Modified Particle Swarm Optimization

Swati Agrawal[1], R.P. Shimpi[2]

[1] Aerospace Engineering Department, IIT Bombay, Mumbai, India, swati.agrawal@iitb.ac.in
[2] Aerospace Engineering Department, IIT Bombay, Mumbai, India, rpshimpi@aero.iitb.ac.in

1. Abstract

Particle Swarm Optimization (PSO) is a very popular optimization technique, but it suffers from a major drawback of a possible premature convergence i.e. convergence to a local optimum and not to the global optimum. This paper attempts to improve on the reliability of PSO by addressing the drawback. This problem of premature convergence is more probable with the problems, which have the global optimum surrounded by the positions returning bad function values as these regions remain unexplored and swarm can get stuck into some local optimum. In the present paper, a modified particle swarm optimization is proposed to address this problem. During each iteration cycle, while deciding new positions, some particles will be chosen to give weightage to the worst solutions instead of good solutions. It will enable them to exploit the region for a probable global optimum. This modified method would free PSO from local optimum solutions; enable it to progress towards the global optimum searching over wider area. So the probability, of not getting trapped into local optima gets enhanced which gives better assurance to the achieved solution. Numerical experiments on the benchmark functions have been discussed in the paper.

2. Keywords: Particle Swarm Optimization, premature convergence, exploration, function value

3. Introduction

Particle Swarm Optimization (PSO) is one of the very popular population based methods which is inspired by social behavior of birds and fish [1]. PSO depends directly upon function values rather than derivative information with some random number association with it, which provides stochastic nature. Thus there are lesser chances of entrapment but particles can converge prematurely, majorly in the case of high dimensionality [2]. The main reason for premature convergence is that the particles in the population are highly affected by particle's personal best and global best, especially in the later stage of iterations. This similar kind of information flow between particles during optimization can result into similar kind of particles (loss in diversity), which increases their chances of being trapped in one of the local optima [3]. In particle swarm optimization, if particles are converging into a local optimum in the initial time period of optimization then there is a chance of getting out due to inertial velocity (momentum resulting from the fraction of velocity carried over from the previous iteration), but with time, velocity decreases due to stagnation. There is no mechanism of escaping in the basic algorithm [2].
Normally in PSO, exploration is possible in the initial phase of simulation due to high velocities but after sufficient time, as mentioned before, the velocities become smaller and even at their expected rate of decrease, the nearest solution to that space is also almost impossible to approach [4]. Several efforts have been made to address the problem as GCPSO [4] and then its extension MPSO [5], inertia weight PSO [6], Constriction factor [7], FIPSO [8], APSO [9] etc. These mechanisms still do not address the problem that for some complex functions where global optimum lies nearby the positions returning bad function values (bad positions), as in all the algorithms, particles move towards the best positions only. With a problem having multiple local optima, there are good chances of particles overflying the region containing global best and converge to a local best position. It is proposed here that this problem can be addressed by methods such as:

(1) Make the velocity steps very small, such that particles cannot overfly the region but then it will take a lot of time to explore the whole region and solutions will considerably depend upon initialization of particles.
(2) Update the velocity equation so that particles explore such regions. For this, one method can be, to increase randomness but then again, the time taken to solve the problem will increase considerably. So it essentially defeats the main advantage of PSO over other evolutionary strategies. Other method can be to update the velocity equation in such a manner that it does not rely only on the best particles.

We are proposing an algorithm which updates velocity equation using "Worst Particles" also, not just the "Best Particles". The paper settings are as follows: The modified PSO Algorithm in section 4, Experimental Results in Section 5, Conclusion in Section 6.

