Applied Soft Computing 12 (2012) 353–359

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A new gradient based particle swarm optimization algorithm for accurate computation of global minimum

Mathew M. Noel

School of Electrical Engineering, VIT University, 2/36 4A East Cross Road, Gandhinagar, Vellore, Tamilnadu 632006, India

ARTICLE INFO

Article history:
Received 24 September 2010
Received in revised form 5 January 2011
Accepted 14 August 2011
Available online 1 September 2011

Keywords:
Particle swarm optimization (PSO)
Gradient descent
Global optimization techniques
Stochastic optimization

ABSTRACT

Stochastic optimization algorithms like genetic algorithms (GAs) and particle swarm optimization (PSO) algorithms perform global optimization but waste computational effort by doing a random search. On the other hand deterministic algorithms like gradient descent converge rapidly but may get stuck in local minima of multimodal functions. Thus, an approach that combines the strengths of stochastic and deterministic optimization schemes but avoids their weaknesses is of interest. This paper presents a new hybrid optimization algorithm that combines the PSO algorithm and gradient-based local search algorithms to achieve faster convergence and better accuracy of final solution without getting trapped in local minima. In the new gradient-based PSO algorithm, referred to as the GPSO algorithm, the PSO algorithm is used for global exploration and a gradient based scheme is used for accurate local exploration. The global minimum is located by a process of finding progressively better local minima. The GPSO algorithm avoids the use of inertial weights and constriction coefficients which can cause the PSO algorithm to converge to a local minimum if improperly chosen. The De Jong test suite of benchmark optimization problems was used to test the new algorithm and facilitate comparison with the classical PSO algorithm. The GPSO algorithm is compared to four different refinements of the PSO algorithm from the literature and shown to converge faster to a significantly more accurate final solution for a variety of benchmark test functions.

1. Introduction

The problem of finding the global minimum of a function with large numbers of local minima arises in many scientific applications. In typical applications the search space is large and multidimensional, prior information about the function is not available and traditional mathematical techniques are not applicable. Global optimization is a NP complete problem and heuristic approaches like genetic algorithms (GAs) and simulated annealing (SA) have been used historically to find near optimal solutions [1,2]. The particle swarm optimization (PSO) algorithm is a new sociologically inspired stochastic optimization algorithm introduced by Kennedy and Eberhart in 1995 [3–5]. The PSO algorithm is easy to implement, has few parameters, and has been shown to converge faster than traditional techniques like GAs for a wide variety of benchmark optimization problems [6,7]. Because of its simplicity, ease of implementation, and high convergence rates, the PSO algorithm has been applied to a variety of problems like evolving the structure as well as weights for artificial neural nets, power system optimization, process control, dynamic optimization, adaptive control and electromagnetic optimization.

Stochastic search algorithms like the PSO algorithm perform a biased random walk to explore the search space. A random search allows stochastic optimization algorithms to escape from local minima and explore flat regions but is computationally expensive and leads to slow convergence rates. On the other hand, deterministic algorithms like gradient-based techniques converge faster by using derivative information to identify a good search direction but get stuck in local minima. Also deterministic techniques perform poorly in minimizing functions for which the global minimum is surrounded by flat regions where the gradient is small. Therefore, a hybrid algorithm that uses deterministic techniques with high convergence rates to locate local minima while using stochastic techniques to escape from local minima and explore flat regions is of interest.

This paper describes a new gradient-based hybrid PSO algorithm, referred to as the GPSO algorithm, which uses the global exploration capability of the PSO algorithm to find new local minima and a gradient descent scheme for accurate local exploration. In the following sections, the PSO algorithm is reviewed and the new GPSO algorithm is presented and its performance is compared to that of the classical PSO algorithm for benchmark optimization problems.

A hybrid PSO algorithm that employs the PSO for exploration and the derivative free Nelder–Mead method for exploitation

E-mail address: mathew.m@vit.ac.in

1568-4946/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2011.08.037

(referred to as NM-PSO) is presented in [14]. NM-PSO starts with $3N+1$ particles where N is the dimension of the search space. The entire population is sorted according to their fitness and the best $N+1$ particles are updated using the Nelder–Mead Simplex algorithm. The remaining $2N$ particles are updated using the classical PSO algorithm. In [14] the NM-PSO is shown to converge faster to a more accurate final solution compared to the Nelder–Mead Simplex and PSO algorithms for a variety of low dimensional (≤ 10) test functions. The GPSO algorithm proposed in this paper is compared to the NM-PSO algorithm and shown to perform significantly better in Section 5.

