01 Mar 2008

Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems

Yamille del Valle

Ganesh K. Venayagamoorthy
Missouri University of Science and Technology

Salman Mohagheghi

Ronald G. Harley

et. al. For a complete list of authors, see https://scholarsmine.mst.edu/ele_comeng_facwork/734

Recommended Citation

Y. del Valle et al., "Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems," *IEEE Transactions on Evolutionary Computation*, Institute of Electrical and Electronics Engineers (IEEE), Mar 2008.
The definitive version is available at https://doi.org/10.1109/TEVC.2007.896686

Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems

Yamille del Valle, *Student Member, IEEE*, Ganesh Kumar Venayagamoorthy, *Senior Member, IEEE*, Salman Mohagheghi, *Student Member, IEEE*, Jean-Carlos Hernandez, *Student Member, IEEE*, and Ronald G. Harley, *Fellow, IEEE*

***Abstract*—Many areas in power systems require solving one or more nonlinear optimization problems. While analytical methods might suffer from slow convergence and the curse of dimensionality, heuristics-based swarm intelligence can be an efficient alternative. Particle swarm optimization (PSO), part of the swarm intelligence family, is known to effectively solve large-scale nonlinear optimization problems. This paper presents a detailed overview of the basic concepts of PSO and its variants. Also, it provides a comprehensive survey on the power system applications that have benefited from the powerful nature of PSO as an optimization technique. For each application, technical details that are required for applying PSO, such as its type, particle formulation (solution representation), and the most efficient fitness functions are also discussed.**

***Index Terms*—Classical optimization, particle swarm optimization (PSO), power systems applications, swarm intelligence.**

I. Introduction

THE ELECTRIC power grid is the largest man-made machine in the world. It consists of synchronous generators, transformers, transmission lines, switches and relays, active/reactive compensators, and controllers. Various control objectives, operation actions, and/or design decisions in such a system require an optimization problem to be solved. For such a nonlinear nonstationary system with possible noise and uncertainties, as well as various design/operational constraints, the solution to the optimization problem is by no means trivial. Moreover, the following issues need attention: 1) the optimization technique selected must be appropriate and must suit the nature of the problem; 2) all the various aspects of the problem have to be taken into account; 3) all the system constraints should be correctly addressed; and 4) a comprehensive yet not too complicated objective function should be defined.

Various methods exist in the literature that address the optimization problem under different conditions. In its simplest form, this problem can be expressed as follows.

Find the minimum[1] of an objective function $f(\underline{x}) : A \rightarrow R$

$$\text{Subject to}: \quad \begin{aligned} g_i(\underline{x}) &= 0, \quad \text{for} \quad i = 1, \ldots, k \\ h_j(\underline{x}) &\leq 0, \quad \text{for} \quad j = 1, \ldots, m. \end{aligned}$$

Different optimization methods are classified based on the type of the search space $A \subseteq R^n$ and the objective (cost) function f. The simplest technique is *linear programming* (LP) which concerns the case where the objective function f is linear and the set A is specified using only linear equality and inequality constraints [1]. LP has been applied for solving various power system problems, such as planning and operation [2]–[5], economic dispatch [6], [7], state estimation [8]–[10], optimal power flow [11], [12], protection coordination [9], [13], unit commitment [10], [14], and maintenance scheduling [15]. For a special case, where some or all variables are constrained to take on integer values, the technique is referred to as *integer programming* [1]. Applications of integer or mixed-integer programming in power systems optimization problems have been reported for power system security assessment [16], unit commitment and generation planning [17]–[19], load management [20], distribution system planning [21], transmission system design and optimization [22]–[25], and reliability analysis [26].

However, in general, the objective function or the constraints or both contain nonlinearities, which raise the concept of *nonlinear programming* (NLP) [27]. This type of optimization technique has been extensively used by researchers for solving problems, such as power system voltage security [28], [29], optimal power flow [30]–[33], power system operation and planning [34]–[38], dynamic security [39], [40], power quality [41], unit commitment [42], reactive power control [43], capacitor placement [44], and optimizing controller parameters [45].

Since NLP is a difficult field, researchers have identified special cases for study. A particularly well-studied case [1] is the one where all the constraints g and h are linear. This problem is referred to as *linearly constrained optimization*. If in addition to linear constraints, the objective function is quadratic, the optimization problem is called *quadratic programming* (QP). This specific branch of NLP has also found widespread applications in the electric power field, in such areas as economic dispatch [46], [47], reactive power control [48]–[50], optimal power flow [48], [51], dc load flow [51], transmission system operation and planning [52], and unit commitment [53].

While deterministic optimization problems are formulated with known parameters, real-world problems almost invariably

Manuscript received August 21, 2006; revised December 15, 2006; accepted February 15, 2007. This work was supported in part by the National Science Foundation (NSF) under CAREER Grant ECCS 0348221 and in part by the Duke Power Company, Charlotte, NC.

Y. del Valle, S. Mohagheghi, J.-C. Hernandez, and R. G. Harley are with Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: yamille.delvalle@gatech.edu; salman@ece.gatech.edu; jean.hernandez@gatech.edu; rharley@ece.gatech.edu).

G. K. Venayagamoorthy is with the Real-Time Power and Intelligent Systems Laboratory, Department of Electrical and Computer Engineering, University of Missouri–Rolla, MO 65409 USA (e-mail: gkumar@ieee.org).

Digital Object Identifier 10.1109/TEVC.2007.896686

[1]The maximization problem of the function f is simply translated into the minimization problem of the function $-f$.

include some unknown parameters. This necessitates the introduction of *stochastic programming* models that incorporate the probability distribution functions of various variables into the problem formulation. In its most general case, the technique is referred to as *dynamic programming* (DP). Most of the applications of this optimization technique have been reported for solving problems such as power system operation and planning at the distribution level [54]–[59]. Research has also been conducted on applying DP to unit commitment [60], [61].

Although the DP technique has been mathematically proven to find an optimal solution, it has its own disadvantages. Solving the DP algorithm in most of the cases is not feasible. Even a numerical solution requires overwhelming computational effort, which increases exponentially as the size of the problem increases (curse of dimensionality). These restrictive conditions lead the solution to a suboptimal control scheme with limited look-ahead policies [62]. The complexity level is even further exacerbated when moving from finite horizon to infinite horizon problems, while also considering the stochastic effects, model imperfections, and the presence of the external disturbances.

Computational intelligence-based techniques, such as genetic algorithm (GA) and particle swarm optimization (PSO) can be solutions to the above problems. GA is a search technique used in computer science and engineering to find the approximate solutions to optimization problems [63]. GA represents a particular class of evolutionary algorithms that uses techniques inspired by evolutionary biology such as inheritance, mutation, natural selection, and recombination (or crossover). While it can rapidly locate good solutions, even for difficult search spaces, it has some disadvantages associated with it: 1) unless the fitness function is defined properly, GA may have a tendency to converge towards local optima rather than the global optimum of the problem; 2) operating on dynamic data sets is difficult; and 3) for specific optimization problems, and given the same amount of computation time, simpler optimization algorithms may find better solutions than GAs.

PSO is another evolutionary computation technique developed by Eberhart and Kennedy [64], [65] in 1995, which was inspired by the social behavior of bird flocking and fish schooling. PSO has its roots in artificial life and social psychology, as well as in engineering and computer science. It utilizes a "population" of particles that fly through the problem hyperspace with given velocities. At each iteration, the velocities of the individual particles are stochastically adjusted according to the historical best position for the particle itself and the neighborhood best position. Both the particle best and the neighborhood best are derived according to a user defined fitness function [65], [67]. The movement of each particle naturally evolves to an optimal or near-optimal solution. The word "swarm" comes from the irregular movements of the particles in the problem space, now more similar to a swarm of mosquitoes rather than a flock of birds or a school of fish [67].

PSO is a computational intelligence-based technique that is not largely affected by the size and nonlinearity of the problem, and can converge to the optimal solution in many problems where most analytical methods fail to converge. It can, therefore, be effectively applied to different optimization problems in power systems. A number of papers have been published in the past few years that focus on this issue. Moreover, PSO has some advantages over other similar optimization techniques such as GA, namely the following.

1) PSO is easier to implement and there are fewer parameters to adjust.
2) In PSO, every particle remembers its own previous best value as well as the neighborhood best; therefore, it has a more effective memory capability than the GA.
3) PSO is more efficient in maintaining the diversity of the swarm [68] (more similar to the ideal social interaction in a community), since all the particles use the information related to the most successful particle in order to improve themselves, whereas in GA, the worse solutions are discarded and only the good ones are saved; therefore, in GA the population evolves around a subset of the best individuals.

This paper provides a review of the PSO technique, the basic concepts and different structures and variants, as well as its applications to power system optimization problems. A brief introduction has been provided in this section on the existing optimization techniques that have been applied to power systems problems. The rest of this paper is arranged as follows. In Section II, the basic concepts of PSO are explained along with the original formulation of the algorithm in the real number space, as well as the discrete number space. The most common variants of the PSO algorithm are described in Section III. Section IV provides an extensive literature survey on the applications of PSO in power systems. Some potential applications of PSO in power systems, which are not yet explored in the literature, are briefly discussed in Section V. Finally, the concluding remarks appear in Section VI.

II. Particle Swarm Optimization (PSO): Concepts and Formulation

A. Basic Concepts

PSO is based on two fundamental disciplines: social science and computer science. In addition, PSO uses the swarm intelligence concept, which is the property of a system, whereby the collective behaviors of unsophisticated agents that are interacting locally with their environment create coherent global functional patterns. Therefore, the cornerstones of PSO can be described as follows.

1) Social Concepts [67]: It is known that "*human intelligence results from social interaction*." Evaluation, comparison, and imitation of others, as well as learning from experience allow humans to adapt to the environment and determine optimal patterns of behavior, attitudes, and suchlike. In addition, a second fundamental social concept indicates that "*culture and cognition are inseparable consequences of human sociality*." Culture is generated when individuals become more similar due to mutual social learning. The sweep of culture allows individuals to move towards more adaptive patterns of behavior.

2) Swarm Intelligence Principles [64]–[67], [69]: Swarm Intelligence can be described by considering five fundamental principles.

1) *Proximity Principle*: the population should be able to carry out simple space and time computations.
2) *Quality Principle*: the population should be able to respond to quality factors in the environment.

3) *Diverse Response Principle*: the population should not commit its activity along excessively narrow channels.
4) *Stability Principle*: the population should not change its mode of behavior every time the environment changes.
5) *Adaptability Principle*: the population should be able to change its behavior mode when it is worth the computational price.

In PSO, the term "particles" refers to population members which are mass-less and volume-less (or with an arbitrarily small mass or volume) and are subject to velocities and accelerations towards a better mode of behavior.

3) Computational Characteristics [67]: Swarm intelligence provides a useful paradigm for implementing adaptive systems. It is an extension of evolutionary computation and includes the softening parameterization of logical operators like AND, OR, and NOT. In particular, PSO is an extension, and a potentially important incarnation of *cellular automata* (CA). The particle swarm can be conceptualized as cells in CA, whose states change in many dimensions simultaneously. Both PSO and CA share the following computational attributes.

1) Individual particles (cells) are updated in parallel.
2) Each new value depends only on the previous value of the particle (cell) and its neighbors.
3) All updates are performed according to the same rules.

Other algorithms also exist that are based on swarm intelligence. The *ant colony optimization* (ACO) algorithm was introduced by Dorigo in 1992 [70]. It is a probabilistic technique for solving computational problems, which can be reduced to finding good paths through graphs. It is inspired by the behavior of ants in finding paths from the colony to the food. In the real world, ants initially wander randomly, and upon finding food, they return to their colony while laying down pheromone trails. If other ants find such a path, they are likely not to keep traveling at random, but rather follow the trail, returning and reinforcing it if they eventually find food [71]. However, the pheromone trail starts to evaporate over time, therefore reducing its attractive strength. The more time it takes for an ant to travel down the path and back again, the longer it takes for the pheromones to evaporate. A short path, by comparison, gets marched over faster, and thus the pheromone density remains high as it is laid on the path as fast as it can evaporate. Pheromone evaporation also has the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained. Thus, when one ant finds a short path from the colony to a food source (i.e., a good solution), other ants are more likely to follow that path, and positive feedback eventually leaves all the ants following a single path.

The idea of the ant colony algorithm is to mimic this behavior with "simulated ants" walking around the graph representing the problem to solve. ACO algorithms have an advantage over simulated annealing (SA) and GA approaches when the graph may change dynamically, since the ant colony algorithm can be run continuously and adapt to changes in real time [71], [72].

Stochastic diffusion search (SDS) is another method from the family of swarm intelligence, which was first introduced by Bishop in 1989 as a population-based, pattern-matching algorithm [73]. The agents perform cheap, partial evaluations of a hypothesis (a candidate solution to the search problem). They then share information about hypotheses (diffusion of information) through direct one-to-one communication. As a result of the diffusion mechanism, high-quality solutions can be identified from clusters of agents with the same hypothesis.

In addition to the above techniques, efforts have been made in the past few years to develop new models for swarm intelligence systems, such as a *honey bee colony* and *bacteria foraging* [74], [75]. The honey bee colony is considered as an intelligent system that is composed of a large number of simplified units (particles). Working together, the particles give the system some intelligent behavior. Recently, research has been conducted on using the honey bee model to solve optimization problems. This can be viewed as modeling the bee foraging, in which the amount of honey has to be maximized within a minimal time and smaller number of scouts [74].

Bacteria foraging emulates the social foraging behavior of bacteria by models that are based on the foraging principles theory [75]. In this case, foraging is considered as an optimization process in which a bacterium (particle) seeks to maximize the collected energy per unit foraging time. Bacteria foraging provides a link between the evolutionary computation in a social foraging environment and the distributed nongradient optimization algorithms that could be useful for global optimization over noisy conditions. This algorithm has been recently applied to power systems as well as adaptive control applications [76], [77].

B. PSO in Real Number Space

In the real number space, each individual possible solution can be modeled as a particle that moves through the problem hyperspace. The position of each particle is determined by the vector $x_i \in R^n$ and its movement by the velocity of the particle $v_i \in R^n$ [78], as shown in (1)

$$\vec{x}_i\,(t) = \vec{x}_i\,(t-1) + \vec{v}_i\,(t). \tag{1}$$

The information available for each individual is based on its own experience (the decisions that it has made so far and the success of each decision) and the knowledge of the performance of other individuals in its neighborhood. Since the relative importance of these two factors can vary from one decision to another, it is reasonable to apply random weights to each part, and therefore the velocity will be determined by

$$\vec{v}_i\,(t) = \vec{v}_i\,(t-1) + \varphi_1 \cdot \mathrm{rand}_1 \cdot \left(\vec{p}_i - \vec{x}_i\,(t-1)\right) \ldots + \varphi_2 \cdot \mathrm{rand}_2 \cdot \left(\vec{p}_g - \vec{x}_i\,(t-1)\right) \tag{2}$$

where φ_1, φ_2 are two positive numbers and rand_1, rand_2 are two random numbers with uniform distribution in the range of [0.0, 1.0].

The velocity update equation in (2) has three major components [79].

1) The first component is sometimes referred to as "inertia," "momentum," or "habit." It models the tendency of the particle to continue in the same direction it has been traveling.

This component can be scaled by a constant as in the modified versions of PSO.

2) The second component is a linear attraction towards the best position ever found by the given particle: p_i (whose corresponding fitness value is called the particle's best: p_{best}), scaled by a random weight $\varphi_1 \cdot \text{rand}_1$. This component is referred to as "memory," "self-knowledge," "nostalgia," or "remembrance."
3) The third component of the velocity update equation is a linear attraction towards the best position found by any particle: p_g (whose corresponding fitness value is called global best: g_{best}), scaled by another random weight $\varphi_2 \cdot \text{rand}_2$. This component is referred to as "cooperation," "social knowledge," "group knowledge," or "shared information."

According to the formulation above, the following procedure can be used for implementing the PSO algorithm [80].

1) Initialize the swarm by assigning a random position in the problem hyperspace to each particle.
2) Evaluate the fitness function for each particle.
3) For each individual particle, compare the particle's fitness value with its p_{best}. If the current value is better than the p_{best} value, then set this value as the p_{best} and the current particle's position, x_i, as p_i.
4) Identify the particle that has the best fitness value. The value of its fitness function is identified as g_{best} and its position as p_g.
5) Update the velocities and positions of all the particles using (1) and (2).
6) Repeat steps 2–5 until a stopping criterion is met (e.g., maximum number of iterations or a sufficiently good fitness value).

Richard and Ventura [81] proposed initializing the particles in a way that they are distributed as evenly as possible throughout the problem space. This ensures a broad coverage of the search space. They concluded that applying a starting configuration based on the centroidal Voronoi tessellations (CVTs) improves the performance of the PSO compared with the original random initialization [81]. As an alternative method, Campana *et al.* [82] proposed reformulating the standard iteration of PSO into a linear dynamic system. The system can then be investigated to determine the initial particles' positions such that the trajectories over the problem hyperspace are orthogonal, improving the exploration mode and convergence of the swarm.

1) Topology of the Particle Swarm: Particles have been studied in two general types of neighborhoods: 1) global best (gbest) and 2) local best (lbest) [67]. In the gbest neighborhood, the particles are attracted to the best solution found by any member of the swarm. This represents a fully connected network in which each particle has access to the information of all other members in the community [Fig. 1(a)]. However, in the case of using the local best approach, each particle has access to the information corresponding to its immediate neighbors, according to a certain swarm topology. The two most common topologies are the ring topology, in which each particle is connected with two neighbors [Fig. 1(b)], and the wheel topology (typical for highly centralized business organizations), in which the individuals are isolated from one another and all the information is communicated to a focal individual [Fig. 1(c)].

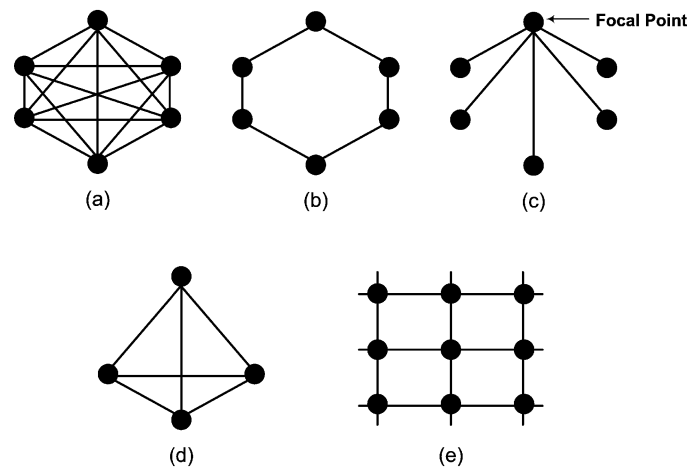


Fig. 1. Swarm topologies. (a) Global best. (b) Ring topology. (c) Wheel topology. (d) Pyramid topology. (e) Von Neumann topology.

Kennedy [83] suggests that the gbest version [Fig. 1(a)] converges fast but may be trapped in a local minimum, while the lbest network has more chances to find an optimal solution, although with slower convergence.

