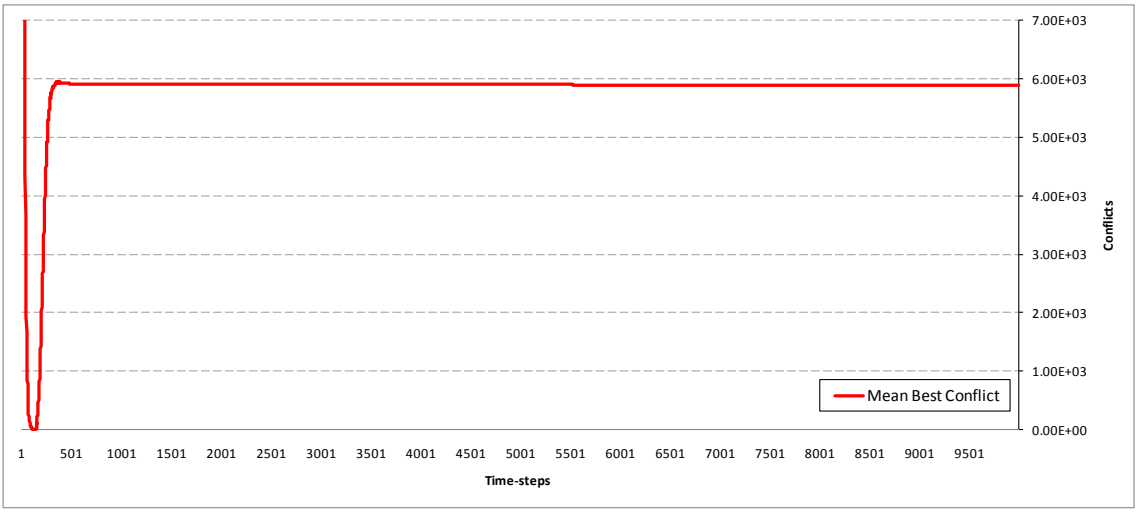


Fig. 8.9. Convergence of the mean best solution for the continuous Pressure Vessel Design (PVD) problem.

8.8.2. Welded Beam Design (WBD)

The problem consists of minimizing the cost of a welded beam subject to constraints on shear stress ($\tau_{\max}$), bending stress in the beam ($\sigma_{\max}$), buckling load on the bar (P_c), end deflection of the beam ($\delta_{\max}$), and side constraints. There are 4 design variables. Refer to Fig. 8.10 for a graphical visualization of the problem.

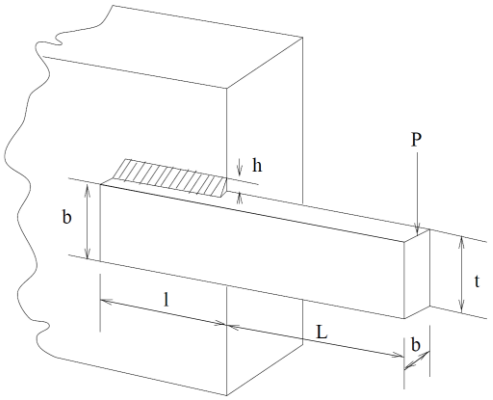


Fig. 8.10. Illustration of the Welded Beam Design (WBD) problem, taken from (Coello Coello, 2000).

The formulation of the problem is as follows:

Minimize:

$$f(\mathbf{x}) = 1.10471 \cdot x_1^{\,2} \cdot x_2 + 0.04811 \cdot x_3 \cdot x_4 \cdot (14 + x_2) \tag{8.26}$$

Subject to:

$$\begin{aligned}
&g_1(\mathbf{x}) = \tau(\mathbf{x}) - \tau_{\max} \le 0 \\
&g_2(\mathbf{x}) = \sigma(\mathbf{x}) - \sigma_{\max} \le 0 \\
&g_3(\mathbf{x}) = x_1 - x_4 \le 0 \\
&g_4(\mathbf{x}) = 0.10471 \cdot x_1^{\,2} + 0.04811 \cdot x_3 \cdot x_4 \cdot (14 + x_2) - 5 \le 0 \\
&g_5(\mathbf{x}) = 0.125 - x_1 \le 0 \\
&g_6(\mathbf{x}) = \delta(\mathbf{x}) - \delta_{\max} \le 0 \\
&g_7(\mathbf{x}) = P - P_c(\mathbf{x}) \le 0
\end{aligned} \tag{8.27}$$

$$\begin{aligned}
&x_j - 2 \le 0 \quad ; \quad -x_j + 0.1 \le 0 \quad ; \quad j = 1, 4 \\
&x_k - 10 \le 0 \quad ; \quad -x_k + 0.1 \le 0 \quad ; \quad k = 2, 3
\end{aligned}$$

where:

$$\begin{aligned}
&P = 6000\, lb \quad ; \quad L = 14\, in \quad ; \quad E = 30 \times 10^6\ psi \quad ; \quad G = 12 \times 10^6\ psi \\
&\tau_{\max} = 13600\ psi \quad ; \quad \sigma_{\max} = 30000\ psi \quad ; \quad \delta_{\max} = 0.25\, in \\
&\tau(\mathbf{x}) = \sqrt{(\tau')^2 + 2 \cdot \tau' \cdot \tau'' \cdot \frac{x_2}{2 \cdot R} + (\tau'')^2} \quad ; \quad \tau' = \frac{P}{\sqrt{2} \cdot x_1 \cdot x_2} \\
&\tau'' = \frac{M \cdot R}{J} \quad ; \quad M = P \cdot \left(L + \frac{x_2}{2} \right) \quad ; \quad R = \sqrt{\frac{x_2^{\,2}}{4} + \left(\frac{x_1 + x_3}{2} \right)^2} \\
&J = 2 \cdot \left\{ \sqrt{2} \cdot x_1 \cdot x_2 \cdot \left[\frac{x_2^{\,2}}{12} + \left(\frac{x_1 + x_3}{2} \right)^2 \right] \right\} \quad ; \quad \sigma(\mathbf{x}) = \frac{6 \cdot P \cdot L}{x_4 \cdot x_3^{\,2}} \\
&\delta(\mathbf{x}) = \frac{4 \cdot P \cdot L^3}{E \cdot x_3^{\,3} \cdot x_4} \quad ; \quad P_c(\mathbf{x}) = \frac{4.013 \cdot E \cdot \sqrt{\frac{x_3^{\,2} \cdot x_4^{\,6}}{36}}}{L^2} \cdot \left(1 - \frac{x_3}{2 \cdot L} \cdot \sqrt{\frac{E}{4 \cdot G}} \right)
\end{aligned} \tag{8.28}$$

Best known solution (to the best of my knowledge):

$$\begin{aligned}
&\mathbf{x}^* = (0.205730, 3.470489, 9.036624, 0.205730) \\
&f(\mathbf{x}^*) = 1.724852
\end{aligned} \tag{8.29}$$

The results are offered in Table 8.17, in Table 8.18, and in Fig. 8.11, where the solution is found in every run. Given that termination conditions other than a maximum number of time-steps are not activated yet, the search extends far beyond necessary.

Table 8.17. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Welded Beam Design (WBD) problem. The authors included are: (Coello Coello, 2000); (de Freitas Vaz & da Graça Pinto Fernandes, 2006); (Foryś & Bochenek, 2004); (He & Wang, 2007); (Takahama & Sakai, 2006a); (Takahama, Sakai, & Iwane, 2006); and (Worasucheep, 2008).

WBD	OPTIIMUM	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Runs
MS-SN NNB P.AD. PFPR	**1.724852**	**1.724852**	**1.724852**	**1.724852**	**1.724852**	**100.00**	**100.00**	**4.08E+05**	**25**
Coello Coello (GA)		1.748309	1.773586	1.771973	1.785835	-	-	-	11
de Freitas Vaz et al. (global PSO + MO)		1.814290	-	-	-	-	-	9.60E+05	-
Forys (PSO with 2-level sociality)		1.724800	-	-	-	-	-	-	-
He et al. (PSO + SA)		1.724852	-	1.749040	1.814295	-	-	8.10E+04	30
Takahama et al. (εPSO)		1.724900	-	1.725200	1.735800	-	-	5.00E+04	30
Takahama et al. (εDE)		1.724900	1.724900	1.724900	1.724900	-	-	1.00E+04	30
Worasucheep (CPSO-DD - NP=20)		1.724852	1.724852	1.724852	1.724852	-	-	2.00E+05	30

Table 8.18. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the WBD problem.

Coordinates of best solution found by the 'MS-SN NNB P.AD. PFPR'			
x_1	x_2	x_3	x_4
0.20572964	**3.47048867**	**9.03662391**	**0.20572964**

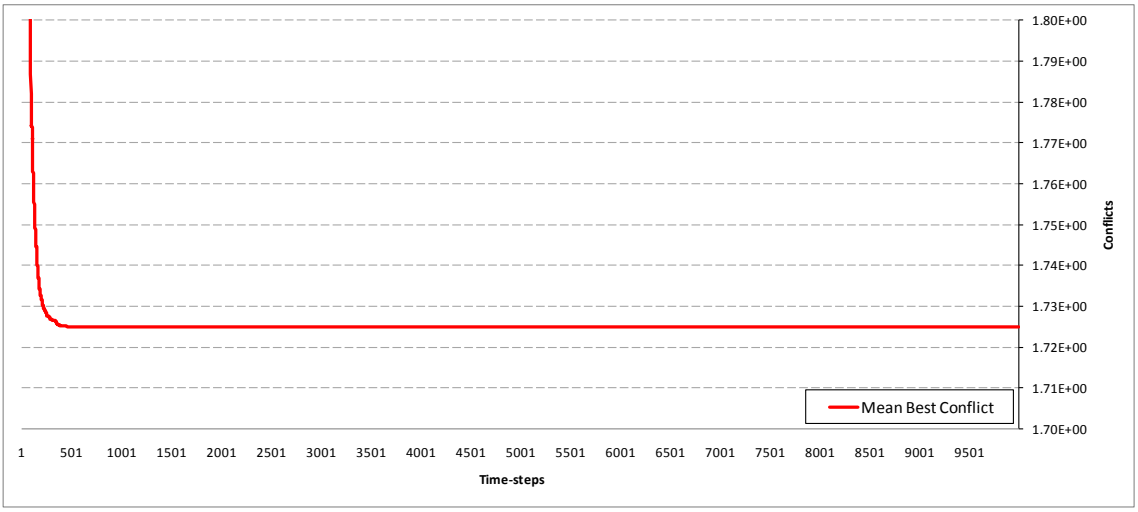


Fig. 8.11. Convergence of the mean best solution for the Welded Beam Design (WBD) problem.

8.8.3. Tension/Compression Spring Design (TCSD)

The problem consists of minimizing the weight of a tension/compression spring subject to constraints on minimum deflection, shear stress, surge frequency, restrictions on the outside diameter, and side constraints. There are 3 design variables: x_1 is the wire diameter, x_2 is the mean coil diameter, and x_3 is the number of active coils, which is not restricted to integer values in their formulation (see Fig. 8.12).

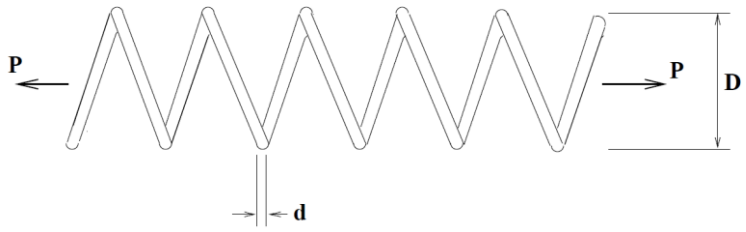


Fig. 8.12. Illustration of the Tension/Compression Spring Design problem (TCSD), taken from (Coello Coello, 2000).

The problem is formulated as follows:

Minimize:

$$f(\mathbf{x}) = x_1^{\,2} \cdot x_2 \cdot (x_3 + 2) \tag{8.30}$$

Subject to:

$$\begin{aligned}
g_1(\mathbf{x}) &= 1 - \frac{x_2^{\,3} \cdot x_3}{71785 \cdot x_1^{\,4}} \le 0 \\
g_2(\mathbf{x}) &= \frac{4 \cdot x_2^{\,2} - x_1 \cdot x_2}{12566 \cdot \left(x_1^{\,3} \cdot x_2 - x_1^{\,4}\right)} + \frac{1}{5108 \cdot x_1^{\,2}} - 1 \le 0 \\
g_3(\mathbf{x}) &= 1 - \frac{140.45 \cdot x_1}{x_2^{\,2} \cdot x_3} \le 0 \\
g_4(\mathbf{x}) &= \frac{x_1 + x_2}{1.5} - 1 \le 0
\end{aligned} \tag{8.31}$$

$$\begin{aligned}
x_1 - 2 \le 0 \quad ; \quad -x_1 + 0.05 \le 0 \\
x_2 - 1.3 \le 0 \quad ; \quad -x_2 + 0.25 \le 0 \\
x_3 - 15 \le 0 \quad ; \quad -x_3 + 2 \le 0
\end{aligned}$$

Best known solution (to the best of my knowledge):

$$\mathbf{x}^* = (0.051755, 0.358316, 11.195851)$$
$$f(\mathbf{x}^*) = 0.012665 \tag{8.32}$$

The results are offered in Table 8.19, in Table 8.20, and in Fig. 8.13, where the proposed approach finds the best results. Notice that there is a much smaller number of FEs in (He & Wang, 2007). As before, there are many more FEs than needed here, where the aim is to keep the settings unchanged.

Table 8.19. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Tension/Compression Spring Design (TCSD) problem. The authors included are: (Coello Coello, 2000); (de Freitas Vaz & da Graça Pinto Fernandes, 2006); and (He & Wang, 2007).

TCSD	OPTIIMUM	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Runs
MS-SN NNB P.AD. PFPR	**0.012665**	**0.012665**	**0.012667**	**0.012670**	**0.012686**	**100.00**	**100.00**	**1.55E+05**	**25**
Coello Coello (GA)		0.012705	0.012756	0.012769	0.012822	-	-	-	11
de Freitas Vaz et al. (global PSO + MO)		0.013193	-	-	-	-	-	7.58E+05	-
He et al. (PSO + SA)		0.012665	-	0.012707	0.012719			8.10E+04	30

Table 8.20. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the TCSD problem.

Coordinates of best solution found by the 'MS-SN NNB P.AD. PFPR'		
x_1	x_2	x_3
0.05168288	**0.35656916**	**11.29768179**

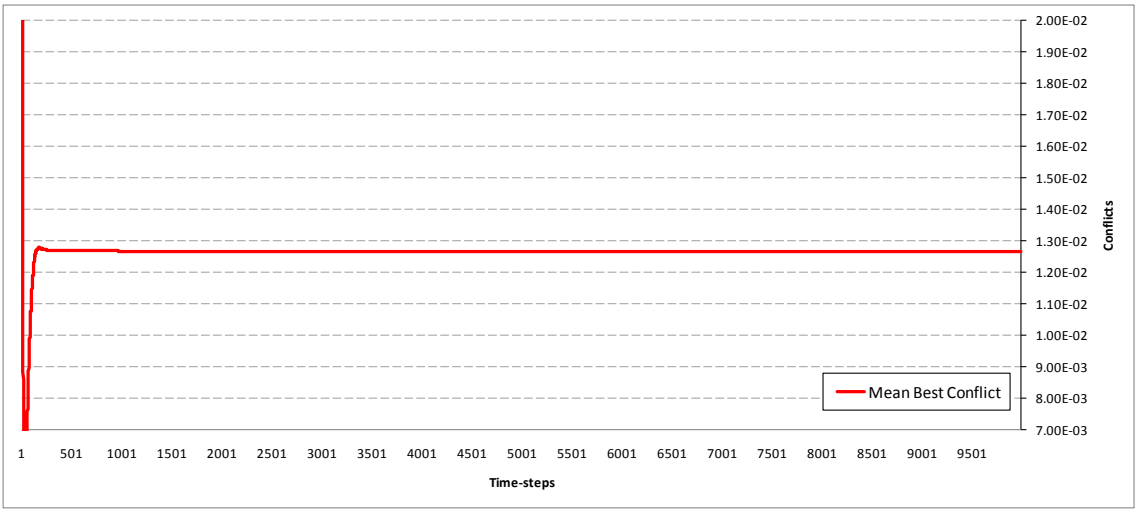


Fig. 8.13. Convergence of the mean best solution for the Tension/Compression Spring Design (TCSD) problem.

8.8.4. Himmelblau's Nonlinear Problem (HNP)

The problem consists of 5 variables subject to 3 inequality constraints in the form of intervals, and also subject to interval constraints. The problem is formulated as follows:

Minimize:

$$f(\mathbf{x}) = 5.3578547 \cdot x_3^{\,2} + 0.8356891 \cdot x_1 \cdot x_5 + 37.2932239 \cdot x_1 - 40792.141 \tag{8.33}$$

Subject to:

$$\begin{aligned}
&g_1(\mathbf{x}) = 85.334407 + 0.0056858 \cdot x_2 \cdot x_5 + 0.00026 \cdot x_1 \cdot x_4 - 0.0022053 \cdot x_3 \cdot x_5 \\
&g_2(\mathbf{x}) = 80.51249 + 0.0071317 \cdot x_2 \cdot x_5 + 0.0029955 \cdot x_1 \cdot x_2 + 0.0021813 \cdot x_3^{\,2} \\
&g_3(\mathbf{x}) = 9.300961 + 0.0047026 \cdot x_3 \cdot x_5 + 0.0012547 \cdot x_1 \cdot x_3 + 0.0019085 \cdot x_3 \cdot x_4 \\
&0 \le g_1(\mathbf{x}) \le 92 \\
&90 \le g_2(\mathbf{x}) \le 110 \\
&20 \le g_3(\mathbf{x}) \le 25
\end{aligned} \tag{8.34}$$

$$\begin{aligned}
&x_1 - 102 \le 0 \quad ; \quad -x_1 + 78 \le 0 \\
&x_2 - 45 \le 0 \quad ; \quad -x_2 + 33 \le 0 \\
&x_j - 45 \le 0 \quad ; \quad -x_j + 27 \le 0 \quad ; \quad j = 3,4,5
\end{aligned}$$

Best known solution (to the best of my knowledge):

$$\begin{aligned}
&\mathbf{x}^* = (78.000000, 33.000000, 27.070997, 45.000000, 44.969242) \\
&f(\mathbf{x}^*) = -31025.561420
\end{aligned} \tag{8.35}$$

Differences can be found in the literature in some formulations. For instance, the 3rd term in the conflict function is formulated in (Coello Coello, 2000) as $37.29329 \cdot x_1$, while other authors formulate it as $37.293239 \cdot x_1$ (e.g. problem g04). Likewise, the third term in $g_1(\mathbf{x})$ can also be found in the literature as $0.0006262 \cdot x_1 \cdot x_4$ (e.g. g04). Here the formulation is as in (Hu, Eberhart, & Shi, 2003). This problem is included here despite not being an engineering problem because it is in the test suites both in (Coello Coello, 2000) and in (Hu, Eberhart, & Shi, 2003).

The results are shown in Table 8.21, in Table 8.22, and in Fig. 8.14.

Table 8.21. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood NNB Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the Himmelblau's Nonlinear Problem (HNP). The authors included are: (Coello Coello, 2000); (de Freitas Vaz & da Graça Pinto Fernandes, 2006); (Takahama & Sakai, 2006a); (Takahama, Sakai, & Iwane, 2006); and (Worasucheep, 2008).

HNLP	OPTIIMUM	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Runs
MS-SN NNB P.AD. PFPR	**-31025.561420**	**-31025.561420**	**-31025.561420**	**-31025.561420**	**-31025.561420**	**100.00**	**100.00**	**2.80E+05**	**25**
Coello Coello (GA)		-31020.859000	-31017.213691	-30984.240703	-30792.407738	-	-	-	11
de Freitas Vaz et al. (global PSO + MO)		-31012.100000	-	-	-	-	-	7.84E+05	-
Takahama et al. (εPSO)		-31025.559900	-	-31025.543200	-31025.467400	-	-	5.00E+04	30
Takahama et al. (εDE)		-31025.560100	-	-31025.557900	-31025.549000	-	-	1.00E+04	30
Worasucheep (CPSO-DD - NP=20)		'-31026.647264 (*)	-	-31002.170814	-30994.129379	-	-	2.00E+05	30

Table 8.22. Coordinates of the best solution found by the 'MS-SN NNB P.AD. PFPR' PSO for the HNP.

Coordinates of best solution found by the 'MS-SN NNB P.AD. PFPR'				
x_1	x_2	x_3	x_4	x_5
78.00000000	**33.00000000**	**27.07099711**	**45.00000000**	**44.96924255**

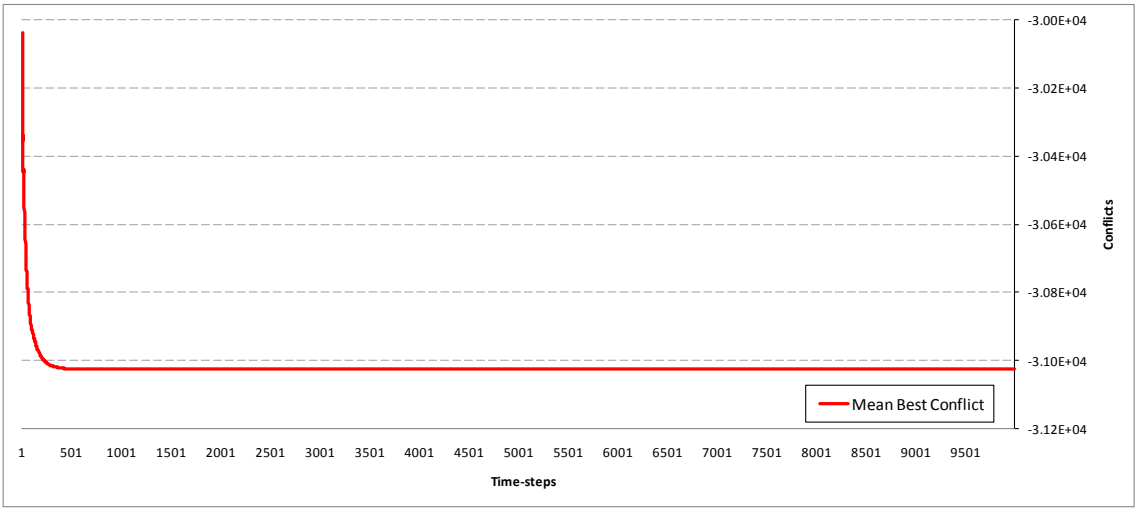


Fig. 8.14. Convergence of the mean best solution for the Tension/Compression Spring Design (TCSD) problem.

8.8.5. 10-Bar Plane Truss

This is a classical engineering problem used to test optimization algorithms. However, there are different versions in the literature, some of them differing in minor details that may go unnoticed. Furthermore, numerical comparisons between results obtained from different formulations are not infrequent. The formulations of the problem implemented

here are taken from (Fleury & Schmit, 1980) and from (Haftka & Gürdal, 1992), and are expressed in SI units. The analysis problem is as shown in Fig. 8.15 and Eq. (8.36).

$$\begin{cases} \text{Aluminium} \\ \text{E} = 68.95 \text{ x } 10^6 \text{ kN/m}^2 \\ \rho = 2768 \text{ kg/m}^3 \\ \sigma_a = \pm 172375 \text{ kN/m}^2 \quad \text{(except for member 9)} \\ \sigma_a^{(9)} = 3 \cdot \left(\pm 172375 \text{ kN/m}^2 \right) \\ A_{\min} = x_{\min} = 0.6452 \text{ cm}^2 \\ P = \begin{pmatrix} 0 & -444800\text{N} & 0 \end{pmatrix} \end{cases} \tag{8.36}$$

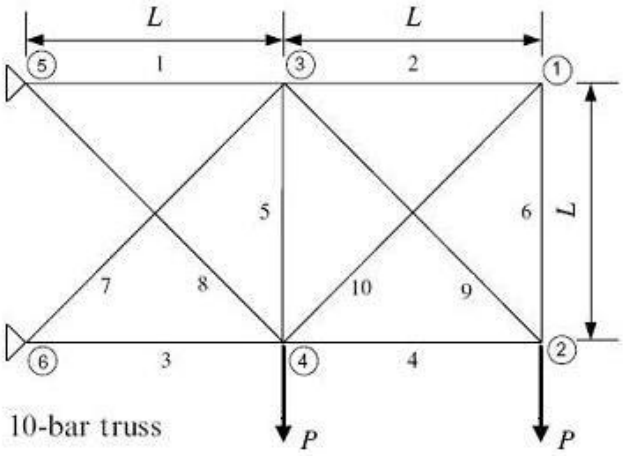


Fig. 8.15. 10-Bar Truss problem, where the objective is to minimize the mass while complying with constraints.

The objective of the optimization problem is to minimize the mass of the structure, subject to constraints. Hence the variables are given by the cross sections of the members. Different optimization problems can be posed according to the constraints imposed.

8.8.5.1. Stress Constraints

The problem is then formulated as follows:

Minimize:

$$f(\mathbf{x}) = \sum_{i=1}^{10} \rho \cdot x_i \cdot l_i \tag{8.37}$$

Subject to:

$$g_i(\mathbf{x}) = \max\left[\left(\frac{\sigma_i}{\sigma_{a+}^{(i)}} - 1\right), \left(\frac{\sigma_i}{\sigma_{a-}^{(i)}} - 1\right)\right] \leq 0 \quad ; \quad i = 1,...,10$$
$$x_i - 200 \leq 0 \quad ; \quad -x_i + 0.6452 \leq 0 \quad ; \quad i = 1,..,10 \tag{8.38}$$

where l_i is the length of member 'i' and $\sigma_a^{(i)}$ its allowable stress. The upper bound of the variables (200) is arbitrarily set here to a high value.

The results are shown in Table 8.23, in Table 8.24, and in Fig. 8.16, where those obtained by other authors are also provided for reference. Other settings delaying convergence –e.g. PSO-RRR2-1 with 100 particles and 5000 time-steps (i.e. approximately the same FEs)– lead to better results. However the same settings as before are maintained for consistency (except that the NNB was deactivated for the truss experiments).

Table 8.23. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the 10-Bar Truss problem with stress constraints. The authors included are: (Haftka & Gürdal, 1992); (Burton, 2004) and (Ghasemi, Hinton, & Wood, 1999).

10-Bar Truss (Stress)	OPTIIMUM (SI)	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Runs
MS-SN P.AD. PFPR	**679.299379**	**684.687377**	**706.201241**	**706.782877**	**744.425308**	**100.00**	**0.00**	**2.29E+05**	**25**
(Haftka & Gürdal, 1992) (Linear)		679.934408	-	-	-	-	-	-	-
(Haftka & Gürdal, 1992) (Reciprocal)		684.016736	-	-	-	-	-	-	-
(Haftka & Gürdal, 1992) (Conservative)		694.902944	-	-	-	-	-	-	-
(Haftka & Gürdal, 1992) (Quadratic)		679.934408	-	-	-	-	-	-	-
(Haftka & Gürdal, 1992) (Recip-quadratic)		679.934408	-	-	-	-	-	-	-
(Haftka & Gürdal, 1992) (Linear force)		679.934408	-	-	-	-	-	-	-
(Burton, 2004) (GA re-birth)		692.634984	-	-	-	-	-	-	-
(Ghasemi, Hinton, & Wood, 1999) (GA)		696.263720	-	-	-	-	-	4.00E+04	-
(Ghasemi, Hinton, & Wood, 1999) (GA re-birth)		687.645472	-	-	-	-	-	4.00E+04	-

Table 8.24. Coordinates of the best solution found by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by different authors and different approaches in the literature for the 10-Bar Truss problem with stress constraints. The authors included are: (Haftka & Gürdal, 1992); (Burton, 2004) and (Ghasemi, Hinton, & Wood, 1999).

	Coordinates of best solution									
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
OPTIMUM (SI)	**50.967640**	**0.645160**	**52.257960**	**25.161240**	**0.645160**	**0.645160**	**37.419280**	**35.548316**	**23.741888**	**0.903224**
MS-SN P.AD. PFPR	**49.673107**	**1.935305**	**53.543717**	**23.868901**	**0.645200**	**1.935305**	**39.229592**	**33.755724**	**22.503816**	**2.736934**
(Burton, 2004) (GA re-birth)	47.980549	3.703218	22.174149	55.328922	0.651612	3.716122	41.735400	31.309615	20.838668	5.245151
(Ghasemi, Hinton, & Wood, 1999) (GA re-birth)	48.503129	2.954833	54.386988	22.864470	0.645160	2.967736	40.561209	32.206387	21.612860	4.161282

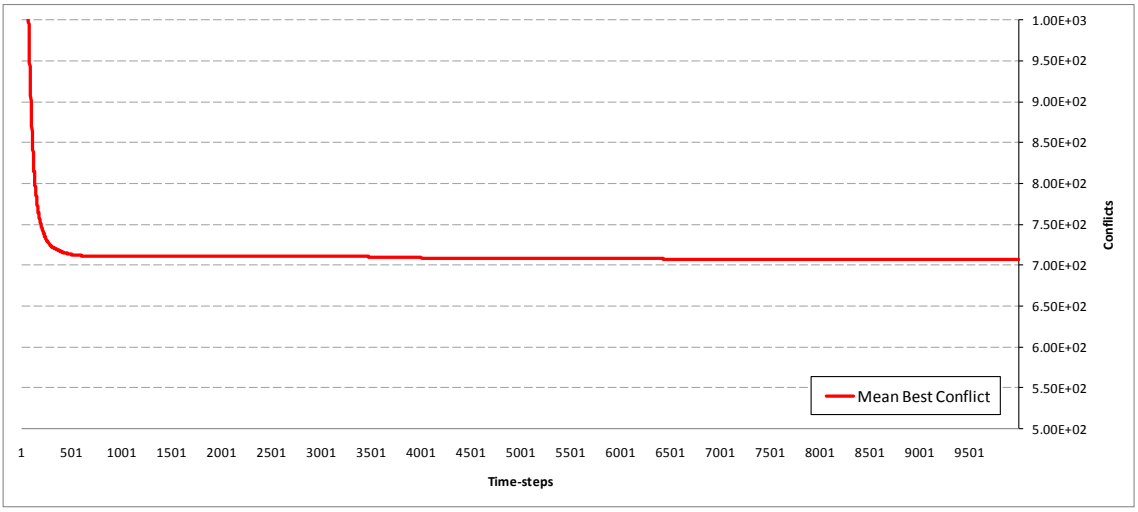


Fig. 8.16. Convergence of the mean best solution for the 10-Bar Truss problem with stress constraints only.

The optimum solution in Table 8.23 is the one reported in (Haftka & Gürdal, 1992, p. 244), where the US customary units are turned into the SI units considering the mass equivalences as 1 lb = 0.453592 kg.

Also note that the lower limit for the variables were set in (Haftka & Gürdal, 1992) to 0.01 inches (equal to 0.64516 cm) whereas the problem was solved here in the SI system as posed in (Fleury & Schmit, 1980), where the lower limit was set to 0.6452 cm (see x_5 in Table 8.24). In addition, the variables for the PSO search are in cm^2 while the Finite Element (FE) analysis is in m^2. These changes of units produce round-off errors.

