REACTIVE POWER CONTROL USING CO–EVOLUTIONARY PARTICLE SWARM OPTIMIZATION (CPSO) FOR REAL POWER LOSS MINIMIZATION

Valiollah Zanganeh1,2, Masud Radmehr3*, Mahmood Ghanbari2
1 Department of Electrical Engineering, Golestan Science and Research Branch, Islamic Azad University, Gorgan, Iran.
2 Department of Electrical Engineering, Gorgan Branch, Islamic Azad University, Gorgan, Iran.
3 Department of Electrical Engineering, AliAbad Katoul Branch, Islamic Azad University, AliAbad Katoul, Iran.
*Corresponding author: Email: masoud_radmehr@yahoo.com

ABSTRACT- Reactive power by maintaining voltage stability and reliability of the system plays an important role in the reinforcement of active power transmission support. Reactive power is also an essential element of the transfer process to provide the electric system reliability by minimizing the total cost function. In this paper, chaotic particle swarm optimization algorithm (CPSO) was presented to optimize the reactive power distribution. The problem of optimal distribution of reactive power is a nonlinear optimization problem with multiple constraints. The purpose of presenting CPSO algorithm was to minimize the total losses through changing the generator’s power, buses voltage and also reactive compensation devices. A 13-bus sample system was used to implement the optimization of the program. The results of optimization program were compared with PSO algorithm to demonstrate the performance of CPSO algorithm. Finally, comparison of results of the CPSO algorithm with PSO showed the CPSO algorithm’s effectiveness and superiority in terms of convergence speed and quality of variables’ response.

Keywords: reactive power, power systems, PSO, CPSO

1. INTRODUCTION

The optimal distribution of reactive power is very effective on the security and economic performance of power systems. Although, producing reactive power does not cost itself during the operation stage, but it is effective on the total cost through impact on the system losses. The optimal distribution of reactive power is considered as an important issue is the optimal power flow (OPF) and mainly, it is done through proper control of reactive power. Parameters (control or decision variables) which must be set in this issue include: Reactive power of the generator and synchronous compensators, changeable transformers tap under the load and the size of the parallel installed capacitors. The objective function is the transmission loss of the network, which is added for some security concerns and physical limitations in terms of equipment. In this problem, Reactive power of the generator and synchronous compensators, continuous variables and transformers tap and the size of the parallel capacitors are the discrete variables. Thus, the mentioned problem is a nonlinear optimization problem with a combination of continuous and discrete variables (MINLP). [1]

Various evolutionary techniques such as genetic algorithms (GA), evolutionary programming (EP), evolutionary strategies (ES) and particle swarm optimization have been applied to optimization problems in power systems. [2]

Differential evolution (DE) is an evolutionary algorithm which is presented to solve very simple and acceptable optimization problems. Actual optimization problems are explained often through irreversible and competitive fitness objectives. The presence of multiple objectives in a problem mainly lead to an optimal solution set is known as the optimal solutions instead of a single optimal solution. In the absence of any further information, it is impossible to decide that the optimal solutions are better than others. Thus, the operator must find optimal solutions to choose the best answer considering the personal need. Using the multi-objective optimization techniques in power systems has advantages such as: (1) manages different objectives, (2) simplifies the process, and (3) it is considered with respect to all objective functions. In this method, system designer had the power of several solutions for better decision-making. [3]

So far, many traditional methods have been proposed such as gradient-based optimization and mathematical programming methods to solve this problem. Recently, various algorithms based on interior point methods have been used which have proper integration and management in dealing with inequality constraints, such as interior point linear programming, quadratic interior point and nonlinear programming. However, these methods are faced with serious limitations in managing non-linear and discontinuous functions and have many local minimum and include discrete variables. The problem of reactive power optimal distribution has these characteristics. In recent years, the random search method is presented for solving global optimization problems. Genetic algorithms, evolutionary programming, evolution strategies and ant colony can be mentioned as the most famous of these methods. Chaotic particle swarm optimization algorithm (CPSO) is a stochastic search algorithm based on collective intelligence. The PSO algorithm converges to the optimal solution in a shorter time compared to other algorithms, meanwhile it has a stronger convergence characteristic. Also, its implementation is very easy and the parameters that must be set are low. PSO algorithm was successful in solving power system problems. The application of this algorithm can be mentioned in economic load flow, planning for development of plants, and optimal capacitor spotting. [4]

In this paper, CPSO algorithm is introduced and an optimal distribution of reactive power is solved on a 13-bus sample system using both PSO and CPSO algorithm. Comparison of the results shows that CPSO algorithm has a more efficient response.