Kennedy and Mendes [84] have evaluated all topologies in Fig. 1, as well as the case of random neighbors. In their investigations with a total number of 20 particles, they found that the best performance occurred in a randomly generated neighborhood with an average size of five particles. The authors also suggested that the Von Neumann configuration may perform better than other topologies including the gbest version. Nevertheless, selecting the most efficient neighborhood structure, in general, depends on the type of problem. One structure may perform more effectively for certain types of problems, yet have a worse performance for other problems. The authors also proposed a fully informed particle swarm (FIPS), where each individual is influenced by the successes of all its neighbors, rather than just the best one and itself [85]. Therefore, instead of adding two terms to the velocity [attraction to the individual and global (or local) best] and dividing the acceleration constant between them, the FIPS distributes the weight of the acceleration constant φ equally across the entire neighborhood [85].

2) Parameter Selection for Particle Swarm: When implementing the particle swarm algorithm, several considerations must be taken into account to facilitate the convergence and prevent an "explosion" of the swarm. These considerations include limiting the maximum velocity, selecting acceleration constants, the constriction factor, or the inertia constant.

a) Selection of maximum velocity: At each iteration step, the algorithm proceeds by adjusting the distance (velocity) that each particle moves in every dimension of the problem hyperspace. The velocity of the particle is a stochastic variable and is, therefore, subject to creating an uncontrolled trajectory, making the particle follow wider cycles in the problem space [86], [87]. In order to damp these oscillations, upper and lower limits can be defined for the velocity v_i [67]

$$\begin{aligned} &\text{If} \quad v_{\text{id}} > v_{\max} \text{ then } v_{\text{id}} = v_{\max} \\ &\text{else if } v_{\text{id}} < -v_{\max} \text{ then } v_{\text{id}} = -v_{\max}. \end{aligned}$$

Most of the time, the value for $v_{\max}$ is selected empirically, according to the characteristics of the problem. It is important to note that if the value of this parameter is too high, then the particles may move erratically, going beyond a good solution; on the other hand, if $v_{\max}$ is too small, then the particle's movement is limited and the optimal solution may not be reached.

Research work performed by Fan and Shi [88] have shown that an appropriate dynamically changing $v_{\max}$ can improve the performance of the PSO algorithm. Additionally, to ensure uniform velocity throughout all dimensions, Abido [89], [90] has proposed a maximum velocity given by

$$v_{\max} = (x_{\max} - x_{\min})/N \tag{3}$$

where N is the number of intervals in the kth dimension selected by the user and $x_{\max}$, $x_{\min}$ are maximum and minimum values found so far by the particles.

b) Selection of acceleration constants: Acceleration constants φ_1 and φ_2 in (2) control the movement of each particle towards its individual and global best position, respectively. Small values limit the movement of the particles, while large numbers may cause the particles to diverge. Ozcan and Mohan conducted several experiments for the special case of a single particle in a one-dimensional problem space in order to examine the effect of a deterministic acceleration constant [67], [91]. In this particular case, the two acceleration constants are considered as a single acceleration constant $\varphi = \varphi_1 + \varphi_2$, since the individual and global best positions are the same. The authors concluded that by an increase in the value of the acceleration constant, the frequency of the oscillations around the optimal point increases. For smaller values of φ, the pattern of the trajectory is similar to a sinusoidal waveform; however, if the value is increased, the complex paths of interwoven cyclic trajectories appear. The trajectory goes to infinity for values of φ greater than 4.0.

The effect of considering a random value for acceleration constant helps to create an uneven cycling for the trajectory of the particle when it is searching around the optimal value. Since the acceleration parameter controls the strength of $(p-x)$ terms, a small value will lead to a weak effect; therefore, the particles will follow a wide path and they will be pulled back only after a large number of iterations. If the acceleration constant is too high then the steps will be limited by $v_{\max}$.

In general, the maximum value for this constant should be $\varphi = 4.0$, meaning $\varphi_1 + \varphi_2 = 4$. A good starting point has been proposed [67], [91] to be $\varphi_1 = \varphi_2 = 2$. It is important to note that φ_1 and φ_2 should not necessarily be equal since the "weights" for individual and group experience can vary according to the characteristics of the problem.

c) Selection of constriction factor or inertia constant: Empirical studies performed on PSO indicate that even when the maximum velocity and acceleration constants are correctly defined, the particles may still diverge, i.e., go to infinity; a phenomena known as "explosion" of the swarm. Two methods are proposed in the literature in order to control this "explosion:" *constriction factor* [92]–[94] and *inertia constant* [95], [96].

—Constriction Factor: The first method to control the "explosion" of the swarm was developed by Clerc and Kennedy [92]. It introduces a constriction coefficient which in the simplest case is called "Type 1" (χ) [67]. In general, when several particles are considered in a multidimensional problem space, Clerc's method leads to the following update rule [86]:

$$\begin{aligned} \vec{v}_i\,(t) &= \chi \cdot \Big\{ \vec{v}_i\,(t-1) + \varphi_1 \cdot \mathrm{rand}_1 \cdot \Big(\vec{p}_i - \vec{x}_i\,(t-1) \Big) \ldots \\ &\qquad + \varphi_2 \cdot \mathrm{rand}_2 \cdot \Big(\vec{p}_g - \vec{x}_i\,(t-1) \Big) \Big\} \\ x_i(t) &= x_i(t-1) + v_i(t) \end{aligned} \tag{4}$$

where

$$\chi = \frac{2}{\left| 2 - \varphi - \sqrt{\varphi^2 - 4\varphi} \right|}, \quad \varphi_1 + \varphi_2 = \varphi > 4.0. \tag{5}$$

Typically, when this method is used, φ is set to 4.1 and the constant χ is thus 0.729. This results in the previous velocity being multiplied by 0.729 and each of the two $(p - x)$ terms being multiplied by 1.49445 $(= 0.729 \times 2.05)$.

In general, the constriction factor improves the convergence of the particle over time by damping the oscillations once the particle is focused on the best point in an optimal region. The main disadvantage of this method is that the particles may follow wider cycles and may not converge when the individual best performance p_i is far from the neighborhood's best performance p_g (two different regions).

—Inertia Weight: The second method (proposed by Shi and Eberhart [95], [96]) suggests a new parameter which will only multiply the velocity at the previous time step, i.e., $v_i(t-l)$, instead of having one parameter multiplying the whole right-hand side as in (4). This parameter can be interpreted as an "inertia constant" (φ_{ic}), which results in the modified equation for the velocity of the particle [67]

$$\begin{aligned} v_i(t) &= \varphi_{ic} \cdot v_i(t-1) + \varphi_1 \cdot \mathrm{rand}_1 \cdot (p_i - x_i(t-1)) \ldots \\ &\qquad + \varphi_2 \cdot \mathrm{rand}_2 \cdot (p_g - x_i(t-1)) \\ x_i(t) &= x_i(t-1) + v_i(t). \end{aligned} \tag{6}$$

The inertia constant can be either implemented as a fixed value or can be dynamically changing [86], [89], [90], [93], [97]. Essentially, this parameter controls the exploration of the search space, therefore an initially higher value (typically 0.9) allows the particles to move freely in order to find the global optimum neighborhood fast. Once the optimal region is found, the value of the inertia weight can be decreased (usually to 0.4) in order to narrow the search, shifting from an exploratory mode to an exploitative mode. Commonly, a linearly decreasing inertia weight (first introduced by Shi and Eberhart [98], [99]) has produced good results in many applications; however, the main disadvantage of this method is that once the inertia weight is decreased, the swarm loses its ability to search new areas because it is not able to recover its exploration mode (which does not happen with Clerc's constriction coefficient [92]).

Recently, Chen and Li used stochastic approximation theory to analyze the dynamics of the PSO [100]. The authors proposed a decreasing coefficient that is reduced to zero as the number of iterations increases, and a stochastic velocity with fixed expectation to enhance the exploratory mode of the swarm. While the former facilitates the particles to spread around the problem hyperspace at the beginning of the search, the stochastic velocity

term provides additional exploration ability, thus helping the particles to escape from local minima.

C. Discrete PSO

The general concepts behind optimization techniques initially developed for problems defined over real-valued vector spaces, such as PSO, can also be applied to discrete-valued search spaces where either binary or integer variables have to be arranged into particles. A brief discussion about the adaptations that correspond to either case is presented in this section.

1) Binary PSO: For the particular case of binary PSO, each individual (particle) of the population has to take a binary decision, either $\mathrm{YES/TRUE} = 1$ or $\mathrm{NO/FALSE} = 0$. In that sense, according to the social approach of PSO, the probability of an individual to decide YES or NO can be modeled as [67], [101]

$$P(X_{\mathrm{id}} = 1) = f\left(X_{\mathrm{id}}(t-1), V_{\mathrm{id}}(t-1), p_{\mathrm{id}}, p_{gd}\right). \quad (7)$$

In this model, the probability that the ith individual chooses 1 for the dth bit in the string, i.e., $P(X_{\mathrm{id}} = 1)$, is a function of the previous state of that bit, i.e., $X_{\mathrm{id}}(t-1)$ and V_{id}, i.e., the measure of the individual's predisposition to choose 1 or 0.

This predisposition is derived based on individual and group performance. Therefore, the probability $P(X_{\mathrm{id}} = 1)$, implicitly depends on p_{id} and p_{gd}. The former is the best individual state found so far; it is 1 if the best individual success occurred when X_{id} was 1, and 0, otherwise. The latter corresponds to the neighborhood best; this parameter is 1 if the best of any member of the neighborhood occurred when X_{id} was 1, and 0, otherwise.

Mathematically, V_{id} determines a threshold in the probability function $P(X_{\mathrm{id}} = 1)$, and therefore should be bounded in the range of [0.0, 1.0]. This threshold can be modeled with the well-known sigmoidal function

$$s(V_{\mathrm{id}}) = \frac{1}{1 + \exp(-V_{\mathrm{id}})}. \quad (8)$$

Applying (8), the state of the dth position in the string for the ith individual at time t, $X_{\mathrm{id}}(t)$, can be expressed as [67], [101]

$$\begin{aligned} X_{\mathrm{id}}(t) &= 1, \quad \text{if} \quad \rho_{\mathrm{id}} < s(V_{\mathrm{id}}) \\ X_{\mathrm{id}}(t) &= 0, \quad \text{otherwise} \end{aligned}$$

where ρ_{id} is a random number with a uniform distribution in the range of [0.0, 1.0]. This procedure is repeatedly iterated over each dimension $(d : 1 \ldots n)$ and each individual $(i : 1 \ldots N)$, testing if the current value $X_{\mathrm{id}}(t)$ results in a better evaluation than p_{id}. In that case, the value of $X_{\mathrm{id}}(t)$ will be stored as the best individual state.

Equation (7) implies that the sociocognitive concepts of particle swarm are included in the function V_{id}, which states that the disposition of each individual towards success is adjusted according to its own experience as well as that of the community. Similar to the case of a real number space, and since the relative importance of individual and social factors may vary from one decision to another, it seems reasonable to consider random weights multiplying each part, as in (9) [67], [101]

$$\begin{aligned} V_{\mathrm{id}}(t) = V_{\mathrm{id}}(t-1) + \varphi_1 \cdot \mathrm{rand}_1 \cdot (p_{\mathrm{id}} - X_{\mathrm{id}}(t-1)) \ldots \\ + \varphi_2 \cdot \mathrm{rand}_2 \cdot (p_{gd} - X_{\mathrm{id}}(t-1)) \end{aligned} \quad (9)$$

where φ_1, φ_2 are two positive numbers and $rand_1$, $rand_2$ are two random numbers with uniform distribution in the range of [0.0, 1.0].

For all equations presented above, some considerations have to be made in order to adjust the limits of the parameters. As for the random weights φ_1, φ_2, the upper limits for the uniform distribution are sometimes set arbitrarily, but often in such a way that the two limits sum up to 4.0. In the case of V_{id}, a maximum limit must be determined in order to avoid the threshold being too close to 0.0 or 1.0. In practice, $V_{\max}$ is typically set to a value of 4.0, so that there is always at least a probability of $s(V_{\max}) = 0.018$ for any bit to change its state (8).

2) Integer PSO: In a more general case, when integer solutions (not necessarily 0 or 1) are needed, the optimal solution can be determined by rounding off the real optimum values to the nearest integer [102]. Equations (1) and (2), developed for a real number space, are used to determine the new position for each particle. Once $x_i(t) \in \Re^n$ is determined, its value in the dth dimension is rounded to the nearest integer value using the bracket function (10)

$$\begin{aligned} X_{\mathrm{id}}(t) &= [x_{\mathrm{id}}(t)], \quad d : 1 \rightarrow n \\ x_{\mathrm{id}}(t) &\in \Re \text{ and } X_{\mathrm{id}}(t) \in Z. \end{aligned} \quad (10)$$

The results presented by Laskari *et al.* [103] using integer PSO indicate that the performance of the method is not affected when the real values of the particles are truncated. Moreover, integer PSO has a high success rate in solving integer programming problems even when other methods, such as Branch and Bound fail [103].

III. PSO: Variants

This section describes different variants of the PSO algorithm. Some of these variants have been proposed to incorporate either the capabilities of other evolutionary computation techniques, such as hybrid versions of PSO or the adaptation of PSO parameters for a better performance (adaptive PSO). In other cases, the nature of the problem to be solved requires the PSO to work under complex environments as in the case of the multi-objective or constrained optimization problems or tracking dynamic systems. This section also presents discrete variants of PSO and other variations to the original formulation that can be included to improve its performance, such as dissipative PSO, which introduces negative entropy to prevent premature stagnation, or stretching and passive congregation techniques to prevent the particles from being trapped in local minima.

A. Hybrid PSO

A natural evolution of the particle swarm algorithm can be achieved by incorporating methods that have already been tested in other evolutionary computation techniques. Many authors have considered incorporating selection, mutation and crossover, as well as the differential evolution (DE), into the PSO algorithm. The main goal is to increase the diversity of the population by: 1) either preventing the particles to move too close to each other and collide [104], [105] or 2) to self-adapt parameters such as the constriction factor, acceleration constants [106], or inertia weight [107].

As a result, hybrid versions of PSO have been created and tested in different applications. The most common ones include hybrid of genetic algorithm and PSO (GA-PSO), evolutionary PSO (EPSO) and differential evolution PSO (DEPSO and C-PSO) which are discussed in this section.

1) Hybrid of Genetic Algorithm and PSO (GA-PSO): GA-PSO combines the advantages of swarm intelligence and a natural selection mechanism, such as GA, in order to increase the number of highly evaluated agents, while decreasing the number of lowly evaluated agents at each iteration step. Therefore, not only is it possible to successively change the current searching area by considering p_{best} and g_{best} values, but also to jump from one area to another by the selection mechanism, which results in accelerating the convergence speed of the whole algorithm.

The GA-PSO algorithm basically employs a major aspect of the classical GA approach, which is the capability of "breeding." However, some authors have also analyzed the inclusion of mutation or a simple replacement of the best fitted value, as a means of improvement to the standard PSO formulation [108], [109].

El-Dib *et al.* [108] considered the application of a reproduction system that modifies both the position and velocity vectors of randomly selected particles in order to further improve the potential of PSO to reach an optimum

$$\begin{aligned} child_1(x) &= p \cdot parent_1(x) + (1-p) \cdot parent_2(x) \\ child_1(v) &= (parent_1(v) + parent_2(v)) \\ &\quad \cdot \frac{|parent_1(v)|}{|parent_1(v) + parent_2(v)|} \\ child_2(x) &= p \cdot parent_2(x) + (1-p) \cdot parent_1(x) \\ child_2(v) &= (parent_1(v) + parent_2(v)) \\ &\quad \cdot \frac{|parent_2(v)|}{|parent_1(v) + parent_2(v)|} \end{aligned} \tag{11}$$

where $p \sim \mathrm{U}[0,1]$, $parent_{1,2}(x)$ represent the position vectors of randomly chosen particles, $parent_{1,2}(v)$ are the corresponding velocity vectors of each parent and $child_{1,2}(x)$, $child1,2(v)$ are the offspring of the breeding process.

Naka *et al.* [109] suggest replacing agent positions with low fitness values, with those with high fitness, according to a selection rate S_T, keeping the p_{best} information of the replaced agent so that a dependence on the past high evaluation position is accomplished (HPSO).

2) Hybrid of Evolutionary Programming and PSO (EPSO): Evolutionary PSO incorporates a selection procedure to the original PSO algorithm, as well as self-adapting properties for its parameters. Angeline [110] proposed adding the tournament selection method used in evolutionary programming (EP) for this purpose. In this approach, the update formulas remain the same as in the original PSO algorithm; however, the particles are selected as follows.

- The fitness value of each particle is compared with k other particles and scores a point for each particle with a worse fitness value. The population is sorted based on this score.
- The current positions and velocities of the best half of the swarm replace the positions and velocities of the worst half.
- The individual best of each particle of the swarm (best and worst half) remain unmodified. Therefore, at each iteration step, half of the individuals are moved to positions of the search space that are closer to the optimal solution than their previous positions while keeping their personal best points.

The difference between this method and the original particle swarm is that the exploitative search mechanism is emphasized. This should help the optimum to be found more consistently than the original particle swarm. In addition to the selection mechanism, Miranda and Fonseca [106], [111], [112] introduced self adaptation capabilities to the swarm by modifying the concept of a particle to include, not only the objective parameters, but also a set of strategic parameters (inertia and acceleration constants, simply called weights).

The general EPSO scheme can be summarized as follows [106], [111], [112].

- Replication: Each particle is replicated r times.
- Mutation: Each particle has its weights mutated.
- Reproduction: Each mutated particle generates an offspring according to the particle movement rule.
- Evaluation: Each offspring has a fitness value.
- Selection: Stochastic tournament is carried out in order to select the best particle which survives to the next generation.

The particle movement is defined as

$$\begin{aligned} v_i(t) &= w_{i1}^* \cdot v_i(t-1) + w_{i2}^* \cdot \mathrm{rand}_1 \cdot (p_i - x_i(t-1)) \ldots \\ &\quad + w_{i3}^* \cdot \mathrm{rand}_2 \cdot \left(p_g^* - x_i(t-1)\right) \\ x_i(t) &= x_i(t-1) + v_i(t) \end{aligned} \tag{12}$$

where

$$w_{ik}^* = w_{ik} + \tau \cdot \mathrm{rand} \tag{13}$$

and rand is a random number with normal distribution, i.e., N(0,1).

The global best is also mutated by

$$p_g^* = p_g + \tau' \cdot \mathrm{rand} \tag{14}$$

where τ and τ' are learning parameters that can be either fixed or dynamically changing as strategic parameters.

3) Hybrid of Differential Evolution and PSO (DEPSO and C-PSO): A differential evolution operator has been proposed to improve the performance of the PSO algorithm in two different ways: 1) it can be applied to the particle's best position to eliminate the particles falling into local minima (DEPSO) [113], [114], [115] or 2) it can be used to find the optimal parameters (inertia and acceleration constants) for the canonical PSO (composite PSO) [116].

a) DEPSO: The DEPSO method proposed by Zang and Xie [113] alternates the original PSO algorithm and the DE operator, i.e., (1) and (2) are performed at the odd iterations and (15) at the even iterations. The DE mutation operator is defined over the particle's best positions p_i with a trial point $T_i = p_i$ which for the dth dimension is derived as

$$\text{If}(\mathrm{rand} < \mathrm{CR} \text{ or } d = k) \text{ then } T_{\text{id}} = p_{gd} + \delta_{2d} \tag{15}$$

where k is a random integer value within $[1, n]$ which ensures the mutation in at least one dimension, CR is a crossover constant ($\mathrm{CR} \leq 1$) and δ_2 is the case of $\mathrm{N} = 2$ for the general difference vector

$$\delta_N = \frac{1}{N}\sum_{1}^{N}\Delta \tag{16}$$

where Δ is the difference between two elements randomly chosen in the p_{best} set.