As stated by Haftka and Gürdal (1992), the Optimal Design and the Fully Stressed Design (FSD) differ in this problem (see (Haftka & Gürdal, 1992, p. 244)). Thus, although the objective function is linear, the constraints seem to generate local attractors where they are active. The details of the cross sections (areas) and stresses are offered in Table 8.25 for the optimal design of the 10-Bar Truss with Stress Constraints from (Haftka & Gürdal, 1992) (mass = 679.299379 kg); and in Table 8.26 for the best design found by the MS-SN P.AD. PFPR PSO (mass = 684.687377 kg). Active or nearly active constraints are shown in bold, red font. As it can be observed, the stresses on each bar are very similar despite the noticeable difference in the designs. This is even more noticeable when observing the FSD in (Haftka & Gürdal, 1992, p. 244). The results of the optimal design by the PSO-RRR2-1 P.AD. PFPR (100-5000) and those of the FSD are shown in Table 8.27, in Table 8.28, and in Fig. 8.24.

Table 8.25. Details of the FE analysis of the optimal design in (Haftka & Gürdal, 1992) of the 10-Bar Truss problem with stress constraints.

Optimal Design of the 10-Bar Truss with Stress Constraints (mass = **679.299379** kg) from (Haftka & Gürdal, 1992)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	50.9676	0.6452	7899.50	1.72E+08	-1.72E+08	1.72E+08	99.99
2	2	0.6452	0.6452	100.00	1.72E+08	-1.72E+08	1.72E+08	99.50
3	3	52.2580	0.6452	8099.50	-1.72E+08	-1.72E+08	1.72E+08	99.99
4	4	25.1612	0.6452	3899.75	-1.72E+08	-1.72E+08	1.72E+08	100.00
5	5	0.6452	0.6452	100.00	-4.86E+05	-1.72E+08	1.72E+08	0.28
6	6	0.6452	0.6452	100.00	1.72E+08	-1.72E+08	1.72E+08	99.50
7	7	37.4193	0.6452	5799.64	1.72E+08	-1.72E+08	1.72E+08	99.96
8	8	35.5483	0.6452	5509.66	-1.73E+08	-1.72E+08	1.72E+08	100.10
9	9	23.7419	0.6452	3679.77	2.58E+08	-5.17E+08	5.17E+08	49.96
10	10	0.9032	0.6452	139.99	-1.73E+08	-1.72E+08	1.72E+08	100.52

Table 8.26. Details of the FE analysis of the best design of the 10-Bar Truss problem with stress constraints found by the 'MS-SN P.AD. PFPR' PSO.

Best Design of the 10-Bar Truss with Stress Constraints (mass = **684.687377** kg) by the **MS-SN P.AD. PFPR PSO**								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	49.6731	0.6452	7698.87	1.72E+08	-1.72E+08	1.72E+08	100.00
2	2	1.9353	0.6452	299.95	1.72E+08	-1.72E+08	1.72E+08	100.00
3	3	53.5437	0.6452	8298.78	-1.72E+08	-1.72E+08	1.72E+08	100.00
4	4	23.8689	0.6452	3699.46	-1.72E+08	-1.72E+08	1.72E+08	100.00
5	5	0.6452	0.6452	100.00	-1.00E-07	-1.72E+08	1.72E+08	0.00
6	6	1.9353	0.6452	299.95	1.72E+08	-1.72E+08	1.72E+08	100.00
7	7	39.2296	0.6452	6080.22	1.72E+08	-1.72E+08	1.72E+08	100.00
8	8	33.7557	0.6452	5231.82	-1.72E+08	-1.72E+08	1.72E+08	100.00
9	9	22.5038	0.6452	3487.88	2.59E+08	-5.17E+08	5.17E+08	50.00
10	10	2.7369	0.6452	424.19	-1.72E+08	-1.72E+08	1.72E+08	100.00

Table 8.27. Statistical results obtained by the 'PSO-RRR2-1 Pseudo Adaptive PFPR' and by the Fully Stressed Design (FSD) in (Haftka & Gürdal, 1992) for the 10-Bar Truss problem with stress constraints.

10-Bar Truss (Stress)	**OPTIIMUM (SI)**	**BEST**	**MEDIAN**	**MEAN**	**WORST**	**[%] Feasible Solutions**	**[%] Successful Solutions**	**Mean FEs**	**Runs**
PSO-RRR2-1 P.AD. PFPR (100-5000)	**679.299379**	**682.145601**	**699.785190**	**698.332731**	**709.496934**	**100.00**	**0.00**	**6.08E+04**	**25**
(Haftka & Gürdal, 1992) (FSD)		782.536918	-	-	-	-	-	-	-

Table 8.28. Coordinates of the best solution found by the 'PSO-RRR2-1 Pseudo Adaptive PFPR' and by the Fully Stressed Design (FSD) in (Haftka & Gürdal, 1992) for the 10-Bar Truss problem with stress constraints.

	Coordinates of best solution									
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
OPTIMUM (SI)	**50.967640**	**0.645160**	**52.257960**	**25.161240**	**0.645160**	**0.645160**	**37.419280**	**35.548316**	**23.741888**	**0.903224**
PSO-RRR2-1 P.AD. PFPR (100-5000)	**50.275648**	**1.332764**	**52.941176**	**24.471442**	**0.645200**	**1.332764**	**38.377471**	**34.607845**	**23.071896**	**1.884813**
(Haftka & Gürdal, 1992) (FSD)	26.516076	25.096724	76.709524	0.709676	0.645160	25.096724	71.999856	0.967740	0.645160	35.548316

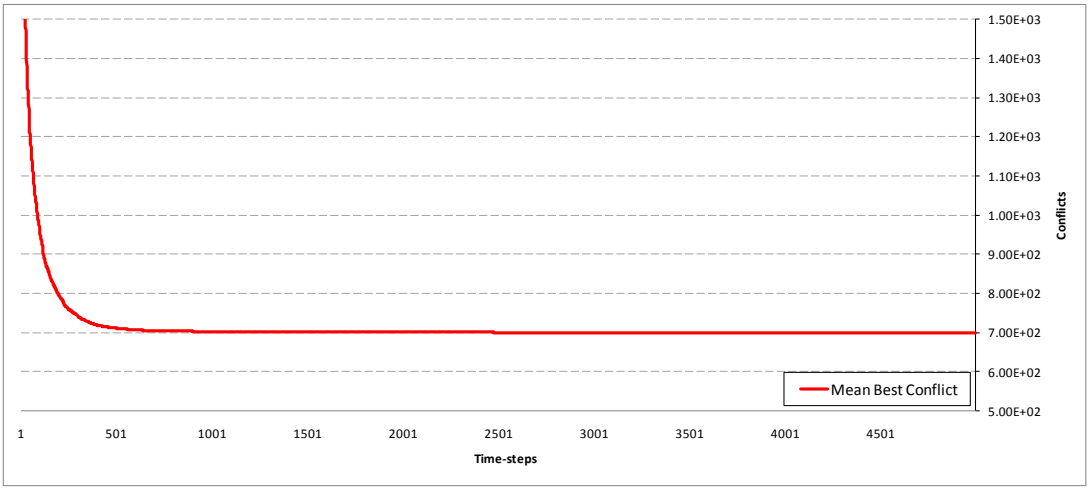


Fig. 8.17. Convergence of the mean best solution for the 10-Bar Truss problem with stress constraints only found by the 'PSO-RRR2-1 P.AD. PFPR (100-5000)'.

The analysis details are shown in Table 8.29 for the optimal design by the PSO-RRR2-1 P.AD. PFPR, and in Table 8.30 for the FSD provided in (Haftka & Gürdal, 1992).

Table 8.29. Details of the FE analysis of the best design of the 10-Bar Truss problem with stress constraints found by the 'PSO-RRR2-1 (100-5000) P.AD. PFPR'.

Best Design of the 10-Bar Truss with Stress Constraints (mass = **682.145601** kg) by the **PSO-RRR2-1 (100-5000) P.AD. PFPR PSO**								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	50.2756	0.6452	7792.26	1.72E+08	-1.72E+08	1.72E+08	**100.00**
2	2	1.3328	0.6452	206.57	1.72E+08	-1.72E+08	1.72E+08	**100.00**
3	3	52.9412	0.6452	8205.39	-1.72E+08	-1.72E+08	1.72E+08	**100.00**
4	4	24.4714	0.6452	3792.85	-1.72E+08	-1.72E+08	1.72E+08	**100.00**
5	5	0.6452	0.6452	**100.00**	1.50E-01	-1.72E+08	1.72E+08	0.00
6	6	1.3328	0.6452	206.57	1.72E+08	-1.72E+08	1.72E+08	**100.00**
7	7	38.3775	0.6452	5948.15	1.72E+08	-1.72E+08	1.72E+08	**100.00**
8	8	34.6078	0.6452	5363.89	-1.72E+08	-1.72E+08	1.72E+08	**100.00**
9	9	23.0719	0.6452	3575.93	2.59E+08	-5.17E+08	5.17E+08	50.00
10	10	1.8848	0.6452	292.13	-1.72E+08	-1.72E+08	1.72E+08	**100.00**

Table 8.30. Details of the FE analysis of the fully stressed design in (Haftka & Gürdal, 1992) of the 10-Bar Truss problem with stress constraints.

Fully Stressed Design (FSD) of the 10-Bar Truss with Stress Constraints (mass = **782.536918** kg) from (Haftka & Gürdal, 1992)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	26.5161	0.6452	4109.75	1.72E+08	-1.72E+08	1.72E+08	**99.90**
2	2	25.0967	0.6452	3889.76	1.72E+08	-1.72E+08	1.72E+08	**100.07**
3	3	76.7095	0.6452	11889.26	-1.72E+08	-1.72E+08	1.72E+08	**100.02**
4	4	0.7097	0.6452	109.99	-1.67E+08	-1.72E+08	1.72E+08	**97.17**
5	5	0.6452	0.6452	**99.99**	-8.29E+05	-1.72E+08	1.72E+08	0.48
6	6	25.0967	0.6452	3889.76	1.72E+08	-1.72E+08	1.72E+08	**100.07**
7	7	71.9999	0.6452	11159.31	1.72E+08	-1.72E+08	1.72E+08	**100.02**
8	8	0.9677	0.6452	149.99	-1.73E+08	-1.72E+08	1.72E+08	**100.32**
9	9	0.6452	0.6452	99.99	2.61E+08	-5.17E+08	5.17E+08	50.39
10	10	35.5483	0.6452	5509.66	-1.72E+08	-1.72E+08	1.72E+08	**99.91**

As shown in Table 8.25, Table 8.26, Table 8.29, and Table 8.30, noticeably different designs may lead to the bars in the truss being subjected to very similar stresses.

Comparing to other authors' results, those obtained by the MS-SN P.AD. PFPR are competitive.

8.8.5.2. Stress and Displacements Constraints Case 1

This is the same problem as in section 8.8.5.1 but with two additional constraints:

1) y-displacement of node '1' equals $u_{y1} = -0.0508$ m
2) y-displacement of node '3' equals $u_{y3} = -0.0254$ m

Therefore Eq. (8.38) is replaced by Eq. (8.39). Beware that the displacements are directly called from the analysis (in [m]) whereas the limits to the variables are used within the PSO search (in [cm^2]).

$$\begin{aligned} & g_i(\mathbf{x}) = \max\left[\left(\frac{\sigma_i}{\sigma_{a+}^{(i)}} - 1\right), \left(\frac{\sigma_i}{\sigma_{a-}^{(i)}} - 1\right)\right] \le 0 \quad ; \quad i = 1,...,10 \\ & g_{11}(\mathbf{x}) = \frac{u_{y1}}{0.0508} + 1 = 0 \\ & g_{12}(\mathbf{x}) = \frac{u_{y3}}{0.0254} + 1 = 0 \\ & x_i - 200 \le 0 \quad ; \quad -x_i + 0.6452 \le 0 \quad ; \quad i = 1,...,10 \end{aligned} \tag{8.39}$$

Therefore there are now 10 interval constraints, 10 inequality constraints, and 2 equality constraints. Notice that there is a mistake in the formulation of the displacement limits for node 3 in (Haftka & Gürdal, 1992). The results are shown in Table 8.31, in Table 8.32, and in Fig. 8.18. The solution obtained in (Fleury & Schmit, 1980) is also reported for reference.

Recall that the equality constraints in Eq. (8.39) are replaced by the inequality constraints in Eq. (8.40) in order to tackle the problem with a PSO optimizer. Hence solutions marginally below the actual optimal are possible. Refer to Table 8.31, where the optimal solution reported for this problem is as in (Fleury & Schmit, 1980).

$$g_{11}(\mathbf{x}) = \text{abs}\left(\frac{u_{y1}}{0.0508} + 1\right) \leq Tol_{eq} = 10^{-4}$$
$$g_{12}(\mathbf{x}) = \text{abs}\left(\frac{u_{y3}}{0.0254} + 1\right) \leq Tol_{eq} = 10^{-4} \quad (8.40)$$

Table 8.31. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by (Fleury & Schmit, 1980) for the 10-Bar Truss problem with stress and two equality constraints (*y*-displacements of nodes '1' and '3').

10-Bar Truss (Stress & Displacement 1)	OPTIIMUM (SI)	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Runs
MS-SN P.AD. PFPR	1836.610000	1836.311164	1838.259962	1838.427791	1841.668990	100.00	4.00	1.02E+05	25
(Fleury & Schmit, 1980)		1836.610000	-	-	-	-	-	-	-

Table 8.32. Coordinates of the best solutions found by the 'MS-SN P.AD. PFPR' and by Fleury and Schmit (1980) for the 10-Bar Truss problem with stress and two equality constraints (*y*-displacements of nodes '1' and '3').

	Coordinates of best solution									
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
MS-SN P.AD. PFPR	145.291157	8.964421	139.789818	54.594323	0.645286	0.645200	81.766020	94.085914	76.859520	12.864237
(Fleury & Schmit, 1980)	146.2000	9.0390	139.2000	54.4200	0.6452	0.6452	81.8800	93.8100	76.9700	12.7900

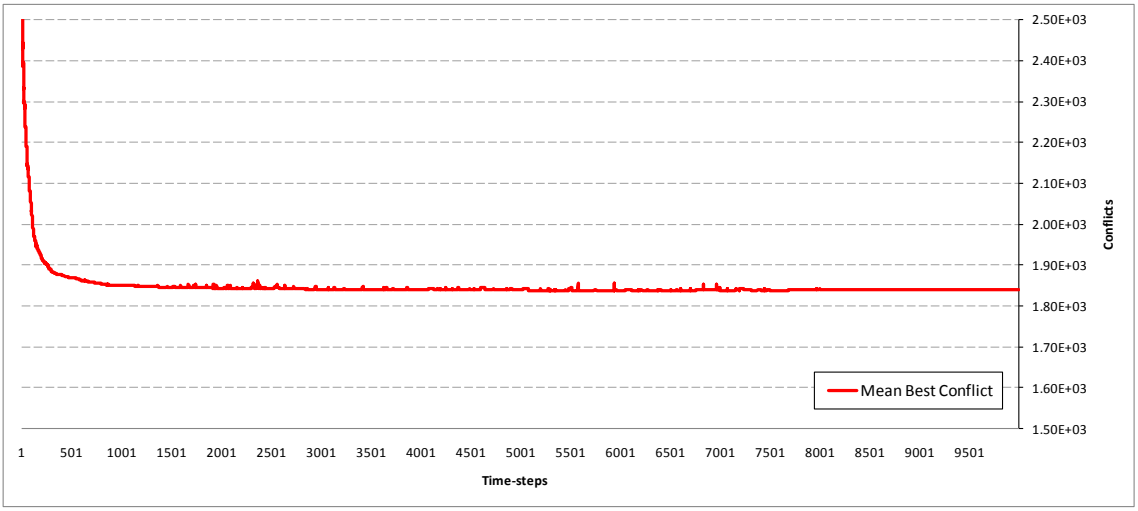


Fig. 8.18. Convergence of the mean best solution for 10-Bar Truss problem with stress and 2 equality constraints (*y*-displacements of nodes '1' and '3').

The details of the of the FE analysis of the optimal design for the 10-Bar Truss problem with Stress and Displacement Constraints (Case 1) from (Fleury & Schmit, 1980)

(where mass = 1836.61 kg) are offered in Table 8.33 and in Table 8.34; whereas those of the best design found by the MS-SN P.AD. PFPR PSO (mass = 1836.311164 kg) are offered in Table 8.35 and in Table 8.36.

Table 8.33. Cross sections (areas) and stresses details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) of the 10-Bar Truss problem with stress and displacement constraints (Case 1).

(Fleury & Schmit, 1980) - 10-Bar Truss with Stress & Displacement Constraints 1 (mass = **1836.61** kg)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	146.2000	0.6452	22659.64	6.02E+07	-1.72E+08	1.72E+08	34.94
2	2	9.0390	0.6452	1400.96	1.23E+07	-1.72E+08	1.72E+08	7.14
3	3	139.2000	0.6452	21574.71	-6.46E+07	-1.72E+08	1.72E+08	37.46
4	4	54.4200	0.6452	8434.59	-7.97E+07	-1.72E+08	1.72E+08	46.23
5	5	0.6452	0.6452	**100.00**	2.98E+07	-1.72E+08	1.72E+08	17.31
6	6	0.6452	0.6452	100.00	1.72E+08	-1.72E+08	1.72E+08	**99.97**
7	7	81.8800	0.6452	12690.64	7.84E+07	-1.72E+08	1.72E+08	45.49
8	8	93.8100	0.6452	14539.68	-6.57E+07	-1.72E+08	1.72E+08	38.10
9	9	76.9700	0.6452	11929.63	7.97E+07	-5.17E+08	5.17E+08	15.41
10	10	12.7900	0.6452	1982.33	-1.23E+07	-1.72E+08	1.72E+08	7.13

Table 8.34. Displacement details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) of the 10-Bar Truss problem with stress and displacement constraints (Case 1).

(Fleury & Schmit, 1980) - 10-Bar Truss with Stress & Displacement Constraints 1					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.9617	-	-	-
	uy	-5.0802	-5.0800	-5.0800	**100.00**
	uz	0.0000	-	-	-
2	ux	-1.9131	-	-	-
	uy	-7.3656	-	-	-
	uz	0.0000	-	-	-
3	ux	0.7986	-	-	-
	uy	-2.5404	-2.5400	-2.5400	**100.02**
	uz	0.0000	-	-	-
4	ux	-0.8563	-	-	-
	uy	-2.9361	-	-	-
	uz	0.0000	-	-	-
5	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
6	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

Note that the *y*-displacements of nodes '1' and '3' in Table 8.36 are marginally violating the equality constraints in the orginal formulation due to the tolerance set in Eq. (8.40). The reason why the *y*-displacement of node '3' in Table 8.34 is marginally

violating the equality constraint is the round-offs of the values of the areas input into the truss analysis. In fact, $u_y = -2.5400$ cm for node '3' in (Fleury & Schmit, 1980, p. 135).

Table 8.35. Cross sections (areas) and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 10-Bar Truss problem with stress and displacement constraints (Case 1).

MS-SN P.AD. PFPR - 10-Bar Truss with Stress & Displacement Constraints 1 (mass = **1836.311164** kg)								
Variables	Bars	Areas [cm²]	Min. areas [cm²]	[%] min. areas	Stresses [N/m²]	Allowable Stresses [N/m²]		[%] max. stress
1	1	145.2912	0.6452	22518.78	6.06E+07	-1.72E+08	1.72E+08	35.15
2	2	8.9644	0.6452	1389.40	1.24E+07	-1.72E+08	1.72E+08	7.20
3	3	139.7898	0.6452	21666.12	-6.43E+07	-1.72E+08	1.72E+08	37.30
4	4	54.5943	0.6452	8461.61	-7.94E+07	-1.72E+08	1.72E+08	46.08
5	5	0.6453	0.6452	**100.01**	2.98E+07	-1.72E+08	1.72E+08	17.29
6	6	0.6452	0.6452	100.00	1.72E+08	-1.72E+08	1.72E+08	**100.00**
7	7	81.7660	0.6452	12672.97	7.85E+07	-1.72E+08	1.72E+08	45.55
8	8	94.0859	0.6452	14582.44	-6.55E+07	-1.72E+08	1.72E+08	37.98
9	9	76.8595	0.6452	11912.51	7.98E+07	-5.17E+08	5.17E+08	15.43
10	10	12.8642	0.6452	1993.84	-1.22E+07	-1.72E+08	1.72E+08	7.09

Table 8.36. Displacement details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 10-Bar Truss problem with stress and displacement constraints (Case 1).

MS-SN P.AD. PFPR - 10-Bar Truss with Stress & Displ. Constraints 1					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.9681	-	-	-
	uy	-5.0805	-5.0800	-5.0800	**100.01**
	uz	0.0000	-	-	-
2	ux	-1.9062	-	-	-
	uy	-7.3665	-	-	-
	uz	0.0000	-	-	-
3	ux	0.8036	-	-	-
	uy	-2.5403	-2.5400	-2.5400	**100.01**
	uz	0.0000	-	-	-
4	ux	-0.8527	-	-	-
	uy	-2.9354	-	-	-
	uz	0.0000	-	-	-
5	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
6	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

8.8.5.3. Stress and Displacements Constraints Case 2

This is the same problem as in section 8.8.5.1 but with the *y*-displacements in nodes '1' to '4' restricted to the interval ± 0.0508. Therefore Eq. (8.38) is replaced by Eq. (8.41):

$$
\begin{aligned}
& g_i(\mathbf{x}) = \max\left[\left(\frac{\sigma_i}{\sigma_{a+}^{(i)}} - 1\right), \left(\frac{\sigma_i}{\sigma_{a-}^{(i)}} - 1\right)\right] \leq 0 \quad ; \quad i = 1,...,10 \\
& g_{11}(\mathbf{x}) = abs\left(\frac{u_{y1}}{0.0508}\right) - 1 \leq 0 \\
& g_{12}(\mathbf{x}) = abs\left(\frac{u_{y2}}{0.0508}\right) - 1 \leq 0 \\
& g_{13}(\mathbf{x}) = abs\left(\frac{u_{y3}}{0.0508}\right) - 1 \leq 0 \\
& g_{14}(\mathbf{x}) = abs\left(\frac{u_{y4}}{0.0508}\right) - 1 \leq 0
\end{aligned} \tag{8.41}
$$

$$
x_i - 200 \leq 0 \quad ; \quad -x_i + 0.6452 \leq 0 \quad ; \quad i = 1,..,10
$$

Therefore there are now 10 interval constraints and 14 inequality constraints. The results are shown in Table 8.37, in Table 8.38, and in Fig. 8.19. The solutions obtained in (Fleury & Schmit, 1980), in (Park & Ryu, 2004), and in (Li, Huang, Liu, & Wu, 2007) are also reported for reference.

Table 8.37. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by other authors for the 10-Bar Truss problem with stress and four inequality-displacement constraints (nodes '1' to '4'). The authors included are: (Fleury & Schmit, 1980); (Park & Ryu, 2004); and (Li, Huang, Liu, & Wu, 2007).

10-Bar Truss (Stress & Displacement 2)	OPTIIMUM (SI)	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	[%] Successful Solutions	Mean FEs	Runs
MS-SN P.AD. PFPR	-	**2295.406788**	**2295.521831**	**2295.832101**	**2302.669693**	**100.00**	**4.00**	**1.67E+05**	**25**
(Fleury & Schmit, 1980) (NEWSUMT)		2308.353600	-	-	-	-	-	-	-
(Fleury & Schmit, 1980) (DUAL 2)		2295.600000	-	-	-	-	-	-	-
(Park & Ryu, 2004) (RSA)		2329.702943	-	-	-	-	-	4.16E+03	
(Li, Huang, Liu, & Wu, 2007) (HPSO)		2295.592825	-	-	-	-	-	1.50E+05	-

Table 8.38. Coordinates of the best solutions obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO and by other authors for the 10-Bar Truss problem with stress and four inequality-displacement constraints (nodes '1' to '4'). The authors included are: (Fleury & Schmit, 1980); (Park & Ryu, 2004); and (Li, Huang, Liu, & Wu, 2007).

	Coordinates of best solution									
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
MS-SN P.AD. PFPR	**196.036900**	**0.645225**	**150.371312**	**98.335803**	**0.645290**	**3.497145**	**48.056234**	**136.008467**	**138.695325**	**0.645201**
(Fleury & Schmit, 1980) (NEWSUMT)	199.7000	0.6452	168.3000	97.0400	0.6452	1.2650	52.7900	130.5000	130.5000	0.6452
(Fleury & Schmit, 1980) (DUAL 2)	196.9000	0.6452	149.7000	98.2000	0.6452	3.5550	48.1100	135.8000	138.9000	0.6452
(Park & Ryu, 2004) (RSA)	182.1287	0.6452	148.4513	82.4514	0.6452	0.6452	57.1612	163.8061	135.6126	0.6452
(Li, Huang, Liu, & Wu, 2007) (HPSO)	198.0899	0.6452	149.4642	97.9546	0.6452	3.5548	48.1289	135.3417	138.7610	0.6452

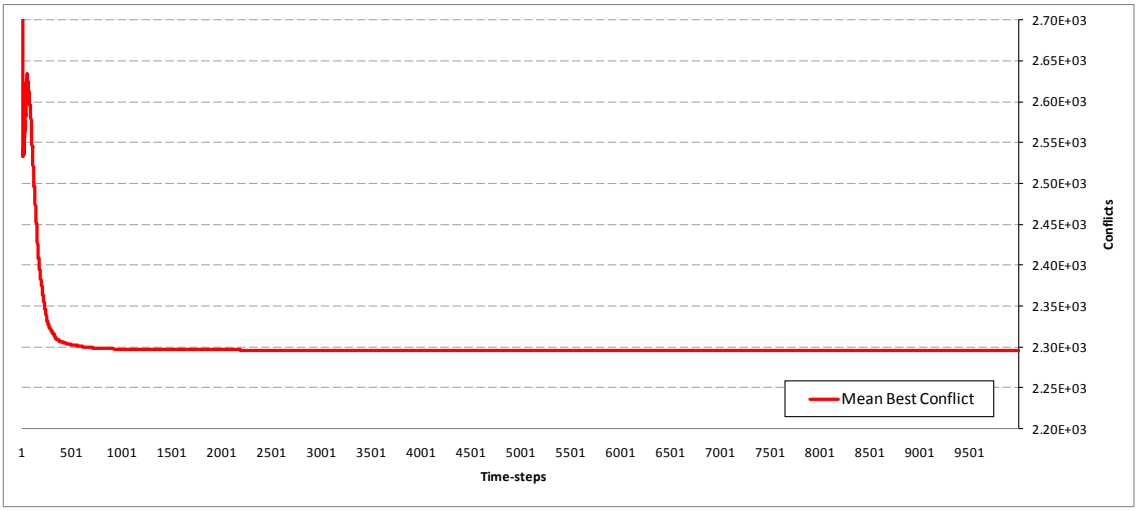


Fig. 8.19. Convergence of the mean best solution for the 10-Bar Truss problem with stress and four inequality-displacement constraints (nodes '1' to '4').

Note that the best solution in Table 8.37 is found by the MS-SN P.AD. PFPR.

The details of the of the FE analysis of the optimal design for the 10-Bar Truss problem with Stress and Displacement Constraints (Case 2) from (Fleury & Schmit, 1980) (where mass = 2295.6 kg) are offered in Table 8.39 and in Table 8.40; whereas those of the best design found by the MS-SN P.AD. PFPR PSO (mass = 2295.406788 kg) are offered in Table 8.41 and in Table 8.42. The stress constraint is marginally violated for bar '5' in Table 8.39, most likely due to the round-off in the input of the cross-sections of the optimal design from (Fleury & Schmit, 1980). The displacement constraints are either active or almost active for nodes '1' and '2' in both Table 8.40 and Table 8.42.

Table 8.39. Cross sections (areas) and stresses details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) of the 10-Bar Truss problem with stress and displacement constraints (Case 2).

(Fleury & Schmit, 1980) DUAL 2 - 10-Bar Truss with Stress & Displacement Constraints 2 (mass = **2295.60** kg)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	196.9000	0.6452	30517.67	4.58E+07	-1.72E+08	1.72E+08	26.56
2	2	0.6452	0.6452	**100.00**	-9.07E+06	-1.72E+08	1.72E+08	5.26
3	3	149.7000	0.6452	23202.11	-5.86E+07	-1.72E+08	1.72E+08	34.02
4	4	98.2000	0.6452	15220.09	-4.54E+07	-1.72E+08	1.72E+08	26.31
5	5	0.6452	0.6452	**100.00**	1.72E+08	-1.72E+08	1.72E+08	**100.02**
6	6	3.5550	0.6452	550.99	-1.65E+06	-1.72E+08	1.72E+08	0.95
7	7	48.1100	0.6452	7456.60	1.27E+08	-1.72E+08	1.72E+08	73.86
8	8	135.8000	0.6452	21047.74	-4.75E+07	-1.72E+08	1.72E+08	27.58
9	9	138.9000	0.6452	21528.21	4.53E+07	-5.17E+08	5.17E+08	8.77
10	10	0.6452	0.6452	**100.00**	1.28E+07	-1.72E+08	1.72E+08	7.44

Table 8.40. Displacement details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) of the 10-Bar Truss problem with stress and displacement constraints (Case 2).

(Fleury & Schmit, 1980) DUAL 2 - 10-Bar Truss with Stress & Displacement Constraints 1					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.4868	-	-	-
	uy	-5.0788	-5.0800	5.0800	99.98
	uz	0.0000	-	-	-
2	ux	-1.3792	-	-	-
	uy	-5.0570	-5.0800	5.0800	99.55
	uz	0.0000	-	-	-
3	ux	0.6071	-	-	-
	uy	-1.8680	-5.0800	5.0800	36.77
	uz	0.0000	-	-	-
4	ux	-0.7777	-	-	-
	uy	-4.1544	-5.0800	5.0800	81.78
	uz	0.0000	-	-	-
5	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
6	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

Table 8.41. Cross sections (areas) and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 10-Bar Truss problem with stress and displacement constraints (Case 2).

MS-SN P.AD. PFPR - 10-Bar Truss with Stress & Displacement Constraints 2 (mass = **2295.406788** kg)								
Variables	Bars	Areas [cm²]	Min. areas [cm²]	[%] min. areas	Stresses [N/m²]	Allowable Stresses [N/m²]		[%] max. stress
1	1	196.0369	0.6452	30383.90	4.60E+07	-1.72E+08	1.72E+08	26.67
2	2	0.6452	0.6452	100.00	-9.03E+06	-1.72E+08	1.72E+08	5.24
3	3	150.3713	0.6452	23306.16	-5.84E+07	-1.72E+08	1.72E+08	33.87
4	4	98.3358	0.6452	15241.14	-4.53E+07	-1.72E+08	1.72E+08	26.28
5	5	0.6453	0.6452	100.01	1.72E+08	-1.72E+08	1.72E+08	100.00
6	6	3.4971	0.6452	542.02	-1.67E+06	-1.72E+08	1.72E+08	0.97
7	7	48.0562	0.6452	7448.27	1.27E+08	-1.72E+08	1.72E+08	73.94
8	8	136.0085	0.6452	21080.05	-4.75E+07	-1.72E+08	1.72E+08	27.54
9	9	138.6953	0.6452	21496.49	4.54E+07	-5.17E+08	5.17E+08	8.78
10	10	0.6452	0.6452	100.00	1.28E+07	-1.72E+08	1.72E+08	7.41

Table 8.42. Displacement details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 10-Bar Truss problem with stress and displacement constraints (Case 2).