2. The PSO algorithm

Consider the following general unconstrained function optimization problem:

$$\begin{array}{ll} \text{minimize} & f(x_1, x_2, \ldots, x_N) \\ \text{Where} & f : R^N \rightarrow R. \end{array} \quad (1)$$

Tentative solutions to this problem will be real vectors of length N. The PSO algorithm starts with a population of points (also referred to as particles) randomly initialized in the search space. The particles are moved according to rules inspired by bird flocking behavior. Each particle is moved towards a randomly weighted average of the best position encountered by that particle so far and the best position found by the entire population of particles according to:

$$\begin{array}{l} V_{id}(k+1) = V_{id}(k) + \phi_{1d}(P_{id}(k) - X_{id}(k)) + \phi_{2d}(G_d(k) - X_{id}(k)) \\ X_{id}(k+1) = X_{id}(k) + V_{id}(k) \end{array} \quad (2)$$

where $V_i = [V_{i1}\, V_{i2} \ldots V_{iN}]$ is the velocity for particle i; $X_i = [X_{i1}\, X_{i2} \ldots X_{iN}]$ is the position of particle i; ϕ_{1d} and ϕ_{2d} are uniformly distributed random number and are generated independently for each dimension; $P_i = [P_{i1}\, P_{i2} \ldots P_{iN}]$ is the best position found by particle i; $G = [G_1\, G_2 \ldots G_N]$ is the best position found by the entire population; N is the dimension of the search space and k is the iteration number. V_{id} is constrained to be bounded between V_{min} and V_{max} to prevent divergence of the swarm. When a particle position is updated the global best solution is replaced by the new solution if $f(X_i) < f(G)$. Thus when a particle is updated it has the opportunity to learn from all particles previously updated resulting in a high convergence rate.

In the PSO described above the global best solution found by the swarm so far G is used to update all particles. The use of G leads to communication bottlenecks in parallel implementations and might result in convergence to a local minimum if initial solutions are near a local minimum. Other approaches in which the swarm is divided into subpopulations and subpopulation bests are used to update particles have been explored [8], but use of multiple subpopulations result in reduced convergence rates. In this paper the classical global version will be used although the approach presented in this paper can be used with other population topologies as well. Performance analysis of the PSO algorithm shows that although the particles reach the vicinity of the global minimum faster than evolutionary computation techniques the convergence after that is slower [6,7]. This is because the particles are not constrained to take smaller steps as they near the global minimum.

3. Exploration versus exploitation: techniques to improve accuracy of the final solution

To accurately locate the global minimum with the PSO algorithm the step size has to be reduced when particles approach the global minimum. This is usually done by incorporating an inertial weight $0 < w(k) < 1$), as in (3) or a constriction coefficient, $x < 1$, as in (4) [9–11]

$$\begin{array}{l} V_{id}(k+1) = w(k)V_{id}(k) + \phi_{1d}(g_{id} - X_{id}(k)) + \phi_{2d}(G_{id} - X_{id}(k)) \\ X_{id}(k+1) = X_{id}(k) + V_{id}(k) \end{array} \quad (3)$$

$$\begin{array}{l} V_{id}(k+1) = \chi(V_{id}(k) + \phi_{1d}(g_{id} - X_{id}(k)) + \phi_{2d}(G_{id} - X_{id}(k)) \\ X_{id}(k+1) = X_{id}(k) + V_{id}(k) \end{array} \quad (4)$$

Use of inertial weight or constriction coefficient essentially narrows the region of search as the search progresses. In (3) the inertial weight $w(k)$ is initially set to a value of 1 but is reduced to 0 as the iterations progress. A value near 1 encourages exploration of new regions of the search space while a small value encourages exploitation or detailed search of the current region. In (4) the value of the constriction coefficient x is less than 1, which reduces the velocity step size to zero as the iterations progress. This is because when the velocity is repeatedly multiplied by a factor less than one (x or $w(k)$) the velocity becomes small and consequently the region of search is narrowed.

