International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161

Available at http://inpressco.com/category/ijcet

Research Article

Load Flow Solution and Active Power Loss Minimization using a New PSO Technique

Pruthviraja L[Á] and Ramesh G B[B*]

[Á]Dept. of E&E, Jain Institute of Technology, Davangere, Karnataka-577004
[B]Dept. of E&E, Gogte Institute of Technology, Belagavi, Karnataka-590008

Accepted 15 Nov 2014, Available online 20 Dec2014, **Vol.4, No.6 (Dec 2014)**

Abstract

Since many years power system field existed with solving of one or more non-linear optimization problems. The existing analytical methods suffer from slow convergence, especially with increase in system size and non linearity's. Particle swarm optimization (PSO) is known to solve large-scale nonlinear optimization problems effectively. This paper presents an overview of PSO and its applications to load flow analysis to minimize the active and reactive power loss with increased speed of convergence and more acceptable results. Several particles in feasible solutions were used to lead swarm's motion and update. The present new PSO algorithm requires the less memory space required to establish the global optimal solution. Simulation results of standard IEEE 30-bus system have shown that this algorithm can improve the searching of best particles and voltage stability, with a fast convergence. It was proved to be efficient and practical during the active and reactive power loss minimization.

Keywords: *Local best, Global best, Power Loss minimization, Evolutionary computation, Particle Swarm Optimization (PSO)*

1. Introduction

Optimizationis a scientific discipline that deals with the detection of optimal solutions for a problem, among alternatives. The optimality of solutions is based on one or several criteria that are usually problem and user-dependent. If a solution fulfils all constraints, it is called a *feasible solution*. Among all feasible solutions, the *global optimization* problem concerns the detection of the optimal one. However, this is not always possible or necessary. Indeed, there are cases where suboptimal solutions are acceptable, depending on their quality compared to the optimal one. This is usually described as *localOptimization*, although the same term has been also used to describe local search in a strict vicinity of the search space. In order to minimize an objective function, Real-world optimization suffers from the following problems:

a. Difficulties in distinguishing global from local optimal solutions.
b. Presence of noise in solution evaluation.
c. The "curse of dimensionality", i.e., exponential growth of the search space with the problem's dimension.
d. Difficulties associated with constraints.

Some of the important optimization techniques are

i).Linear optimization (or linear programming): In cases where the objective function and constraints are linear.

*ii).Nonlinear optimization (or nonlinear programming):*It deals with cases where at least one nonlinear function is involved in the optimization problem.

*iii).Convex optimization:*It studies problems with convex objective functions and convex feasible sets.

*iv).Quadratic optimization (or quadratic programming):*It involves the minimization of quadratic objective functions and linear constraints.

*v). Stochastic optimization:*It refers to minimization in the presence of randomness, which is introduced either as noise in function evaluations or as probabilistic selection of problem variables and parameters, based on statistical distributions.

Optimization algorithms are available for both Deterministic and Stochastic problem formulation. Evolutionary algorithms combine elements such as stochasticity, adaptation and learning, in order to produce *intelligence optimization* schemes. All Evolutionary optimization problems come under stochastic nature. Particle Swarm Optimization (PSO) is a *Stochastic Optimization*. ***Particle swarm optimization*** was developed by Kennedy and Eberhart (1995) as a stochastic optimization algorithm based on social simulation models (Konstantinos E et al). The algorithm employs a population of search points that moves stochastically in the search space. Concurrently, the best position ever attained by each individual, also called its *experience*, is retained in memory. This experience is then communicated to part

*Corresponding author: **Ramesh G B**

or the whole population, biasing its movement towards the most promising regions detected so far. The communication scheme is determined by a fixed or adaptive social network that plays a crucial role on the convergence properties of the algorithm. The development of particle swarm optimization was based on concepts and rules that govern socially organized populations in nature, such as bird flocks, fish schools, and animal herds. Unlike the ant colony approach, where stigmergy is the main communication mechanism among individuals through their environment, in such systems communication is rather direct without altering the environment.