MS-SN P.AD. PFPR - 10-Bar Truss with Stress & Displ. Constraints 2					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.4899	-	-	-
	uy	-5.0800	-5.0800	5.0800	100.00
	uz	0.0000	-	-	-
2	ux	-1.3749	-	-	-
	uy	-5.0579	-5.0800	5.0800	99.56
	uz	0.0000	-	-	-
3	ux	0.6097	-	-	-
	uy	-1.8687	-5.0800	5.0800	36.79
	uz	0.0000	-	-	-
4	ux	-0.7742	-	-	-
	uy	-4.1547	-5.0800	5.0800	81.79
	uz	0.0000	-	-	-
5	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
6	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

8.8.6. 25-Bar Space Truss

This is another classical engineering problem used to test optimization algorithms. Again, there are different versions in the literature. The formulations of the problem implemented here are taken from (Fleury & Schmit, 1980), (Li, Huang, Liu, & Wu, 2007), and (Park & Ryu, 2004), and are expressed in SI units. The optimization problem is as shown in Fig. 8.20 and Eq. (8.42) to Eq. (8.44).

$$\begin{cases} \text{Aluminium} \\ \text{E} = 68.95 \text{ x } 10^6 \text{ kN/m}^2 \\ \rho = 2768 \text{ kg/m}^3 \\ A_{\min} = x_{\min} = 0.06452 \text{ cm}^2 \\ \sigma_{a+} = +275800 \text{ kN/m}^2 \quad \text{(allowable tension stress)} \end{cases} \tag{8.42}$$

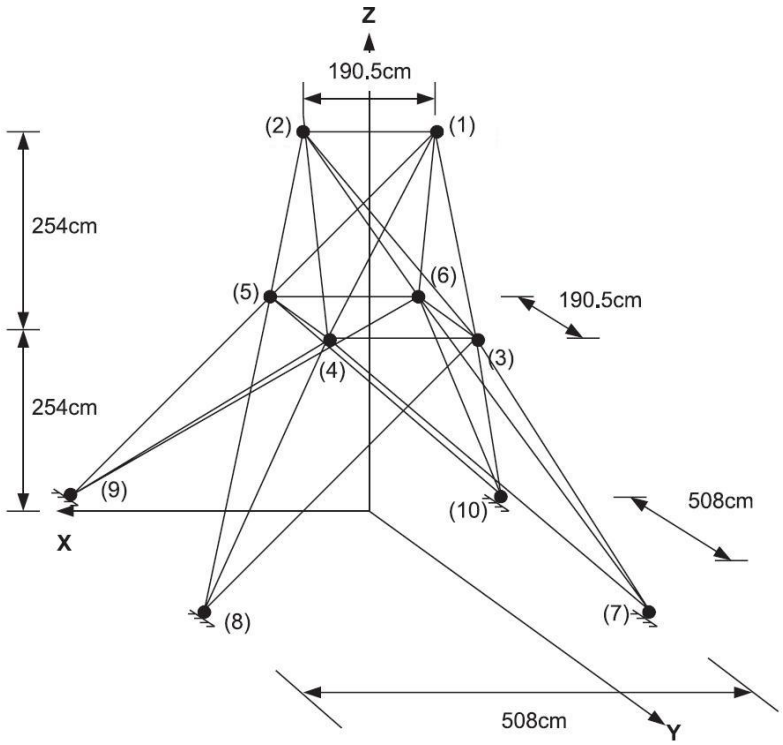
Minimize:

$$f(\mathbf{x}) = \sum_{i=1}^{10} \rho \cdot x_i \cdot l_i \tag{8.43}$$

Subject to:

$$\begin{aligned} & g_i(\mathbf{x}) = \max\left[\left(\frac{\sigma_i}{\sigma_{a+}^{(i)}} - 1\right), \left(\frac{\sigma_i}{\sigma_{a-}^{(i)}} - 1\right)\right] \le 0 \quad ; \quad i = 1,...,25 \\ & g_{26}(\mathbf{x}) = abs\left(\frac{u_{x1}}{0.00889}\right) - 1 \le 0 \\ & g_{27}(\mathbf{x}) = abs\left(\frac{u_{y1}}{0.00889}\right) - 1 \le 0 \\ & g_{28}(\mathbf{x}) = abs\left(\frac{u_{z1}}{0.00889}\right) - 1 \le 0 \\ & g_{29}(\mathbf{x}) = abs\left(\frac{u_{x2}}{0.00889}\right) - 1 \le 0 \\ & g_{30}(\mathbf{x}) = abs\left(\frac{u_{y2}}{0.00889}\right) - 1 \le 0 \\ & g_{31}(\mathbf{x}) = abs\left(\frac{u_{z2}}{0.00889}\right) - 1 \le 0 \end{aligned} \tag{8.44}$$

$$x_i - 100 \le 0 \quad ; \quad -x_i + 0.06452 \le 0 \quad ; \quad i = 1,...,8$$



Bar	Nodes	
1	1	2
2	1	4
3	2	3
4	1	5
5	2	6
6	2	4
7	2	5
8	1	3
9	1	6
10	3	6
11	4	5
12	3	4
13	5	6
14	3	10
15	6	7
16	4	9
17	5	8
18	4	7
19	3	8
20	5	10
21	6	9
22	6	10
23	3	7
24	4	8
25	5	9

Fig. 8.20. 25-Bar Truss problem in SI units, from (Azid, Kwan, & Seetharamu, 2002) with marginal modification. The objective is to minimize the mass while complying with constraints. The table of connectivity is also provided.

8.8.6.1. Case 1 (Fleury & Schmit, 1980)

The 25-Bar truss problem as posed in (Fleury & Schmit, 1980) presents two Load Cases. Here the problem is solved for each Load Case (LC) separately, followed by a classical ad-hoc design. Finally, the problem is solved so that both independent Load Cases are considered simultaneously throughout the optimization process (thus doubling the number of constraints). The allowable compression stresses are offered in Table 8.43, which were calculated considering Euler's critical buckling load.

Table 8.43. Allowable compression stress for the members of the 25-Bar Truss from (Fleury & Schmit, 1980).

Sections (variables)	Members	Allowable compression stress [kN/m^2]
x_1	1	-241959
x_2	2 – 5	-79913
x_3	6 – 9	-119318
x_4	10, 11	-241959
x_5	12, 13	-241959
x_6	14 – 17	-46603
x_7	18 – 21	-47982
x_8	22 – 25	-76410

Ad-hoc design

Load Case 1

The load components are shown in Table 8.44, while the results for this Load Case alone are offered in Table 8.45, in Table 8.46, and in Fig. 8.21.

Table 8.44. Load components on the nodes (Load Case 1), from (Fleury & Schmit, 1980).

NODES	LOAD COMPONENTS		
	x	*y*	*z*
1	4448	44480	-22240
2	0	44480	-22240
3	2224	0	0
6	2224	0	0

Table 8.45. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with Load Case 1, as in (Fleury & Schmit, 1980).

25-Bar Truss, Load Case 1 (Fleury & Schmit, 1980)	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	Mean FEs	[%] FR	[%] FR Initial Tol.	Runs
MS-SN P.AD. PFPR	207.051222	207.052911	207.053792	207.060904	100.00	1.86E+05	92.77	99.80	25

Table 8.46. Coordinates of the best solution obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with Load Case 1, as in (Fleury & Schmit, 1980).

Coordinates of best solution							
x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
0.180386	0.472271	23.105736	0.064520	13.440916	4.953560	0.145487	25.471843

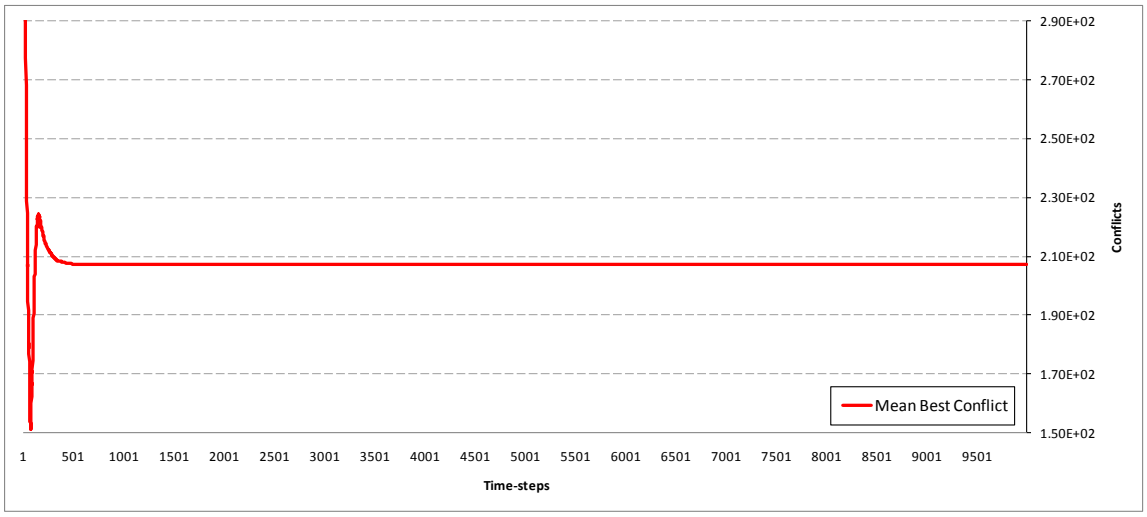


Fig. 8.21. Convergence of the mean best solution for the 25-Bar Truss problem with Load Case 1.

The details of the of the FE analysis are offered in Table 8.47 and in Table 8.48.

Table 8.47. Cross sections (areas) and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with Load Case 1 (see Table 8.44).

MS-SN P.AD. PFPR - 25-Bar Truss Load Case 1 (mass = **207.051222** kg)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	0.1804	0.06452	279.58	-7.39E+07	-2.42E+08	2.76E+08	30.56
2	2	0.4723	0.06452	731.98	-7.99E+07	-7.99E+07	2.76E+08	**100.00**
	3	0.4723	0.06452	731.98	2.03E+06	-7.99E+07	2.76E+08	0.74
	4	0.4723	0.06452	731.98	-3.48E+07	-7.99E+07	2.76E+08	43.58
	5	0.4723	0.06452	731.98	4.71E+07	-7.99E+07	2.76E+08	17.08
3	6	23.1057	0.06452	35811.74	-3.26E+07	-1.19E+08	2.76E+08	27.31
	7	23.1057	0.06452	35811.74	2.15E+07	-1.19E+08	2.76E+08	7.79
	8	23.1057	0.06452	35811.74	-3.12E+07	-1.19E+08	2.76E+08	26.16
	9	23.1057	0.06452	35811.74	2.29E+07	-1.19E+08	2.76E+08	8.29
4	10	0.0645	0.06452	**100.00**	-7.29E+06	-2.42E+08	2.76E+08	3.01
	11	0.0645	0.06452	**100.00**	-1.20E+07	-2.42E+08	2.76E+08	4.95
5	12	13.4409	0.06452	20832.17	-3.25E+07	-2.42E+08	2.76E+08	13.43
	13	13.4409	0.06452	20832.17	2.13E+07	-2.42E+08	2.76E+08	7.71
6	14	4.9536	0.06452	7677.56	-3.08E+07	-4.66E+07	2.76E+08	65.98
	15	4.9536	0.06452	7677.56	2.35E+07	-4.66E+07	2.76E+08	8.51
	16	4.9536	0.06452	7677.56	-3.34E+07	-4.66E+07	2.76E+08	71.68
	17	4.9536	0.06452	7677.56	2.08E+07	-4.66E+07	2.76E+08	7.55
7	18	0.1455	0.06452	225.49	-1.83E+06	-4.80E+07	2.76E+08	3.82
	19	0.1455	0.06452	225.49	-4.80E+07	-4.80E+07	2.76E+08	**100.00**
	20	0.1455	0.06452	225.49	3.99E+07	-4.80E+07	2.76E+08	14.46
	21	0.1455	0.06452	225.49	-6.27E+06	-4.80E+07	2.76E+08	13.07
8	22	25.4718	0.06452	39478.99	2.35E+07	-7.64E+07	2.76E+08	8.51
	23	25.4718	0.06452	39478.99	-3.07E+07	-7.64E+07	2.76E+08	40.23
	24	25.4718	0.06452	39478.99	-3.37E+07	-7.64E+07	2.76E+08	44.05
	25	25.4718	0.06452	39478.99	2.05E+07	-7.64E+07	2.76E+08	7.45

Table 8.48. Displacement details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with Load Case 1 (see Table 8.44).

MS-SN P.AD. PFPR - 25-Bar Truss Load Case 1 (mass = 207.051222 kg)					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.3803	-0.8890	0.8890	42.78
	uy	0.8890	-0.8890	0.8890	**100.00**
	uz	-0.1390	-0.8890	0.8890	15.63
2	ux	0.1760	-0.8890	0.8890	19.79
	uy	0.8890	-0.8890	0.8890	**100.00**
	uz	0.0148	-0.8890	0.8890	1.67
3	ux	0.1891	-	-	-
	uy	-0.0850	-	-	-
	uz	-0.3730	-	-	-
4	ux	0.0994	-	-	-
	uy	-0.0915	-	-	-
	uz	-0.2159	-	-	-
5	ux	0.1736	-	-	-
	uy	-0.0584	-	-	-
	uz	0.2798	-	-	-
6	ux	0.1149	-	-	-
	uy	-0.0648	-	-	-
	uz	0.1227	-	-	-
7	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
8	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
9	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
10	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

Load Case 2

The load components on the nodes are shown in Table 8.49, while the results are provided in Table 8.50, in Table 8.51, and in Fig. 8.22.

Table 8.49. Load components on the nodes (Load Case 2), from 'Table 9 Case 1' in (Li, Huang, Liu, & Wu, 2007).

NODES	LOAD COMPONENTS		
	x	*y*	*z*
1	0	88960	-22240
2	0	-88960	-22240

Table 8.50. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with Load Case 2, as in (Li, Huang, Liu, & Wu, 2007).

25-Bar Truss, Load Case 2 (Li, Huang, Liu, & Wu, 2007)	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	Mean FEs	[%] FR	[%] FR Initial Tol.	Runs
MS-SN P.AD. PFPR	**177.836584**	**177.962599**	**177.971916**	**178.263114**	**100.00**	**1.58E+05**	**85.72**	**94.93**	**25**

Table 8.51. Coordinates of the best solution obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with Load Case 2, as in (Li, Huang, Liu, & Wu, 2007).

Coordinates of best solution							
x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
0.064520	0.064520	0.064520	8.702263	1.126640	13.324509	0.109402	6.985344

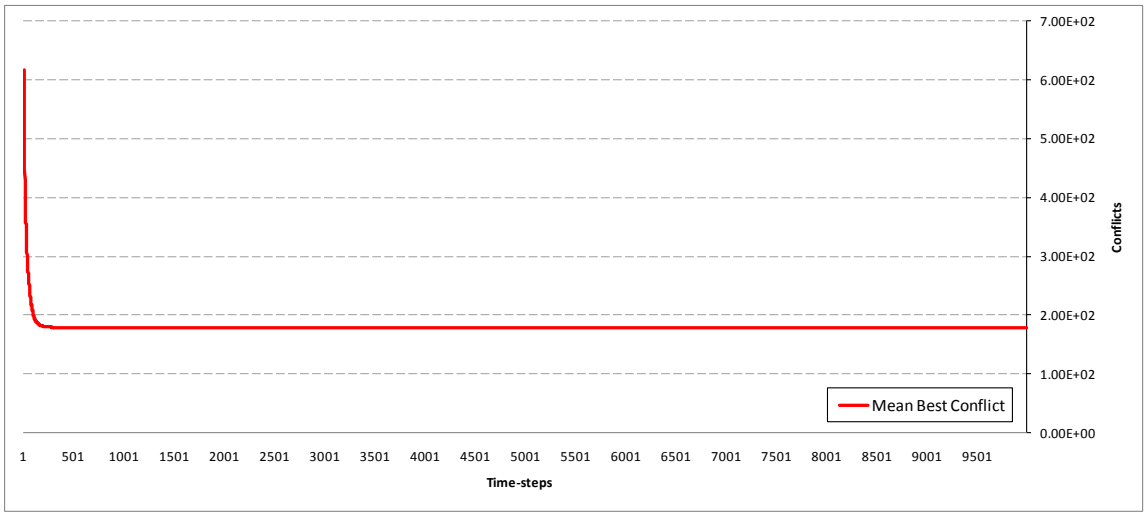


Fig. 8.22. Convergence of the mean best solution for the 25-Bar Truss problem with Load Case 2.

The details of the of the FE analysis are offered in Table 8.52 and in Table 8.53.

Table 8.52. Cross sections (areas) and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with Load Case 2 (see Table 8.49).

MS-SN P.AD. PFPR - 25-Bar Truss Load Case 2 (mass = **177.836584** kg)								
Variables	Bars	Areas [cm²]	Min. areas [cm²]	[%] min. areas	Stresses [N/m²]	Allowable Stresses [N/m²]		[%] max. stress
1	1	0.0795	0.06452	123.14	-7.88E+07	-2.42E+08	2.76E+08	32.58
2	2	13.1804	0.06452	20428.47	-4.40E+07	-7.99E+07	2.76E+08	55.01
	3	13.1804	0.06452	20428.47	4.48E+07	-7.99E+07	2.76E+08	16.24
	4	13.1804	0.06452	20428.47	4.48E+07	-7.99E+07	2.76E+08	16.24
	5	13.1804	0.06452	20428.47	-4.40E+07	-7.99E+07	2.76E+08	55.01
3	6	17.8010	0.06452	27589.92	3.74E+07	-1.19E+08	2.76E+08	13.54
	7	17.8010	0.06452	27589.92	-5.12E+07	-1.19E+08	2.76E+08	42.91
	8	17.8010	0.06452	27589.92	-5.12E+07	-1.19E+08	2.76E+08	42.91
	9	17.8010	0.06452	27589.92	3.74E+07	-1.19E+08	2.76E+08	13.54
4	10	0.0648	0.06452	**100.43**	8.83E+07	-2.42E+08	2.76E+08	32.01
	11	0.0648	0.06452	**100.43**	8.83E+07	-2.42E+08	2.76E+08	32.01
5	12	0.5671	0.06452	878.99	1.09E+08	-2.42E+08	2.76E+08	39.67
	13	0.5671	0.06452	878.99	1.09E+08	-2.42E+08	2.76E+08	39.67
6	14	2.0592	0.06452	3191.61	-4.66E+07	-4.66E+07	2.76E+08	**100.00**
	15	2.0592	0.06452	3191.61	1.71E+07	-4.66E+07	2.76E+08	6.21
	16	2.0592	0.06452	3191.61	1.71E+07	-4.66E+07	2.76E+08	6.21
	17	2.0592	0.06452	3191.61	-4.66E+07	-4.66E+07	2.76E+08	**100.00**
7	18	12.3056	0.06452	19072.56	2.82E+07	-4.80E+07	2.76E+08	10.21
	19	12.3056	0.06452	19072.56	-4.80E+07	-4.80E+07	2.76E+08	**100.00**
	20	12.3056	0.06452	19072.56	-4.80E+07	-4.80E+07	2.76E+08	**100.00**
	21	12.3056	0.06452	19072.56	2.82E+07	-4.80E+07	2.76E+08	10.21
8	22	0.5623	0.06452	871.45	-7.64E+07	-7.64E+07	2.76E+08	**100.00**
	23	0.5623	0.06452	871.45	-5.22E+07	-7.64E+07	2.76E+08	68.37
	24	0.5623	0.06452	871.45	-7.64E+07	-7.64E+07	2.76E+08	**100.00**
	25	0.5623	0.06452	871.45	-5.22E+07	-7.64E+07	2.76E+08	68.37

Table 8.53. Displacement details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with Load Case 2 (see Table 8.49).

MS-SN P.AD. PFPR - 25-Bar Truss Load Case 2 (mass = 177.836584 kg)					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.1089	-0.8890	0.8890	12.25
	uy	0.8890	-0.8890	0.8890	**100.00**
	uz	-0.3262	-0.8890	0.8890	36.70
2	ux	-0.1089	-0.8890	0.8890	12.25
	uy	-0.8890	-0.8890	0.8890	**100.00**
	uz	-0.3262	-0.8890	0.8890	36.70
3	ux	0.1186	-	-	-
	uy	-0.1102	-	-	-
	uz	-0.4858	-	-	-
4	ux	0.4209	-	-	-
	uy	0.3541	-	-	-
	uz	-0.0170	-	-	-
5	ux	-0.1186	-	-	-
	uy	0.1102	-	-	-
	uz	-0.4858	-	-	-
6	ux	-0.4209	-	-	-
	uy	-0.3541	-	-	-
	uz	-0.0170	-	-	-
7	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
8	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
9	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
10	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

It is fair to note that, if the FE analysis is performed on the optimal design reported in (Fleury & Schmit, 1980) for the second Load Case posed in (Fleury & Schmit, 1980, p. 142), the stress constraints of bars '10', '11, and '17' are violated by a large margin. It appears that there is a mistake in the formulation of the second Load Case. If the latter is replaced by the first Load Case in (Li, Huang, Liu, & Wu, 2007), all constraints are satisfied, while the stress constraints of bars '19' and '20' as well as the *y*-displacement-constraints on nodes '1' and '2' are nearly active as reported in (Fleury & Schmit, 1980, p. 102). This is the reason why the second Load Case considered here (see Table 8.49) is as in (Li, Huang, Liu, & Wu, 2007) rather than as in (Fleury & Schmit, 1980, p. 142).

A possible ad-hoc design for both Load Cases consists of choosing the greater cross-section of each bar among both Load Cases, as shown in Table 8.54. The resulting mass for this design equals **315.629616** kg.

Table 8.54. Coordinates of an ad-hoc solution for both Load Cases, from the solutions for each independent Load Case obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem as in (Fleury & Schmit, 1980) and in (Li, Huang, Liu, & Wu, 2007).

Coordinates of best solution							
x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
0.175509	13.180447	23.096234	0.064797	13.411980	4.952039	12.305618	25.487831

The details of the FE analysis of this ad-hoc design are offered in Table 8.55 and in Table 8.56 for the Load Case 1 (see Table 8.44), and in Table 8.57 and in Table 8.58 for the Load Case 2 (see Table 8.49). As it can be observed in the tables, no stress or displacement constraint is active (or nearly active).

Table 8.55. Cross sections (areas) and stresses details from the FE analysis of the ad-hoc design in Table 8.54 for the 25-Bar Truss problem with Load Case 1 (see Table 8.44).

MS-SN P.AD. PFPR - Ad-hoc Design - 25-Bar Truss Load Case 1 (mass = **315.629616** kg)								
Variables	Bars	Areas [cm²]	Min. areas [cm²]	[%] min. areas	Stresses [N/m²]	Allowable Stresses [N/m²]		[%] max. stress
1	1	0.1755	0.06452	272.02	1.05E+07	-2.42E+08	2.76E+08	3.82
2	2	13.1804	0.06452	20428.47	-1.78E+07	-7.99E+07	2.76E+08	22.23
	3	13.1804	0.06452	20428.47	-1.48E+07	-7.99E+07	2.76E+08	18.56
	4	13.1804	0.06452	20428.47	1.16E+07	-7.99E+07	2.76E+08	4.22
	5	13.1804	0.06452	20428.47	1.46E+07	-7.99E+07	2.76E+08	5.29
3	6	23.0962	0.06452	35797.01	-2.56E+07	-1.19E+08	2.76E+08	21.49
	7	23.0962	0.06452	35797.01	1.55E+07	-1.19E+08	2.76E+08	5.61
	8	23.0962	0.06452	35797.01	-2.43E+07	-1.19E+08	2.76E+08	20.34
	9	23.0962	0.06452	35797.01	1.68E+07	-1.19E+08	2.76E+08	6.11
4	10	0.0648	0.06452	**100.43**	-9.48E+06	-2.42E+08	2.76E+08	3.92
	11	0.0648	0.06452	**100.43**	-1.42E+07	-2.42E+08	2.76E+08	5.86
5	12	13.4120	0.06452	20787.32	-8.01E+06	-2.42E+08	2.76E+08	3.31
	13	13.4120	0.06452	20787.32	3.39E+06	-2.42E+08	2.76E+08	1.23
6	14	4.9520	0.06452	7675.20	-3.07E+07	-4.66E+07	2.76E+08	65.95
	15	4.9520	0.06452	7675.20	2.35E+07	-4.66E+07	2.76E+08	8.52
	16	4.9520	0.06452	7675.20	-3.34E+07	-4.66E+07	2.76E+08	**71.65**
	17	4.9520	0.06452	7675.20	2.08E+07	-4.66E+07	2.76E+08	7.56
7	18	12.3056	0.06452	19072.56	-1.60E+07	-4.80E+07	2.76E+08	33.25
	19	12.3056	0.06452	19072.56	-1.65E+07	-4.80E+07	2.76E+08	34.38
	20	12.3056	0.06452	19072.56	1.10E+07	-4.80E+07	2.76E+08	3.97
	21	12.3056	0.06452	19072.56	1.04E+07	-4.80E+07	2.76E+08	3.78
8	22	25.4878	0.06452	39503.77	1.97E+07	-7.64E+07	2.76E+08	7.15
	23	25.4878	0.06452	39503.77	-2.51E+07	-7.64E+07	2.76E+08	32.79
	24	25.4878	0.06452	39503.77	-2.80E+07	-7.64E+07	2.76E+08	36.61
	25	25.4878	0.06452	39503.77	1.68E+07	-7.64E+07	2.76E+08	6.09

Table 8.56. Displacement details from the FE analysis of the ad-hoc design in Table 8.54 for the 25-Bar Truss problem with Load Case 1 (see Table 8.44).

MS-SN P.AD. PFPR - Ad-hoc Design - 25-Bar Truss Load Case 1					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.0250	-0.8890	0.8890	2.81
	uy	0.6955	-0.8890	0.8890	**78.23**
	uz	-0.0424	-0.8890	0.8890	4.77
2	ux	0.0541	-0.8890	0.8890	6.08
	uy	0.6955	-0.8890	0.8890	**78.23**
	uz	-0.0609	-0.8890	0.8890	-6.85
3	ux	0.0175	-	-	-
	uy	-0.1035	-	-	-
	uz	-0.2401	-	-	-
4	ux	-0.0046	-	-	-
	uy	-0.1100	-	-	-
	uz	-0.2552	-	-	-
5	ux	0.0111	-	-	-
	uy	-0.0709	-	-	-
	uz	0.1615	-	-	-
6	ux	0.0018	-	-	-
	uy	-0.0773	-	-	-
	uz	0.1766	-	-	-
7	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
8	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
9	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
10	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

Table 8.57. Cross sections (areas) and stresses details from the FE analysis of the ad-hoc design in Table 8.54 for the 25-Bar Truss problem with Load Case 2 (see Table 8.49).

MS-SN P.AD. PFPR - Ad-hoc Design - 25-Bar Truss Load Case 2 (mass = **315.629616** kg)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	0.1755	0.06452	272.02	2.12E+07	-2.42E+08	2.76E+08	7.67
2	2	13.1804	0.06452	20428.47	-4.55E+07	-7.99E+07	2.76E+08	56.93
	3	13.1804	0.06452	20428.47	4.50E+07	-7.99E+07	2.76E+08	16.32
	4	13.1804	0.06452	20428.47	4.50E+07	-7.99E+07	2.76E+08	16.32
	5	13.1804	0.06452	20428.47	-4.55E+07	-7.99E+07	2.76E+08	56.93
3	6	23.0962	0.06452	35797.01	2.87E+07	-1.19E+08	2.76E+08	10.40
	7	23.0962	0.06452	35797.01	-3.87E+07	-1.19E+08	2.76E+08	32.47
	8	23.0962	0.06452	35797.01	-3.87E+07	-1.19E+08	2.76E+08	32.47
	9	23.0962	0.06452	35797.01	2.87E+07	-1.19E+08	2.76E+08	10.40
4	10	0.0648	0.06452	**100.43**	-1.18E+07	-2.42E+08	2.76E+08	4.88
	11	0.0648	0.06452	**100.43**	-1.18E+07	-2.42E+08	2.76E+08	4.88
5	12	13.4120	0.06452	20787.32	-2.25E+06	-2.42E+08	2.76E+08	0.93
	13	13.4120	0.06452	20787.32	-2.25E+06	-2.42E+08	2.76E+08	0.93
6	14	4.9520	0.06452	7675.20	-1.74E+07	-4.66E+07	2.76E+08	37.31
	15	4.9520	0.06452	7675.20	7.50E+06	-4.66E+07	2.76E+08	2.72
	16	4.9520	0.06452	7675.20	7.50E+06	-4.66E+07	2.76E+08	2.72
	17	4.9520	0.06452	7675.20	-1.74E+07	-4.66E+07	2.76E+08	37.31
7	18	12.3056	0.06452	19072.56	3.50E+07	-4.80E+07	2.76E+08	12.69
	19	12.3056	0.06452	19072.56	-4.05E+07	-4.80E+07	2.76E+08	**84.41**
	20	12.3056	0.06452	19072.56	-4.05E+07	-4.80E+07	2.76E+08	**84.41**
	21	12.3056	0.06452	19072.56	3.50E+07	-4.80E+07	2.76E+08	12.69
8	22	25.4878	0.06452	39503.77	-5.09E+06	-7.64E+07	2.76E+08	6.66
	23	25.4878	0.06452	39503.77	-3.18E+06	-7.64E+07	2.76E+08	4.16
	24	25.4878	0.06452	39503.77	-5.09E+06	-7.64E+07	2.76E+08	6.66
	25	25.4878	0.06452	39503.77	-3.18E+06	-7.64E+07	2.76E+08	4.16

Table 8.58. Displacement details from the FE analysis of the ad-hoc design in Table 8.54 for the 25-Bar Truss problem with Load Case 2 (see Table 8.49).

MS-SN P.AD. PFPR - Ad-hoc Design - 25-Bar Truss Load Case 2					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	-0.0292	-0.8890	0.8890	3.29
	uy	0.7987	-0.8890	0.8890	**89.84**
	uz	-0.0543	-0.8890	0.8890	6.10
2	ux	0.0292	-0.8890	0.8890	3.29
	uy	-0.7987	-0.8890	0.8890	**89.84**
	uz	-0.0543	-0.8890	0.8890	6.10
3	ux	0.2344	-	-	-
	uy	-0.0947	-	-	-
	uz	-0.2265	-	-	-
4	ux	0.2282	-	-	-
	uy	0.0620	-	-	-
	uz	0.1480	-	-	-
5	ux	-0.2344	-	-	-
	uy	0.0947	-	-	-
	uz	-0.2265	-	-	-
6	ux	-0.2282	-	-	-
	uy	-0.0620	-	-	-
	uz	0.1480	-	-	-
7	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
8	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
9	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
10	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

Simultaneous, optimal design

A more clever optimal design may be carried out by simultaneously complying with the constraints associated with both independent Load Cases throughout the search. Thus the number of constraints is doubled, replacing Eq. (8.44) by Eq. (8.45). Note that in this case, two FE analyses are performed for each candidate solution.