If the fitness value of T_i is better than the one for p_i, then T_i will replace p_i. After the DE operator is applied to all the particles' individual best values, the g_{best} value is chosen among the p_{best} set providing the social learning capability, which might speed up the convergence.

b) Composite PSO (C-PSO): In most of the previously presented algorithms, the selection of the PSO parameters $(\varphi_{ic}, \varphi_1, \varphi_2)$ is made basically by trial and error. The use of algorithms such as GA, EP, or DE may help make this selection procedure more efficient. Composite PSO algorithm is a method that employs DE in order to solve the problem of parameter selection. The resulting algorithm is summarized next [116].

- Step 1) Initialize t to 1 and set the maximum number of iterations as T. Generate initial position of particles $(x(0))$, initial velocity $(v(0))$, and the initial PSO parameters $X = (\varphi_{ic}, \varphi_1, \varphi_2)$ randomly. The size of x, v, and X is equal to N_p, the size of the population, and t is the current iteration number.
- Step 2) For each X, calculate $v(t)$ and $x(t)$ as

$$\begin{aligned} v_i(t) &= \varphi_{ic}\cdot v_i(t-1) + \varphi_1\cdot \mathrm{rand}_1\cdot(p_i - x_i(t-1))\ldots \\ &\quad + \varphi_2\cdot \mathrm{rand}_2\cdot(p_g - x_i(t-1)) \\ x_i(t) &= x_i(t-1) + v_i(t). \end{aligned} \tag{17}$$

  Calculate the fitness function value for each particle.
- Apply mutation, crossover, and selection operators of the DE algorithm to X. Let X^* be the best individual produced by this process. Replace X by X^* and repeat the procedure until a terminal number of iterations of DE (selected *a priori*) is reached.
- The process continues from Step 2) until the stopping criterion (maximum number of iterations T) is met.

B. Adaptive PSO

Other authors have suggested other adjustments to the parameters of the PSO algorithm: adding a random component to the inertia weight [86], [117], [118], applying Fuzzy logic [119], [120], using a secondary PSO to find the optimal parameters of a primary PSO [121], Q-learning [122], or adaptive critics [123], [124].

Zhang *et al.* [125] have also considered the adjustment of the number of particles and the neighborhood size. The PSO algorithm is modified by adding an improvement index for the particles of the swarm

$$\delta(x_i) = \frac{f(x_i(t_0)) - f(x_i(t))}{f(x_i(t_0))} \tag{18}$$

where $f(x_i(t))$ is the fitness function value for particle i at iteration t.

An improvement threshold has to be defined as the limit for the minimum acceptable improvement. Then, the adaptive strategies are as follows [125].

1) Adjust the swarm size: If the particle has enough improvement but it is the worst particle in its neighborhood, then remove the particle. On the other hand, if the particle does not have enough improvement but it is the best particle in its neighborhood, then generate a new particle.
2) Adjust the inertia weight: The more a particle improves itself, the smaller the area this particle needs to explore. In contrast, if the particle has a deficient improvement then it is desirable to increase its search space. The adjustment of the inertia weight is done accordingly.
3) Adjust the neighborhood size: If the particle is the best in its neighborhood but it has not improved itself enough, then the particle needs more information and the size of the neighborhood has to be increased. If the particle has improved itself satisfactorily, then it does not need to ask many neighbors and its neighborhood size can be reduced.

In a similar fashion, Li [126] has proposed a species-based PSO (SPSO). According to this method, the swarm population is divided into species of subpopulations based on their similarity. Each species is grouped around a dominating particle called the species seed. At each iteration step, the species seeds are identified and adopted as neighborhood bests for the species groups. Over successive iterations, the adaptation of the species allows the algorithm to find multiple local optima, from which the global optimum can be identified.

C. PSO in Complex Environment

1) Multiobjective Particle Swarm Optimization (MOPSO): Multiobjective optimization problems consist of several objectives that need to be achieved simultaneously. One simple way to approach this problem is to aggregate the multiple objectives into one objective function considering weights that can be fixed or dynamically changing during the optimization process [127]. The main disadvantage of this approach is that it is not always possible to find the appropriate weighted function. Moreover, it is sometimes desired to consider the tradeoffs between the multiple objectives and, therefore, to find the multiple Pareto optimal solutions (Pareto front) [102].

Recently, several MOPSO algorithms have been developed based on the Pareto optimality concept. The main issue to be addressed is the selection of the cognitive and social leaders (p_{best} and l_{best}) such that they can provide an effective guidance to reach the most promising Pareto front region but at the same time maintain the population diversity.

For the selection procedure two typical approaches are suggested in the literature: selection based on quantitative standards and random selection. In the first case, the leader is determined by some procedure, without any randomness involved, such as the Pareto ranking scheme [128], the sigma method [129] or the dominated tree [130]. However, in the random approach, the selection for a candidate is stochastic and proportional to certain weights assigned to maintain the population diversity (crowding radius, crowding factor, niche count, etc.) [131]. For instance,

Ray and Liew [132] choose the particles that perform better to be the leaders (SOL) and the remaining particles tend to move towards a randomly selected leader from this leader group where the leader with fewer followers has the highest probability of being selected.

Coello and Lechuga [133] have also incorporated the Pareto dominance into the PSO algorithm. In this case, the nondominated solutions are stored in a secondary population and the primary population uses a randomly selected neighborhood best from this secondary population to update their velocities. The authors proposed an adaptive grid to generate well-distributed Pareto fronts and mutation operators to enhance the exploratory capabilities of the swarm [134].

Keeping the same two goals (obtaining a set of nondominated solutions as close as possible to the Pareto front and maintaining a well-distributed solution set along the Pareto front), Li [135] proposed sorting the entire population into various nondomination levels such that the individuals from better fronts can be selected. In this way, the selection process pushes towards the true Pareto front.

Other authors have developed different approaches such as combining canonical PSO with auto fitness sharing concepts [136], dynamic neighborhood PSO, or vector evaluated PSO, being the last two explained in the next sections.

a) Dynamic Neighborhood PSO (DN-PSO): The dynamic neighborhood method for solving multiobjective optimization problems has been developed by Hu and Eberhart [137], [138]. In this approach, the PSO algorithm is modified in order to locate the Pareto front.

- The multiple objectives are divided into two groups: F_1 and F_2. F_1 is defined as the neighborhood objective, while F_2 is defined as the optimization objective. The choices of F_1 and F_2 are arbitrary.
- At each iteration step, each particle defines its neighborhood by calculating the distance to all other particles and choosing the M closest neighbors. In this case, the distance is described as the difference between fitness values for the first group of objective functions.
- Once the neighborhood has been determined, the best local value is found among the neighbors in terms of the fitness value of the second group of objective functions.
- The global best updating strategy considers only the solutions that dominate the current p_{best} value.

An extended memory, for storing all Pareto optimal solutions in a current generation, has been introduced in order to reduce computational time and make the algorithm more efficient [138]. Bartz–Beielstein *et al.* [139] proposed having an archive of fixed size in which the decision of selection or deletion is taken according to the influence of each particle on the diversity of the Pareto front.

b) Vector Evaluated PSO (VEPSO): Parsopoulos and Vrahatis [102] proposed the vector evaluated particle swarm optimization (VEPSO) algorithm, which is based in the concept of the vector evaluated genetic algorithm (VEGA). In the VEPSO algorithm, two or more swarms are used in order to search the problem hyperspace. Each swarm is evaluated according to one of the objective functions and the information is exchanged between them. As a result the knowledge coming from other swarms is used to guide each particle's trajectory towards Pareto optimal points. The velocity update equation for an M-objective function problem can be formulated as [140]

$$\begin{aligned} v_i^{[j]}(t) = \chi^{[j]} \Big\{ & \varphi_{ic}^{[j]} \cdot v_i^{[j]}(t-1) \ldots + \varphi_1 \cdot \mathrm{rand}_1 \\ & \cdot \Big(p_i^{[j]} - x_i(t-1) \Big) \ldots + \varphi_2 \cdot \mathrm{rand}_2 \\ & \cdot \Big(p_g^{[s]} - x_i(t-1) \Big) \Big\} \end{aligned} \tag{19}$$

where

Index j	defines the swarm number $(j = 1, 2, \ldots, M)$;
Index i	corresponds to the particle number $(i = 1, 2, \ldots, N)$;
$\chi^{[j]}$	is the constriction factor of swarm j;
$\varphi_{ic}^{[j]}$	is the inertia weight of swarm j;
$p_i^{[j]}$	is the best position found by particle i in swarm j;
$p_g^{[s]}$	is the best position found for any particle in swarm s.

If the ring topology [Fig. 1(b)] is used, then

$$s = \begin{cases} M, & \text{if } j = 1 \\ j-1, & \text{if } j = 2, 3, \ldots, M \end{cases}. \tag{20}$$

The VEPSO algorithm also enables the swarms to be implemented in parallel computers that are connected in an Ethernet network [141]. In this case, the algorithm is called parallel VEPSO.

2) Constraint Handling in PSO: Real problems are often subject to different constraints that limit the search space to a certain feasible region. Two different approaches exist in the literature that handle constraints applied to a PSO algorithm. One approach is to include the constraints in the fitness function using penalty functions, while the second approach deals with the constraints and fitness separately.

The main advantage of the second approach is that there are no additional parameters introduced in the PSO algorithm and there is also no limit to the number or format of the constraints [131]. The PSO basic equations for velocity and position update remain unchanged. After the new positions are determined for all the particles, each solution is checked to determine if it belongs to the feasible space or not. If the feasibility conditions are not met, one of the following actions can be taken: the particle is reset to the previous position, or the particle is reset to its p_{best}, or the nonfeasible solution is kept, but the p_{best} is not updated (just feasible solutions are stored in the memory) [131], or the particle is rerandomized [142]. In addition, during the initialization process, all particles can be reinitialized until they find feasible solutions [131].

In his work with several popular benchmark functions, Hu [131] concluded that the PSO algorithm is efficient in handling constrained optimization problems by finding better solutions in less time. The PSO algorithm does not require domain knowledge or complex techniques, and no additional parameters need to be tuned. The limitations of the method appear in problems

with extremely small feasible spaces, where other constraint handling techniques may need to be developed.

3) Dynamic Tracking in PSO: The classical particle swarm algorithm has been proven to be very effective and computationally efficient in solving static optimization problems. However, this method might not be as efficient when applied to a dynamic system in which the optimal value may change repeatedly. An adaptive approach has been introduced to the original PSO algorithm in order to compensate for this problem. The concept of adaptation has been incorporated by either rerandomizing particles or dynamically changing the parameters of the PSO [87], [143].

Hu and Eberhart [144] introduced two methods to detect environmental changes: the "changed-gbest-value" and the "fixed-gbest-values." The former method suggests reevaluating the fitness function for g_{best} at each iteration step. If g_{best} refers to the same particle but its corresponding fitness function value is different, then it is assumed that the dynamics of the system has changed. Since this assumption may not be necessarily true for all dynamic systems, the second method is proposed, in which the locations of g_{best} and the second best particle are monitored. If none of them change in a certain number of iterations, the algorithm assumes that a possible optimum has been found. Various strategies are employed in both methods to deal with environmental changes by adapting the swarm. These include rerandomizing a certain number of particles (10%, 50%, or 100% of the population size), resetting certain particles, rerandomizing the g_{best} or a combination of the previous strategies [144], [145].

In a similar approach, Das and Venayagamoorthy [146], [147] have proposed a modification to the standard PSO called small population PSO (SPPSO). The algorithm uses a small population of particles (five or less) which is regenerated every N iterations; all particles are replaced except by the g_{best} particle in the swarm and the population p_{best} attributes are transmitted to the new generation to keep the memory characteristics algorithm. Under this scheme, the performance of the PSO is improved under dynamic conditions, making it more suitable for online applications, as well as hardware implementation.

D. Discrete PSO Variants

Further modifications to the Binary version of PSO have been developed to improve the performance of the algorithm in different applications. Mohan and Al-Kazemi have proposed the following variations [148].

- Direct approach, in which the classical PSO algorithm is applied and the solutions are converted into bit strings using a hard decision decoding process.
- Bias vector approach, in which the velocity's update is randomly selected from the three parts in the right-hand side of (2), using probabilities depending on the value of the fitness function.
- Mixed search approach, where the particles are divided into multiple groups and each of them can dynamically adopt a local or a global version of PSO.

The authors have also suggested unifying PSO with other evolutionary algorithms and with quantum theory. In the latter case, the use of a quantum bit (Q-bit) is proposed to probabilistically represent a linear superposition of states (binary solutions) in the search space [80], [149], [150]. Their results show that the proposed method is faster and more efficient compared to the classical binary PSO and other evolutionary algorithms, such as the GA.

A different approach was proposed by Cedeño and Agrafiotis [151], in which the original particle swarm algorithm is adapted to the discrete problem of feature selection by normalizing the value of each component of the particle's position vector at each run. In this way, the location of the particles can be viewed as the probabilities that are used in a roulette wheel to determine whether the entry x_{ij} takes 1 or 0, which determines whether the jth feature in the ith particle is selected or not in the next generation.

E. Other Variants of PSO

1) Gaussian PSO (GPSO): The classical PSO algorithm performs its search in the median between the global and local best. The way in which the search is performed, as well as the convergence of the swarm in the optimal area, depends on how the parameters such as acceleration and inertia constants are adjusted. In order to correct these perceived weaknesses, some authors have introduced Gaussian functions for guiding the movements of the particles [152]–[154]. In this approach, the inertia constant is no longer needed and the acceleration constant is replaced by random numbers with Gaussian distributions [153], [154].

Secrest and Lamont [152] proposed the following update formula:

$$\begin{aligned} |v(t)| &= \text{GRand}\left((1-C_1)\right. \\ &\qquad \left.\cdot\, |p_i(t-1)-p_g(t-1)|\right. \text{ when rand} > C_1 \\ |v(t)| &= \text{GRand}(C_2) \\ &\qquad \cdot\, |p_i(t-1)-p_g(t-1)| \text{ when rand} \le C_1 \\ v(t) &= |v(t)| \cdot \text{Rand}(\theta) \end{aligned} \tag{21}$$

where

$\lvert p_i(t-1)-p_g(t-1)\rvert$	distance between global and local best. If both points are the same, then it is set to one;
C_1	a constant between zero and one that determines the "trust" between the global and local best. The larger C_1 is, the more particles will be placed around the global best;
C_2	a constant between zero and one that establishes the point between the global $(p_g(\text{t}))$ and the local best $(p_i(\text{t}))$ that is a standard deviation from both;
$\text{GRand}(y)$	a zero-mean Gaussian random number with standard deviation of y;

rand	a random number between zero to one with uniform distribution;
$\mathrm{Rand}(\theta)$	a random vector with magnitude of one, and its angle is uniformly distributed from zero to 2π.

Considering this modification to the PSO algorithm, the area around the global and local best is predominately searched. As the global and local best get closer together, the standard deviation decreases and the area being searched converges.

Krohling [153], [154] has proposed a different method for updating the velocity at each iteration step, namely

$$v_i(t) = \mathrm{rand}_1 \cdot (p_i - x_i(t-1)) + \mathrm{rand}_2 \cdot (p_g - x_i(t-1)) \tag{22}$$

where rand_1 and rand_2 are positive random numbers generated according to the absolute value of the Gaussian probability distribution, i.e., $abs[\mathrm{N}(0,1)]$.

Considering the previous modifications in the velocity update formula, the coefficients of the two $(p-x)$ terms are automatically generated by using a Gaussian probability distribution. So, there is no need to specify any other parameters. Furthermore, the author claims that by using the Gaussian PSO, the maximum velocity $V_{\max}$ is no longer needed.

2) Dissipative PSO (DPSO): DPSO introduces negative entropy to stimulate the model in PSO, creating a dissipative structure that prevents premature stagnation [155], [156]. The negative entropy introduces an additional chaos in the velocity of the particles as follows:

$$\mathrm{If}(\mathrm{rand} < c_v)\ \mathrm{Then}\ v_{\mathrm{id}} = \mathrm{rand} \cdot V_{\max,d} \tag{23}$$

where rand and c_v are both random numbers between 0 and 1.

Analogously, the chaos for the location of the particles is represented by

$$\mathrm{If}(\mathrm{rand} < c_l)\ \mathrm{Then}\ x_{\mathrm{id}} = \mathrm{Rand}(l_d, u_d) \tag{24}$$

where rand is a random number between 0 and 1 and $\mathrm{Rand}(l_d, u_d)$ is another random number with predefined lower and upper limits [155].

The chaos introduces the negative entropy that keeps the system out of the equilibrium state. Then, the self organization of dissipative structures, along with the inherent nonlinear interactions in the swarm, lead to sustainable development from fluctuations [156].

3) PSO With Passive Congregation (PSOPC): Passive congregation, a mechanism that allows animals to aggregate into groups, has been proposed by He *et al.* [157] as a possible alternative to prevent the PSO algorithm from being trapped in local optima and to improve its accuracy and convergence speed. The inclusion of passive congregation modifies the original velocity update formula to

$$\begin{aligned} v_i(t) &= \varphi_{ic} \cdot v_i(t-1) + \varphi_1 \cdot r_1 \cdot (p_i - x_i(t-1)) \ldots \\ &+ \varphi_2 \cdot r_2 \cdot (p_g - x_i(t-1)) + \varphi_3 \cdot r_3 \cdot (X - x_i(t-1)) \end{aligned} \tag{25}$$

where r_1, r_2, and r_3 are random numbers between 0 and 1, φ_3 is the passive congregation coefficient, and X is a particle randomly selected from the swarm.

However, the work presented by He *et al.* in [157] does not include specifications for the value of the congregation coefficient, or how it affects the performance of the algorithm. These two aspects are important aspects for future research.

4) Stretching PSO (SPSO): The main issue in many global optimization techniques is the problem of convergence in the presence of local minima. Under these conditions, the solution may fall in the local minima when the search begins, and it may stagnate itself. Parsopoulos and Vrahatis [102] presented a modified PSO algorithm called "stretching" (SPSO) that is oriented towards solving the problem of finding all global minima.

In this algorithm, the so-called deflection and stretching techniques, as well as a repulsion technique are incorporated into the original PSO. The first two techniques apply the concept of transforming the objective function by incorporating the already found minimum points. The latter (repulsion technique) adds the ability to guarantee that all particles will not move toward the already found minima [102], [116]. Hence, the proposed algorithm can avoid the already found solutions and, therefore, have more chances to find the global optimal solution to the objective function.