The first version of PSO was intended to handle only nonlinear continuous optimization problems. However, many advances in PSO development elevated its capabilities to handle a wide class of complex engineering and science optimization problems. Different variants of the PSO algorithm were proposed but the most standard one is the global version of PSO (Gbest model) introduced by Shi and Eberhart (Konstantinos E et al), where thewhole populations considered as a single neighbourhood throughout the optimization process. A key attractive feature of the PSO approach is its simplicity as it involves only two model equations. In PSO, the Coordinates of each particle represent a possible solution associated with two vectors, the position (x_i) and velocity (v_i) vectors. In N-dimensional search space, $X_i = [x_{i1}, x_{i2}, x_{i3,}....,x_{iN}]$ and $V_i = [v_{i1}, v_{i2}, v_{i3,} \ldots, v_{iN}]$ are the two vectors associated with each particle *i* .

2. Load Flow Review

The primary function of an Electric Power System is to supply the power demand in an efficient, economic, high quality and reliable way. The power system can operate in an infinite number of states – voltage and power sets in the buses – in order to comply with standard requisites. The solution of a Power Flow problem consists in the determination of these possible operational states through the knowledge *a priori* of certain variables of the system buses. The objective of this kind of problem is to obtain the system buses voltages magnitudes and angle – in order to determine later the power adjustments in the generation buses and the power flow in the system lines. The power flow study provides the system status in the steady-state. Once the steady-state of the system known, it is possible to estimate the amount of power generation necessary to supply the power demand plus the power losses in the system lines, moreover the voltage levels must be kept within the boundaries and overloaded operations, besides the operations in the stability limit must be avoided. The general form of the Static Load Flow Equations (SLFE) is given by equation (1):

$$\mathbf{P_i\text{-}jQ_i\text{-}yi_1V_1(V_i^*)\text{-}y_{i2}V_2(V_i^*)\text{-}\ldots..\text{-}y_inV_n\,(V_i^*)=0} \quad (a)$$

Where: i = 1,..., n, bus number; Pi = active power generated or injected in the bus i; Q_i= reactive power generated or injected in the bus i; $|V_i|$ = voltage module of the bus i; δ_i= voltage angle of the bus i; $V_i = |V_i|e^{j\delta i}$, i. e., the voltage in the polar form; y_{ik}= element of the nodal admittance matrix Ybus. The nodal admittance matrix is obtained through the following explanation: if i = k, y_{ik} is the sum of the admittances that come out of the bus *i*; and if i ≠ k, yik is the admittance between the buses *i* and *k*, multiplied by -1. The power system buses are classified in types, according to the variables known *a priori* and to the variables that will be obtained through the SLFE(Xiuhua Wu,Zailion Piao et al, 2010).

- Type 1 Bus OR PQ Bus: *Pi* &*Qi* are specified and |*Vi*| and δ*i* are obtained through the SLFE;
- Type 2 Bus OR PV Bus: *Pi* &|*Vi*| are specified and *Qi* and δ*i* are obtained through the SLFE;
- Type 3 Bus OR Vδ Bus ("Slack Bus"): |*Vi*| and δ*i* are specified and *Pi* and *Qi* are obtained through the SLFE. Equation (1) performs a complex and non-linear equations system, and its solution is obtained through approximations using numeric computational methods.

3. Particle Swarm Optimization (PSO)

The original precursors of PSO were simulators of social behaviour for visualizing bird flocks. Nearest neighbour velocity matching and acceleration by distance were the main rules employed to produce swarming behaviour by simple agents in their search for food in simulation experiments conducted by Russell C. Eberhart (Purdue School of Engineering and Technology, Indiana University Purdue University Indianapolis) and James Kennedy (Bureau of Labor Statistics, Washington, DC). After realizing the potential of these simulation models to perform optimization, Eberhart and Kennedy refined their model and published the first version of PSO in 1995 (Eberhart & Kennedy, 1995; Kennedy & Eberhart, 1995).
Putting it in a mathematical framework, let, $A \subset \mathbf{R}n$, be the search space, and, $f{:}A \rightarrow Y \subseteq \mathbf{R}$, be the objective function. In order to keep descriptions as simple as possible, we assume that A is also the feasible space of the problem at hand, i.e., there are no further explicit constraints posed on the candidate solutions. Also, note that no additional assumptions are required regarding the form of the objective function and search space. PSO is a population-based algorithm, i.e., it exploits a population of potential solutions to probe the search space concurrently. The population is called the *swarm* and its individuals are called the *particles*; a notation retained by nomenclature used for similar models in social sciences and particle physics. The swarm is defined as a set:

$\boldsymbol{S} = \{x1, x2, \ldots, xN\}$,
of N particles (candidate solutions), defined as:
$xi = (xi1, xi2, \ldots, xin)\mathrm{T} \in A$, $i = 1, 2, \ldots, N$.
Indices are arbitrarily assigned to particles, while N is a user-defined parameter of the algorithm. The objective function, $f(x)$, is assumed to be available for all points in A. Thus, each particle has a unique
function value, $fi = f(xi) \in Y$. The particles are assumed to move within the search space, A, iteratively. This is possible by adjustingtheir *position* using a proper position shift, called *velocity*, and denoted as:
$vi = (vi1, vi2, \ldots, vin)\mathrm{T}$, $i = 1, 2, \ldots, N$.

Velocity is also adapted iteratively to render particles capable of potentially visiting any region of *A*. If *t* denotes the iteration counter, then the current position of the *i*-th particle and its velocity will behenceforth denoted as *xi*(*t*) and *vi*(*t*), respectively. Velocity is updated based on information obtained in previous steps of the algorithm. This is implemented in terms of a memory, where each particle can store the *best position* it has ever visited during its search. For this purpose, besides the swarm, ***S***, which contains the current positions of the particles,
PSO maintains also a *memory* set:

P = {*p*1, *p*2,…, *pN*},
which contains the best positions:
$pi = (pi1, pi2, \ldots, pin)^T \in A$, $i = 1, 2, \ldots, N$,
ever visited by each particle.
These positions are defined as:
$p_i(t) = \text{argmin } f_i(t)$,
where *t* stands for the iteration counter.

PSO is based on simulation models of social behaviour; thus, an information exchange mechanism shall exist to allow particles to mutually communicate their experience. The algorithm approximates the global minimiser with the best position ever visited by all particles. Therefore, it is a reasonable choice to share this crucial information. Let *g* be the index of the best position with the lowest function value in ***P*** at a given iteration *t*, i.e.

$$\mathbf{p_g(t) = \text{argmin} f(p_i(t))}$$

Then, the early version of PSO is defined by the following equations (Eberhart & Kennedy, 1995; Eberhart 1996; Kennedy & Eberhart, 1995):

$$\mathbf{v_{ij}(t+1) = v_{ij}(t) + c1R1\,(p_{ij}(t)-x_{ij}(t)) + c2R2\,(p_{gj}(t)-x_{ij}(t))} \quad (1)$$
$$\mathbf{x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1)} \quad (2)$$
$i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, n$,

Where *t* denotes the iteration counter; *R*1 and *R*2 are random variables uniformly distributed within [0,1];
and *c*1, *c*2, are weighting factors, also called the *cognitive* and *social* parameter, respectively. In the first version of PSO, a single weight, $c = c1 = c2$, called *acceleration constant*, was used instead of the two distinct weights in equation (1). However, the latter offered better control on the algorithm, leading to its predominance over the first version. At each iteration, after the update and evaluation of particles, best positions (memory) are also updated. Thus, the new best position of *xi* at iteration *t*+1 is defined as follows:

$$\mathbf{p_i(t+1)=\{x_i(t+1),\ If\ } f(\mathbf{x_i(t+1)\le} f(\mathbf{p_i(t)),}$$
$$\mathbf{p_i(t+1)=\{p_i(t),}\ \text{otherwise}$$

The new determination of index *g* for the updated best positions completes an iteration of PSO.
The velocity clamping can also be used to control the velocity of different swarms, within their bounds as follows:
Maximum velocity for all direction components is given by

$$\mathbf{v_{max}=\min_i\{b_i-a_i\}/k}$$

where *a* and *b* are the lower and upper limits of the swarms.