However, since the location of the global minimum is not known a priori, reducing the step size to a small value too soon might result in premature convergence to a local minimum. Also, a step size that is too small discourages exploration and wastes computational effort. Thus, to accurately locate the global minimum without getting trapped in local minima, the step size must be reduced only in the neighborhood of a tentative global minimum. However, there is no procedure that allows the values of the inertial weight or the constriction coefficient to be set or adjusted over the optimization period to manipulate the step size as needed to maintain an optimal balance between exploration and exploitation in the PSO algorithm. Proper choice of inertial weights or constriction coefficients is problematic because the number of iterations necessary to locate the global minimum is not known a priori. This paper presents an approach where the balance between exploration and exploitation is achieved by using the PSO algorithm without constriction coefficient or inertial weight for global exploration and a deterministic local search algorithm for accurate location of good local minima. This approach is advantageous because it allows for exploration of new regions of the search space while retaining the ability to improve good solutions already found.

4. A new gradient based PSO (GPSO) algorithm

In a classical gradient descent scheme [12] a point is chosen randomly in the search space and small steps are made in the direction of the negative of the gradient according to (5).

$$X(k+1) = X(k) - \eta \nabla(C(X(k))) \quad (5)$$

where η is the step size; $X(k)$ is the approximation to the local minimum at iteration k and $\nabla(C(X(k)))$ is the gradient of the cost function evaluated at $X(k)$. The gradient is the row vector composed of the first order partial derivatives of the cost function with respect to X. The gradient can be computed using a forward difference approximation for the partial derivatives, or by more advanced methods. With gradient descent, a large η will result in fast convergence of $X(k)$ towards the minimum but once in the vicinity of the nearest local minimum oscillations will occur as $X(k)$ overshoots the minimum. On the other hand, a small step size will result in slow convergence towards the minimum but the final solution will be more accurate. Since the choice of η is problematic, a random step size can be used; for example η uniformly distributed in an interval [0,0.5] might be used. Since the negative gradient always points in the direction of steepest decrease in the function, the nearest local minimum will be reached eventually. Since the gradient is zero at a local minimum, smaller steps will automatically be taken when a minimum is approached. Also, movement in a direction other

