Applied Mathematics and Computation 217 (2011) 5208–5226

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Particle swarm optimization: Hybridization perspectives and experimental illustrations

Radha Thangaraj [a,*], Millie Pant [b], Ajith Abraham [c], Pascal Bouvry [a]

[a] *Faculty of Science, Technology and Communications, University of Luxembourg, Luxembourg*
[b] *Department of Paper Technology, Indian Institute of Technology Roorkee, Roorkee 247667, India*
[c] *Machine Intelligent Research Labs (MIR Labs), Scientific Network for Innovation and Research Excellence, USA*

A R T I C L E I N F O

Keywords:
Particle swarm optimization
Hybridization
Differential evolution
Genetic algorithms

A B S T R A C T

Metaheuristic optimization algorithms have become popular choice for solving complex and intricate problems which are otherwise difficult to solve by traditional methods. In the present study an attempt is made to review the hybrid optimization techniques in which one main algorithm is a well known metaheuristic; particle swarm optimization or PSO. Hybridization is a method of combining two (or more) techniques in a judicious manner such that the resulting algorithm contains the positive features of both (or all) the algorithms. Depending on the algorithm/s used we made three classifications as (i) Hybridization of PSO and genetic algorithms (ii) Hybridization of PSO with differential evolution and (iii) Hybridization of PSO with other techniques. Where, other techniques include various local and global search methods. Besides giving the review we also show a comparison of three hybrid PSO algorithms; hybrid differential evolution particle swarm optimization (DE-PSO), adaptive mutation particle swarm optimization (AMPSO) and hybrid genetic algorithm particle swarm optimization (GA-PSO) on a test suite of nine conventional benchmark problems.

1. Introduction

Optimization is ubiquitous and spontaneous process that forms an integral part of our day-to-day life. In the most basic sense, it can be defined as an art of selecting the best alternative among a given set of options. Optimization problems arise in various disciplines such as engineering designs, agricultural sciences, manufacturing systems, economics, physical sciences, pattern recognition etc. in fact optimization techniques are being extensively used in various spheres of human activities, where decisions have to be taken in some complex situation that can be represented by a mathematical model. Optimization can thus be viewed as one of the major quantitative tools in network of decision making, in which decisions have to be taken to optimize one or more objectives in some prescribed set of circumstances. In view of the practical utility of optimization problems there is a need for efficient and robust computational algorithms, which can numerically solve on computers the mathematical models of medium as well as large size optimization problem arising in different fields.

In the past few decades several global optimization algorithms have been developed that are based on the nature inspired analogy. These are mostly population based metaheuristics also called general purpose algorithms because of their applicability to a wide range of problems. Some popular global optimization algorithms include genetic algorithms (GA) [1], particle swarm optimization (PSO) [2], Differential Evolution (DE) [3], evolutionary programming (EP) [4], ant colony optimization

* Corresponding author.
E-mail address: t.radha@ieee.org (R. Thangaraj).

0096-3003/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2010.12.053

(ACO) [5] etc. A chronological order of the development of some of the popular nature inspired metaheuristics is given in Fig. 1. These algorithms have proved their mettle in solving complex and intricate optimization problems arising in various fields.

However, despite having several attractive features, it has been observed that these algorithms do not always perform as per expectations. The success of most of the metaheuristics optimization algorithms depends to a large extent on the careful balance of two conflicting goals, *exploration (diversification)* and *exploitation (intensification)*. While exploration is important to ensure that every part of the solution domain is searched enough to provide a reliable estimate of the global optimum; exploitation, on the other hand, is important to concentrate the search effort around the best solutions found so far by searching their neighborhoods to reach better solutions [6]. The search algorithms achieve these two goals by using local search methods, or global search approaches, or an integration of both global and local strategies: these algorithms are commonly known as *hybrid methods*. The focus of the present study is on the review of hybrid algorithms in which one of the main algorithm is a well known and popular search strategy; PSO. It is a simple and robust strategy based on the social and cooperative behavior shown by various species like flock of bird, school of fish etc. PSO and its variants have been effectively applied to a wide range of benchmark as well as real life optimization problems. A brief discussion on PSO is given in Section 2.

Hybrid algorithms are chosen as the topic of the present article because they are a growing area of intelligent systems research, which aims to combine the desirable properties of different approaches to mitigate their individual weaknesses. Hybridization can be done in several ways some of which are; (1) initiate the algorithm with one technique and then apply the other technique on the final population obtained by the first technique, (2) embed the unique operators of a particular technique into the other technique for example mutation and crossover operators of GA can be used in PSO (3) apply local search to improve the solution obtained by the global search and so on. In the present study we have concentrated our work on the hybridization of PSO algorithm with other search techniques which may be local as well as global.

Besides the review, this paper also presents a comparison of some hybridized PSO versions; DE-PSO, AMPSO, GA-PSO on a set of nine conventional benchmark problems. All these algorithms have been successfully used for solving continuous global optimization problems.

The present article is divided into six sections including the introduction. In Section 2, a brief description of basic PSO is given; Section 3 gives brief review on hybrid PSO algorithms. In Section 4, we give a brief outline of the three hybrid PSO algorithms used for the purpose of comparison. In Section 5 we give the experimental settings and result analysis of the algorithms used for comparison. Finally the conclusions based on the present study are drawn in Section 6.

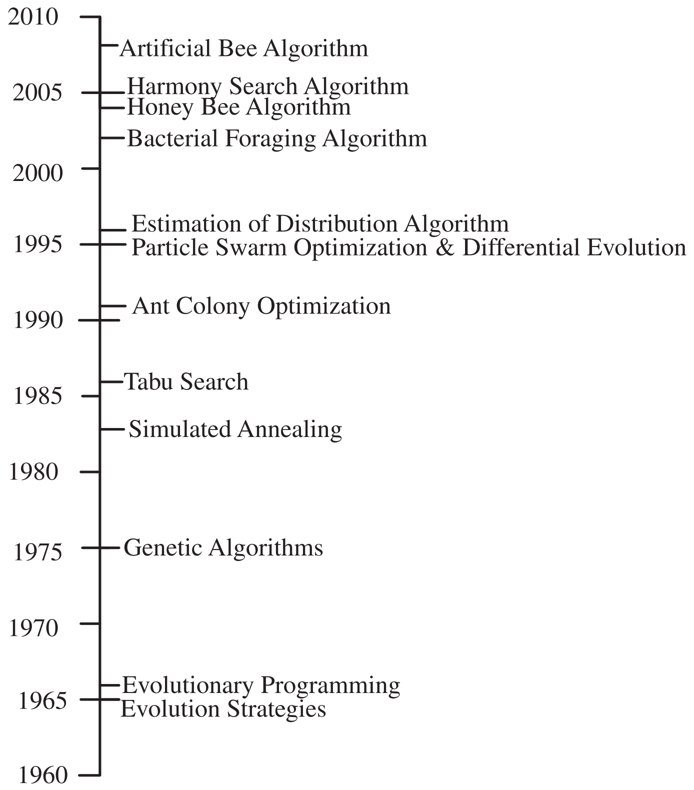


Fig. 1. Popular meta-heuristics in chronological order.

2. Working of PSO algorithm

The concept of PSO was first suggested by Kennedy and Eberhart [2] in 1995. Since its development, PSO has become one of the most promising optimizing techniques for solving global optimization problems. Its mechanism is inspired by the social and cooperative behavior displayed by various species like birds, fish, termites, ants and even human beings. The PSO system consists of a population (swarm) of potential solutions called particles. These particles move through the search domain with a specified velocity in search of optimal solution. Each particle maintains a memory which helps it in keeping the track of its previous best position. The positions of the particles are distinguished as personal best and global best. In the past several years, PSO has been successfully applied in many research and application areas. It has been demonstrated that PSO gets better results in a faster, cheaper way in comparison to other methods like GA, simulated annealing (SA) etc.

The particles or members of the swarm fly through a multidimensional search space looking for a potential solution. Each particle adjusts its position in the search space from time to time according to the flying experience of its own and of its neighbors (or colleagues).

For a D-dimensional search space, the position of the ith particle is represented as:

$$X_i = (x_{i1}, x_{i2}, \ldots, x_{id}, \ldots, x_{iD}). \tag{1}$$

Each particle maintains a memory of its previous best position which is represented as:

$$P_i = (p_{i1}, p_{i2}, \ldots, p_{id}, \ldots, p_{iD}). \tag{2}$$

The best one among all the particles in the population is represented as:

$$P_g = (p_{g1}, p_{g2}, \ldots, p_{gd}, \ldots, p_{gD}). \tag{3}$$

The velocity of each particle is represented as:

$$V_i = (v_{i1}, v_{i2}, \ldots, v_{id}, \ldots, v_{iD}). \tag{4}$$

The maximum velocity is represented as:

$$V_{\max} = (v_{\max,1}, v_{\max,2}, \ldots, v_{\max,d}, \ldots, v_{\max,D}). \tag{5}$$

The velocity V_i of each particle is clamped to a maximum velocity $V_{\max}$ which is specified by the user. $V_{\max}$determines the resolution with which regions between the present position and the target position are searched. Large values of $V_{\max}$ facilitate global exploration, while smaller values encourage local exploitation. If $V_{\max}$ is too small, the swarm may not explore sufficiently beyond locally good regions. On the other hand, too large values of $V_{\max}$ risk the possibility of missing a good region [7].

At each iteration a new velocity value for each particle is evaluated according to its current velocity, the distance from the global best position. The new velocity value is then used to calculate the next position of the particle in the search space. This process is then iterated a number of times or until a minimum error is achieved. The two basic equations which govern the working of PSO are that of velocity vector and position vector given by:

$$v_{id} = v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \tag{6}$$

$$x_{id} = x_{id} + v_{id} \tag{7}$$

Here c_1 and c_2 are acceleration constants. They represent the weighting of the stochastic acceleration terms that pull each particle towards personal best and global best positions. Therefore, adjustment of these constants changes the amount of tension in the system. Small values of these constants allow particles to roam far from the target regions before tugged back, while high values result in abrupt movement toward, or past, target regions [8]. The constants r_1, r_2 are the uniformly generated random numbers in the range of $[0,1]$.

The first part of Eq. (6), v_{id}, represents particle's previous velocity, which serves as a memory of the previous flight direction. This memory term can be visualized as a momentum, which prevents the particle from drastically changing its direction and biases it towards the current direction. The second part, $c_1 r_1 (p_{id} - x_{id})$, is called the cognition part and it indicates the personal experience of the particle. We can say that, this cognition part resembles individual memory of the position that was best for the particle. The effect of this term is that particles are drawn back to their own best positions, resembling the tendency of individuals to return to situations or places that were most satisfying in the past. The third part, $c_2 r_2 (p_{gd} - x_{id})$, represents the cooperation among particles and is therefore named as the social component [9]. This term resembles a group norm or standard which individuals seek to attain. The effect of this term is that each particle is also drawn towards the best position found by its neighbor.

