Particle Swarms Reformulated towards a Unified and Flexible Framework

Mauro Sebastián Innocente[0000−0001−8836−2839]

Autonomous Vehicles & Artificial Intelligence Laboratory (AVAILab),
Coventry University, Coventry, UK
Mauro.S.Innocente@coventry.ac.uk
https://www.availab.org/

Abstract. The Particle Swarm Optimisation (PSO) algorithm has undergone countless modifications and adaptations since its original formulation in 1995. Some of these have become mainstream whereas others have faded away. A myriad of alternative formulations have been proposed raising the question of what the basic features of an algorithm must be to belong in the PSO family. The aim of this paper is to establish what defines a PSO algorithm and to attempt to formulate it in such a way that it encompasses many existing variants. Therefore, different versions of the method may be posed as settings within the proposed unified framework. In addition, the proposed formulation generalises, decouples and incorporates features to the method providing more flexibility to the behaviour of each particle. The closed forms of the trajectory difference equation are obtained, different types of behaviour are identified, stochasticity is decoupled, and traditionally global features such as sociometries and constraint-handling are re-defined as particle's attributes.

Keywords: particle swarm optimisation · coefficients' settings · types of behaviour · trajectory · learning strategy · unstructured neighbourhood.

1 Introduction

Proposed in 1995 [20], the Particle Swarm Optimisation (PSO) method is a global optimiser in the sense that it is able to escape poor suboptimal attractors by means of a parallel collaborative search. The overall system behaviour emerges from a combination of each particle's individual and social behaviours. The former is manifested by the trajectory of a particle pulled by its attractors, governed by a second order difference equation with three control coefficients. In the classical (and in most) versions of the algorithm, there is one individual attractor given by the particle's best experience, and one social attractor given by the best experience in its neighbourhood. The social behaviour is governed by the way the individually acquired information is shared among particles and therefore propagated throughout the swarm, which is controlled by the neighbourhood topology. The individual and social behaviours interact through the update of the social attractor. Thus, the two main features of the algorithm are

Accepted Version - The final authenticated article is published by Springer-Nature:
M.S. Innocente (2021). Particle Swarms Reformulated towards a Unified and Flexible Framework. In: Tan Y., Shi Y. (eds.) Advances in Swarm Intelligence. ICSI 2021. Lecture Notes in Computer Science, vol 12689. Springer Nature.
DOI: 10.1007/978-3-030-78743-1_25

the *trajectory difference equation* (and the setting of its coefficients) and the *neighbourhood topology* (a.k.a. *sociometry*).

In the early days, numerous empirical studies were carried out to investigate the influence of the coefficients in the *trajectory difference equation* on the overall performance of the method, and to provide guidelines for their settings [28, 21]. Early theoretical work [25, 5, 30] provided insight into how the method works and interesting findings of practical use such as constriction factor(s) [5] to ensure convergence. These pioneering studies were a source of inspiration and set the foundations for an explosion of theoretical work [17, 22, 27, 9, 3, 14, 10, 4, 2].

1.1 Trajectory Difference Equation

In *classical PSO* (CPSO), three forces govern a particle's trajectory: the inertia from its previous displacement, the attraction to its own best experience, and the attraction to the best experience in its neighbourhood. The importance awarded to each of them is controlled by three coefficients: the inertia (ω), the individuality (iw), and the sociality (sw) weights. Stochasticity is introduced to enhance exploration via random weights applied to iw and sw. The behaviour of a particle, and by extension of the PSO algorithm as a whole, is very sensitive to the settings of these control coefficients. The system of two 1st-order difference equations for position and velocity updates in the CPSO algorithm proposed in [29] is rearranged in (1) as a single 2nd-order *Trajectory Difference Equation*:

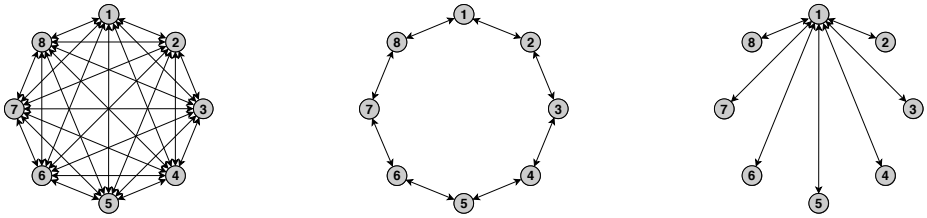
$$\begin{aligned} x_{ij}^{(t+1)} = x_{ij}^{(t)} &+ \omega_{ij}^{(t)} \left(x_{ij}^{(t)} - x_{ij}^{(t-1)} \right) \\ &+ iw_{ij}^{(t)} U_{(0,1)} \left(xb_{ij}^{(t)} - x_{ij}^{(t)} \right) + sw_{ij}^{(t)} U_{(0,1)} \left(xb_{kj}^{(t)} - x_{ij}^{(t)} \right) \end{aligned} \tag{1}$$