The equations used are two-stage transformations. Assuming that a fitness function f is chosen for the problem, the first transformation stage transforms the original fitness function $f(x)$ into $G(x)$ with x representing any particle, which eliminates all the local minima that are located above $f(\overline{x})$, where $\overline{x}$ represents a detected local minimum

$$G(x) = f(x) + \gamma_1 \|x - \overline{x}\| \cdot (\mathrm{sgn}\,(f(x) - f(\overline{x})) + 1)\,. \tag{26}$$

The second stage stretches the neighborhood of $\overline{x}$ upwards, since it assigns higher function values to the points in the upward neighborhood

$$H(x) = G(x) + \gamma_2 \frac{\mathrm{sgn}\,(f(x) - f(\overline{x})) + 1}{\tanh\,(\mu\,(G(x) - G(\overline{x})))}. \tag{27}$$

In (26) and (27), γ_1, γ_2. and μ are arbitrarily chosen positive constants and $\mathrm{sgn}(y)$ is the triple valued sign function

$$\mathrm{sgn}(y) = \begin{cases} 1, & \text{if } y > 0 \\ 0, & \text{if } y = 0\,. \\ -1, & \text{if } y < 0 \end{cases} \tag{28}$$

None of the stages alter the local minima located below $\overline{x}$; therefore, the location of the global minimum is left unchanged [102].

5) Cooperative PSO (CPSO): The cooperative PSO (CPSO), as a variant of the original PSO algorithm, is presented by Van den Bergh and Engelbrecht [94]. CPSO employs cooperative behavior in order to significantly improve the performance of the original PSO algorithm. It uses multiple swarms to optimize different components of the solution vector cooperatively.

Following the same approach as Potter's cooperative coevolutionary genetic algorithm (CCGA), in CPSO, the search space is explicitly partitioned by splitting the solution vectors into smaller vectors. Two new algorithms are proposed $\mathrm{CPSO} - S_k$ and $\mathrm{CPSO} - H_k$.

In the CPSO-S algorithm, a swarm with n-dimensional vectors is partitioned into n-swarms of one-dimensional vectors, with each swarm attempting to optimize a single component of

TABLE I
APPLICATION OF PSO TECHNIQUE TO POWER SYSTEMS BY TECHNICAL AREAS

Area Name	PSO types	References
Reactive power and voltage control	Conventional PSO, Integer PSO, Adaptive PSO	[125], [163]-[168]
Economic Dispatch	Conventional PSO, Evolutionary Programming PSO (EPSO)	[169]-[179]
Power System Reliability and Security	Conventional PSO, Binary PSO	[181]-[188]
Generation Expansion Problem	Conventional PSO, Stretching PSO (SPSO), Composite PSO (C-PSO)	[116], [190]-[192]
State Estimation	Conventional PSO, Hybrid PSO (GA-PSO)	[109], [193]
Load Flow and Optimal Power Flow	Conventional PSO, Hybrid PSO (GA-PSO), Vector Evaluated PSO (VEPSO), PSO with Passive Congregation (PSOPC), Dissipative PSO (DPSO)	[90], [108], [141], [155], [157]
Power System Identification and Control		
- Controller Tuning	Conventional PSO	[147], [195]-[204]
- System Identification and Intelligent Control	Conventional PSO, Hybrid PSO (GA-PSO)	[117], [118], [205]-[216]
Electric Machinery	Conventional PSO	[217], [218]
Capacitor Placement	Conventional PSO, Integer PSO	[97], [220]-[222]
Generator Maintenance Scheduling	Conventional PSO, Evolutionary Programming PSO (EPSO)	[223]-[224]
Short-Term Load Forecasting	Conventional PSO	[225]
Generator Contributions to Transmission System	Vector Evaluated PSO (VEPSO)	[226]

the solution vector. A credit assignment mechanism is designed to evaluate each particle in each swarm; for instance the original fitness function for the jth swarm can be evaluated, keeping all other $n-1$ components constant. The advantage of the CPSO-S approach is that only one component is modified at a time, therefore, many combinations are formed using different members from different swarms, yielding the desired fine-grained search and a significant increase in the solution diversity.

The algorithm called $\mathrm{CPSO}-S_k$ is a modification of the previous method in which the position vector is divided in k parts instead of n.

On the other hand, given that the PSO has the ability to escape from pseudominimizers, and the $\mathrm{CPSO}-S_k$ algorithm has faster convergence on certain functions, the $\mathrm{CPSO}-H_k$ combines these two techniques by executing one iteration of $\mathrm{CPSO}-S_k$ followed by one iteration of the standard PSO algorithm.

Baskar and Suganthan [158] have proposed a cooperative scheme, referred to as concurrent PSO (CONPSO), where the problem hyperspace is implicitly partitioned by having two swarms searching concurrently for a solution with frequent message passing of information (g_{best}).

Recently, a new hierarchal cooperative particle swarm optimizer was proposed by combining the implicit and explicit space decomposition techniques adopted in CPSO-S and CONPSO [159]. The combination is achieved by having two swarms concurrently searching for a solution, while each one employs the CPSO-S technique. The results provided in [159] show that the proposed approach outperforms the CONPSO, the CPSO-S, and the CPSO-H for four selected benchmark functions, namely, the Rosenbrock function (unimodal), the Griewank function (multimodal), the Ackley function (multimodal), and the Rastrigin function (multimodal) [159].

6) Comprehensive Learning PSO (CLPSO): In this new strategy, the conventional equation for the velocity update is modified to [160]

$$v_i^d(t) = \varphi_{ic} \cdot v_i^d(t-1) + \varphi \cdot \mathrm{rand}_1 \cdot \left(pbest_{fi(d)}^d - x_i^d(t-1)\right) \tag{29}$$

where d correspond to the dimension index $(d : 1 \rightarrow D)$ and $f_i(d)$ defines which particles' p_{best} the particle i should follow.

For each dimension of particle i, a random number is generated; if this number is greater than a certain value Pc_i (where Pc is called the learning probability), then the particle will follow its own p_{best}, otherwise it will learn from another particle's p_{best}. In the latter case, a tournament selection is applied to determine which particle's p_{best} will be used.

1) Two random particles are selected from the swarm

$$f1_i^d = \lceil \mathrm{rand}_1 \cdot ps \rceil, \quad f2_i^d = \lceil \mathrm{rand}_2 \cdot ps \rceil \tag{30}$$

where ps is the population size.

2) Their p_{best}values are compared and the best one is selected.

3) The winner particle is used as an exemplar to learn from.

Additionally, to ensure that the particles learns from good exemplars and to minimize the time wasted following poor directions, the particles are allowed to learn until a refreshing gap m, defined as a certain number of iterations, is reached. After that the values of f_i are reassigned for all particles in the swarm.

In the CLPSO algorithm, the parameters, φ, Pc, and m have to be tuned. In the case of the learning probability Pc, Liang *et al.* [160] have proposed to use a different value for each particle to given them different levels of exploration and exploitation ability. In this scheme, the advantages of this learning strategy are that all the particles are potential leaders; therefore, the chances of getting trapped in local minima are reduced by the cooperative behavior of the swarm. In addition, the particles use different exemplars for each dimension, which are renewed after some iterations (refreshing gap), giving more diversity in the searching process.

IV. PSO: APPLICATIONS TO POWER SYSTEMS

This section presents an overview of the applications of the PSO technique to power systems problems. Table I summarizes the applications where PSO has been applied for solving the optimization problem, along with the type of PSO used and the major publications associated with the application.

The topics addressed in this section include those presented by AlRashidi and El-Hawari [161], plus some new areas under development. In addition, technical details are offered for each

application to allow the reader to perform similar experiments and solve problems of the same characteristics.

A. Reactive Power and Voltage Control

In general, an electric power network is prone to sudden changes to its configuration, as the lines and loads are switched on and off. In such a dynamic system, keeping the voltage within an allowable range for the consumers is one of the most important operating tasks of the utilities. In order to achieve this, the power utility operators can control the synchronous generators, transformer tap settings, FACTS devices and shunt reactors in a way that they generate the required amount of reactive power to maintain the bus voltages at the desired level. An online control strategy to achieve this is referred to as reactive power and voltage control, or volt/var control (VVC) in short.

Essentially, any VVC strategy needs to ensure that voltage security is met, such that the system conditions do not migrate towards a voltage collapse. Several conventional techniques exist in the literature for evaluating voltage contingency analysis [162]. In general, VVC can be formulated as follows:

$$\text{Minimize} \sum \text{Loss} \tag{31}$$

subject to the following constraints.

- Voltage at each node must lie within its permissible range.
- Power flow of each branch must be lower than the maximum allowable.
- Transformer tap positions must be within the available range of steps.
- Generator reactive powers must lie within their permissible range.

In addition to the above constraints, VVC can be formulated to keep the voltage security of the power system [163].

The conventional methods using analytical optimization techniques tend to get more complicated as the dimensions of the system increase. PSO is an effective optimization strategy which handles mixed-integer nonlinear optimization problems (MINLP) with ease and can be an alternative solution to the above problem [163]–[167]. It is specifically helpful when sources of intermittent generation present complications to the VVC problem due to additional physical and economic constraints [164].

It has been shown that PSO-based VVC is faster than the conventional enumeration method [163], [166] converges to a solution with lower losses [164] or does both [165]. Mantawy and Al-Ghamdi compared the efficiency of PSO with those of GAs, improved linear and NLP in a multimachine power system, and showed that PSO provided the best solution in terms of the total active power losses [167]. Zhang and Liu successfully applied the technique to part of the power network in China with 151 buses and 22 transmission lines [166]. Zhang *et al.* [125] also proposed an adaptive PSO algorithm in which the size of the swarm population and the neighborhood size can be adaptively adjusted according to the rate of improvement in the particles. The technique outperforms the conventional PSO in both the final solution and the simulation runtime.

An objective function should be defined for the PSO. This function can be the total active power losses in the network [125], [163], [165]–[167] or the total reactive power losses [164]. The conventional PSO should be expanded in order to be able to incorporate both continuous and discrete variables in a single particle. The continuous variables include the generator AVR operating values, while the discrete values include the on-load tap changers (OLTC) of the transformers, the number of the reactive power compensation equipment and the number of the wind turbine generators [163], [164]. A particle can be defined as

$$x = [V_{G1}, \ldots, V_{GN_G}, T_{T1}, \ldots, T_{TN_T}, Q_{C1}, \ldots, Q_{CN_C}]^T \tag{32}$$

where V_G, T_T, and Q_C are the generator terminal voltage (AVR setting), transformer tap setting and the shunt compensator, respectively, defined for every single component [166].

Also, Krami *et al.* [168] presented a new method based on Pareto dominance together with PSO for the reactive power planning problem. Two disagreeing objective functions related to the cost and the active power losses are optimized. The approach considers the maximum allowable voltage deviation at each bus and assumes that all load busses are possible candidates for reactive compensation. The experimental results demonstrate the efficiency of the proposed approach for reactive power planning problem [168].

B. Economic Dispatch (ED)

Economic dispatch (ED) is a milestone problem in power systems. The objective here is to find an optimal operating condition for the generation units in order to minimize their operational costs whilst constraints are met. When the load operation condition of the system is assumed to be constant, the ED is called static ED. However, the problem is referred to as dynamic economic dispatch (DED) when an online approach is used in order to meet the load demand over a given operational period. It is an extension to the conventional ED in the sense that it takes the nonlinear characteristics of generation units into account. The nonlinear characteristics of the generation units include discontinuous prohibited zones, ramp rate limits, and nonsmooth or convex cost functions. In general, DED is solved for short periods of time in which the system load conditions can be assumed constant, where it is solved to meet the power balance constraints and generation limits in each short period of time. As a more accurate tool for the ED problem, DED is fundamental for real-time control of power systems and is considered a difficult dynamic optimization problem.

The ED or the DED problems can be formulated with or without smooth cost functions. In the case of ED with a smooth cost function, the operating cost of each generator is represented by a quadratic function as given by

$$\begin{aligned} \text{Cost} = C &= \sum_{i \in I} F_i(P_i) \\ F_i(P_i) &= a_i + b_i P_i + c_i P_i^2 \end{aligned} \tag{33}$$

where

C	is the total generation cost;
F_i	is the cost function of generation i;
a_i, b_i, c_i	are the cost coefficients of generator i;
P_i	is the electrical output of generator i;
I	is the group of all generators.

The total generation cost optimization is subject to power balance and operational limits constraints.

In the case of the ED problem with a nonsmooth cost function, the total generation cost function has non differentiable points. This represents a more realistic problem since real issues are considered, e.g., valve-point effects and change of fuels.

For the valve-point effect problem, the total generation cost is in most of the cases described as the addition of quadratic and sinusoidal functions, which makes the problem nonconvex. The operating cost of each generation unit for this case is given by

$$F_i(P_i) = a_i + b_i P_i + c_i P_i^2 + |e_i \sin(f_i(P_{i\min} - P_i))| \quad (34)$$

where e_i, f_i are the coefficients regarding generator i valve-point effects.

For a multiple fuels problem, the total generation cost is expressed as a piecewise quadratic function. The operating cost of each generation unit for this case is shown in (35)

$$F_i(P_i) = \begin{cases} a_{i1} + b_{i1}P_i + c_{i1}P_i^2, & \text{if } P_{i\min} \le P_i \le P_{i1} \\ a_{i2} + b_{i2}P_i + c_{i2}P_i^2, & \text{if } P_{i1} \le P_i \le P_{i2} \\ \vdots & \vdots \quad \vdots \\ a_{\text{in}} + b_{\text{in}}P_i + c_{in}P_i^2, & \text{if } P_{i(n-1)} \le P_i \le P_{\text{in}} \end{cases} \quad (35)$$

where a_{ik}, b_{ik}, and c_{ik} are the cost coefficients of generator i for the kth power level.

In order to solve the ED problem, many methods have been proposed in the literature, which include the mathematical approach (MA), DP, EP, SA, tabu search (TS), artificial neural networks (ANNs), and GA [169]. A recent summary on the application of PSO to ED problems [170] indicates that the PSO-based application outperforms most of these heuristic and mathematical algorithms.

Aruldoss *et al.* [171] reported the application of PSO to reserve constrained dynamic dispatch of units with valve-point effects problem. A hybrid approach, by integrating the PSO with the sequential quadratic programming (SQP), is presented and compared with EP-SQP approach. For all test cases, the quality of the solution, convergence, reliability, constraints handling, and computational effort, the PSO-SQP is shown to be superior. Also, Park *et al.* [169] have applied PSO to ED problem with nonsmooth cost functions. Their new approach provided high probability solution for a three generator test system and quasi-optimums for a 40 generator test system. The proposed methodology is shown to be superior compared with conventional numeric methods, ANN, and EP. Similar results have been reported in [172]. In addition, Gaing [173] has applied PSO to solve the ED problem considering the generator constraints. For different test systems with 6, 15, and 40 generators, the PSO again gave better results regarding solution quality, convergence, and computational effort when compared with the GA method.

Several other papers have been published regarding the application of PSO to the ED problem: Sinha and Purkayastha [174] proposed a hybrid method that integrates the main features of PSO with EP, Pancholi and Swarup [175] applied PSO for security constrained ED, Zhao *et al.* [176] solved the DED problem applying PSO technique in bid-based environment in a competitive electricity market, El-Gallad *et al.* [177] proposed the application of PSO to solve the ED problem considering prohibited operating zones, and Kumar *et al.* [178] applied PSO to emission and economic dispatch problem. In general, a particle can be defined as

$$x_j = [P_{1j} \;\; P_{2j} \;\; \ldots \;\; P_{ij} \;\; \ldots \;\; P_{nj}]^T \quad (36)$$

where P_{ij} is the electrical output of generator i for particle j.

Stochastic and deterministic models for the power dispatch are also used and optimized by an improved PSO algorithm. The Pareto-dominance concept is used to tackle the multiobjective optimization problem which is designed to handle the economic power dispatch, while considering the environmental impact [179]. Results showed that PSO produces more economical solutions as compared with weighted aggregation and evolutionary optimization [179].

C. Power System Reliability and Security

Distribution system reliability is defined in terms of adequacy of supply, which is related to the existence of sufficient facilities in the system to satisfy the load demands within the system constraints, and security, which is the ability of the system to overcome the disturbances occurring inside it [180]. Analyzing the reliability indices of a power system is historically done using contingency analysis, which in a large network, considering multiple failures, can be extremely complicated and time consuming. Robinson successfully applied binary PSO for identifying the set of network elements which if disrupted, would possibly lead to a cascading series of events resulting in a widespread damage to the network [181].

Feeder reconfiguration is a technique used by some researchers for improving the quality/price of service provided to the customers, while maintaining/increasing the reliability of the network. The problem is formulated as a nonlinear optimization problem subject to the security constraints of the distribution network such as not exceeding the feeder capacities or busbar voltage magnitudes, while keeping the radial network structure. Several authors have successfully reported the application of binary PSO in feeder reconfiguration in the distribution system [182]–[184]. In all cases, the particle is defined as an array consisting of binary numbers. Each binary number corresponds to a switch in the distribution network that can be open or closed. By changing the status of the switches, the configuration of the distribution network can be modified.

Binary PSO has been used in order to reconfigure the distribution network for aggregating the loads [182], [183]. Chang and Lu [182] used an objective function defined as the sum of the feeder load factors to be maximized, which results in the feeder

load being at or near peaks most of the time. This will bring savings for customers and reduce generation costs for producers, which in turn helps increase the customer purchasing power in the market. Shen and Lu [183] defined the objective function as maximizing the matching index between the customer power quality requirements with the feeder service quality, so that more reliable feeders can supply customers with special power quality requirements. Jin *et al.* [184] applied PSO for feeder reconfiguration in order to balance the load between the feeders. This helps to reduce the losses, as well as to increase the system reliability margin. The sum of the load balancing indices is defined as the objective function that needs to be minimized subject to the network security constraints.

As another technique used for increasing the reliability of the power system, Kurutach and Tuppadung [185], [186] applied PSO in order to find the most appropriate positions to place sectionalized devices in the distribution lines. This directly affects the outage duration of the feeders and if done properly, can improve the reliability of the network. The objective function was defined as minimizing the annual feeder interruption cost, while the particles were considered to be the location of the switch in the network. It is shown in [185] that PSO convergences within a few iterations, but due to its nonlinearity and nondifferentiability, the problem cannot be easily formulated and solved using traditional linear or NLP methods.

Research has also been done on the applications of PSO in identifying points on the security border of the power system, thereby identifying a vulnerability margin metric for the operating point [187]. In their proposed method, Kassabalidis *et al.* [187] showed that PSO can be used to evaluate the security index of the power system in real time. The goal is to identify as many points on the security border as possible and this is achieved by defining a separate swarm for each point to be placed on the border, where each operating condition is defined as a particle.

Recently, PSO has also been used to tackle the *under voltage load shedding* (UVLS) problem. The concept of dynamic security-constrained optimal power flow is employed to develop the model to be optimized. Results from PSO and GA are compared, where PSO produces better solutions compared to GA [188].

D. Generation Expansion Problem (GEP)

The deregulation of electrical markets has created competition between electric utilities. This competition has caused the generation expansion problem (GEP) to become an increasingly important issue to be considered by investors when making economical decisions. The GEP consists in determining what, when, where, and how to install new generation units in order to meet the power system requirements while constraints regarding load demand, reliability, operating conditions, power quality, and security are met. GEP maximizes profits, as well as minimizes the investment risks over the long-term planning horizon. The GEP can be mathematically formulated as a high dimensional, nonlinear, nonconvex, mix-integer and highly constrained optimization problem with the least cost of the investment as the objective function. The complexity of the problem rapidly increases if many practical constraints are taken into account. Thus, mathematical formulations have to be simplified in order to get solutions.