4. The modified PSO algorithm

4.1 Defining optimization problem and brief introduction to Inertia weight PSO

In mathematical terms, optimization of unconstrained problem is the minimization or maximization of a function (called objective function or fitness function). All problems here will be converted into minimization problem. So that the optimization could be written as

$$\text{minimize } f(\vec{x}) \text{ for } x \epsilon R^n \quad (1)$$

where $\vec{x}$ is the vector of variables, also known as unknowns or parameters; f is the objective function with variables x to be optimized. For swarmsize s, a problem with n-dimensions can be defined by a vector with n-dimensions, where i^{th} particle is defined as $\overrightarrow{x_i} = [x_{i1}, x_{i2}, x_{i3}, \dots\dots\dots\dots, x_{in}]$. Each particle keeps memory of its own personal best position Pbest based on its fitness value (fPbest). Particles also remember the best position of the whole swarm so far (Gbest) and its best value (fGbest) [3].

In inertia weight PSO, proposed by Eberhart and Shi, for each iteration, positions of particles are updated depending on the current Pbest and Gbest positions and previous velocity. In this method, inertia weight varies linearly with iterations from large to small value, so PSO will have more exploration in the beginning and then exploitation in the later stage [6].

4.2 Methodology of proposed modified PSO

The proposed algorithm here, considers the worst position also along with the best positions, so we keep track of particle's worst and global worst positions as we do for the best positions in normal PSO. As the name suggests, worst particle here, will be the particle having maximum function value. This modification can be applied to any of the PSO variants, which will be named here after as "base PSO". In this paper we are using inertia weight PSO as base PSO. Here, in each iteration, s1 particles are selected and named as "bad particles"; others are "good particles". There can be many methods for selecting "bad particles". For these "bad particles", velocity is updated using particle's worst and global worst positions. Other particles will follow the base PSO's velocity update rules. Here particles, going towards worst positions can explore the region nearby the bad function values during the run. There is possibility that these "bad particles" find good positions during their search. Then they will transform into the "good particles" and attract the other particles also towards them as they are ruled by the best ones.

In this paper we are choosing particles already performing worse than others as "bad particles" in each iteration and get velocity update by worst positions. As the particles which are already performing bad, do not participate much into the velocity update of whole swarm. Thus we still contain the essence of "base PSO". Results of numerical experiments on benchmark functions can be compared for both, with and without modification. For this modified method, results should not get much worse than we get from the base PSO. Though for some complex problems results can get better due to its better exploration abilities and there is better assurance for the results.

4.3 Implementation of modified PSO

Equation of velocity update for modified PSO is as follows; for i^{th} particle and j^{th} iteration with total p iterations

$$v_{id}(\mathrm{j}+1) = w_{now} * v_{id}(\mathrm{j}) + c1 * r1 * \mathrm{k1} * \left(p_{id}(\mathrm{j}) - x_{id}(j)\right) + c2 * r2 * \mathrm{k2} * \left(g_{id}(\mathrm{j}) - x_{id}(j)\right) + c3 * r3 * \mathrm{k3} * \left(p_{iw}(\mathrm{j}) - x_{id}(j)\right) c4 * r4 * \mathrm{k4} * \left(g_{iw}(\mathrm{j}) - x_{id}(j)\right) \quad (2)$$

$$x_{id}(\mathrm{j}+1) = x_{id}(\mathrm{j}) + v_{id}(j) \quad (3)$$

where,

$$w_{now} = w_{start} - j(w_{start} - w_{end})/p) \quad (4)$$

For i = 1,2,…..,s calculate the new velocity using equation (2) and update the position using equation (3). Here,

$c1, \mathrm{c3}$	is the cognitive acceleration coefficient,
$c2, \mathrm{c4}$	is the social acceleration coefficient,
g_{id}	is the best position vector for the entire swarm consisting of s particles (i.e. Gbest),
g_{iw}	is the worst position vector for the entire swarm consisting of s particles (i.e. Gworst),
$\mathrm{j}+1$	is the next iteration number,
j	is the current iteration number,
$\mathrm{k} = [\mathrm{k1}, \mathrm{k2}, \mathrm{k3}, \mathrm{k4}]$	is switch matrix. In this paper, for particles affected by best ones, $k = [1,1,0,0]$, which will switch for bad particles to $k = [0,0,1,1]$,

p_{id} is the best position vector for the i[th] particle so far (i.e. Pbest of the particle),
p_{iw} is the worst position vector for the i[th] particle so far (i.e. Pworst of the particle),
$r1, r2, r3$ & $r4$ are n –dimensional column vector whose elements are random numbers selected from a uniform distribution U (0,1),
x_{id} is the position of i[th] particle,
v_{id} is the velocity vector of i[th] particle,
w is the static inertia weight chosen in the interval [0,1]