Alternatively, separate maximum velocity thresholds per component can be used as

$$\mathbf{v_{max,i}=(b_i-a_i)/k,\ i=1,2,\ldots,n}$$

with *k*=user defined value, being a common choice. If the problem at hand requires smaller particles steps then higher value of *k* shall be used. The search of best position requires a strong attraction of particles towards their bounds, so that the new parameter called *inertia weight* is used , the velocity now became as shown below:

$$\mathbf{v_{ij}(t+1) = w(t)v_{ij}(t) + c1R1(p_{ij}(t)-x_{ij}(t)) + c2R2\,(p_{gj}(t)-x_{ij}(t))} \quad (3)$$

In general, a linearly decreasing scheme for *w* can be mathematically described as follows:

$$\mathbf{w(t)=w_{up}-(w_{up}-w_{low})*\{t/T_{max}\}} \quad (4)$$

where *t* stands for the iteration counter; w_{low} and w_{up} are the desirable lower and upper bounds of *w*; and $T_{\max}$ is the total allowed number of iterations. Equation (4) produces a linearly decreasing time-dependent inertia weight with starting value, w_{up}, at iteration, $t = 0$, and final value, w_{low}, at the last iteration, $t = T\text{max}$.

The basic PSO pseudocode for random uniform initialization.

Input: Number of particles *N*, dimension *n*, velocity bounds [-*v*max, *v*max], and search space, $A = [a1, b1] \times [a2, b2] \times \ldots \times [an, bn]$

Step 1. **Do** ($i = 1 \ldots N$)
Step 2. **Do** ($j = 1 \ldots n$)
Step 3. **Set** particle component $xij = a_j + \text{rand}(\,) * (b_j - a_j)$.
Step 4. **Set** best position component $p_{ij} = x_{ij}$.
Step 5. **Set** velocity component $v_{ij} = -v\text{max} + 2\ \text{rand}(\,) * v\text{max}$.
Step 6. **End Do**
Step 7. **End Do**

Where a_i=lower limits of swarms, b_i= upper limits of swarms.

If there are items of information available regarding the location of the global minimizer in the search space, it makes more sense to initialize the majority of the swarm around it. This can be done by replacing the uniform distribution with a Gaussian one (Konstantinos E et al).

A. PSO Methodology and its Pseudocode

There are several PSO techniques, which are getting more and more advanced in recent decade. Among those PSO techniques here we would like mention few as (A.H Mantawy et al, 2003):

- *Unified PSO*,
- *Memetic PSO*,
- *Composite PSO*,
- *Vector Evaluated PSO*,
- *Guaranteed Convergence PSO*,
- *Cooperative PSO*,
- *Niching PSO*, *TRIBES*, And
- *Quantum PSO*. Albeit possibly omitting an interesting approach, the aforementioned variants sketch a rough picture of the current status in PSO literature, exposing the main ideas and features that constitute the core of research nowadays.

In this paper we have explored the first two methods to illustrate the application of PSO to load flow computation and to minimize the active and reactive power loss. Here we mainly focus on memetic algorithm with Random walk with Directional Exploitation Local search.

Memetic PSO (MPSO) is a hybrid algorithm that combines PSO with local search techniques. Memetic algorithms (MAs) were first proposed by Moscato (1989), MPSO consists of two main components: a global one, which is responsible for global search, and a local one, which performs more refined search around roughly detected solutions. Although MAs bear a similarity with genetic algorithms (GAs) (Goldberg, 1989), they mimic cultural rather than biological evolution. Indeed, GAs employ a combination of selection, recombination, and mutation, similar to that applied to genes in natural organisms. However, genes are usually not modified during a lifetime, whereas memes are. Therefore, most MAs can be interpreted rather as a cooperative/competitive algorithm of optimizing agents. MPSO combines PSO with a local search algorithm. Besides the selection of the most appropriate local search method, three major questions arise spontaneously. Henceforth, we will call these questions as *fundamental memetic questions* (FMQs):

(FMQ 1) When local search has to be applied?
(FMQ 2) Where local search has to be applied?
(FMQ3) What computational budget shall be accredited to the local search algorithm?