where $x_{ij}^{(t)}$ is the coordinate j of the *position* of particle i at time-step t; $xb_{ij}^{(t)}$ is the coordinate j of the *best experience* of particle i by time-step t; k is the index identifying the particle with the best experience in the neighbourhood of particle i at time-step t; ω, iw and sw are the inertia, individuality, and sociality weights, respectively (which may depend on i, j, t); and $U_{(0,1)}$ is a random number from a uniform distribution within [0,1] resampled anew every time it is referenced.

In the original formulation [20], $\omega = 1$ and $iw = sw = 2$. This leads to an unstable system, as particles tend to diverge. The first strategy to prevent this was to bound the size of each component of a particle's displacement, which helps prevent the so-called *explosion* but does not ensure convergence or a fine-grain search. Instead, the coefficients in (1) can be set to ensure that.

1.2 Neighbourhood Topology

The original PSO algorithm [20] presented a global topology in which every particle has access to the memory of every other particle in the swarm. Local topologies were proposed soon thereafter [8]. Since then, a plethora of sociometries have been proposed [23, 24, 1]. Three classical ones are shown in Fig. 1.



(a) Global (b) Ring (c) Wheel

Fig. 1. Three classical neighbourhood topologies in PSO.

The global topology tends to lead to a rapid loss of diversity, which may lead to premature convergence to a poor suboptimal solution. Whilst this can be controlled to some extent by the settings of the coefficients in the trajectory equation, numerous neighbourhood topologies have been proposed reducing connectivity to delay the propagation of information throughout the swarm.

1.3 Other Features

Other important features of the PSO algorithm are the initialisation of the particles [11, 19, 18], the synchrony of the memory updates, the size of the swarm [7, 26], and the handling of constraints [16].

The PSO algorithm is an unconstrained search method, therefore requiring an external constraint-handling technique (CHT) to be integrated to handle these types of problems. A straightforward CHT is the *Preserving Feasibility Method* [12], in which infeasible experiences are banned from memory. Another one is the *Penalty Method*, in which infeasible solutions are penalised by augmenting the objective function and treating the problem as unconstrained. Some authors propose adaptive penalties by using adaptive coefficients in the penalty function [6] or by adapting the tolerance relaxation [15]. Innocente et al. [13] propose using a *Preserving Feasibility with Priority Rules Method*, in which the objective function values and the constraint violations are treated separately.

Since its original formulation in 1995, countless PSO variants have been proposed. Some of them have become mainstream whereas many others have faded away. Thus, a myriad of alternative formulations have been proposed raising the question of what the basic features of an algorithm must be to belong in the PSO family. The aim of this paper is to establish what defines a PSO algorithm, and to attempt to formulate it in such a way that it encompasses many existing variants so that different versions may be posed as settings within the proposed unified framework. In addition, the proposed formulation generalises, decouples and incorporates new features providing more flexibility to the behaviour of each particle. The remainder of this paper is organised as follows: the overall proposed *Reformulated PSO* is introduced in Section 2, with the *Global Features*, the *Indi-*

vidual Behaviour Features and the *Social Behaviour Features* discussed in more details in Sections 3, 4 and 5, respectively. Conclusions are provided in Section 6.

2 Reformulated Particle Swarm Optimisation

The proposed *Reformulated Particle Swarm Optimisation* (RePSO) method is structured in three sets of features: 1) *Global Features* (GFs), 2) *Individual Behaviour Features* (IBFs), and 3) *Social Behaviour Features* (SBFs). Fig. 2 shows a high-level description of RePSO, where IBFs and SBFs are both viewed as individual attributes of a particle (*Particle Attributes*).

