Particle swarm optimisation for dynamic optimisation problems: A review

Article *in* Neural Computing and Applications · December 2014

DOI: 10.1007/s00521-014-1661-6

1 author:

Ahmad rezaee jordehi
Islamic Azad University of Rasht
81 PUBLICATIONS **4,485** CITATIONS

SEE PROFILE

Neural Comput & Applic (2014) 25:1507–1516
DOI 10.1007/s00521-014-1661-6

REVIEW

Particle swarm optimisation for dynamic optimisation problems: a review

Ahmad Rezaee Jordehi

Received: 18 January 2014 / Accepted: 5 July 2014 / Published online: 24 July 2014

Abstract Some real-world optimisation problems are dynamic; that is, their objective function and/or constraints vary over time. Solving such problems is very challenging. Particle swarm optimisation (PSO) is a well-known and efficient optimisation algorithm. In this paper, the PSO variants, devised for dynamic optimisation problems, are reviewed. This is the first comprehensive review that is conducted on PSO variants in dynamic environments. The author believes that this paper can be useful for researchers who intend to solve dynamic optimisation problems.

Keywords Particle swarm optimisation · Optimisation · Dynamic optimisation problem

1 Introduction

There are very diverse optimisation problems in different areas of science and engineering [1–6]. For solving them, there exist two groups of approaches: classical approaches and heuristic approaches. Classical approaches are not generally efficient in solving complex optimisation problems [7–9], because they entail preconditions such as continuity, convexity and differentiability of objective functions that usually are not met [10].

Heuristic approaches encompass a lot of approaches such as genetic algorithms, imperialistic competitive algorithm, krill herd optimisation, cuckoo search optimisation and teaching–learning-based optimisation. Heuristics do not demonstrate most of the drawbacks of classical approaches. Particle swarm optimisation (PSO) is a heuristic optimisation algorithm that is very common in solving optimisation problems.

PSO is a population-based optimisation technique devised by Kennedy and Eberhart [11]. It belongs to the family of swarm intelligence computational paradigms and is inspired of social interaction in birds [12–16].

Some real-world optimisation problems are dynamic, meaning that their objective function or constraints vary over time, whereas basic PSO variants are only able to solve static (non-dynamic) optimisation problems. Therefore, for enabling PSO to solve dynamic problems, some modifications should be applied. In this paper, the existing PSO-based strategies for solving dynamic optimisation problems are reviewed. The paper is organised as follows; in Sect. 2, overview of PSO is given. In Sect. 3, the existing PSO-based strategies for solving dynamic problems are reviewed. Concluding remarks are made in Sect. 4.

2 Overview of PSO

PSO is launched with random initialisation of a swarm of individuals particles in the n-dimensional search space (n is the dimension of problem) [12]. The particles flyover search space with adjusted velocities. In PSO, each particle keeps two values in its memory; its own best experience, whose position and objective value are called P_i and P_{best}, respectively, and the best experience of the whole swarm, whose position and objective value are called P_g and g_{best}, respectively. The position and velocity of ith particle are represented as follows.

$$\begin{aligned} X_i &= (X_{i1}, X_{i2}, \ldots, X_{id}, \ldots, X_{in}) \\ V_i &= (V_{i1}, V_{i2}, \ldots, V_{id}, \ldots, V_{in}) \end{aligned} \quad (1)$$

A. Rezaee Jordehi (✉)
Department of Electrical Engineering, University Putra Malaysia, Serdang, Malaysia
e-mail: ahmadrezaeejordehi@gmail.com

The velocities and positions of particles are updated in each time step based on the following equations.

$$V_{id}(t+1) = \omega V_{id}(t) + C_1 r_{1d}(P_{id} - X_{id}) + C_2 r_{2d}(P_{gd} - X_{id}) \quad (2)$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \quad (3)$$

where ω is a coefficient named inertia weight, C_1 and C_2 are cognitive and social acceleration coefficients, respectively, and r_{1d} and r_{2d} are random numbers in [0,1].

The steps of PSO are as follows [12].

1. Particles' velocities and positions are initialised.
2. Particles' velocities and positions are updated by Eqs. (2) and (3).
3. Objective values are computed for all particles
4. Each particle's P_{best} and P_i are updated.
5. P_g and g_{best} are updated.
6. Steps 2–5 are iterated till stopping criterion is met.

3 PSO for dynamic environments

The general representation of a dynamic optimisation problem (DOP) is as follows.

$$\begin{array}{ll} \text{Minimize} & f(X,t) \\ \text{Subject to} & g(X,t) \leq 0 \end{array} \quad (4)$$

In DOP's due to the change of objective functions or constraints, the position and value of optima vary over time. The role of optimisation technique is to track the changing optimum/optima. If the change is radical, the best option is to start optimisation process from scratch [17]. However, in most practical instances, the changes are gradual. If the change is gradual, the current data gathered by optimisation process can be utilised to track new optimum/optima.

There are two main difficulties that should be addressed in DOPs:

1. Outdated memory: When the environment changes, previously good solutions may no longer be good and mislead the swarm towards false optima. This issue is more acute when the environment change is severe.
2. Diversity loss: If the environment changes when the new optimum is within the collapsing swarm, it is expected that new optima can be traced successfully and promptly. However, if the new optimum is outside the swarm's expansion, particles' low velocities inhibit re-diversification and the swarm may even oscillate around a false attractor in a phenomenon called linear collapse, so the new optimum is difficult to be traced.

3.1 Addressing challenges in DOP's

For addressing the challenges in dynamic environments, three following crucial actions should be undertaken:

3.1.1 Detecting change

In many applications, the time of change occurrence is known to the system; otherwise, it should be detected. Change detection strategy should successfully detect the change and trigger response mechanism. In [18], two major types of change detection mechanisms, namely population-based and sensor-based mechanisms have been introduced.

