Optimal Power Flow of Distribution Network

Dr.R.Muthu Kumar¹ and Ms. Swathi M²

¹ Associate Professor , Department of EEE, Shree Venkateshwara Hi-Tech Engineering College, Erode, India
²PG Scholar, ME (PSE),Department of EEE, Shree Venkateshwara Hi-Tech Engineering College, Erode, India

Abstract-This paper presents solution of optimal power flow (OPF) problem of a power system via a simple Particle Swarm Optimization (PSO) algorithm. This method is dynamic in nature and it overcomes the shortcomings of all other evolutionary computation techniques such as premature convergence and provides high quality solutions. The objective of OPF is to minimize the fuel cost and keep the power outputs of generators, bus voltages, shunt capacitors/reactors and transformers tap settings in their secure limits. The effectiveness of PSO was compared to that of OPF solution by ETAP [Electrical Transient analyzer Program]. The potential and superiority of PSO have been demonstrated through the results of IEEE 30-bus system.

Key Words---Particle Swarm Optimization, Optimal Power Flow,load Flow.

I.INTRODUCTION

The mainaim of electric power utilities is to provide high quality reliable supply to the consumers at the lowest possible cost without violating the limits and constraints imposed by the system. Dispatch of available generation to meet the demand in such a way that minimizes /maximizes the objective function is called Optimal Power Flow. OPF can fully represent the network equations and nodal power balance. It also maintains limits on bus voltage, branch flow and generator outputs.

The optimal power flow has been frequently solved using classical optimization methods such as nonlinear programming, quadratic programming, linear programming, interior-point method. Conventional optimization methods are based on successive linearization using the first and the second derivatives of objective functions and their constraint as the search directions [3][6]. The conventional optimization methods usually converge to a local minimum [7]. Recently, intelligence heuristic algorithms, such as genetic algorithm [8], evolutionary programming [9], and metaheuristic algorithms [10] have been proposed for solving the OPF problem. Like other metaheuristic algorithms, particle swarm optimization (PSO) algorithm was developed through simulation of a Simplified social system such as bird flocking and fishing school. PSO is an optimization method based on population [11], and it can be used to solve many complex optimization problems, which are nonlinear, non differentiable and multi-modal. The most prominent merit of PSO is its fast convergence speed. In addition, PSO algorithm can be realized simply for less parameters need adjusting. PSO has been applied to various power system optimization problems with impressive success [12]. The results for a 30-bus system show that PSO is an effective method to solve OPF problem.

II.MATHEMATICAL FORMULATION OF OPF PROBLEM

The main objective of OPF is to optimize certain objective function by adjusting the power system control variables (real power generation, generator set point voltage, tap ratio, shunt capacitor). In solving the OPF using Newton method, the marginal cost data are determined as a byproduct of the solution technique [1][15]. Depending on the specific objectives and constraints, there are different OPF formulations. The typical objectives are:

• Minimization of total generation cost;
• Minimization of the active power losses;
• Maximization of the degree of security of the system;
• Optimization of the voltage-reactive power.

The objective function for the OPF reflects the costs associated with generating power in the system and it is assumed to be approximated by a quadratic function of generator active power output as:

\[ C_1 = a_1P_{G}^2 + b_1P_{G} + c_1 \]  \hspace{1cm} (1)

\[ C_i = \sum_{i=1}^{n} C_i \]  \hspace{1cm} (2)

Where \( C_i \) - cost function of individual plant.

Generally, this objective is given by
\[ F = \sum_{i=1}^{n_g} (a_i P_{gi}^2 + b_i P_{gi} + c_i) \]  
(3)

Where

- \( n_g \) is the number of generation including the slack bus,
- \( P_{gi} \) is the generated real power at bus \( i \),
- \( a_i \) - fuel cost coefficient ($/MW^2h$)
- \( b_i \) - fuel cost Coefficient ($/MWh$)
- \( c_i \) - fuel cost Coefficient ($) of unit costs curve for \( i^{th} \) generator.

(i) Equality constraints.

Power flow equations are used as equality constraints.

\[
P_i(V, \theta) = P_{gi} - P_{di} = \sum_{j=1}^{nb} V_i V_j (g_{ij} \cos \theta_{ij} + b_{ij} \sin \theta_{ij})
\]
(4)

\[
Q_i(V, \theta) = Q_{gi} - Q_{di} = \sum_{j=1}^{nb} V_i V_j (g_{ij} \sin \theta_{ij} + b_{ij} \cos \theta_{ij})
\]
(5)

Where,

- \( P_{gi}, Q_{gi} \) : the total real and reactive power generation at bus \( i \).
- \( P_{di}, Q_{di} \) : the total real and reactive power load at bus \( i \).
- \( V_i \) : the voltage magnitude at bus \( i \).
- \( g_{ij} \) : the real part of admittance matrix.
- \( \theta_{ij} \) : the voltage angle of the \( ij \)th element of admittance Matrix.
- \( b_{ij} \) : the imaginary part of admittance matrix.
- \( nb \) : number of bus.