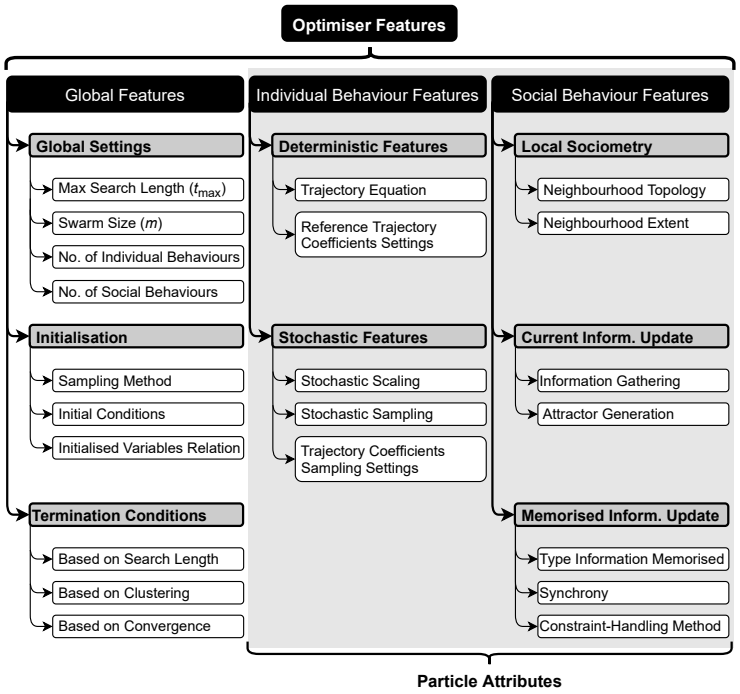


Fig. 2. High-level description of the proposed Reformulated PSO (RePSO).

3 Global Features

Despite being a swarm-intelligent method, some characteristics must still be defined at the swarm level. We define here three main subsets of global features: 1) *Global Settings*, 2) *Initialisation*, and 3) *Termination Conditions*. The first one consists of scalar settings like maximum search length (t_{max}) and swarm size (m), whereas the other two involve methods.

3.1 Initialisation

It is important to identify two aspects of the initialisation in PSO: 1) the *sampling method* to place m points over the search-space, and 2) what *variables* are to be initialised. Note that the particle's position update in RePSO is a 2^{nd} order difference equation as opposed to the classical system of two 1^{st} order difference equations (position and velocity). Therefore, the *variables* potentially involved in the initialisation are the initial, the previous, and the memorised positions ($\mathbf{x}^{(1)}$, $\mathbf{x}^{(0)}$, $\mathbf{xm}^{(1)}$) instead of two positions and one velocity ($\mathbf{x}^{(1)}$, $\mathbf{xm}^{(1)}$, $\mathbf{v}^{(1)}$).

Sampling Method. Originally, initialisation was purely random from uniform distributions: $x_{ij}^{(1)} = x_{\min\, ij} + U_{(0,1)}\,(x_{\max\, ij} - x_{\min\, ij})$. *Random Sampling* is easy to implement but does not usually result in good coverage of the search-space. More advanced sampling methods may be used, such as *Latin Hypercube Sampling*, *Orthogonal Sampling* or a range of different *Tesselations*.

Initial Conditions. Four types are proposed here:

1. *Stagnation*: $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} = \mathbf{xm}^{(1)}$
   This requires the *sampling* of each particle's position at the initial time-step ($\mathbf{x}^{(1)}$). Stagnation implies that the previous position $\mathbf{x}^{(0)} = \mathbf{x}^{(1)}$, and that the particle has converged to its attractor: $\mathbf{xm}^{(1)} = \mathbf{x}^{(1)}$. Thus, movement starts purely due to cooperation (no inertia, no individual attractor).
2. *Two Positions*: $\mathbf{x}^{(1)} \neq \mathbf{x}^{(0)}$ and either $\mathbf{xm}^{(1)} = \mathbf{x}^{(1)}$ or $\mathbf{xm}^{(1)} = \mathbf{x}^{(0)}$
   Two positions per particle are *sampled* and compared, with the better one becoming $\mathbf{x}^{(1)}$, the other becoming $\mathbf{x}^{(0)}$, and $\mathbf{xm}^{(1)} = \mathbf{x}^{(1)}$. Thus, movement starts both due to cooperation and to inertia (no individual attractor).
3. *One Position and One Memory*: $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} \neq \mathbf{xm}^{(1)}$
   Two positions per particle are *sampled* and compared, with the better one becoming $\mathbf{xm}^{(1)}$ and the other $\mathbf{x}^{(1)} = \mathbf{x}^{(0)}$. Movement starts due to both cooperation and acceleration towards its individual attractor (no inertia).
4. *Two Positions and One Memory* $\mathbf{x}^{(1)} \neq \mathbf{x}^{(0)} \neq \mathbf{xm}^{(1)} \neq \mathbf{x}^{(1)}$
   Three positions per particle are *sampled* and compared, with the best one becoming $\mathbf{xm}^{(1)}$. Thus, movement starts both due to all three sources: cooperation, inertia, and acceleration towards its individual attractor.