In population-based mechanisms, the fitness evaluations of the population is used for detecting the changes, while in the latter approach, additional measurement of the landscape's fitness on prescribed points is used for change detection. In population-based mechanisms, the sets $S(t)$ and $S(t+1)$ are formed consisting of fitness functions of individuals at iterations t and $t+1$, respectively. Then, the change detection problem is transformed into the problem of testing whether the data set $S(t)$ and $S(t+1)$ are coming from different distributions or not. For judging about this, a statistical hypothesis testing method is employed.

On the other hand, sensor-based approach is based on conducting additional measurements in the fitness landscape using fitness landscape sensors. If any of the sensors detects an altered fitness value, the change is considered to be occurred. The advantage of this approach in comparison with population-based approach is that it does not need elaborated statistical analysis and there can be no false positives; however, additional fitness evaluation is required.

In [19], artificial immune system is employed for change detection. A negative selection algorithm has been used to find whether the fitness landscape has changed or not. This is solely done with fitness information from the population on a sample base. Also in [20], a change detection strategy for constrained environments has been put forward.

Despite the existence of all above-mentioned efficient change detection strategies, in dynamic PSO literature, change detection is usually done by re-evaluating one or more personal bests, and if the fitness value of at least one of them has changed, the change occurrence is concluded. In this paper, during the explanation of various dynamic-based PSO variants, their change detection schemes will also be explained.

3.1.2 Memory update

For solving outdated memory problem, mostly either re-evaluating or forgetting the memory is used. In the latter one, each particle's memory is set to the particle's current position and the global best is updated as $P_g = \arg\min f(P_i)$.

3.1.3 Diversity enhancement

Diversity loss is considered as the most concern in DOPs. There are 4 principle mechanisms for diversity enhancement: re-diversification, repulsion, multi-populations and dynamic neighbourhood topology.

3.2 Common benchmarks and metrics in DOP literature

Now, before explaining various PSO variants for DOP's, benchmarks and metrics that are commonly used in specialised DOP literature are introduced.

3.2.1 Moving peaks benchmark (MPB)

These benchmarks that somehow represent real-world DOPs are commonly used in the specialised literature. In MPB problems, the optima can vary in three terms: the height, width and location of peaks. MPB is defined as Eq. (5) [21].

$$F(X,t) = \text{Max}\frac{H_i(t)}{1 + W_i(t)\sum_{d=1}^{n}(X_d(t) - X_{id}(t))^2} \quad \text{for } i = 1,\ldots,p \tag{5}$$

where H_i (t) and W_i (t) are, respectively, height and width of ith peak at time t, X_{id} represents the dth dimension of position vector of ith peak, n is the number of decision variables, and p denotes the number of peaks. The position of each peak is shifted in a random direction as follows.

$$\begin{aligned} X_i(t) &= X_i(t-1) + V_i(t) \\ V_i(t) &= \frac{s}{r + V_i(t+1)}((1-\lambda)r + \lambda V_i(t+1)) \end{aligned} \tag{6}$$

where V_i (t) is called shift vector and is a linear combination of a random vector r and the previous shift vector $V_i\,(t-1)$, and λ is called correlation factor and indicates the level of correlation between two successive environmental changes. The final parameter is U indicating that the environment change happens by every U function evaluations.

3.2.2 Performance metric

For measuring the performance of a dynamic optimisation technique, commonly a metric called offline error is used. It is defined by Eq. (7).

$$e = \frac{1}{z}\sum_{k=1}^{z}(h_k - f_k) \tag{7}$$

where f_k is the best solution, achieved just before the kth environmental change and h_K is optimum value of the kth environment, and z is the total number of environmental changes.

3.3 Classification of PSO variants adapted for DOP's

Here, due to the importance of diversity-enhancement strategies, all different variants for DOPs are classified according their diversity-enhancement schemes. According to this criterion, most of them are classified into five main groups. In this paper, the diversity-enhancement schemes which could not be assigned to any of these five groups will be analysed under name of "other variants" in Sect. 3.3.5.

3.3.1 Repulsion-based variants

A constant degree of swarm diversity can be maintained at all times through some repulsive strategies. One repulsive strategy is to use charged particles in swarm wherein diversity is maintained by coulomb repulsion among particles. In one type, called charged PSO, all particles are charged while in the second type, called atomic PSO, just some particles are charged and others are neutral particles. Charged particles through diversity enhancement help PSO to trace new optimum [22–26]. In [23], the performance of conventional PSO, charged PSO and atomic PSO in solving DOP's are assessed. The experiments have proved the superiority of atomic PSO over conventional and charged PSO's. In [25], dynamic problems are categorised based on their severity, and for each category, a certain PSO variant among conventional, charged and atomic PSO is recommended.

In [24] and [26–29], quantum particles are employed for diversity enhancement. Quantum particles are re-positioned in each iteration within a hypersphere of r_{cloud} centred on the P_g according to a specified probability distribution. In [28], the centre of position distribution is the personal best of a randomly selected particle rather than P_g, and in [29], a mutation operator is applied on g_{best} to enhance diversity further. In the literature, various probability distributions including Gaussian, uniform volume and non-uniform volume are used for quantum particles. In comparison with charged particles, quantum particles behave better due to lower complexity and easy controllability.

3.3.2 Re-randomisation-based variants

In [30], for responding to the change, re-randomisation of part or whole of the swarm, and re-randomisation of g_{best} and resetting the whole particles are conducted. The experiments show that lower re-randomisation rates outperform higher re-randomisation rates when change severity is low while there is no significant difference when the change severity is high. For change detection, in contrary to changed-g_{best} value method in [31], fixed-g_{best} value method is used wherein the g_{best} value and the second

g_{best} value are monitored. If they do not change during a certain number of iterations, it is concluded that the change has occurred.