The results obtained are provided in Table 8.59, in Table 8.60, and in Fig. 8.23. The best solutions from (Fleury & Schmit, 1980) and from (Li, Huang, Liu, & Wu, 2007) are also provided for reference.

Note that the feasibility ratio (FR) equals **92.77%** for the problem with Load Case 1 (see Table 8.45), it is equal to **85.72%** for the problem with Load Case 2 (see Table 8.50), while for the problem with both Load Cases the FR equals **82.22**%.

$$g_i(\mathbf{x}) = \max\left[\left(\frac{\sigma_i^{(lc=1)}}{\sigma_{a+}^{(i)}} - 1\right), \left(\frac{\sigma_i^{(lc=1)}}{\sigma_{a-}^{(i)}} - 1\right)\right] \leq 0 \quad ; \quad i = 1,...,25$$

$$g_{26}(\mathbf{x}) = abs\left(\frac{u_{x1}^{(lc=1)}}{0.00889}\right) - 1 \leq 0$$

$$g_{27}(\mathbf{x}) = abs\left(\frac{u_{y1}^{(lc=1)}}{0.00889}\right) - 1 \leq 0$$

$$g_{28}(\mathbf{x}) = abs\left(\frac{u_{z1}^{(lc=1)}}{0.00889}\right) - 1 \leq 0$$

$$g_{29}(\mathbf{x}) = abs\left(\frac{u_{x2}^{(lc=1)}}{0.00889}\right) - 1 \leq 0$$

$$g_{30}(\mathbf{x}) = abs\left(\frac{u_{y2}^{(lc=1)}}{0.00889}\right) - 1 \leq 0$$

$$g_{31}(\mathbf{x}) = abs\left(\frac{u_{z2}^{(lc=1)}}{0.00889}\right) - 1 \leq 0$$

$$g_{i+31}(\mathbf{x}) = \max\left[\left(\frac{\sigma_i^{(lc=2)}}{\sigma_{a+}^{(i)}} - 1\right), \left(\frac{\sigma_i^{(lc=2)}}{\sigma_{a-}^{(i)}} - 1\right)\right] \leq 0 \quad ; \quad i = 1,...,25 \tag{8.45}$$

$$g_{57}(\mathbf{x}) = abs\left(\frac{u_{x1}^{(lc=2)}}{0.00889}\right) - 1 \leq 0$$

$$g_{58}(\mathbf{x}) = abs\left(\frac{u_{y1}^{(lc=2)}}{0.00889}\right) - 1 \leq 0$$

$$g_{59}(\mathbf{x}) = abs\left(\frac{u_{z1}^{(lc=2)}}{0.00889}\right) - 1 \leq 0$$

$$g_{60}(\mathbf{x}) = abs\left(\frac{u_{x2}^{(lc=2)}}{0.00889}\right) - 1 \leq 0$$

$$g_{61}(\mathbf{x}) = abs\left(\frac{u_{y2}^{(lc=2)}}{0.00889}\right) - 1 \leq 0$$

$$g_{62}(\mathbf{x}) = abs\left(\frac{u_{z2}^{(lc=2)}}{0.00889}\right) - 1 \leq 0$$

$$x_i - 100 \leq 0 \quad ; \quad -x_i + 0.06452 \leq 0 \quad ; \quad i = 1,...,8$$

Table 8.59. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with two Load Cases as in (Fleury & Schmit, 1980) and in (Li, Huang, Liu, & Wu, 2007).

25-Bar Truss, 2 Load Cases (Fleury & Schmit, 1980) (Li, Huang, Liu, & Wu, 2007)	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	Mean FEs	[%] FR	[%] FR Initial Tol.	Runs
MS-SN P.AD. PFPR	**247.262219**	**247.305477**	**247.372550**	**247.712946**	**25.00**	**1.95E+05**	**82.22**	**91.49**	**25**
(Fleury & Schmit, 1980)	247.310000	-	-	-	-	-	-	-	-
(Li, Huang, Liu, & Wu, 2007)	247.293822	-	-	-	-	-	-	-	-

Table 8.60. Coordinates of the best solution obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem with two Load Cases as in (Fleury & Schmit, 1980) and in (Li, Huang, Liu, & Wu, 2007).

	Coordinates of best solution							
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
MS-SN P.AD. PFPR	**0.064772**	**12.777407**	**19.371632**	**0.064520**	**0.064520**	**4.419923**	**10.824245**	**17.146269**
(Fleury & Schmit, 1980)	0.064500	12.820000	19.298000	0.064500	0.077400	4.407000	10.833000	17.188000
(Li, Huang, Liu, & Wu, 2007)	0.064516	12.709652	19.458026	0.064516	0.064516	4.477410	10.845140	17.051579

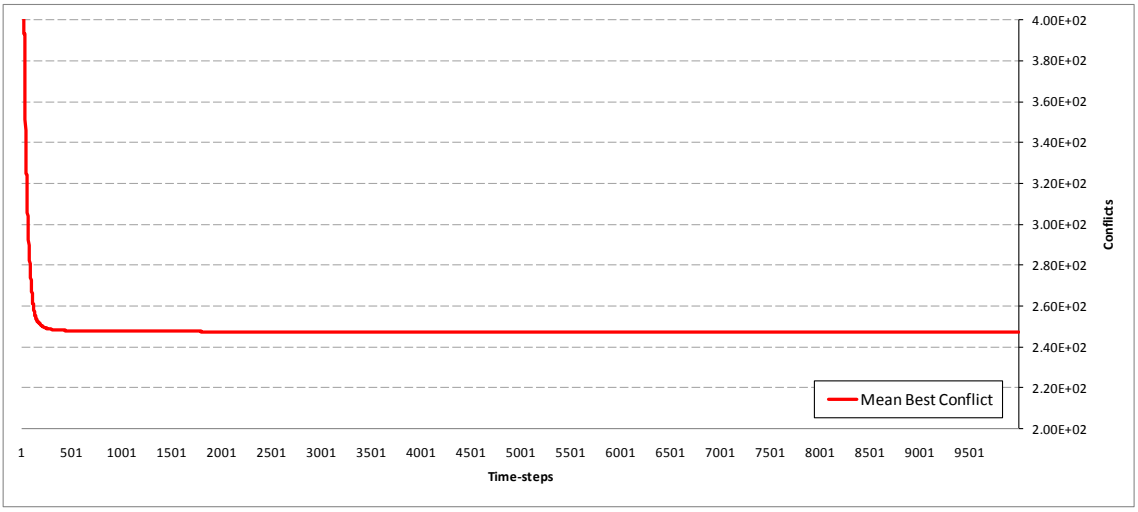


Fig. 8.23. Convergence of the mean best solution for the 25-Bar Truss problem with two Load Cases as in (Fleury & Schmit, 1980) and in (Li, Huang, Liu, & Wu, 2007).

The solution obtained by the MS-SN P.AD. PFPR PSO is better than those reported in (Fleury & Schmit, 1980) and in (Li, Huang, Liu, & Wu, 2007).

The details from the FE analysis of the design obtained by the MS-SN P.Ad. PFPR PSO are provided in Table 8.61 and Table 8.62 for the first Load Case; and in Table 8.63 and Table 8.64 for the second Load Case. The details from the FE analysis of the optimal design reported in (Fleury & Schmit, 1980) are shown in Table 8.65 and Table 8.66 for the first Load Case; and in Table 8.67 and Table 8.68 for the second Load Case.

Table 8.61. Cross sections (areas) and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with two Load Cases as in (Li, Huang, Liu, & Wu, 2007) and in (Fleury & Schmit, 1980), when loaded with the first Load Case (see Table 8.44).

MS-SN P.AD. PFPR - Simultaneous Design - 25-Bar Truss Load Case 1 (mass = **247.262218** kg)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	0.0648	0.06452	**100.39**	2.45E+07	-2.42E+08	2.76E+08	8.90
2	2	12.7774	0.06452	19803.79	-2.09E+07	-7.99E+07	2.76E+08	26.15
	3	12.7774	0.06452	19803.79	-1.79E+07	-7.99E+07	2.76E+08	22.36
	4	12.7774	0.06452	19803.79	1.46E+07	-7.99E+07	2.76E+08	5.30
	5	12.7774	0.06452	19803.79	1.77E+07	-7.99E+07	2.76E+08	6.40
3	6	19.3716	0.06452	30024.23	-2.92E+07	-1.19E+08	2.76E+08	24.46
	7	19.3716	0.06452	30024.23	1.70E+07	-1.19E+08	2.76E+08	6.18
	8	19.3716	0.06452	30024.23	-2.75E+07	-1.19E+08	2.76E+08	23.09
	9	19.3716	0.06452	30024.23	1.87E+07	-1.19E+08	2.76E+08	6.77
4	10	0.0645	0.06452	**100.00**	-1.10E+07	-2.42E+08	2.76E+08	4.55
	11	0.0645	0.06452	**100.00**	-1.38E+07	-2.42E+08	2.76E+08	5.72
5	12	0.0645	0.06452	**100.00**	-2.69E+07	-2.42E+08	2.76E+08	11.11
	13	0.0645	0.06452	**100.00**	7.63E+05	-2.42E+08	2.76E+08	0.28
6	14	4.4199	0.06452	6850.47	-3.44E+07	-4.66E+07	2.76E+08	73.85
	15	4.4199	0.06452	6850.47	2.64E+07	-4.66E+07	2.76E+08	9.55
	16	4.4199	0.06452	6850.47	-3.74E+07	-4.66E+07	2.76E+08	80.28
	17	4.4199	0.06452	6850.47	2.33E+07	-4.66E+07	2.76E+08	8.47
7	18	10.8242	0.06452	16776.57	-2.54E+07	-4.80E+07	2.76E+08	52.93
	19	10.8242	0.06452	16776.57	-2.60E+07	-4.80E+07	2.76E+08	54.22
	20	10.8242	0.06452	16776.57	1.47E+07	-4.80E+07	2.76E+08	5.32
	21	10.8242	0.06452	16776.57	1.40E+07	-4.80E+07	2.76E+08	5.09
8	22	17.1463	0.06452	26575.12	2.83E+07	-7.64E+07	2.76E+08	10.25
	23	17.1463	0.06452	26575.12	-3.39E+07	-7.64E+07	2.76E+08	44.33
	24	17.1463	0.06452	26575.12	-3.82E+07	-7.64E+07	2.76E+08	50.00
	25	17.1463	0.06452	26575.12	2.39E+07	-7.64E+07	2.76E+08	8.68

Table 8.62. Displacement details from the same FE analysis as in Table 8.61.

MS-SN P.AD. PFPR - Simultaneous Design - 25-Bar Truss Load Case 1					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.0166	-0.8890	0.8890	1.87
	uy	0.8890	-0.8890	0.8890	**100.00**
	uz	-0.0577	-0.8890	0.8890	6.49
2	ux	0.0844	-0.8890	0.8890	9.49
	uy	0.8890	-0.8890	0.8890	**100.00**
	uz	-0.0827	-0.8890	0.8890	9.31
3	ux	0.0461	-	-	-
	uy	-0.0969	-	-	-
	uz	-0.3116	-	-	-
4	ux	-0.0281	-	-	-
	uy	-0.1008	-	-	-
	uz	-0.3313	-	-	-
5	ux	0.0100	-	-	-
	uy	-0.0626	-	-	-
	uz	0.2025	-	-	-
6	ux	0.0079	-	-	-
	uy	-0.0665	-	-	-
	uz	0.2222	-	-	-
7	ux		-	-	-
	uy		-	-	-
	uz		-	-	-
8	ux		-	-	-
	uy		-	-	-
	uz		-	-	-
9	ux		-	-	-
	uy		-	-	-
	uz		-	-	-
10	ux		-	-	-
	uy		-	-	-
	uz		-	-	-

Table 8.63. Cross sections (areas) and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem with two Load Cases as in (Li, Huang, Liu, & Wu, 2007) and in (Fleury & Schmit, 1980), when loaded with the second Load Case (see Table 8.49).

MS-SN P.AD. PFPR - Simultaneous Design - 25-Bar Truss Load Case 2 (mass = **247.262218** kg)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	0.0648	0.06452	**100.39**	3.64E+07	-2.42E+08	2.76E+08	13.21
2	2	12.7774	0.06452	19803.79	-4.82E+07	-7.99E+07	2.76E+08	60.37
	3	12.7774	0.06452	19803.79	4.79E+07	-7.99E+07	2.76E+08	17.38
	4	12.7774	0.06452	19803.79	4.79E+07	-7.99E+07	2.76E+08	17.38
	5	12.7774	0.06452	19803.79	-4.82E+07	-7.99E+07	2.76E+08	60.37
3	6	19.3716	0.06452	30024.23	3.34E+07	-1.19E+08	2.76E+08	12.11
	7	19.3716	0.06452	30024.23	-4.55E+07	-1.19E+08	2.76E+08	38.12
	8	19.3716	0.06452	30024.23	-4.55E+07	-1.19E+08	2.76E+08	38.12
	9	19.3716	0.06452	30024.23	3.34E+07	-1.19E+08	2.76E+08	12.11
4	10	0.0645	0.06452	**100.00**	-1.24E+07	-2.42E+08	2.76E+08	5.12
	11	0.0645	0.06452	**100.00**	-1.24E+07	-2.42E+08	2.76E+08	5.12
5	12	0.0645	0.06452	**100.00**	-1.29E+07	-2.42E+08	2.76E+08	5.32
	13	0.0645	0.06452	**100.00**	-1.29E+07	-2.42E+08	2.76E+08	5.32
6	14	4.4199	0.06452	6850.47	-1.81E+07	-4.66E+07	2.76E+08	38.76
	15	4.4199	0.06452	6850.47	7.00E+06	-4.66E+07	2.76E+08	2.54
	16	4.4199	0.06452	6850.47	7.00E+06	-4.66E+07	2.76E+08	2.54
	17	4.4199	0.06452	6850.47	-1.81E+07	-4.66E+07	2.76E+08	38.76
7	18	10.8242	0.06452	16776.57	3.67E+07	-4.80E+07	2.76E+08	13.30
	19	10.8242	0.06452	16776.57	-4.80E+07	-4.80E+07	2.76E+08	**100.00**
	20	10.8242	0.06452	16776.57	-4.80E+07	-4.80E+07	2.76E+08	**100.00**
	21	10.8242	0.06452	16776.57	3.67E+07	-4.80E+07	2.76E+08	13.30
8	22	17.1463	0.06452	26575.12	-8.01E+06	-7.64E+07	2.76E+08	10.48
	23	17.1463	0.06452	26575.12	-1.95E+06	-7.64E+07	2.76E+08	2.55
	24	17.1463	0.06452	26575.12	-8.01E+06	-7.64E+07	2.76E+08	10.48
	25	17.1463	0.06452	26575.12	-1.95E+06	-7.64E+07	2.76E+08	2.55

Table 8.64. Displacement details from the same FE analysis as in Table 8.63.

MS-SN P.AD. PFPR - Simultaneous Design - 25-Bar Truss Load Case 2					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	-0.0503	-0.8890	0.8890	5.66
	uy	0.8890	-0.8890	0.8890	**100.00**
	uz	-0.0735	-0.8890	0.8890	8.26
2	ux	0.0503	-0.8890	0.8890	5.66
	uy	-0.8890	-0.8890	0.8890	**100.00**
	uz	-0.0735	-0.8890	0.8890	8.26
3	ux	0.2836	-	-	-
	uy	-0.1028	-	-	-
	uz	-0.2543	-	-	-
4	ux	0.2480	-	-	-
	uy	0.0686	-	-	-
	uz	0.1453	-	-	-
5	ux	-0.2836	-	-	-
	uy	0.1028	-	-	-
	uz	-0.2543	-	-	-
6	ux	-0.2480	-	-	-
	uy	-0.0686	-	-	-
	uz	0.1453	-	-	-
7	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
8	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
9	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
10	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

Table 8.65. Cross sections (areas) and stresses details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) for the 25-Bar Truss problem with two Load Cases as in (Li, Huang, Liu, & Wu, 2007) and in (Fleury & Schmit, 1980), when loaded with the first Load Case (see Table 8.44).

(Fleury & Schmit, 1980) - 25-Bar Truss Load Case 1 (mass = **247.31** kg)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	0.0645	0.06452	**99.97**	2.46E+07	-2.42E+08	2.76E+08	8.92
2	2	12.8200	0.06452	19869.81	-2.09E+07	-7.99E+07	2.76E+08	26.14
	3	12.8200	0.06452	19869.81	-1.79E+07	-7.99E+07	2.76E+08	22.36
	4	12.8200	0.06452	19869.81	1.46E+07	-7.99E+07	2.76E+08	5.31
	5	12.8200	0.06452	19869.81	1.77E+07	-7.99E+07	2.76E+08	6.40
3	6	19.2980	0.06452	29910.11	-2.93E+07	-1.19E+08	2.76E+08	24.52
	7	19.2980	0.06452	29910.11	1.71E+07	-1.19E+08	2.76E+08	6.19
	8	19.2980	0.06452	29910.11	-2.76E+07	-1.19E+08	2.76E+08	23.15
	9	19.2980	0.06452	29910.11	1.87E+07	-1.19E+08	2.76E+08	6.78
4	10	0.0645	0.06452	**99.97**	-1.10E+07	-2.42E+08	2.76E+08	4.57
	11	0.0645	0.06452	**99.97**	-1.39E+07	-2.42E+08	2.76E+08	5.76
5	12	0.0774	0.06452	119.96	-2.66E+07	-2.42E+08	2.76E+08	11.00
	13	0.0774	0.06452	119.96	6.45E+05	-2.42E+08	2.76E+08	0.23
6	14	4.4070	0.06452	6830.44	-3.45E+07	-4.66E+07	2.76E+08	74.06
	15	4.4070	0.06452	6830.44	2.64E+07	-4.66E+07	2.76E+08	9.58
	16	4.4070	0.06452	6830.44	-3.75E+07	-4.66E+07	2.76E+08	80.52
	17	4.4070	0.06452	6830.44	2.34E+07	-4.66E+07	2.76E+08	8.49
7	18	10.8330	0.06452	16790.14	-2.53E+07	-4.80E+07	2.76E+08	52.75
	19	10.8330	0.06452	16790.14	-2.59E+07	-4.80E+07	2.76E+08	54.04
	20	10.8330	0.06452	16790.14	1.46E+07	-4.80E+07	2.76E+08	5.30
	21	10.8330	0.06452	16790.14	1.40E+07	-4.80E+07	2.76E+08	5.07
8	22	17.1880	0.06452	26639.80	2.82E+07	-7.64E+07	2.76E+08	10.24
	23	17.1880	0.06452	26639.80	-3.38E+07	-7.64E+07	2.76E+08	44.26
	24	17.1880	0.06452	26639.80	-3.81E+07	-7.64E+07	2.76E+08	49.92
	25	17.1880	0.06452	26639.80	2.39E+07	-7.64E+07	2.76E+08	8.67

Table 8.66. Displacement details from the same FE analysis as in Table 8.65.

(Fleury & Schmit, 1980) - 25-Bar Truss Load Case 1					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.0164	-0.8890	0.8890	1.84
	uy	0.8888	-0.8890	0.8890	**99.98**
	uz	-0.0577	-0.8890	0.8890	6.49
2	ux	0.0844	-0.8890	0.8890	9.49
	uy	0.8888	-0.8890	0.8890	**99.98**
	uz	-0.0828	-0.8890	0.8890	9.31
3	ux	0.0457	-	-	-
	uy	-0.0976	-	-	-
	uz	-0.3115	-	-	-
4	ux	-0.0278	-	-	-
	uy	-0.1016	-	-	-
	uz	-0.3312	-	-	-
5	ux	0.0099	-	-	-
	uy	-0.0631	-	-	-
	uz	0.2025	-	-	-
6	ux	0.0081	-	-	-
	uy	-0.0671	-	-	-
	uz	0.2221	-	-	-
7	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
8	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
9	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
10	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

Table 8.67. Cross sections (areas) and stresses details from the FE analysis of the optimal design in (Fleury & Schmit, 1980) for the 25-Bar Truss problem with two Load Cases as in (Li, Huang, Liu, & Wu, 2007) and in (Fleury & Schmit, 1980), when loaded with the second Load Case (see Table 8.49).

(Fleury & Schmit, 1980) - 25-Bar Truss Load Case 2 (mass = **247.31** kg)								
Variables	Bars	Areas [cm^2]	Min. areas [cm^2]	[%] min. areas	Stresses [N/m^2]	Allowable Stresses [N/m^2]		[%] max. stress
1	1	0.0645	0.06452	**99.97**	3.65E+07	-2.42E+08	2.76E+08	13.22
2	2	12.8200	0.06452	19869.81	-4.82E+07	-7.99E+07	2.76E+08	60.26
	3	12.8200	0.06452	19869.81	4.78E+07	-7.99E+07	2.76E+08	17.34
	4	12.8200	0.06452	19869.81	4.78E+07	-7.99E+07	2.76E+08	17.34
	5	12.8200	0.06452	19869.81	-4.82E+07	-7.99E+07	2.76E+08	60.26
3	6	19.2980	0.06452	29910.11	3.35E+07	-1.19E+08	2.76E+08	12.14
	7	19.2980	0.06452	29910.11	-4.56E+07	-1.19E+08	2.76E+08	38.23
	8	19.2980	0.06452	29910.11	-4.56E+07	-1.19E+08	2.76E+08	38.23
	9	19.2980	0.06452	29910.11	3.35E+07	-1.19E+08	2.76E+08	12.14
4	10	0.0645	0.06452	**99.97**	-1.25E+07	-2.42E+08	2.76E+08	5.15
	11	0.0645	0.06452	**99.97**	-1.25E+07	-2.42E+08	2.76E+08	5.15
5	12	0.0774	0.06452	119.96	-1.28E+07	-2.42E+08	2.76E+08	5.30
	13	0.0774	0.06452	119.96	-1.28E+07	-2.42E+08	2.76E+08	5.30
6	14	4.4070	0.06452	6830.44	-1.80E+07	-4.66E+07	2.76E+08	38.72
	15	4.4070	0.06452	6830.44	6.95E+06	-4.66E+07	2.76E+08	2.52
	16	4.4070	0.06452	6830.44	6.95E+06	-4.66E+07	2.76E+08	2.52
	17	4.4070	0.06452	6830.44	-1.80E+07	-4.66E+07	2.76E+08	38.72
7	18	10.8330	0.06452	16790.14	3.66E+07	-4.80E+07	2.76E+08	13.29
	19	10.8330	0.06452	16790.14	-4.79E+07	-4.80E+07	2.76E+08	**99.83**
	20	10.8330	0.06452	16790.14	-4.79E+07	-4.80E+07	2.76E+08	**99.83**
	21	10.8330	0.06452	16790.14	3.66E+07	-4.80E+07	2.76E+08	13.29
8	22	17.1880	0.06452	26639.80	-8.08E+06	-7.64E+07	2.76E+08	10.57
	23	17.1880	0.06452	26639.80	-1.87E+06	-7.64E+07	2.76E+08	2.44
	24	17.1880	0.06452	26639.80	-8.08E+06	-7.64E+07	2.76E+08	10.57
	25	17.1880	0.06452	26639.80	-1.87E+06	-7.64E+07	2.76E+08	2.44

Table 8.68. Displacement details from the same FE analysis as in Table 8.67.

(Fleury & Schmit, 1980) - 25-Bar Truss Load Case 2					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	-0.0504	-0.8890	0.8890	5.67
	uy	0.8887	-0.8890	0.8890	**99.97**
	uz	-0.0735	-0.8890	0.8890	8.27
2	ux	0.0504	-0.8890	0.8890	5.67
	uy	-0.8887	-0.8890	0.8890	**99.97**
	uz	-0.0735	-0.8890	0.8890	8.27
3	ux	0.2834	-	-	-
	uy	-0.1029	-	-	-
	uz	-0.2537	-	-	-
4	ux	0.2480	-	-	-
	uy	0.0685	-	-	-
	uz	0.1448	-	-	-
5	ux	-0.2834	-	-	-
	uy	0.1029	-	-	-
	uz	-0.2537	-	-	-
6	ux	-0.2480	-	-	-
	uy	-0.0685	-	-	-
	uz	0.1448	-	-	-
7	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
8	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
9	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
10	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

A model of the 25-Bar Truss problem in 'Altair Hyperworks' subject to the Load Case 1 can be observed in Fig. 8.24.

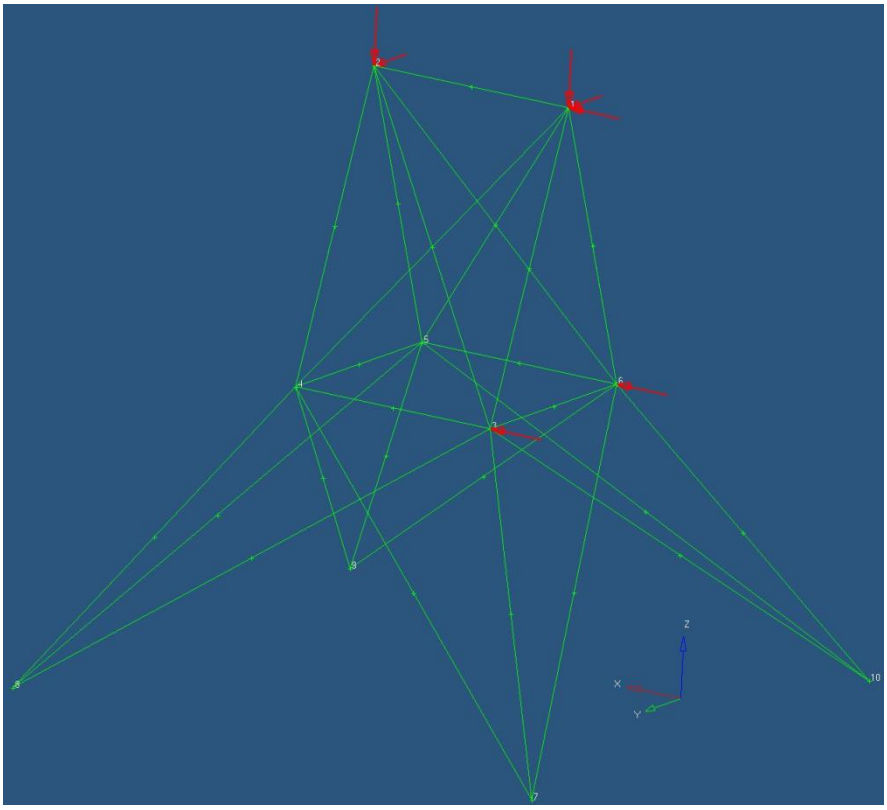


Fig. 8.24. Altair Hyperworks (OptiStruct) model of the 25-bar truss problem with Load Case 1.

It should be noted that every time a reference to other authors' results was made, such results were presented in SI units, performing a conversion of units when necessary.

8.8.6.2. Case 2 (Park & Ryu, 2004)

In this case the allowable compression stress is the same for all members, and the same in magnitude as the allowable tension stress, as shown in Eq. (8.46). The truss is optimized for a single Load Case, which is shown in Table 8.69.

$$\sigma_a = -275800\ \text{kN/m}^2 \quad \text{(allowable compression stress)} \tag{8.46}$$

The solutions obtained are provided in Table 8.70, in Table 8.71, and in Fig. 8.25. The solution reported in (Park & Ryu, 2004) is also presented for reference.

Table 8.69. Load components on the nodes for the 25-Bar Truss problem as in (Park & Ryu, 2004).

NODES	LOAD COMPONENTS		
	x	*y*	*z*
1	4448	44480	-44480
2	0	44480	-44480
3	2224	0	0
6	2668.8	0	0

Table 8.70. Statistical results obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem as formulated in (Park & Ryu, 2004).

25-Bar Truss (Park & Ryu, 2004)	BEST	MEDIAN	MEAN	WORST	[%] Feasible Solutions	Mean FEs	[%] FR	[%] FR Initial Tol.	Runs
MS-SN P.AD. PFPR	**210.657550**	**210.658297**	**210.658706**	**210.663929**	**100.00**	**2.02E+05**	**94.88**	**99.78**	**25**
(Park & Ryu, 2004) (RSA)	230.034647	-	-	-	-	9.20E+03	-	-	-

Table 8.71. Coordinates of the best solution obtained by the 'Multi-Swarm Multi-Sub-neighbourhood Pseudo Adaptive PFPR' PSO for the 25-Bar Truss problem as formulated in (Park & Ryu, 2004).

	Coordinates of best solution							
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
MS-SN P.AD. PFPR	**0.064824**	**0.254690**	**23.479418**	**0.064533**	**12.813289**	**5.016230**	**1.026282**	**25.258462**
(Park & Ryu, 2004) (RSA)	0.645160	0.903224	21.548344	0.645160	5.741924	6.387084	9.677400	19.612864

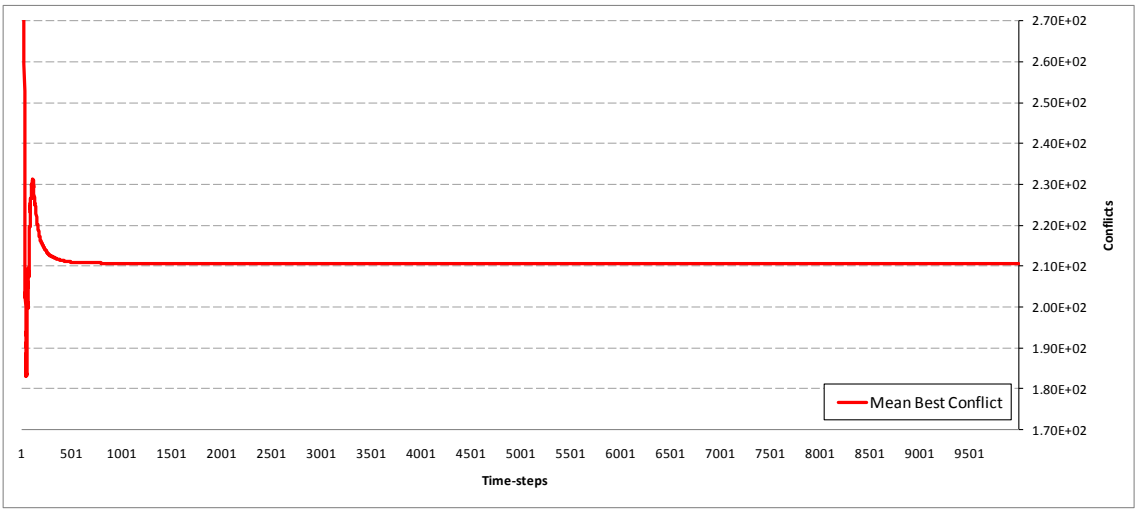


Fig. 8.25. Convergence of the mean best solution for the 25-Bar Truss problem as in (Park & Ryu, 2004).