In the following schemes regarding the point of application of the local search (which is related to FMQ 2) were proposed: (Scheme 1) Local search is applied on the overall best position, p_g, of the swarm, where g is the index of the best particle.

(Scheme 2) For each best position, p_i, $i = 1, 2,\ldots, N$, a random number, $r \in [0,1]$, is generated and, if $r < \varepsilon$ for a prescribed threshold, $\varepsilon > 0$, local search is applied on p_i.
(Scheme 3) Local search is applied on the overall best position, p_g, as well as on some randomly selected best positions, p_i, $i \in \{1, 2,\ldots, N\}$.
(Scheme 4) Local search is applied on the overall best position, *pg*, as well as on the best positions, *pi*, $i \in \{1, 2,\ldots, N\}$,

A pseudocode for the MPSO algorithm is reported below. In addition, a proof of convergence in probability was derived in for the approach with the random walk with directional exploitation (RWDE) local search algorithm:

Input: N, c1, c2, x_{min}, x_{max} (lower and upper bounds),
f(x) (objective function), N=population size.
Step 1. **Set** $t \leftarrow 0$.
Step 2. **Initialize** $xi(t)$, $vi(t)$, $pi(t)$, $i = 1, 2,\ldots, N$.
Step 3. **Evaluate** $f(xi(t))$, $i = 1, 2,\ldots, N$.
Step 4. **Update** indices, g_i, of best particles.
Step 5. **While** (stopping condition not met)
Step 6. **Update** velocities, $v_i(t+1)$, and particles, $xi(t+1)$, $i = 1, 2,\ldots, N$.
Step 7. **Constrain** particles within bounds $[x_{min}, x_{max}]$.
Step 8. **Evaluate** $f(xi(t+1))$, $i = 1, 2,\ldots, N$.
Step 9. **Update** best positions, $pi(t+1)$, and indices, gi.
Step 10. **If** (local search is applied) **Then**
Step11. **Choose** a position $p_q(t+1)$, $q \in \{1, 2,\ldots, N\}$, according to Schemes 1-4.
Step12. **Apply** local search on $p_q(t+1)$ and obtain a solution, y.
Step 13. **If** $(f(y) < f(pq(t+1))$ **Then** $pq(t+1) \leftarrow y$.
Step 14. **End If**
Step 15. **Set** $t \leftarrow t+1$.
Step 16. **End While**

Depending on the problem formulation the changes will take place in the above algorithm. The parameter values also depend on the velocity limits and the inertia weight. Normally c1 and c2 are kept constant.

B. PSO Methodology applied to Load Flow

The basis of the PSO algorithm consists in, instant time, analyzing the displacement of each particle in search for the best position and updating its velocity and position using specific equations. The iterative process proceeds until all the particles converge to the obtained global best, which is the adopted solution to the treated problem. The particles‟ positions are defined as the voltage modules and angles of the buses. Applying the PSO algorithm, instead of calculating these voltages through the SFLE, initial estimated values are adopted and updated at each process‟ iteration with the PSO equations, in order to obtain the lowest possible power mismatches.