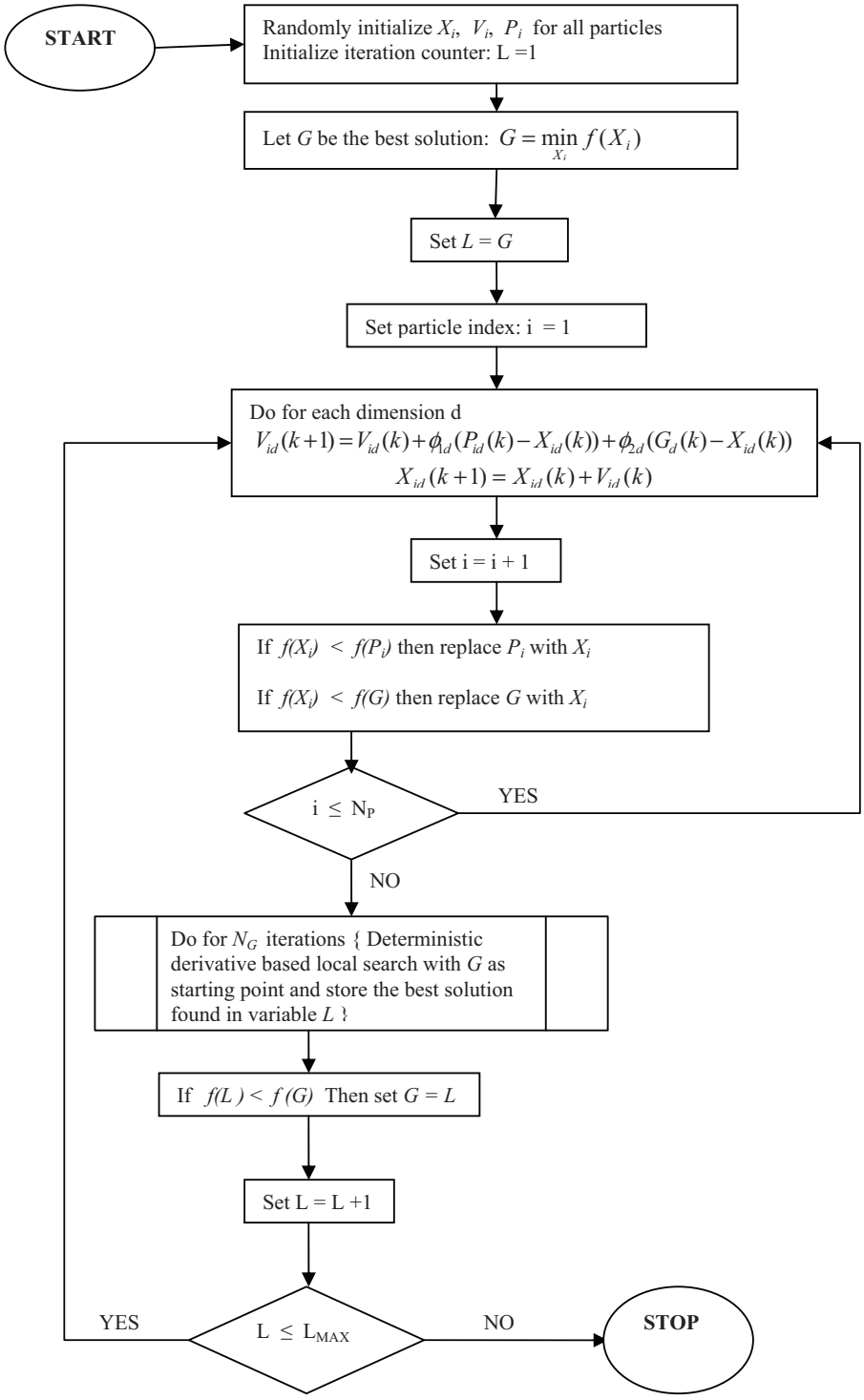

Fig. 1. Flowchart for the GPSO algorithm.

than the direction defined by the negative gradient will result in a smaller decrease in the value of the cost function. Thus, in the case of a function with a single minimum (unimodal function) the gradient descent algorithm converges faster than stochastic search algorithms because stochastic search algorithms waste computational effort doing a random search. For multimodal functions, the gradient descent algorithm will converge to the local minimum nearest to the starting point. In the new GPSO algorithm, the PSO algorithm is first used to approximately locate a good local minimum. Then a gradient based local search is done with the best solution found by the PSO algorithm as its starting point. If the best solution found by the PSO algorithm (G) has a larger cost than the final solution found by local search during the previous iteration (L), then L is used as the starting point for the local search. This ensures that the local search is done in the neighborhood of the best solution found by the GPSO in all previous iterations. Thus, the PSO algorithm is used to go near the vicinity of a good local minima and the gradient descent scheme is used to find the local minimum accurately. Next, this accurately computed local minimum is used as the global best solution in the PSO algorithm to identify

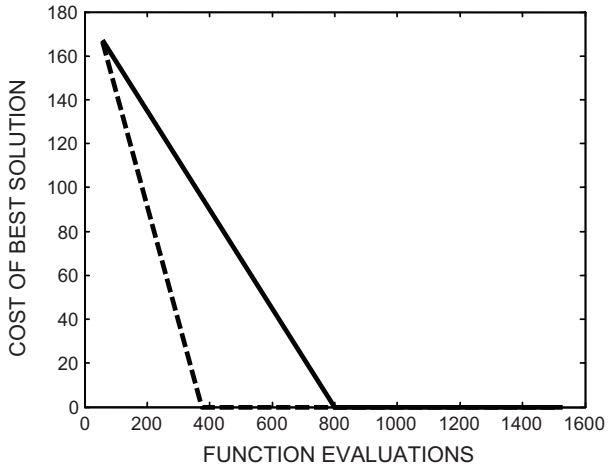

Fig. 2. Solid line shows convergence of GPSO algorithm with 50 local search iterations done every 10 PSO iterations. Dotted line shows convergence of GPSO algorithm with 5 local search iterations done every PSO iteration. Total number of PSO and local search iterations are 100 and 500 respectively in both cases. Rastrigin's test function was used.

still better local minima and the cycle is repeated. In this way the GPSO algorithm locates the global minimum by locating progressively better local minima. Fig. 1 below shows the flowchart for the GPSO algorithm. The number of particles N_P, the number of iterations L_{MAX} and the number of local search iterations N_G are defined at the start of the algorithm.

In this paper the quasi Newton–Raphson (QNR) algorithm was used to perform the local search in Fig. 1. The QNR algorithm optimizes by locally fitting a quadratic surface and finding the minimum of that quadratic surface. The QNR algorithm uses the second derivative or Hessian and makes use of information about the curvature in addition to the gradient information. The GPSO algorithm given in Fig. 1 can be generalized by using more powerful local search techniques to refine the best solution found in the PSO portion. Derivative free methods like Nelder–Mead Simplex method can be used for the local search when the objective function is not continuously differentiable.