A few important and interesting modifications in the basic structure of PSO are as follows:

Shi and Eberhart [10] introduced a new parameter called inertia weight into the velocity vector Eq. (6), after which the equation becomes:

$$v_{id} = \omega^* v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}). \tag{8}$$

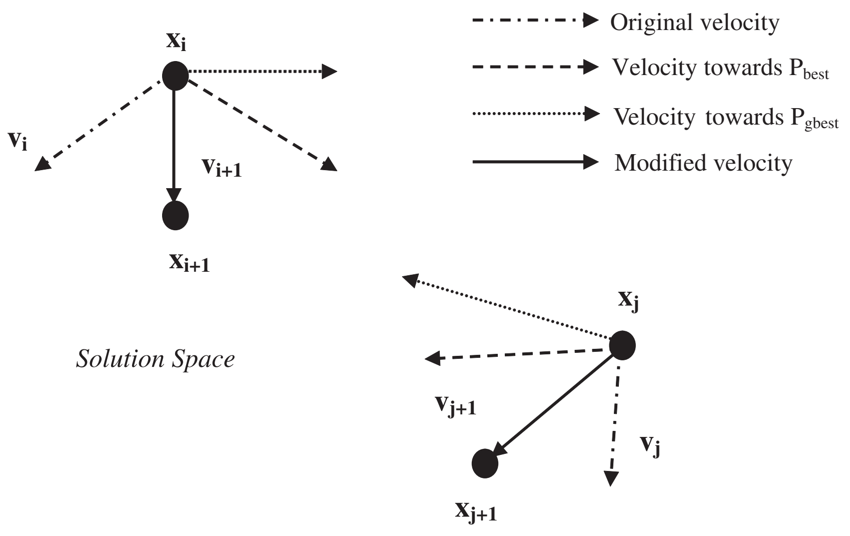


Fig. 2. Searching mechanism of PSO.

The inertia weight ω is employed to control the impact of the previous history of velocities on the current velocity, thereby influencing the trade-off between global and local exploration abilities of the particles. It can be a positive constant or even a positive linear or nonlinear function of time. A larger inertia weight encourages global exploration while a smaller inertia weight tends to facilitate local exploration to fine-tune the current search area. Suitable selection of the inertia weight provides a balance between global and local exploration abilities and thus requires less iteration on an average to find the optimum [11]. Initially the inertia weight was kept static during the entire search process for every particle and dimension. However, with the due course of time dynamic inertia weights were introduced.

Maurice Clerc [12,13] introduced a new parameter called constriction factor '*K*' that improves PSO's ability to constrain and control velocities. Eberhart and Shi [14] later verified that *K*, combined with constraints on V_{max}, significantly improved the PSO performance.

The value of the constriction factor is calculated as follows:

$$K = \frac{2}{|2-\varphi-\sqrt{\varphi^2-4\varphi}|}, \quad \varphi = c_1 + c_2, \quad \varphi > 4. \tag{9}$$

With the constriction factor *K*, the PSO equation for computing the velocity is:

$$v_{id} = K(v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id})). \tag{10}$$

Usually, when the constriction factor is used, φ is set to 4.1 ($c_1 = c_2 = 2.05$), which gives the value of constriction factor *K* as 0.729. Carlisle and Dozier [15] showed that cognitive and social values of $c_1 = 2.8$ and $c_2 = 1.3$ also yield good results for their chosen set of problems.

The constriction approach is effectively equivalent to the inertia weight approach. Both approaches have the objective of balancing exploration and exploitation, and thereby improving convergence time and the quality of solution found. It should be noted that low values of ω and *K* result in exploitation with little exploration, while large values result in exploration with difficulties in refining solutions [7].

The working of PSO in space is illustrated in Fig. 2.

3. Survey on hybrid PSO algorithms

In this Section we discuss the main topic of this article which is a survey on hybrid algorithms in which PSO is one of the prime algorithms. A natural evolution of the population based search algorithms like that of PSO can be achieved by integrating the methods that have already been tested successfully for solving complex problems. Researchers have enhanced the performance of PSO by incorporating in it the fundamentals of other popular techniques like selection, mutation and crossover of GA and DE. Also, attempts have been made to improve the performance of other evolutionary algorithms like GA, DE, and ACO etc. by integrating in them the velocity and position update equations of PSO. The main goal, as we see is to harness the strong points of the algorithms in order to keep a balance between the *exploration* and *exploitation* factors thereby preventing the stagnation of population and preventing premature convergence. The following subsections present the hybrid versions of PSO with GA, DE and other search techniques.

3.1. Hybridization of PSO with GA

GA's, developed by John Holland [1] and colleagues and later modified by Goldberg [16] are one of the earliest and perhaps the most commonly used evolutionary algorithms. Researchers have observed that a combination of PSO and GA may result in a potent algorithm capable of dealing with complex optimization problems. In this section we give an overview of the attempts made to hybridize PSO with GA.

In 2002, Robinson et al. [17] proposed two hybrid algorithms; GA-PSO and PSO-GA for solving a particular electromagnetic application of profiled corrugated horned antenna. In GA-PSO algorithm, GA was used to generate the initial population for PSO, while in the second algorithm PSO-GA, PSO generates an initial population for GA. His results showed that the PSO-GA hybrid algorithm outperforms the GA-PSO version as well as simple PSO and GA versions. In the same year, a '*Life-Cycle Model*' inspired by the life cycle stages found in nature was proposed in [18]. In this algorithm PSO was hybridized with the concepts of GA and Hill Climbing approaches and was used for solving unconstrained global optimization problems. The authors showed that either Particle Swarm Optimization algorithm or Genetic Algorithm or Hill Climbing search algorithm can be applied to a different sub-population of individuals in which each individual is dynamically assigned according to some pre-designed rules. Conradie et al. [19] proposed a symbiotic neuro memetic evolution (SMNE) by combining 'symbiotic genetic algorithm' with PSO for neural network controllers in a reinforcement learning framework. In SMNE, GA was used for global search while PSO was used for local search procedure.

Grimaldi et al. [20] proposed a hybrid technique combining GA and PSO called genetical swarm optimization (GSO) for solving combinatorial optimization problems. They applied their algorithm for solving an electromagnetic optimization problem. In GSO, the population is divided into two parts and is evolved with these two techniques (GA and PSO) in every iteration. The populations are then recombined in the updated population, that is again divided randomly into two parts in the next iteration for another run of genetic or particle swarm operators. They defined a new parameter HC (or Hybridization Constant) that expresses the percentage of population that is evolved with GA in every iteration: so $HC = 0$ implies that the procedure is pure PSO (the whole population is updated according to PSO operators) and $HC = 1$ implied that pure GA is being followed. However, $0 < HC < 1$ means that the corresponding percentage of the population is developed by GA and the rest with PSO. In [21], the authors proposed several hybridization strategies (static, dynamic, alternate, self adaptive etc.) for GSO algorithm and validated them with some multimodal benchmark problems. The application of GSO has also been shown in [22–24].

Another hybridization strategy of PSO and GA (HGAPSO) was proposed by Juang [25] where the upper half of the best-performing individuals in a population is regarded as elite. Before using GA operators, the algorithm is first enhanced by means of PSO, instead of being reproduced directly to the next generation. They applied HGAPSO to neural/fuzzy network design. Settles and Soule [26] introduced a GA and PSO hybrid called the breeding swarm (BS) algorithm. The BS algorithm combines the standard velocity and position update rules of PSOs with the ideas of selection, crossover and mutation from GAs. They also included an additional parameter called the breeding ratio to determine the proportion of the population which undergoes breeding in the current generation. Their algorithm consisted of four main steps; in the first step, n best individuals are copied into a temporary population, in the second step some individuals are selected to undergo the velocity and position update rules of PSO. In the next step the remaining individuals are selected by some selection criteria for crossover and mutation criteria. Finally the temporary population is copied into the working population and the fitness is evaluated. This process is repeated till the optimum is obtained. The BS algorithm was validated on four unconstrained scalable optimization problems for different dimensions.

Jian and Chen [27] introduced a PSO hybrid with the GA recombination operator and dynamic linkage discovery called PSO-RDL. They assumed that the relation between different dimensions is dynamically changed along with the search process. Thus, the linkage configuration should be updated accordingly. Instead of incorporating extra artificial criteria for linkage adaptation, they entrusted the task to the mechanism of natural selection. The idea is to update the linkage configuration according to the fitness feedback. They applied their algorithm for solving 25 unconstrained test problems with varying degrees of complexities. Mohammadi and Jazaeri [28] proposed a hybrid algorithm in which the initial population for GA is generated by PSO. They used their algorithm for solving an IEEE 68 bus system. While Esmin et al. [29] proposed a PSO algorithm coupled with a GA mutation operator only. They named their algorithm HPSOM and used it for solving unconstrained global optimization problems. An improved GA called GA-PSO was proposed by Kim [30] by using PSO and the concept of Euclidean distance. In GA-PSO, the performance of GA was improved by using PSO and Euclidian data distance on mutation procedure of GA. He applied his algorithm for obtaining the local and global optima of Foxhole function.

In the PSO-GA based hybrid evolutionary algorithm (HEA) of Yang et al. [31], evolution process is divided into two stages. The first stage is similar to PSO where the particle flies in hyperspace and adjusts its velocity by following particles with better fitness according to flying experience of itself and its neighbors. The second stage is similar to GA where genetic operators of selection, reproduction, crossover, and mutation are exerted on particles at predetermined probability. By combination of PSO and GA, evolution process is accelerated by flying behavior and population diversity is enhanced by genetic mechanism. They used a single point crossover, Gaussian mutation and Roulette wheel for selection process. They applied their algorithm for solving 3 unconstrained as well as 3 constrained optimization problems.

Kao and Zahara [32] proposed a hybrid version called GA-PSO. In this algorithm, for solving a D-dimensional problem, a population of $4D$ individuals is generated. This population is sorted according to the fitness value and the top $2D$ individuals are fed into a real coded GA to create $2D$ new individuals by crossover and mutation operations. For GA, they used a linear crossover and random mutation. The new *2D* individuals created from GA are used to update the remaining *2D* individuals by PSO, using the concepts of global best and neighborhood best particles and position updates etc. They applied their algorithm for solving 17 unconstrained multimodal test functions. Ru and Jianhua et al. [33] introduced a hybrid of GA and PSO, which synthesizes the merits of GA and PSO and integrates the concept of evolving individuals originally modeled by GA with the concept of self-improvement of PSO, where individuals enhance themselves based on social interactions and their private cognition. The proposed hybrid algorithm performs the population alternation to the features of the evolution of the populations in natural. Shunmugalatha and Slochanal [34] proposed a hybrid particle swarm optimization (HPSO), which incorporates the breeding and subpopulation process in genetic algorithm into particle swarm optimization. The implementations of HPSO to test systems showed that it converges to better solution much faster.

