Journal of Engineering and Applied Sciences 1(4): 288-292, 2006

Attractive and Repulsive Particle Swarm Optimization for Reactive Power Optimization

K. Lenin and M.R. Mohan
College of Engineering, Anna University, Chennai-600 025, India

Abstract: Reactive Power Optimization is a complex combinatorial optimization problem involving non-linear function having multiple local minima, non-linear and discontinuous constrains. This study presents Attractive and repulsive Particle Swarm Optimization (ARPSO) in trying to overcome the Problem of premature convergence. ARPSO is applied to Reactive Power Optimization problem and is evaluated on standard IEEE 30Bus System. The results show that ARPSO prevents premature convergence to high degree but still keeps a rapid convergence. It gives best solution when compared to Genetic Algorithm (GA) and Particle Swarm Optimization (PSO).

Key words: Attractive and repulsive, particle swarm, reactive power optimization

INTRODUCTION

The reactive power optimization problem has a significant influence on secure and economic operation of power systems. The reactive power generation, although itself having no production cost, does however affect the overall generation cost by the way of the transmission loss. A procedure, which allocates the reactive power generation so as to minimize the transmission loss, will consequently result on the lowest production cost for which the operation constraints are satisfied. The operation constraints may include reactive power optimization problem. The conventional gradient-based optimization algorithm has been widely used to solve this problem for decades. Obviously, this problem is in nature a global optimization problem, which may have several local minima and the conventional optimization methods easily lead to local optimum. On the other hand, in the conventional optimization algorithms, many mathematical assumptions, such as analytic and differential properties of the objective functions and unique minima existing in problem domains, have to be given to simplify the problem. Otherwise it is very difficult to calculate the gradient variables in the conventional methods. Further, in practical power system operation, the data acquired by the SCADA (Supervisory Control and Data Acquisition) system are contaminated by noise. Such data may cause difficulties in computation of gradients. Consequently, the optimization could not be carried out in many occasions. In the last decade, many new stochastic search methods have been developed for the global optimization problems such as simulated annealing, genetic algorithms and evolutionary programming.

A major problem with Evolutionary Algorithms (EAs) in multi-modal optimization is Premature Convergence (PC), which results in great performance loss and sub-optimal solutions. As far as GAs are concerned, the main reason for premature convergence is a too high selection pressure or a too high gene flow between population individuals. With PSOs the fast information flow between particles seems to be the reason for clustering of particles.

Diversity declines rapidly, leaving the PSO algorithm with great difficulties of escaping local optima. Consequently, the clustering leads to low diversity with a fitness stagnation as an overall result.

Recently R. Ursem has suggested a model called the Diversity-Guided Evolutionary Algorithm (DGEA)[1]. He redefines the traditional mutation operator, the Gaussian mutation, to be a directed mutation instead. The important issue is that this directed mutation, in general, increases the diversity, whereas normal Gaussian mutation is not likely to do this, because it simply adds random noise from some distribution with a mean of zero, normally $N(0; \sigma^2)$. Consequently, the DGEA applies diversity-decreasing operators (selection, recombination) and diversity-increasing operators (mutation) to alternate between two modes based upon a distance-to-average-point measure. The performance of the DGEA clearly shows its potential in multi-modal optimization. As[1]

Corresponding Author: K. Lenin, College of Engineering, Anna University, Chennai-600 025, India

rightfully pinpoints, the diversity measure is traditionally used to analyze the evolutionary algorithms rather than guide them. We are great believers of adaptive controlling; that measuring and using different properties of the swarm/population while running, adds significant potential to the algorithm. We have therefore adopted the idea from R. Ursem with the decreasing and increasing diversity operators used to control the population into the basic PSO model. We find, it is a natural modification of the PSO and the idea behind it is surprisingly simple. The modified model uses a diversity measure to have the algorithm alternate between exploring and exploiting behavior. We introduce two phases attraction and repulsion. By measuring the diversity we let the swarm alternate between these phases. As long as the diversity is above a certain threshold d_{low} the particles attract each other. When the diversity declines below d_{low} the particles simply switch and start to repel each other until the threshold d_{high} is met. With this simple scheme we obtain our modified model, which we have chosen to call the ARPSO model-the attractive and repulsive PSO.

Problem formulation: The objective of the reactive power optimization problem is to minimize the active power loss in the transmission Network as well as to improve the voltage profile of the system. Adjusting reactive power controllers like Generator bus voltages, reactive Power of VAR sources and transformer taps performs reactive Power scheduling.

$$\min P_L = \sum_{i=1}^{NB} P_i(X, Y, \delta) \quad (1)$$

Subject to

I) The control vector constraints

$$X_{min} \le X \le X_{max} \quad (2)$$

ii) The dependent vector constraints

$$Y_{min} \le Y \le Y_{max} \quad (3)$$

and

iii) The power flow constraint

$$F(X, Y, \delta) = 0 \quad (4)$$

where

$$X = [V_G, T, Q_C] \quad (5)$$

$$Y = [Qg, V_L, I] \quad (6)$$

NB- Number of buses in the system.