The particles positions can assume continuing values within the limits specified in the input data. The rule function parameters that will be minimized in the PSO algorithm are defined as *grades*. The grades are defined as the arithmetic mean of the buses apparent power. Each particle has a *local grade*, value obtained by its local best. The *global grade* is the grade related to the best global of all the particles. The *current grade* is the grade obtained by a particle at a given iteration. The first step of the algorithm is to generate the initial values to the particles positions, velocities, local best parameters and global best parameters. The angles receive a random initial value within the specified boundary. Before the initialization of the module value of each particle, the bus type needs to be verified and related in the equation. In the case of a PQ bus, the voltage module receives a random value within the specified boundary; for a PV bus, the voltage module receives the related value specified in the input data. The

initial velocities are null. The local best parameters receive the particles positions values and the global best parameter receives the first particle value, arbitrarily. The grades are initialized with high values in order to be minimized later. Having that accomplished, the iterations are initialized. The following process is accomplished to each particle of the swarm. Firstly the buses voltages receive the particles positions. The reactive power of the PV buses is calculated using equation (a), then the active and reactive power of the slack bus are also calculated using this equation. Finally the power flow in the system lines is calculated in accordance to the equation.

$$S_{ij} = P_{ij} - jQ_{ij} = V_i(V_i^* - V_j^*)Y_{ij}^* - V_i V_i^* Y_{sh,i} \quad (5)$$

Where: S_{ij}= complex apparent power between the buses *i* and *j*;
P_{ij}= active power between the buses *i* and *j*;
Q_{ij}= reactive power between the buses *i* and *j*;
V_i= bus *i* voltage;
V_j= bus *j* voltage; Y_{ij}= admittance between the buses *i* and *j*;
Y_{sh},*i* =*shunt* admittance of the bus *i*.

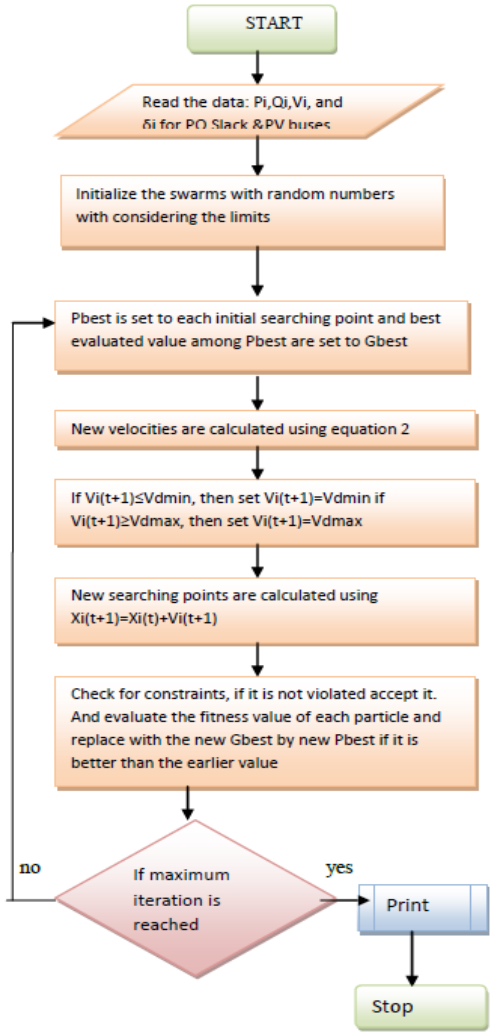
Thus, once all the power of the buses and of the lines is known, the active and reactive power mismatches of each bus are calculated. They are calculated as the sum of the injected power in the approached bus. The apparent power mismatches arithmetic mean is obtained, and this is the value that is desired to be minimized. The local best is replaced by the current particle position in case of the particle current grade is considered better than the local grade. Thus, after all the particles pass through the described process, a similar criterion is used to the global best updating. Next each particle is verified in the following criteria: whether the local grade or global grade is best, the best global is replaced by the approached best local. The velocities as well as the particles positions are updated.