3.3.3 Dynamic neighbourhood topology-based variants

In [32] and [33], hierarchical PSO and also its extension called partitioned hierarchical PSO (PH-PSO) have been applied to DOP's in the hope that these variants would be more compatible with dynamic environments due to their dynamic neighbourhood topology. In PH-PSO, after change detection which is done by changed-g_{best} value method, the hierarchy is partitioned into a set of sub-hierarchies or sub-swarms. These sub-swarms continue to search for the optimum independently. After a certain number of iterations, the sub-swarms are reunified by connections of the hierarchy that have been cut. In a more extended variant called adaptive PH-PSO, the number of these iterations is adapted. In [33], a new change detection strategy called hierarchy monitoring strategy is proposed. In this strategy, at each iteration, the total number of swaps is measured and if it exceeds a certain threshold, the change is considered to have occurred. The rationale behind this is that when change occurs, the number of particles swapped between nodes of tree drastically increases. The advantage of this strategy is that it does not need additional function evaluations. The experiments reveal that PH-PSO outperforms H-PSO and conventional PSO.

Nevertheless, besides the above-mentioned dynamic neighbourhood topologies, some static topologies have also been utilised for tackling DOP's. In [34], in a variant named fine-grained PSO, Von-Neumann neighbourhood topology is adopted wherein each particle is neighboured to the four particles in its four sides. This neighbourhood topology provides slower information propagation in comparison with g_{best} PSO, leading to higher diversity level. The experiments strongly approve its superiority. Furthermore, in [35], the swarm is partitioned into two sub-swarms: a sub-swarm with global neighbourhood topology responsible for finding new optima and another sub-swarm with local neighbourhood topology responsible for compensating the diversity loss. The two sub-swarms exchange their best particles in some checkpoints. Moreover, a mutation operator is applied to g_{best} to ensure enough diversity.

3.3.4 Multi-swarm variants

The rationale behind multi-swarm PSO's is to partition the swarm into a number of sub-swarms in order to position each sub-swarm on a different peak of landscape. So many multi-swarm variants for tackling DOP's have been proposed in the literature. In this section, they will be reviewed.

3.3.4.1 Multi-swarm charged/quantum PSO In this variant, the swarm is partitioned into a number of sub-swarms in order to position each sub-swarm on a different promising peak of the landscape. Here, in addition to multi-swarm concept, three following diversity operators are adopted [26, 36].

1. Quantum or charged particles: diversity loss caused by environmental change is counterbalanced by using charged or quantum particles. In charged multi-swarm PSO, consisting of neutral and charged particles, diversity is maintained by coulomb repulsion among charged particles. In quantum multi-swarm PSO, consisting neutral and quantum particles, the quantum particles are randomised within a ball of radius r_{cloud} centred on the sub-swarm attractor [26, 36].
2. Anti-convergence: when all sub-swarms have converged, or in other words, when their neutral swarm size gets less than convergence radius, r_{conv}, anti-convergence operator expels the worst sub-swarm from its peak and reinitialises it in the search space. Therefore, at each time, there is at least one sub-swarm patrolling search space for new peaks [26, 36].
3. Exclusion: when the attractor of two sub-swarms are within an exclusion radius, r_{excl}, the sub-swarm with worse objective function is expelled and reinitialised in the search space. Actually, exclusion is a local interaction between colliding swarms, preventing them from settling on the same peak [26, 36].

The experiments on benchmark functions show that the above-mentioned multi-swarm PSO behaves well but its main issue is the large number of the parameters to be set. The number of sub-swarms, the number of quantum particles, quantum cloud radius (or charge in charged multi-swarms), exclusion distance and convergence radius are the parameters that should be tuned.

3.3.4.2 Extensions and improvements of multi-swarm charged/quantum PSO After introduction of multi-swarm charged/quantum PSO, a high amount of research effort has been put to extend it and improve its computational behaviour. The improvements and extensions are explained below.

3.3.4.2.1 Adaptation of the number of sub-swarms and exclusion radius In [37], self-adaptation strategies for the number of sub-swarms and also exclusion radius have been presented.

The rationale behind this extension is that in multi-swarm PSO, if all current sub-swarms are converging, a

new free (patrol) sub-swarm is required, and on the other side, too many free sub-swarms highly increase computational burden and should be skipped. Therefore, at each iteration, the number of sub-swarms is set by Eq. (8) [37].

$$M(t) = \begin{cases} M(t+1)+1 & \text{if } M_{\text{free}} = 0 \\ M(t+1)-1 & \text{if } M_{\text{free}} > n_{\text{excess}} \end{cases} \quad M(0) = 1 \tag{8}$$

where n_{excess} denotes the desired number of free sub-swarms.

When the number of sub-swarms is adapted, assuming that M corresponds to the number of peaks, exclusion radius r_{excl} is adapted via the following equation [37].

$$r_{\text{excl}} = \frac{X}{2M^{1/d}} \tag{9}$$

where X is the length of search space in each dimension.

3.3.4.2.2 Particle conversion-based multi-swarm PSO Since quantum particles are mainly useful during environment change, in [36], it is proposed that one iteration after environment change, all neutral particles are converted to quantum particles and after this iteration, they are reverted back to neutral particles. This strategy diminishes the number of quantum particles and leads to better computational behaviour.

3.3.4.2.3 Finding optimum proportion of diversifier particles in multi-swarm PSO In [38], the optimum proportions of quantum and neutral particles in multi-QPSO have been determined. The experiments reveal that higher number of neutral particles (than quantum particles) is the best strategy. Experiments also show that quantum particles have their maximal influence on their corresponding sub-swarm's best in the first iteration after environmental change, while the influence of neutral particles increases over time.