Methods to solve the GEP can be generally categorized into two types: traditional mathematical programming methods and methods based on heuristic techniques. The traditional mathematical methods include DP, mix-integer programming (MIP), branch and bound, Benders' decomposition, network flow methods, and others. These methods are usually strict in mathematics, strong in numerical manipulations, weak in handling qualitative constraints, and slower in computational performance [189]. The heuristic techniques mainly include the application of Artificial Intelligence (AI) approaches, such as GAs, DE, EP, evolutionary strategy (ES), ACO, PSO, TS, SA, and a hybrid approach [190]. The main advantage of the heuristic techniques is their flexibility for handling numerous qualitative constraints that are common in the GEP in the deregulated power industry.

The application of PSO for solving GEP has been reported in the literature [190]–[192]. The application and comparison of metaheuristic techniques to solve the GEP has been presented in [190], where PSO is compared with eight other metaheuristic techniques in terms of the success rate and execution time. The performance of the metaheuristic techniques is improved by the application of a virtual mapping procedure, intelligent initial population generation, and penalty factor approach. Optimal and near-optimal solutions are reached within a reasonable amount of time. In addition, Kannan *et al.* [116] have presented the application of the PSO technique and its variants to the GEP. In this case, three different test cases are used for comparison with DP. The comparisons are based on computation time, success rate, and error limit. The virtual mapping procedure is also used here. For all test cases, the PSO technique and its variants produce better results in much less time compared with DP. Among the PSO variants, it was observed that the constriction factor approach performed comparatively better. Additionally, Sensarna *et al.* [191] have presented the application of PSO to GEP with a parametric approach for planning and transmission expansion including topology optimization, capital expenses, power losses, network security, and revenue. Likewise, Slochanal *et al.* [192] have presented the application of PSO to the generation expansion planning problem in a competitive environment.

E. State Estimation

Online state estimation is a crucial factor in distribution control centers, especially with the introduction of distributed generators (DGs) to the power system. Power utilities need to have an online and accurate estimate of the loads and DG outputs, while only limited measurements are normally available throughout the network. State estimation can be formulated as a weighted least mean square problem

$$\text{Minimize} \quad J(x) = \sum_i w_i.(z_i - f_i(x_i))^2 \tag{37}$$

where x_i, z_i, and w_i are the state variable i, measurement value (normally voltages and currents) for measurement variable i and its related weighting factor respectively. Also, f_i is the power flow equation that relates the state variable x_i to the measurement variable z_i.

Assuming that the magnitudes of the voltages and currents are available at the sending end and the remote terminal units (RTU) across the network, conventional techniques using statistical or sensitivity analysis methods can be applied to perform the above minimization problem [109]. These techniques assume that the objective function is differentiable and continuous. However, with nonlinear devices present in the network such as var compensators, DGs and transformers with on load tap changers, the system equations and, therefore, the objective function are nonlinear, discontinuous, and not differentiable. With its strong nonlinear optimization capabilities PSO can be an efficient alternative solution to this problem. Hybrid PSO (GA-PSO) is proposed in the literature that is able to tackle the above optimization problem [109], [193]. The same objective function is defined for PSO, whose particles consist of the states of the power system. Naka *et al.* [109] showed that hybrid PSO is more accurate in estimating the states of the system than the conventional PSO.

F. Load Flow and Optimal Power Flow

Load flow study is the most basic tool for power system analysis and design; it is used for power system operation and planning, economic scheduling, as well as transient stability, voltage stability, and contingency studies. The load flow equations include power balance equations, for both active and reactive powers at each bus. Therefore, the load flow problem can be formulated as an optimization problem, where the objective is to find the voltage magnitudes and angles that minimize the difference between the input and the output power at each bus [108]. Usually, the load flow problem is solved by conventional numerical methods like Newton–Raphson (NR) or Gauss–Seidel (GS), however, these techniques may fail under certain power system conditions (heavy loaded system) or due to an ill-conditioned Jacobian matrix [108]. For this reason, an approach considering an evolutionary computational technique such as PSO may be interesting to analyze.

The problem formulation can be stated as follows:

$$\text{Minimize} \quad f(V,\delta) = \sum_i \left(F_{pi}^2 + F_{qi}^2\right) \tag{38}$$

where F_{pi} and F_{qi} are defined according to the nonlinear power flow equations.

The optimization problem is subject to the following:

- scheduled value of slack bus voltage;
- scheduled values of PV bus voltages and powers.

In this case, a particle can be defined as

$$x = [V_1, \ldots, V_n, \delta_1, \ldots, \delta_n] \tag{39}$$

where (V_i, δ_i) correspond to the magnitude and phase of the voltage at bus i.

In order to improve the performance of PSO, EL-Dib *et al.* [108] have proposed the use of a decreasing constriction factor, as well as establishing mutation in the particles as in GAs. It was reported that by using these techniques, the PSO is able to solve the load flow problem with the desirable tolerance and find the solution in high loading cases where NR fails to do so. A similar approach can be used to find the contributions of each generator in the transmission system. The problem can be formulated as a multiobjective optimization problem that takes into account nonlinear characteristics of the power systems such as generator prohibited operating zones and line thermal limits. Particularly, the contributions of generators to the real-power flows in transmission lines are modeled as positions of agents in the swarm.

Vlachogiannis and Lee [141] consider the parallel VEPSO to solve this problem in four different test systems. The feasibility of the proposed methodology is demonstrated and compared with other analytical methods. Results show that the parallel VEPSO algorithm is slower than the analytical method, although it is capable of obtaining high accuracy results within a reasonable computing time.

In the case of optimal power flow, the purpose is to find an optimal solution for a certain power system objective, such as the overall fuel cost or total losses, by adjusting the system control variables, i.e., real and reactive powers, voltages at different buses, etc. [90]. Traditionally, conventional optimization techniques are used to solve this nonlinear problem, including linear and nonlinear programming (LP and NLP), interior point method, quadratic programming (QM), and sequential unconstrained minimization techniques. All of these methods give satisfactory results, however they present some disadvantages in terms of the computational effort or the convergence speed. As an answer to these drawbacks, stochastic optimization techniques, including PSO, have been applied to solve the optimal power flow problem.

The problem can be formulated as [90]

$$\text{Minimize } f(x,u) \tag{40}$$

where $f(x,u)$ is the desired power system objective function, x is the vector of dependent variables such as load bus voltages or apparent power flow and u is the vector of control variables such as active and reactive powers, voltage magnitudes at generator buses and suchlike.

The problem is subject to the following constraints.

- Load flow equations.
- Generation constraints: lower and upper limits for each generator.
- Transformers constraints: minimum and maximum tap setting.
- Shunt VAR constraints.
- Upper and lower limits for control variables.
- Security constraints.

The previous formulation can be modified to convert it into an unconstrained optimization problem where the constraints are included in the objective function using penalty functions [90]. In this case, each particle has to be defined according to the control variables such as voltage magnitudes at different generators V_{Gi}, transformer tap settings T_i, and reactive compensation Q_i

$$x = [V_{G1}, \ldots, V_{Gn}, T_1, \ldots, T_m, Q_1, \ldots, Q_k] \tag{41}$$

For this particular problem, the performances of PSO and PSO with passive congregation have been compared with classical optimization methods and other evolutionary computational techniques like GAs [90], [157]. The reported results on

small size power systems (IEEE 30 bus system) indicate that, with the appropriate definition of the parameters (decreased inertia constant, adjusted velocity limits), PSO and its variants perform well, finding better solutions with less computational effort. However, tests performed on large power systems (IEEE one-area RTS96 or IEEE 118 bus system), showed that the quality of the PSO solution deteriorates as the number of control variables increases [155], therefore, the capabilities of PSO to provide satisfactory solutions for large-scale power systems has yet to be demonstrated.

G. Power System Identification and Control

Power system stability is defined as "the ability of an electric power system, for a given initial operating condition, to regain a state of operating equilibrium after being subjected to a physical disturbance, with most system variables bounded so that practically the entire system remains intact" [194]. Obtaining and maintaining stability conditions in any large power system, even in the presence of disturbances, has always been a difficult task to achieve, due to the high complexity of the network and the increasing incorporation of nonlinear switching devices. Several control technologies have been used in the past to overcome this problem, including classical tools such as PI controllers and modern approaches like neurocontrollers or fuzzy logic-based controllers. Currently, the challenge lies in incorporating evolutionary computational techniques in order to improve the performance of the existing and proposed controllers, as well as to develop more accurate system identifiers.

1) Controller Tuning: Specifically, PSO has been proposed in the literature as an efficient algorithm to tune the gains in classical PID controllers, as well as in nonlinear controllers [195]–[201]. Each particle is normally defined as a possible set of proportional, derivative and integral gains plus an integrative time constant

$$x = [k_P, k_I, k_D, T_s]. \tag{42}$$

By defining the objective function in terms of the system response overshoot, rise time, settling time or steady-state error, each particle can be evaluated in terms of the fitness function. Stability criteria such as Routh–Hurwitz can be used as a means to bound the particles within the stable region of the system response.

Abido [195] and Das and Venayagamoorthy [147], [196] have presented the application of PSO to optimal design of power system stabilizers (PSSs). Time-domain as well as frequency-domain eigenvalue analyses are conducted in order to evaluate the effectiveness of the approach under different operating conditions, i.e., disturbances, loading conditions, and system configuration changes. The proposed approach proves to be more efficient in providing damping for the system compared with the GA-based method. Similarly, Karimi *et al.* [197] have investigated the application of PSO to power system stability enhancement by tuning of a backstepping controller.

Okada *et al.* [198] have shown the use of PSO for parameter tuning of a fixed structure controller. Abdel–Magid and Abido [199], and Gaing [200] independently studied the tuning of an automatic generation control (AGC) systems using PSO. Venayagamoorthy has applied PSO for finding the optimal parameters of the PI controllers in a unified power flow controller (UPFC) [201] and Das *et al.* have done similarly with SPPSO in the case of a static VAR compensator (SVC) [146]. The design of optimal SVC damping controllers using SPPSO is reported in [202]. These studies indicate that, in general, the PSO-based approach has some advantages over the other techniques such as GA. These advantages include easy implementation, stable convergence characteristics, good computational efficiency, and more robustness.

In addition, Qiao *et al.* [203] have used the PSO algorithm to find the optimal parameters of the PI controllers for the rotor side converter of a doubly fed induction generator powered by a variable speed wind turbine. The main goal is to minimize the overcurrent in the rotor circuit during grid faults. Results show that the proposed approach is successful in finding the optimal parameters of the PI controllers and it improves the transient performance of the wind turbine generator system over a wide range of operating conditions.

Furthermore, the PSO algorithm has also been used in a fault tolerant control scheme for a static synchronous series compensator [204]. The system consists of a sensor evaluation and missing sensor restoration scheme SERS cascaded with a P-Q decoupled control scheme. An auto-encoder is used to capture the correlations between all of its input data. These correlations are then used by the PSO to search for the optimal estimates of the missing data when sensor readings are lost.

2) System Identification and Intelligent Control: PSO has also been applied to more complex structures such as ANNs in order to solve the problem of power systems identification and control. Application of PSO as a training algorithm of neural networks has been reported in [117], [205], and [206]. For such applications, the particles can be defined as arrays including the synaptic weight matrices of the hidden and output layers of the neural network

$$x = [w_{11}, \ldots, w_{ij}, \ldots, w_{mn}]. \tag{43}$$

The objective function corresponds to the mean-squared error (MSE) between the estimated and the actual physical values of the output

$$\mathrm{MSE} = \frac{1}{N}\sum_{1}^{N} |e_i|^2. \tag{44}$$

Gudise and Venayagamoorthy [205] have compared the efficiency of PSO with the conventional backpropagation (BP) training algorithm for a multilayer perceptron neural network (MLPN). Their simulation results show that for a small static system PSO converges to a smaller MSE in (44), with less computational effort (six times less the number of operations) compared with BP.

Mohagheghi *et al.* [117] have also investigated the efficiency of PSO-based training of radial basis function neural network (RBFN) and MLPN-based identifiers for a small power system. For a RBFN identifier, simulation results indicate comparable performances between PSO and BP, while in the case of a MLPN identifier, the performance of PSO is considerably

better, obtaining an average reduction of 84% in the MSE value compared with BP [117], [118].

Kiran *et al.* [207] and Jetti *et al.* [208] have also shown the efficiency of PSO in training a generalized neuron (GN) and dual function neuron to identify and control the dynamics of a SVC in the 12 bus FACTS benchmark power system, respectively. Also, Chia–Feng [206] has applied a hybrid of GA and PSO for designing and training recurrent networks and has demonstrated its superiority over the conventional techniques.

Moreover, Reynolds *et al.* [209] have developed an interesting implementation of PSO for inversion of large neural networks. Although results are promising, the algorithm is still too slow to be used in real-time applications. Also, Hirata *et al.* [210] investigated the performance of PSO in finding the optimal weights for a Lyapunov-based neurocontroller for the governor of a synchronous machine. In this study, they define a generic objective function as

$$\text{Minimize} \quad J(W, V) \tag{45}$$

where $J = \text{Max}(F)$ and F is the energy function of the system including the control signal of the neurocontroller. Since J is a nondifferentiable function, BP cannot be applied in this case, therefore, PSO represents an interesting alternative to adjust the parameters of the controller.

Regarding fuzzy logic-based controllers, an important issue is to define both the membership functions and weights of the rule set. Several authors have dealt with this problem for controllers: Control of permanent magnet synchronous motor for electric vehicle [211], load frequency control (LFC) [212], optimal control of a grid independent photovoltaic system [213], [214], and control of flexible AC transmission systems (FACTS) devices, particularly, a thyristor controlled series capacitor (TCSC) [215]. For this type of problem, the suggested definition of the particle is as follows [216]:

$$x = [S_1, X_{11}, X_{21}, W_1, \ldots, S_N, X_{1N}, X_{2N}, W_N] \tag{46}$$

where

S_i	is the shape function defined by integer value, e.g., $S = 1$ for Gaussian functions;
X_{1i} and X_{2i}	are the left and right boundaries, respectively;
W_i	is the weight.

If a more detailed shape is desirable, then other parameters can be incorporated to the particle such as means and standard deviations for the particular case of Gaussian membership functions [212], [215].

The objective function may vary according to the nature of the process to be controlled. For instance, Juang and Lu propose a fuzzy-based load frequency control, in which the fitness function is given by the minimization of the frequency deviations of two interconnected areas [212]. The same authors propose to use the root mean square error (RMSE) for speed deviations in the case of using a TCSC. In both applications, the hybrid between GA and PSO is used, which is found to be efficient in the search for better performing fuzzy rules in the proposed Fuzzy logic-base controllers [212], [215].

H. Other Applications

Several other applications have been reported in the literature in which PSO is used as a solution for a nonlinear optimization problem.

1) Electric Machinery: The application of PSO in the field of electric machines has not been extensively explored. Nevertheless, Emara *et al.* [217] applied PSO to estimate the stator fault in induction motors. Their proposed scheme detects the fault occurrence by monitoring the spectral content of the line current. PSO is used to design an estimator, which iteratively changes the estimated percentage of the stator faulty turns and compares the motor model outputs (stator currents) with the online measured faulty motor's currents. The difference between the measured and estimated motor currents is the function to be minimized by the swarm optimizer. They verified the validity and reliability of the proposed technique on experimental setups [217].

Bao *et al.* [218] applied PSO for an optimal design of a transverse flux permanent motor with the objective of reducing the cogging torque. Cogging torque is a part of the total torque developed by the motor that does not contribute to the net effective torque, and can result in pulsating torque, as well as speed ripple and vibration. Magnetic arc length is one of the most important parameters that directly affect the pitch and the surface area of the permanent magnet machine and, therefore, the resultant cogging torque produced by the machine. Reducing the magnet arc length decreases the surface area of the permanent magnet and, hence, the resultant cogging torque. However, this also reduces the magnetic flux produced by the machine and worsens the performance of the machine. Therefore, there should be a tradeoff between the two factors. Bao *et al.* applied PSO to find the optimum magnet arc length and thickness which gives a net torque which is still above a predefined desired value [218].

2) Capacitor and FACTS Placement: Capacitor allocation plays an important role in distribution system planning and operation. Optimal placement of capacitors in a network can help reduce the losses, improve the power factor, improve the voltage profile, provide on the spot reactive power generation, and therefore release the capacity of lines and feeders [219]. The nature of the problem is a nonlinear optimization approach which can be efficiently solved using PSO.

Each particle in this approach can be defined as a vector of integer numbers. Each entry of the vector corresponds to the amount of shunt compensation at a specific location (node) in the system. The capacitors can be fixed or variable. Different objective functions can then be defined for the purpose of updating the particles. Esmin *et al.* have used the total sum of the branch losses in the system as the objective function which is evaluated by executing the load flow after every iteration [97]. Yu *et al.* have considered the system total harmonic distortion as the objective function [220]. A harmonic load flow is performed after every iteration in order to determine the total harmonic distortion of the system.

Following a similar approach, Hernandez *et al.* [221] and del Valle *et al.* [222] have used PSO in finding the optimal STATCOM location and size in a medium size power system (45 bus power network). STATCOM is a shunt type of a FACTS device. In this case, the fitness function is defined based on the voltage profile throughout the power system, in a way that

the voltage deviations of the buses, with respect to their corresponding nominal values, are minimized by installing a minimum STATCOM size.

3) Generator Maintenance and Unit-Commitment Scheduling: Koay and Srinivasan have introduced the application of PSO to generator maintenance scheduling [223]. Results are obtained for a practical scheduling and optimization problem for which evolutionary computation (EC)-based approaches have previously been applied and found to be effective. It has been shown that PSO-based approaches produce superior performance compared with GA or ES. The paper also presents a hybrid spawning PSO and ES; in this approach, valuable features from both PSO and ES are combined to provide a simple hybrid model that could be readily used in other applications.

For the unit-commitment scheduling problem, Miranda and Win–Oo [224] have presented the application of differential evolutionary particle swarm optimization to the unit commitment-generator scheduling power system problem. In this case, given a set of generators and their generation cost curves, it is determined which generators should operate or not and at which generation level, thus minimizing the overall cost which include startup and operational costs for a particular set of system operating conditions. The used algorithm is an improved version of the EPSO that incorporates an improvement in the movement rule by adjusting the memory element. The PSO-based algorithm is compared with other GA-based algorithms and produces better results [224].

4) Short-Term Load Forecasting (STLF): Another area in which PSO has recently found application is short-term load forecasting (STLF). One publication [225] has applied PSO in order to identify the autoregressive moving average with exogenous variable (ARMAX) model of the load. The paper presented an alternative technique of model order determination and parameter estimation to achieve a global minimum forecasting error within an efficient computation time. The fitting accuracy of the model is also improved. The proposed technique is tested on four different types of practical power load data for one-day and one-week ahead. The PSO-based load forecasting has a better forecasting accuracy, with superior convergence characteristics, and shorter executing time than the EP and traditional stochastic time series (STS) methods.