2. Problem Definition

One of the main purposes of reactive power optimal distribution is minimizing the real power losses in the transmission network which therefore, must be considered as the objective function of the optimization problem. Losses in the transmission network can be calculated as follows:

May-June
\[ f_Q = \sum_{k \in N_k} P_{k^{\text{loss}}} = \sum_{k \in N_k} g_k (V_i^2 + V_j^2 - 2V_iV_j \cos(\theta_{ij})) \]

(1)

In which, \( N_k \) is transmission lines set, \( P_{k^{\text{loss}}} \) is the loss of the transmission line \( k \), \( V_i, V_j \) are voltage buses of \( i \) and \( j \), 
\( g_k \) is conductivity of \( k \) branch and \( \theta_{ij} \) angle between the voltages of \( i \) and \( j \). Minimization function is performed by the following constraints:

\[ 0 = P_{G_{ij}} - P_{D_{ij}} - V_i \sum_{j \in N_0} V_j (G_{ij} \cos(\theta_{ij}) + B_{ij} \sin(\theta_{ij})) \] (2)

\[ 0 = Q_{G_{ij}} - Q_{D_{ij}} - V_i \sum_{j \in N_0} V_j (G_{ij} \cos(\theta_{ij}) + B_{ij} \sin(\theta_{ij})) \] (3)

\[ V_i^{\text{min}} \leq V_i \leq V_i^{\text{max}} \] (4)

\[ T_k^{\text{min}} \leq T_k \leq T_k^{\text{max}} \] (5)

\[ Q_{G_{ij}}^{\text{min}} \leq Q_{G_{ij}} \leq Q_{G_{ij}}^{\text{max}} \] (6)

\[ Q_{C_{ij}}^{\text{min}} \leq Q_{C_{ij}} \leq Q_{C_{ij}}^{\text{max}} \] (7)

In this equations, \( P_{G_{ij}} \) and \( Q_{G_{ij}} \) respectively are active and reactive power injection at bus \( i \), \( P_{D_{ij}} \) and \( Q_{D_{ij}} \) respectively are active reactive power at bus \( i \), \( N_0 \) is the set of all buses except reference bus, \( N_{PQ} \) is the set of all load buses, \( G_{ij} \) is transfer conductivity between buses \( i \) and \( j \) and \( B_{ij} \) is transfer susceptance between buses \( i \) and \( j \). Load flow equations are considered as equal constraints. Limitation of voltage buses, limitation of tap changers’ tap, limitation of generators’ reactive power and parallel capacitors are inequality constraints of the problem. In some methods for solving optimization problems (such as random search solving methods) limitations are added by applying fine coefficient to the objective function. In the problem of reactive power distribution optimization, buses voltage, tap changers’ tap and injected reactive power of capacitors are the variables which directly can be limited and placed in the desired interval. [5]

3. PSO and CPSO algorithm

3.1. PSO algorithm

PSO algorithm is a global optimization method that can answer the problems whose answer is a point or level in N-dimensional space. In such a space, some assumptions are discussed and an initial velocity of the particle is assigned. Also, some communication channels are considered between the particles. Then, these particles move in answer space and the results are calculated based on a competency criterion after each period. Over time, particles are accelerated toward particles that have higher competency criteria and they are in the same communication group. The main advantage of this method over other optimization strategies is that a large number of particles cause flexibility for the method in response to the local optimum. Each particle has a position that determines the coordinates of the particle in the multidimensional search space. The position of the particle changes by moving it over the time. \( x_i(t) \) is the position of \( i \)th particle at time \( t \). By adding speed to the position of each particle, the particle can be considered a new location. The equation for updating any particle is as following:

\[ x_i(t + 1) = x_i(t) + v_i(t + 1) \] (8)

\[ v_i(t) \sim U(x_{\text{min}}, x_{\text{max}}) \] (9)