(ii) Inequality constraints

The most usual types of inequality constraints are upper bus voltage limits at generations and load buses, lower bus voltage limits at load buses, VAR limits at generation buses, maximum/minimum active power limits at generators, maximum line loading limits and limits on tap settings and phase shifter. The inequality constraints on the problem variables considered include:

- **Upper and lower limits on the active power generations at generator buses**
  \[
P_{gi}^{\min} \leq P_{gi} \leq P_{gi}^{\max}, i = 1,ng
\]

- **Upper and lower limits on the reactive power generations at generator buses and reactive power injection at buses with VAR compensation**
  \[
  Q_{gi}^{\min} \leq Q_{gi} \leq Q_{gi}^{\max}, i = 1, npv
\]

- **Upper and lower limits on the voltage magnitude at the all buses**
  \[
  V_i^{\min} \leq V_i \leq V_i^{\max}, i = 1, nbus
\]

- **Upper and lower limits on the bus voltage phase angles**
  \[
  \theta_i^{\min} \leq \theta_i \leq \theta_i^{\max}, i = 1, nbus
\]

The generalized objective function \( f \) is a non-linear one, the number of the equality and inequality constraints increase with the size of the power distribution systems. Applications of a conventional optimization technique such as the gradient-based algorithms, to large power distribution systems with a very nonlinear objective functions and great number of constraints, are not good enough to solve this problem. Because they depend on the existence of the first and the second derivatives of the objective function.

(iii) Security constraints

Programs which can make control adjustments to the base case or pre-contingency operation to prevent violations in the post-contingency conditions are called “Security Constrained Optimal Power Flow “or SCOPF.

The constraints of the voltage at load buses and transmission line loadings are considered.

\[
V_{L_i}^{\min} \leq V_{L_i} \leq V_{L_i}^{\max}, i = 1,\ldots,N_L
\]

\[
S_{li} \leq S_{li}^{\max}, i = 1,\ldots,N_L
\]

II. PSO ALGORITHM IN OPTIMAL POWER FLOW

(a) Basics of Particle Swarm Optimization

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Dr. Eberharth and Dr. Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling. In PSO, the potential solutions called particles fly through the problem space by following the current optimum particles. Compared to GA, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust. PSO has been successfully applied in many areas: function optimization, artificial neural
network training, fuzzy system control, and other areas where GA can be applied. As stated before, PSO simulates the behaviors of bird flocking.

Consider the following scenario: a group of birds are randomly searching food in an area. There is only one piece of food in the area being searched. All the birds do not know where the food is. But they know how far the food is in each iteration. The effective way to find the food quickly is to follow the bird which is nearest to the food. PSO learned from the scenario and used it to solve the optimization problems. In PSO, each particle (bird) in the search space is considered as a potential solution. All the particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles.

PSO is initialized with a group of random particles and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far called Pbest. (The fitness value is also stored). Another "best" value is the best value among all individual best obtained so far. This best value is a global best and called gbest. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called Lbest.

Compared with genetic algorithms (GAs), the information sharing mechanism in PSO is significantly different. In GAs, chromosomes share information with each other. So the whole population moves like a one group towards an optimal area. In PSO, only gbest (or Lbest) gives out the information to others. It is a one-way information sharing mechanism. The evolution only looks for the best solution. Compared with GA, all the particles tend to converge to the best solution quickly even in the local version in most cases. The basic principles in "classical" PSO are very simple. A set of moving particles (the swarm) is initially "thrown" inside the search space. Each particle has the following features:

- It has a position and a velocity
- It knows its position, and the objective function value for this position
- It knows its neighbors, best previous position and objective function value (variant: current position and objective function value)
- It remembers its best previous position

At each time step, the behavior of a given particle is a compromise between three possible choices:

- To follow its own way
- To go towards its best previous position
- To go towards the best neighbor's best previous position, or towards the best neighbor (variant)

This compromise is formalized by the following equations:

\[ v_{k+1} = w v_k + c_1 r_1 (p_1 - x_k) + c_2 r_2 (p_2 - x_k) \] 
\[ x_{k+1} = x_k + v_{k+1} \]

where:

- \( v_{k+1} \): the current velocity of particle.
- \( w \): The inertia weighting function.
- \( v_k \): the previous velocity of particle
- \( c_1, c_2 \): the cognitive and the social parameters, respectively.
- \( r_1, r_2 \): random numbers uniformly distributed within [0,1]
- \( p_1 \): the best previous position of the \( k \)th particle (pbest)
- \( p_2 \): the global best in the \( k \)th swarm.
- \( x_{1:k} \): the current position of particle
- \( x_k \): the previous position of particle

The first part of (6) is the inertia velocity of particle, which reflects the memory behavior of particle; the second part is cognition part, which represents the private thinking of the particle itself; the third part is the social part, which shows the particle's behavior system from the experience of other particles in the population. The particles find the optimal solution by cooperation and competition among the particles [12]. Using the above equation, a certain velocity, that gradually gets close to \( p_1 \) and \( p_2 \), can be calculated. The position of each particle (searching point in the solution space) can be modified according to (7).

(b). PSO application to Optimal Power Flow.

The cost function is defined as:

\[ F = \sum_{i=1}^{ng} (a_i P_{gi}^2 + b_i P_{gi} + c_i) \]
\[ P_{gi}^{\min} \leq P_{gi} \leq P_{gi}^{\max} \]

Minimize \( F \) is equivalent to getting a maximum fitness value in the searching process. The particle that has lower cost function should be assigned a larger fitness value. The objective of OPF has to be changed to the maximization of fitness to be used as follows:
$$\text{fitness} = \begin{cases} f_{\text{max}} / F & \text{if } f_{\text{max}} > F \\ 0 & \text{otherwise} \end{cases}$$

(9)

The PSO algorithm applied to OPF can be described in the following steps:

**Step 1:** Get the input parameters of the power system network and specify the lower /upper limits of each variable.

**Step 2:** Initialize the particles position (x) and velocity (v) randomly.

**Step 3:** Calculate the evaluation value of each particle using the objective function.

**Step 4:** Calculate the fitness value of objective function of each particle using (9). \( p_1 \) is set as the \( k \)th particle's initial position; \( P_2 \) is set as the best one of \( p_1 \), and the current evolution is \( t = 1 \).

**Step 5:** Initialize learning factor \( c_1, c_2 \) \( [i.e. c_1+c_2=4] \), inertia weight 'w' \( [0.4 \text{ to } 0.9] \) and the initial velocity \( v_t [v_{1}=0] \)

**Step 6:** Modify the velocity ‘v’ of each particle according to (6).

**Step 7:** Modify the position of each particle according to (7).

If a particle violates its position limits in any dimension, set its position at the proper limits. Calculate each particle’s new fitness, if it is better than the previous \( p_2 \), the current value is set to be \( P_2 \).

**Step 8:** To each particles of the population, employ the Newton-Raphson method to calculate power flow and the transmission loss.

**Step 9:** Update the time counter \( t=t+1 \).

**Step 10:** If one of the stopping criteria is satisfied then go to step 11. Otherwise go to step 6.

**Step 11:** The particle that generates the latest p is the best optimal value.

(c). Load flow calculation

Once the global best position is found, or an allowable generation (iterations) is attained by the PSO algorithm, it’s required to perform a load flow solution in order to make better adjustments on the optimum values obtained from the PSO-OPF procedure.

This will provide updated voltages, angles and transformer taps and points out generators having exceeded reactive limits. Examples of reactive constraints are the min / max. Reactive rate of the generators buses and the min. and max. of the voltage levels of all buses. All these require a fast and robust load flow program with best convergence properties. The developed load flow process is based upon the full Newton-Raphson algorithm using the optimal multiplier technique [14].

**IV. CASE STUDY**

The PSO-OPF algorithm is tested using the modified IEEE 30-bus system. The system consists of 41 lines, 6 generators, 4 Tap-changing transformers, and shunt capacitor banks located at bus 10 and 24. The Total real power demand is 283.4MW. The system is optimized using the PSO-OPF algorithm. The parameter settings to execute PSO-OPF are \( w = 0.9 \), \( c_1 = 2 \), \( c_2 = 2 \), \( V_{inc} = 1.98 \), no. of particles=20, max. Generation=200, the power mismatch tolerance is 0.0001 p.u; the maximum voltage magnitude of all generating bus is 1.1 p.u and other buses are 1.0 p.u while the minimum voltage magnitude is 0.95 p.u.

Other power system parameters are presented in Table I. To compare these results with other methods the same problem with same limits and constraints was solved by ETAP [Electrical Transient Analyzer Program]. The comparison of the results (active power generation, bus voltages, total generation cost and power losses) are presented in Table II.