In [35], Li et al. proposed a hybrid of PSO and GA called PGHA for optimization design. They used improved genetic mechanisms like nonlinear ranking selection [36] for GA process. They generated three offspring from two parents and used a newly defined mutation operator. The working of PGHA is once again similar to most of the hybridized PSO/GA versions where the entire population is divided into two parts and each part is evolved separately using GA and PSO mechanism. Ting et al. [37] proposed a hybrid constrained GA/PSO algorithm for solving load flow problem. Jeong et al. [38] proposed a hybrid algorithm called GA/PSO for solving multiobjective optimization problems and validated their algorithm on test problems and also on engineering design problems. In this algorithm, the first multiple solutions are generated randomly as an initial population and objective function values are evaluated for each solution. After the evaluation, the population is divided into two sub-populations one of which is updated by the GA operation, while the other is updated by PSO operation. New solutions created by each operation are combined in the next generation, and non-dominated solutions in the combined population are archived. The archive data are shared between the GA and PSO, i.e., non-dominated solutions created by the PSO can be used as parents in GA, while non-dominated solutions created by GA can be used as global guides in PSO.

Valdez et al. [39] integrated the concept Fuzzy Logic in the hybrid of GA and PSO algorithm. The new hybrid algorithm called PSO + GA also works by dividing the individuals into GA and PSO populations. However, it differs from the previous approaches as in GA + PSO fuzzy rules are applied to decide whether to take GA individuals or to take PSO individuals. Three rules were considered; (1) for decision making (called main fuzzy), (2) for changing the parameters of GA (called fuzzyga) and for changing the parameters of PSO (fuzzypso). GA + PSO algorithm was validated on a set of 5 unconstrained benchmark problems. Bhuvaneswari et al. [40] reported a hybrid approach using genetic algorithm and particle swarm optimization called HGAPSO. Though the name of the algorithm is same as the one developed by Ru and Jianhua [33], its working is quite similar to the GA-PSO algorithm developed by Kao and Zahara [32]. In HGAPSO algorithm, for solving a *D*-dimensional problem, *4D* individuals are generated randomly. These *4D* individuals are sorted according to the fitness and the top *2D* individuals are fed into real coded GA while the bottom *2D* individuals are fed into PSO. Genetic operators such as selection, crossover and mutation are applied to the best *2D* individuals and PSO operators such as velocity and position update are applied to the worst *2D* individuals to create *4D* individuals for next generation. Bhubaneshwari et al. [40] applied their algorithm for optimizing the problem of alternator design.

Premalatha and Natarajan [41] proposed a discrete PSO with GA operators for document clustering. The strategy induces a reproduction operation by GA operators when the stagnation in movement of the particle is detected. They named their algorithm as DPSO with mutation and crossover. Abdel-Kader [42] proposed GAI-PSO algorithm, which combines the standard velocity and position update rules of PSO with the ideas of selection and crossover from GAs. The GAI-PSO algorithm searches the solution space to find the optimal initial cluster centroids for the next phase. The second phase is a local refining stage utilizing the k-means algorithm which can efficiently converge to the optimal solution. The proposed algorithm combines the ability of the globalized searching of the evolutionary algorithms and the fast convergence of the k means algorithm and can avoid the drawback of both. The summary of the main features of the hybridized PSO and GA algorithms are given in Table 1.

3.2. Hybridization of PSO with DE

The DE algorithm, originated in the same year as that of PSO i.e. in 1995, was proposed by Price and Storn [3] for solving global optimization problem. DE uses the same evolutionary operators (mutation, crossover and selection) as that of GA but it's the working of these operators that distinct DE from GA. Like PSO, DE has also been successfully used to solve a wide range of benchmark and real life problems. In this section we give an overview of the hybridization of these two promising techniques DE and PSO.

Hendtlass [43] used the DE perturbation approach to adapt particle positions. In his algorithm, named SDEA, particles' positions are updated only if their offspring have better fitness. The DE reproduction process is applied to the particles in swarm at specified intervals. At the specified intervals, the PSO swarm serves as the population for DE algorithm, and the DE is executed for a number of generations. After execution of DE, the evolved population is further optimized using PSO. Hendtlass applied his algorithm SDEA on 4 unconstrained benchmark problems for different dimensions and for different population sizes. Zhang and Xie [44] used different techniques in random, rather than combining them, in their Differential Evolution (DE) PSO (DEPSO). In this case the DE and canonical PSO operators were used on alternate generations; when DE was in use, the trial mutation replaced the individual best at a rate controlled by a crossover constant and a random dimension selector that ensured at least one mutation occurred each time.

Table 1
A summary of hybridized PSO + GA algorithms.

Author/s	Year	Name of the Algorithm	Application
Robinson et al. [17]	2002	GA-PSO and PSO-GA	Engineering design
Krink and Lovbjerg [18]	2002	Life cycle model	Unconstrained global optimization
Conradie et al. [19]	2002	SMNE	Neural networks
Grimaldi et al. [20]	2004	GSO	Electromagnetic application
Juang [25]	2004	GAPSO	Network design
Settles and Soule [26]	2005	Breeding swarm	Unconstrained global optimization
Jian and Chen [27]	2006	PSO-RDL	Unconstrained global optimization
Esmin et al. [29]	2006	HPSOM	Unconstrained global optimization
Kim [30]	2006	GA-PSO	Unconstrained global optimization
Mohammadi and Jazaeri [28]	2007	PSO-GA	Power system
Gandelli et al. [21]	2007	GSO	Unconstrained global optimization
Yang et al. [31]	2007	HEA	Constrained and unconstrained global optimization
Kao and Zahara [32]	2008	GA-PSO	Unconstrained global optimization
Ru and Jianhua [33]	2008	HGAPSO	Unconstrained global optimization
Shunmugalatha and Slochanal [34]	2008	HPSO	Power system
Li et al. [35]	2008	PGHA	Antenna design
Ting et al. [37]	2008	CGA/PSO	Power system
Jeong et al. [38]	2009	GA\PSO	Multiobjective optimization problems
Valdez et al. [39]	2009	GA + PSO	Unconstrained optimization problems
Bhubaneshwari et al. [40]	2009	HGAPSO	Alternator design
Premlatha and Natrajan [41]	2009	DPSO-mutation-crossover	Document clustering
Abdel Kader [42]	2010	GAI-PSO	Data clustering

Kannan et al. [45] applied DE to each particle for a finite number of iterations, and replaced the particle with the best individual obtained from the DE process. Talbi and Batauche [46] proposed an algorithm named DEPSO. It differs from the DEPSO algorithm proposed by Zhang and Xie as the DE operators are applied only to the best particle obtained by PSO. They applied their algorithm on medical image processing problem. In Hao et al.'s [47] hybrid version, which is also named as DEPSO, the candidate solution is generated either by DE or by PSO according to some fixed probability distribution. They applied their algorithm for solving unconstrained global optimization problems. Das et al. [48] proposed a scheme of adjusting velocities of the particles in PSO with a vector differential operator borrowed from the DE family. In their proposed PSO-DV algorithm, they omitted the cognitive term of the canonical PSO and updated the particle velocities by a new term containing the weighted difference (inspired by DE mutation scheme) of the position vectors of any two distinct particles randomly chosen from the swarm. They also applied their algorithm on a test suite of selected unconstrained benchmark problems.

Omran et al. [49] proposed a hybrid version of Bare Bones PSO and DE called it BBDE. In their approach, they combined the concept of barebones PSO with self adaptive DE strategies. The mutation operator of DE is used to explore around the current attractor by adding a difference vector to the attractor. Crossover is done with randomly selected personal best as these personal bests represent a memory of the best solution found by individuals since the start of the search process. They validated their algorithm on a set of unconstrained benchmark problems and also applied to image classification problem. The work of Jose et al. [50] evaluated a Particle Swarm Optimizer hybridized with Differential Evolution and applied it to the Black-Box Optimization Benchmarking for noisy functions. Their algorithm was once named as DEPSO algorithm. In this version of DEPSO, the differential variation schemes of DE are used for updating the velocities of the swarm particles. Zhang et al. [51] developed a hybrid of DE and PSO called DE-PSO, where three alternative updating strategies are used. The DE updating strategy is executed once in every l generations and if a particle's best encountered position and the position of its current best neighbor are equal then random updating strategy is executed otherwise PSO updating strategy is used.

Liu et al. [52] proposed a novel hybrid algorithm named PSO-DE, in which DE is incorporated to update the previous best positions of PSO particles to force them to jump out of local attractor in order to prevent stagnation of population. Capanio et al. [53] developed a Superfit Memetic Differential Evolution (SFMDE) by hybridizing DE and PSO with two other local search methods; Nelder Mead algorithm and Rosenbrock algorithm. In SFMDE, PSO assists DE in the beginning of the optimization process to generate a '*super-fit individual*', the local searches are then applied adaptively by means of a parameter which measures the quality of super-fit individual. SFMDE was applied for solving unconstrained standard benchmark problems and two engineering problems. Xu and Gu [54] proposed particle swarm optimization with prior crossover differential evolution (PSOPDE). Their version is distinct from other PSO-DE hybrids in three ways; (1) the particles in the swarm are not just led towards global and personal best positions but also depend on the average position and velocity of the particles (2) DE is integrated with PSO for local search and (3) a new crossover operation between the target and an extra population is implemented before the DE component. They applied their algorithm for solving 5 unconstrained benchmark problems.

The DE-PSO algorithm suggested by Pant et al. [55] starts like the usual DE algorithm. It enters the PSO phase if the optimality criteria are not met by the DE algorithm. They tested their algorithm on a set of unconstrained benchmark problems. Khamsawang et al. [56] proposed an improved hybrid algorithm based on conventional particle swarm optimization and differential evolution (called PSO-DE) for solving an economic dispatch (ED) problem with the generator constraints. In PSO-DE, the mutation operators of the differential evolution are used for improving diversity exploration of PSO and are activated if velocity values of PSO are near to zero or violate the boundary conditions. The summary of the main features of the hybridized PSO and DE algorithms are given in Table 2.

3.3. Hybridization of PSO with other algorithms

Besides GA and DE, PSO has been hybridized with some other local and global search techniques as well. These techniques ACO, Nelder Mead algorithm [57], Tabu Search [58], Fuzzy Logic etc. In this section we try to cover the hybrid versions in which PSO is integrated with different search techniques.

Shelokar et al. [59] proposed a hybrid of PSO with Ant Colony Optimization (ACO) and called their algorithm PSACO. It is a two stage algorithm, in which PSO is applied in the first stage and PCO is implemented in the second stage. Here, ACO works as a local search procedure in which the 'ants' apply the pheromone guided mechanism to update the positions found by the particles in the earlier stages. They applied their algorithm for solving 17 unconstrained benchmark problems. In [60], ACO is combined with PSO by Hendtlass and Randall. A list of best positions found so far is recorded and the neighborhood best is randomly selected from the list instead of the current neighborhood best. Victoire and Jeyakumar [61] hybridized PSO with another algorithm called sequential quadratic programming (SQP) [62] for solving economic dispatch problem. In their algorithm PSO was used as a main optimizer while SQP was used to fine tune the solution obtained after every PSO run. SQP is a non linear gradient based programming technique. The best values obtained by PSO at the end of each iteration are treated as a starting point of SQP and are further improved with the hope of finding a better solution.