Effect of performing the local search periodically instead of every PSO iteration as in Fig. 1 was considered. Fig. 2 shows that for multimodal functions distributing local search iterations evenly among the PSO iterations leads to high convergence rates. In the results presented the best solution found is plotted as a function of number of function evaluations rather than iterations. Convergence of different optimization algorithms cannot be done based on iterations because an algorithm can do more computation per iteration and appear to converge faster. Hence the number of function evaluations provides a more objective measure of convergence rates than number of iterations.

5. Comparison of GPSO and PSO algorithms

The performance of the GPSO algorithm was compared to the PSO algorithm by evaluating convergence and best solution found for important functions in the De Jong test suite of benchmark optimization problems. The problems provide common challenges that a generic optimization algorithm is expected to face, such as multiple local minima and flat regions surrounding global minimum. Table 1 shows the test functions used in this paper.

The Sphere and Ellipsoid test functions are convex and unimodal (single local minimum). The Rosenbrock test function has a single global minimum located in a long narrow parabolic shaped flat valley and tests the ability of an optimization algorithm to navigate flat regions with small gradients. The Rastrigin and Griewangk test

Table 1
Test functions used for comparison of GPSO and PSO algorithms.

Test function	Range of search	Equation
Sphere	$[-100,100]^N$	$f_1(x) = \sum_{i=1}^{N} x_i^2$
Ellipsoid	$[-100,100]^N$	$f_2(x) = \sum_{i=1}^{N} i x_i^2$
Rosenbrock	$[-100,100]^N$	$f_3(x) = \sum_{i=1}^{N-1} 100\left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2$
Rastrigin	$[-100,100]^N$	$f_4(x) = 10N + \sum_{i=1}^{N} \left(x_i^2 - 10\cos(2\pi x_i)\right)$
Griewangk	$[-600,600]^N$	$f_5(x) = 1 + \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos\left(\frac{x_i}{\sqrt{i}}\right)$

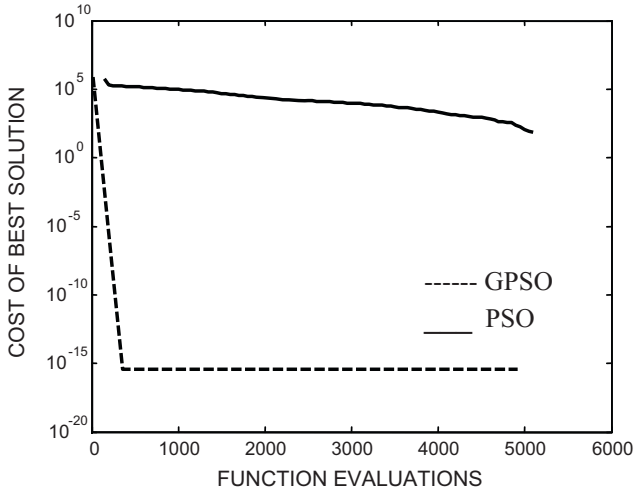

Fig. 3. Mean best solution versus number of function evaluations for the Sphere test function.

functions are highly multimodal and test the ability of an optimization algorithm to escape from local minima.

A population size of 20 was used for the GPSO and PSO algorithms. When the PSO algorithm was used with a constriction coefficient (4), the value of x was set to 0.7. Since the best solution found at a given iteration is a random variable affected by the random initialization and the stochastic nature of the optimization process, average values of the cost over 50 independent runs were used in evaluating and comparing the convergence performance of the algorithms. Figs. 2–6 show the best cost solution over 50 independent runs versus number of function evaluations for the GPSO and PSO algorithms.

The Sphere and Ellipsoid test functions are unimodal, convex and differentiable test functions without flat regions. For such test functions, the classical gradient descent algorithm will perform better than stochastic optimization techniques that waste computational effort by making a random search of the solution space as discussed in Section 4 (Figs. 3 and 4).