The details from the FE analysis of the design obtained by the MS-SN P.Ad. PFPR PSO are provided in Table 8.72 and Table 8.73, whereas the details from the FE analysis of the design reported in (Park & Ryu, 2004) are offered in Table 8.74 and Table 8.75.

Table 8.72. Cross sections (areas) and stresses details from the FE analysis of the best design found by the 'MS-SN P.AD. PFPR' PSO for the 25-Bar Truss problem formulated as in (Park & Ryu, 2004).

MS-SN P.AD. PFPR - 25-Bar Truss - Park & Ryu formulation (mass = **210.657550** kg)								
Variables	Bars	Areas [cm²]	Min. areas [cm²]	[%] min. areas	Stresses [N/m²]	Allowable Stresses [N/m²]		[%] max. stress
1	1	0.0648	0.06452	**100.47**	-1.75E+08	-2.76E+08	2.76E+08	63.46
2	2	0.2547	0.06452	394.75	-1.35E+08	-2.76E+08	2.76E+08	49.00
	3	0.2547	0.06452	394.75	1.60E+07	-2.76E+08	2.76E+08	5.80
	4	0.2547	0.06452	394.75	-9.12E+07	-2.76E+08	2.76E+08	33.08
	5	0.2547	0.06452	394.75	6.15E+07	-2.76E+08	2.76E+08	22.30
3	6	23.4794	0.06452	36390.91	-3.72E+07	-2.76E+08	2.76E+08	13.50
	7	23.4794	0.06452	36390.91	1.63E+07	-2.76E+08	2.76E+08	5.92
	8	23.4794	0.06452	36390.91	-3.59E+07	-2.76E+08	2.76E+08	13.01
	9	23.4794	0.06452	36390.91	1.77E+07	-2.76E+08	2.76E+08	6.41
4	10	0.0645	0.06452	**100.02**	-1.68E+07	-2.76E+08	2.76E+08	6.11
	11	0.0645	0.06452	**100.02**	-2.12E+07	-2.76E+08	2.76E+08	7.68
5	12	12.8133	0.06452	19859.41	-3.74E+07	-2.76E+08	2.76E+08	13.57
	13	12.8133	0.06452	19859.41	1.61E+07	-2.76E+08	2.76E+08	5.83
6	14	5.0162	0.06452	7774.69	-3.53E+07	-2.76E+08	2.76E+08	12.79
	15	5.0162	0.06452	7774.69	1.83E+07	-2.76E+08	2.76E+08	6.63
	16	5.0162	0.06452	7774.69	-3.79E+07	-2.76E+08	2.76E+08	13.74
	17	5.0162	0.06452	7774.69	1.56E+07	-2.76E+08	2.76E+08	5.67
7	18	1.0263	0.06452	1590.64	-2.53E+07	-2.76E+08	2.76E+08	9.16
	19	1.0263	0.06452	1590.64	-3.19E+07	-2.76E+08	2.76E+08	11.56
	20	1.0263	0.06452	1590.64	1.80E+07	-2.76E+08	2.76E+08	6.51
	21	1.0263	0.06452	1590.64	7.55E+06	-2.76E+08	2.76E+08	2.74
8	22	25.2585	0.06452	39148.27	1.83E+07	-2.76E+08	2.76E+08	6.62
	23	25.2585	0.06452	39148.27	-3.54E+07	-2.76E+08	2.76E+08	12.84
	24	25.2585	0.06452	39148.27	-3.83E+07	-2.76E+08	2.76E+08	13.90
	25	25.2585	0.06452	39148.27	1.52E+07	-2.76E+08	2.76E+08	5.51

Table 8.73. Displacement details from the same FE analysis as in Table 8.72.

MS-SN P.AD. PFPR - 25-Bar Truss - Park & Ryu formulation					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.8889	-0.8890	0.8890	**99.99**
	uy	0.8707	-0.8890	0.8890	97.95
	uz	-0.1287	-0.8890	0.8890	14.48
2	ux	0.4054	-0.8890	0.8890	45.60
	uy	0.8890	-0.8890	0.8890	**100.00**
	uz	-0.1173	-0.8890	0.8890	13.20
3	ux	0.0765	-	-	-
	uy	-0.0970	-	-	-
	uz	-0.3408	-	-	-
4	ux	-0.0269	-	-	-
	uy	-0.1032	-	-	-
	uz	-0.3330	-	-	-
5	ux	0.0587	-	-	-
	uy	-0.0447	-	-	-
	uz	0.1642	-	-	-
6	ux	0.0143	-	-	-
	uy	-0.0505	-	-	-
	uz	0.1425	-	-	-
7	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
8	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
9	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
10	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

Table 8.74. Cross sections (areas) and stresses details from the FE analysis of the optimal design in (Park & Ryu, 2004) for the 25-Bar Truss problem formulated as in (Park & Ryu, 2004).

Park & Ryu - 25-Bar Truss - Park & Ryu formulation (mass = **210.657550** kg)								
Variables	Bars	Areas [cm²]	Min. areas [cm²]	[%] min. areas	Stresses [N/m²]	Allowable Stresses [N/m²]		[%] max. stress
1	1	0.6452	0.06452	999.94	-1.60E+07	-2.76E+08	2.76E+08	5.81
2	2	0.9032	0.06452	1399.91	-5.60E+07	-2.76E+08	2.76E+08	20.31
	3	0.9032	0.06452	1399.91	-1.32E+07	-2.76E+08	2.76E+08	4.77
	4	0.9032	0.06452	1399.91	-9.77E+06	-2.76E+08	2.76E+08	3.54
	5	0.9032	0.06452	1399.91	3.31E+07	-2.76E+08	2.76E+08	11.99
3	6	21.5483	0.06452	33397.93	-4.00E+07	-2.76E+08	2.76E+08	14.49
	7	21.5483	0.06452	33397.93	1.72E+07	-2.76E+08	2.76E+08	6.25
	8	21.5483	0.06452	33397.93	-3.85E+07	-2.76E+08	2.76E+08	13.96
	9	21.5483	0.06452	33397.93	1.87E+07	-2.76E+08	2.76E+08	6.78
4	10	0.6452	0.06452	999.94	-6.87E+06	-2.76E+08	2.76E+08	2.49
	11	0.6452	0.06452	999.94	-6.89E+06	-2.76E+08	2.76E+08	2.50
5	12	5.7419	0.06452	8899.45	-3.43E+07	-2.76E+08	2.76E+08	12.44
	13	5.7419	0.06452	8899.45	1.36E+07	-2.76E+08	2.76E+08	4.94
6	14	6.3871	0.06452	9899.39	-2.72E+07	-2.76E+08	2.76E+08	9.87
	15	6.3871	0.06452	9899.39	1.48E+07	-2.76E+08	2.76E+08	5.37
	16	6.3871	0.06452	9899.39	-2.93E+07	-2.76E+08	2.76E+08	10.63
	17	6.3871	0.06452	9899.39	1.27E+07	-2.76E+08	2.76E+08	4.61
7	18	9.6774	0.06452	14999.07	-2.83E+07	-2.76E+08	2.76E+08	10.25
	19	9.6774	0.06452	14999.07	-2.90E+07	-2.76E+08	2.76E+08	10.51
	20	9.6774	0.06452	14999.07	1.31E+07	-2.76E+08	2.76E+08	4.75
	21	9.6774	0.06452	14999.07	1.20E+07	-2.76E+08	2.76E+08	4.35
8	22	19.6129	0.06452	30398.12	1.93E+07	-2.76E+08	2.76E+08	7.01
	23	19.6129	0.06452	30398.12	-3.64E+07	-2.76E+08	2.76E+08	13.20
	24	19.6129	0.06452	30398.12	-4.02E+07	-2.76E+08	2.76E+08	14.57
	25	19.6129	0.06452	30398.12	1.54E+07	-2.76E+08	2.76E+08	5.58

Table 8.75. Displacement details from the same FE analysis as in Table 8.74.

Park & Ryu - 25-Bar Truss - Park & Ryu formulation					
Nodes	Displacement (u) [cm]		Max. Displacement [cm]		[%] max u
1	ux	0.2332	-0.8890	0.8890	26.23
	uy	0.8858	-0.8890	0.8890	**99.64**
	uz	-0.1146	-0.8890	0.8890	12.89
2	ux	0.1889	-0.8890	0.8890	21.25
	uy	0.8871	-0.8890	0.8890	**99.79**
	uz	-0.1347	-0.8890	0.8890	15.16
3	ux	0.0557	-	-	-
	uy	-0.0451	-	-	-
	uz	-0.3019	-	-	-
4	ux	-0.0391	-	-	-
	uy	-0.0454	-	-	-
	uz	-0.3165	-	-	-
5	ux	0.0286	-	-	-
	uy	-0.0264	-	-	-
	uz	0.1354	-	-	-
6	ux	-0.0090	-	-	-
	uy	-0.0262	-	-	-
	uz	0.1488	-	-	-
7	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
8	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
9	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-
10	ux	0.0000	-	-	-
	uy	0.0000	-	-	-
	uz	0.0000	-	-	-

As shown in Table 8.70, the optimal design returned by the MS-SN P.Ad. PFPR PSO is considerably –and suspiciously– better than the one reported in (Park & Ryu, 2004). The details from the FE analyses for both designs (see Table 8.72 to Table 8.75) show that both of them comply with all constraints stated in their formulations in (Park & Ryu, 2004).

One reason to explain the noticeable difference in the mass of the optimal design may be that they set a greater value for the minimum cross section, which is not clear in their work. Notice that the minimum cross section in their optimal design is ten times greater than in the design obtained here (refer to Table 8.71). Another reason may be that they may have considered buckling constraints in their optimization process, although they do not specify so. Literally, they state (...) *The constraints are imposed on the allowable stresses of the truss members and the allowable maximum displacements at the joints* (Park & Ryu, 2004). While the buckling constraint is indirectly included in the values of the allowable compression stress in (Fleury & Schmit, 1980) by means of Euler's critical load, the magnitudes of the allowable compression and tensions stresses are the same in (Park & Ryu, 2004). Nevertheless, both designs in Table 8.71 were tested on a FE analysis incorporating buckling (not included here). The results show that the design in (Park & Ryu, 2004) comply with the buckling constraints whereas the design obtained here violates the buckling constraints for bars '2' and '4'. Therefore, the considerable difference in the masses for the optimal designs obtained may be due to their consideration of additional buckling constraints not specified in (Park & Ryu, 2004).

8.9. Closure

A pseudo adaptive constraint-handling technique was proposed and extensively tested in this chapter. The technique is said to be pseudo adaptive because, in spite of presenting adaptive features, it still requires some parameters to be set by the user, such as the percentage of individual experiences that need to be feasible for a tolerance update to take place 'adaptively'.

The pseudo adaptive scheme starts by self-tuning the initial relaxation of the tolerances aiming for a target initial feasibility ratio of the problem, and then adaptively decreases

its value until the final, desired tolerance is reached. This scheme was successfully coupled with a Multi-Swarm PSO optimizer with three sub-neighbourhoods and a dynamic forward topology taken from previous chapters, and with two different constraint-handling techniques (CHTs): the preserving feasibility with priority rules (PFPR) and the penalization method (PM). The performances of both CHTs were improved by the pseudo adaptive scheme when testing the algorithm on a set of 13 benchmark constrained problems taken from the literature. Especially, but not only, when facing highly constrained problems with low feasibility ratios.

The algorithm with the pseudo adaptive PFPR technique was also tested on a number of engineering problems typically used in the literature to assess the performance of optimization algorithms. The optimizer proved itself capable of dealing with a wide variety of problems with reasonably good performance and without any kind of tuning. Recall that this algorithm was previously tested on a set of hard benchmark unconstrained problems in previous chapters.

SECTION III

CONCLUSIONS

Chapter 9

CONCLUSIONS

A summary of the main achievements, findings, and contributions throughout this thesis is presented, followed by potential lines for future research to continue this work.

9.1. Contributions

The objective of this thesis was to develop a fully working particle swarm optimizer for real-valued –exceptionally discrete or mixed-discrete– hard constrained and unconstrained optimization problems. It was aimed at a general-purpose algorithm in the sense that it would be suitable to cope with a wide range of problems with reasonably good performance, without requiring much adaptation or tuning.

While a practical algorithm was pursued, some mathematical framework was desirable to support the developments from the bottom up. Therefore an extensive study of the influence of the settings of the coefficients in the velocity update equation –as well as those of the velocity constraint– on the behaviour of an isolated particle pulled by stationary attractors were carried out, partly theoretically, partly heuristically, and partly by simply visualizing trajectories.

Although there were previous outstanding works on the trajectories of the deterministic particle and on the influence of the coefficients' settings on the dynamics of the swarm, they are typically hard-to-follow and therefore their use becomes limited. A somewhat rigorous yet more accessible study of the convergence was carried out in chapter 5, providing insight into the trajectory of the particle that goes beyond the speed of convergence and the settings of three coefficients.

Choosing the desired average behaviour and controlling the strength of randomness (chapter 6) add even more flexibility into the algorithm, and provides more control over the behaviour of a stochastic method like particle swarm optimization. In fact, it was observed that reducing the range of ϕ for the same pair 'ϕ_{mean}–w' remarkably speeds up

convergence regardless of the magnitudes of the corresponding roots of the characteristic polynomial. Thus a convenient reformulation of the update equations was proposed so that both the average behaviour and the strength of randomness can be controlled by the settings of the coefficients. This reformulation encompasses both the classical and the constricted PSO. Chapter 6 was concluded with useful guidelines as to how to choose the coefficients in order to obtain the behaviour desired.

It was also shown that –rather counter-intuitively– setting the individuality weight to a value perceptively higher than that of the sociality weight may not be effective in delaying convergence and improving exploration. In fact, it might turn out to be harmful for the exploration abilities of the algorithm. Conversely, handling the settings of the coefficients in the velocity update equation comprises an effective means to control the form and speed of convergence.

It was concluded that setting the velocity constraint to half the feasible interval is, at worst, harmless, and it is therefore advisable. Smaller values appear over-restrictive, as they excessively narrow the search space and disrupt the normal dynamics of the swarm. The extensive study of the coefficients carried out allows controlling the trajectories by means of their settings instead of by an external, disruptive $v_{\max}$ constraint.

Aiming for an optimizer that performs reasonably well on a wide range of different problems without adaptations or tuning, some robust coefficients' settings and combination of settings –as well as robust dynamic neighbourhood topologies– were proposed and tested on a set of unconstrained benchmark problems. This algorithm coupled with a proposed pseudo adaptive constraint-handling mechanism resulted in a robust particle swarm optimizer which can cope with a wide variety of hard unconstrained and constrained problems, such as discontinuous, multimodal, or implicit problems, or simply those with extremely low feasibility ratios. This optimizer was tested on a number of problems without any sort of tuning, exhibiting good performance.

In addition to the fully working robust particle swarm optimizer, the pseudo adaptive constraint-handling mechanism proposed is suitable for use on any other evolutionary algorithm. In turn, the sub-neighbourhoods techniques may be useful for multi-solution search, while the nearest neighbourhood procedure may be convenient for problems where the actual distance travelled by a particle matters. That is, in problems translated

into the physical space such as in *swarm robotics*. Thus, the transfer of information throughout the swarm is guaranteed thanks to a given topological neighbourhood, while the nearest neighbour procedure forms neighbourhoods with some physical meaning.

9.2. Future research

Even regarding topics effectively addressed within this thesis, there are aspects that still require further research and testing. For instance, it has been extensively studied how to choose the coefficients so as to obtain a given desired behaviour, while perhaps it is more important to address how to choose the desired behaviour. In addition to that, stochastic convergence studies have not been formally addressed.

With regards to the neighbourhoods, there are some classical topologies such as the 'von Neumann' and the 'stochastic star' structures which were not tested. Furthermore, even the experiments performed were carried out on a small benchmark, which in addition is now known to be biased. The next natural step is to confirm the findings and conclusions resulting from this thesis on new, extended and unbiased benchmarks. After all, the algorithm is expected to work well on both biased and unbiased problems. In addition, there are some failures to achieve a given (demanding) success condition in the experiments performed that need to be investigated further.

The study of the coefficients, neighbourhood and constraint-handling were necessarily incomplete. Only three coefficients' settings and a combination of them were extensively tested numerically, and on a small set of problems. Even in the proposed pseudo adaptive constraint-handling mechanism, there are parameters involved whose sensitivity needs to be investigated. For instance, the initial feasibility ratio, the maximum time allowed between updates, or the minimum percentage of individual best experiences required to trigger an update.

Finally, there are important aspects of the particle swarm optimization method that were not studied within this thesis, such as the swarm-size and the types of initialization.

References and Bibliography

Abraham, A., Liu, H., & Chang, T.-G. (2006). Variable Neighborhood Particle Swarm Optimization Algorithm. *GECCO 2006.*

Akat, S. B., & Gazi, V. (2008). Particle Swarm Optimization with Dynamic Neighborhood Topology: Three Neighborhood Strategies and Preliminary Results. *2008 IEEE Swarm Intelligence Symposium*, (pp. 21-23).

Alves Da Silva, A. P., & Falcão, D. M. (2008). Fundamentals of Genetic Algorithms. In K. Y. Lee, & M. A. El-Sharkawi, *Modern Heuristic Optimization Techniques - Theory and Applications to Power Systems* (pp. 25-42). New Jersey: John Wiley & Sons Inc.

Angeline, P. J. (1998). Evolutionary Optimization versus Particle Swarm Optimization: Philosophy and Performance Differences. *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, (pp. 601-610).

Auger, A., Blackwell, T., Bratton, D., Clerc, M., Croussette, S., Dattasharma, A., et al. (2007). *Standard PSO*. Retrieved 2010, from Particle Swarm Central: http://www.particleswarm.info/Programs.html#Standard_PSO_2007

Azid, I. A., Kwan, A. S., & Seetharamu, K. N. (2002). An evolutionary approach for layout optimization of a three-dimensional truss. *Structural and Multidisciplinary Optimization* (24), 333–337.

Bäck, T. (1996). Evolution Strategies: An Alternative Evolutionary Algorithm. In J. -M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, & D. Snyers, *Artificial Evolution - Lecture Notes in Computer Science.* Springer.

Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice.* Oxford University Press.

Bäck, T., Fogel, D. B., & Michalewicz, Z. (2000). *Evolutionary Computation 1 - Basic Algorithms and Operators.* Philadelphia: Institute of Physics Publishing Ltd.

Bäck, T., Hoffmeister, F., & Schwefel, H.-P. (1991). A Survey of Evolution Strategies. *Proceedings of the Fourth International Conference on Genetic Algorithms* (pp. 2-9). San Mateo, CA: Morgan Kaufmann.

Bäck, T., & Schwefel, H.-P. (1996). Evolutionary Computation: An Overview. *International Conference on Evolutionary Computation*, (pp. 20-29).

Beekman, M., Sword, G. A., & Simpson, S. J. (2008). Biological Foundations of Swarm Intelligence. In C. Blum, & D. Merkle, *Swarm Intelligence: Introduction and Applications* (pp. 3-41). Springer-Verlag.

Ben Ghalia, M. (2008). Particle Swarm Optimization with an improved exploration-exploitation balance. *51st Midwest Symposium on Circuits and Systems*, (pp. 759-762).

Beni, G. (2005). From Swarm Intelligence to Swarm Robotics. In E. Sahin, & W. (. Spears, *Proceedings of the First International Workshop on Swarm Robotics (at SAB 2004)* (Vol. Lecture Notes on Computer Science Vol. 3342, pp. 1-9). Berlin: Springer.

Beni, G., & Wang, J. (1989). Swarm Intelligence. *Proceedings of the Seventh Annual Meeting of the Robotics Society of Japan*, (pp. 425-428). Tokyo.

Beyer, H. -G. (1998). On the "explorative power" of ES/EP-like algorithms. In V. W. Porto, N. Saravanan, D. Waagen, & A. E. Eiben, *Proceedings of the 7th Annual Conference on Evolutionary Programming.*

Blum, C., & Li, X. (2008). Swarm Intelligence in Optimization. In C. Blum, & D. Merkle, *Swarm Intelligence: Introduction and Applications* (pp. 43-85). Springer-Verlag.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems.* Oxford University Press.

Bratton, D., & Kennedy, J. (2007). Defining a Standard for Particle Swarm Optimization. *Proceedings of the 2007 IEEE Swarm Intelligence Symposium.*

Burton, A. (2004). Truss Optimization Using Genetic Algorithms. *GECCO'04.* Washington.

Carlisle, A., & Dozier, G. (2001). An Off-The-Shelf PSO. *Proceedings of the Workshop on Particle Swarm Optimization*, (pp. 1-6). Indianapolis.

Chen, X.-H., Lee, W.-P., Liao, C.-Y., & Dai, J.-T. (2007). Adaptive Constriction Factor for Location-related Particle Swarm. *Proceedings of the 8th WSEAS International Conference on Evolutionary Computing.* Vancouver.

Clerc, M. (1999). The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization. *Proceedings of the IEEE Congress on Evolutionary Computation, Volume 3*, pp. 1951-1957.

Clerc, M. (2004). Discrete Particle Swarm Optimization, illustrated by the Traveling Salesman Problem. In G. C. Onwubolu, & B. V. Babu, *New Optimization Techniques in Engineering (Studies in Fuzziness and Soft Computing)* (pp. 219-238). Springer-Verlag.

Clerc, M. (2006a). *Particle Swarm Optimization.* Iste.

Clerc, M. (2006b). *Stagnation analysis in particle swarm optimization or what happens when nothing happens*. Retrieved 2010, from http://hal.archives-ouvertes.fr/hal-00122031

Clerc, M. (2007). *Confinements and Biases in Particle Swarm Optimisation*. Retrieved 2010, from HAL: hal-00122799, version 1: http://hal.archives-ouvertes.fr/hal-00122799

Clerc, M. (2008a). *The mythical balance, or When Particle Swarm Optimisation does not exploit*. Retrieved 2010, from http://clerc.maurice.free.fr/pso/Balanced_PSO/Balanced_PSO.pdf

Clerc, M. (2008b). Why does it work? *International Journal of Computational Intelligence Research , 4*, 79-91.

Clerc, M. (2008c). *Initialisations for Particle Swarm Optimisation*. Retrieved 2010, from http://clerc.maurice.free.fr/pso/Initialisations.pdf

Clerc, M., & Kennedy, J. (2002). The Particle Swarm—Explosion, Stability, and Convergence in a Multidimensional Complex Space. *IEEE Transactions on Evolutionary Computation, Vol. 6, No. 1* , 58-73.

Coello Coello, C. A. (1999). A comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems , Vol. 1* (No. 3), 269-308.

Coello Coello, C. A. (1999). Constraint-handling through a multiobjective optimization technique. *Proceedings of the 1999 genetic and evolutionary computation conference, workshop program,* (pp. 117-8). Orlando.

Coello Coello, C. A. (2000). Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry , Vol. 41,* 113-127.

Cooren, Y., Clerc, M., & Siarry, P. (2009). Performance evaluation of TRIBES, an adaptive particle swarm optimization algorithm. In *Swarm Intelligence* (pp. 149-178).

Cui, Z., & Zeng, J. (2004). A Guaranteed Global Convergence Particle Swarm Optimizer. In *Rough Sets and Current Trends in Computing* (pp. 762-767).

Darwin, C. (1859). *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life.* London: John Murray, Albemarle Street.

Das, S., Abraham, A., Chakraborty, U. K., & Konar, A. (2009). Differential evolution using a neighborhood-based mutation operator. *IEEE Transactions on Evolutionary Computation , Vol. 13* (Issue 3), 526-553.

Davis, L. D., & Mitchell, M. (1991). *Handbook of Genetic Algorithms.* Van Nostrand Reinhold.

de Freitas Vaz, A. I., & da Graça Pinto Fernandes, E. M. (2006). Optimization of Nonlinear Constrained Particle Swarm. *Technological and Economic Development of Economy , Vol. XII* (No. 1), 30-36.

De Jong, K. A. (2006). *EVOLUTIONARY COMPUTATION - A unified approach.* Massachusetts Institute of Technology.

DeBao, C., & ChunXia, Z. (2009). Particle swarm optimization with adaptive population size and its application. *Applied Soft Computing , 9* (1), 39-48.

Diosan, L., & Oltean, M. (2008). What else is the evolution of PSO telling us? *Journal of Artificial Evolution and Applications* .

Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms (Ph.D. Thesis in Italian).* Italy: Dipartimento di Elettronica, Politecnico di Milano.

Dorigo, M., & Gambardella, L. (1997). Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation , Vol.1* (Nr.1).

Dorigo, M., & Sahin, E. (2004). Special Issue: Swarm Robotics. *Autonomous Robots , 17,* 111-113.

Dorigo, M., & Stützle, T. (2004). *Ant Colony Optimization.* The MIT Press.

Eberhart, R. C., & Kennedy, J. (1995). A New Optimizer Using Particle Swarm Theory. *Sixth International Symposium on Micro Machine and Human Science* (pp. 39-43). Nagoya (Japan): IEEE Service Center.

Engelbrecht, A. P. (2005). *Fundamentals of Computational Swarm Intelligence.* John Wiley & Sons Ltd.

Evers, G. I. (2009). *An Automatic Regrouping Mechanism to deal with Stagnation in Particle Swarm Optimization (Master Thesis).* University of Texas-Pan American.

Farmani, R., & Wright, J. A. (2003). Self-Adaptive Fitness Formulation for Constrained Optimization. *IEEE Transactions on Evolutionary Computation , Vol. 7* (No. 5), 445-455.

Ferreira, C. (2006). *Gene Expression Programming: Mathematical Modelling by an Artificial Intelligence* (2nd ed.). Berlin: Springer-Verlag.

Fleury, C., & Schmit, L. A. (1980). *Dual Methods and Approximation Concepts in Structural Synthesis.* NASA CR-3226.

Fogel, D. B. (1992). *Evolving Artificial Intelligence (Ph.D. Thesis).* San Diego: University of California.

Fogel, D. B. (1995). *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* Piscataway, NJ: IEEE Press Series on Computational Intelligence.

Fogel, D. B. (1995). Phenotypes, Genotypes, and Operators in Evolutionary Computation. *Proceedings of the Second IEEE International Conference on Evolutionary Computation. IEEE.*

Fogel, D. B. (1998). *Evolutionary Computation: A Fossil Record.* Piscataway, NJ: IEEE Press.

Fogel, D. B. (2008). Introduction to Evolutionary Computation. In K. Y. Lee, & M. A. El-Sharkawi, *Modern Heuristic Optimization Techniques* (pp. 3-23). New Jersey: John Wiley & Sons.

Fogel, L., Angeline, P., & Fogel, D. (1995). An Evolutionary Programming Approach to Self-Adaptation on Finite State Machines. In J. McDonnel, R. Reynolds, & D. Fogel, *Proceeding of the 4th Annual Conference on Evolutionary Programming.* MIT Press.

Fogel, L., Owens, A., & Walsh, M. (1966). *Artificial Intelligence through Simulated Evolution.* New York: John Wiley & Sons.

Foryś, P., & Bochenek, B. (2004). A New Particle Swarm Optimizer and its Application to Structural Optimization. *Proceedings of the 5th ASMO UK / ISSMO Conference on Engineering Design Optimization.* Stratford upon Avon.

Friedberg, R. (1959). A learning machine: Part 1. *IBM Research Journal 3(7)* , 282-287.

Fuentes Cabrera, J. C., & Coello Coello, C. A. (2007). Handling Constraints in Particle Swarm Optimization Using a Small Population Size. In S. B. Heidelberg, *MICAI 2007: Advances in Artificial Intelligence* (pp. 41-51).

Gen, M., & Cheng, R. (2000). *Genetic Algorithms & Engineering Optimization.* John Wiley & Sons, Inc.

Ghasemi, M. R. (1996). *Structural Optimization of Trusses and Axisymmetric Shells using Gradient-Based Methods and Genetic Algorithms (Ph.D. thesis).* University of Wales Swansea.

Ghasemi, M. R., Hinton, E., & Wood, R. D. (1999). Optimization of trusses using genetic algorithms for discrete and continuous variables. *Engineering Computations , 16* (3), 272-301.

Haftka, R. T., & Gürdal, Z. (1992). *Elements of Structural Optimization.* Kluwer Academic Publishers.

Hamida, S. B., & Schoenauer, M. (2002). ASCHEA: New Results Using Adaptive Segregational Constraint Handling. *Proceedings of the Congress on Evolutionary Computation 2002 (CEC'2002), Vol. 1*, pp. 884-889. Piscataway, New Jersey.

Haupt, R. L., & Haupt, S. E. (2004). *PRACTICAL GENETIC ALGORITHMS* (2nd ed.). John Wiley & Sons, Inc.

He, Q., & Wang, L. (2007). A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. *Applied Mathematics and Computation , Vol. 186*, 1407-1422.

Helwig, S., & Wanka, R. (2007). Particle Swarm Optimization in High-Dimensional Bounded Search Spaces. *IEEE Swarm Intelligence Symposium (SIS 2007).*

Helwig, S., & Wanka, R. (2008). Theoretical Analysis on Initial Particle Swarm Behaviour. *10th International Conference on Parallel Problem Solving from Nature (PPSN 2008)* (pp. 889-898). Springer.

Heppner, F., & Grenander, U. (1990). A stochastic nonlinear model for coordinated bird flocks. In S. Krasner, *The Ubiquity of Chaos* (pp. 233-238). Washington, DC: AAAS Publications.

Hernández Aguirre, A., Villa Diharce, E., & Coello Coello, C. A. (2007). Constraint Handling Techniques for a Non-parametric Real-valued Estimation Distribution Algorithm. *IEEE Congress on Evolutionary Computation 2007 (CEC 2007)*, (pp. 654-661).

Herrera, F., Lozano, M., & Verdegay, J. L. (1998). Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. *Artificial Intelligence Review* (12), 265–319.

Holland, J. (1962). Outline for a logical theory of adaptive systems. *JACM 9* , 297-314.

Holland, J. (1967). Nonlinear environments permitting efficient adaptation. In *Computer and Information Sciences II.* Academic Press.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press.

Hu, X., & Eberhart, R. C. (2002). Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization. *Proceedings of the 6th World Multi-conference on Systemics, Cybernetics and Informatics.* Orlando.

Hu, X., Eberhart, R. C., & Shi, Y. (2003). Engineering Optimization with Particle Swarm. *Proceedings of the IEEE Swarm Intelligence Symposium*, (pp. 53-57). Indianapolis.

Huang, V. L., Qin, A. K., & Suganthan, P. N. (2006). Self-adaptive Differential Evolution Algorithm for Constrained Real-Parameter Optimization. *IEEE Congress on Evolutionary Computation*, (pp. 17-24). Vancouver, BC, Canada.

IJCIR. (2008). Special Issue: Particle Swarm Optimization. *4* (2).

Innocente, M. S. (2006). Population-Based Methods: Particle Swarm Optimization - Development of a General-Purpose Optimizer and Applications. *Master's Thesis* , 1-476.