Grosan et al. [63] proposed a modified variant of the PSO for solving the well known geometrical place problems. In case of such problems the search space consists of more than one point each of which is to be located. Therefore technique named independent neighborhoods particle swarm optimization (INPSO) consisted of independent subswarms so that at the end of iteration multiple points can be generated. The performance of the INPSO approach is compared with Geometrical Place Evolutionary Algorithms (GPEA) [64]. To further enhance the performance of the INPSO approach, the authors combined INPSO with GPEA. The hybridized version proved to be superior to both INPSO and GPEA. Chuanwen and Bompard [65] hybridized the chaotic version of PSO with linear interior point to handle the problems remaining in the traditional arithmetic of time-consuming convergence and demanding initial values. They classified their algorithm into two phases; first the chaotic PSO is applied to search the solution space and then linear interior method is used to search the global optimum. They validated their algorithm on an IEEE 30 system.

In order to prevent the problem of premature convergence and stagnation of population, Liu et al. [66] proposed a turbulent PSO (TPSO) to overcome the drawbacks of standard PSO. Further, they hybridized TPSO with a fuzzy logic controller to produce a Fuzzy Adaptive TPSO (FATPSO). The TPSO used the principle that PSO's premature convergence is caused by particles stagnating about a sub-optimal location. A minimum velocity was introduced with the velocity memory being replaced by a random turbulence operator when a particle exceeded it. The fuzzy logic extension was then applied to adaptively regulate the velocity parameters during an optimization run thus enabling coarse-grained explorative searches to occur in the early phases before being replaced by fine-grained exploitation later. They analyzed their algorithm on a set of 10 unconstrained benchmark problems. A hybridization of PSO with Tabu search (TS) was suggested by Sha and Xsu [67] for solving job shop problem (JSP). They modified the PSO for solving the discrete JSP and applied TS for refining the quality of solutions.

Another hybrid of PSO was suggested with Simulated Annealing (SA) for solving constrained optimization problems. It was proposed by He and Wang [68]. In their hybrid algorithm, SA was applied to the best solution of the swarm to help the algorithm escape from local optima. Mo et al. [69] proposed an algorithm called Particle Swarm Assisted Incremental Evolution Strategy (PIES). In this version of PSO, the process of evolution is divided into several phases. Each phase is composed of two steps; in the first step, a population is evolved with regard to a certain concerned variable on some moving

Table 2
A summary of hybridized PSO + DE algorithms.

Author/s	Year	Name of the Algorithm	Application/s
Hentdlass [43]	2001	SDEA	Unconstrained global optimization
Zhang and Xie [44]	2003	DEPSO	Unconstrained global optimization
Kannan et al. [45]	2004	–	Generation expansion planning
Talbi and Batouche [46]	2004	DEPSO	Medical image processing
Hao et al. [47]	2007	DEPSO	Unconstrained global optimization
Das et al. [48]	2008	PSO-DV	Engineering design
Omran et al. [49]	2008	BBDE	Unconstrained optimization problems and image processing
Jose et al. [50]	2009	DEPSO	Noisy functions
Zhang et al. [51]	2009	DE-PSO	Unconstrained optimization
Liu et al. [52]	2009	PSO-DE	Constrained optimization and engineering problems
Caponio et al. [53]	2009	SFMDE	Unconstrained global optimization and engineering design problems
Xu and Gu [54]	2009	PSOPDE	Unconstrained global optimization
Pant et al. [55]	2009	DE-PSO	Unconstrained global optimization
Khamsawang et al. [56]	2010	PSO-DE	Power systems

cutting plane adaptively. In the next step the better-performing individuals obtained from step one and the population obtained from the last phase are joined together in second step to form an initial population. Here, PSO is applied as a global phase while Evolutionary strategy is applied as a local phase.

Fan and Zahara [70] and Fan et al. [71] hybridized PSO with a well know local search namely Nelder Mead Simplex method and used it for solving unconstrained benchmark problems. This hybrid version was named as NM-PSO, which was later extended for solving constrained optimization problems by Zahara and Kao [72]. Guo et al. [73] on the other hand hybridized PSO with gradient descent (GD) method for fault diagnosis. Shen et al. [74] developed a hybrid algorithm of PSO and TS called HPSOTS approach for gene selection for tumor classification. TS was incorporated as a local improvement procedure to enable the algorithm to overleap local optima. A hybrid of PSO and Artificial Immune System (AIS) was proposed by Ge et al. [75] for job scheduling problem. An effective hybrid particle swarm cooperative optimization (HPSCO) algorithm combining simulated annealing method and simplex method was proposed by Song et al. [76]. The main idea of HPSCO is to divide particle swarm into several sub-groups and achieve optimization through cooperativeness of different sub-groups among the groups.

The NM-PSO algorithm proposed in [70,71] was hybridized with K-means algorithm by Kao et al. [77] and was applied to data clustering problem. Another hybrid PSO algorithm for cluster analysis was proposed by Firouzi et al. [78]. Here, PSO was integrated with SA and k-means algorithms. Ramana Murthy et al. [79] introduced a hybrid PSO, which combined the merits of the parameter-free PSO (pf-PSO) and the extrapolated particle swarm optimization like algorithm (ePSO). Various hybrid combinations of pf-PSO and ePSO methods were considered and tested with the standard benchmark problems. Kuo et al. [80] presented a new hybrid particle swarm optimization model named HPSO that combines a random-key (RK) encoding scheme, individual enhancement (IE) scheme, and particle swarm optimization (PSO) for solving the flow-shop scheduling problem. Chen et al. [81] proposed a hybrid of PSO and Extremal Optimization (EO) [82], called PSO-EO algorithm, where EO is invoked in PSO at l generation intervals (l is predefined by the user). A heuristic and discrete heuristic particle swarm ant colony optimization (HPSACO and DHPSACO) have been presented for optimum design of trusses by Kavehand and Talatahari [83,84].

Table 3
A summary of hybridized PSO + *other* algorithms.

Author/s	Year	Name of the algorithm	*Other* Algorithm	Application
Hendtlass and Randall [60]	2001	–	Ant colony optimization	Discrete optimization problems
Wei et al. [86]	2002	SFEP	Evolutionary programming	Unconstrained global optimization
Victorie and Jeyakumar [61]	2004	PSO-SQP	Sequential quadratic programming	Power systems
Fan et al. [71]	2004	NM-PSO	Nelder Mead simplex method	Multimodal functions
Grosan et al. [63]	2005	INPSO INPSO-GPEA	Subswarms and GPEA	Geometric place problems
Chuanwen and Bompard [65]	2005	New PSO	Linear interiormethod	Power systems
Sha and Hsu [67]	2006	HPSO	Tabu search	Job shop scheduling
Guo et al. [73]	2006	HGDPSO	Gradient descent	Machine diagnosis
Liu et al. [66]	2007	FATPSO	Fuzzy logic	Unconstrained global optimization
He and Wang [68]	2007	HPSO	Simulated annealing	Constrained optimization
Mo et al. [69]	2007	PIES	Evolution strategy	Unconstrained global optimization
Fan and Zahara [70]	2007	NM-PSO	Nelder Mead simplex method	Unconstrained global optimization
Shen et al. [74]	2007	HPSOTS	Tabu search	Gene selection and tumor classification
Shelokar [59]	2007	PSACO	Ant colony optimization	Unconstrained global optimization
Ge et al. [75]	2008	HIA	Artificial immune system	Job shop scheduling
Song et al. [76]	2008	HPSCO	Cooperative optimization	industrial optimization
Kao et al. [77]	2008	K-NM-PSO	K-means, NM	Data clustering
Firouzi et al. [78]	2008	PSO-SA-K	Simulated annealing, K-means	Cluster analysis
Pant et al. [87]	2008	AMPSO	Evolutionary programming	Unconstrained global optimization
Li et al. [89]	2008	Hybrid algorithm	Simplex + immune	unconstrained global optimization, system of nonlinear equations
Cui et al. [90]	2008	Hybrid algorithm	Fitness uniform selection strategy + random walk strategy	Unconstrained global optimization
Holden and Freitas [88]	2008	PSO/ACO	Ant colony optimization	Data mining
Kavehand and Talatahari [83]	2009	HPSACO	Ant colony optimization	Engineering design
Kavehand and Talatahari [84]	2009	DHPSACO	Ant colony optimization	Engineering design
Zahara and Kao [72]	2009	NM-PSO	Nelder Mead simplex method	Engineering design
Murthy et al. [79]	2009	–	Pf-PSO, ePSO	Unconstrained global optimization
Kuo et al. [80]	2009	HPSO	Random-key encoding and individual enhancement	Flow shop scheduling
Chen et al. [81]	2010	PSO-EO	Extremal Optimization	Unconstrained global optimization

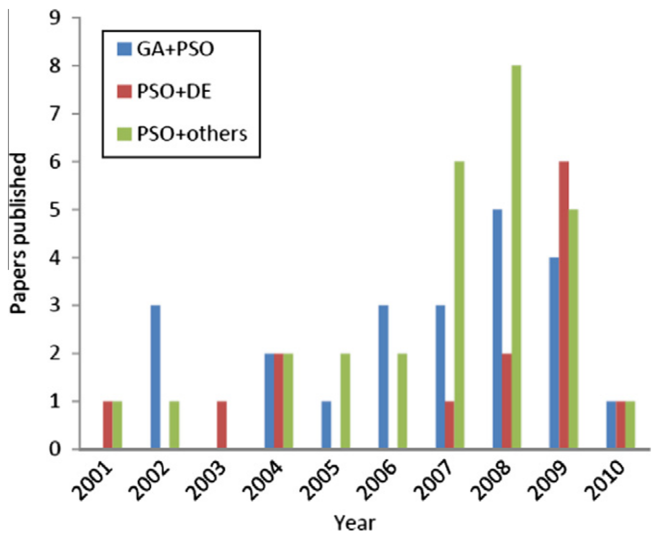


Fig. 3. An year wise distribution of number of hybridized PSO papers published.

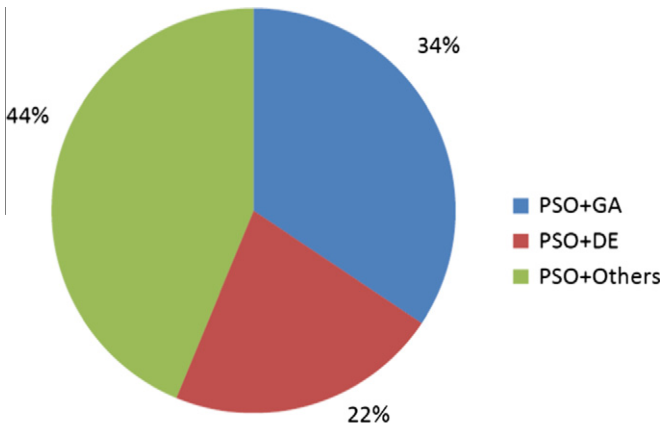


Fig. 4. Pie chart showing the distribution of publication of research articles having hybrid versions of PSO.