<table>
<thead>
<tr>
<th>Bus</th>
<th>( P_g ) [MW]</th>
<th>( Q_g ) [Mvar]</th>
<th>( V_g ) [p.u.]</th>
<th>( a )</th>
<th>( b )</th>
<th>( C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>50</td>
<td>200</td>
<td>-</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>80</td>
<td>-20</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>50</td>
<td>-30</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>30</td>
<td>-40</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>15</td>
<td>35</td>
<td>-15</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>15</td>
<td>40</td>
<td>-30</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>30</td>
<td>-40</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>30</td>
<td>10</td>
<td>-30</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>20</td>
<td>20</td>
<td>-30</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>20</td>
<td>-20</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>5</td>
<td>5</td>
<td>-5</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>10</td>
<td>10</td>
<td>-10</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>15</td>
<td>15</td>
<td>-15</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

![Fig.1 PSO simulation output graphs](image-url)
The results show that PSO algorithm gives much better results than solved by ETAP. The difference in generation cost between these two methods (801.84 $/hr compared to 805.66 $/hr) and in Real power loss (9.3761 MW compared to 10.407 MW) clearly shows The advantage of this method. In addition, it is important to point out that this simple PSO algorithm OPF converge in an acceptable time. For this case time taken was approximately 30 seconds, and it converged to the optimal solutions set after 200 generations tested with Intel(R) Pentium(R) CPU B950 at 2.10GHz. The security constraints are also checked for voltage magnitudes and angles. The voltage magnitudes are minimum of 0.9957 p.u. at bus 30 and maximum of 1.082 p.u. at bus 11. (Fig. 2), and the angles are the minimum of -17.91° [at bus 30] and the maximum of 0.0° [at bus 1].

TABLE II : COMPARISON OF PSO AND ETAP OPF

<table>
<thead>
<tr>
<th>Control Variables</th>
<th>ETAP</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁(MW)</td>
<td>200.00</td>
<td>176.75</td>
</tr>
<tr>
<td>P₂(MW)</td>
<td>41.42</td>
<td>48.80</td>
</tr>
<tr>
<td>P₅(MW)</td>
<td>15.00</td>
<td>21.48</td>
</tr>
<tr>
<td>P₆(MW)</td>
<td>15.39</td>
<td>21.65</td>
</tr>
<tr>
<td>P₁₁(MW)</td>
<td>10.00</td>
<td>12.11</td>
</tr>
<tr>
<td>P₁₃(MW)</td>
<td>12.00</td>
<td>12.00</td>
</tr>
<tr>
<td>Total Generation(MW)</td>
<td>293.81</td>
<td>292.79</td>
</tr>
<tr>
<td>V₁(p.u)</td>
<td>1.110</td>
<td>1.060</td>
</tr>
<tr>
<td>V₂(p.u)</td>
<td>1.082</td>
<td>1.043</td>
</tr>
<tr>
<td>V₅(p.u)</td>
<td>1.054</td>
<td>1.010</td>
</tr>
<tr>
<td>V₆(p.u)</td>
<td>1.058</td>
<td>1.010</td>
</tr>
<tr>
<td>V₁₁(p.u)</td>
<td>1.160</td>
<td>1.082</td>
</tr>
<tr>
<td>V₁₃(p.u)</td>
<td>1.109</td>
<td>1.071</td>
</tr>
<tr>
<td>Demand (mw)</td>
<td>283.4</td>
<td>283.4</td>
</tr>
<tr>
<td>Real power loss(mw)</td>
<td>10.407</td>
<td>9.3761</td>
</tr>
<tr>
<td>Total Gen.cost($/hr)</td>
<td>805.66</td>
<td>801.84</td>
</tr>
</tbody>
</table>

Fig 2: Comparison between ETAP and PSO for the voltage magnitude after optimization

V. CONCLUSION
For the solution of constrained optimization problems such as OPF problem in power systems, penalty function based methods is the most popular approach. However, since the penalty function approach is generic and applicable to any type of constraints, their performance is not always satisfactory and consistent. In order to overcome this drawback, in this paper a PSO method is used for the solution of OPF problem. In view of the memory characteristics of the PSO, a new constraints handling strategy is developed. The proposed PSO approach is efficiently and effectively minimizing the total generation production cost in an Optimal Power Flow problem. As a study case, the IEEE 30 Bus system with 6-generating units has been selected. The simulation results show that a simple PSO can give best result than ETAP. The effectiveness of the PSO for solving OPF problem with some other objective function will be investigated in the future research work. The simulation results show that the proposed PSO method is superior to the ETAP and the results confirm its good performance in terms of the solution quality, computational cost as well as the convergence stability. PSO is dynamic in nature and it overcomes the shortcomings of other evolutionary computation techniques such as premature convergence and provides high quality solutions.

REFERENCES