Innocente, M. S., & Sienz, J. (2006). Particle Swarm Optimization: Development of a General-Purpose Optimizer. *6th ASMO UK / ISSMO conference.* Oxford.

Innocente, M. S., & Sienz, J. (2007). A Study of the Fundamental Parameters of Particle Swarm Optimizers. *Proceedings of the 7th World Congress on Structural and Multidisciplinary Optimization*, (pp. 735-744). Seoul.

Innocente, M. S., & Sienz, J. (2008). Constraint-Handling Techniques for Particle Swarm Optimization Algorithms. *Proceedings of the 6th ASMO UK / ISSMO Conference on Engineering Design Optimization*, (pp. 203-223). Bath.

Jiang, M., Luo, Y. P., & Yang, S. Y. (2007). Stochastic convergence anaysis and parameter selection of the standard particle swarm optimization algorithm. *Information Processing Letters* (102), 8-16.

Johnsonbaugh, R. (1999). *Matemáticas discretas* (4th ed.). Prentice Hall Hispanoamericana.

Kennedy, J. (1998). Methods of Agreement: Inference Among the EleMentals. *Proceedings of the 1998 IEEE ISIC/CIRA/ISAS Joint Conference*, (pp. 883-887). Gaithersburg.

Kennedy, J. (1999). Small Worlds and Mega-Minds: Effect of Neighbourhood Topology on Particle Swarm Performance. *Proceedings of the IEEE Congress on Evolutionary Computation, Vol.3*, pp. 1931-1938.

Kennedy, J. (2008). How it works: Collaborative Trial and Error. *International Journal of Computational Intelligence Research , 4*, 71-78.

Kennedy, J., & Eberhart, R. C. (1995). Particle Swarm Optimization. *Proceedings of the IEEE International Conference on Neural Networks*, (pp. 1942-1948). Piscataway.

Kennedy, J., & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. *Proceedings of the Conference on Systems, Man, and Cybernetics*, (pp. 4104-4109). Piscataway.

Kennedy, J., & Eberhart, R. C. (2001). *Swarm Intelligence.* Morgan Kaufmann Publishers.

Kennedy, J., & Mendes, R. (2006). Neighbourhood Topologies in Fully-Informed and Best-of-Neighbourhood Particle Swarms. *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews , Vol.6* (No.4), 515-519.

Korte, B., & Vygen, J. (2006). *Combinatorial Optimization - Theory and Algorithms* (3rd ed.). Springer-Verlag.

Koza, J. R. (1990). *Genetic Programming: A Paradigm for Genetically Breeding Populations of Computer Programs to Solve Problems.* Stanford University Computer Science Department Technical Report STAN-CS-90-1314.

Koza, J. R. (1992). *Genetic Programming.* Cambridge: The MIT Press.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Natural Selection.* Cambridge, MA, USA: MIT Press.

Koza, J. R. (1993). *Genetic Programming: On the Programming of Computers by Means of Natural Selection V.1.* MIT Press.

Koza, J. R. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and Computing* , 87-112.

Koza, J. R. (1994). Genetic Programming as a means for programming computers by natural selection. *Statistics and Computing , 4* (2), 87-112.

Koza, J. R., & Poli, R. (2005). Genetic Programming. In E. K. Burke, & G. Kendall, *Search Methodologies - Introductory tutorials in Optimization and Decision Support Techniques* (pp. 127-164). New York: Springer Science+Business Media, Inc.

Koziel, S., & Michalewicz, Z. (1999). Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization. *Evolutionary Computation , Vol. 7* (No. 1), 19-44.

Landa Becerra, R., & Coello Coello, C. A. (2005). Optimization with Constraints using a Cultured Differential Evolution Approach. *Proceedings of the 2005 Genetic and Evolutionary Computation Conference (GECCO'05).* Washington, DC.

Langdon, W., & Poli, R. (2005). Evolving problems to learn about particle swarm and other optimizers. *Congress on Evolutionary Computation*, (pp. 81-88).

Li, L. J., Huang, Z. B., Liu, F., & Wu, Q. H. (2007). A heuristic particle swarm optimizer for optimization of pin connected structures. *Computers and Structures* (85), 340-349.

Li, X. (2004). Adaptively Choosing Neighbourhood Bests Using Species in Particle Swarm Optimizer for Multimodal Function Optimization. *GECCO 2004.*

Liang, J. J., & Suganthan, P. N. (2006). Dynamic Multi-Swarm Particle Swarm Optimizer with a Novel Constraint-Handling Mechanism. *2006 IEEE Congress on Evolutionary Computation*, (pp. 9-16). Vancouver.

Liang, J. J., Runarsson, T. P., Mezura-Montes, E., Clerc, M., Suganthan, P. N., Coello Coello, C. A., et al. (2006, September 18). *Problem Definitions and Evaluation Criteria for the CEC 2006 Special Session on Constrained Real-Parameter Optimization*. Retrieved October 14, 2009, from Special Session on Constrained Real-Parameter Optimization, CEC-06, Vancouver, Canada, 17-21 July: http://www3.ntu.edu.sg/home/EPNSugan/

Mendes, R. (2004). *Population Topologies and Their Influence in Particle Swarm Performance (Ph.D. Thesis).* Universidade do Minho.

Mendes, R., Kennedy, J., & Neves, J. (2004). The Fully Informed Particle Swarm: Simpler, Maybe Better. *IEEE Transactions on Evolutionary Computation , Vol.8* (No.3), 204-210.

Mezura-Montes, E., Velázquez-Reyes, J., & Coello Coello, C. A. (2006). Modified Differential Evolution for Constrained Optimization. *IEEE Congress on Evolutionary Computation*, (pp. 25-32). Vancouver, BC, Canada.

Michalewicz, Z., & Fogel, D. B. (2004). *How to solve it: Modern Heuristics* (2nd ed.). Berlin: Springer-Verlag.

Miranda, V., Keko, H., & Duque, A. J. (2008). Stochastic Star Communication Topology in Evolutionary Particle Swarms (EPSO). *International Journal of Computational Intelligence Research* .

Mitchell, M. (1999). *An Introduction to Genetic Algorithms.* The MIT Press.

Mohais, A. (2007). *Random Dynamic Neighbourhood Structures in Particle Swarm Optimisation (Ph.D. Thesis).* University of the West Indies.

Mohan, C. K., & Al-Kazemi, B. (2001). Discrete particle swarm optimization. *Proceedings of the Workshop on Particle Swarm Optimization.* Indianapolis: IUPUI (in press).

Monson, C. K., & Seppi, K. D. (2005). Exposing Origin-Seeking Bias in PSO. *GECCO'05*, (pp. 241-248).

Morrison, R. W. (2004). *Designing Evolutionary Algorithms for Dynamic Environments.* Springer-Verlag.

Muñoz Zavala, Á. E., Hernández Aguirre, A., & Villa Diharce, E. R. (2005). Constrained Optimization via Particle Evolutionary Swarm Optimization Algorithm (PESO). *Proceedings of the 2005 Genetic and Evolutionary Computation Conference (GECCO'05)*, (pp. 209-216). Washington, DC.

Muñoz Zavala, Á. E., Hernández Aguirre, A., Villa Diharce, E. R., & Botello Rionda, S. (2006). PESO+ for Constrained Optimization. *2006 IEEE Congress on Evolutionary Computation (CEC '06)*, (pp. 231-238). Vancouver.

Naudts, B., & Schippers, A. (1999). *A Motivated Definition of Exploitation and Exploration (Technical Report).*

Nocedal, J., & Wright, S. J. (2006). *Numerical Optimization* (2nd ed.). Springer Science+Business Media, LLC.

Novo Sanjurjo, V. (1999). *Teoría de la Optimización* (2nd ed.). Madrid: Universidad Nacional de Educación a Distancia.

Ozcan, E., & Mohan, C. K. (1998). Analysis of a simple particle swarm optimization system. In *Intelligent Engineering Systems Through Artificial Neural Networks: Volume 8* (pp. 253-258). ASME books.

Ozcan, E., & Mohan, C. K. (1999). Particle Swarm Optimization: Surfing the Waves. *Proceedings of the IEEE Congress on Evolutionary Computation*, (pp. 1939-1944). Washington, DC.

Park, J., & Ryu, M. (2004). Optimal Design of Truss Structures by Rescaled Simulated Annealing. *KSME International Journal , 18* (9), 1512-1518.

Parsopoulos, K. E., & Vrahatis, M. N. (2002). Particle Swarm Optimization Method for Constrained Optimization Problems. *Proceedings of the Euro-International Symposium on Computational Intelligence.*

Pedregal, P. (2004). *Introduction to Optimization.* New York: Springer-Verlag.

Poli, R. (2008). Dynamics and Stability of the Sampling Distribution of Particle Swarm Optimizers via Moment Analysis. *Journal of Artificial Evolution and Applications* .

Price, K. V., Storn, R. M., & Lampinen, J. A. (2005). *Differential Evolution - A Practical Approach to Global Optimization.* Berlin: Springer-Verlag.

Rechenberg, I. (1965). Cybernetic solution path of an experimental problem. In *Library translation No. 1122.* Farnborough, Hants, UK: Royal Aircraft Establishment.

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Stuttgart: Frommann-Holzboog.

Reynolds, C. W. (1987). Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics , Vol. 21* (No. 4), 25-34.

Richards, M., & Ventura, D. (2003). Dynamic Sociometry in Particle Swarm Optimization. *International Conference on Computational Intelligence and Natural Computing.*

Runarsson, T. P., & Yao, X. (2000). Stochastic Ranking for Constrained Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation , Vol. 4* (No. 3), 284-294.

Sahin, E. (2005). Swarm robotics: From sources of inspiration to domains of application. In E. Sahin, & W. (. Spears, *Proceedings of the First Intenational Workshop on Swarm Robotics (at SAB 2004)* (Vol. Lecture Notes in Computer Science Vol. 3342, pp. 10-20).

Sahin, E., Girgin, S., Bayindir, L., & Turgut, A. E. (2008). Swarm Robotics. In C. Blum, & D. Merkle, *Swarm Intelligence: Introduction and Applications* (pp. 87-100). Springer-Verlag.

Sastry, K., Goldberg, D., & Kendall, G. (2005). Genetic Algorithms. In E. K. Burke, & G. Kendall, *Search Methodologies - Introductory tutorials in Optimization and Decision Support Techniques* (pp. 97-125). Springer Science+Business Media, Inc.

Schwefel, H.-P. (1975). *Evolutionsstrategie und numerishe Optimierung (Ph.D. Thesis).* Technical University of Berlin.

Schwefel, H.-P. (1981). *Numerical Optimization of Computer Models.* New York: John Wiley & Sons, Inc.

Schwefel, H.-P. (1994). On the Evolution of Evolutionary Computation. In J. Zurada, R. Marks II, & C. Robinson (Eds.), *Computational Intelligence: Imitating Life.* Piscataway: IEEE Press.

Shang, Y.-W., & Qiu, Y.-H. (2006). A Note on the Extended Rosenbrock Function. *Evolutionary Computation , 14*, 119-126.

Shi, Y., & Eberhart, R. C. (1998a). A modified particle swarm optimizer. *Proceedings of the IEEE International Conference on Evolutionary Computation*, (pp. 69-73). Piscataway.

Shi, Y., & Eberhart, R. C. (1998b). Parameter Selection in Particle Swarm Optimization. *The Seventh Annual Conference on Evolutionary Programming*, (pp. 591-600).

Shi, Y., & Eberhart, R. C. (1999). Empirical Study of Particle Swarm Optimization. *Proceedings of the IEEE Congress on Evolutionary Computation. Vol.3*, pp. 1945-1950. IEEE Press.

Sienz, J., & Innocente, M. S. (2008). Particle Swarm Optimization: Fundamental Study and its Application to Optimization and to Jetty Scheduling Problems. In M. Papadrakakis, & B. H. Topping, *Trends in Engineering Computational Technology* (pp. 103-126). Stirlingshire: Saxe-Coburg Publications.

Sienz, J., & Innocente, M. S. (2010). Individual and Social Behaviour in Particle Swarm Optimizers. In B. H. Topping, J. N. Adam, F. J. Pallarés, R. Bru, & M. L. Romero, *Developments and Applications in Engineering Computational Technology* (Vol. 26, pp. 219-243). Stirlingshire, UK: Saxe-Coburg Publications.

Storn, R., & Price, K. (1995). *Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces (Technical Report).* Technical Report TR-95-012, ICSI.

Storn, R., & Price, K. (1996). Minimizing the real functions of the ICEC'96 constest by differential evolution. *Proceedings of the IEEE International Conference on Evolutionary Computation* (pp. 842-844). Nagoya, Japan: IEEE Press.

Suganthan, P. N. (1999). Particle Swarm Optimiser with Neighbourhood Operator. *Proceedings of the 1999 Congress on Evolutionary Computation (CEC 99)*, (pp. 1958-1962).

Takahama, T., & Sakai, S. (2004). Constrained Optimization by α Constrained Genetic Algorithm. *Systems and Computers in Japan , Vol. 35* (No. 5).

Takahama, T., & Sakai, S. (2005). Constrained Optimization by Applying the α Constrained Method to the Nonlinear Simplex Method with Mutations. *IEEE Transactions on Evolutionary Computation , Vol. 9* (No. 5), 437-451.

Takahama, T., & Sakai, S. (2006a). Solving Constrained Optimization Problems by the ε Constrained Particle Swarm Optimizer with Adaptive Velocity Limit Control. *Proceedings of the International Conference on Computational Intelligence and Security (CIS).* Guangzhou.

Takahama, T., & Sakai, S. (2006b). Constrained Optimization by the ε Constrained Differential Evolution with Gradient-Based Mutation and Feasible Elites. *IEEE Congress on Evolutionary Computation*, (pp. 1-7). Vancouver, BC, Canada.

Takahama, T., Sakai, S., & Iwane, N. (2006). Solving Nonlinear Constrained Optimization Problems by the ε Constrained Differential Evolution. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics.* Taipei.

Toscano Pulido, G., & Coello Coello, C. A. (2004). A Constraint-Handling Mechanism for Particle Swarm Optimization. *Proceedings of the IEEE Congress on Evolutionary Computation, Vol. 2*, pp. 1396-1403. Portland.

Trelea, I. C. (2003). The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters 85* , 317-325.

Trianni, V., Nolfi, S., & Dorigo, M. (2008). Evolution, Self-organization and Swarm Robotics. In C. Blum, & D. Merkle, *Swarm Intelligence: Introduction and Applications* (pp. 163-191). Springer-Verlag.

van den Bergh, F. (2001). *An Analysis of Particle Swarm Optimizers (Ph.D. Thesis).* Pretoria: University of Pretoria.

van den Bergh, F., & Engelbrecht, A. P. (2001). Effects of swarm size on cooperative particle swarm optimisers. *Proceedings of the Genetic and Evolutionary Computation Conference 2001*.

Venter, G., & Haftka, R. T. (2008). Constrained Particle Swarm Optimisation Using a Multi-Objective Formulation. In M. Papadrakakis, & B. H. Topping (Ed.), *Proceedings of the 6th International Conference on Engineering Computational Technology*. Athens: Civil-Comp Press.

Wang, J., & Yin, Z. (2008). A ranking selection-based particle swarm optimizer for engineering design optimization problems. *Structural and Multidisciplinary Optimization , Vol. 37*, 131-147.

Wolpert, D. H., & Macready, W. G. (1995). *No Free Lunch Theorems for Search.* Santa Fe Institute.

Wolpert, D. H., & Macready, W. G. (1997). No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation , 1* (1), 67-82.

Wolsey, L. A. (1998). *Integer Programming.* John Wiley & Sons, Inc.

Worasucheep, C. (2008). Solving Constrained Engineering Optimization Problems by the Constrained PSO-DD. *Proceedings of the ECTI-CON*, (pp. 5-8). Krabi.

Wright, S. (1932). The roles of mutation, inbreeding, crossbreeding, and selection in evolution. *Proceedings of the 6th International Congress on Genetics*, *Vol. 1*, pp. 356-366.

Xie, X.-F., Zhang, W.-J., & Bi, D.-C. (2004). Handling Equality Constraints by Adaptive Relaxing Rule for Swarm Algorithms. *Congress on Evolutionary Computation*, (pp. 2012-2016). Oregon.

XPS. (2004-2007). *eXtended Particle Swarm Project*. Retrieved 2010, from http://xps-swarm.essex.ac.uk/

Yu, X., Liu, J., & Li, H. (2009). An Adaptive Inertia Weight Particle Swarm Optimization Algorithm for IIR Digital Filter. *2009 International Conference on Artificial Intelligence and Computational Intelligence.*

Zheng, J., Wu, Q., & Song, W. (2007). An Improved Particle Swarm Algorithm for Solving Nonlinear Constrained Optimization Problems. *Proceedings of the Third International Conference on Natural Computation (ICNC 2007).* Haikou, Hainan, China.

Zielinski, K., & Laur, R. (2006). Constrained Single-Objective Optimization Using Particle Swarm Optimization. *2006 IEEE Congress on Evolutionary Computation*, (pp. 443-450). Vancouver.

SECTION IV

APPENDICES

Appendix I

AUXILIARY ARITHMETICS

Some auxiliary, complementary and cumbersome arithmetic work supporting Chapter 5 is provided in this section to facilitate the understanding of the formulae provided. Borrowing the term from French Mathematician Maurice Clerc (Clerc, Particle Swarm Optimization, 2006a), this section is for *amateurs*.

AI.1. Deterministic particle's position equation

By *deterministic particle* it is referred to the single particle studied in Chapter 5, with constant coefficient ϕ and stationary attractors. Somewhat tedious and burdensome arithmetical work is presented here so as to confirm that the equation derived for the deterministic particle's position is indeed the solution of the recurrence relation. Thus, considering the first two positions $x^{(0)}$ and $x^{(1)}$ as the initial conditions –recall that $v^{(0)}$ can be directly obtained from $x^{(1)}$–, the same $x^{(2)}$ is calculated from the recurrence relation and from the positions' equation. Also, the initial conditions are obtained from replacing the initial two time-steps in the positions' equation. Both the recurrence relation and the position's equation are re-written hereafter for convenience. The former is as shown in Eq. (AI.1), while the latter is given by Eqs. (AI.2) to (AI.4).

$$\boxed{x^{(t)} + (\phi - w - 1)\cdot x^{(t-1)} + w\cdot x^{(t-2)} = \phi\cdot p} \tag{AI.1}$$

$$\boxed{x^{(t)} = p + \frac{r_2\cdot\left(p - x^{(0)}\right) - \left(p - x^{(1)}\right)}{\gamma}\cdot r_1^t + \frac{-r_1\cdot\left(p - x^{(0)}\right) + \left(p - x^{(1)}\right)}{\gamma}\cdot r_2^t} \tag{AI.2}$$

where

$$\gamma = \sqrt{\phi^2 - (2\cdot w + 2)\cdot\phi + (w-1)^2} \tag{AI.3}$$

$$r_1 = \frac{(1+w)}{2} - \frac{\phi}{2} + \frac{\gamma}{2} \quad ; \quad r_2 = \frac{(1+w)}{2} - \frac{\phi}{2} - \frac{\gamma}{2} \tag{AI.4}$$

AI.1.1. Initial conditions

By replacing $t = 0$ and $t = 1$ in Eq. (AI.2), the initial conditions $x^{(0)}$ and $x^{(1)}$ must be obtained. Thus, for $t = 0$:

$$x^{(t=0)} = p + \frac{r_2 \cdot \left(p - x^{(0)}\right) - \left(p - x^{(1)}\right)}{\gamma} + \frac{-r_1 \cdot \left(p - x^{(0)}\right) + \left(p - x^{(1)}\right)}{\gamma} \tag{AI.5}$$

$$x^{(t=0)} = p + \frac{r_2 \cdot \left(p - x^{(0)}\right)}{\gamma} - \frac{\left(p - x^{(1)}\right)}{\gamma} - \frac{r_1 \cdot \left(p - x^{(0)}\right)}{\gamma} + \frac{\left(p - x^{(1)}\right)}{\gamma} \tag{AI.6}$$

From Eq. (AI.4), $r_1 - r_2 = \gamma$. Hence,

$$x^{(t=0)} = p + \frac{r_2 \cdot \left(p - x^{(0)}\right)}{\gamma} - \frac{(\gamma + r_2) \cdot \left(p - x^{(0)}\right)}{\gamma} \tag{AI.7}$$

$$\boxed{x^{(t=0)} = x^{(0)}} \tag{AI.8}$$

In turn, for $t = 1$:

$$x^{(t=1)} = p + \frac{r_2 \cdot \left(p - x^{(0)}\right) - \left(p - x^{(1)}\right)}{\gamma} \cdot r_1 + \frac{-r_1 \cdot \left(p - x^{(0)}\right) + \left(p - x^{(1)}\right)}{\gamma} \cdot r_2 \tag{AI.9}$$

$$x^{(t=1)} = p + \frac{r_2 \cdot \left(p - x^{(0)}\right)}{\gamma} \cdot r_1 - \frac{\left(p - x^{(1)}\right)}{\gamma} \cdot r_1 - \frac{r_1 \cdot \left(p - x^{(0)}\right)}{\gamma} \cdot r_2 + \frac{\left(p - x^{(1)}\right)}{\gamma} \cdot r_2 \tag{AI.10}$$

$$x^{(t=1)} = p - \frac{\left(p - x^{(1)}\right)}{\gamma} \cdot r_1 + \frac{\left(p - x^{(1)}\right)}{\gamma} \cdot r_2 \tag{AI.11}$$

$$x^{(t=1)} = p + \frac{\left(p - x^{(1)}\right)}{\gamma} \cdot \left(-r_1 + r_2\right) \tag{AI.12}$$

$$\boxed{x^{(t=1)} = x^{(1)}} \tag{AI.13}$$

Thus the initial conditions are confirmed. It is more laborious to derive $x^{(2)}$ from the recurrence relation in Eq. (AI.1) and from the particle's position equation in Eq. (AI.2).

AI.1.2. First position derived from initial conditions

From the recurrence relation in Eq. (AI.1),

$$x^{(2)} + (\phi - w - 1)\cdot x^{(1)} + w\cdot x^{(0)} = \phi\cdot p \tag{AI.14}$$

$$x^{(2)} = -(\phi - w - 1)\cdot x^{(1)} - w\cdot x^{(0)} + \phi\cdot p \tag{AI.15}$$

$$\boxed{x^{(2)} = (1 + w - \phi)\cdot x^{(1)} - w\cdot x^{(0)} + \phi\cdot p} \tag{AI.16}$$

From the particle's position equation in Eq. (AI.2),

$$x^{(t=2)} = p + \frac{r_2\cdot\left(p - x^{(0)}\right) - \left(p - x^{(1)}\right)}{\gamma}\cdot r_1^2 + \frac{-r_1\cdot\left(p - x^{(0)}\right) + \left(p - x^{(1)}\right)}{\gamma}\cdot r_2^2 \tag{AI.17}$$

$$\gamma\cdot x^{(2)} = \gamma\cdot p + r_2\cdot\left(p - x^{(0)}\right)\cdot r_1^2 - \left(p - x^{(1)}\right)\cdot r_1^2 - r_1\cdot\left(p - x^{(0)}\right)\cdot r_2^2 + \left(p - x^{(1)}\right)\cdot r_2^2 \tag{AI.18}$$

$$\gamma\cdot x^{(2)} = \left(r_1^2 - r_2^2\right)\cdot x^{(1)} + \left(r_1\cdot r_2^2 - r_1^2\cdot r_2\right)\cdot x^{(0)} + \left(\gamma + r_1^2\cdot r_2 - r_1^2 - r_1\cdot r_2^2 + r_2^2\right)\cdot p \tag{AI.19}$$

$$\boxed{\gamma\cdot x^{(2)} = \left(r_1^2 - r_2^2\right)\cdot x^{(1)} + \left(r_1\cdot r_2^2 - r_1^2\cdot r_2\right)\cdot x^{(0)} + \left(\gamma - \left(r_1^2 - r_2^2\right) - \left(r_1\cdot r_2^2 - r_1^2\cdot r_2\right)\right)\cdot p} \tag{AI.20}$$

In order to make the calculations neater, the coefficients of $x^{(1)}$, $x^{(0)}$ and p in Eq. (AI.20) will be considered separately. From Eq. (AI.4), $r_1 - r_2 = \gamma$. Therefore,

$$r_2^2 = r_1^2 - 2\cdot r_1\cdot\gamma + \gamma^2 \tag{AI.21}$$

$$r_1^2 - r_2^2 = 2\cdot r_1\cdot\gamma - \gamma^2 \tag{AI.22}$$

Replacing r_1 from Eq. (AI.4) yields

$$r_1^2 - r_2^2 = 2\cdot\left(\frac{1 + w - \phi + \gamma}{2}\right)\cdot\gamma - \gamma^2 = \gamma + w\cdot\gamma - \phi\cdot\gamma \tag{AI.23}$$

$$\boxed{r_1^2 - r_2^2 = \gamma\cdot(1 + w - \phi)} \tag{AI.24}$$

In turn,

$$r_1^2 = \frac{(1+w)^2 - (1+w)\cdot\phi + (1+w)\cdot\gamma - (1+w)\cdot\phi + \phi^2 - \phi\cdot\gamma + (1+w)\cdot\gamma - \phi\cdot\gamma + \gamma^2}{4} \quad \text{(AI.25)}$$

$$r_1^2 = \frac{1}{4}\cdot\left((1+w)^2 + \phi^2 + \gamma^2 - 2\cdot(1+w)\cdot\phi + 2\cdot(1+w)\cdot\gamma - 2\cdot\phi\cdot\gamma\right) \quad \text{(AI.26)}$$

Likewise,

$$r_2^2 = \frac{1}{4}\cdot\left((1+w)^2 + \phi^2 + \gamma^2 - 2\cdot(1+w)\cdot\phi - 2\cdot(1+w)\cdot\gamma + 2\cdot\phi\cdot\gamma\right) \quad \text{(AI.27)}$$

Note that Eq. (AI.24) can also be derived from Eqs. (AI.26) and (AI.27).

Thus,

$$\left(r_1\cdot r_2^2\right) = \frac{1}{8}\cdot\begin{pmatrix} (1+w)^3 + (1+w)\cdot\phi^2 + (1+w)\cdot\gamma^2 - 2\cdot(1+w)^2\cdot\phi - 2\cdot(1+w)^2\cdot\gamma + \\ +2\cdot(1+w)\cdot\phi\cdot\gamma + \\ -(1+w)^2\cdot\phi - \phi^3 - \phi\cdot\gamma^2 + 2\cdot(1+w)\cdot\phi^2 + 2\cdot(1+w)\cdot\phi\cdot\gamma - 2\cdot\phi^2\cdot\gamma + \\ +(1+w)^2\cdot\gamma + \phi^2\cdot\gamma + \gamma^3 - 2\cdot(1+w)\cdot\phi\cdot\gamma - 2\cdot(1+w)\cdot\gamma^2 + 2\cdot\phi\cdot\gamma^2 \end{pmatrix} \quad \text{(AI.28)}$$

Operating arithmetically,

$$\left(r_1\cdot r_2^2\right) = \frac{1}{8}\cdot\begin{pmatrix} (1+w)^3 - \phi^3 + \gamma^3 + 3\cdot(1+w)\cdot\phi^2 - (1+w)\cdot\gamma^2 - 3\cdot(1+w)^2\cdot\phi + \\ -(1+w)^2\cdot\gamma + 2\cdot(1+w)\cdot\phi\cdot\gamma + \phi\cdot\gamma^2 - \phi^2\cdot\gamma \end{pmatrix} \quad \text{(AI.29)}$$

Likewise,

$$\left(r_1^2\cdot r_2\right) = \frac{1}{8}\cdot\begin{pmatrix} (1+w)^3 - \phi^3 - \gamma^3 + 3\cdot(1+w)\cdot\phi^2 - (1+w)\cdot\gamma^2 - 3\cdot(1+w)^2\cdot\phi + \\ +(1+w)^2\cdot\gamma - 2\cdot(1+w)\cdot\phi\cdot\gamma + \phi\cdot\gamma^2 + \phi^2\cdot\gamma \end{pmatrix} \quad \text{(AI.30)}$$

Hence, from Eqs. (AI.29) and (AI.30),

$$\left(r_1\cdot r_2^2 - r_1^2\cdot r_2\right) = \frac{1}{8}\cdot\left(2\cdot\gamma^3 - 2\cdot(1+w)^2\cdot\gamma + 4\cdot(1+w)\cdot\phi\cdot\gamma - 2\cdot\phi^2\cdot\gamma\right) \quad \text{(AI.31)}$$

$$\left(r_1\cdot r_2^2 - r_1^2\cdot r_2\right) = \frac{\gamma}{2}\cdot\left(\frac{\gamma^2}{2} - \frac{(1+w)^2}{2} + (1+w)\cdot\phi - \frac{\phi^2}{2}\right) \quad \text{(AI.32)}$$

Replacing γ^2 from Eq. (AI.3),

$$\left(r_1 \cdot r_2^2 - r_1^2 \cdot r_2\right) = \frac{\gamma}{2} \cdot \left(\frac{\phi^2 - (2 \cdot w + 2) \cdot \phi + (w-1)^2}{2} - \frac{(1+w)^2}{2} + (1+w) \cdot \phi - \frac{\phi^2}{2} \right) \tag{AI.33}$$

$$\left(r_1 \cdot r_2^2 - r_1^2 \cdot r_2\right) = \frac{\gamma}{2} \cdot \left(\frac{\phi^2 - (2 \cdot w + 2) \cdot \phi - 2 \cdot w - 2 \cdot w}{2} + (1+w) \cdot \phi - \frac{\phi^2}{2} \right) \tag{AI.34}$$

$$\left(r_1 \cdot r_2^2 - r_1^2 \cdot r_2\right) = \frac{\gamma}{2} \cdot \left(\frac{\phi^2}{2} - (w+1) \cdot \phi - 2 \cdot w + (1+w) \cdot \phi - \frac{\phi^2}{2} \right) \tag{AI.35}$$

$$\boxed{\left(r_1 \cdot r_2^2 - r_1^2 \cdot r_2\right) = -\gamma \cdot w} \tag{AI.36}$$

Introducing Eqs. (AI.24) and (AI.36) into Eq. (AI.20),

$$\gamma \cdot x^{(2)} = \gamma \cdot (1 + w - \phi) \cdot x^{(1)} + (-\gamma \cdot w) \cdot x^{(0)} + \left(\gamma - \gamma \cdot (1 + w - \phi) - (-\gamma \cdot w)\right) \cdot p \tag{AI.37}$$

$$\gamma \cdot x^{(2)} = \gamma \cdot (1 + w - \phi) \cdot x^{(1)} - \gamma \cdot w \cdot x^{(0)} + (\gamma - \gamma - \gamma \cdot w + \gamma \cdot \phi + \gamma \cdot w) \cdot p \tag{AI.38}$$

$$\boxed{x^{(2)} = (1 + w - \phi) \cdot x^{(1)} - w \cdot x^{(0)} + \phi \cdot p} \tag{AI.39}$$

As can be observed, Eqs. (AI.16) and (AI.39) are the same, *quod erat demonstrandum*. This confirms that the equation derived for the position of the deterministic particle with stationary attractors is indeed Eq. (AI.2).

