Swarm Intelligence manuscript No.
(will be inserted by the editor)

Effects of Particle Fidelity in Particle Swarm Optimisation

Caleb Rascon · Prof. Barry Lennox

Received: ? / Accepted: ?

Abstract Keywords Particle fidelity · Particle Swarm Optimisation · PSO

1 Introduction

Particle Swarm Optimisation (PSO) has recently been increasing in popularity as a Black-Box optimisation algorithm (Ressom et al 2005; Kennedy and Eberhart 1995; Rascon et al 2008; Meissner et al 2006). An important reason for this is its simple structure which can be modified, to incorporate improved methods by which particles exchange information. For example, improvements in performance were seen when historical accounts were exchanged between particles (Veeramachaneni et al 2003). The strength of PSO relies on the flow of information throughout the swarm, which ensures that the global optimum is reached. To this effect, it is important to define which neighbours a particle can exchange information with. This exchange of information is referred to as particle topologies (Montes de Oca and Stützle 2008).

In (Meng and Jia 2008) it was stated that the two most popular variations of topologies between particles are the 'ring', in which the particle only has two neighbours, and the 'full', in which all the particles exchange information with every one another (Mendes et al 2004). The latter was very popular, as, when introduced, it gave promising results when applied to many PSO variations (Mendes et al 2004). The reason for the consistently good results was that the information flow throughout the swarm remained constant. It also provided the user with a simple topology, compared to the neighbourhood-based topologies that rely on the information gathered from only a handful of neighbours near the particle. However, further experimentation has found that the Full Topology PSO variation is too dependent on the type of solution space encountered (Montes de Oca and Stützle 2008) and that, in fact, the larger the neighbourhood, the more stagnant the particle movements become. This suggests that the use of small neighbourhoods is important in the development of a widely-applicable PSO variation.

Caleb Rascon
Control Systems Centre, University of Manchester. PO Box 88, Sackvile Street Building, Manchester, M60 1QD
E-mail: caleb.rascon@postgrad.manchester.ac.uk

The most popular definition of a neighbourhood, with which the particle exchanges information, is the nearest particles to it at any given time. This definition has been implicitly used since the inception of PSO, and there has been little mention of its limitations to the information flow throughout the swarm.

In this paper, a new type of 'neighbourhood' is defined. This neighbourhood is based on the concept of particle fidelity is introduced and the effect that this new definition has on the performance of PSO is explored. In the following section, a brief background on PSO is provided. This is followed in Section 3 with a brief definition of 'particle fidelity'. In Section 4, the effects that particle fidelity has on the performance of PSO are explored, with a set of popular benchmark solution spaces. Finally, conclusions are provided.

2 Background

Particle Swarm Optimisation (PSO) is a search algorithm that was introduced by James Kennedy and Russell Eberhart in 1995 (Kennedy and Eberhart 1995). It is based on the inner social behaviour of a flock or a school to find food.

A group of particles (or *swarm*) is randomly placed inside the solution space defined by an objective function. Each particle can 'move' towards different locations in the solution space, and each location is graded by the objective function. Every particle is able to remember the best-graded location it has found, and makes it known to a pre-defined number of neighbours. During each iteration, the velocity of each particle is modified by considering the best-graded location found by the particle and the best one found by its neighbours, i.e.

$$V_{x_i}(k+1) = V_{x_i}(k) + 2 * r * (pbest_{x_i} - present_{x_i}) + 2 * r * (gbest_{x_i} - present_{x_i}) \quad (1)$$

where k is the iteration index, V_{x_i} is the velocity of the particle in the direction x_i, $pbest_{x_i}$ is the best-graded location in direction x_i found by the particle, $gbest_{x_i}$ is the best-graded location in direction x_i found by the neighbours of the particle, $present_{x_i}$ is the current location of the particle in direction x_i, and r is a stochastic factor that prevents several particles being in the same location. The inclusion of r makes the particles 'spread out' in an *area*, rather than focus on a single point. This improves significantly the chances of finding the true global optimum. All the V_{x_i}s of all the particles are modified according to (1) until the best-graded location found by the whole swarm converges or the maximum number of iterations is exceeded.