5) Generator Contributions to Transmission System: One recent publication has been devoted to the application of PSO to generator contributions to transmission systems [226]. It presents a parallel VEPSO algorithm for determining generator contributions to the transmission system under various generator and transmission line constraints. The problem is formulated as a multiobjective optimization problem that takes into account nonlinear characteristics of the power systems such as generator prohibited operating zones and thermal limits of lines. The contributions of generators to the real-power flows in transmission lines are modeled as positions of agents or particles in the swarm. For four different test systems, the feasibility of the proposed methodology is demonstrated and compared with other analytical methods. Although the results show that the parallel VEPSO algorithm is slower than the analytical method, it is capable of obtaining high accuracy results within a reasonable computing time.

V. Future Applications

The previous section provided an overview of the existing applications of PSO in power systems. However, the capabilities of PSO are by no means limited to those applications. Many decision problems in planning, operation and control of power systems require a nonlinear stochastic optimization problem to be solved and this is too sophisticated for any analytically based approach to handle. An increase in the size of the power network can further degrade the effectiveness of these analytical techniques. Many of these optimization problems can be readily formulated and addressed by PSO. Some potential power system related applications of PSO that are not yet explored in the literature are briefly discussed in this section.

Previously, it has been shown that PSO-based techniques can be used in order to solve the nonlinear optimization problem of capacitor placement in a power network [220]. This can be further extended to include more components in the network, such as the on-load tap changing transformers and/or FACTS devices. While an analytical-based technique might suffer from the curse of dimensionality, PSO can still manage to provide the optimum solution in terms of the appropriate tap/switch position or firing angle.

Power system design at the distribution level is another area where PSO can further prove its efficiency. Successful applications of PSO have been reported in distribution system reconfigurations for aggregating the loads [182], [183], balancing loads between the feeders [184], or finding the most appropriate locations for sectionalized devices [185], [186]. However, the problem can be extended to more general cases of designing the distribution system layout, such as deciding on the locations of substations, configurations of the overhead lines, transformer design, and size or topology of the cable system. Financial computations as well as the future expansion considerations can be incorporated into the fitness function.

Electric machinery design is the area which seems least explored by PSO. In fact, as far as the authors know, the only work published to date on the application of PSO to electric machinery design was reported by Bao *et al.* [218]. In this work, PSO was applied in order to optimize the design to reduce the cogging torque of a transverse flux permanent motor. Nevertheless, the literature has reported the application of other artificial intelligence-based techniques such as GA, ANN, and fuzzy logic to the electric machinery design problem [227]–[229]. In general, results show that the application of such techniques can improve the design by optimizing the desired parameters, while meeting the required technical specifications. Clearly, detailed studies need to be carried out in order to evaluate the effectiveness of PSO compared with the other available techniques in this field. Potentially, PSO can be successfully applied for designing various aspects of electric machines, such as the winding, mass/volume, air gap, and operational parameters.

In all the applications reviewed in this paper, the PSO algorithm is executed offline. It uses the previously stored system data in order to solve an optimization problem. Any change to the problem/power system environment requires saving the new information and re-executing the PSO algorithm offline. However, many applications in power systems need an online solution, which is updated at every specified time step. Controller

design is perhaps the most important application that belongs to this category. Modifications need to be done to the PSO algorithm in such a way that it is able to efficiently work in online applications. Preliminary work by one of the authors has reported the use of PSO to train a generalized neuron online [230].

VI. Conclusion

Solving an optimization problem is one of the common scenarios that occur in most engineering applications. Classical techniques such as LP and NLP are efficient approaches that can be used in special cases. As the complexities of the problem increase, especially with the introduction of uncertainties to the system, more complicated optimization techniques, such as stochastic programming or DP have to be used. However, these analytical methods are not easy to implement for most of the real-world problems. In fact, for many problems, the curse of dimensionality makes the approach unfeasible to implement.

The above issues are of particular importance when solving optimization problems in a power system. As a highly nonlinear, nonstationary system with noise and uncertainties, a power network can have a large number of states and parameters. Implementing any of the classical analytical optimization approaches might not be feasible in most of the cases. On the other hand, PSO can be an alternative solution. It is a stochastic-based search technique that has its roots in artificial life and social psychology, as well as in engineering and computer science. It utilizes a "population," called particles, which flows through the problem hyperspace with given velocities; in each iteration, velocities are stochastically adjusted considering the historical best position for the particle itself and the neighborhood best position (both of them defined according to a predefined fitness function). Then, the movement of each particle naturally evolves to an optimal or near-optimal solution.

This paper has described the basic concepts of PSO along with its numerous variants that can be employed in different optimization problems. In addition, a review of the applications of PSO in power systems-based optimization problems is presented to give the reader some insight of how PSO can serve as a solution to some of the most complicated engineering optimization problems.

References

[1] A. Schrijver, *Theory of Linear and Integer Programming*. New York: Wiley, 1998.

[2] J. Delson and S. Shahidehpour, "Linear programming applications to power system economics, planning and operations," *IEEE Trans. Power Syst.*, vol. 7, no. 3, pp. 1155–1163, Aug. 1992.

[3] C. Kurucz, D. Brandt, and S. Sim, "A linear programming model for reducing system peak through customer load control programs," *IEEE Trans. Power Syst.*, vol. 11, no. 4, pp. 1817–1824, Nov. 1996.

[4] H. Khodr, J. Gomez, L. Barnique, J. Vivas, P. Paiva, J. Yusta, and A. Urdaneta, "A linear programming methodology for the optimization of electric power-generation schemes," *IEEE Trans. Power Syst.*, vol. 17, no. 3, pp. 864–869, Aug. 2002.

[5] K. Ng and G. Shelbe, "Direct load control—A profit-based load management using linear programming," *IEEE Trans. Power Syst.*, vol. 13, no. 2, pp. 688–694, May 1998.

[6] A. Farag, S. Al-Baiyat, and T. Cheng, "Economic load dispatch multiobjective optimization procedures using linear programming techniques," *IEEE Trans. Power Syst.*, vol. 10, no. 2, pp. 731–738, May 1995.

[7] R. Jabr, A. Coonick, and B. Cory, "A homogeneous linear programming algorithm for the security constrained economic dispatch problem," *IEEE Trans. Power Syst.*, vol. 15, no. 3, pp. 930–936, Aug. 2000.

[8] A. El-Keib and H. Singh, "Fast linear programming state estimation using the dual formulation," *IEEE Trans. Power Syst.*, vol. 7, no. 2, pp. 620–628, May 1992.

[9] B. Chattopadhyay, M. Sachdev, and T. Sidhu, "An on-line relay coordination algorithm for adaptive protection using linear programming technique," *IEEE Trans. Power Del.*, vol. 11, no. 1, pp. 165–173, Jan. 1996.

[10] I. Habiballah and M. Irving, "Observability analysis for state estimation using linear programming," *IEE Proc. Generation, Transmission, Distrib.*, vol. 148, no. 2, pp. 142–145, Mar. 2001.

[11] G. Fahd and G. Shelbe, "Optimal power flow emulation of interchange brokerage systems using linear programming," *IEEE Trans. Power Syst.*, vol. 7, no. 2, pp. 497–504, May 1992.

[12] D. Kirschen and H. Van Meeteren, "MW/voltage control in a linear programming based optimal power flow," *IEEE Trans. Power Syst.*, vol. 3, no. 2, pp. 481–489, May 1988.

[13] H. Elrefaie, M. Irving, and S. Zitouni, "A parallel processing algorithm for co-ordination of directional overcurrent relays in interconnected power systems," *Proc. Inst. Elect. Eng. Generation, Transmission, Distrib.*, vol. 141, no. 5, pp. 514–520, Sep. 1994.

[14] S. Tong and S. Shahidehpour, "Combination of Lagrangian-relaxation and linear-programming approaches for fuel-constrained unit-commitment problems," *Proc. Inst. Elect. Eng.*, vol. 136, no. 3, pt. Part C, pp. 162–174, May 1989.

[15] D. Chattopadhyay, "A practical maintenance scheduling program mathematical model and case study," *IEEE Trans. Power Syst.*, vol. 13, no. 4, pp. 1475–1480, Nov. 1998.

[16] A. Motto, J. Arroyo, and F. Galiana, "A mixed-integer LP procedure for the analysis of electric grid security under disruptive threat," *IEEE Trans. Power Syst.*, vol. 20, no. 3, pp. 1357–1365, Aug. 2005.

[17] H. Khodr, J. Gomez, L. Barnique, J. Vivas, P. Paiva, J. Yusta, and A. Urdaneta, "A linear programming methodology for the optimization of electric power-generation schemes," *IEEE Trans. Power Syst.*, vol. 17, no. 3, pp. 864–869, Aug. 2002.

[18] J. Restrepo and F. Galiana, "Unit commitment with primary frequency regulation constraints," *IEEE Trans. Power Syst.*, vol. 20, no. 4, pp. 1836–1842, Nov. 2005.

[19] G. Chang, M. Aganagic, J. Waight, J. Medina, T. Burton, S. Reeves, and M. Christoforidis, "Experiences with mixed integer linear programming based approaches on short-term hydro scheduling," *IEEE Trans. Power Syst.*, vol. 16, no. 4, pp. 743–749, Nov. 2001.

[20] S. Ashok and R. Banerjee, "An optimization mode for industrial load management," *IEEE Trans. Power Syst.*, vol. 16, no. 4, pp. 879–884, Nov. 2001.

[21] P. Paiva, H. Khodr, J. Dominguez, J. Yusta, and A. Urdaneta, "Integral planning of primary-secondary distribution systems using mixed integer linear programming," *IEEE Trans. Power Syst.*, vol. 20, no. 2, pp. 1134–1143, May 2005.

[22] N. Alguacil, A. Motto, and A. Conejo, "Transmission expansion planning: A mixed-integer LP approach," *IEEE Trans. Power Syst.*, vol. 18, no. 3, pp. 1070–1077, Aug. 2003.

[23] F. Lima, F. Galiana, I. Kockar, and J. Muñoz, "Phase shifter placement in large-scale systems via mixed integer linear programming," *IEEE Trans. Power Syst.*, vol. 18, no. 3, pp. 1029–1034, Aug. 2003.

[24] L. Bahiense, G. Oliveira, M. Pereira, and S. Granville, "A mixed integer disjunctive model for transmission network expansion," *IEEE Trans. Power Syst.*, vol. 16, no. 3, pp. 560–565, Aug. 2001.

[25] M. Farrag and M. El-Metwally, "New method for transmission planning using mixed-integer programming," *Proc. Inst. Elect. Eng. Generation, Transmission, Distrib.*, vol. 135, no. 4, pp. 319–323, Jul. 1988.

[26] F. Soudi and K. Tomsovic, "Optimized distribution protection using binary programming," *IEEE Trans. Power Del.*, vol. 13, no. 1, pp. 218–224, Jan. 1998.

[27] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York: Springer, 1999.

[28] L. Zarate, C. Castro, J. Ramos, and E. Ramos, "Fast computation of voltage stability security margins using nonlinear programming techniques," *IEEE Trans. Power Syst.*, vol. 21, no. 1, pp. 19–27, Feb. 2006.

[29] A. Schellenberg, W. Rosehart, and J. Aguado, "Cumulant-based stochastic nonlinear programming for variance constrained voltage stability analysis of power systems," *IEEE Trans. Power Syst.*, 2006, accepted for publication.

[30] W. Hua, H. Sasaki, J. Kubokawa, and R. Yokoyama, "An interior point nonlinear programming for optimal power flow problems with a novel data structure," *IEEE Trans. Power Syst.*, vol. 13, no. 3, pp. 870–877, Aug. 1998.

[31] W. Hua, H. Sasaki, J. Kubokawa, and R. Yokoyama, "Large scale hydrothermal optimal power flow problems based on interior point nonlinear programming," *IEEE Trans. Power Syst.*, vol. 15, no. 1, pp. 396–403, Feb. 2000.

[32] H. Habibollahzadeh, G. Luo, and A. Semlyen, "Hydrothermal optimal power flow based on a combined linear and nonlinear programming methodology," *IEEE Trans. Power Syst.*, vol. 4, no. 2, pp. 530–537, May 1989.

[33] G. Torres and V. Quintana, "On a nonlinear multiple-centrality-corrections interior-point method for optimal power flow," *IEEE Trans. Power Syst.*, vol. 16, no. 2, pp. 222–228, May 2001.

[34] A. Ramos, I. Perez, and J. Bogas, "A nonlinear programming approach to optimal static generation expansion planning," *IEEE Trans. Power Syst.*, vol. 4, no. 3, pp. 1140–1146, Aug. 1989.

[35] D. Lidgate and B. H. Amir, "Optimal operational planning for a hydroelectric generation system," *IEE Proc. Inst. Elect. Eng. Generation, Transmission, Distrib.*, vol. 135, no. 3, pp. 169–181, May 1988.

[36] G. Shrestha, B. Pokharel, T. Tek, and S. Fleten, "Medium term power planning with bilateral contracts," *IEEE Trans. Power Syst.*, vol. 20, no. 2, pp. 627–633, May 2005.

[37] A. Escobar, R. Gallego, and R. Romero, "Multistage and coordinated planning of the expansion of transmission systems," *IEEE Trans. Power Syst.*, vol. 19, no. 2, pp. 735–744, May 2004.

[38] E. Aponte and J. Nelson, "Time optimal load shedding for distributed power systems," *IEEE Trans. Power Syst.*, vol. 21, no. 1, pp. 269–277, Feb. 2006.

[39] S. Bruno, E. De Tuglie, M. La Scala, and P. Scarpellini, "Dynamic security corrective control by UPFCs," *IEEE Trans. Power Syst.*, vol. 16, no. 3, pp. 490–497, Aug. 2001.

[40] E. De Tuglie, M. Dicorato, M. La Scala, and P. Scarpellini, "A static optimization approach to assess dynamic available transfer capability," *IEEE Trans. Power Syst.*, vol. 15, no. 3, pp. 1069–1076, Aug. 2000.

[41] W. Chang and W. Grady, "Minimizing harmonic voltage distortion with multiple current-constrained active power line conditioners," *IEEE Trans. Power Del.*, vol. 12, no. 2, pp. 837–843, Apr. 1997.

[42] S. Virmani, E. Adrian, K. Imhof, and S. Mukherjee, "Implementation of a Lagrangian relaxation based unit commitment problem," *IEEE Trans. Power Syst.*, vol. 4, no. 4, pp. 1373–1380, Nov. 1989.

[43] D. Pudjianto, S. Ahmed, and G. Strbac, "Allocation of VAr support using LP and NLP based optimal power flows," *Proc. Inst. Elect. Eng. Generation, Transmission, Distrib.*, vol. 149, no. 4, pp. 377–383, Jul. 2002.

[44] R. Gallego, A. Monticelli, and R. Romero, "Optimal capacitor placement in radial distribution networks," *IEEE Trans. Power Syst.*, vol. 16, no. 4, pp. 630–637, Nov. 2001.

[45] C. Tse, K. Wang, C. Chang, and K. Tsang, "Parameter optimisation of robust power system stabilisers by probabilistic approach," *Proc. Inst. Elect. Eng. Generation, Transmission, Distrib.*, vol. 147, no. 2, pp. 69–75, Mar. 2000.

[46] J. Fan and L. Zhang, "Real-time economic dispatch with line flow and emission constraints using quadratic programming," *IEEE Trans. Power Syst.*, vol. 13, no. 2, pp. 320–325, May 1998.

[47] M. Yoshikawa, N. Toshida, H. Nakajima, Y. Harada, M. Tsurugai, and Y. Nakata, "On-line economic load dispatch based on fuel cost dynamics," *IEEE Trans. Power Syst.*, vol. 12, no. 1, pp. 315–320, Feb. 1997.

[48] C. Lehmkoster, "Security constrained optimal power flow for an economical operation of FACTS-devices in liberalized energy markets," *IEEE Trans. Power Del.*, vol. 17, no. 2, pp. 603–608, Apr. 2002.

[49] N. Grudinin, "Reactive power optimization using successive quadratic programming method," *IEEE Trans. Power Syst.*, vol. 13, no. 4, pp. 1219–1225, Nov. 1998.

[50] X. Lin, A. David, and C. Yu, "Reactive power optimisation with voltage stability consideration in power market systems," *Proc. Inst. Elect. Eng. Generation, Transmission, Distrib.*, vol. 150, no. 3, pp. 305–310, May 2003.

[51] I. Nejdawi, K. Clements, and P. Davis, "An efficient interior point method for sequential quadratic programming based optimal power flow," *IEEE Trans. Power Syst.*, vol. 15, no. 4, pp. 1179–1183, Nov. 2000.

[52] A. Conejo, N. Alguacil, and G. Fernandez, "Allocation of the cost of transmission losses using a radial equivalent network," *IEEE Trans. Power Syst.*, vol. 18, no. 4, pp. 1353–1358, Nov. 2003.

[53] E. Finardi and E. da Silva, "Solving the hydro unit commitment problem via dual decomposition and sequential quadratic programming," *IEEE Trans. Power Syst.*, 2006, accepted for publication.

[54] E. Diaz and J. C. Pidre, "Optimal planning of unbalanced networks using dynamic programming optimization," *IEEE Trans. Power Syst.*, vol. 19, no. 4, pp. 2077–2085, Nov. 2004.

[55] N. Boulaxis and M. Papdopoulos, "Optimal feeder routing in distribution system planning using dynamic programming technique and GIS facilities," *IEEE Trans. Power Del.*, vol. 17, no. 1, pp. 242–247, Jan. 2002.

[56] Y. Kun, "Demand subscription services-an iterative dynamic programming for the substation suffering from capacity shortage," *IEEE Trans. Power Syst.*, vol. 18, no. 2, pp. 947–953, May 2003.

[57] E. Diaz, E. Miguez, and J. Cidras, "Design of large rural low-voltage networks using dynamic programming optimization," *IEEE Trans. Power Syst.*, vol. 16, no. 4, pp. 898–903, Nov. 2001.

[58] F. Lu and Y. Hsu, "Reactive power/voltage control in a distribution substation using dynamic programming," *Proc. Inst. Elect. Eng. Generation, Transmission, Distrib.*, vol. 142, no. 6, pp. 639–645, Nov. 1995.

[59] Y. Hsu and H. Kuo, "Dispatch of capacitors on distribution system using dynamic programming," *Proc. Inst. Elect. Eng. Generation, Transmission, Distrib.*, vol. 140, no. 6, pp. 433–438, Nov. 1993.

[60] Z. Ouyang and S. Shahidehpour, "An intelligent dynamic programming for unit commitment application," *IEEE Trans. Power Syst.*, vol. 6, no. 3, pp. 1203–1209, Aug. 1991.

[61] W. Hobbs, G. Hermon, S. Warner, and G. Shelbe, "An enhanced dynamic programming approach for unit commitment," *IEEE Trans. Power Syst.*, vol. 3, no. 3, pp. 1201–1205, Aug. 1988.

[62] D. Bertsekas, *Dynamic Programming and Optimal Control*. Boston, MA: Athena Scientific, 2000, pp. 269–359.

[63] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Norwell, MA: Kluwer, 1989.