The fitness of a particle position is specified in the search space by a fitness function. Particles are able to memorize the best position during their life. The best individual experience of a particle or the best met position is called \( y_i \) (in some algorithms, \( y_i \) is also named as pbest) and the particles can be aware of the best met position by the whole group and this information is named \( y_i \) (in some algorithms \( y_i \) is also named as gbest). Particle velocity vector in the optimization process reflects the empirical knowledge of particle and particles’ society information. Each particle considers two components to move in the research space:

Cognitive component: \( y_i(t) - x_i(t) \) is the best solution that a particle yields alone.

Social component: \( y_i(t) - x_i(t) \) the best solution that is detected by the whole group.

In this paper, some modifications are applied about the concept of coevolution. Then, we integrate it in PSO solving constrained optimization problems. Principle of CPSO coevolutionary model is shown in Figure 1. In CPSO method, two types of swarm are used. One kind of swarm is single or individual (which is shown by Swarm2) as M2 which is used especially for adaptation of the appropriate penalty coefficients and the other kind is multiple swarm (which is shown by Swarm1, 1 and Swarm1, 2 and ... and Swarm1,m2) as M1 which is used to parallel search for good answers. Each Bj particle in Swarm2 represents a set of penalty coefficients for the particles in j, Swarm1 in which, each particle represents an answer or solution for decision-making. In each generation of the coevolutionary process, each Swarm1, j will be evolved using PSO for a certain number of generations (G1) with particle Bj in Swarm2 as the penalty coefficients to evaluate the results in order to obtain a new Swarm1, j. Then, the suitability of each particle in Swarm2 Bj will be determined. After evaluating all the particles in Swarm2, Swarm2 will be evolved using PSO with another generation to obtain a new Swarm2 with modified and adjusted penalty coefficients. The above coevolutionary process will be repeated until meeting the predetermined stopping criteria (for example, meet the maximum number of coevolution G2 generations). In summary, two types of particle swarm evolve in interaction, according to which, Swarm1, j is used to develop decision solutions and Swarm2 is used to adjust penalty coefficients to assess the answers. Due to the coevolutionary process, solutions or decision answers are not only detected in an evolutionary form, but also penalty coefficients are adjusted and refined in self-
regulation form avoid the problem of determining the appropriate coefficient through trial and error method. [7]

![Figure 1. Co-evolutionary model in CPSO [7]](image)

### 3.2.1. Evaluation function for Swarm1, j

We design penalty function for constrained optimization problems by following the guidelines suggested by Richard Sen et al. (1990). This is done not only for detecting the number of violated constraints, but also it is used for the values in that the constraints are violated. Specifically, the ith particle in Swarm1 and j is evaluated in CPSO using Equation (10):

\[
F_i(x) = f_i(x) + \text{sum} \_\text{viol} \times w_1 + \text{num} \_\text{viol} \times w_2
\]

In which, \(f_i(x)\) is the objective function at ith particle, \(\text{sum} \_\text{viol}\) represents the sum of all the values that constraints are violated by them, \(\text{num} \_\text{viol}\) represents the number of violated constraints and \(W_1\) and \(W_2\) are penalty coefficients corresponding with Bj particle in Swarm2. The amount of \(\text{sum} \_\text{viol}\) is calculated using Equation (11):

\[
\text{sum} \_\text{viol} = \sum_{i=1}^{N} g_i(x), \forall g_i(x) > 0
\]

In which, \(N\) is the number of inequality constraints (here it is assumed that all constraints have become equal to inequality constraints).