3.3.4.2.4 Two rules for enhancement of multi-swarm PSO In [39–41], two various rules for enhancing multi-QPSO have been introduced.

- Rule 1: When the environment changes temporarily, the number of quantum particles is increased and the number of neutral particles is decreased. This leads to a high burst of local diversity to all sub-swarms during a short period of time.
- Rule 2: If a sub-swarm performs undesirably, it is relocated randomly in the whole search space or it is paused if there is not enough time. This rule prevents wasting function evaluations and enhances diversity.

The experiments show that both rules lead to an improvement in multi-QPSO, although the performance of the second rule is better. Surprisingly, the simultaneous application of both rules does not result in any improvement in multi-QPSO.

3.3.4.2.5 Exclusion radius adaptation In [42], exclusion radius is adapted based on environmental situation and particles' density. All the present particles are clustered, and then, the mean of minimum distance between each cluster and other clusters is set as r_{excl}.

3.3.4.2.6 Trajectory control in multi-swarm PSO In this variant, if a particle fails to create a better fitness value, its number of failure (N_{fail}) is incremented and when N_{fail} reaches a prespecified threshold, particle's position and velocity will be updated via following equations [43].

$$X(t+1) = P_g \tag{10}$$

$$V(t+1) = \frac{r.V(t)}{2} \tag{11}$$

where P_g represents the position with the best fitness value in the corresponding sub-swarm and r is a random number in (0,1).

Above-mentioned equations imply that particles moving in trajectories with consecutive failures will be positioned around the best position of their corresponding swarm. The efficiency of this variant in dealing with diverse dynamic problems has not been tested.

3.3.4.3 Species-based PSO Species-based PSO was originally introduced to solve multi-modal optimisation problems. It dynamically distributes particles into species. Each species is centred by its best particle, called species seed. All particles that fall within radius r_s from the species seed belong to the same species. Each species seed is considered as the neighbourhood best for all particles in that species. The species seeds are updated at each iteration [36, 44–46].

In an extended version of species-based PSO, the number of particles of each species is bounded to a maximum value. By crowding too many particles in a certain species, some promising regions of search space may not be explored efficiently. Bounding the number of particles of species hinders this problem.

Species-based PSO possesses two important aspects that make it potentially well suited for tackling with dynamic problems, especially multi-modal dynamic problems. The first aspect is that since it updates species at each iteration, immediately after change, new species are formed according to new environment, whereas in multi-swarm PSO, after environmental change, the members of sub-swarms do not change. The second promising aspect of species-based PSO is that the number of species is not required to be set a priori, but it is dynamically adjusted during the run, so the concerns for setting number of sub-swarms in multi-swarm PSO and other multi-population-based approaches do not exist here.

For adapting species-based PSO to dynamic environments, three following adaptations have been proposed in the literature [36, 44–46].

1. One iteration after change, all conventional particles are converted to quantum particles in a trigger-based manner and are reverted back to conventional particles after one iteration.
2. Anti-convergence operator is adopted to re-initialise the worst species
3. The distribution of quantum particles is centred around the centre of species seeds and particle positions not around species seeds.

For detecting change, the P_i's of a specified number of best species seeds are re-evaluated at each iteration. If any of them changes, an environmental change is concluded to be detected. For memory update, all particles personal bests are reset to their corresponding current positions.

3.3.4.4 Clustering-based PSO In [47], a hierarchical clustering approach has been used. It starts from an initial swarm called cradle swarm. The global search method is implemented for a specified number of iterations so that the whole population is well distributed in different subregions, and then, rough clustering is implemented to create temporary sub-swarms till the cradle swarm gets empty. For each temporary sub-swarm, global search method is implemented for that specified number of iterations. Afterwards, refining clustering is started to create final subswarms, and then, local search is implemented on all final sub-swarms. Also, overlapping, convergence and overcrowding checks are conducted at each iteration.

In [48], refining clustering and training process have been removed from the algorithm that leads to lower computational burden.

For detecting change, changed-g_{best} value is applied, and for response, a restart scheme is applied, while the best positions found in previous environment are reserved.

In overall, despite some of it capabilities, clustering-PSO is not capable of creating accurate sub-swarms, particularly when a single peak is covered by one particle.

3.3.4.5 Child and parent swarms PSO In [49], the PSO population consists of one parent swarm and some child swarms. The parent swarm implements explorative actions in order to find promising regions of search space, while child swarms exploit the discovered promising regions. After change detection, particles are temporarily converted to quantum particles. Two strategies for avoiding redundant search in the same area are adopted; according to the first strategy, if a parent swarm overlaps with a child swarm, it is re-initialised and according to the second one, if two child swarms overlap, the worse swarm is removed.

In an extension of [49], after tracking the optima of changed environment, the fruitless child swarms hibernate until next environmental change [50]. This strategy hinders wasting of fitness evaluations, so alleviates computational complexity.

3.3.4.6 Fast multi-swarm PSO This variant adapts fast PSO, proposed in [51], to dynamic environments. It consists of two types of swarms. Parent swarm for search space exploration and child swarms for exploitation. For implementation of fast multi-swarm PSO, firstly, the parent swarm is initialised randomly and evolved by fast evolutionary programming. Then, following steps are iterated till termination criterion is met [52].

- If the best particle of the parent swarm improves, a child swarm is created around the best particle.
- The parent swarm is updated, and if any of its particles is within the area of a child swarm, it is moved into that child swarm and a new particle is randomly created in the parent swarm.
- The child swarms are evolved by PSO, and according to the improvement in their best value, they may be subject to mutation operator.
- Each child swarm is updated for overlapping check.

The downside of this variant is that its performance is too sensitive to the radius of child swarms and no strategy has been presented for setting it.