AI.2. Region in 'ϕ–w' plane for complex roots

The boundaries of the region of the 'ϕ–w' plane where the roots of the characteristic polynomial are complex are obtained by setting $\gamma = 0$. Thus, from Eq. (AI.3):

$$\phi^2 - (2 \cdot w + 2) \cdot \phi + (w-1)^2 = \phi^2 - 2 \cdot w \cdot \phi - 2 \cdot \phi + w^2 - 2 \cdot w + 1 = 0 \tag{AI.40}$$

$$\Rightarrow \qquad w^2 - 2 \cdot (\phi + 1) \cdot w + (\phi - 1)^2 = 0 \tag{AI.41}$$

$$w = (\phi + 1) \pm 2 \cdot \sqrt{\phi} \tag{AI.42}$$

Since the complex conjugate roots occur for $w^2 - 2 \cdot (\phi + 1) \cdot w + (\phi - 1)^2 < 0$, the complex region is defined as in Eq. (AI.43), where the signs of the inequalities are illustrated in Fig. AI.1 for the particular case of ϕ = 4. For the curves bounding the region with complex roots, refer to section 5.2.3.2 in chapter 5.

$$\boxed{(\phi + 1) - 2 \cdot \sqrt{\phi} < w < (\phi + 1) + 2 \cdot \sqrt{\phi}} \tag{AI.43}$$

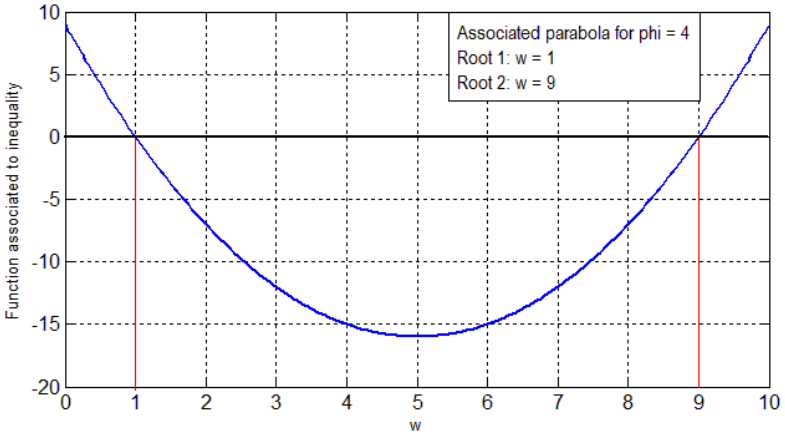


Fig. AI.1. Example of the parabola associated to Eq. (AI.41) for ϕ = 4. The interval where the function is smaller than zero (complex roots in the 'ϕ–w' plane) is between the roots of the parabola.

AI.3. Diagonalization of the system matrix *M*

This section is complementary to section 5.3 in chapter 5. Given a system matrix M of size n (n = 2 in this case) and n linearly independent families of eigenvectors associated to the eigenvalues of M, there exists a matrix A such that

$$A^{-1} \cdot M \cdot A = D = \begin{pmatrix} e_1 & 0 \\ 0 & e_2 \end{pmatrix} \tag{AI.44}$$

where each of the n columns of matrix A comprises one of the eigenvalues.

Let us call ev_j to the jth eigenvector, so that

$$A = \begin{pmatrix} ev_1 & \dots & ev_j & \dots & ev_n \end{pmatrix} \tag{AI.45}$$

Therefore, and calling A_j the jth column of matrix A,

$$\begin{aligned} & M \cdot A = \begin{pmatrix} M \cdot A_1 & \dots & M \cdot A_j & \dots & M \cdot A_n \end{pmatrix} \\ \Rightarrow & M \cdot A = \begin{pmatrix} M \cdot ev_1 & \dots & M \cdot ev_j & \dots & M \cdot ev_n \end{pmatrix} \\ \Rightarrow & M \cdot A = \begin{pmatrix} e_1 \cdot ev_1 & \dots & e_j \cdot ev_j & \dots & e_n \cdot ev_n \end{pmatrix} \end{aligned} \tag{AI.46}$$

Introducing A^{-1},

$$\begin{aligned} & A^{-1} \cdot M \cdot A = A^{-1} \cdot \begin{pmatrix} e_1 \cdot ev_1 & \dots & e_j \cdot ev_j & \dots & e_n \cdot ev_n \end{pmatrix} \\ \Rightarrow & A^{-1} \cdot M \cdot A = \begin{pmatrix} e_1 \cdot A^{-1} \cdot ev_1 & \dots & e_j \cdot A^{-1} \cdot ev_j & \dots & e_n \cdot A^{-1} \cdot ev_n \end{pmatrix} \\ \Rightarrow & A^{-1} \cdot M \cdot A = \begin{pmatrix} e_1 \cdot A^{-1} \cdot A_1 & \dots & e_j \cdot A^{-1} \cdot A_j & \dots & e_n \cdot A^{-1} \cdot A_n \end{pmatrix} \end{aligned} \tag{AI.47}$$

Expanding Eq. (AI.47) for $n = 2$, the diagonalization of the system matrix M in terms of its eigenvalues can be explicitly observed:

$$A^{-1} \cdot M \cdot A = \begin{pmatrix} e_1 & 0 \\ 0 & e_2 \end{pmatrix} \tag{AI.48}$$

Thus, in the case dealt with in chapter 5,

$$M = \begin{pmatrix} w & \phi \\ -w & (1-\phi) \end{pmatrix} \quad ; \quad ev_i = \begin{pmatrix} v \\ y \end{pmatrix} \quad ; \quad e_i = \frac{1+w}{2} - \frac{\phi}{2} \pm \frac{\gamma}{2} \tag{AI.49}$$

Given that $M \cdot ev_i = e_i \cdot ev_i$ and from Eq. (AI.49),

$$\begin{aligned} & -w \cdot v + (1-\phi) \cdot y = \left(\frac{1+w}{2} - \frac{\phi}{2} + \frac{\gamma}{2} \right) \cdot y \\ \Rightarrow & v = \frac{1-w-\phi-\gamma}{2 \cdot w} \cdot y \end{aligned} \tag{AI.50}$$

$$\Rightarrow ev_1 = \begin{pmatrix} \left(\dfrac{1-w-\phi-\gamma}{2\cdot w}\right)\cdot y \\ y \end{pmatrix} \tag{AI.51}$$

Likewise,

$$\begin{aligned} -w\cdot v+(1-\phi)\cdot y &= \left(\frac{1+w}{2}-\frac{\phi}{2}-\frac{\gamma}{2}\right)\cdot y \\ \Rightarrow v &= \frac{1-w-\phi+\gamma}{2\cdot w}\cdot y \end{aligned} \tag{AI.52}$$

$$\Rightarrow ev_2 = \begin{pmatrix} \left(\dfrac{1-w-\phi+\gamma}{2\cdot w}\right)\cdot y \\ y \end{pmatrix} \tag{AI.53}$$

Hence, for $y = 2\cdot w$, the matrix *A* used for the diagonalization of *M* is as in Eq. (AI.54):

$$\boxed{A = \begin{pmatrix} (1-w-\phi-\gamma) & (1-w-\phi+\gamma) \\ 2\cdot w & 2\cdot w \end{pmatrix}} \tag{AI.54}$$

Appendix II

BENCHMARK TEST PROBLEMS

The mathematical formulations of two sets of benchmark optimization problems are offered in this appendix. The first one is composed of five unconstrained functions taken from (Trelea, 2003), whereas the second set is composed of 13 constrained functions taken from (Toscano Pulido & Coello Coello, 2004).

AII.1. Introduction

Although performing well or badly on a set of problems does not guarantee that the trend would persist for other problems, sets of especially selected benchmark problems posing different difficulties are the most popular mechanism to assess the performance of an optimizer, compare it to those of others, and infer similar performance for problems of the same characteristics. Two sets of benchmark problems are offered hereafter. The first one is composed of unconstrained problems presenting different difficulties, which also allow testing different dimensionalities. The second set is composed of problems with inequality, equality, or both types of constraints.

Maximization problems are turned into minimization ones.

AII.2. Unconstrained problems

This test suite of unconstrained problems is taken from (Trelea, 2003), and each function is considered for 2-, 10-, and 30-dimensional search-spaces.

AII.2.1. Sphere

This is a simple, unimodal function, easy for gradient-descent methods as all gradients point towards the global optimum. However, not only do optimization algorithms have to perform well on extremely complex problems but also on the simple ones. Hence it is often used for benchmarking gradient-free optimization methods.

$$f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 \qquad \text{(AII.1)}$$

Interval constraints: $[-100,100]^n$

Dimensionality: $n = 2$; $n = 10$; $n = 30$

Location of global optimum: $x_i^* = 0 \quad \forall i$

Global optimum: $f(\mathbf{x}^*) = 0$

A plot of the 2-dimensional sphere function is offered in Fig. AII.1.

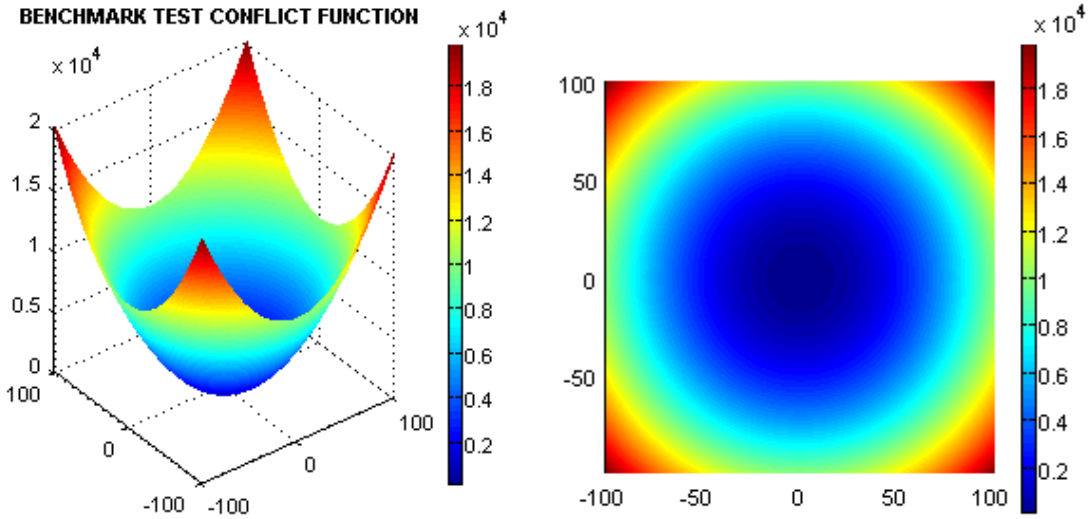


Fig. AII.1: Surface plot and colour-map of the 2-dimensional Sphere function within the region $[-100,100]^2$.

AII.2.2. Rosenbrock

This is a harder unimodal function –suitable for search-spaces of more than one dimension–, which displays an extensive flat surface around the optimum. For more than three dimensions, the problem becomes moderately multimodal.

$$f(\mathbf{x}) = \sum_{i=1}^{n-1} \left[100 \cdot \left(x_{i+1} - x_i^{\,2} \right)^2 + (x_i - 1)^2 \right] \qquad \text{(AII.2)}$$

Interval constraints: $[-30,30]^n$

Dimensionality: $n = 2$; $n = 10$; $n = 30$

Location of the global optimum: $x_i^* = 1 \quad \forall i$

Global optimum: $f(\mathbf{x}^*) = 0$

Plots of the 2-dimensional Rosenbrock function are offered in Fig. AII.2 to Fig. AII.4.

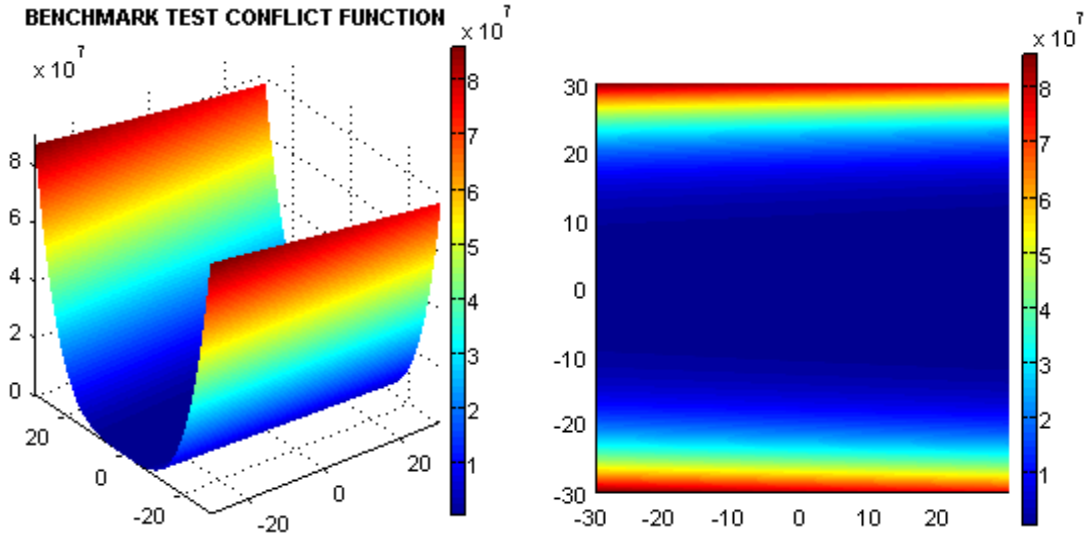


Fig. AII.2: Surface plot and colour-map of the 2-dimensional Rosenbrock function within the region $[-30,30]^2$.

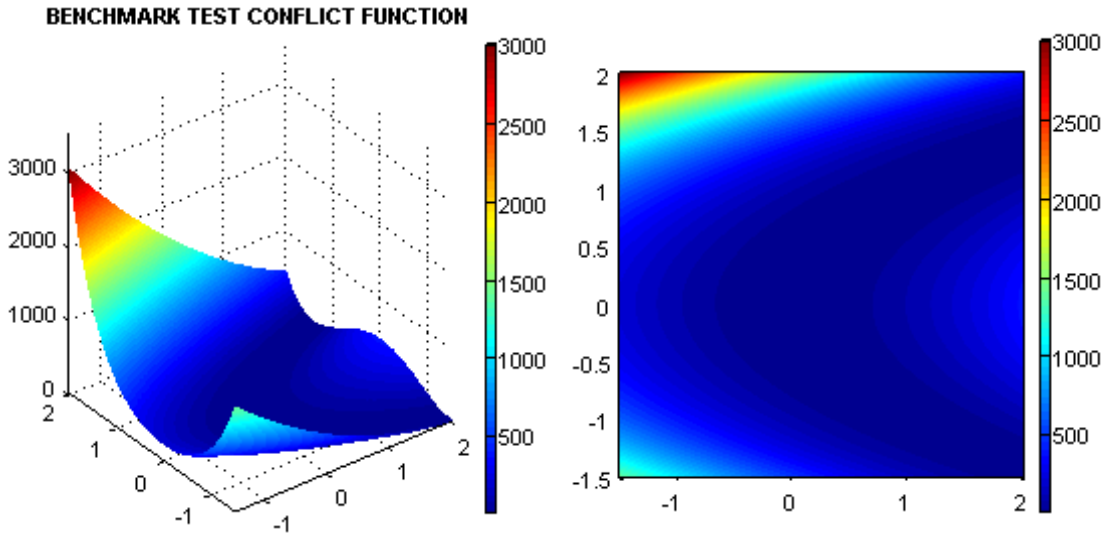


Fig. AII.3: Surface plot and colour-map of the 2-dimensional Rosenbrock function within the region $[-1.5,2.0]^2$.

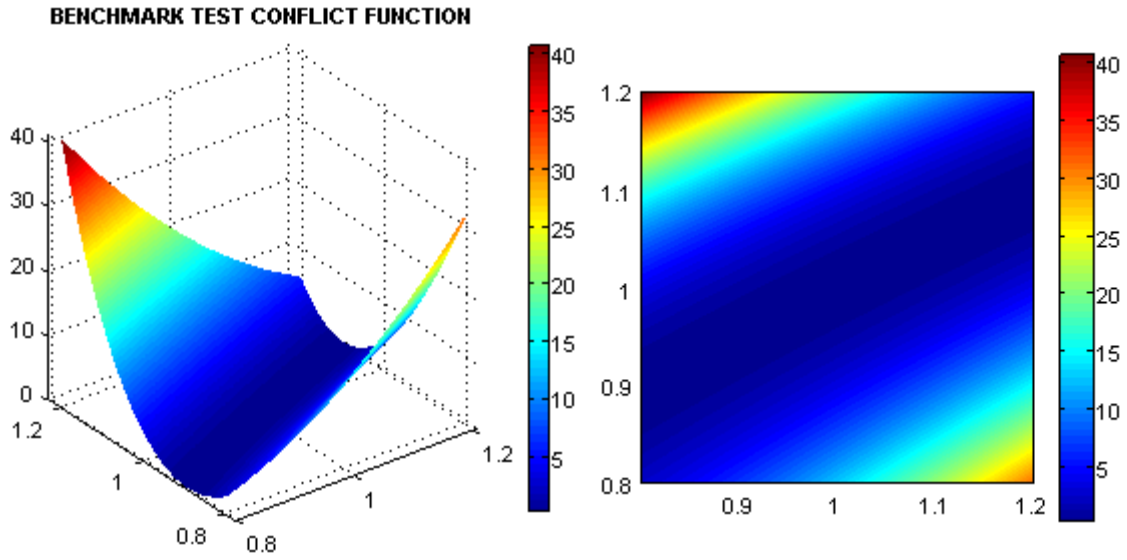


Fig. AII.4: Surface plot and colour-map of the 2-dimensional Rosenbrock function within the region $[0.8,1.2]^2$.

AII.2.3. Rastrigin

This function displays a single global optimum but many local optima in the form of valleys. The difficulty in finding the global optimum is in the numerous local optima whose conflict values are close to the global optimum.

$$f(\mathbf{x}) = \sum_{i=1}^{n} \left[x_i^{\,2} - 10 \cdot \cos(2 \cdot \pi \cdot x_i) + 10 \right] \qquad \text{(AII.3)}$$

Interval constraints: $[-5.12, 5.12]^n$

Dimensionality: $n = 2$; $n = 10$; $n = 30$

Location of the global optimum: $x_i^* = 0 \quad \forall i$

Global optimum: $f(\mathbf{x}^*) = 0$

The multimodality of the function can be observed in the plots offered in Fig. AII.5 and Fig. AII.6 for 2-dimensional search-spaces. Note that multimodality may refer either to local or to global optima. Thus, the Rastrigin function is multimodal in the sense that there are local optima, but unimodal in the sense that the global optimum is unique.

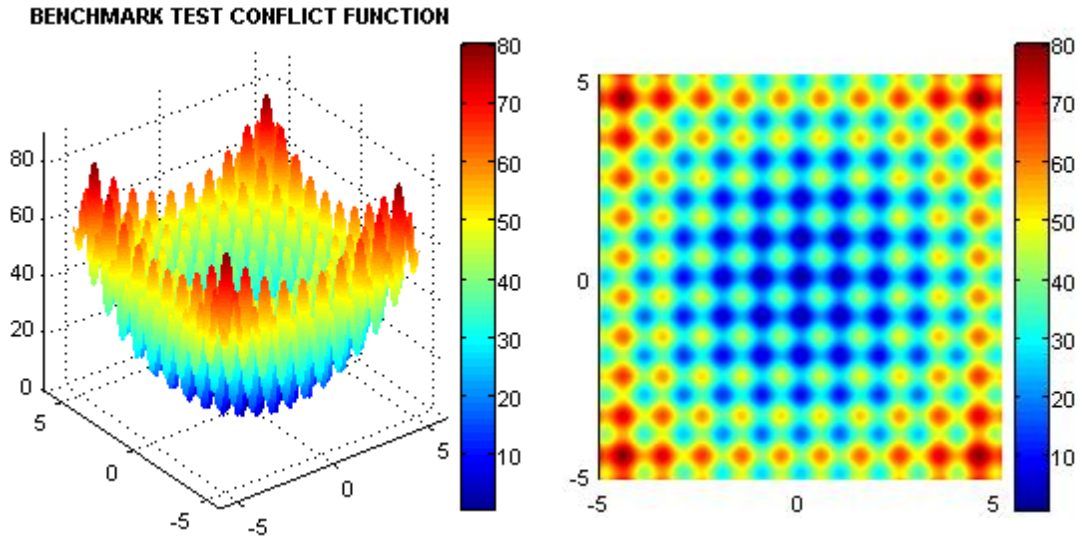


Fig. AII.5: Surface plot and colour-map of the 2-dimensional Rastrigin function within the region $[-5.12,5.12]^2$.

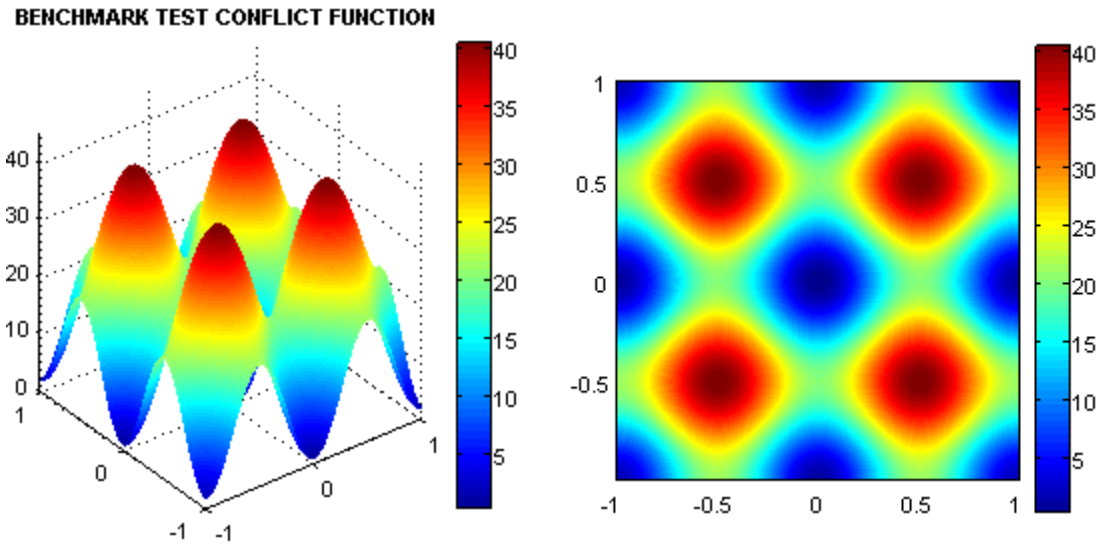


Fig. AII.6: Surface plot and colour-map of the 2-dimensional Rastrigin function within the region $[-1.0,1.0]^2$.

AII.2.4. Griewank

This function also displays a single global optimum and numerous local optima in the form of valleys. It can be viewed as a sort of noisy sphere function, where the amount of noise decreases with the increase of dimensionality.

$$f(\mathbf{x}) = \frac{1}{4000} \cdot \sum_{i=1}^{n} x_i^{\,2} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \qquad \text{(AII.4)}$$

Interval constraints: $[-600,600]^n$

Dimensionality: $n = 2$; $n = 10$; $n = 30$

Location of the global optimum: $x_i^* = 0 \quad \forall i$

Global optimum: $f(\mathbf{x}^*) = 0$

Thus, the influence of the term $\prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$ decreases as n increases. Therefore higher dimensions do not necessarily imply an increase in the difficulty of the problem.

Several plots of the 2-dimensional Griewank function are offered in Fig. AII.7 to Fig. AII.9 for different regions of the search-space.

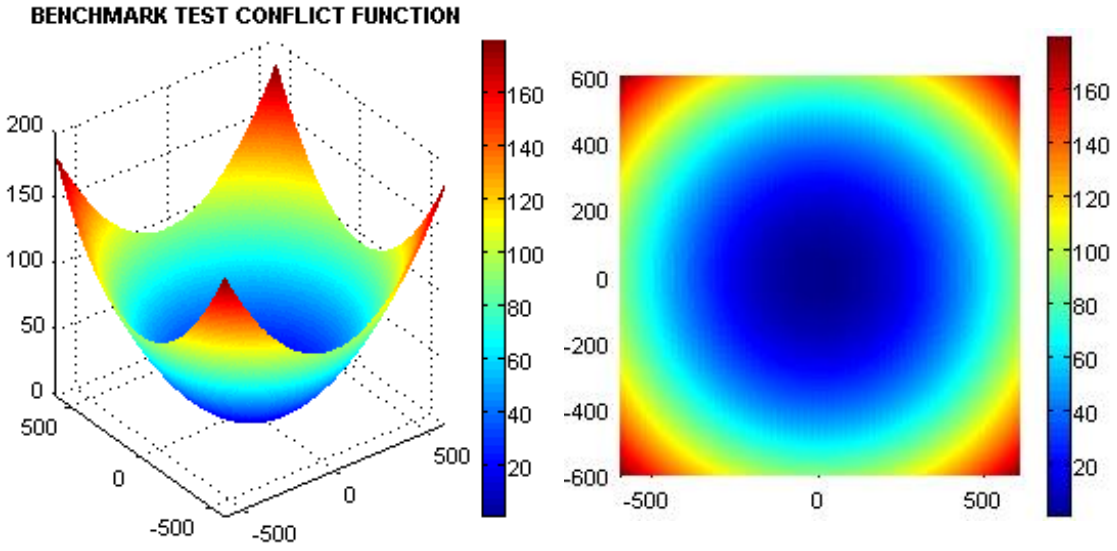


Fig. AII.7: Surface plot and colour-map of the 2-dimensional Griewank function within the region $[-600,600]^2$.

AII.2.5. Schaffer f6

This comprises a very hard function to be optimized, which presents numerous local optima in the form of ring-like depressions rather than the more typical valleys. Thus, there are a high number of local optima of different conflict values (different *iso-rings*).

In turn, there are infinite local optima within each ring. It is nevertheless a unimodal function in the sense that there is a single global optimum.

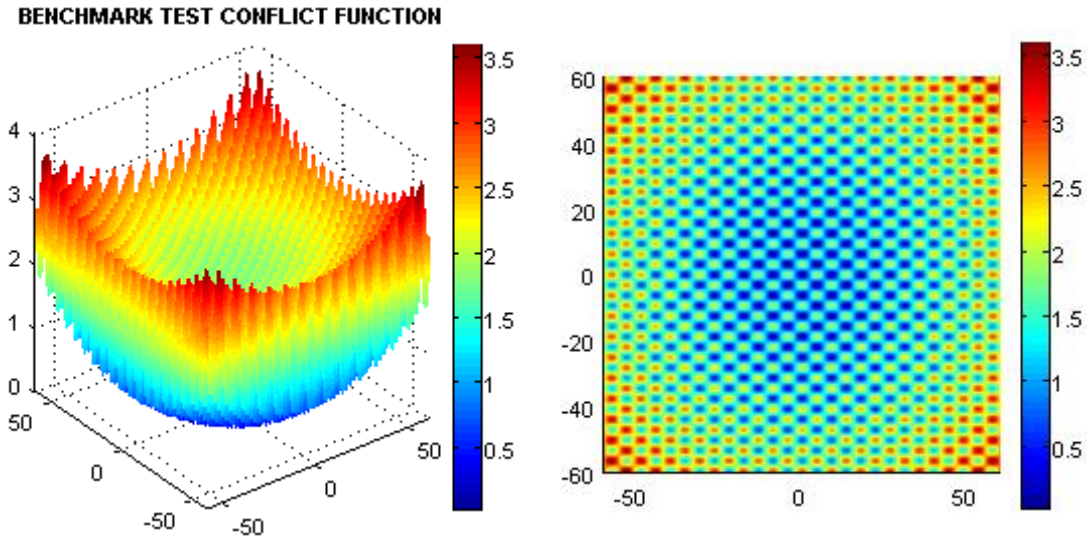


Fig. AII.8: Surface plot and colour-map of the 2-dimensional Griewank function within the region $[-60,60]^2$.

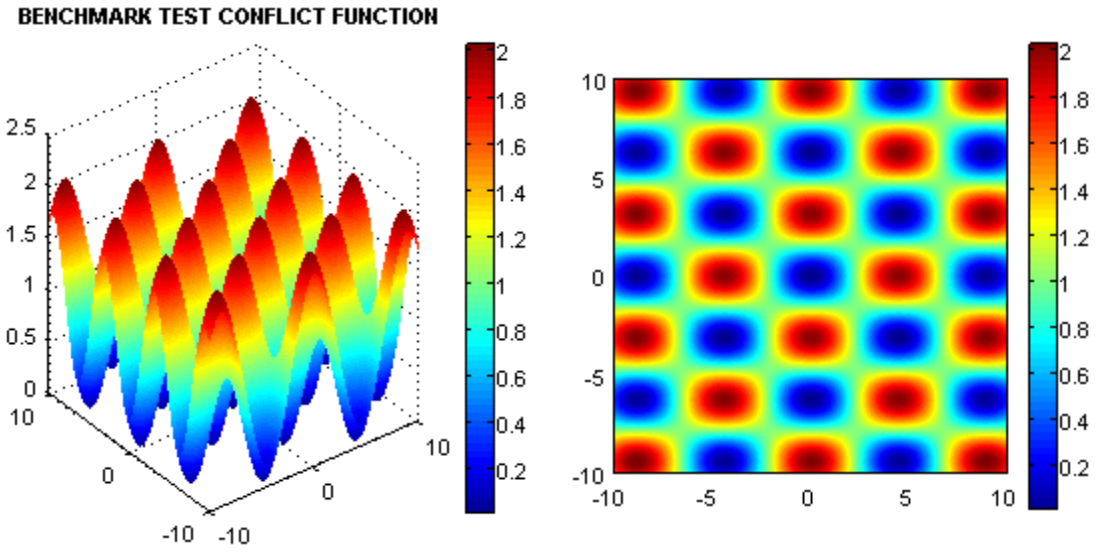


Fig. AII.9: Surface plot and colour-map of the 2-dimensional Griewank function within the region $[-10,10]^2$.

$$f(\mathbf{x}) = \frac{\left[\sin\left(\sqrt{\sum_{i=1}^{n} x_i^{\,2}}\right)\right]^2 - 0.5}{\left(1 + 0.001 \cdot \sum_{i=1}^{n} x_i^{\,2}\right)^2} + 0.5 \qquad \text{(AII.5)}$$

Interval constraints: $[-100,100]^n$

Dimensionality: $n = 2$; $n = 10$; $n = 30$

Location of the global optimum: $x_i^* = 0 \quad \forall i$

Global optimum: $f(\mathbf{x}^*) = 0$

Plots of the 2-dimensional Schaffer f6 function are offered in Fig. AII.10 to Fig. AII.13.