PSO Algorithm is as follows:

The basic elements of the PSO techniques are briefly stated and defined as follows:
1. **Particle X(t):** It is a candidate solution represented by a k-dimensional real-valued vector, where k is the number of optimized parameters. At time t, the i^{th} particle $X_i(t)$ can be described as Xi(t)=[xi,1(t); xi,2(t); ……;xi, k(t)].
2. **Population:** it is a set of n particles at time t.
3. **Swarm and its direction:** it is an apparently disorganized population of moving particles that tend to cluster together while each particle seems to be moving in a random direction.
4. **Particle velocity V(t):** It is the velocity of the moving particles represented by a k-dimensional real-valued vector. At time t, the i^{th} particle $V_i(t)$ can be described as $V_i(t)$=[vi,1(t);vi,2(t); ……;vi,k(t)].
5. **Inertia weight w(t):** it is a control parameter that is used to control the impact of the previous velocity on the current velocity. All the control variables transformer tap positions and switch-able shunt capacitor banks are integer variables and not continuous variables. Therefore, the value of the inertia weight is considered to be 1 in this study.
6. **Individual best X*(t):** As the particle moves through the search space, it compares its fitness value at the current position to the best fitness value it has ever attained at any time up to the current time. The best position that is associated with the best fitness encountered so far is called the individual best X*(t). For each particle in the swarm, X* (t) can be determined and updated during the search.
7. **Global best X**(t):** It is the best position among all of the individual best positions achieved so far.
8. **Stopping criteria:** These are the conditions under which the search process will terminate. In this study, the search will terminate if one of following criteria is satisfied:

- The number of the iterations since the last change of the best solution is greater than a pre-specified number.

The number of iterations reaches the maximumallowable number.



A new PSO algorithm for load flow solution

A new PSO algorithm as applied to load flow has been shown above. We are using IEEE 30-bus system to show the practicability of the proposed algorithm and to find the optimal settings for generator voltages, transformer taps

and switch-able VAR sources while maintain the allowable limit for constraints. The program has run by considering all the limits of generator voltages and load tap changing transformers.

4. Example Analysis

The proposed algorithm has been successfully run using Matlab 2011b for standard IEEE-30- bus system for active and reactive power loss minimization. The system has 41 branches , 6 generator nodes and 22 load nodes. Six generator nodes are 1, 2, 5, 8, 11, 13 where node 1 is a balance node (reference/slack) whereas rest of the five nodes are PV nodes, the other nodes in the system are PQ nodes. The adjustable tap changing transformers branches are 6-9, 6-10, 4-12, 27-28; shunt capacitors nodes are 3, 10, 24. The base value for reference is 100MVA. System parameters quoted reference (Xiuhua Wu,Zailion Piao et al, 2010), the upper and lower limits of each variable as sown in table I-III.

Table I Limits of Voltage and Reactivepower of Generators

Node Number	$V_{g,max}$/pu	$V_{g,min}$/pu	$Q_{g,max}$/pu	$Q_{g,min}$/pu
1	1.1	0.9	0.596	-0.298
2	1.1	0.9	0.480	-0.240
5	1.1	0.9	0.600	-0.300
8	1.1	0.9	0.530	-0.265
11	1.1	0.9	0.150	-0.075
13	1.1	0.9	0.155	-0.078

Table II Voltage Limits of Pq Nodes and Value of Transformer Taps

$V_{load,max}$/pu	$V_{load,min}$/pu	$T_{k,max}$	$T_{k,min}$
1.05	0.95	1.04	0.9

Table III Limits of Capacity and Voltage Of Reactive Power

$Q_{c,max}$	$Q_{c,min}$	$V_{c,max}$,	$V_{c,min}$
0.36	-0.12	1.05	0.95

The overall system loads are given as follows:

P_{load}=283.4MW, Q_{load}= 126.2Mvar

The initial generator voltages and transformers taps are set to 1.0p.u.

The average value obtained for this setting is

P_{loss}=0.059998p.u.

This system was simulated using a New PSO algorithm, and compared with the other optimization techniques (Xiuhua Wu,Zailion Piao et al, 2010) & (Dan Li, Liqun Gao et al, 2010) which are listed in the table IV. The results given here are taken after 30 trials.

The initial settings for PV buses and transformer taps are set to 1.0 and the result obtained after 30 trials taken as average is P_{loss}/pu=0.05215.

After setting the values of these control variables the results has been reduced and the average value taken is given in the comparison table IV.