δ- Vector of bus phase angles

P^i- Real Power injection into the i^{th} bus

V_G- Vector of Generator Voltage Magnitudes

T- Vector of Tap settings of on load Transformer Tap changer.

Q_C- Vector of reactive Power of switchable VAR sources.

V_L- Vector of load bus Voltage magnitude.

I- Vector of current in the lines.

P_L- Vector of current in the lines.

Basic PSO model: The basic PSO model consists of a swarm of particles moving in an n-dimensional, real valued search space of possible problem solutions[2,3]. For the search space, in general, a certain quality measure, the fitness, is defined making it possible for particles to compare different problem solutions. Every particle has a position vector x and a velocity vector v. Moreover, each particle contains a small memory storing its own best position seen so far p and a global best position g obtained through communication with its fellow neighbor particles. This information flow is obtained by defining a neighborhood topology on the swarm telling particles about immediate neighbors.

The intuition behind the PSO model is that by letting information about good solutions spread out through the swarm, the particles will tend to move to good areas in the search space. At each time step t the velocity is updated and the particle is moved to a new position[4]. This new position is simply calculated as the sum of the previous position and the new velocity:

$$\bar{x}(t+1) = \bar{x}(t) + \bar{u}(t+1). \quad (7)$$

The update of the velocity from the previous velocity to the new velocity is, as implemented in this study, determined by:

$$\acute{u}(t+1) = \omega.\acute{u}(t) + \phi_1(p(t) - \bar{x}(t)) + \phi_2(p(t) - \bar{x}(t)), \quad (8)$$

where ϕ_1 and ϕ_2 are real numbers chosen uniformly and at random in a given interval, usually [0,2]. These values determine the significance of p(t) and g(t), respectively. The parameter ω is the inertia weight and controls the magnitude of the old velocity $\acute{u}(t)$ in the calculation of the new velocity $\acute{u}(t+1)^2$.

The modified model-ARPSO: We define the attraction phase merely as the basic PSO algorithm. The particles will then attract each other, since in general they attract each other in the basic PSO algorithm because of the information flow of good solutions between particles[5]. We define the second phase repulsion, by inverting the velocity-update formula of the particles:

$$\bar{v}(t+1)=\omega\bar{v}(t)-\phi_1(\bar{p}(t)-\bar{x}(t))-\phi_2(\bar{g}(t)-\bar{x}(t)). \quad (9)$$

In the repulsion phase the individual particle is no longer attracted to, but instead repelled by the best known particle position vector g(t) and its own previous best position vector p(t).

In the attraction phase the swarm is contracting and consequently the diversity decreases. When the diversity drops below a lower bound, d_{low}, we switch to the repulsion phase, in which the swarm expands due to the above inverted update-velocity formula (9). Finally, when a diversity of d_{high} is reached, we switch back to the attraction phase. The result of this is an algorithm that alternates between phases of exploiting and exploring-attraction and repulsion-low diversity and high diversity. The pseudo-code for the ARPSO algorithm is shown below,

```
Program PSO
 init();
 while not done do
      Set direction();           // new!
      Update velocity();
      New position();
      Assign fitness();
      Calculate diversity();  // new!

Function set direction
  If (dir > 0 &-&- diversity < dLow) dir = -1;
 If (dir < 0 &-&- diversity > dHigh) dir = -1;
```

The first of the two new functions, set direction determines which phase the algorithm is currently in, simply by setting a sign-variable, dir, either to 1 or -1 depending on the diversity. In the second function, calculate diversity, the diversity of the swarm (in the pseudo-code stored in the variable diversity), is set according to the diversity-measure:

$$\text{diversity}(S)=\frac{1}{|S|.|L|}.\sum_{i=1}^{|S|}\sqrt{\sum_{j=1}^{N}(P_{1j}-\bar{P}_j)^2}, \quad (10)$$

where S is the swarm, |S| is the swarmsize, |L| is the length of longest the diagonal in the search space, N is the dimensionality of the problem, P_{ij} is the j'th value of the i'th particle and P_j is the j'th value of the average point P. Note that this diversity measure is indipendent of swarmsize, the dimensionality of the problem as well as the search range in each dimension.