Fig. 5 shows the performance of the GPSO and PSO algorithms for the Rosenbrock test function, which has flat regions. Flat regions pose a challenge to deterministic local optimization algorithms since small changes about a point in a flat region do not lead to improvement. Stochastic optimization algorithms explore flat regions by performing a biased random search, however a pure random search is computationally costly and ineffective in

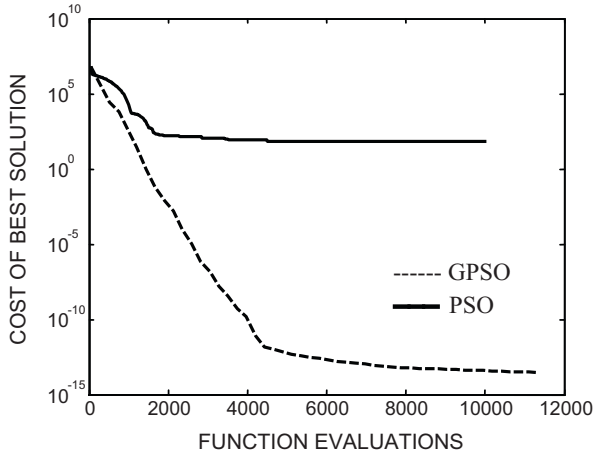

Fig. 4. Mean best solution versus number of function evaluations for the Ellipsoid test function.

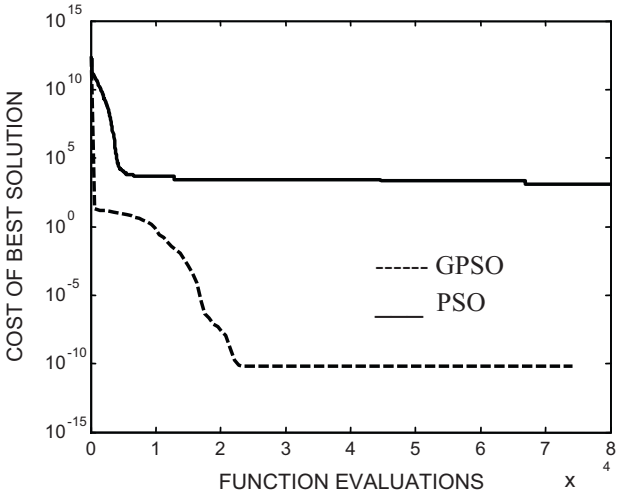

Fig. 5. Mean best solution versus number of function evaluations for the Rosenbrock's test function.

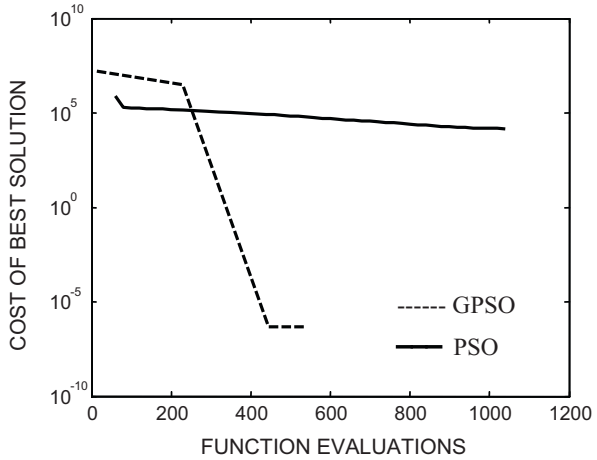

Fig. 6. Mean best solution versus number of function evaluations for the Rastrigin test function.

exploring regions with steep gradients since gradient information is not used.

Figs. 6 and 7 show the performance of the GPSO and PSO algorithms for the multimodal Rastrigin and Griewangk test

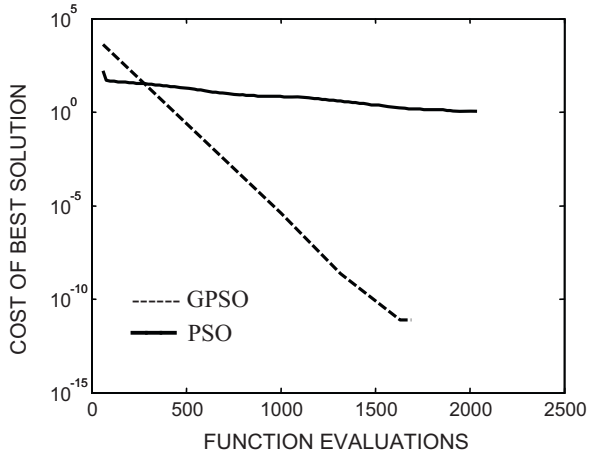

Fig. 7. Mean best solution versus number of function evaluations for the Griewangk's test function.

functions. These test functions have steep gradients throughout the solution space, and the GPSO converged significantly faster to a better solution than the PSO algorithm because the gradient descent portion of the GPSO algorithm uses gradient information to accurately find local minima.