Initialised Variables Relation. For all *initial conditions* other than *stagnation*, more than one position is to be *sampled* per particle. The question is then whether these should be somehow related. Three alternatives are proposed here:

1. *Perturbation*: $\mathbf{x}^{(0)}$ is generated from controlled perturbations on $\mathbf{x}^{(1)}$. If applicable, $\mathbf{xm}^{(1)}$ is also generated from perturbations on $\mathbf{x}^{(1)}$.
2. *Independent*: Each population of positions is sampled independently.
3. *Simultaneous*: All populations of positions are sampled at once. For instance, if using the Latin Hypercube Sampling, there would be one single sampling with as many points as twice or three times the swarm size, as applicable.

3.2 Termination Conditions

The population-based nature of the method enables termination conditions different from the classical ones in numerical optimisation: 1) maximum number of iterations, and 2) convergence. Three types of conditions are identified here: 1) *based on search length* (or maximum number of iterations), 2) *based on clustering measures* (diversity loss), and 3) *based on measures of convergence*.

4 Individual Behaviour Features

These are the features of the algorithm which control the individual behaviour of a particle. Each particle has its own set of IBFs, which are viewed as particle attributes. The individual behaviour of a particle is materialised by its trajectory as it is pulled by its attractor. This is governed by a second order difference equation and the setting of its coefficients. The IBFs are grouped here in two main families, namely *Deterministic Features* and *Stochastic Features*.

4.1 Deterministic Features

Instead of viewing PSO as a *guided random search method*, it is viewed as a *randomly-weighted deterministic search method*. Thus, its desired deterministic behaviour is defined, adding only as much stochasticity as deemed beneficial.

By formulating the position update as in (5), it is clear that any given particle at any given time is pulled by a single attractor which results from a randomly weighted average of the components of the individual and social attractors. Thus, the *Trajectory Difference Equation* in (1) may be expressed as in (5).

$$iw_{ij}^{(t)} U_{(0,1)} \left(xb_{ij}^{(t)} - x_{ij}^{(t)} \right) + sw_{ij}^{(t)} U_{(0,1)} \left(xb_{kj}^{(t)} - x_{ij}^{(t)} \right) = \phi_{ij}^{(t)} \left(p_{ij}^{(t)} - x_{ij}^{(t)} \right) \quad (2)$$

$$\phi_{ij}^{(t)} = \iota_{ij}^{(t)} + \sigma_{ij}^{(t)} = iw_{ij}^{(t)}\ U_{(0,1)} + sw_{ij}^{(t)}\ U_{(0,1)} \quad (3)$$

$$p_{ij}^{(t)} = \frac{\iota_{ij}^{(t)}\ xb_{ij}^{(t)} + \sigma_{ij}^{(t)}\ xb_{kj}^{(t)}}{\phi_{ij}^{(t)}} \quad (4)$$

$$x_{ij}^{(t+1)} = x_{ij}^{(t)} + \omega_{ij}^{(t)} \left(x_{ij}^{(t)} - x_{ij}^{(t-1)} \right) + \phi_{ij}^{(t)} \left(p_{ij}^{(t)} - x_{ij}^{(t)} \right) \quad (5)$$

Trajectory Equation. Since we are dealing with a single particle, sub-index i is dropped. For simplicity, let us assume (ω, ϕ) constant in every dimension and $\forall t$, dropping sub-index j and super-index (t). If stochasticity is removed, the deterministic coefficients $(\hat{\omega}, \hat{\phi})$ are referred to as *Reference Trajectory Coefficients*.