Table IV Comparison of Different Methods

	∑Pg/pu	∑Qg/pu	P_{loss} /p.u	P_{save} /p.u
SGA	2.88380	1.02774	0.04980	0.00235
AGA	2.88326	0.66049	0.04926	0.00289
EP	2.88362	0.87346	0.04963	0.00252
BPSO	2.88330	0.82500	0.049262	0.00288
IPSO	2.88312	0.81867	0.048899	0.00325
A new PSO Present paper	2.8841	1.00676	0.047945	0.00420

Table V shows the comparison of best, worst and average and standard deviation values for different methods. Due to stochastic/probabilistic characteristic of evolutionary algorithms, results reported here correspond to average form 30 trials.

Table V Comparison of best, Worst, Average and Standard Deviation Values for Different Methods

	Best P_{loss}/pu	Worst P_{loss}/pu	Average P_{loss}/pu
SGA	0.049800	0.052461	0.050012
PSO	0.049262	0.052274	0.051008
APSO	0.048951	0.049774	0.049011
A new PSO Present paper	0.04176	0.05584	0.047945

Conclusion

This paper presented a new PSO algorithm, using a memetic algorithm with Random Walk Directional Exploitation Local Search and ensured better search results in the global scope. The program has been modified by new idea in order to get the converged results using a new PSO algorithm. The proposed algorithm has been successfully applied to an IEEE-30-bus system and the results obtained, i.e. the active and reactive power loss shows that the method had superior computational efficiency and better convergence in less iterations. The developed new PSO algorithm provides flexibility to add or delete any system constraints and objective functions. It was suitable for reactive power and voltage integrated control of power system greatly. In this algorithm all cases, i.e. PQ, PV and slack/reference buses are considered and the results shows the acceptable values. The major advantage of the proposed method is that, it is simple in programming and takes less time to get converge.

References

Konstantinos E. Parsopoulos and Michael N. Vrahatis, Particle Swarm Optimization and Intelligence: Advances and

Applications, Information Science Reference, Hershey.New York. USA.ISBN 978-1-61520-666-7.

A.H Mantawy and M.S Al-Ghamdi (2003), A New Reactive Power Optimization Algorithm IEEE tran., On Bologna Power Tech Conference,. In Italy

Pathak Smita, Prof.B.N.Vaidya (2012), Optimal Power Flow by Particle Swarm Optimization for Reactive Power Loss Minimization, IJSTR.

Xiuhua Wu,Zailion Piao, Yongiang Liu and Haiyan Luo(2010), Reactive power and voltage control based on improved PSO in power system, IEEE Tran, Preceedings of the 8th World congress on Intelligent control and Automation, Jinan, China, 978-1-4244-6712.

Dan Li, Liqun Gao, Junzheng Zhang and Yang Li (2010), Power system reactive optimization based on Adaptive PSO algorithm, IEEE Tran., system, Proceedings of the 8th World congress on Intelligent control and Automation, Jinan, China.. 1-4244-0332-4, 2006.

Zhang, Wen . Liu,Yutian (2010), Reactive power optimization based on Cooperative Coevolution PSO, UPEC.

Jianhua Wu, Nan Li, LIhong He Bin Yin, Jianhu Guo, Yaqiong Liu (2010), Research on Multi-Objective reactive optimization based on modified PSO algorithm, IEEE Tran. 978-1-4244-5182.

Ahmed A. A. Esmin, Germano Lambert-Torre(2005), A hybrid PSO Applied to loss power minimization, IEEE Tran. on Power Systems.Vol.20.NO.2. (0885-8950).

Mohammad Yumus Ali and Kaamran Raahemifar (2012), Reactive power optimization based on Hybrid PSO algorithm', 25th IEEE Canadian Conference on Electrical and Computer Engineering(CCECE), 978-1-4673-1433-6/12.

Authors Profiles

Pruthviraja L
Assistant Professor
Dept. of E&E,
Jain Institute of Technology (JIT)
Davangere, Karnataka, India-577005.

Ramesh G B
Lecturer,
Dept. of E&E,
Gogte Institute of Technology (GIT),
Belagavi, Karnataka, India--590008.