[64] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw. (ICNN)*, Nov. 1995, vol. 4, pp. 1942–1948.

[65] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Machine and Human Science (MHS)*, Oct. 1995, pp. 39–43.

[66] S. Yang, M. Wang, and L. Jiao, "A quantum particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2004, vol. 1, pp. 320–324.

[67] R. Eberhart, Y. Shi, and J. Kennedy, *Swarm Intelligence*. San Mateo, CA: Morgan Kaufmann, 2001.

[68] A. P. Engelbrecht, "Particle swarm optimization: Where does it belong?," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 48–54.

[69] M. Millonas, "Swarms, phase transitions, and collective intelligence," in *Artificial Life III, Proc. Santa Fe Institute Studies in the Sciences of Complexity*, C. G. Langton, Ed. New York: Addison-Wesley, 1994, vol. XVII, pp. 417–445.

[70] M. Dorigo, "Optimization, learning and natural algorithms," (in Italian) Ph.D. dissertation, Politecnico di Milano, Milan, Italy, 1992.

[71] M. Dorigo and T. Stützle, *Ant Colony Optimization*. Cambridge, MA: MIT Press, 2004, ISBN 0-262-04219-3.

[72] Ant Colony Optimization. [Online]. Available: http://en.wikipedia.org/wiki/Ant_colony_optimization

[73] J. M. Bishop, "Stochastic Searching Networks," in *Proc. Inst. Elect. Eng. Conf. Artif. Neural Netw.*, London, Oct. 1989, pp. 329–331.

[74] A. Loengarov and V. Tereshko, "A minimal model of honey bee foraging," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 175–182.

[75] Y. Liu and K. Passino, "Biomimicry of social foraging bacteria for distributed optimization: Models, principles, and emergent behaviors," *J. Optim. Theory Appl.*, vol. 115, no. 3, pp. 603–628, Dec. 2002.

[76] S. Mishra, "A hybrid least square-Fuzzy bacterial foraging strategy for harmonic estimation," *IEEE Trans. Evol. Comput.*, vol. 9, no. 1, pp. 61–73, Feb. 2005.

[77] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE. Control Syst. Mag.*, pp. 52–67, Jun. 2002.

[78] J. Kennedy, "The particle swarm: Social adaptation of knowledge," in *Proc. IEEE Int. Conf. Evol. Comput.*, Apr. 1997, pp. 303–308.

[79] D. Boeringer and D. Werner, "Particle swarm optimization versus genetic algorithms for phased array synthesis," *IEEE Trans. Antennas Propagat.*, vol. 52, no. 3, pp. 771–779, Mar. 2004.

[80] Y. Shi, "Feature article on particle swarm optimization," *IEEE Neural Network Society, Feature Article*, pp. 8–13, Feb. 2004.

[81] M. Richards and D. Ventura, "Choosing a starting configuration for particle swarm optimization," in *Proc. IEEE Int. Joint. Conf. Neural Netw.*, Jul. 2004, vol. 3, pp. 2309–2312.

[82] E. F. Campana, G. Fasano, and A. Pinto, "Dynamic system analysis and initial particles position in particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 202–209.

[83] J. Kennedy, "Small worlds and mega-minds: Effects of neighborhood topology on particle swarm performance," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 1999, vol. 3, pp. 1931–1938.

[84] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1671–1676.

[85] J. Kennedy and R. Mendes, "Neighborhood topologies in fully-informed and best-of-neighborhood particle swarms," in *Proc. IEEE Int. Workshop Soft Computing in Industrial Applications*, Jun. 2003, pp. 45–50.

[86] R. Eberhart and Y. Shi, "Particle swarm optimization: Developments, applications and resources," in *Proc. IEEE Congr. Evol. Comput.*, May 2001, vol. 1, pp. 81–86.

[87] X. Hu, Y. Shi, and R. Eberhart, "Recent advances in particle swarm," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2004, vol. 1, pp. 90–97.

[88] H. Fan and Y. Shi, "Study on Vmax of particle swarm optimization," in *Proc. Workshop on Particle Swarm Optimization, Purdue School of Engineering and Technology*, Indianapolis, IN, Apr. 2001.

[89] A. Abido, "Particle swarm optimization for multimachine power system stabilizer design," in *Proc. Power Engineering Society Summer Meeting (PES)*, Jul. 2001, vol. 3, pp. 1346–1351.

[90] A. Abido, "Optimal power flow using particle swarm optimization," *Int. J. Elect. Power Energy Syst.*, vol. 24, no. 7, pp. 563–571, Oct. 2002.

[91] E. Ozcan and C. Mohan, "Particle swarm optimization: Surfing the waves," in *Proc. IEEE Congress Evol. Comput.*, Jul. 1999, vol. 3, pp. 1939–1944.

[92] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, Feb. 2002.

[93] R. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proc. IEEE Congress Evol. Comput.*, Jul. 2000, vol. 1, pp. 84–88.

[94] F. Bergh and A. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, Jun. 2004.

[95] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE World Congr. Comput. Intell.*, May 1998, pp. 69–73.

[96] Y. Shi and R. Eberhart, "Empirical study of particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 1999, vol. 3, pp. 1945–1950.

[97] A. Esmin, G. Torres, and A. Zambroni, "A hybrid particle swarm optimization applied to loss power minimization," *IEEE Trans. Power Syst.*, vol. 20, no. 2, pp. 859–866, May 2005.

[98] Y. Shi and R. Eberhart, "Parameter selection in particle swarm optimization," in *Proc. 7th Int. Conf. Evol. Program. (EP)*, May 1998, pp. 591–600.

[99] Y. Shi and R. Eberhart, "Empirical study of particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 1999, vol. 3, pp. 1945–1950.

[100] X. Chen and Y. Li, "An improved stochastic PSO with high exploration ability," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 228–235.

[101] J. Kennedy and R. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.: Computational Cybernetics and Simulation (ICSMC)*, Oct. 1997, vol. 5, pp. 4104–4108.

[102] K. Parsopoulos and M. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," *Natural Computing*, vol. 1, pp. 235–306, May 2002.

[103] E. Laskari, K. Parsopoulos, and M. Vrahatis, "Particle swarm optimization for integer programming," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1582–1587.

[104] T. Blackwell and P. Bentley, "Don't push me! Collision-avoiding swarms," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1691–1696.

[105] T. Krink, J. Vesterstrom, and J. Riget, "Particle swarm optimization with spatial particle extension," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1474–1479.

[106] V. Miranda and N. Fonseca, "New evolutionary particle swarm algorithm (EPSO) applied to voltage/VAR control," in *Proc. 14th Power Syst. Comput. Conf.*, Jun. 2002.

[107] M. Lovbjerg and T. Krink, "Extending particle swarms with self-organized criticality," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1588–1593.

[108] A. El-Dib, H. Youssef, M. El-Metwally, and Z. Osman, "Load flow solution using hybrid particle swarm optimization," in *Proc. Int. Conf. Elect., Electron., Comput. Eng.*, Sep. 2004, pp. 742–746.

[109] S. Naka, T. Genji, T. Yura, and Y. Fukuyama, "A hybrid particle swarm optimization for distribution state estimation," *IEEE Trans. Power Syst.* , pp. 60–68, Feb. 2003.

[110] P. Angeline, "Using selection to improve particle swarm optimization," in *Proc. IEEE Int. Conf. Evol. Comput.*, May 1998, pp. 84–89.

[111] V. Miranda and N. Fonseca, "EPSO best of two worlds meta-heuristic applied to power system problems," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1080–1085.

[112] V. Miranda and N. Fonseca, "EPSO evolutionary particle swarm optimization, a new algorithm with applications in power systems," in *IEEE/PES Transmission and Distribution Conf. Exhibition Asia Pacific*, Oct. 2002, vol. 2, pp. 745–750.

[113] W. Zhang and X. Xie, "DEPSO: Hybrid particle swarm with differential evolution operator," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2003, vol. 4, pp. 3816–3821.

[114] H. Talbi and M. Batouche, "Hybrid particle swarm with differential evolution for multimodal image registration," in *Proc. IEEE Int. Conf. Ind. Technol.*, Dec. 2004, vol. 3, pp. 1567–1572.

[115] P. Moore and G. Venayagamoorthy, "Evolving combinational logic circuits using particle swarm, differential evolution and hybrid DEPSO," *Int. J. Neural Syst.*, vol. 16, no. 2, pp. 163–177, 2006.

[116] S. Kannan, S. Slochanal, and N. Padhy, "Application of particle swarm optimization technique and its variants to generation expansion problem," *ELSERVIER Electric Power Syst. Res.*, vol. 70, no. 3, pp. 203–210, Aug. 2004.

[117] S. Mohagheghi, Y. Del Valle, G. Venayagamoorthy, and R. Harley, "A comparison of PSO and backpropagation for training RBF neural networks for identification of a power system with STATCOM," in *Proc. IEEE Swarm Intell. Symp.*, Jun. 2005, pp. 381–384.

[118] Y. del Valle, S. Mohagheghi, G. Venayagamoorthy, and R. Harley, "Training MLP neural networks for identification of a small power system: Comparison of PSO and backpropagation," in *Proc. 6th Int. Conf. Power Syst. Operation and Planning*, May 2005, pp. 153–157.

[119] Y. Shi and R. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, May 2001, vol. 1, pp. 101–106.

[120] Y. Shi and R. Eberhart, "Particle swarm optimization with fuzzy adaptive inertia weight," in *Proc. Workshop on Particle Swarm Optimization, Purdue School of Engineering and Technology*, Indianapolis, IN, Apr. 2001.

[121] S. Doctor, G. Venayagamoorthy, and V. Gudise, "Optimal PSO for collective robotic search applications," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2004, vol. 2, pp. 1390–1395.

[122] M. Khajenejad, F. Afshinmanesh, A. Marandi, and B. N. Araabi, "Intelligent particle swarm optimization using Q-learning," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 7–12.

[123] G. Venayagamoorthy, "Adaptive critics for dynamic particle swarm optimization," in *Proc. IEEE Int. Symp. Intell. Control*, Sep. 2004, pp. 380–384.

[124] S. Doctor and G. Venayagamoorthy, "Improving the performance of particle swarm optimization using adaptive critics designs," in *Proc. IEEE Swarm Intell. Symp.*, Jun. 2005, pp. 393–396.

[125] W. Zhang, Y. Liu, and M. Clerc, "An adaptive PSO algorithm for reactive power optimization," in *Proc. 6th Int. Conf. Advances in Power System Control, Operation and Management*, Nov. 2003, pp. 302–307.

[126] X. Li, "Adaptively choosing neighborhood bests using species in a particle swarm optimizer for multimodal function optimization," in *Proc. Genetic Evol. Comput. Conf.*, Jun. 2004, pp. 105–116.

[127] K. Parsopoulos and M. Vrahatis, "Particle swarm optimization method in multiobjective problems," in *Proc. ACM Symp. Appl. Comput.*, Mar. 2002, pp. 603–607.

[128] T. Ray, "Constrained robust optimal design using a multiobjective evolutionary algorithm," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 1, pp. 419–424.

[129] S. Mostaghim and J. Teich, "Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO)," in *Proc. IEEE Swarm Intell. Symp.*, Apr. 2003, pp. 26–33.

[130] J. E. Fieldsend, R. M. Everson, and S. Singh, "Using unconstrained elite archives for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 7, no. 3, pp. 305–323, Jun. 2003.

[131] X. Hu, "Particle swarm optimization," in *Tutorial of the IEEE Swarm Intell. Symp.*, 2006.

[132] T. Ray and K. Liew, "A swarm metaphor for multi-objective design optimization," *Engineering Optimization*, vol. 34, no. 2, pp. 141–153, Mar. 2002.

[133] C. Coello and M. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1051–1056.

[134] C. Coello, G. Pulido, and M. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 256–279, Jun. 2004.

[135] X. Li, "A non-dominated sorting particle swarm optimizer for multi-objective optimization," in *Proc. Genetic and Evol. Comput. Conf.*, Jul. 2003, pp. 37–48.

[136] M. Salazar-Lechuga and J. E. Rowe, "Particle swarm optimization and auto-fitness sharing to solve multi-objective optimization problems," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 90–97.

[137] X. Hu and R. Eberhart, "Multiobjective optimization using dynamic neighborhood particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1677–1681.

[138] X. Hu, R. Eberhart, and Y. Shi, "Particle swarm with extended memory for multiobjective optimization," in *Proc. IEEE Swarm Intell. Symp.*, Apr. 2003, pp. 193–197.

[139] T. Bartz-Beielstein, P. Limbourg, J. Mehnen, K. Schmitt, K. Parsopoulos, and M. Vrahatis, "Particle swarm optimizers for Pareto optimization with enhanced archiving techniques," in *Proc. IEEE Congr. Evol. Comput.*, Dec. 2003, vol. 3, pp. 1780–1787.

[140] K. Parsopoulos, D. Tasoulis, and M. Vrahatis, "Multiobjective optimization using parallel vector evaluated particle swarm optimization," in *Proc. Int. Conf. Artif. Intell. Appl.*, Feb. 2004, vol. 2, pp. 823–828.

[141] J. Vlachogiannis and K. Lee, "Determining generator contributions to transmission system using parallel vector evaluated particle swarm optimization," *IEEE Trans. Power Syst.*, vol. 20, no. 4, pp. 1765–1774, Nov. 2005.

[142] Y. del Valle, J. C. Hernandez, G. Venayagamoorthy, and R. G. Harley, "Multiple STATCOM allocation and sizing using particle swarm optimization," in *Proc. IEEE PES Power Syst. Conf. Expo.*, Oct. 2006, pp. 1884–1891.

[143] R. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proc. IEEE Congr. Evol. Comput.*, May 2001, vol. 1, pp. 94–100.

[144] X. Hu and R. Eberhart, "Adaptive particle swarm optimization: Detection and response to dynamic systems," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1666–1670.

[145] A. Carlisle and G. Dozier, "Adapting particle swarm optimization to dynamic environments," in *Proc. Int. Conf. Artif. Intell.*, Jun. 2000, pp. 429–434.

[146] T. K. Das, S. R. Jetti, and G. K. Venayagamoorthy, "Optimal design of a SVC controller using a small population based PSO," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 156–161.

[147] T. Das and G. Venayagamoorthy, "Optimal design of power system stabilizers using a small population based PSO," in *Proc. IEEE PES General Meeting*, Jun. 2006.

[148] C. Mohan and B. Al-Kazemi, "Discrete particle swarm optimization," in *Proc. Workshop on Particle Swarm Optimization, 2001. Purdue School of Engineering and Technology*, Indianapolis, IN, Apr. 2001.

[149] S. Yang, M. Wang, and L. Jiao, "A quantum particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2004, vol. 1, pp. 320–324.

[150] P. Moore and G. Venayagamoorthy, "Evolving combinational logic circuits using a hybrid quantum evolution and particle swarm inspired algorithm," in *Proc. NASA/DoD Conf. Evolvable Hardware (EH)*, Jun. 2005, pp. 97–102.

[151] W. Cedeño and D. Agrafiotis, "A comparison of particle swarms techniques for the development of quantitative structure-activity relationship models for drug design," in *Proc. IEEE Comput. Syst. Bioinformatics Conf.*, Aug. 2005, pp. 322–331.

[152] B. Secrest and G. Lamont, "Visualizing particle swarm optimization—Gaussian particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, Apr. 2003, pp. 198–204.

[153] R. Krohling, "Gaussian particle swarm with jumps," in *Proc. IEEE Congr. Evol. Comput.*, Sep. 2005, vol. 2, pp. 1226–1231.

[154] R. Krohling, "Gaussian swarm: A novel particle swarm optimization algorithm," in *Proc. IEEE Con. Cybern. Intell. Syst.*, Dec. 2004, vol. 1, pp. 372–376.

[155] P. Biskas, N. Ziogos, A. Tellidou, C. Zoumas, A. Bakirtzis, V. Petridis, and A. Tsakoumis, "Comparison of two metaheuristics with mathematical programming methods for the solution of OPF," in *Proc. Int. Conf. Intell. Syst. Appl. Power Syst.*, Nov. 2005, pp. 510–515.

[156] X. Xie, W. Zhang, and Z. Yang, "A dissipative particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, May 2002, vol. 2, pp. 1456–1461.

[157] S. He, J. Wen, E. Prempain, Q. Wu, J. Fitch, and S. Mann, "An improved particle swam optimization for optimal power flow," in *Proc. Int. Conf. Power Syst. Technol.*, Nov. 2004, pp. 1633–1637.

[158] S. Baskar and P. N. Suganthan, "A novel concurrent particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2004, vol. 1, pp. 792–796.

[159] M. El-Abd and M. S. Kamel, "A hierarchal cooperative particle swarm optimizer," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 43–47.

[160] J. Liang, A. Qin, P. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, Jun. 2006.

[161] M. R. AlRashidi and M. E. El-Hawary, "A survey of particle swarm optimization applications in electric power operations," *Electric Power Components and Systems*, vol. 34, no. 12, pp. 1349–1357, Dec. 2006.

[162] T. Cutsem and T. Vournas, *Voltage Stability of Electric Power Systems*. New York: Springer, 2005.

[163] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment," *IEEE Trans. Power Syst.*, vol. 15, no. 4, pp. 1232–1239, Nov. 2000.

[164] G. Coath, M. Al-Dabbagh, and S. Halgamuge, "Particle swarm optimization for reactive power and voltage control with grid-integrated wind farms," in *Proc. IEEE Power Eng. Soc. General Meeting*, Jun. 2004, vol. 1, pp. 303–308.

[165] B. Zhao, C. Guo, and Y. J. Cao, "A multiagent-based particle swarm optimization approach for optimal reactive power dispatch," *IEEE Trans. Power Syst.*, pp. 1070–1078, May 2005.

[166] W. Zhang and Y. Liu, "Reactive power optimization based on PSO in a practical power system," in *Proc. IEEE Power Eng. Soc. General Meeting*, Jun. 2004, vol. 1, pp. 239–243.

[167] A. Mantawy and M. Al-Ghamdi, "A new reactive power optimization algorithm," in *Proc. IEEE Power Technol. Conf.*, Jun. 2003.

[168] N. Krami, M. El-Sharkawi, and M. Akherraz, "Multi objective particle swarm optimization technique for reactive power planning," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 170–174.

[169] J. Park, K. Lee, J. Shin, and K. Y. Lee, "A particle swarm optimization for economic dispatch with nonsmooth cost functions," *IEEE Trans. Power Syst.*, vol. 20, no. 1, pp. 34–42, Feb. 2005.

[170] K. Y. Lee and J. Park, "Application of particle swarm optimization to economic dispatch problem: Advantages and disadvantage," in *Proc. IEEE PES Power Syst. Conf. Expo.*, Oct. 2006, pp. 188–192.

[171] T. Aruldoss, A. Victoire, and A. Jeyakumar, "Reserve constrained dynamic dispatch of units with valve-point effects," *IEEE Trans. Power Syst.*, vol. 20, no. 3, pp. 1273–1282, Aug. 2005.