3.3.4.7 Triggered memory-based PSO In this variant, an enhanced memory scheme is applied to tri-island PSO framework. In tri-island PSO, the whole swarm is partitioned into three sub-swarms: explorer sub-swarm, memory sub-swarm and exploiter sub-swarm [46]. While in conventional memory scheme, explorer particles are initialised and the memory is retrieved after each environmental change, in trigger memory-based scheme, initialisation of explorer particles and memory retrieval occur once a peak has been discovered.

3.3.4.8 Multi-strategy ensemble PSO In this variant, the swarm is partitioned into two parts. Particles in part I provide exploitative capability and fly according to conventional flight equations while the best position of both parts is considered as their P_g [53].

The role of particles in part II is to provide required diversity for dynamic problems. According to Eq. (12), they patrol around particles of part I in order to cover the regions which are not covered by part I.

$$V_{id}(t+1) = Sgn(r_3 - 0.5)(\omega V_{id}(t) + C_1 r_{1d}(P_i - X_{id}) + C_2 r_{2d}(P_{xd} - X_{id})) \quad (12)$$

where x is the index of a randomly selected particle in part I. According to (12), each particle in part II, with probability 0.5 flies to gets closer to the personal best of a randomly selected particle of part I and with the same probability, flies to get farther away from it.

For change detection, five sentry particles are employed and change in each one is considered as environmental change.

3.3.4.9 PSO with composite particles This variant is inspired from the notion of composite particles in physics. It divides the swarm into some composite particles. For constructing composite particles, at each iteration, a list containing all particles in increasing order of fitness value is created. Firstly, the worst particle of list is selected. Then, two particles in the list which are the closest to the first one are added to form a triangular-shaped composite particle. This process continues till all particles in list are located in different composite particles.

The crucial point in this variant is that contrary to the information sharing among different composite particles that done by conventional PSO, the interaction among particles within a composite particle does not obey conventional PSO model, but it is implemented via following two operators [54, 55].

- VAR scheme

This scheme is based on replacing the worst particle with another reflection point so that the new composite particle helps to track promising peaks in new environment [54, 55]. In this scheme, reflection step size is a crucial parameter. In [55], an adaptation strategy for reflection step size is presented.

- Integral movement

This operator conveys the velocity of a pioneer particle (the best elementary particle of a composite particle) to the other two elementary particles in each composite particle, so enhances less-fit particles. Furthermore, for enhancing diversity among particles in a composite particle, a special repulsion mechanism has been proposed.

The experiments show the promising performance of this variant in DOPs. Nevertheless, it is a highly complex approach in comparison with other multi-swarm variants.

3.3.4.10 Forking PSO This variant introduces a forking mechanism wherein a main swarm explores for new peaks and some child swarms are created adaptively to exploit the discovered peaks [56]. At each iteration, a convergence index γ is computed for main swarm by Eq. (13).

$$\gamma = \max_i \{D(X_i, P_g)\} \tag{13}$$

where D represents the Euclidean distance.

If γ is below a prespecified threshold, the forking condition is met meaning that a peak has been discovered, and then, some best-fit particles in main swarm create a new child swarm to exploit the achieved peak, and the rest of the particles will be reinitialised randomly as a new main swarm to explore the search space continuously. Furthermore, exclusion operation is effective between the child swarms and between main swarm and child swarms.

3.3.4.11 Multidimensional multi-swarm PSO In [57], a technique called multidimensional PSO modifies the regular structure of PSO so that inter-dimensional passes is implemented with a special dimensional PSO process. Furthermore, fractional global best formation (FGBF) technique has been proposed which gathers all the best dimensional components and fractionally produces a superior artificial g_{best} [57]. Application of the two mentioned strategies has led to promising results in dynamic environments.

3.3.4.12 Dynamic niching PSO In dynamic niching PSO (DNPSO), the particles are partitioned into some sub-swarms and a group of free particles [58]. The sub-swarms are generated and are dynamically removed at each iteration, while the number of particles of each sub-swarm is changing. The experiments show promising behaviour of DNPSO in dynamic optimisation.

3.3.4.13 DE-PSO (Collaborative PSO) In this variant, the whole population is split into two equal sub-populations. The first sub-population uses crowding-based differential evolution in order to maintain diversity and the second one is a PSO sub-population, well suited for tracking the optimum [59].

3.3.4.14 Swarm core evolutionary PSO (SCEPSO) In this variant, the swarm is partitioned into three sub-swarms called core, near and far sub-swarms. The core sub-swarm adopts the concepts of evolutionary programming and ensures sufficient searching near the optimum. The particles of near sub-swarm are specialised in finding new personal and swarm bests, and far sub-swarm particles provide an appropriate level of diversity [60]. The experiments show the efficiency of this variant in solving DOP's.

3.3.5 Other variants

3.3.5.1 Mutation-aided PSO In this variant, a dynamic micro-mutation is invoked to counterbalance the loss of diversity after environmental change. The variant provides high diversity immediately after change to explore promising regions in new environment, then, gradually

decreases the diversity so that an appropriate fine tuning can be implemented. Therefore, right after change detection, the mutation is applied with probability $P_{\max}$ and the mutation probability is decreased linearly with the iteration number till it reaches a lower bound, $P_{\min}$, right before next change [61, 62].

$$P_m = \frac{(P_{\max} - P_{\min}) \cdot (E - t)}{E} \tag{14}$$

where P_m denotes mutation probability, E is the distance between two successive changes, and t is the number of iterations after last change.