Table 1: Optimal control values

	ARPSO	PSO	GA
VG1	1.05	1.06	1.05
VG2	1.03	1.04	1.04
VG3	1.01	1.01	1.02
VG4	1.01	1.02	1.00
VG5	1.07	1.09	1.08
VG6	1.08	1.08	1.08
T_1	0.99	0.98	0.97
T_2	0.95	0.95	0.94
T_3	1.00	1.00	1.00
T_4	0.94	0.93	0.93

Table 2: Parameter sensitivity analysis of IEEE 30 (100 trails)

Method	Compared item	IEEE 30 bus	Time (sec)	Iterations
ARPSO	Min. loss	9.4769	8.227	178
	Avg. loss value	9.4792		
PSO	Min. loss	9.4911	12.425	225
	Avg. loss value	9.5001		
GA	Min. loss	9.4971	15.771	250
	Avg. loss value	9.5114		

Table 3: Parameter sensitivity analysis of IEEE 30 (100 trails)

W_{max} / W_{min}		C_i: 1.0	1.5	2.0	2.5	3.0
0.9	Avg.	9.4799	9.4827	9.4827	9.4827	9.4827
0.4	Min	9.4769	9.4818	9.4817	9.4817	9.4817
2.0	Avg.	9.4827	9.4826	9.4826	9.4823	9.4825
0.9	Min	9.4818	9.4818	9.4818	9.4817	9.4817
2.0	Avg.	9.4827	9.4827	9.4827	9.4827	9.4827
0.4	Min	9.4819	9.4819	9.4819	9.4819	9.4819

W = Weight function for velocity of agent, C_i = Weight co-efficient for each term

Finally, the velocity-update formula, Eq. (9) is changed by multiplying the sign-variable direction to the two last terms in it. This decides directly whether the particles attract or repel each other:

$$\bar{v}(t+1)=\omega\bar{v}(t)-\phi_1(\bar{p}(t)-\bar{x}(t))-\phi_2(\bar{g}(t)-\bar{x}(t)). \quad (11)$$

Algorithm for RPO using ARPSO: The proposed RPO algorithm using the ARPSO can be expressed as follows:

Step 1: Initial searching points and velocities of agents are generated.

Step 2: Ploss to the searching points for each agent is calculated using the load flow calculation. If the constraints are violated, the penalty is added to the loss (evaluation value of agent).

The fitness function of each particle is calculated as:

$$f_n = P^n_L + \alpha\sum_{j=1}^{NG}Q^{lim,n}_{G,j} + \beta\sum_{j=1}^{NL}V^{lim,n}_{L,j} \; ; n = 1,2..,N_n \cdots \quad (12)$$

α, β	=	penalty factors
P^n_L	=	total real power losses of the n-th particle

Table 4: Line data-30-bus system

Branch No.	From	To	R(p.u)	X(p.u)	Y/2(p.u)	Line phase angle limit (deg.)
1	2	1	0.0192	0.0575	0.0264	5.0
2	1	3	0.0452	0.1852	0.0204	17.0
3	2	4	0.0570	0.1737	0.0184	7.0
4	3	4	0.0132	0.0379	0.0042	3.5
5	2	5	0.0472	0.1983	0.0209	15.0
6	2	6	0.0581	0.1763	0.0187	7.0
7	4	6	0.0119	0.0414	0.0045	2.5
8	5	7	0.0460	0.1160	0.0102	5.5
9	6	7	0.0267	0.0820	0.0085	6.0
10	6	8	0.0120	0.0420	0.0045	2.0
13	9	11	0.0000	0.2080	0.0000	4.0
14	9	10	0.0000	0.1100	0.0000	4.0
16	12	13	0.0000	0.1400	0.0000	4.0
17	12	14	0.1231	0.2559	0.0000	5.0
18	12	15	0.0662	0.1304	0.0000	3.0
19	12	16	0.0945	0.1987	0.0000	4.0
20	14	15	0.2210	0.1997	0.0000	2.5
21	16	17	0.0824	0.1932	0.0000	2.0
22	15	18	0.1070	0.2185	0.0000	2.5
23	18	19	0.0639	01292	0.0000	1.5
24	19	20	0.0340	0.0680	0.0000	3.0
25	10	20	0.0360	0.2090	0.0000	4.0
26	10	17	0.0324	0.0845	0.0000	2.0
27	10	21	0.0348	0.0749	0.0000	2.0
28	10	22	0.0727	0.1499	0.0000	2.0
29	21	22	0.0116	0.0236	0.0000	1.5
30	15	23	0.1000	0.2020	0.0000	3.0
31	22	24	0.1150	0.1790	0.0000	3.5
32	23	24	0.1320	0.2700	0.0000	3.0
33	24	25	0.1885	0.3292	0.0000	2.0
34	25	26	0.2544	0.3800	0.0000	2.0
35	25	27	0.1093	0.2087	0.0000	2.5
37	27	29	0.2198	0.4153	0.0000	3.5
38	27	30	0.3202	0.6027	0.0000	5.0
39	29	30	0.2399	0.4533	0.0000	4.5
40	8	28	0.0636	0.2000	0.0214	4.0
41	6	28	0.0169	0.0599	0.0065	3.0