Positions of particles are randomly initialized between [lb , ub]. Velocities are also initialized such as they lie between [$-V_{max}$, V_{max}] and subsequently trapped in the same velocity interval. Position of the particle will be trapped between [$-X_{max}$, X_{max}].

4.4 Work Flow for modified PSO

Work flow for modified PSO is as follows:

1. Initialize:
    a) Set parameters population s, $c1$, $c2$, no. of iterations (p), w_{start} and w_{end}
    b) Generate a swarm with s particles randomly distributed in the design domain S
    c) Generate the initial velocities randomly for each particle, $-V_{max} < V^i < V_{max}$
    d) Evaluate fitness values for each initial particle
    e) Find the global best position Gbest, g_{id}
    f) Find the global worst position Gworst, g_{iw}
2. Optimization process by velocity update:
    For each particle $i \epsilon S$
    If iteration $\leq$ p
    a) Update switch matrix, velocity and position
        i. Find the worst s1 particles (bad particles)
        ii. Update switch matrix k according for bad particles
        iii. Update particle's velocity $v_{id}(j+1)$ using equation 2
        iv. Update particle's position $x_{id}(j+1)$ using equation 3
    b) Update particle's best position
    Evaluate fitness value using coordinates $x_{id}(j+1)$ in design space
    If f $\left(x_{id}(j+1)\right) \leq f(p_{id}(\mathrm{j}))$, then
    Set $p_{id}(\mathrm{j}+1) = x_{id}(j+1)$
    Else
    Set $p_{id}(\mathrm{j}+1) = p_{id}(j)$
    c) If f $\left(p_{id}(j+1)\right) \leq f(g_{id}(\mathrm{j}))$, then
    Set $g_{id}(\mathrm{j}+1) = p(j+1)$
    Else
    Set $g_{id}(\mathrm{j}+1) = g_{id}(j)$
3. If stopping criteria is not satisfied then go to 2; otherwise terminate.

5. Results of numerical experiments on benchmark functions and discussion

For this modified PSO, results have been compared with the base PSO, which is inertia weight PSO here [6], to demonstrate its better exploration and problem solving abilities. In Table1, 20,000 and 3,00,000 function evaluations are used to compare inertia weight PSO and modified PSO. Discussion for the results is as follows:

- Results for Ackley and Rastrigin function after sufficient (300000) iterations, are noteworthy for better accuracy since this benchmark generally returns high function values due to stagnation of the swarm [2] in literature [2]. This means that modified PSO is able to take particles out of some local minimum and make them go towards the better values.

- Results for other benchmark functions are almost as good as base PSO.
  As discussed earlier, we are choosing particles returning bad values to go towards the worst positions, so it does not affect the performance for some functions.
- It can be seen that for less iterations, modified PSO gives higher mean values which shows its better exploration abilities but after sufficient iterations (here 300000), it returns better or almost same mean values as base PSO for each benchmark function. This gives better assurance for the results obtained by modified PSO as here exploration of wider region has been tried.
- Standard deviation, median, minimum values and maximum values are also almost same or better than the base pso, which signifies the whole distribution of the results.

Settings for experiments on all the benchmark functions here are as follows: c1=c2=2, w_{start}= 0.9, w_{end}= 0.4, s1=2, s= 20, number of runs =25. The mean of each benchmark function here signifies the mean of best values for all the trials (as best value in each trial is different due to randomness). The mean is almost constant after about 20 runs, similar thing applies to other statistical measures also.