3.3.5.2 Activity factor-based PSO In this variant, a portion of particles are set as active particles and their activity factor given by following equation is calculated [63].

$$A_i(t) = \sum_{d=1}^{n} W_d A_{id}(t) \tag{15}$$

where A_i (t) is the whole activity factor of ith particle at iteration t, A_{id} (t) is its dth dimension, W_d is the weight of dth dimension, and A_{id} (t) is calculated by:

$$A_{id}(t) = C\frac{V_{id}(t)}{R_d} \tag{16}$$

where V_{id} (t) is dth dimension of ith particle's velocity at iteration t, and R_d is the scope of search space in dth dimension. According to (16), the particles which move slowly will have a low activity factor.

In this variant, activity factor for each active particle is computed at each iteration, and if it is below a specified value that particle is considered dormant and is reinitialised in search space. This strategy enhances diversity among particles, thereby increases PSO's capability of solving DOP's.

3.3.5.3 Chaos mutation-based PSO In this variant, after environmental change, each dimension's coordinate of a chosen particle i is mapped into the chaos space $[-1,1]$ via following equations [64].

$$r_{id}(t) = -1 + \frac{2}{(X_{d,\max} - X_{d,\min}) \cdot X_{id}(t)} \tag{17}$$

$$r_{id}(t+1) = \mu \cdot r_{id}(t) \cdot (1 - r_{id}(t)) \tag{18}$$

where $X_{\mathrm{d,max}}$ and $\mathrm{X_{d,min}}$ are upper and lower bounds of particle i in its dth dimension, respectively, and μ is a control parameter.

Then, the chaos space of particle i is mapped into the search space via Eq. (19).

$$X_{id}(t) = X_{d,\min} + \frac{r_{id} \cdot (X_{d,\max} - X_{d,\min})}{2} \tag{19}$$

In this strategy, each particle not only adjusts its position according to individual extreme point and swarm extreme point, but also modifies the diversity through the chaotic mutation. For change detection, some particles are randomly chosen as sentry particles and change in each of sentry particles is considered as environmental change.

This variant just has been tested on some simple test functions, and its applicability to complex real-world dynamic problems is under question.

3.3.5.4 Adaptive sentry-based PSO This variant possesses three following features [65, 66].

- Constricted PSO has been adopted and to compensate diversity loss.
- A few particles, named sentry particles, are chosen randomly and monitored at each iteration. Their fitness change implies environmental change.
- For memory update, after environmental change, each particle's new fitness value is compared with its memory and the better one is set as its new memory.

3.3.5.5 Distributed adaptive PSO In this variant, each particle updates its knowledge based on the local information that it receives. If the new fitness value is not better than the previous fitness, following equation is utilised [67, 68].

$$P_i(t+1) = \begin{cases} P_i(t) \cdot T & \text{if } f(X_i(t+1)) \le P_i(t) \cdot T \\ X_i(t+1) & \text{otherwise} \end{cases} \tag{20}$$

where T is called evaporation factor and is a number in (0,1).

Equation (20) implies that if the particle does not improve, the impact of the stored knowledge on its decision for next searching direction and next velocity decreases gradually. This variant has not been tested on complex benchmark functions, and its competency has not been approved.

3.3.5.6 PSO hybridised with LA and GD In [69], learning automation (LA) and great deluge (GD) algorithm are hybridised with PSO to adapt it for dynamic problems. The role of learning automation is to induce perturbation, and the great deluge algorithm works as a local search algorithm.

3.3.5.7 Diversity-guided PSO In this variant, after environmental change, a diversity control mechanism maintains diversity in between the bounds $\mathrm{d_{low}}$ and $\mathrm{d_{max}}$ [70]. When diversity reaches lower bound, the particles positions are reset to increase diversity and when it reaches higher bound, diversity-decreasing operator is activated. Nevertheless, it has not been applied to complex dynamic problems.

3.3.5.8 Stochastic diffusion search—aided PSO In this variant, stochastic diffusion search (SDS) has been

incorporated into PSO. SDS utilises the notion of partial evaluation of fitness function to decrease computational cost. Furthermore, selection and variation mechanisms are employed as diversity-enhancement strategies [71].

3.3.5.9 Unified PSO In [72], unified PSO has been used for solving DOP's. Unified PSO is a combination of global neighbourhood PSO and local neighbourhood PSO with the aim of maintaining an appropriate trade-off between exploration and exploitation. Also, in order to further enhance swarm diversity, a mutation mechanism is incorporated into flight equations. In this variant, the main problem is the setting of unification factor that greatly affects the behaviour of algorithm in dynamic problems.

4 Conclusions and future research directions

PSO is a well-established and efficient optimisation algorithm that has successfully been applied for solving different optimisation algorithm. Some real-life optimisation algorithms are dynamic, meaning that their objective function/constraints vary over time, whereas basic PSO variants are only capable of solving static optimisation problems. In this paper, the PSO variants, designed for dynamic optimisation problems, have been reviewed.