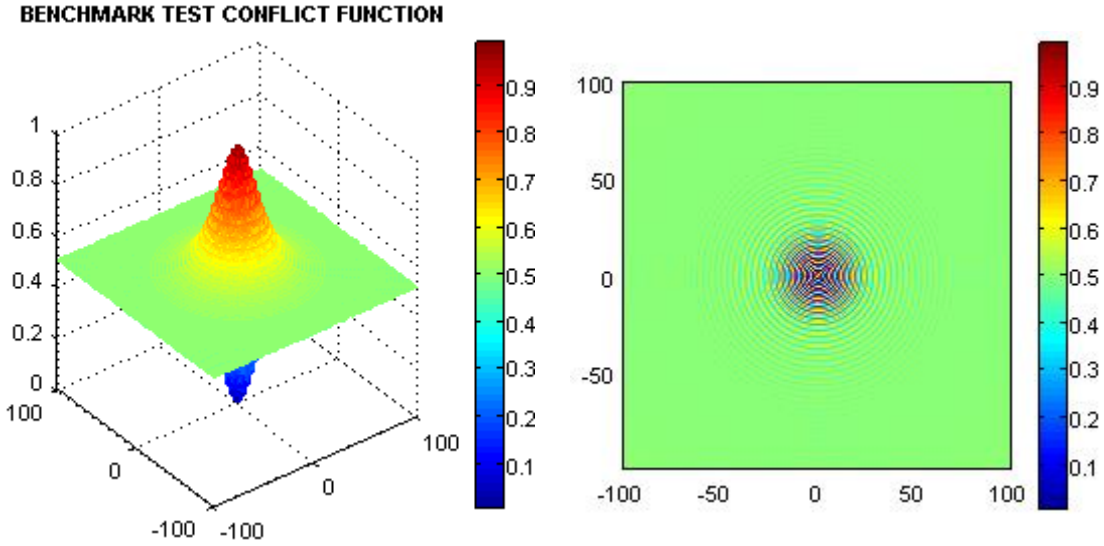


Fig. AII.10: Surface plot and colour-map of the 2-dimensional Schaffer f6 function within the region $[-100,100]^2$.

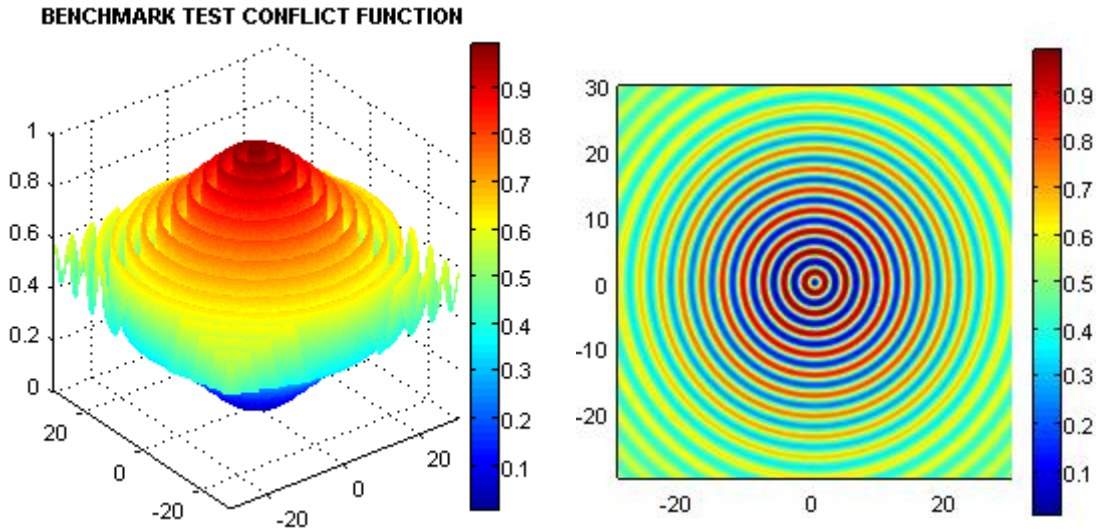


Fig. AII.11: Surface plot and colour-map of the 2-dimensional Schaffer f6 function within the region $[-30,30]^2$.

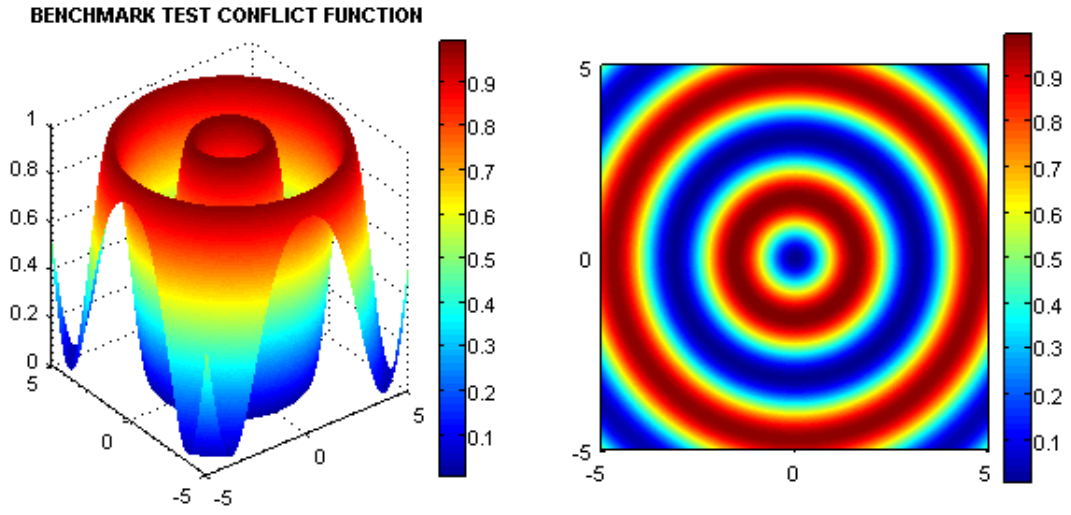


Fig. AII.12: Surface plot and colour-map of the 2-dimensional Schaffer f6 function within the region $[-5,5]^2$.

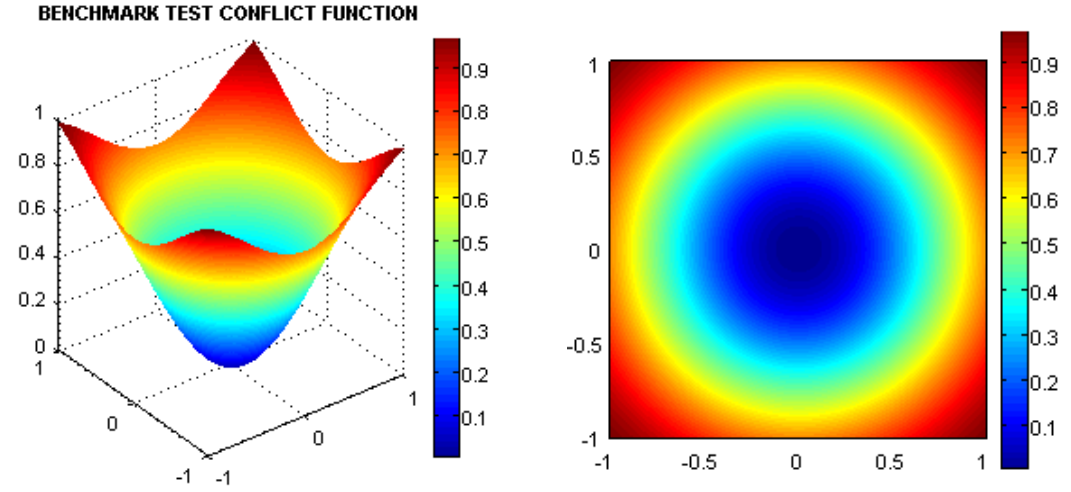


Fig. AII.13: Surface plot and colour-map of the 2-dimensional Schaffer f6 function within the region $[-1,1]^2$.

It is not very difficult for the particles in PSO to approach the region that contains the global optimum, since the closer to the global optimum the better the local optima. However, the fact that there are very good local optima completely surrounding the global optimum makes the latter harder to find, especially for higher dimensions.

AII.3. Constrained problems

This set has been successively extended through the years by different researchers. Thus, the first 11 problems were tested in (Hamida & Schoenauer, 2002), the first 12

problems in (Hu & Eberhart, 2002), and the 13 problems in (Toscano Pulido & Coello Coello, 2004). More recently, the benchmark has been remarkably extended to 24 problems in (Liang, et al., 2006) for *CEC'06* special session on *Constrained Real-Parameter Optimization*. Due to time constraints, only the first 13 problems –i.e. those in (Toscano Pulido & Coello Coello, 2004)– are dealt with in this thesis. Nonetheless, the solutions presented hereafter are taken from (Liang, et al., 2006), as the same tolerance for violations of equality constraints is adopted here: $Tol_{eq} = 0.0001$.

AII.3.1. Problem 01 (g01)

The problem consists of a 13-dimensional search-space, a nonlinear (quadratic) objective function in all 13 variables, 9 linear inequality constraints, and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = 5 \cdot \left(\sum_{j=1}^{4} x_j - \sum_{j=1}^{4} {x_j}^2 \right) - \sum_{j=5}^{13} x_j \tag{AII.6}$$

Subject to:

$$\begin{aligned}
g_1(\mathbf{x}) &= 2 \cdot x_1 + 2 \cdot x_2 + x_{10} + x_{11} - 10 \le 0 \\
g_2(\mathbf{x}) &= 2 \cdot x_1 + 2 \cdot x_3 + x_{10} + x_{12} - 10 \le 0 \\
g_3(\mathbf{x}) &= 2 \cdot x_2 + 2 \cdot x_3 + x_{11} + x_{12} - 10 \le 0 \\
g_4(\mathbf{x}) &= -8 \cdot x_1 + x_{10} \le 0 \\
g_5(\mathbf{x}) &= -8 \cdot x_2 + x_{11} \le 0 \\
g_6(\mathbf{x}) &= -8 \cdot x_3 + x_{12} \le 0 \\
g_7(\mathbf{x}) &= -2 \cdot x_4 - x_5 + x_{10} \le 0 \\
g_8(\mathbf{x}) &= -2 \cdot x_6 - x_7 + x_{11} \le 0 \\
g_9(\mathbf{x}) &= -2 \cdot x_8 - x_9 + x_{12} \le 0
\end{aligned} \tag{AII.7}$$

$$\begin{aligned}
x_j - 1 \le 0 \quad ; \quad -x_j \le 0 \quad ; \quad j = 1,\ldots,9,13 \\
x_j - 100 \le 0 \quad ; \quad -x_j \le 0 \quad ; \quad j = 10,11,12
\end{aligned}$$

Solution:

$$\mathbf{x}^* = (1,1,1,1,1,1,1,1,1,3,3,3,1)$$
$$f(\mathbf{x}^*) = -15 \qquad \text{(AII.8)}$$
$$g_1(\mathbf{x}^*), g_2(\mathbf{x}^*), g_3(\mathbf{x}^*), g_4(\mathbf{x}^*), g_5(\mathbf{x}^*), \text{and } g_6(\mathbf{x}^*) \text{ are active}$$

AII.3.2. Problem 02 (g02)

The problem consists of a 20-dimensional search-space, a nonlinear objective function in all 20 object variables, 1 linear and 1 nonlinear inequality constraints, and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = -\left| \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2 \cdot \prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i \cdot x_i^2}} \right| \qquad \text{(AII.9)}$$

Subject to:

$$g_1(\mathbf{x}) = 0.75 - \prod_{i=1}^{n} x_i \le 0$$
$$g_2(\mathbf{x}) = \sum_{i=1}^{n} x_i - 7.5 \cdot n \le 0 \qquad \text{(AII.10)}$$

$$x_j - 10 \le 0 \quad ; \quad -x_j \le 0 \quad ; \quad j = 1,\ldots,n \quad ; \quad n = 20$$

Solution:

$$\mathbf{x}^* = \begin{pmatrix} 3.16246061572185, 3.12833142812967, 3.09479212988791, \\ 3.06145059523469, 3.02792915885555, 2.99382606701730, \\ 2.95866871765285, 2.92184227312450, 0.49482511456933, \\ 0.48835711005490, 0.48231642711865, 0.47664475092742, \\ 0.47129550835493, 0.46623099264167, 0.46142004984199, \\ 0.45683664767217, 0.45245876903267, 0.44826762241853, \\ 0.44424700958760, 0.44038285956317 \end{pmatrix} \qquad \text{(AII.11)}$$

$$f(\mathbf{x}^*) = -0.80361910412559 \quad ; \quad g_1(\mathbf{x}^*) \text{ is close to being active } \left(-10^{-8}\right)$$

AII.3.3. Problem 03 (g03)

The problem consists of a 10-dimensional search-space, a nonlinear objective function in all 10 object variables, 1 nonlinear equality constraint, and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = -\sqrt{n}^{n} \cdot \prod_{j=1}^{n} x_j \tag{AII.12}$$

Subject to:

$$abs\left(g_1(\mathbf{x}) = \sum_{j=1}^{n} x_j^{\,2} - 1 \right) \le Tol_{eq}$$
$$x_j - 1 \le 0 \quad ; \quad -x_j \le 0 \quad ; \quad j = 1,...,n \quad ; \quad n = 10 \tag{AII.13}$$

Solution:

$$\mathbf{x}^*: x_j = \frac{1}{\sqrt{n}} \quad ; \quad j = 1,...,n$$
$$f(\mathbf{x}^*) = -1 \tag{AII.14}$$

Solution for Tol_{eq} = 0.0001:

$$\mathbf{x}^* = \begin{pmatrix} 0.31624357647283069, 0.31624357741433839, \\ 0.31624357801234592 7, 0.31624357566401789 5, \\ 0.31624357820552606 6, 0.31624357738855069, \\ 0.31624357547294951 2, 0.31624357716488393 8, \\ 0.31624357815592030 2, 0.31624357614737491 6 \end{pmatrix} \tag{AII.15}$$
$$f(\mathbf{x}^*) = -1.00050010001000$$

AII.3.4. Problem 04 (g04)

The problem consists of a 5-dimensional search-space, a nonlinear (quadratic) objective function in all 4 variables, 3 nonlinear inequality constraints in the form of intervals

(hence sometimes referred to as six constraints), and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = 5.3578547 \cdot x_3^{\,2} + 0.8356891 \cdot x_1 \cdot x_5 + 37.293239 \cdot x_1 - 40792.141 \qquad \text{(AII.16)}$$

Subject to:

$$\begin{aligned} & g_1(\mathbf{x}) = 85.334407 + 0.0056858 \cdot x_2 \cdot x_5 + 0.0006262 \cdot x_1 \cdot x_4 - 0.0022053 \cdot x_3 \cdot x_5 \\ & g_2(\mathbf{x}) = 80.51249 + 0.0071317 \cdot x_2 \cdot x_5 + 0.0029955 \cdot x_1 \cdot x_2 + 0.0021813 \cdot x_3^{\,2} \\ & g_3(\mathbf{x}) = 9.300961 + 0.0047026 \cdot x_3 \cdot x_5 + 0.0012547 \cdot x_1 \cdot x_3 + 0.0019085 \cdot x_3 \cdot x_4 \\ & 0 \le g_1(\mathbf{x}) \le 92 \\ & 90 \le g_2(\mathbf{x}) \le 110 \\ & 20 \le g_3(\mathbf{x}) \le 25 \end{aligned} \qquad \text{(AII.17)}$$

$$\begin{aligned} & x_1 - 102 \le 0 \quad ; \quad -x_1 + 78 \le 0 \\ & x_2 - 45 \le 0 \quad ; \quad -x_2 + 33 \le 0 \\ & x_j - 45 \le 0 \quad ; \quad -x_j + 27 \le 0 \quad ; \quad j = 3,4,5 \end{aligned}$$

Solution:

$$\begin{aligned} & \mathbf{x}^* = (78,33,29.995256025681598 5,45,36.7758129057882073) \\ & f(\mathbf{x}^*) = -30665.53867178332 \\ & g_1(\mathbf{x}^*) \text{ and } g_6(\mathbf{x}^*) \text{ are active} \end{aligned} \qquad \text{(AII.18)}$$

AII.3.5. Problem 05 (g05)

The problem consists of a 4-dimensional search-space, a nonlinear objective function in 2 of the variables only, 1 nonlinear inequality constraint in the form of an interval (hence sometimes referred to as two constraints), 3 nonlinear equality constraints, and interval constraints on the variables. All 4 variables are involved in the constraint functions. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = 3 \cdot x_1 + 0.000001 \cdot x_1^{\,3} + 2 \cdot x_2 + 0.000002/3 \cdot x_2^{\,3} \qquad \text{(AII.19)}$$

Subject to:

$$\begin{aligned} &g_1(\mathbf{x}) = abs(x_3 - x_4) - 0.55 \le 0 \\ &abs(g_2(\mathbf{x}) = 1000 \cdot [\sin(-x_3 - 0.25) + \sin(-x_4 - 0.25)] + 894.8 - x_1) \le Tol_{eq} \\ &abs(g_3(\mathbf{x}) = 1000 \cdot [\sin(x_3 - 0.25) + \sin(x_3 - x_4 - 0.25)] + 894.8 - x_2) \le Tol_{eq} \\ &abs(g_4(\mathbf{x}) = 1000 \cdot [\sin(x_4 - 0.25) + \sin(-x_3 + x_4 - 0.25)] + 1294.8) \le Tol_{eq} \end{aligned} \quad \text{(AII.20)}$$

$$\begin{aligned} x_j - 1200 \le 0 \quad ; \quad -x_j + 0 \le 0 \quad ; \quad j = 1,2 \\ x_j - 0.55 \le 0 \quad ; \quad -x_j - 0.55 \le 0 \quad ; \quad j = 3,4 \end{aligned}$$

Solution for Tol_{eq} = 0.0001:

$$\begin{aligned} \mathbf{x}^* &= \begin{pmatrix} 679.945148297028709, 1026.06697600004691, \\ 0.118876369094410433, -0.39623348521517826 \end{pmatrix} \\ f(\mathbf{x}^*) &= 5126.4967140071 \end{aligned} \quad \text{(AII.21)}$$

AII.3.6. Problem 06 (g06)

The problem consists of a 2-dimensional search-space, a nonlinear objective function in the 2 variables, 2 nonlinear inequality constraints, and interval constraints on all variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = (x_1 - 10)^3 + (x_2 - 20)^3 \quad \text{(AII.22)}$$

Subject to:

$$\begin{aligned} &g_1(\mathbf{x}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0 \\ &g_2(\mathbf{x}) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0 \\ &x_1 - 100 \le 0 \quad ; \quad -x_1 + 13 \le 0 \\ &x_2 - 100 \le 0 \quad ; \quad -x_2 + 0 \le 0 \end{aligned} \quad \text{(AII.23)}$$

Solution:

$$\mathbf{x}^* = (14.095000000000000064, 0.84296078921547956 68)$$
$$f(\mathbf{x}^*) = -696181387558015 \quad ; \quad \text{Both } g_1(\mathbf{x}^*) \text{ and } g_2(\mathbf{x}^*) \text{ are active} \tag{AII.24}$$

AII.3.7. Problem 07 (g07)

The problem consists of a 10-dimensional search-space, a nonlinear objective function in all variables, 3 linear and 5 nonlinear inequality constraints, and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = x_1^2 + x_2^2 + x_1 \cdot x_2 - 14 \cdot x_1 - 16 \cdot x_2 + (x_3 - 10)^2 + 4 \cdot (x_4 - 5)^2 + (x_5 - 3)^2 + \\ + 2 \cdot (x_6 - 1)^2 + 5 \cdot x_7^2 + 7 \cdot (x_8 - 11)^2 + 2 \cdot (x_9 - 10)^2 + (x_{10} - 7)^2 + 45 \tag{AII.25}$$

Subject to:

$$\begin{aligned}
g_1(\mathbf{x}) &= 4 \cdot x_1 + 5 \cdot x_2 - 3 \cdot x_7 + 9 \cdot x_8 - 105 \le 0 \\
g_2(\mathbf{x}) &= 10 \cdot x_1 - 8 \cdot x_2 - 17 \cdot x_7 + 2 \cdot x_8 \le 0 \\
g_3(\mathbf{x}) &= -8 \cdot x_1 + 2 \cdot x_2 + 5 \cdot x_9 - 2 \cdot x_{10} - 12 \le 0 \\
g_4(\mathbf{x}) &= 3 \cdot (x_1 - 2)^2 + 4 \cdot (x_2 - 3)^2 + 2 \cdot x_3^2 - 7 \cdot x_4 - 120 \le 0 \\
g_5(\mathbf{x}) &= 5 \cdot x_1^2 + 8 \cdot x_2 + (x_3 - 6)^2 - 2 \cdot x_4 - 40 \le 0 \\
g_6(\mathbf{x}) &= x_1^2 + 2 \cdot (x_2 - 2)^2 - 2 \cdot x_1 \cdot x_2 + 14 \cdot x_5 - 6 \cdot x_6 \le 0 \\
g_7(\mathbf{x}) &= 0.5 \cdot (x_1 - 8)^2 + 2 \cdot (x_2 - 4)^2 + 3 \cdot x_5^2 - x_6 - 30 \le 0 \\
g_8(\mathbf{x}) &= -3 \cdot x_1 + 6 \cdot x_2 + 12 \cdot (x_9 - 8)^2 - 7 \cdot x_{10} \le 0
\end{aligned} \tag{AII.26}$$

$$x_j - 10 \le 0 \quad ; \quad -x_j - 10 \le 0 \quad ; \quad j = 1, \ldots, 10$$

Solution:

$$\mathbf{x}^* = \begin{pmatrix} 2.17199634142692, 2.3636830416034, 8.77392573913157, \\ 5.09598443745173, 0.990654756560493, 1.43057392853463, \\ 1.32164415364306, 9.82872576524495, 8.2800915887356, \\ 8.3759266477347 \end{pmatrix} \tag{AII.27}$$
$$f(\mathbf{x}^*) = 24.30620906818$$
$$g_1(\mathbf{x}^*), g_2(\mathbf{x}^*), g_3(\mathbf{x}^*), g_4(\mathbf{x}^*), g_5(\mathbf{x}^*), \text{and } g_6(\mathbf{x}^*) \text{ are active}$$

AII.3.8. Problem 08 (g08)

The problem consists of a 2-dimensional search-space, a nonlinear objective function in the 2 variables, 2 nonlinear inequality constraints, and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = -\frac{\sin^3(2\cdot\pi\cdot x_1)\cdot\sin(2\cdot\pi\cdot x_2)}{x_1^{\,3}\cdot(x_1+x_2)} \tag{AII.28}$$

Subject to:

$$\begin{aligned} g_1(\mathbf{x}) &= x_1^{\,2} - x_2 + 1 \le 0 \\ g_2(\mathbf{x}) &= 1 - x_1 + (x_2 - 4)^2 \le 0 \\ x_j - 10 \le 0 \quad ; \quad -x_j + 0 \le 0 \quad ; \quad j &= 1,2 \end{aligned} \tag{AII.29}$$

Solution:

$$\begin{aligned} \mathbf{x}^* &= (1.2279713526075 2599, 4.24537336612274885) \\ f(\mathbf{x}^*) &= -0.0958250414180359 \end{aligned} \tag{AII.30}$$

AII.3.9. Problem 09 (g09)

The problem consists of a 7-dimensional search-space, a nonlinear objective function in all 7 variables, 4 nonlinear inequality constraints, and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$\begin{aligned} f(\mathbf{x}) = {} & (x_1-10)^2 + 5\cdot(x_2-12)^2 + x_3^{\,4} + 3\cdot(x_4-11)^2 + \\ & + 10\cdot x_5^{\,6} + 7\cdot x_6^{\,2} + x_7^{\,4} - 4\cdot x_6\cdot x_7 - 10\cdot x_6 - 8\cdot x_7 \end{aligned} \tag{AII.31}$$

Subject to:

$$
\begin{aligned}
&g_1(\mathbf{x}) = 2\cdot x_1^{\,2} + 3\cdot x_2^{\,4} + x_3 + 4\cdot x_4^{\,2} + 5\cdot x_5 - 127 \le 0\\
&g_2(\mathbf{x}) = 7\cdot x_1 + 3\cdot x_2 + 10\cdot x_3^{\,2} + x_4 - x_5 - 282 \le 0\\
&g_3(\mathbf{x}) = 23\cdot x_1 + x_2^{\,2} + 6\cdot x_6^{\,2} - 8\cdot x_7 - 196 \le 0\\
&g_4(\mathbf{x}) = 4\cdot x_1^{\,2} + x_2^{\,2} - 3\cdot x_1\cdot x_2 + 2\cdot x_3^{\,2} + 5\cdot x_6 - 11\cdot x_7 \le 0
\end{aligned}
\qquad \text{(AII.32)}
$$

$$x_j - 10 \le 0 \quad ; \quad -x_j - 10 \le 0 \quad ; \quad j = 1,...,7$$

Solution:

$$
\mathbf{x}^* = \begin{pmatrix}
2.33049935147405174, 1.95137236847114592,\\
-0.47754139951061580 5, 4.36572624923625874,\\
-0.62448695910038898 3, 1.03813099410962173,\\
1.5942266780671519)
\end{pmatrix}
\qquad \text{(AII.33)}
$$

$$f(\mathbf{x}^*) = 680.630057374402 \quad ; \quad g_1(\mathbf{x}^*) \text{ and } g_4(\mathbf{x}^*) \text{ are active}$$

AII.3.10. Problem 10 (g10)

The problem consists of an 8-dimensional search-space, a linear objective function in only 3 variables, 3 linear and 3 nonlinear inequality constraints, and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = x_1 + x_2 + x_3 \qquad \text{(AII.34)}$$

Subject to:

$$
\begin{aligned}
&g_1(\mathbf{x}) = -1 + 0.0025\cdot(x_4 + x_6) \le 0\\
&g_2(\mathbf{x}) = -1 + 0.0025\cdot(x_5 + x_7 - x_4) \le 0\\
&g_3(\mathbf{x}) = -1 + 0.01\cdot(x_8 - x_5) \le 0\\
&g_4(\mathbf{x}) = -x_1\cdot x_6 + 833.33252\cdot x_4 + 100\cdot x_1 - 83333.333 \le 0\\
&g_5(\mathbf{x}) = -x_2\cdot x_7 + 1250\cdot x_5 + x_2\cdot x_4 - 1250\cdot x_4 \le 0\\
&g_6(\mathbf{x}) = -x_3\cdot x_8 + 1250000 + x_3\cdot x_5 - 2500\cdot x_5 \le 0
\end{aligned}
\qquad \text{(AII.35)}
$$

$$
\begin{aligned}
&x_1 - 10000 \le 0 \quad ; \quad -x_1 + 100 \le 0 \quad ; \quad j = 1,...,7\\
&x_j - 10000 \le 0 \quad ; \quad -x_j + 1000 \le 0 \quad ; \quad j = 2,3\\
&x_j - 1000 \le 0 \quad ; \quad -x_j + 10 \le 0 \quad ; \quad j = 4,...,8
\end{aligned}
$$

Solution:

$$\mathbf{x}^* = \begin{pmatrix} 579.306685017979589, 1359.97067807935605, 5109.97065743133317, \\ 182.01769963061534, 295.601173702746792, 217.982300369384632, \\ 286.41652592786852, 395.601173702746735 \end{pmatrix} \quad \text{(AII.36)}$$

$$f(\mathbf{x}^*) = 7049.24802052867$$

$g_1(\mathbf{x}^*)$, $g_2(\mathbf{x}^*)$ and $g_3(\mathbf{x}^*)$ are active

AII.3.11. Problem 11 (g11)

The problem consists of a 2-dimensional search-space, a nonlinear (quadratic) objective function in the 2 variables, 1 nonlinear equality constraint, and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = x_1^{\,2} + (x_2 - 1)^2 \quad \text{(AII.37)}$$

Subject to:

$$abs\left(g_1(\mathbf{x}) = x_2 - x_1^{\,2}\right) \le Tol_{eq}$$
$$x_j - 1 \le 0 \quad ; \quad -x_j - 1 \le 0 \quad ; \quad j = 1,2 \quad \text{(AII.38)}$$

Solution:

$$\mathbf{x}^* = \left(\pm\frac{1}{\sqrt{2}}, \frac{1}{2}\right) \quad \text{(AII.39)}$$
$$f(\mathbf{x}^*) = 0.75$$

Solution for Tol_{eq} = 0.0001:

$$\mathbf{x}^* = (\pm 0.707036070037170616, 0.500000004333606807) \quad \text{(AII.40)}$$
$$f(\mathbf{x}^*) = 0.7499$$

AII.3.12. Problem 12 (g12)

The problem consists of a 3-dimensional search-space, a nonlinear (quadratic) objective function in all 3 variables, 1 nonlinear inequality constraint composed of 9^3 (729) disjointed spheres, and interval constraints on the variables. Although it is commonly said in the literature that there are 729 constraints, in reality it is only one, as complying with one of them automatically means violating all others. That is to say that only one can be met at a time. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = -\frac{100 - (x_1 - 5)^2 - (x_2 - 5)^2 - (x_3 - 5)^2}{100} \tag{AII.41}$$

Subject to:

$$\begin{gathered} g_1(\mathbf{x}) = (x_1 - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \le 0 \\ \text{where} \quad p, q, r = 1, \ldots, 9 \end{gathered} \tag{AII.42}$$

$$x_j - 10 \le 0 \quad ; \quad -x_j \le 0 \quad ; \quad j = 1,2,3$$

Solution:

$$\begin{gathered} \mathbf{x}^* = (5,5,5) \\ f(\mathbf{x}^*) = -1 \end{gathered} \tag{AII.43}$$

AII.3.13. Problem 13 (g13)

The problem consists of a 5-dimensional search-space, a nonlinear objective function in all 5 variables, 3 nonlinear equality constraints, and interval constraints on the variables. The problem can be formulated as follows:

Minimize:

$$f(\mathbf{x}) = e^{x_1 \cdot x_2 \cdot x_3 \cdot x_4 \cdot x_5} \tag{AII.44}$$

Subject to:

$$
\begin{aligned}
&abs\left(g_1(\mathbf{x}) = x_1^{\,2} + x_2^{\,2} + x_3^{\,2} + x_4^{\,2} + x_5^{\,2} - 10\right) \le Tol_{eq} \\
&abs\left(g_2(\mathbf{x}) = x_2 \cdot x_3 - 5 \cdot x_4 \cdot x_5\right) \le Tol_{eq} \\
&abs\left(g_3(\mathbf{x}) = x_1^{\,3} + x_2^{\,3} + 1\right) \le Tol_{eq}
\end{aligned}
\qquad \text{(AII.45)}
$$

$$
\begin{aligned}
&x_j - 2.3 \le 0 \quad ; \quad -x_j - 2.3 \le 0 \quad ; \quad j = 1,2 \\
&x_j - 3.2 \le 0 \quad ; \quad -x_j - 3.2 \le 0 \quad ; \quad j = 3,4,5
\end{aligned}
$$

Solution:

$$
\begin{aligned}
&\mathbf{x}^* = \begin{pmatrix} -1.71714224003, 1.59572124049468, 1.8272502406271, \\ -0.763659881912867, -0.76365986736498 \end{pmatrix} \\
&f\left(\mathbf{x}^*\right) = 0.053941514041898
\end{aligned}
\qquad \text{(AII.46)}
$$