The performance of the GPSO algorithm was compared to three different refinements of the PSO algorithm found in the literature [13]. In the classical PSO algorithm the parameters that determine convergence behavior are kept constant. In [13], three variants of the PSO algorithm (PSO-1, PSO-2 and PSO-3) with time varying parameters were introduced and shown to converge faster than the classical PSO algorithm.

In PSO-1 the inertial weight parameter used in (3) is reduced linearly from 0.9 at the start to 0.4 at the end of the iterations. In PSO-2 the inertial weight parameter used in (3) is set to change randomly as $0.5 + u/2$, where u is a uniformly distributed random number between 0 and 1. In PSO-3 the parameters ϕ_{1d} and ϕ_{2d} in (2) are reduced linearly. Table 2 shows the mean best solution found by the GPSO and three refinements of the PSO algorithm introduced in [13].

Table 2 shows that the GPSO algorithm converged faster to a more significantly more accurate final solution than the PSO algorithm variants with time varying coefficients for all test functions used. This is because good solutions (points in the neighborhood of deep local minima) found by the PSO are refined using a deterministic gradient based local search technique with high convergence rate avoiding costly random search.

The GPSO was then compared to a hybrid PSO algorithm that employs the PSO for exploration and the derivative free Nelder–Mead method for exploitation (NM-PSO) [14]. Table 3 shows the test functions used for comparison of the GPSO and NM-PSO algorithms.

Table 4 compares the average performance of the GPSO and NM-PSO algorithms for 100 independent runs. Table 4 shows that the GPSO algorithm converged to a significantly better solution than the NM-PSO algorithm [14] in fewer function evaluations for the Sphere, Rosenbrock, Griewangk, B2 and Zakharov test functions. For the Goldstein–Pierce and Easom test functions the GPSO converged to a more accurate solution but required more function evaluations. The NM Simplex algorithm used in the NM-PSO algorithm is computationally more efficient than the QNR algorithm used in the GPSO algorithms for lower dimensions. For higher dimensional problems the gradient based QNR algorithm performs significantly better than the NM Simplex algorithm. In accordance with this observation, the results presented in [14] indicate that

Table 2
Mean best solution found in 50 independent trials.

Test function	Dimension	Mean best solution			
		GPSO	PSO-1 [13]	PSO-2 [13]	PSO-3 [13]
Sphere	10	0	0.01	0.01	0.01
	20	0	0.01	0.01	0.01
	30	0	0.01	0.01	0.01
Rosenbrock	10	1.3553E−006	27.11	2.102	9.946
	20	6.2106E−011	51.56	28.17	17.94
	30	6.9424E−011	63.35	35.27	28.97
Rastrigin	10	0	2.069	4.63	2.268
	20	0	11.74	26.29	15.32
	30	0	29.35	69.72	36.23
Griewangk	10	0	0.067	0.066	0.054
	20	0	0.028	0.027	0.029
	30	0	0.016	0.017	0.019

Table 3
Test functions used for comparison of GPSO and NM-PSO algorithms.

Test function	Range of search	Equation
B2	$[-100,100]^2$	$B2(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_1) + 0.7$
Easom	$[-100,100]^2$	$ES(x) = -\cos(x_1)\cos(x_2) \times \exp(-((x_1-\pi)^2 + (x_2-\pi)^2))$
Goldstein & Price	$[-100,100]^2$	$GP(x) = \left[1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \times \left[30 + (2x_1 - 3x_2)^2 \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right]$
Zakharov	$[-600,600]^N$	$Z_N(x) = \sum_{j=1}^{N} x_j^2 + \left(\sum_{j=1}^{N} 0.5jx_j^2\right)^2 + \left(\sum_{j=1}^{N} 0.5jx_j^2\right)^4$

Table 4
Comparison of the performance of GPSO and NM-PSO algorithms.