PSO can incorporate the concept of a *time-decreasing inertia* (Shi and Eberhart 1998), which forcefully decreases velocities later in the search. This technique is in fact an implementation of the temperature decrease of a Simulated Annealing search (Černý 1985). Applying it to PSO results in an initial exploration of the whole solution space, pinpointing the area where the global optimum is suspected of being located, and then evolves into an exploitation of this area for the remainder of the search. It has been shown that using time-decreasing inertia in PSO provides faster and more accurate results than without (Shi and Eberhart 1998). It has also been shown that by a choosing an appropriate value for the inertia weight, and ensuring that the sum of the weights for each influential location is greater than 4, using time-decreasing inertia ensures convergence (Eberhart and Shi 2000).

3 Particle Fidelity

When a particle 'asks' its neighbours for the best location they have found, the different variations of the algorithm do not specify if the neighbours are those closest to the particle at that precise moment, or if they are the initial neighbours of the particle when the algorithm was initialised.

A particle, in the initial stages of the algorithm, can define its closest neighbours as its 'family', and only ask those particles throughout the search, rather than relying on the findings of the closest neighbours it has at each iteration. In this paper, the behaviour of only gathering information from the initial particle neighbourhood is referred to as *particle fidelity*.

4 Experiments & Results

To test whether particles being faithful has any impact on the performance of the algorithm, two variations of PSO were implemented: one in which the particles relied on the findings of the closest neighbours at each iteration (Neighbour PSO) and another in which the particles remained faithful to their respective families (Family PSO). Each method was applied 100 times to each of four different two-dimensional optimisation problems. The optimisation problems studied in this section have been popular in testing optimisation algorithms (Oltean 2003; Meissner et al 2006).

Optimisation Problem 1: De Jong's Sphere. (Jong 1975) This is a convex solution space that features a single wide peak. It is a relatively simple problem that is applied here as a frame of reference, as all optimisation algorithms are expected to be able to solve it. It is described as:

$$Z = -\sum_{d=1}^{D} x_d^2 \tag{2}$$

where D is the number of dimensions (2 in this case) and x_d is a dimension. The maximum is located at $x_d = 0$ for all d, with a value of 0. A graphical representation of De Jong's Sphere is shown in Figure 1(a).

Optimisation Problem 2: Rastrigin Function. (Rastrigin 1974)The Rastrigin Function features many local minima, but only one global optimum at its center. As shown in Figure 1(b), the shape of the Rastrigin Function is similar to that of De Jong's Sphere, but much 'bumpier'. It is described as:

$$Z = -10 \cdot D - \sum_{d=1}^{D} x_d^2 - 10cos(2\pi x_d) \tag{3}$$

It was modified such that the optimum was the maximum value of the function, with a value of 0, located at $x_d = 0$ for all d.

Optimisation Problem 3: Schaffer F6 Function. (Schaffer et al 1989) As shown in Figure 1(c), this function simulates a set of 'waves' similar to those that appear after throwing a rock into a pond. All the points in the top of each wave have values very similar to that of the global optimum, and the closer the wave is to the center, the closer these values are to the global optimum. This translates to having an infinite number of local minima to avoid. The function is described as:

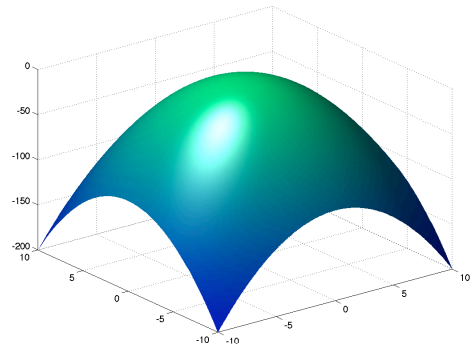
$$Z = -0.5 - \frac{sin^2\sqrt{x_1^2 + x_2^2} - 0.5}{1 + 0.01 \cdot (x_1^2 + x_2^2)} \tag{4}$$

It was modified such that its optimum was the maximum value of the function, with a value of 0, located at $x_d = 0$ for all d.