Table 5: Transformer data-30-bus system

Branch No.	From	To	R(p.u)	X(p.u)	Tap	Tap max	Tap min	Tap step
11	6	9	0.0000	0.2080	1.0155	1.1000	0.9000	0.0250
12	6	10	0.0000	0.5560	0.9629	1.1000	0.9000	0.0250
15	4	12	0.0000	0.2560	1.0129	1.1000	0.9000	0.0250
36	28	27	0.0000	0.3960	0.9581	1.1000	0.9000	0.0250

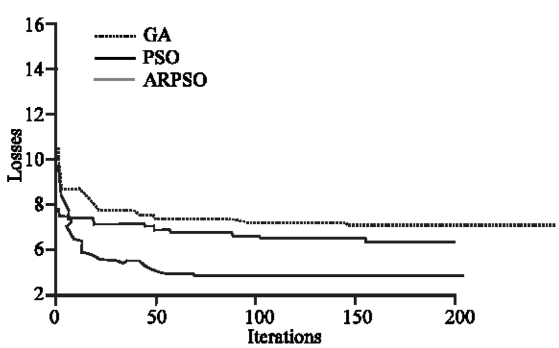

Fig. 1: Comparative study of Convergence characteristics of IEEE 30 Bus system with ARPSO, PSO and GA

$$Q_{G,j}^{lim,n} = \begin{cases} Q_{G,min} - Q_{G,j}^{n} & \text{if } Q_{G,j}^{n} < Q_{G,min} \\ Q_{G,j}^{n} - Q_{G,max} & \text{if } Q_{G,j}^{n} > Q_{G,max} \end{cases} \quad \ldots (13)$$

and

$$V_{L,j}^{lim,n} = \begin{cases} \left| V_{L,j}^{n} \right| - V_{L,max}, & \text{if } \left| V_{L,j}^{n} \right| > V_{L,max} \\ 0 & \text{otherwise} \end{cases} \quad \ldots (14)$$

Step 3: Pbest is set to each initial searching point. The initial best evaluated value (loss with penalty) among pbests is set to gbest.

Step 4: New velocities are calculated using Eq. (7).

Step 5: Update the velocity from previous velocity to the new velocity using Eq. (8).

Step 6: To new function applied.

i.setdirection

ii. calculateDiversity to control swarm.

Step 7: Ploss to the new searching points and the evaluation values are calculated.

Step 8: If the evaluation value of each agent is better than the previous pbest, the value is set to pbest. If the best pbest is better than gbest, the value is set to gbest. All of gbests are stored as candidates for the final control strategy.

Step 9: If the iteration number reaches the maximum iteration number, then stop. Otherwise, go to Step 4. If the voltage and power flow constraints are violated, the absolute violated value from the maximum and minimum boundaries is largely weighted and added to the objective function (1) as a penalty term. The maximum iteration number should be determined by pre-simulation. As mentioned below, PSO requires less than 100 iterations even for large-scale problems.

SIMULATION RESULTS

NB = 30, NL = 41, NG = 6, NTR = 4 Population size = 50
d_{low} = 0.01, d_{high} = 0.1

NOMENCLATURE

NB = total no. of buses, NL = total no. of load buses,
NG = total no. of generator buses
TR = total no. of transformers, VG = generator voltage V_g is a vector of generator bus voltages.
Q_s is a vector of switchable VAR sources and T is a vector of tap settings of on-load tap changing (OLTC) of transformers.
Q_g is a vector of reactive power generations of the generator buses and V_L is a vector of load bus voltages.
Table 1, 2, 3 are simulation Tables,
4,5 are data Tables.

CONCLUSION

In this study ARPSO algorithm has been developed for determination of global optimum solution for reactive power optimization problem. The performance of the proposed algorithm demonstrated through its evaluation on IEEE 30 bus power system shows that ARPSO is able to undertake global search with a fast converges rate and a future of robust computation. From the simulation study it has been found that ARPSO converges to the global optimum than PSO and GA.

REFERENCES

1. Ursem, R.K. Diversity guided evolutionary algorithms, in submission for the problem solving from nature conference, (PPSN 7).
2. Fukuyama, Y., 2001. A Particle Swarm Optimization for Reactive Power and voltage control in Electric Power System, IEEE Trans. on Power Systems, pp: 87-93.
3. Jakob S. Vesterstrom and Jacques Riget, 2002. A Diversity-Guided Particle Swarm Optimizer the ARPSO, EVALife Technical report no.
4. Fukuyama, Y., *et al.*, 2001. Practical Distribution StateEstimation Using Hybrid Particle Swarm Optimization, Proc. of IEEE Power Engineering Society Winter Meeting, Columbus.
5. Eberhart, R.C. and Y. Shi, 2000. Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization, Proc. of CEC.