Table1: Results of numerical experiments over benchmark functions

Benchmark Function	Statistical Measures	Iteration = 20000		Iteration = 300000	
		Base PSO	Modified PSO	Base PSO	Modified PSO
Ackley[2]	Mean	3.10E-04	7.10E-03	1.27E-07	3.35E-08
	Med	8.38E-05	1.50E-03	8.74E-08	2.17E-08
	STD	4.96E-04	1.21E-02	1.42E-07	3.48E-08
	Min	1.85E-05	2.79E-04	1.47E-08	6.50E-09
	Max	0.0011	0.0251	3.19E-06	8.41E-08
Rastrigin[2]	Mean	3.1957	11.7335	4.54E-04	8.11E-06
	Med	1.1534	10.7004	1.38E-04	2.00E-06
	STD	4.6402	9.8907	6.19E-04	1.49E-05
	Min	2.37E-04	1.2795	5.63E-08	1.61E-08
	Max	13.938	31.9717	1.45E-03	3.80E-05
Rosenbrock[2]	Mean	0.0282	3.9454	0*	0
	Med	0.0026	0.0132	0	0
	STD	0.0677	7.7833	0	0
	Min	2.42E-05	1.06E-04	0	0
	Max	0.3247	19.6889	0	0
Sphere[2]	Mean	3.79E-04	1.30E-03	2.40E-09	4.16E-10
	Med	1.64E-04	7.35E-04	2.40E-09	4.16E-10
	STD	4.36E-04	1.80E-03	2.25E-09	4.47E-11
	Min	9.31E-06	2.76E-05	8.15E-10	3.85E-10
	Max	1.00E-03	4.80E-03	3.99E-09	4.48E-10

*Values less than E-30 (10^{-30}) have been considered as 0.

Ackley[2] means that this function has been described in reference [2]; same thing is applicable to other functions also.

6. Conclusion

In this paper modified PSO is presented to address the problem of premature convergence. The proposed method gives weightage to the worst positions of particles also, unlike others. Numerical experiments done on some benchmark functions show that modified method has better exploration abilities than the base PSO so it gives better assurance of the achieved result being the global optimum.

7. References

[1] Kennedy J., Eberhart R., *Particle Swarm Optimization*, Proc. IEEE Int. Conf. on Neural Networks, pp. 1942-1948, 1995.

[2] Evers G.I. and Ghalia M. B., Regrouping Particle Swarm Optimization: *A new global optimization algorithm with improved performance consistency across benchmarks*, IEEE International Conference on Systems, Man and Cybernetics, 2009

[3] Yang B., Zhang Q., *Frame Sizing and Topological Optimization Using a Modified Particle Swarm Algorithm*, Second WRI Global Congress , Intelligent Systems (GCIS), 2010

[4] Van den Bergh F. and Engelbrecht A.P., *A new locally convergent particle swarm optimizer, In IEEE Conference on Systems*, Man, and Cybernetics, 2002

[5] Van den Bergh F., *An Analysis of Particle Swarm Optimizers. PhD thesis*, University of Pretoria, Pretoria, 2002

[6] Shi Y., Eberhart R., *Empirical study of particle swarm optimization*, Proc. Congress on Evolutionary Computation, pp. 1945-1950, 1999.

[7] Clerc M. and Kennedy J., *The particle swarm-explosion, stability, and convergence in a multidimensional complex space*, Evolutionary Computation, IEEE Transactions on, 6(1):58–73, 2002.

[8] Mendes R., Kennedy, J. and Neves J, *The fully informed particle swarm: simpler, maybe better, Evolutionary Computation*, IEEE Transactions on, 8(3):204–210, 2004.

[9] Xie X.F., Zhang W.J., and Yang Z.L., *Adaptive particle swarm optimization on individual level*, In 6th International Conference on Signal Processing, 2002