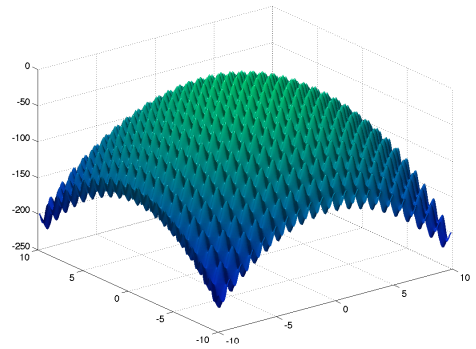
Optimisation Problem 4: Rosenbrock Function. (Rosenbrock 1960) The Rosenbrock function aims to 'trick' the optimisation algorithm into finding an area of local minima from which it is difficult to 'jump out' of and find the global optimum. It is described as:

$$Z = -\sum_{d=1}^{D-1} 100 \cdot (x_{d-1} - x_d^2)^2 + (1 - x_d)^2 \tag{5}$$

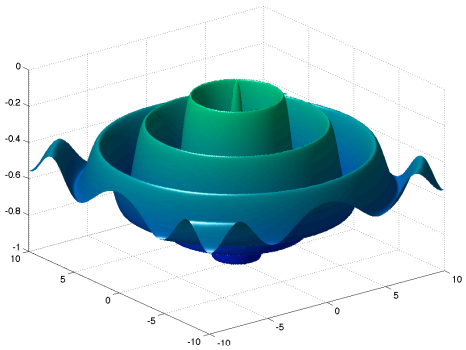
It was modified such that its optimum was the maximum value of the function, with a value of 0, located at $x_d = 1$ for every d. A graphical representation of this modified version of the Rosenbrok Function is shown in Figure 1(d).



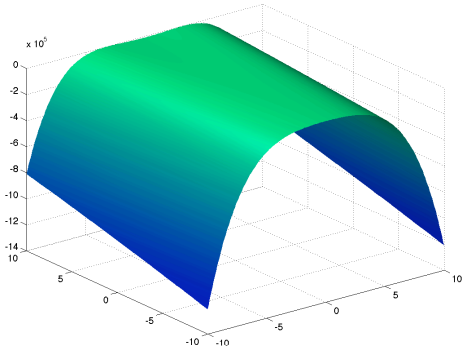
(a) De Jong's Sphere.



(b) Rastrigin Function.



(c) Schaffer F6 Function.



(d) Rosenbrock Function.

Fig. 1: Graphical representation of the solution spaces used in testing.

In the tests, the number of particles in the swarm was set to 10, with each particle having 5 neighbours/family members. The optimal value in all the problems was 0, and, because the accuracy of the MATLAB software was $2.2204 * 10^{-16}$, any point found with a fitness below this value was considered to be the global optimum and given the value of 0. Inertia was reduced from 1 to 0.2 in the first 300 iterations of the search. The number of iterations was limited to 30,000; if the maximum number of iterations was reached, the best solution found at this time was returned as the solution of the search.

The results when testing Neighbour PSO are shown in Table 1, and those using Family PSO are shown in Table 2.

Table 1: Results using Neighbour PSO.

	De Jong	Rastrigin	Schaffer F6	Rosenbrock
Mean Error from Optimum	0	0	0	$0.4342 * 10^{-7}$
Mean Error from Optimal Dimension Values	0	$0.0021 * 10^{-6}$	$0.0021 * 10^{-6}$	$0.7459 * 10^{-6}$
Mean Number of Iterations	559.7	1189.4	604.8	6096.1
Standard Deviation of Number of Iterations	24.9	4138.8	106.9	3461.6

Table 2: Results using Family PSO.

	De Jong	Rastrigin	Schaffer F6	Rosenbrock
Mean Error from Optimum	0	$0.1776 * 10^{-7}$	0	0
Mean Error from Optimal Dimension Values	0	$0.0865 * 10^{-8}$	$0.2230 * 10^{-8}$	0
Mean Number of Iterations	617.8	966.7	958	5664.5
Standard Deviation of Number of Iterations	34.8	2936.5	261.8	761.6

In all the tests, both PSO variations arrived at a value close to the optimal, and, as expected, the De Jong function was optimised consistently. However, because of the nature of the test functions, a close-to-optimal value may belong to a local optimum. An analysis of the mean error from the optimal values of each dimension shows that it decreased substantially when the particles were faithful to their family, from errors in the range of 10^{-6} to 10^{-8}.

Another factor that was considered was the number of iterations necessary to reach a global optimum. Although the mean number of iterations did not appear to deviate when using either of the two variations, the substantial decrease of the standard deviation when testing the Rastrigin and Rosenbrock function indicates that, when the particles were being faithful, the number of iterations needed to find the optimal solution tends to vary less from its mean. This suggests that, when applying Family PSO, the search is more likely to find an optimal solution in the expected number of iterations. When testing with the De Jong function, the change in the standard deviations between using the PSO variations is small. The only function in which Family PSO takes considerably more iterations to converge than Neighbour PSO is with the Schaffer F6 function, but Family PSO consistently obtains a better result than Neighbour PSO when being tested with that function.