The idea of embedding swarm directions in Fast Evolutionary Programming (FEP) [85] to improve the performance of the latter was proposed by Wei et al. [86]. They applied their algorithm on a set of 6 unconstrained benchmark problems. The AMPSO algorithm proposed by Pant et al. [87] is hybrid version of PSO including EP based adaptive mutation operator using Beta distribution. They tested their algorithm on a set of unconstrained benchmark problems. Holden and Freitas [88] proposed a hybrid version PSO and ACO and they applied their algorithm in data mining. Li et al. [89] proposed the hybridization of PSO in which the orthogonal principal is applied to initialize the particles so that the particles are evenly distributed and then during the process of evolution, cloning and mutation operations are performed. Also the velocity of the particles is modified by using the simplex principle. The authors applied their algorithm for solving a multimodal Griewank function and a system of nonlinear equations. Another version of hybridizing PSO was suggested by Cui et al. [90] by combining PSO with *fitness uniform selection strategy* or FUSS and with *random walk strategy* or RWS. The former strategy induces a weak selection pressure into the standard PSO algorithm while the latter helps in enhancing the exploration capabilities to escape the local minima. They applied their algorithm for solving high dimension unconstrained global optimization problems with dimensions up to 3000. The summary of the main features of the hybridized PSO versions with other optimization algorithms are given in Table 3.

Fig. 3 gives the bar diagram of the number of hybrid PSO papers published in the year 2001 to 2010. The distribution of publication of research articles having hybrid versions of PSO is shown in Fig. 4 via pie chart.

4. Some hybrid PSO algorithms taken for comparison

In this section we give comparison of some simple hybrid versions of PSO viz. DE-PSO [55], AMPSO [87] and GA-PSO [32]. The DE-PSO and GA-PSO algorithms are, as the name suggests, hybrid versions of PSO with Differential Evolution and genetic algorithms. AMPSO combines PSO with EP. The authors have given two versions of AMPSO; AMPSO1 and AMPSO2. All these algorithms are easy to apply and have been successfully applied to solve unconstrained benchmark problems.

4.1. DE-PSO: a hybrid algorithm of DE and PSO

The DE-PSO algorithm was proposed by Pant et al. [55]. It starts like the usual DE algorithm up to the point where the trial vector is generated. If the trial vector is better than the corresponding target vector, then it is included in the population otherwise the algorithm enters the PSO phase and generates a new candidate solution using Particle Swarm's velocity and position update equations with the hope of finding a better solution. The method is repeated iteratively till the optimum value is reached. The inclusion of PSO phase creates a perturbation in the population, which in turn helps in maintaining diversity of the population and producing an optimal solution. The pseudo code of the Hybrid DE-PSO algorithm is given in Fig. 5. In Fig. 5, w, c_1, c_2, r_1, r_2 are particle swarm's control parameters and P_i and P_g are particle's personal best and global best positions. F and Cr represents the control parameters amplitude factor and crossover rate respectively for the DE algorithm.

```
Initialize particle's position and velocity vectors
Do
For i = 1 to N (Population size) do
    Select $r_1$, $r_2$, $r_3 \in$ N randomly
    // $r_1$, $r_2$, $r_3$ are selected such that $r_1 \neq r_2 \neq r_3$//
    For j = 1 to D (dimension) do
        Select $j_{rand} \in D$
        If (rand () < CR or j = $j_{rand}$)
            // rand () denotes a uniformly distributed random number
            between 0 and 1  //
            $U_{ij,g+1} = x_{r_1,g} + F*(x_{r_2,g} - x_{r_3,g})$
        End if
    End for
    If ( $f(U_{i,g+1}) < f(X_{i,g})$ ) then
        $X_{i,g+1} = U_{i,g+1}$
    Else
        //  PSO phase activated
        Find a new particle using Particle Swarm's velocity and position
        update equations.
        Let this particle be $TX = (tx_1, tx_2, ..., tx_D)$  //
        For j = 1 to D (dimension) do
            $v_{ij,g+1} = w*v_{ij,g} + c_1 r_1 (P_{ij,g} - x_{ij,g}) + c_2 r_2 (P_{gj,g} - x_{ij,g})$
            $tx_{ij} = x_{ij,g} + v_{ij,g+1}$
        End for
        If ( $f(TX_i) < f(X_{i,g})$ ) then
            $X_{i,g+1} = TX_i$
        Else
            $X_{i,g+1} = X_{i,g}$
        End if
    End if
End for.
Until stopping criteria is reached
```

Fig. 5. Pseudo code of DE-PSO algorithm.

4.2. AMPSO: A hybrid algorithm of PSO and EP

The AMPSO algorithm proposed by Pant et al. [87] is a simple, modified version of PSO including EP based adaptive mutation operator using Beta distribution. Two versions of AMPSO namely AMPSO1 and AMPSO2 are proposed. AMPSO1 and AMPSO2 differ from each other in the sense that in AMPSO1, the personal best (P_i) position of the swarm particle is mutated and in AMPSO2, the global best (P_g) position of the particle is mutated.

The EP phase is activated at the end of each iteration (after the velocity and position vector update is complete), where the particles are mutated according to the following rule:

$$x_{ij} = x_{ij} + \sigma'_{ij} * Betarand_j() \tag{11}$$

where, $\sigma'_{ij} = \sigma_{ij} * \exp(\tau N(0,1) + \tau' N_j(0,1))$

$N(0,1)$ denotes a normally distributed random number with mean zero and standard deviation one. $N_j(0,1)$ indicates that a different random number is generated for each value of j. τ and τ' are set as $1/\sqrt{2n}$ and $1/\sqrt{2\sqrt{n}}$ respectively [7], where n is the population size. $Betarand_j()$ is a random number generated by beta distribution with parameters less than 1.

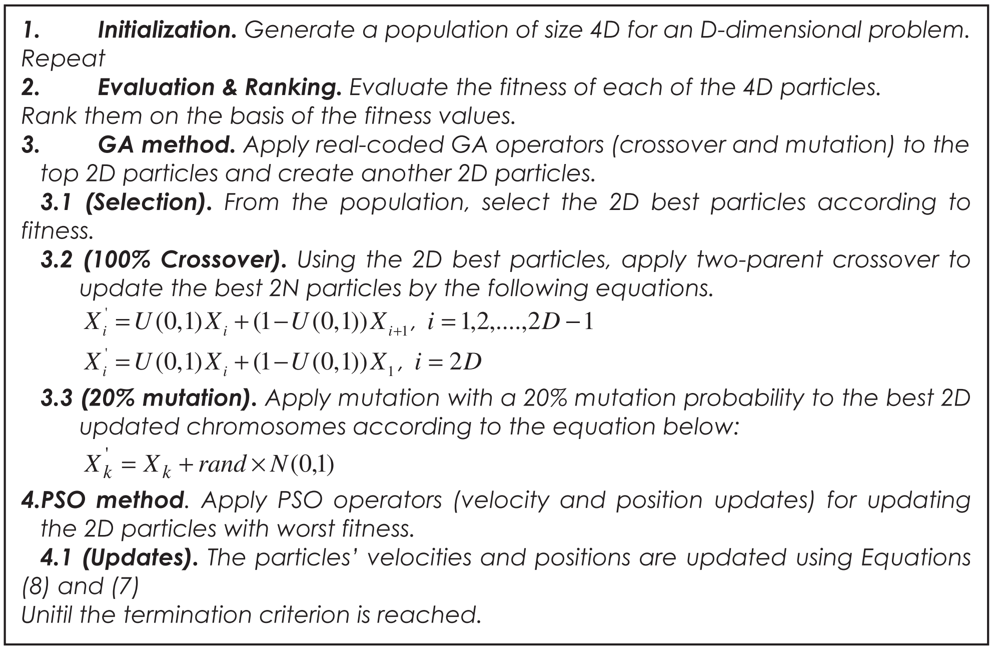
The pseudo code of AMPSO1 algorithm is given in Fig. 6. Here, $U(0,1)$ is an uniformly distributed random number in the interval $(0,1)$ and D is the dimension. The algorithmic steps of AMPSO2 algorithm are the same as the above, where the global *best particle (Pg)* is mutated instead of the personal best position (P_i).

4.3. GAPSO: a hybrid algorithm of GA and PSO

The GA-PSO algorithm was proposed by Kao and Zahara [32]. In GAPSO, for solving a D dimensional problem, the hybrid approach takes *4D* individuals that are randomly generated. These individuals may be regarded as chromosomes in the case

```
Initialize particle's position and velocity vectors. Each particle is taken as a pair of real-
valued vectors, $(X_i, \sigma_i)$. The $X_i$'s give the i^th particle of the swarm and $\sigma_i$'s the
associated strategy parameters. i varies from 1 to N (population size)
Do
// Update velocity and position vector
For j = 1 to D (dimension) do
        $v_{ij} = w * v_{ij} + c_1 r_1 (P_{ij} - x_{ij}) + c_2 r_2 (P_{gj} - x_{ij})$
        $x_{ij} = x_{ij} + v_{ij}$
End for
Calculate the fitness value, $f(X_i)$, of each particle
If ( $f(X_i) < f(P_i)$ ) $P_i = X_i$
        If ( $f(P_i) < f(P_g)$ ) $P_g = P_i$
        End if
End if
If (U (0, 1) < 1/D) then
    // Apply mutation to $P_i$ (personal best position of particle) using Eqn. (11) //
        For j = 1 to D (dimension) do
                $P'_{ij} = P_{ij} + \sigma'_{ij} * Betarand_j()$
                $\sigma'_{ij} = \sigma_{ij} * \exp(\tau N(0,1) + \tau' N_j(0,1))$
        End for
End if
Evaluate the fitness value of $P'_i$, $f(P'_i)$
If ( $f(P'_i) < f(P_i)$ ) $P_i = P'_i$
        If ( $f(P_i) < f(P_g)$ ) $P_g = P_i$
        End if
End if
Until stopping criteria is reached
```

Fig. 6. Pseudo code of AMPSO1 algorithm.



```
1.    Initialization. Generate a population of size 4D for an D-dimensional problem.
Repeat
2.    Evaluation & Ranking. Evaluate the fitness of each of the 4D particles.
Rank them on the basis of the fitness values.
3.    GA method. Apply real-coded GA operators (crossover and mutation) to the
  top 2D particles and create another 2D particles.
  3.1 (Selection). From the population, select the 2D best particles according to
fitness.
  3.2 (100% Crossover). Using the 2D best particles, apply two-parent crossover to
      update the best 2N particles by the following equations.
      $X_i^{'} = U(0,1)X_i + (1-U(0,1))X_{i+1},\ i = 1,2,....,2D-1$
      $X_i^{'} = U(0,1)X_i + (1-U(0,1))X_1,\ i = 2D$
  3.3 (20% mutation). Apply mutation with a 20% mutation probability to the best 2D
      updated chromosomes according to the equation below:
      $X_k^{'} = X_k + rand \times N(0,1)$
4.PSO method. Apply PSO operators (velocity and position updates) for updating
  the 2D particles with worst fitness.
  4.1 (Updates). The particles' velocities and positions are updated using Equations
(8) and (7)
Unitil the termination criterion is reached.
```

Fig. 7. Flow of GA-PSO algorithm.

of GA, or as particles in the case of PSO. These *4D* individuals are sorted by fitness, and the top *2D* individuals are fed into the real-coded GA to create *2D* new individuals by crossover and mutation operations. The crossover operator of the real-coded GA is implemented by borrowing the concept of linear combination of two vectors, which represent two individuals in this algorithm, with a 100% crossover probability.