CPSO Recurrence Formulation. The *CPSO Recurrence Formulation* is as in (6),

which is the deterministic version of (5). The generation of the overall attractor $\mathbf{p}_i^{(t)}$ is now decoupled, comprising a *Social Behaviour Feature* (SBF).

$$\boxed{x_{ij}^{(t+1)} = x_{ij}^{(t)} + \hat{\omega}\left(x_{ij}^{(t)} - x_{ij}^{(t-1)}\right) + \hat{\phi}\left(p_{ij}^{(t)} - x_{ij}^{(t)}\right)} \tag{6}$$

CPSO Closed-Form Formulation. This is obtained by solving the difference equation in (6). The roots of the characteristic polynomial are as in (7) and (8). The solution is per dimension and per particle (therefore dropping indices i and j), and the attractor $\mathbf{p}$ is stationary.

$$r_1 = \frac{1 + \hat{\omega} - \hat{\phi}}{2} + \frac{\gamma}{2} \quad ; \quad r_2 = \frac{1 + \hat{\omega} - \hat{\phi}}{2} - \frac{\gamma}{2} \tag{7}$$

$$\gamma = \sqrt{\hat{\phi}^2 - (2\hat{\omega} + 2)\,\hat{\phi} + (\hat{\omega} - 1)^2} \tag{8}$$

Case 1 ($\gamma^2 > 0$). The two roots of the characteristic polynomial are real-valued and different ($r_1 \neq r_2$). Therefore the closed-form for *Case 1* is as in (9).

$$\boxed{x^{(t)} = p + \frac{r_2\left(p - x^{(0)}\right) - \left(p - x^{(1)}\right)}{\gamma} r_1^t + \frac{-r_1\left(p - x^{(0)}\right) + \left(p - x^{(1)}\right)}{\gamma} r_2^t} \tag{9}$$

Case 2 ($\gamma^2 = 0$). The two roots of the characteristic polynomial are the same ($r_1 = r_2$), as shown in (10). Therefore the closed-form for *Case 2* is as in (11).

$$r = r_1 = r_2 = \frac{1 + \hat{\omega} - \hat{\phi}}{2} \tag{10}$$

$$\boxed{\begin{aligned} x^{(t)} &= p + \\ &\left[-\left(p - x^{(0)}\right) + \left(\left(p - x^{(0)}\right) - \frac{2\left(p - x^{(1)}\right)}{1 + \hat{\omega} - \hat{\phi}}\right) t\right] \left(\frac{1 + \hat{\omega} - \hat{\phi}}{2}\right)^t \end{aligned}} \tag{11}$$

Case 3 ($\gamma^2 < 0$). The two roots are complex conjugates.

$$r_1 = \frac{1 + \hat{\omega} - \hat{\phi}}{2} + \left(\frac{\gamma'}{2}\right) i \quad ; \quad r_2 = \frac{1 + \hat{\omega} - \hat{\phi}}{2} - \left(\frac{\gamma'}{2}\right) i \tag{12}$$

$$\gamma' = \sqrt{-\gamma^2} = \sqrt{-\hat{\phi}^2 + (2\hat{\omega} + 2)\,\hat{\phi} - (\hat{\omega} - 1)^2} \tag{13}$$

Using polar coordinates (ρ, θ), the closed-form for *Case 3* is as in (16).

$$\rho = \sqrt{\hat{\omega}} \quad ; \quad \theta = \operatorname{acos}\left(\frac{1+\hat{\omega}-\hat{\phi}}{2\sqrt{\hat{\omega}}}\right) \tag{14}$$

$$\cos(\theta) = \frac{1}{\sqrt{\hat{\omega}}}\left(\frac{1+\hat{\omega}-\hat{\phi}}{2}\right) \quad ; \quad \sin(\theta) = \frac{1}{\sqrt{\hat{\omega}}}\left(\frac{\gamma'}{2}\right) \tag{15}$$

$$\boxed{\begin{aligned} x^{(t)} &= p - \sqrt{\hat{\omega}}^{t}\left(p - x^{(0)}\right)\cos(\theta t) + \\ &\sqrt{\hat{\omega}}^{t}\left(\frac{\left(1+\hat{\omega}-\hat{\phi}\right)\left(p-x^{(0)}\right) - 2\left(p-x^{(1)}\right)}{\gamma'}\right)\sin(\theta t) \end{aligned}} \tag{16}$$

Thus, the chosen trajectory equation in RePSO may be given by the *Recurrence Formulation* in (6) or by the *Closed-Form Formulations* in (9), (11) and (16). Other recurrence formulations as well as some considerantions to be taken into account for the closed-form formulation are left for future work.