[172] L. Lai, T. Nieh, Y. Ma, Y. Lu, Y. Yang, and H. Braun, "Particle swarm optimization for economic dispatch of units with non-smooth input-output characteristic functions," in *Proc. IEEE Intell. Syst. Appl. Power Conf.*, Nov. 2005, pp. 499–503.

[173] Z. Gaing, "Particle swarm optimization to solving the economic dispatch considering the generator constraints," *IEEE Trans. Power Syst.*, vol. 18, no. 3, pp. 1187–1195, Aug. 2003.

[174] N. Sinha and B. Purkayastha, "PSO embedded evolutionary programming technique for non-convex economic load dispatch," in *Proc. IEEE-PES Power Syst. Conf. Exposition*, Oct. 2004, vol. 1, pp. 66–71.

[175] R. Pancholi and K. Swarup, "Particle swarm optimization for security constrained economic dispatch," in *Proc. Int. Conf. Intell. Sensing Inf. Process.*, Jan. 2004, pp. 7–12.

[176] B. Zhao, C. Guo, and Y. Cao, "Dynamic economic dispatch in electricity market using particle swarm optimization algorithm," in *Proc. IEEE 5th World Congr. Intell. Control Autom.*, Jun. 2004, pp. 5050–5054.

[177] A. El-Gallad, M. El-Hawary, A. Sallam, and A. Kalas, "Particle swarm optimizer for constrained economic dispatch with prohibited operating zones," in *Proc. 2002 IEEE Canadian Conf. Elect. Comput. Eng.*, May 2002, vol. 1, pp. 78–81.

[178] A. Kumar, K. Dhanushkodi, J. Kumar, and C. Paul, "Particle swarm optimization solution to emission and economic dispatch problem," in *Conf. Convergent Technologies for Asia-Pacific Region (TENCON-2003)*, Oct. 2003, vol. 1, pp. 435–439.

[179] L. Wang and C. Singh, "Multi-objective stochastic power dispatch through a modified particle swarm optimization algorithm," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 128–135.

[180] J. Meeuwsen, *Reliability Evaluation of Electric Transmission and Distribution Systems*. The Netherlands: Ponsen & Looijen BV, 1998.

[181] D. Robinson, "Reliability analysis of bulk power systems using swarm intelligence," in *Proc. Annu. Reliability and Maintainability Symp.*, Jan. 2005, pp. 96–102.

[182] R. Chang and C. Lu, "Feeder reconfiguration for load factor improvement," in *Proc. IEEE Power Eng. Soc. Winter Meeting*, Oct. 2002, vol. 2, pp. 980–984.

[183] C. Shen and C. Lu, "Feeder reconfiguration for power quality requirement and feeder service quality matching," in *Proc. IEEE Transmission and Distrib. Conf. Exhibition 2002: Asia Pacific*, Oct. 2002, vol. 1, pp. 226–231.

[184] X. Jin, J. Zhao, K. Li, and B. Zhang, "Distribution network reconfiguration for load balancing using binary particle swarm optimization," in *Proc. Int. Conf. Power Syst. Technol. (POWERCON)*, Nov. 2004, pp. 507–510.

[185] W. Kurutach and Y. Tuppadung, "Feeder-switch relocation based upon risk analysis of tree-caused interruption and value-based distribution reliability assessment," in *Proc. 2004 IEEE Region 10 Conf.*, Nov. 2004, vol. 3, pp. 577–580.

[186] W. Kurutach and Y. Tuppadung, "Feeder-switch relocation for value-based distribution reliability assessment," in *Proc. 2004 IEEE Int. Eng. Manage. Conf.: Innovation and Entrepreneurship for Sustainable Development, IEMC 2004*, Oct. 2004, pp. 402–406.

[187] Kassabalidis, M. El-Sharkawi, R. Marks, L. Moulin, and A. Silva, "Dynamic security border identification using enhanced particle swarm optimization," *IEEE Trans. Power Syst.*, pp. 723–729, Aug. 2002.

[188] B. Mozafari, T. Amraee, and A. Ranjbar, "An approach for under voltage load shedding using particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 217–222.

[189] F. Wu, Z. Yen, Y. Hou, and Y. Ni, "Applications of AI techniques to generation planning and investment," in *Proc. IEEE Power Eng. Soc. General Meeting*, Jun. 2004, vol. 1, pp. 936–940.

[190] S. Kannan, S. Slochanal, and N. Padhy, "Application and comparison of metaheuristic techniques to generation expansion planning problem," *IEEE Trans. Power Syst.*, vol. 20, no. 1, pp. 466–475, Feb. 2005.

[191] P. Sensarna, M. Rahmani, and A. Carvalho, "A comprehensive method for optimal expansion planning using particle swarm optimization," in *Proc. IEEE Power Eng. Soc. Winter Meeting 2002*, Jan. 2002, vol. 2, pp. 1317–1322.

[192] S. Slochanal, S. Kannan, and R. Rengaraj, "Generation expansion planning in the competitive environment," in *Proc. 2004 Int. Conf. Power Syst. Technol.*, Nov. 2004, pp. 1546–1549.

[193] Y. Fukuyama, "State estimation and optimal setting of voltage regulator in distribution systems," in *Proc. IEEE Power Eng. Soc. Winter Meeting 2001*, Jan. 2001, vol. 2, pp. 930–935.

[194] P. Kundur, J. Paserba, V. Ajjarapu, G. Andersson, A. Bose, C. Canizares, N. Hatziargyriou, D. Hill, A. Stankovic, C. Taylor, T. Van Cutsem, and V. Vittal, "Definition and classification of power system stability," *IEEE Trans. Power Syst.*, vol. 19, no. 3, pp. 1387–1401, Aug. 2004.

[195] M. Abido, "Optimal design of power-system stabilizers using particle swarm optimization," *IEEE Trans. Energy Conversion*, vol. 17, no. 3, pp. 406–413, Sep. 2002.

[196] T. Das and G. Venayagamoorthy, "Bio-inspired algorithms for the design of multiple optimal power system stabilizers: SPPSO and BFA," in *Proc. IEEE Industry Appl. Soc. 41st Annu. Meeting*, Tampa, FL, Oct. 2006.

[197] A. Karimi, A. Al-Hinai, K. Schoder, and A. Feliachi, "Power system stability enhancement using backstepping controller tuned by particle swarm optimization technique," in *Proc. IEEE Power Eng. Soc. General Meeting (PES)*, Jun. 2005, pp. 2955–2962.

[198] T. Okada, T. Watanabe, and K. Yasuda, "Parameter tuning of fixed structure controller for power system stability enhancement," in *Proc. IEEE-PES Transmission and Distrib. Conf. Exhibition 2002: Asia Pacific*, Oct. 2002, vol. 1, pp. 162–167.

[199] Y. Abdel-Magid and M. A. Abido, "AGC tuning of interconnected reheat thermal systems with particle swarm optimization," in *Proc. IEEE Int. Conf. Electron., Circuits, Syst.*, Dec. 2003, vol. 1, pp. 376–379.

[200] Z. Gaing, "A particle swarm optimization approach for optimum design of PID controller in AVR system," *IEEE Trans. Energy Conversion*, vol. 19, no. 2, pp. 384–391, Jun. 2004.

[201] G. Venayagamoorthy, "Optimal control parameters for a UPFC in a multimachine using PSO," in *Proc. Int. Conf. Intell. Syst. Appl. Power Syst.*, Nov. 2005, pp. 488–493.

[202] T. Das, S. Jetti, and G. Venayagamoorthy, "Optimal design of SVC damping controllers with wide area measurements using small population based PSO," in *Proc. IEEE Int. Joint. Conf. Neural Netw.*, Jul. 2006, pp. 2255–2260.

[203] W. Qiao, G. K. Venayagamoorthy, and R. Harley, "Design of optimal PI controllers for doubly fed induction generators driven by wind turbines using particle swarm optimization," in *Proc. IEEE World Congr. Comput. Intell.*, Jul. 2006.

[204] W. Qiao, R. Harley, and G. Venayagamoorthy, "A fault-tolerant P-Q decoupled control scheme for static synchronous series compensator," in *Proc. IEEE Power Eng. Soc. General Meeting*, Montreal, Canada, Jun. 2006.

[205] G. Gudise and G. Venayagamoorthy, "Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks," in *Proc. IEEE Swarm Intell. Symp.*, Apr. 2003, pp. 110–117.

[206] J. Chia-Feng, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Trans. Syst., Man, Cybern., Part B: Cybern.*, vol. 34, no. 2, pp. 997–1006, Apr. 2004.

[207] R. Kiran, S. R. Jetti, and G. K. Venayagamoorthy, "Offline training of a generalized neuron using particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 111–117.

[208] S. Jetti and G. Venayagamoorthy, "Real-time implementation of a dual function neuron based wide area SVC damping controller," in *Proc. IEEE Industry Appl. Soc. 41st Annu. Meeting*, Oct. 2006.

[209] P. Reynolds, R. Duren, M. Trumbo, and R. Marks, "FPGA implementation of particle swarm optimization for inversion of large neural network," in *Proc. IEEE Swarm Intell. Symp.*, Jun. 2005, pp. 389–392.

[210] N. Hirata, A. Ishigame, and H. Nishigaito, "Neuro stabilizing control based on Lyapunov method for power system," in *Proc. 41st SICE Annu. Conf.*, Aug. 2002, vol. 5, pp. 3169–3171.

[211] A. Elwer, S. Wahsh, M. Khalil, and A. Nur-Eldeen, "Intelligent fuzzy controller using particle swarm optimization for control of permanent magnet synchronous motor for electric vehicle," in *Proc. 29th Annu. Conf. IEEE Ind. Electron. Soc.*, Nov. 2003, vol. 2, pp. 1762–1766.

[212] C. Juang and C. Lu, "Power system load frequency control by evolutionary fuzzy PI controller," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Jul. 2004, vol. 2, pp. 715–719.

[213] R. Welch and G. Venayagamoorthy, "Comparison of two optimal control strategies for a grid independent photovoltaic system," in *Proc. IEEE Ind. Appl. Soc. 41st Annu. Meeting*, Oct. 2006.

[214] R. Welch and G. Venayagamoorthy, "A Fuzzy-PSO based controller for a grid independent photovoltaic system," in *Proc. IEEE Swarm Intell. Symp.*, Apr. 2007.

[215] C. Juang and C. Lu, "Evolutionary fuzzy control of flexible AC transmission system, generation," *Proc. Inst. Elect. Eng. Transmission Distribution*, vol. 152, no. 4, pp. 441–448, Jul. 2005.

[216] Y. Liu, X. Zhu, J. Zhang, and S. Wang, "Application of particle swarm optimization algorithm for weighted fuzzy rule-based system," in *Proc. 30th Annu. Conf. IEEE Ind. Electron. Soc.*, Nov. 2004, vol. 3, pp. 2188–2191.

[217] H. Emara, M. Ammar, A. Bahgat, and H. Dorrah, "Stator fault estimation in induction motors using particle swarm optimization," in *Proc. IEEE Conf. Electric Machines and Drives*, Jun. 2003, vol. 3, pp. 1469–1475.

[218] G. Bao, D. Zhang, J. Shi, and J. Jiang, "Optimal design for cogging torque reduction of transverse flux permanent motor using particle swarm optimization algorithm," in *Proc. IEEE Power Electron. Motion Control Conf.*, Aug. 2004, vol. 1, pp. 260–263.

[219] T. Gönen, *Electrical Power Distribution System Engineering*. New York: McGraw-Hill, 1986.

[220] X. Yu, X. Xiong, and Y. Wu, "A PSO-based approach to optimal capacitor placement with harmonic distortion consideration," *Electric Power Syst. Res.*, vol. 71, pp. 27–33, Sep. 2004.

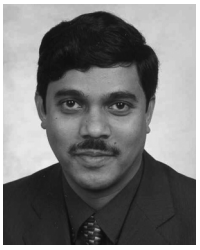
[221] J. Hernandez, Y. del Valle, G. Venayagamoorthy, and R. G. Harley, "Optimal allocation of a STATCOM in a 45 bus section of the Brazilian power system using particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 69–76.

[222] Y. del Valle, J. Hernandez, G. Venayagamoorthy, and R. Harley, "Optimal STATCOM sizing and placement using particle swarm optimization," in *Proc. IEEE PES Transmission and Distrib. Conf. Exposition, Latin America*, Aug. 2006.

[223] C. Koay and D. Srinivasan, "Particle swarm optimization-based approach for generator maintenance scheduling," in *Proc. IEEE Swarm Intell. Symp.*, Apr. 2003, pp. 167–173.

[224] M. Miranda and N. Win-Oo, "New experiments with EPSO—Evolutionary particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, May 2006, pp. 162–169.
[225] C. Huang, C. J. Huang, and M. Wang, "A particle swarm optimization to identifying the ARMAX model for short-term load forecasting," *IEEE Trans. Power Syst.*, vol. 20, no. 2, pp. 1126–1133, May 2005.
[226] J. Vlachogiannis and K. Lee, "Determining generator contributions to transmission system using parallel vector evaluated particle swarm optimization," *IEEE Trans. Power Syst.*, vol. 20, no. 4, pp. 1765–1774, Nov. 2005.
[227] N. Bianchi and S. Bolognani, "Design optimization of electric motors by genetic algorithm," *Proc. Inst. Elect. Eng. Power Appl.*, vol. 145, no. 5, pp. 475–483, Sep. 1998.
[228] I. Kamel, C. Liuchen, and D. Heping, "A neural network-based optimization approach for induction motor design," in *Proc. IEEE Canadian Conf. Elect. Comput. Eng.*, May 1996, vol. 2, pp. 951–954.
[229] X. Alabern and E. Valero, "Optimized design of an induction motor using Fuzzy logic as a calculus and decision tool," in *Proc. IEEE Electric Machines and Drives Int. Conf.*, Jun. 2003, vol. 1, pp. 83–87.
[230] R. Kiran, S. Jetti, and G. Venayagamoorthy, "Online training of a generalized neuron using particle swarm optimization," in *Proc. IEEE Int. Joint. Conf. Neural Netw.*, Jul. 2006, pp. 5088–5095.

Yamille del Valle (S'06) received the B.S. degree in civil and industrial engineering from the Universidad Católica de Chile, Chile, in 2001 and the M.S. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, in 2005. She is currently working towards the Ph.D. degree in applications of evolutionary computation techniques to power systems at Georgia Institute of Technology.



Ganesh Kumar Venayagamoorthy (S'91–M'97–SM'02) received the B.Eng. degree (with first class honors) in electrical and electronics engineering from Abubakar Tafawa Balewa University, Bauchi, Nigeria, in 1994 and the M.Sc.Eng. and Ph.D. degrees in electrical engineering from the University of Natal, Durban, South Africa, in 1999 and 2002, respectively.

He was a Senior Lecturer with the Durban Institute of Technology, South Africa, prior to joining the University of Missouri–Rolla (UMR) in May 2002. He is currently an Associate Professor of Electrical and Computer Engineering and the Director of the Real-Time Power and Intelligent Systems Laboratory at UMR. He has attracted over 3.5 million U.S. dollars in research funding to date. He has published 2 edited books, 3 book chapters, 45 refereed journals papers, and 190 refereed international conference proceeding papers. His research interests are in the development and applications of computational intelligence for power systems stability and control, FACTS devices, alternative sources of energy, sensor networks, and adaptive circuits and systems.

Dr. Venayagamoorthy was a recipient of a 2007 Office of Naval Research Young Investigator Award, a 2004 NSF CAREER Award, the 2006 IEEE Power Engineering Society Walter Fee Outstanding Young Engineer Award, the 2006 IEEE St. Louis Section Outstanding Section Member Award, the 2005 IEEE Industry Applications Society (IAS) Outstanding Young Member Award, the 2005 SAIEE Young Achievers Award, the 2004 IEEE St. Louis Section Outstanding Young Engineer Award, the 2003 International Neural Network Society (INNS) Young Investigator Award, the 2001 IEEE CIS Walter Karplus Summer Research Award, five prize papers from the IEEE IAS and IEEE CIS, a 2006 UMR School of Engineering Teaching Excellence Award, and a 2005 UMR Faculty Excellence Award. He is listed in the 2007 and 2008 editions of *Who's Who in America*, 2008 edition of *Who's Who in Science and Engineering*, and 2008 edition of *Who's Who in the World*. He is a Senior Member of the South African Institute of Electrical Engineers (SAIEE). He is also a member of the INNS, The Institution of Engineering and Technology, U.K., and the American Society for Engineering Education. He is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS and the IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT. He is currently the IEEE St. Louis Computational Intelligence Society (CIS) and IAS Chapter Chairs, the Chair of the Task Force on Intelligent Control Systems, the Secretary of the Intelligent Systems subcommittee and the Vice-Chair of the Student Meeting Activities subcommittee of the IEEE Power Engineering Society, and the Chair of the IEEE CIS Task Force on Power System Applications. He has organized and chaired several panels, invited, and regular sessions, and tutorials at international conferences and workshops.

Salman Mohagheghi (S'99) received the B.Eng. degree from the University of Tehran, Tehran, Iran, in 1999, the M.Sc. degree from the Sharif University of Technology, Tehran, in 2001, both in power electrical engineering, and the Ph.D. degree in electrical engineering from the Georgia Institute of Technology, Atlanta, in 2006.

He is currently doing research as a Postdoctoral Fellow at the Georgia Institute of Technology. His research focuses on wide area control in power systems, protective relaying, and distributed state estimation.

Jean-Carlos Hernandez (S'05) received the B.S. degree in electrical engineering from the Universidad de Los Andes, Los Andes, Venezuela, in 2000 and the M.S. degree in electrical and computer engineering (ECE) from the Georgia Institute of Technology, Atlanta, in 2005. He is currently working towards the Ph.D. degree researching defect characterization and cable diagnostics at the Georgia Institute of Technology.

Ronald G. Harley (M'77–SM'86–F'92) received the MSc.Eng. degree (*cum laude*) in electrical engineering from the University of Pretoria, Pretoria, South Africa, in 1965, and the Ph.D. degree from London University, London, U.K., in 1969.

In 1971, he was appointed to the Chair of Electrical Machines and Power Systems at the University of Natal, Durban, South Africa. At the University of Natal, he was a Professor of Electrical Engineering for many years, including the Department Head and Deputy Dean of Engineering. He is currently the Duke Power Company Distinguished Professor at the Georgia Institute of Technology, Atlanta. His research interests include the dynamic behavior and condition monitoring of electric machines, motor drives, power systems and their components, and controlling them by the use of power electronics and intelligent control algorithms. He has coauthored some 380 papers in refereed journals and international conferences and has three patents.

Dr. Harley received the Cyrill Veinott Award in 2005 from the Power Engineering Society for "Outstanding Contributions to the Field of Electromechanical Energy Conversion." He attracted prizes from journals and conferences for ten of his papers. He is a Fellow of the British IEE. He is also a Fellow of the Royal Society in South Africa, and a Founder Member of the Academy of Science in South Africa formed in 1994. During 2000 and 2001, he was one of the IEEE Industry Applications Society's six Distinguished Lecturers. He was the Vice-President of Operations of the IEEE Power Electronics Society (2003–2004) and Chair of the Atlanta Chapter of the IEEE Power Engineering Society. He is currently Chair of the Distinguished Lecturers and Regional Speakers program of the IEEE Industry Applications Society.