They proposed a random mutation operator for the real-coded GA so as to modify an individual with a random number in the problem's domain with a 20% probability. The new *2D* individuals created from real-coded GA are used to adjust the remaining *2D* particles by the PSO method. The procedure of adjusting the *2D* particles in the PSO method involves selection of the global best particle, selection of the neighborhood best particles, and finally velocity updates. The global best particle of the population is determined according to the sorted fitness values. The neighborhood best particles are selected by first evenly dividing the *2D* particles into *D* neighborhoods and designating the particle with the better fitness value in each neighborhood as the neighborhood best particle. The velocity and position updates for each of the *2D* particles are then carried out with the help of Eqs. (8) and (7). The result is sorted in preparation for repeating the entire run. The algorithm terminates when it satisfies a convergence criterion that is based on the standard deviation of the objective function values of D + 1 best individuals of the population. The algorithmic steps of GA-PSO algorithm is given in Fig. 7. In Fig. 7, $U(0,1)$ is an uniformly distributed random number between 0 and 1; $N(0,1)$ is a Gaussian distributed random number; D is the dimension.

5. Experimental settings and results discussion

5.1. Parameter settings

In order to make a fair comparison of all the algorithms, we fixed the same seed for random number generation so that the initial population is same for all the algorithms. For PSO, the control parameters are the inertia weight *w* and the acceleration constants c_1 and c_2. The two main control parameters associated with DE are the crossover rate C_r and the scaling factor *F*. For GA, the control parameters are mutation probability and the crossover probability. The control parameters require a proper setup (fine tuning) for the optimum performance of the algorithm. In order to have comparable results, the parameter settings for DE-PSO and AMPSO were chosen based on the results of GA-PSO algorithm reported in [32], which are given in Table 4.

5.2. Benchmark problems

For the present study, a test suit of nine unconstrained benchmark problems are considered. Mathematical models of the problems are given in Table 5. Most of these problems have been considered for validating an optimization algorithm. In this test suite, five problems are having the non zero optimum and the remaining four are having zero optimum. For the present

Table 4
Experimental setting for DE-PSO, AMPSO1, AMPSO2 and GA-PSO algorithms.

Parameters	DE-PSO	AMPSO 1 and AMPSO2	GA-PSO
Dimension	2–10	2–10	2–10
Population Size	4D	4D	4D
w	Linearly decreasing from 0.9 to 0.4	Linearly decreasing from 0.7 to 0.4	0.5 + r/2.0, r is a uniformly distributed random number in the range [0,1]
c_1, c_2	2.0	1.49	2.0
Probability distribution for mutation	-NA*-	Beta distribution	-NA*-
C_r	0.2	-NA*-	-NA*-
F	0.5	-NA*-	-NA*-
Mutation rate	-NA*-	-NA*-	20%
Crossover rate	-NA*-	-NA*-	100%
Number of runs	A total of 100 runs for each experimental setting were conducted.		
Stopping criteria	$\|f_{\min} - f^*\| \leqslant e^{-4}$, f^* is the global optimum of function f.		
Computer settings	All experiments are conducted on a PIV PC using DEV C++		

*NA – Not applicable.

Table 5
Numerical benchmark problems.

Function	Dimension	Range	Optimum
Rosenbrock function (R_n) $f(x) = \sum_{i=0}^{n-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$	2, 5	$[-10, 10]$	0
DeJong function (DJ) $f(x) = \sum_{i=1}^{n} x_i^2$	2	$[-5.12, 5.12]$	0
Goldstein and Price function (GP) $(x) = \{1 + (x_0 + x_1 + 1)^2(19 - 14x_0 + 3x_0^2 - 14x_1 + 6x_0x_1 + 3x_1^2)\}$ $\{30 + (2x_0 - 3x_1)^2(18 - 32x_0 + 12x_0^2 + 48x_1 - 36x_0x_1 + 27x_1^2)\}$	2	$[-2, 2]$	3
Branin RCOC function (RC) $f(x) = \left(x_1 - \frac{5.1}{4\pi^2}x_0^2 + \frac{5}{\pi}x_0 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_0) + 10$	2	$[-10, 10]$	0.397887
Shubert function (SH) $f(x) = \sum_{j=1}^{5} j\cos((j+1)x_1 + j)\sum_{j=1}^{5} j\cos((j+1)x_2 + j)$	2	$[-10, 10]$	-186.7309
Easom function (ES) $f(x) = -\cos(x_1)\cos(x_2)\exp(-((x_1 - \pi)^2 + (x_2 - \pi)^2))$	2	$[-100, 100]$	-1
Hartmann function ($H_{3,4}$) $f(x) = -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2\right)$	3	[0, 1]	-3.86278
B2 function (B2) $f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$	2	$[-100, 100]$	0
Zakharov function (Z_n) $f(x) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^4$	2, 5, 10	$[-10, 10]$	0

For function $H_{3,4}$,

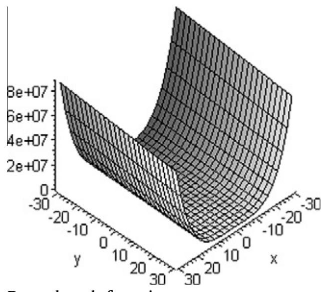
$$\alpha = [1\ 1.2\ 3\ 3.2], \quad A = \begin{bmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}, \quad P = \begin{bmatrix} 0.3689 & 0.117 & 0.2673 \\ 0.4699 & 0.4387 & 0.747 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{bmatrix}.$$

study, the benchmark problems are tested with dimensions 2 to 10. The three dimensional graphs of the test functions are illustrated in Table 6.

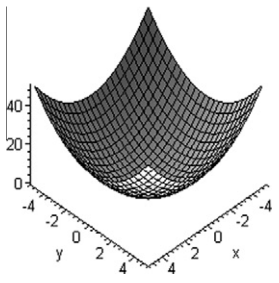
The special properties of the benchmark functions taken in this study may be described as:

- The first function is Rosenbrock function. It is a classic optimization problem with a narrow global optimum hidden inside a long, narrow, curved flat valley. It is unimodal, yet due to a saddle point it is very difficult to locate the minimum. This function is also known as banana valley function.
- The second function is DeJong, also known as spherical function is a continuous, strictly convex and unimodal function and usually do not pose much difficulty for an optimization algorithm.
- Goldstein and Price function is the third function. It is a multimodal function having 4 local minima and one global minimum.
- The fourth function is the Branin function. It is a multimodal function having 3 global minima.

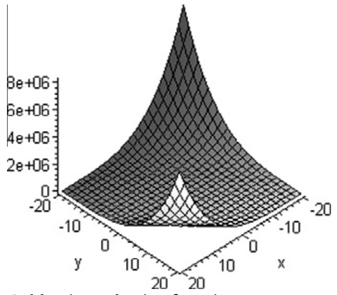
Table 6
3D plots of benchmark problems.



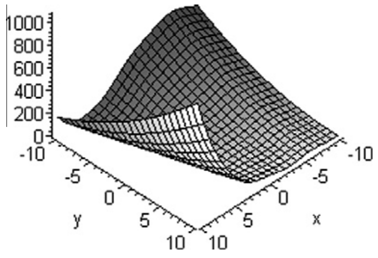
Rosenbrock function



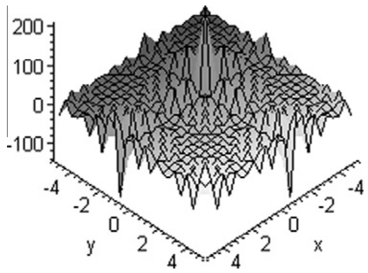
DeJong function



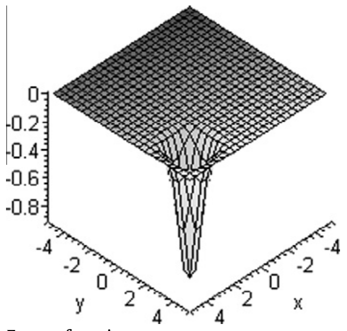
Goldstein and Price function



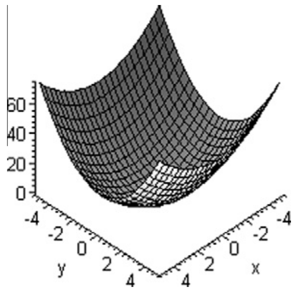
Branin function



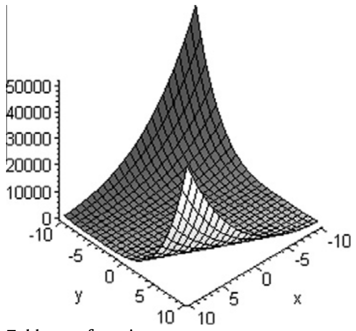
Shubert function



Easom function



B2 function



Zakharov function

- Shubert function is the fifth function. It is highly multimodal function having several local minima.
- The sixth function is Easom function; it is also highly multimodal function having several local minima.
- Hartmann function is the seventh function; which is multimodal and having 4 local minima and one global minimum.
- The function B2 is a multimodal function and is having several local minima and one global minimum.
- The ninth function Zakharov is unimodal in nature.

5.3. Results discussion

Performance analysis of the considered algorithms DE-PSO, AMPSO1, AMPSO2 and GA-PSO is given in Table 7. In Table 7, bold values refer the best result among all the compared algorithms for each function. Here we would like to mention that

Table 7
Comparison results of DE-PSO [55], AMPSO1 [87], AMPSO2 [87] and GA-PSO [32] algorithms.