Reference Trajectory Coefficients Settings. An analysis of the trajectory closed-forms shows that the magnitude of the dominant root $r = \max(\|r_1\|, \|r_2\|)$ controls convergence. Fastest convergence occurs for $(\hat{\phi}, \hat{\omega}) = (1, 0)$, where $r = 0$ (see Fig. 3 (a)). The resulting convergence conditions are shown in (17), which define the area inside the convergence triangle ($r < 1$) shown in Fig. 3.

$$\boxed{\begin{aligned} &1 > \hat{\omega} > \frac{\hat{\phi}}{2} - 1 \\ &\hat{\phi} > 0 \end{aligned}} \tag{17}$$

Whilst the magnitude of the dominant root controls the speed of convergence, the existence and sign of the dominant root controls the *Type of Behaviour*:

1. *Oscillatory*: Roots are complex conjugates (no dominant root).
2. *Monotonic*: Dominant root is real-valued and positive.
3. *Zigzagging*: Dominant root is real-valued and negative.

These *Types of Behaviour* are bounded within specific *Sectors* in the $(\hat{\omega}, \hat{\phi})$ plane, each associated with one edge of triangular isolines (same r). These three *Sectors* are shown in Fig. 3 (b), where the white triangle separates the *Convergence* (inside) and *Divergence* regions. The settings of $(\hat{\omega}, \hat{\phi})$ can be chosen so as to achieve the desired behaviour and convergence speed. For example:

1. Choose *Type of Behaviour*: e.g. *Oscillatory*.
2. Set *Convergence Speed*: $\sqrt{\hat{\omega}} \in [0, 1]$, with fastest convergence for $\sqrt{\hat{\omega}} = 0$.
3. Set *Reference Acceleration Coefficient*: $\hat{\phi} \in \left(\left(\sqrt{\hat{\omega}}-1\right)^2, \left(\sqrt{\hat{\omega}}+1\right)^2\right)$.

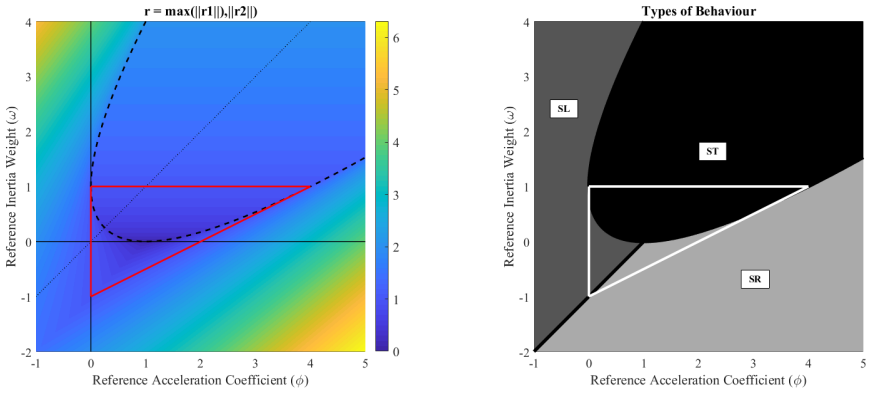


Fig. 3. On the left, magnitude of the dominant root. Settings inside red triangle ensure convergence ($r < 1$). On the right, *Sectors* for three *Types of Behaviour* in CPSO: black region is *Oscillatory*, dark grey region is *Monotonic*, and light grey region is *Zigzagging*.

4.2 Stochastic Features

The random weights in (1) affect the trajectory of a particle towards the overall attractor whilst also affecting its generation as a stochastic convex combination of the individual and the social attractors, as shown in (4). These two features are decoupled here. The *Stochastic Features* are concerned only with the former.

Stochastic Scaling. This refers to whether the stochastic variables in (5) are sampled once per particle position update (*vector scaling*) or resampled anew per dimension as well (*component scaling*). The former is often used by mistake.

Stochastic Sampling. In classical PSO, $\omega = \hat{\omega}$ (deterministic) whereas the probability distribution of ϕ results from the sum of two stochastic terms sampled from uniform distributions: $\phi = \iota + \sigma$ as in (3). If they are sampled from the same interval, the resulting distribution of ϕ is triangular. Otherwise, it is trapezoidal. In RePSO, the user is allowed to choose any distribution for (ω, ϕ).

Trajectory Coefficients Sampling Settings. Once the distributions have been chosen, the parameters defining them must be set. For example, $(\phi_{\min}, \phi_{\max})$ for a uniform distribution, or the standard deviation for a normal distribution.