References

1. Bashiri M (2014) Optimal scheduling of distributed energy resources in a distribution system based on imperialist competitive algorithm considering reliability worth. Neural Comput Appl 1–8. doi:10.1007/s00521-014-1581-5
2. Geyik F, Dosdoğru A (2013) Process plan and part routing optimization in a dynamic flexible job shop scheduling environment: an optimization via simulation approach. Neural Comput Appl 23:1631–1641
3. Orlowska-Kowalska T, Kaminski M (2014) Influence of the optimization methods on neural state estimation quality of the drive system with elasticity. Neural Comput Appl 24:1327–1340
4. Chen W-C, Jiang X-Y, Chang H-P, Chen H-P (2014) An effective system for parameter optimization in photolithography process of a LGP stamper. Neural Comput Appl 24:1391–1401
5. Hsu C-M (2014) Application of SVR, Taguchi loss function, and the artificial bee colony algorithm to resolve multiresponse parameter design problems: a case study on optimizing the design of a TIR lens. Neural Comput Appl 24:1293–1309
6. Jordehi AR, Joorabian M (2011) Optimal placement of multi-type FACTS devices in power systems using evolution strategies. In: Power engineering and optimization conference (PEOCO), 2011 5th International, IEEE. pp 352–357
7. Jordehi AR, Jasni J (2011) A comprehensive review on methods for solving FACTS optimization problem in power systems. Int Rev Electr Eng 6:1916–1926
8. Jordehi R (2011) Heuristic methods for solution of FACTS optimization problem in power systems. In: 2011 IEEE student conference on research and development. pp 30–35
9. Rezaee Jordehi A, Jasni J, Abdul Wahab NI, Kadir A, Abidin MZ (2013) Particle swarm optimisation applications in FACTS optimisation problem. In: Power engineering and optimization conference (PEOCO), 2013 IEEE 7th International, IEEE. pp 193–198. doi:10.1109/PEOCO.2013.6564541
10. Jordehi AR, Jasni J, Approaches for FACTS optimization problem in power systems. In: Power engineering and optimization conference (PEDCO) Melaka, Malaysia, 2012 Ieee International, IEEE. pp 355–360
11. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks. Perth, Australia. pp 1942–1948
12. Rezaee Jordehi A, Jasni J (2013) Parameter selection in particle swarm optimisation: a survey. J Exp Theor Artif Intell 25:527–542
13. Jordehi AR, Jasni J (2013) Particle swarm optimisation for discrete optimisation problems: a review. Artif Intell Rev 1–16
14. Rezaee Jordehi A (2014) A comprehensive review on mutation operators in particle swarm optimisation. J Exp Theor Artif Intell 26. doi:10.1080/0952813X.2014.921735
15. Rezaee Jordehi A (2014) Particle swarm optimisation for multimodal optimisation problems: a review. J Exp Theor Artif Intell 26. doi:10.1080/0952813X.2014.924581
16. Rezaee Jordehi A (2014) Particle swarm optimisation for multiobjective optimisation problems: a review. J Exp Theor Artif Intell 26. doi:10.1080/0952813X.2014.924579
17. Branke J (2002) Evolutionary optimization in dynamic environments. Kluwer Academic Publishers, Norwell, MA. ISBN: 0792376315
18. Richter H (2009) Detecting change in dynamic fitness landscapes. In: IEEE. pp 1613–1620
19. Richter H (2009) Change detection in dynamic fitness landscapes: an immunological approach. In: IEEE. pp 719–724
20. Richter H, Dietel F (2010) Change detection in dynamic fitness landscapes with time-dependent constraints. In: IEEE. pp 580–585
21. Branke J (1999) The moving peaks benchmark website. http://www.aifb.unikarl-sruhe.de/jbr/MovPeaks
22. Blackwell TM, Bentley P (2002) Don't push me! collision-avoiding swarms. In: IEEE. pp 1691–1696
23. Blackwell TM, Bentley PJ (2002) Dynamic search with charged swarms. In: Citeseer. pp 19–26
24. Blackwell T, Branke J (2004) Multi-swarm optimization in dynamic environments. In: Applications of evolutionary computing. pp 489–500
25. Blackwell T (2003) Swarms in dynamic environments. In: Springer, pp 200–200
26. Blackwell T, Branke J (2006) Multiswarms, exclusion, and anti-convergence in dynamic environments. Evolut Comput IEEE Trans 10:459–472
27. Zhao J, Sun J, Chen W, Xu W (2009) Tracking extrema in dynamic environments with quantum-behaved particle swarm optimization. In: IEEE. pp 103–108
28. Sun J, Lai C, Xu W, Chai Z (2007) A novel and more efficient search strategy of quantum-behaved particle swarm optimization. In: Adaptive and natural computing algorithms. pp 394–403
29. Sun J, Xu W, Fang W (2006) A diversity-guided quantum-behaved particle swarm optimization algorithm. In: Simulated evolution and learning. pp 497–504
30. Hu X, Eberhart RC (2002) Adaptive particle swarm optimization: detection and response to dynamic systems. In: IEEE. pp 1666–1670
31. Hu X, Eberhart R (2001) Tracking dynamic systems with PSO: where's the cheese. pp 80–83
32. Janson S, Middendorf M (2004) A hierarchical particle swarm optimizer for dynamic optimization problems. In: Applications of evolutionary computing. pp 513–524