5 Conclusions & Discussion

Particle neighbourhoods have been shown to be the PSO variations with the widest applicability, however, the definition of 'neighbourhood' has not been discussed. In this paper, a new definition of neighbourhood was introduced, referred to as 'family'. In the Family PSO variation proposed in this paper, a particle gathers information from only its initial family, regardless of their position in the solution space.

Tests showed that, when optimising functions with a high number of local optima, there is increased reliability and performance in the PSO search if the particles are faithful to their initial family. An explanation for this is that, because of their initial uniform distribution, all the particles will have a different family, resulting in all the particles communicating with each other. If one family finds an important location, the members of that family will communicate it to their respective families, and so forth. In addition, because of their fidelity, the flow of information remains constant, resulting in consistent findings. In the case of Neighbour PSO, there is a high probability that the neighbours of one particle are the same as another, as their close distance is the only factor that joins them, which can result in a high number of particles not communicating with the rest of the swarm.

References

Černý V (1985) Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. Journal of Optimization Theory and Applications 45(1):41–51

Eberhart RC, Shi Y (2000) Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 Congress on Evolutionary Computation, vol 1, pp 84–88 vol.1, DOI 10.1109/CEC.2000.870279, URL http://dx.doi.org/10.1109/CEC.2000.870279

Jong KAD (1975) An analysis of the behavior of a class of genetic adaptive systems. PhD thesis, University of Michigan, Ann Arbor, MI, USA

Kennedy J, Eberhart R (1995) Particle swarm optimization. Proceeding of the IEEE International Conference on Neural Networks IV:1942–1948

Meissner M, Schmuker M, Schneider G (2006) Optimized particle swarm optimization (opso) and its application to artificial neural network training. BMC Bioinformatics 7:125, DOI 10.1186/1471-2105-7-125

Mendes R, Kennedy J, Neves J (2004) The fully informed particle swarm: simpler, maybe better. Evolutionary Computation, IEEE Transactions on 8(3):204–210, DOI 10.1109/TEVC.2004.826074

Meng X, Jia L (2008) A new kind of pso–convergent fuzzy particle swarm optimization and performance analysis. In: Networked Computing and Advanced Information Management, 2008. NCM '08. Fourth International Conference on, vol 2, pp 102–107, DOI 10.1109/NCM.2008.222

Montes de Oca MA, Stützle T (2008) Convergence behavior of the fully informed particle swarm optimization algorithm. In: GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation, ACM, New York, NY, USA, pp 71–78, DOI http://doi.acm.org/10.1145/1389095.1389106

Oltean M (2003) Evolving evolutionary algorithms for function optimization. In: (et al) KC (ed) The 7th Joint Conference on Information Sciences, Association for Intelligent Machinery, North Carolina, vol 1, pp 295–298, URL http://citeseer.ist.psu.edu/640208.html

Rascon C, Lennox B, Marjanovic O (2008) Using lagged spectral data in feedback control using particle swarm optimisation. Proceedings of the UKACC Control 2008 Conference

Rastrigin L (1974) Systems of Extremal Control. Moscow: Nauka

Ressom H, Varghese R, Saha D, Orvisky E, Goldman L, Petricoin E, Conrads T, Veenstra T, Abdel-Hamid M, Loffredo C, Goldman R (2005) Particle swarm optimization for analysis of mass spectral serum profiles. Proceedings of the Genetic and Evolutionary Computation Conference pp 431–438

Rosenbrock HH (1960) An automatic method for finding the greatest or least value of a function. The Computer Journal 3(3):175–184, DOI 10.1093/comjnl/3.3.175, URL http://dx.doi.org/10.1093/comjnl/3.3.175

Schaffer JD, Caruana RA, Eshelman LJ, Das R (1989) A study of control parameters affecting online performance of genetic algorithms for function optimization. In: Proceedings of the third international conference on Genetic algorithms, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 51–60

Shi Y, Eberhart RC (1998) Parameter selection in particle swarm optimization. Proceedings of the 7th International Conference on Evolutionary Programming VII pp 591–600

Veeramachaneni K, Peram T, Mohan C, Osadciw LA (2003) Optimization using particle swarms with near neighbor interactions. In: Lecture Notes Computer Science, Springer Verlag, pp 110–121