Test function	Dim.	Mean function evaluations GPSO	Mean function evaluations NM-PSO [14]	Mean error NM-PSO [14]	Mean error GPSO
Sphere	3	130.84	291	5E−5	0
Rosenbrock	5	2119	3308	3E−5	1.18E−8
B2	2	187	240	3E−5	0
Goldstein & Price	2	5248.8	217	3E−5	5.33E−15
Zakharov	5	1195	1394	2.6E−4	0
Easom	2	6709	165	4E−5	1.33E−15

the performance of the NM-PSO algorithm degrades with increasing dimension. Tables 2 and 4 show the quality of final solution is significantly better for all test cases although the GPSO algorithm required more function evaluations in a minority of cases.

Costly deterministic local search is done only around the global best solution in case of the GPSO algorithm while in case of the NM-PSO algorithm roughly one third of the particles perform local search.

The NM-PSO algorithm requires $3N+1$ particles to solve an N dimensional problem making it computationally inefficient for higher dimensions. Thus, to optimize a test function of dimension 30, the NM-PSO requires 91 particles. On the other hand, Tables 1 and 2 show that good results can be obtained with the GPSO algorithm while using only 20 particles.

6. Conclusion

In this paper, a new hybrid optimization algorithm, referred to as the GPSO algorithm, that combines the stochastic PSO algorithm and the deterministic gradient descent algorithm is presented. The key ideas explored are the use of the PSO to provide an initial point within the domain of convergence of the quasi-Newton algorithm, synergistic interaction between local and global search and avoidance of wastage of computation time in stochastic search (when local minima are not present). In the GPSO algorithm, a gradient descent scheme is used for accurate local exploration around the best solution to complement the global exploration provided by the PSO. This approach allows an accurate final solution to be computed while retaining the ability to explore better solutions. The GPSO algorithm was compared to four different refinements of the PSO algorithm from literature and shown to perform significantly better for a variety of test functions. The GPSO algorithm avoids the use of constriction coefficients and inertial weights which, if improperly chosen can result in premature convergence to a local minimum for the PSO algorithm.

Parallel implementations of the GPSO algorithm will be considered in future work. One strategy for parallelization is to divide the population into subpopulations that explore the search space in parallel. The best solution found by each subpopulation can be communicated to a central computation node periodically. This population of best solutions can be evolved using the GPSO algorithm.

References

[1] D. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1989.

[2] T. Bäck, Evolutionary Algorithms in Theory and Practice, Oxford Univ. Press, New York, 1996.
[3] J. Kennedy, R.C. Eberhart, Swarm Intelligence, Morgan Kaufmann Academic Press, 2001.
[4] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machines and Human Science, 4–6 October, 1995, pp. 39–43.
[5] R.C. Eberhart, Y. Shi, Particle swarm optimization: applications and resources, in: Proceedings of the 2001 Congress on Evolutionary Computation, vol. 1, 27–30 May, 2001, pp. 81–86.
[6] P.J. Angeline, Evolutionary optimization versus particle swarm optimization: philosophy and performance differences, in: Evolutionary Programming VII, Lecture Notes in Computer Science 1447, Springer, 1998, pp. 601–610.
[7] R.C. Eberhart, Y. Shi, Comparison between genetic algorithms and particle swarm optimization, in: Evolutionary Programming VII, Lecture Notes in Computer Science 1447, Springer, 1998, pp. 611–616.
[8] J. Kennedy, R. Mendes, Population structure and particle swarm performance, in: Proceedings of the 2002 Congress on Evolutionary Computation, vol. 2, 12–17 May, 2002, pp. 1671–1676.
[9] R.C. Eberhart, Y. Shi, A modified particle swarm optimizer, in: Proceedings of the IEEE International Conference on Evolutionary Computation, vol. 6, 1998, pp. 9–73.
[10] M. Clerc, J. Kennedy, The particle swarm: exploration, stability and convergence in a multi-dimensional complex space, in: Proceedings of the IEEE International Conference on Evolutionary Computation, vol. 6, February, 2002, pp. 58–73.
[11] R.C. Eberhart, Y. Shi, Parameter selection in particle swarm optimization, in: Evolutionary Programming VII, Lecture Notes in Computer Science 1447, Springer, 1998, pp. 591–600.
[12] R. Horst, H. Tuy, Global Optimization—Deterministic Approaches, Springer-Verlag, New York, 1996.
[13] A. Ratanaweera, S.K. Halgamuge, H.C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Transactions on Evolutionary Computation 8 (June) (2004) 240–255.
[14] S.K. Fan, Y.C. Liang, E. Zahara, Hybrid simplex search and particle swarm optimization for the global optimization of multimodal functions, Engineering Optimization 36 (August (4)) (2004) 401–418.