33. Janson S, Middendorf M (2006) A hierarchical particle swarm optimizer for noisy and dynamic environments. Genet Program Evolvable Mach 7:329–354
34. Xiaodong L, Khanh Hoa D (2003) Comparing particle swarms for tracking extrema in dynamic environments. In: Evolutionary computation, 2003. CEC '03. The 2003 Congress on, 2003, vol 1773. pp 1772–1779
35. Zheng X, Liu H (2009) A different topology multi-swarm PSO in dynamic environment. In: IT in medicine and education. ITIME '09. IEEE International Symposium on, 2009. pp 790–795
36. Blum C, Merkle D, Blackwell T, Branke J, Li X (2008) Particle swarms for dynamic optimization problems. In: Swarm intelligence. Berlin, pp 193–217
37. Blackwell T (2007) Particle swarm optimization in dynamic environments. In: Evolutionary computation in dynamic and uncertain environments. pp 29–49
38. del Amo IG, Pelta DA, González JR, Novoa P (2010) An analysis of particle properties on a multi-swarm pso for dynamic optimization problems. In: Current topics in artificial intelligence. Springer, pp 32–41
39. del Amo IG, Pelta DA, González JR (2010) Using heuristic rules to enhance a multiswarm PSO for dynamic environments. In: Evolutionary computation (CEC), 2010 IEEE Congress on, IEEE. pp 1–8
40. Novoa-Hernández P, Pelta DA, Corona CC (2010) Improvement strategies for multi-swarm pso in dynamic environments. In: Nature inspired cooperative strategies for optimization (NICSO 2010). Springer, pp 371–383
41. Novoa-Hernández P, Corona CC, Pelta DA (2011) Efficient multi-swarm PSO algorithms for dynamic environments. Memet Comput 3:163–174
42. Rezazadeh I, Meybodi MR, Naebi A (2011) Adaptive particle swarm optimization algorithm for dynamic environments. In: Advances in swarm intelligence. Springer, pp 120–129
43. Novoa P, Pelta DA, Cruz C, del Amo IG (2009) Controlling particle trajectories in a multi-swarm approach for dynamic optimization problems. In: Methods and models in artificial and natural computation. a homage to Professor Mira's scientific legacy. Springer, pp 285–294
44. Parrott D, Li X (2006) Locating and tracking multiple dynamic optima by a particle swarm model using speciation. Evolut Comput IEEE Trans 10:440–458
45. Parrott D, Li X (2004) A particle swarm model for tracking multiple peaks in a dynamic environment using speciation. In: IEEE, vol 101. pp 98–103
46. Li X, Branke J, Blackwell T (2006) Particle swarm with speciation and adaptation in a dynamic environment. In: ACM. pp. 51–58
47. Li C, Yang S (2009) A clustering particle swarm optimizer for dynamic optimization. In: IEEE. pp 439–446
48. Yang S, Li C (2010) A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. Evolut Comput IEEE Trans 14:959–974
49. Kamosi M, Hashemi AB, Meybodi MR (2010) A new particle swarm optimization algorithm for dynamic environments. In: Swarm, evolutionary, and memetic computing. Springer, pp 129–138
50. Kamosi M, Hashemi AB, Meybodi MR (2010) A hibernating multi-swarm optimization algorithm for dynamic environments. In: Nature and biologically inspired computing (NaBIC), 2010 Second World Congress on, IEEE, 2010. pp 363–369
51. Li C, Liu Y, Zhou A, Kang L, Wang H (2007) A fast particle swarm optimization algorithm with Cauchy mutation and natural selection strategy. In: Advances in computation and intelligence. pp 334–343
52. Li C, Yang S (2008) Fast multi-swarm optimization for dynamic optimization problems. In: IEEE. pp 624–628
53. Du W, Li B (2008) Multi-strategy ensemble particle swarm optimization for dynamic optimization. Inf Sci 178:3096–3109
54. Liu L, Yang S, Wang D (2010) Particle swarm optimization with composite particles in dynamic environments. Syst Man Cybern Part B Cybern IEEE Trans 40:1634–1648
55. Liu L, Wang D, Yang S (2008) Compound particle swarm optimization in dynamic environments. In: Applications of evolutionary computing. pp 616–625
56. Wang H, Wang N, Wang D (2008) Multi-swarm optimization algorithm for dynamic optimization problems using forking. In: IEEE. pp 2415–2419
57. Kiranyaz S, Pulkkinen J, Gabbouj M (2011) Multi-dimensional particle swarm optimization in dynamic environments. Expert Syst Appl 38:2212–2223
58. Nickabadi A, Ebadzadeh MM, Safabakhsh R (2008) Evaluating the performance of DNPSO in dynamic environments. In: Systems, man and cybernetics, 2008. SMC 2008. IEEE International Conference on, IEEE. pp 2640–2645
59. Lung RI, Dumitrescu D (2007), A collaborative model for tracking optima in dynamic environments. In: IEEE. pp 564–567
60. Pan G, Dou Q, Liu X (2006) Performance of two improved particle swarm optimization in dynamic optimization environments. In: IEEE. pp 1024–1028
61. Esquivel SC, Coello Coello CA (2006) Hybrid particle swarm optimizer for a class of dynamic fitness landscape. Eng Optim 38:873–888
62. Esquivel SC, Coello CAC (2004) Particle swarm optimization in non-stationary environments. In: Advances in artificial intelligence—IBERAMIA. Springer, pp 757–766
63. Shan S, Deng G (2006) Tracking changing extrema with modified adaptive particle swarm optimizer. In: Intelligent control and automation, 2006. WCICA 2006. The Sixth World Congress on, IEEE. pp 3305–3309
64. Dong D, Jie J, Zeng J, Wang M (2008) Chaos-mutation-based particle swarm optimizer for dynamic environment. In: IEEE. pp 1032–1037
65. Carlisle A, Dozler G (2002) Tracking changing extrema with adaptive particle swarm optimizer. In: IEEE,, pp 265–270
66. Carlisle A, Dozier G (2000) Adapting particle swarm optimization to dynamic environments. pp 429–434
67. Cui X, T.E. Potok, Distributed adaptive particle swarm optimizer in dynamic environment. In: IEEE. pp 1–7
68. Cui X, Hardin C, Ragade R, Potok T, Elmaghraby A (2005) Tracking non-stationary optimal solution by particle swarm optimizer. In: IEEE. pp 133–138
69. Parvin H, Minaei B, Ghatei S (2011) A new particle swarm optimization for dynamic environments, In: Computational intelligence in security for information systems. Springer, pp 293–300
70. Hu J, Zeng J, Tan Y (2007) A diversity-guided particle swarm optimizer for dynamic environments. In: Bio-inspired computational intelligence and applications. pp 239–247
71. M. De, N. Slawomir, B. Mark, Stochastic diffusion search: Partial function evaluation in swarm intelligence dynamic optimisation. In: Stigmergic optimization. pp 185–207
72. Parsopoulos K, Vrahatis M (2005) Unified particle swarm optimization in dynamic environments. In: Applications of evolutionary computing. pp 590–599