### 3.2.2. Evaluation function for Swarm2

Each particle in Swarm2 represents a set of coefficients (\(W_1, W_2\)). After the evolution of Swarm1 and j for a certain number of generations (G1), the ith particle Bj is evaluated in Swarm2:

1. If there is at least one possible answer Swarm1, j, then, particle Bj is evaluated by Equation (12) and it is called a valid particle:

\[
P(B_j) = \frac{\sum f_{\text{feasible}}}{\text{num} \_\text{feasible}} - \text{num} \_\text{feasible}
\]

\(\sum f_{\text{feasible}}\) in Equation 12 represents the sum of the values of the objective function, possible solutions in j and Swarm1 and num_feasible is the number of possible answers in j and Swarm1. The only reason to consider possible solutions is the orientation of Swarm1 j toward the possible areas. In addition, num_feasible subtraction in Equation 12 is to avoid from recession or stagnation of Swarm1, j in the certain areas in which, only very few particles will have suitable objective function values or even a few numbers are possible. Therefore, j and Swarm1 will encourage to move toward areas containing a large number of possible values of the objective function. Furthermore, num_feasible acts as a scale factor to divide \(\sum f_{\text{feasible}}\).

2. If there is no possible answer in j and Swarm1 (it can be considered that the penalty or fine is too low), then particle Bj in Swarm2 is evaluated in Equation 13 which is called an invalid particle:

\[
P(B_j) = \max(P_{\text{valid}}) + \frac{\sum \text{sum} \_\text{viol}}{\text{num} \_\text{viol}} - \text{num} \_\text{viol}
\]

In which, \(\max(P_{\text{valid}})\) represents the maximum fitness value of all the valid particles in Swarm2, \(\sum \text{sum} \_\text{viol}\) represents the sum of constraint violations for all particles in j and Swarm1 and \(\text{num} \_\text{viol}\) is the total number of constraint violations for all particles j and Swarm1. Specifically, in Equation 13, the particle in Swarm2 which leads to less constraint violations in Swarm1 and j is considered as the best answer. Therefore, searching may cause orientation for Swarm1 and j toward an area in that the total violations of provisions is low (i.e., the feasible or possible boundary). Furthermore, the addition of items max (Pvalid) to ensure that the valid particle is always better than any other valid particle and searching may be directed towards the search area.

\(\sum \text{num} \_\text{viol}\) acts as a scaling factor.

### 3.2.3. Assessment of J and Swarm1 and Swarm2

Particle in both types swarm are explained using PSO method and they will be evolved in Section 2. Especially, particle in Swarm2 encodes a set of penalty coefficients (\(w_1, w_2\)), while the particle in j and Swarm1 encodes a set of decision variables. Both types of particles use equations 4 and 5 to modify their positions to obtain good results and making appropriate use of penalty coefficients. Due to properties of PSO such a process can easily be implemented that the simulation results show the efficiency of this method. After description of the main elements of the coevolutionary PSO, the CPSO framework is clearly shown in Figure 2. CPSO properties can be summarized as follows:

A) Interaction evolution of the two types of congestion using PSO in which, a congestion of answer is for decision and the other type is for penalty coefficients. B) Penalty coefficients are modified and adjusted using a self-regulation method. C) CPSO is based on population and its parallel implementation is easy.

### 4. Implementation stages of PSO and CPSO algorithm on a sample system of IEEE 6-bus

Implementation stages of the algorithm for solving the problem of reactive power optimal distribution are as follows: Initial position and velocity of the particles in the allowed range is generated randomly. Position is the variables of the problem that in this system are considered as the position (variables) of the algorithm.

\[
x = [V_1, V_2, T_{43}, T_{65}, Q_4, Q_6]
\]

\(x\) = The position of a particle which is composed of six different systems.

- \(V_1\) = Bus 1 voltage
- \(V_2\) = Bus 2 voltage
- \(T_{43}\) = Transformer tap between the third and fourth bus

May-June
Transformer tap between the fifth and sixth bus

Injective or absorptive reactive power on the fourth bus

Injective or absorptive reactive power on the sixth bus

Explaining that by considering the positions (variables) and displacement in the desired range (which is the system limitations such as voltage range, injective or absorptive reactive power, and etc.) The algorithm will go towards a direction to minimize the objective function in the positions (variables). [1]

For the beginning, a series of primary particles should be considered. In this system, the number of particles is 30. Whatever this number be higher, the final answer will be more accurate, because coneltier covers the desired range, but it will reduce the speed of response in a same range. The particles in each epoch of the algorithm should be selected randomly. Of course, random selection of the particles should be such that the constituent particles (the six parameters of the system) remain in their designated areas.