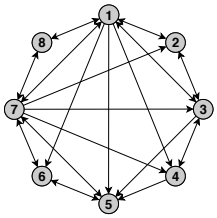
5 Social Behaviour Features

These are the features of the algorithm which control the social behaviour of a particle. Despite being SBFs, they are defined as *Particle Attributes* in RePSO. A particle's social behaviour is governed by its access to other particles' memories (*Local Sociometry*) and by how it handles this information (social influence).

5.1 Local Sociometry

In classical PSO, the sociometry is a global feature. It can be defined as a regular graph, or irregulary by defining one connection at a time. In the latter case, the structure cannot be automatically generated nor is it scalable. In RePSO, a *Local Sociometry* is defined for each particle, with the *Global Sociometry* resulting from their assembly. This has the advantange that sociometry is a particle attribute, facilitating object-oriented implementation. Also that different social behaviours can be exhibited by different particles, and that irregular global sociometries are possible without renouncing automation or scalability.

The Local Sociometry is generated by defining the *Neighbourhood Topology* and the *Neighbourhood Extent*. Examples of the former are the *Global*, *Ring*, *Forward* and *Wheel* topologies. The *Topology* defines a methodology to generate connections from the particle informed to its informers. The *Extent* defines the neighbourhood size (number of neighbours, distance of influence). An example of an emergent unstructured neighbourhood is shown in Fig. 4, where the Local Sociometry of particle 1 is the *Global* topology whilst that of particle 2 is the *Ring* topology. Other aspects may be considered, such as whether a particle's memory is part of its neighbourhood (X in the connectivity matrix in Fig. 4).



(a) Sociometry

	1	2	3	4	5	6	7	8
1	X	1	1	1	1	1	1	1
2	1	X	1	0	0	0	0	0
3	1	1	X	1	1	0	0	0
4	0	0	1	X	1	0	0	0
5	0	0	0	0	X	1	1	0
6	0	0	0	0	1	X	1	0
7	1	1	1	1	1	1	X	1
8	1	0	0	0	0	0	1	X

(b) Connectivity Matrix

Fig. 4. Unstructured sociometry emerging from local sociometries.

5.2 Current Information Update

Any particle holds two types of information: *current* and *memorised*. The update of the former takes place by gathering information, generating an overall attractor using the information gathered, and applying the trajectory equation. A particle may access the information currently held, the one memorised, or both from its neighbours (*Information Gathering*). This is an extension to classical formulations, where a particle can only access their memorised information.

5.3 Memorised Information Update

This controls the update of a particle's memory when it accesses new information. This is performed directly rather than through a trajectory equation. The question is what *Type of Information* is accessible to a particle's memory.

Another feature affecting this update is the *Synchrony*, which defines whether a particle's memory is updated immediately after its currently held information is updated (asynchronous) or only after the currently held information of every particle is updated (synchronous). Typically, the update is synchronous.

RePSO also proposes to include the CHTs here. Thus, different particles may have different CHTs, and therefore may value a given location differently.

6 Conclusions

A general framework has been proposed aiming to encompass many variants of the PSO algorithm under one umbrella so that different versions may be posed as settings within the proposed unified framework. In addition, some extensions to the classical PSO method have been made such as the decoupling of the stochasticity that affects both the acceleration coefficient (ϕ) and the generation of the overall attractor; an extended treatment of the swarm initialisation; the particle trajectory closed forms; the identification of three types of deterministic behaviour to inform the setting of the control coefficients; and the global sociometry resulting from assembling local sociometries defined as particle attributes. Due to space constraints, most of these features are discussed only superficially.

References

1. Blackwell, T., Kennedy, J.: Impact of communication topology in particle swarm optimization. IEEE Trans. on Evolutionary Computation **23**(4), 689–702 (2019)
2. Bonyadi, M., Michalewicz, Z.: Impacts of coefficients on movement patterns in the particle swarm optimization algorithm. IEEE Transactions on Evolutionary Computation (2017)
3. Campana, E.F., Fasano, G., Pinto, A.: Dynamic analysis for the selection of parameters and initial population, in particle swarm optimization. Journal of Global Optimization **48**, 347–397 (2010)
4. Cleghorn, C.W., Engelbrecht, A.P.: Particle swarm variants: standardized convergence analysis. Swarm Intelligence **9**(2-3), 177–203 (2015)
5. Clerc, M., Kennedy, J.: The particle swarm – explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation **6**(1), 58–73 (2002)
6. Coello Coello, C.A.: Use of a self-adaptive penalty approach for engineering optimization problems. Computers in Industry **41**(2), 113–127 (2000)
7. Dhal, K.G., Das, A., Sahoo, S., Das, R., Das, S.: Measuring the curse of population size over swarm intelligence based algorithms. Evolving Systems (2019)
8. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Micro Machine and Human Science, 1995. MHS '95., Proceedings of the Sixth International Symposium on. pp. 39–43 (1995)
9. Fernández Martínez, J.L., García Gonzalo, E.: The pso family: deduction, stochastic analysis and comparison. Swarm Intelligence **3**(4), 245 (2009)
10. García-Gonzalo, E., Fernández-Martínez, J.L.: Convergence and stochastic stability analysis of particle swarm optimization variants with generic parameter distributions. Applied Mathematics and Computation **249**, 286–302 (2014)