Functions	Average number of function evaluations				Average error			
	GA-PSO	DE-PSO	AMPSO1	AMPSO2	GA-PSO	DE-PSO	AMPSO1	AMPSO2
R_2	140894	**2351**	2668	2750	0.00064	2.46e−5	**2.16e−5**	3.03e−5
R_5	1358064	**63106**	50992	50064	0.00013	**7.74e−5**	9.73e−5	9.93e−5
DJ	**206**	938	980	620	0.0004	**1.67e−5**	4.23e−5	3.34e−5
GP	25706	1155	**1120**	1196	0.00012	**3.67e−5**	3.81e−5	9.89e−6
RC	8254	1148	**754**	2778	0.00009	**2.29e−6**	1.52e−5	1.49e−5
SH	96211	3689	**2132**	2146	0.00007	3.03e−6	1.72e−5	**2.54e−7**
ES	**809**	1792	1258	1432	0.00003	**1.67e−5**	4.10e−5	3.71e−5
$H_{3,4}$	2117	1059	**1020**	1132	0.00020	**1.74e−5**	1.96e−5	2.04e−5
B2	**174**	928	622	679	0.00001	**1.11e−5**	1.17e−6	2.64e−5
Z_2	**95**	834	756	780	0.00005	1.12e−5	2.54e−5	**2.61e−6**
Z_5	**398**	2677	1992	2018	**0.00000**	3.49e−5	3.44e−5	5.07e−5
Z_{10}	**872**	7258	5980	6218	**0.00000**	6.91e−5	7.41e−5	9.11e−5

	Success rate (%)			
	GA-PSO	DE-PSO	AMPSO1	AMPSO2
R_2	100	100	100	100
R_5	100	100	100	100
DJ	100	100	100	100
GP	100	100	100	100
RC	100	100	100	100
SH	100	100	100	100
ES	100	100	100	100
$H_{3,4}$	100	100	100	100
B2	100	100	100	100
Z_2	100	100	100	100
Z_5	100	100	100	100
Z_{10}	100	100	100	100

the results of GA-PSO algorithm reported in Table 7 were taken from [32]. In these comparisons, we have recorded the average number of function evaluations (NFE), average error and success rate because they are the only performance measures mentioned in [32]. If we compare the results in terms of NFE then from the numerical results we can see that the performance of GA-PSO algorithm is better than the other compared algorithms in 6 test cases out of 12 cases tried; AMPSO1 algorithm gave better results than all the others in 4 test cases; and the remaining two test cases DE-PSO is superior with others. But it is wonder in terms of average error, the DE-PSO algorithm is performed well in 7 test cases in comparison with all the other compared algorithms. The next performance measure is success rate. In terms of success rate all the algorithms gave the same performance i.e. 100% success for all the considered problems.

6. Conclusion

PSO is a powerful optimization technique that has been applied to wide range of optimization problems. Nevertheless its performance can be enhanced manyfolds with the aid of certain modifications. The present research article focuses on the concept of hybridization, which nowadays is a popular idea being applied to evolutionary algorithms in order to increase their efficiency and robustness. In this paper we present a brief review on the hybrid algorithms in which PSO is one of the main algorithms.

For the present article we studied 64 research articles. Here, we would also like to add that although we have tried to include as many references as possible but there may be several instances available in literature which we might have missed. Out of the considered articles, it was observed that the integration of PSO with GA has been the most popular choice among researchers with 22 references. The second choice opted by several researchers is a combination of PSO and DE algorithm with 14 references. Regarding the blending of PSO and other algorithms we observe that PSO/ACO combination is favored for solving discrete optimization problems like that of scheduling. This is understandable as the structure of ACO is naturally suited for solving discrete problems. The other algorithms also include TS, SA, Fuzzy logic etc. Among the local search methods Nelder Mead Simplex Method appears to be the researcher's choice for mixing with PSO. Some other local search methods include Quadratic Programming approach, Interior point method etc. Overall we considered 28 articles which give a hybrid of PSO with other algorithms.

Mainly, the integration of strategies is done in one of the following ways; (1) one algorithm being used as a pre-optimizer for the initial population of the other algorithm, (2) dividing the entire population into subpopulations and evolving these populations using PSO and other algorithms and (3) incorporating the unique operators of one algorithm as a local search enhancement tool for other algorithm.

If we talk about the application of the hybrid algorithms considered in this article we see that these algorithms have been applied to solve a diverse set of problems including medical image processing, engineering design problems, scheduling, data clustering, geometric place problems etc. However the most common application of these algorithms has been to solve the unconstrained global optimization problems, the second choice has been their application to power system and engineering design problems.

Besides giving the review, this paper also analyses the performance of three simple and easy hybrid PSO algorithms for solving unconstrained global optimization problems. These are; GA-PSO, DE-PSO and AMPSO. GA-PSO is an integration of PSO and GA algorithm; DE-PSO is a hybrid version of DE and PSO and AMPSO is a combination of PSO and EP. The performance of these algorithms is analyzed on a set of nine benchmark problems taken from the literature and it was observed that DE-PSO, the hybrid of DE with PSO was somewhat better than the other suggested algorithms in terms of average error. However, in terms of average NFE all the algorithms performed more or less in a similar manner while in terms of success rate all algorithms gave 100% success in all the test cases.

On a concluding note we would like to say that hybridization of algorithms is an interesting and promising field that can give us more insights regarding the behavior and potential advantages and disadvantages of different metaheuristics. The present study may motivate and help the researchers working the field of evolutionary algorithms to develop new hybrid models or to apply the existing models to new application areas.

Acknowledgements

This work was carried out during the tenure of an ERCIM ‘‘Alain Bensoussan’’ Fellowship Programme. The second author would like to acknowledge the DST for financial support. Also all the authors are grateful to the constructive comments of the unknown referees.

References

[1] J.H. Holland, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence, Ann. Arbor., MI, University of Michigan Press.
[2] J. Kennedy, R.C. Eberhart, Particle Swarm Optimization, in: Proceedings of the IEEE International Conference on Neural Networks, Piscataway, vol. IV, 1995, pp. 1942–1948.
[3] K. Price, R. Storn, Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report, International Computer Science Institute, Berkley, 1995.
[4] L.J. Fogel, A.J. Owens, M.J. Walsh, Artificial intelligence through a simulation of evolution, in: M. Maxfield, A. Callahan, L.J. Fogel, (eds.), Biophysics and Cybernetic systems, Proceedings of the 2nd Cybernetic Sciences Symposium, 1965, pp. 131–155, (Spartan Books).
[5] M. Dorigo, V. Maniezzo, A. Colorni, Positive feedback as a search strategy, Technical Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, IT, 1991.
[6] Torn, A. Zilinskas (Eds.), Global Optimization, Lecture Notes in Computer Science, vol. 350, Springer-Verlag, 1989.
[7] P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, John Wiley & Sons Ltd, 2005.
[8] R.C. Eberhart, Y. Shi, Particle swarm optimization: developments, applications and resources, Proceedings of the IEEE Congress of Evolutionary Computation 1 (2001) 27–30.
[9] J. Kennedy, The particle swarm: social adaptation of knowledge, Proceedings of the IEEE International Conference on Evolutionary Computation (1997) 303–308.
[10] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proceedings of the IEEE Congress of Evolutionary Computation, 1998, pp. 69–73.
[11] R.C. Eberhart, Y. Shi, Comparison between genetic algorithms and particle swarm optimization, in: Proceedings of the Seventh Annual Conference on Evolutionary Programming, 1998, pp. 611–616.
[12] M. Clerc, The swarm and the queen: towards a deterministic and adaptive particle swarm optimization, Proceedings of the IEEE Congress on Evolutionary Computation 3 (1999) 1951–1957.
[13] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multimodal complex space, IEEE Transactions on Evolutionary Computation 6 (2002) 58–73.
[14] R.C. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, Proceedings of the IEEE Congress on Evolutionary Computation 1 (2000) 84–88.
[15] Carlisle, G. Dozier, An off-the-shelf PSO, in: Proceedings of the Workshop on Particle Swarm Optimization, Indianapolis, USA, 2001, pp. 1–6.
[16] D. Goldberg, Genetic Algorithms in Search Optimization and Machine Learning, Addison Wesley Publishing Company, Reading, Massa chutes, 1986.
[17] J. Robinson, S. Sinton, Y.R. Samii, Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna, in: Proceedings of the IEEE International Symposium in Antennas and Propagation Society, 2002, pp. 314–317.
[18] T. Krink, M. Løvbjerg, The lifecycle model: combining particle swarm optimization, genetic algorithms and hill climbers, Proceedings of the Parallel Problem Solving From Nature (2002) 621–630.
[19] E. Conradie, R. Miikkulainen, C. Aldrich, Intelligent process control utilising symbiotic memetic neuro-evolution, Proceedings of the IEEE Congress on Evolutionary Computation 1 (2002) 623–628.
[20] E.A. Grimaldi, F. Grimacia, M. Mussetta, P. Pirinoli, R.E. Zich, A new hybrid genetical – swarm algorithm for electromagnetic optimization, in: Proceedings of International Conference on Computational Electromagnetics and its Applications, Beijing, China, 2004, pp. 157–160.
[21] Gandelli, F. Grimaccia, M. Mussetta, P. Pirinoli, R.E. Zich, Development and validation of different hybridization strategies between GA and PSO, Proceedings of the IEEE Congress on Evolutionary Computation (2007) 2782–2787.
[22] Gandelli, F. Grimaccia, M. Mussetta, P. Pirinoli, R.E. Zich, Genetical swarm optimization: a new hybrid evolutionary algorithm for electromagnetic application, in: Proceedings of the 18th International Conference on Applied Electromagnetics, ICECcom 2005, Dubrovnik, Croatia, 2005, pp. 269–272.
[23] Grimaccia, M. Mussetta, P. Pirinoli, R.E. Zich, Genetical swarm optimization: self adaptive hybrid evolutionary algorithm for electromagnetic, IEEE Transactions on Antennas and Propagation 55 (3) (2007) 781–785.
[24] Gandelli, F. Grimaccia, M. Mussetta, P. Pirinoli, R.E. Zich, Genetical swarm optimization: an evolutionary algorithm for antenna design, Journal of AUTOMATIKA 47 (3-4) (2006) 105–112.
[25] C – F. Juang, A hybrid of genetic algorithm and particle swarm optimization for recurrent network design, IEEE Transactions On Systems, Man, And Cybernetics-Part B: Cybernetics 34 (2004) 997–1006.
[26] M. Settles, T. Soule, Breeding swarms: a GA/PSO hybrid, Proceedings of Genetic and Evolutionary Computation Conference (2005) 161–168.
[27] M. Jian, Y. Chen, Introducing recombination with dynamic linkage discovery to particle swarm optimization, Proceedings of the Genetic and Evolutionary Computation Conference (2006) 85–86.
[28] A. Mohammadi, M. Jazaeri, A hybrid particle swarm optimization-genetic algorithm for optimal location of SVC devices in power system planning, in: Proceedings of 42nd International Universities Power Engineering Conference, 2007, pp.1175–1181.