Figure 2. Flowchart of CPSO algorithm [7]

Also, in this system the initial velocity of all particles is considered zero.

Net loss which is the objective function for the implementation of the algorithm is obtained using load flow time and it is considered as the first P_best and G_best.

Then, the variables of each particle are moved toward the optimum point using the equations and 30 epoch and in each epoch load flow is done again. The results of this stage will be the results of PSO method.

Given that in this paper, optimization of total active loss of the system (which is listed as the objective method) will be done by reactive power control. Controlling the considered variables (bus voltage, reactive power injected to the bus and tap transformer) has a direct relationship with the reactive power control system. 30 particles are generated as the initial population with these 2 variables.

Table 1. Considered variables for sample 13-bus system

| Bus 1 voltage | V₁ |
| Bus 2 voltage | V₂ |
| Injective reactive power on the fourth bus | Q₄ |
| Injective reactive power on the fifth bus | Q₅ |
| Transformer tap between the sixth and seventh line | Tap₆₇ |
| Transformer tap between the sixth and eighth line | Tap₆₈ |

Table 2. Information of the sample 13-bus system

<table>
<thead>
<tr>
<th>Start bus</th>
<th>End bus</th>
<th>Branch Impedance</th>
<th>Transformer tap</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>0.00139+0.00296j</td>
<td>0.975</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0.003132+0.05241j</td>
<td>0.975</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>0.00122+0.00243j</td>
<td>0.975</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>0.063953+0.37796j</td>
<td>0.975</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>0.00075+0.00063j</td>
<td>0.975</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>0.059184+0.355104j</td>
<td>0.975</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>0.043142+0.345142j</td>
<td>0.95</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>0.00157+0.00131j</td>
<td>0.975</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>0.058286+0.37887j</td>
<td>0.975</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>0.00109+0.00091j</td>
<td>0.975</td>
</tr>
<tr>
<td>11</td>
<td>12</td>
<td>0.0557533+0.3624j</td>
<td>0.975</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>0.0121813+0.14616j</td>
<td>0.95</td>
</tr>
</tbody>
</table>

Figure 3. Single-line diagram of the sample 13-bus system

4.1. Implementation process of the problem

The considered prototype system to evaluate the performance of PSO and CPSO methods is a 13-bus sample system. [8] Initially, variables are specified for PSO method according to the mentioned procedure. Table 1 shows the list of considered variables for this system.
(the best particle) are considered as the initial population for CPSO method in order to prevent the dispersion of results. Then, after 30 epoch each particle variable is moved toward the optimal point (by keeping other particle variables). The best result for the first variable along with other stabilized variables will be considered as the initial population for the second variable. Similarly, it moves towards the optimum point until checking out the variables of a variable by fixing other variables. Finally, the best result of the last variable will be introduced as the final result of CPSO method.

5. SIMULATION RESULTS

According to the mentioned procedure in the previous section, the loss results of PSO and CPSO methods are shown in Figure 4.

As can be seen in the figure, by applying CPSO method on the PSO method’s results, the loss has been reduced from 0.00657 MW to 0.00633 MW. This indicates the proper performance of CPSO method. Figure 5 shows the loss diagram in each epoch for one variable (respectively from the first variable to the last variable). Figure 5 indicates that the loss in each stage of changing a variable reduces. The last stage will be the optimal point of CPSO. The interesting point in this diagram is that the loss was fixed at 30 epoch for two variables and this suggests that these values in the two methods are the best values.

Table 3 shows the optimal values of the two methods. It can be seen in the table that all values of the variables are changed except the value of the variable one that has no change in CPSO method compared to PSO method.