11. Helwig, S., Wanka, R.: Theoretical analysis of initial particle swarm behavior. In: PPSN 2008. pp. 889–898 (2008)
12. Hu, X., Eberhart, R.: Solving constrained nonlinear optimization problems with particle swarm optimization. In: Proceedings of the 6th World Multi-conference on Systemics, Cybernetics and Informatics (SCI 2002) (2002)
13. Innocente, M.S., Afonso, S.M.B., Sienz, J., Davies, H.M.: Particle swarm algorithm with adaptive constraint handling and integrated surrogate model for the management of petroleum fields. Applied Soft Computing **34**, 463–484 (2015)
14. Innocente, M.S., Sienz, J.: Particle swarm optimization with inertia weight and constriction factor. In: Proceedings of the 2011 International conference on swarm intelligence (ICSI 2011). pp. id–1–id–11 (2011)
15. Innocente, M., Sienz, J.: Pseudo-adaptive penalization to handle constraints in particle swarm optimizers. In: Proceedings of the Tenth International Conference on Computational Structures Technology. Civil-Comp Press
16. Jordehi, A.R.: A review on constraint handling strategies in particle swarm optimisation. Neural Computing and Applications **26**(6), 1265–1275 (2015)
17. Kadirkamanathan, V., Selvarajah, K., Fleming, P.: Stability analysis of the particle dynamics in particle swarm optimizer. IEEE Transactions on Evolutionary Computation **10**(3), 245–255 (2006)
18. Kazimipour, B., Li, X., Qin, A.K.: A review of population initialization techniques for evolutionary algorithms. In: IEEE Congress on Evolutionary Computation (CEC). IEEE (2014)
19. Kazimipour, B., Li, X., Qin, A.K.: Why advanced population initialization techniques perform poorly in high dimension? In: Lecture Notes in Computer Science, pp. 479–490. Springer International Publishing (2014)
20. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks. vol. 4, pp. 1942–1948 (1995)
21. Kwok, N., Liu, D., Tan, K., Ha, Q.: An empirical study on the settings of control coefficients in particle swarm optimization. In: Proceedings of the 2006 IEEE Congress on Evolutionary Computation (CEC 2006). pp. 823–830 (2006)
22. Liu, J., Liu, H., Shen, W.: Stability analysis of particle swarm optimization. In: Huang, D.S., Heutte, L., Loog, M. (eds.) Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence, Lecture Notes in Computer Science, vol. 4682, pp. 781–790. Springer Berlin / Heidelberg (2007)
23. Liu, Q., Wei, W., Yuan, H., Zhan, Z.H., Li, Y.: Topology selection for particle swarm optimization. Information Sciences **363**, 154–173 (2016)
24. Lynn, N., Ali, M.Z., Suganthan, P.N.: Population topologies for particle swarm optimization and differential evolution. Swarm and Evolutionary Computation **39**, 24 – 35 (2018)
25. Ozcan, E., Mohan, C.: Particle swarm optimization: surfing the waves. In: Proceedings of the IEEE Congress on Evolutionary Computation. vol. 3 (1999)
26. Piotrowski, A.P., Napiorkowski, J.J., Piotrowska, A.E.: Population size in particle swarm optimization. Swarm and Evolutionary Computation **58**, 100718 (2020)
27. Poli, R.: Mean and variance of the sampling distribution of particle swarm optimizers during stagnation. IEEE Trans. on Evolutionary Computation **13**(4) (2009)
28. Shi, Y., Eberhart, R.: Empirical study of particle swarm optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation. vol. 3 (1999)
29. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: Proceedings of the IEEE International Conference on Evolutionary Computation (1998)
30. Trelea, I.C.: The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters **85**(6), 317–325 (2003)