[29] A.A. Esmin, G. Lambert-Torres, G.B. Alvarenga, Hybrid evolutionary algorithm based on PSO and GA mutation, in: Proceedings of 6th International Conference on Hybrid Intelligent Systems, 2006, pp. 57–62.
[30] H. Kim, Improvement of genetic algorithm using PSO and Euclidean data distance, International Journal of Information Technology 12 (2006) 142–148.
[31] Yang, Y. Chen, Z. Zhao, A hybrid evolutionary algorithm by combination of PSO and GA for unconstrained and constrained optimization problems, in: Proceedings of the IEEE International Conference on Control and Automation, 2007, pp. 166–170.
[32] Y.T. Kao, E. Zahara, A hybrid genetic algorithm and particle swarm optimization for multimodal functions, Applied Soft Computing 8 (2008) 849–857.
[33] N. Ru, Y. Jianhua, A GA and particle swarm optimization based hybrid algorithm, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2008, pp. 1047–1050.
[34] A. Shunmugalatha, S.M.R. Slochanal, Optimum cost of generation for maximum loadability limit of power system using hybrid particle swarm optimization, International Journal of Electrical Power & Energy Systems 30 (2008) 486–490.
[35] T. Li, L. Xu, X.W. Shi, A hybrid of genetic algorithm and particle swarm optimization for antenna design, PIERS online 4 (2008) 56–60.
[36] G.K. Mahanti, A. Chakrabarty, Phase-only and amplitude-phase synthesize of dual- pattern linear antenna arrays using floating-point genetic algorithms, Progress in Electro-magnetics Research, PIER 68 (2007) 247–259.
[37] T.O. Ting, K.P.Wong, C.Y. Chung, Hybrid constrained genetic algorithm/particle swarm optimization load flow algorithm, in: IET Proceedings of Generation, Transmission & Distribution, vol. 2, 2008, pp. 800–812.
[38] S. Jeong, S. Hasegawa, K. Shimoyama, S. Obayashi, Development and investigation of efficient GA/PSO-hybrid algorithm applicable to real-world design optimization, IEEE Computational Intelligence Magazine (2009) 36–44.
[39] F. Valdez, P. Melin, O. Castillo, Evolutionary method combining particle swarm optimization and genetic algorithms using fuzzy logic for decision making, in: Proceedings of the IEEE International Conference on Fuzzy Systems, 2009, pp. 2114–2119.
[40] R. Bhuvaneswari, V.P. Sakthivel, S. Subramanian, G.T. Bellarmine, Hybrid approach using GA and PSO for alternator design, in: Proceedings of the IEEE International Conference Southeastcon, Atlanda, 2009, pp. 169–174.
[41] K. Premalatha, A.M. Natarajan, Discrete PSO with GA operators for document clustering, International Journal of Recent Trends in Engineering 1 (2009) 20–24.
[42] R.F. Abdel-Kader, Genetically improved PSO algorithm for efficient data clustering, in: Proceedings of International Conference on Machine Learning and Computing, 2010, pp. 71–75.
[43] T. Hendtlass, A Combined Swarm differential evolution algorithm for optimization problems, in: Proceedings of 14th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Lecture Notes in Computer Science, vol. 2070, Springer Verlag, 2001, pp. 11–18.
[44] W.J. Zhang, X.F. Xie, DEPSO: hybrid particle swarm with differential evolution operator, in: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMCC), Washington DC, USA, 2003, pp. 3816–3821.
[45] S. Kannan, S.M.R. Slochanal, P. Subbaraj, N.P. Padhy, Applications of particle swarm optimization techniques and its variants to generation expansion planning, Electric Power Systems Research 70 (2004) 203–210.
[46] H. Talbi, M. Batouche, Hybrid particle swarm with differential evolution for multimodal image registration, Proceedings of the IEEE International Conference on Industrial Technology 3 (2004) 1567–1573.
[47] Z. -F. Hao, G. -H. Gua, H. Huang, A particle swarm optimization algorithm with differential evolution, in: Proceedings of Sixth International Conference on Machine Learning and Cybernetics, 2007, pp. 1031–1035.
[48] S. Das, A. Abraham, A. Konar, Particle swarm optimization and differential evolution algorithms: technical analysis, applications and hybridization perspectives, in: Ying Liu et al. (Eds.), Advances of Computational Intelligence in Industrial Systems, Studies in Computational Intelligence, Springer Verlag, Germany, 2008, pp. 1–38.
[49] M. Omran, A.P. Engelbrecht, A. Salman, Bare bones differential evolution, European Journal of Operational Research 196 (2008) 128–139.
[50] G –N. Jose, E. Alba, J. Apolloni, Particle swarm hybridized with differential evolution: black box optimization benchmarking for noisy functions, in: Proceedings of International Conference Genetic and Evolutionary Computation, 2009, pp. 2343–2350.
[51] C. Zhang, J. Ning, S. Lu, D. Ouyang, T. Ding, A novel hybrid differential evolution and particle swarm optimization algorithm for unconstrained optimization, Operations Research Letters 37 (2009) 117–122.
[52] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Applied Soft Computing 10 (2009) 629–640.
[53] Caponio, F. Neri, V. Tirronen, Superfit control adaption in memetic differential evolution frameworks, Soft Computing 13 (2009) 811–831.
[54] W. Xu, X. Gu, A hybrid particle swarm optimization approach with prior crossover differential evolution, in: Proceedings of ACM/SIGEVO Summit on Genetic and Evolutionary Computation, 2009, pp. 671–678.
[55] M. Pant, R. Thangaraj, A. Abraham, DE-PSO: a new hybrid meta-heuristic for solving global optimization problems, New Mathematics and Natural Computation, Accepted for publication, 2009.
[56] S. Khamsawang, P. Wannakarn, S. Jiriwibhakorn Hybrid PSO-DE for solving the economic dispatch problem with generator constraints, in: Proceedings of the IEEE International Conference on Computer and Automation Engineering, vol. 5, 2010, pp. 135–139.
[57] J.A. Nelder, R. Mead, A simplex method for function minimization, Computer Journal 7 (1965) 308–313.
[58] F. Glover, Tabu search methods in artificial intelligence and operations research, in: ORSA Artificial Intelligence 1, Kluwer Academic Publishers., 1987.
[59] P.S. Shelokar, P. Siarry, V.K. Jayaraman, B.D. Kulkarni, Particle swarm and ant colony algorithms hybridized for improved continuous optimization, Applied Mathematics and Computation 188 (2007) 129–142.
[60] T. Hendtlass, M. Randall, A survey of ant colony and particle swarm meta-heuristics and their application to discrete optimization problems, in: Proceedings of the Inaugural Workshop on Artificial Life, 2001, pp. 15–25.
[61] T.A.A. Victoire, A.E. Jeyakumar, Hybrid PSO–SQP for economic dispatch with valve-point effect, Electric Power Systems Research 71 (2004) 51–59.
[62] P.T. Boggs, J.W. Tolle, Sequential quadratic programming, Acta Numerica, vol. 4, Cambridge University Press, Cambridge, 1995. pp.1–52.
[63] Grosan, A. Abraham, M. Nicoara, Search optimization using hybrid particle sub-swarms and evolutionary algorithms, International Journal of Simulation Systems, Science and Technology 6 (2005) 60–79.
[64] Grosan, Solving geometrical place problems by using evolutionary algorithms, in: M. Kaaniche (Ed.), Proceedings of World Computer Congress, Toulouse, France, 2004, pp. 365–375.
[65] J. Chuanwen, E. Bompard, A hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimization, Mathematics and Computers in Simulation 68 (2005) 57–65.
[66] H. Liu, A. Abraham, W. Zhang, A fuzzy adaptive turbulent particle swarm optimization, International Journal of Innovative Computing and Applications 1 (2007) 39–47.
[67] D.Y. Sha, C.-Y. Hsu, A hybrid particle swarm optimization for job shop scheduling problem, Computers & Industrial Engineering 51 (2006) 791–808.
[68] Q. He, L. Wang, A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization, Applied Mathematics and Computation 186 (2007) 1407–1422.
[69] W. Mo, S.-U. Guan, S.K. Puthusserypady, A novel hybrid algorithm for function optimization: particle swarm assisted incremental evolution strategy, Studies in Computational Intelligence 75 (2007) 101–125.
[70] S.K. Fan, E. Zahara, A hybrid simplex search and particle swarm optimization for unconstrained optimization, European Journal of Operational Research 181 (2007) 527–548.
[71] S.K. Fan, Y.C. Liang, E. Zahara, Hybrid simplex search and particle swarm optimization for the global optimization of multimodal functions, Engineering Optimization 36 (2004) 401–418.

[72] Zahara, Y.-T. Kao, Hybrid Nelder–Mead simplex search and particle swarm optimization for constrained engineering design problems, Expert Systems with Applications 36 (2009) 3880–3886.
[73] Q.J. Guo, H.B. Yu, A.D. Xu, A hybrid PSO-GD based intelligent method for machine diagnosis, Digital Signal Processing 16 (2006) 402–418.
[74] Q. Shen, W-M. Shi, W. Kong, Hybrid particle swarm optimization and tabu search approach for selecting genes for tumor classification using gene expression data, Computational Biology and Chemistry 32 (2007) 53–60.
[75] H. -W. Ge, L. Sun, Y. -C. Liang, F. Qian, An effective PSO and AIS-based hybrid intelligent algorithm for job-shop scheduling, IEEE Transactions on Systems, Man, And Cybernetics—Part A: Systems and Humans 38 (2008) 358–368.
[76] S. Song, L. Kong, Y. Gan, R. Su, Hybrid particle swarm cooperative optimization algorithm and its application to MBC in alumina production, Progress in Natural Science 18 (2008) 1423–1428.
[77] Y.-T. Kao, E. Zahara, I.-W. Kao, A hybridized approach to data clustering, Expert Systems with Applications 34 (2008) 1754–1762.
[78] B. Firouzi, T. Niknam, M. Nayeripour, A new evolutionary algorithm for cluster analysis, World Academy of Science, Engineering, and Technology 36 (2008) 605–609.
[79] R. Murthy, M.S. Arumugam, C.K. Loo, Hybrid particle swarm optimization algorithm with fine tuning operators, International Journal of Bio-Inspired Computation 1 (2009) 14–31.
[80] I. -H. Kuo, S. -J. Horng, T.-W. Kao, T.-L. Lin, C.-L. Lee, T. Terano, Y. Pan, An efficient flow-shop scheduling algorithm based on a hybrid particle swarm optimization model, Expert Systems with Applications 36 (2009) 7027–7032.
[81] M – R. Chen, X. Li, X. Zhang, Y – Z. Lu, A novel particle swarm optimizer hybridized with extremal optimization, Applied Soft Computing 10 (2010) 367–373.
[82] S. Boettcher, A.G. Percus, Extremal optimization: methods derived from coevolution, in: Proceedings of the Genetic and Evolutionary Computation Conference, 1999, pp. 825–832.
[83] Kavehand, S. Talatahari, Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures, Computer Structure 87 (2009) 267–283.
[84] Kavehand, S. Talatahari, A particle swarm ant colony optimization for truss structures with discrete variables, Constructional Steel Research 65 (2009) 1558–1568.
[85] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Transactions on Evolutionary Computation 3 (2) (1999) 82–102.
[86] Wei, Z. He, W. Pei, Swarm directions embedded in fast evolutionary programming, in: Proceedings of the World Congress on Computational Intelligence, vol. 2, 2002, pp. 1278–1283.
[87] M. Pant, R. Thangaraj, A. Abraham, Particle swarm optimization using adaptive mutation, in: Proceedings of 19th International Conference on Database and Expert Systems Application, Italy, 2008, pp. 519–523.
[88] N. Holden, A.A. Freitas, A hybrid PSO/ACO algorithm for discovering classification rules in data mining, Journal of Artificial Evolution and Applications (2008) 1–11, doi:10.1155/2008/316145.
[89] X. Li, H. Xu, Z. Cheng, One immune simplex particle swarm optimization and its application, in: Proceedings of the Fourth International Conference on Natural Computation, 2008, pp. 331–335.
[90] Z. Cui, X. Cai, J. Zeng, G. Sun, Particle swarm optimization with FUSS and RWS for high dimension functions, Applied Mathematics and Computation 205 (2008) 98–108.