Table 4 shows the optimal values in each change stage for a variable. Table 4 shows the value of the best particle in each stage of implementing CPSO method. As shown in the algorithm of Figure 2, in the first stage, variable \( V_1 \) is placed in the optimization process and all other variables remain constant. The first row of Table 4 shows the best particle of the first stage that the best value of variable \( V_1 \) after optimization is per unit. In the second row. Variable \( V_2 \) is optimized (according to the CPSO method) and its value is changed compared to its value in the previous stage from 1.13210 to 1.132146. By reviewing the values of variables at different stages, it can be seen that their value are close to the optimum point compared to the condition before using CPSO method. In the last row, the best values of the variables are obtained by changing the last variable (5 variables have the same amount of the previous stage) and these values are the final results of CPSO method.

### Table 3. Comparison of different variables using PSO and CPSO

<table>
<thead>
<tr>
<th>Tap 6-8</th>
<th>Tap 6-7</th>
<th>( Q_s ) (Mvar)</th>
<th>( Q_a ) (Mvar)</th>
<th>( V_{bus1} ) (p.u.)</th>
<th>( V_{bus2} ) (p.u.)</th>
<th>Variable value</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.007873</td>
<td>0.950339</td>
<td>0.618077</td>
<td>6.373648</td>
<td>1.132100</td>
<td>1.000</td>
<td>( PSO )</td>
<td></td>
</tr>
<tr>
<td>0.983739</td>
<td>0.931916</td>
<td>0.529989</td>
<td>5.491242</td>
<td>1.132146</td>
<td>1.000</td>
<td>( CPSO )</td>
<td></td>
</tr>
</tbody>
</table>

### Table 4. Observation of different variables’ changes to each other

<table>
<thead>
<tr>
<th>Tap 6-8</th>
<th>Tap 6-7</th>
<th>( Q_s ) (Mvar)</th>
<th>( Q_a ) (Mvar)</th>
<th>( V_{bus1} ) (p.u.)</th>
<th>( V_{bus2} ) (p.u.)</th>
<th>Variable value change in variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.007873</td>
<td>0.950339</td>
<td>0.618077</td>
<td>6.373648</td>
<td>1.132100</td>
<td>1.000</td>
<td>( V_{bus1} )</td>
</tr>
<tr>
<td>1.007873</td>
<td>0.950339</td>
<td>0.618077</td>
<td>6.373648</td>
<td>1.132146</td>
<td>1.000</td>
<td>( V_{bus2} )</td>
</tr>
<tr>
<td>1.007873</td>
<td>0.950339</td>
<td>0.618077</td>
<td>5.491242</td>
<td>1.132146</td>
<td>1.000</td>
<td>( Q_4 )</td>
</tr>
<tr>
<td>1.007873</td>
<td>0.950339</td>
<td>0.529989</td>
<td>5.491242</td>
<td>1.132146</td>
<td>1.000</td>
<td>( Q_5 )</td>
</tr>
<tr>
<td>1.007873</td>
<td>0.931916</td>
<td>0.529989</td>
<td>5.491242</td>
<td>1.132146</td>
<td>1.000</td>
<td>Tap6-7</td>
</tr>
<tr>
<td>0.983739</td>
<td>0.931916</td>
<td>0.529989</td>
<td>5.491242</td>
<td>1.132146</td>
<td>1.000</td>
<td>Tap6-8</td>
</tr>
</tbody>
</table>
6. CONCLUSION

In this paper, a reactive OPF is presented to solve the problem of the reactive optimal distribution. In the 13-bus sample system, load flow was done and PSO and CPSO were used to compare the optimal distribution of reactive power. Finally, CPSO method achieved the optimum answer faster and acceptable results were obtained for using this algorithm for optimizing distribution of reactive power. Also, by evaluating the answers of different variables with both methods, it was specified that CPSO has obtained more optimal answers for all variables. Therefore, CPSO has the ability to solve the problem of the reactive optimal distribution better than PSO method. Also, it has more optimal answers and better convergence characteristics.

REFERENCES:
[5] P.R.Sujin, Dr.T.Ruban Deva Prakash and M.Mary Linda, “Particle Swarm Optimization Based Reactive Power Optimization”, JOURNAL OF COMPUTING, VOLUME 2, ISSUE 1, JANUARY 2010, ISSN 2151-9617