The Impact of Inertia, Particle Increment & Global Increment on Swarms in a Dynamic Environment

Omar Sarhan

School of Computer Science & Informatics

University College Dublin

Abstract

The purpose of this paper and the experiments conducted herein is to assess the impact of variation in particle inertia (w), particle increment (c_1) and global increment (c_2), on the fitness of particles and there impact upon particle swarm optimization in a dynamic environment. In this paper a 3-dimensional parabolic function is utilized to illustrate an example of a dynamic environment. Results show that the impact of PSO in this environment will not be significantly enhanced by the alteration of w, c_1 or c_2.

1 Introduction

Kennedy and Eberhart formally introduced the idea of Particle Swarm Optimization (PSO) in their 1995 paper [1], draw on information and theories from a wide range of authors, from several different schools of thought; they arrive at the conclusion, that several key ideas are inherent to a collaborative system.

Kennedy and Shi [2] built on previous research, with the development of an inertia factor which is designed for "balancing the global search and local search" [2]. This idea of enabling a degree of flexibility in the way in which the PSO searches, is a technique by which the particles can get trapped by local maximum, and alter a swarm's explore-exploit dynamic.

The idea of dynamic environment for PSO is not a new idea, much work has been done in this field with regards to reworking the traditional PSO algorithm in order give it extra capability to work within a dynamic environment [3], (by the resetting of a particles at various stages and other methods). It should be noted there is quite a degree of overlap between the motives of this paper and [4], as both are dealing with a dynamic environment; however the means with which the swarm is dealt with are intrinsically different, by the alteration of the inertia and increment values.

The experimentation section of this paper will focus on testing various modifications of the cognitive (c_1) and Social (c_2) weightings as well as the inertia value (w) of the particles of the swarm to understand better the way in which different variants may possibly lead to an all-round more efficient standard PSO configuration.

2 Related Works

The idea of using a PSO algorithm to solve a dynamic problem has been researched quite widely [4] [6] [7], many authors see it as representing realistic real world challenges [4] [6].

Preliminary works in the field [4], Suggested two main modifications that could be made to the PSO algorithm in order to provide better swarm for tracking a dynamic global optima. The modifications are based on the idea of resetting each particle personal best position in the environment; which can be undertook in two different ways.

The first being that particles reset there *pbest* score at fixed points in time, such as after a set amount of evolutions. The second approach is for the reset to occur when there is a certain magnitude of change in the environment.

Overall [4] does show that there is a general benefit in searching for an optimum when both approaches are used, However, certain constraints such as the dynamic optima's velocity in the search space being not greater than the max velocity of the swarm, as well as the additional performance overheads inflicted upon the particle update algorithm are recognised as possible problems.

Another approach provided by [6], suggests the idea of an enhanced update function for each particles storage of its best fitness, whereby with the introduction of evaporation weighting to a particles previous best position will enable a particle, over the course of time to reset its *pbest* position. The TDPSO put forward by [6], does show a marked improvement over the standard PSO algorithm. The paper also gives guidelines regarding the most effective bounds for the evaporation factor.

The most recent paper in this field [7], looks into 3 factors dictating how a single swarm, acting as collaborative sub-swarms, can be tweaked in order to encourage greater adaptability to a dynamic environment. It should be noted that [7] is working within a multimodal dynamic environment which adds a greater level of complexity to search. In [7], particle interaction is altered in two ways.

Firstly, exclusion, in order to include the idea that at a local level particles will repel on another in order to stop two particles converging on a local maxima and thwarting the benefit which could be gained by achieving a more diverse *gbest*.

Secondly, an anti-convergence measure is employed in order to enhance the possibility of a higher global fitness being available. This measure is implemented by re-initialising the worst sub swarm once convergence of all swarms has occurred within a given area.

Finally, the use of quantum particles, particles initialized at random, in a uniformed way around the global best, is used and is seen by the author as similar to the notion of charged particles.

Overall, when compared to other approaches dealing with multimodal dynamic environments, the multi-pronged approach employed by [7] does show a degree of benefit and is a generally more effective at problem solving. In general there is a great deal of work in the use of PSO in a dynamic environment, however rather than alter the PSO algorithm, this paper attempts to understand the role of variables in a standard PSO algorithm and investigate whether it is possible to refine an optimisation by using different values.

3 The Particle Swarm Algorithm

The core components of the PSO algorithm that will be dealt with in this paper can be traced back to work in [1] and [2]. Prior to the initialization of a PSO algorithm two very important aspects must be defined. Firstly, in the search space in which the particles must be conceptualised into a series of dimensions, there must then be an evaluation function which will allow a particle to grade itself and deduce fitness value based on the values of its own dimensions.

Once the above have been successfully conceptualised, a swarm of particles (of a defined size) will be initialised at random points throughout the search space. The evaluation or fitness function can then be called to help every particle grade itself and establish a personal best (*Pbest*) position in the search space. This process will also help to evaluate which particle currently holds the highest personal best fitness. This particle will be known as the fittest particle in the search space and its location will be referred to as the global best (*Gbest).*

These global (g_i) and personal (p_i) maximums, alongside the previous velocity of the particle ($v_i(t)$), as seen in equation 1, will then be incorporated into a velocity function ($v_i(t+1)$) which will used to dictate the direction ($x_i(t+1)$) and velocity for each of particles (i) movement upon the next evolution of the swarm.

This cycle of updating of *Pbest* and *Gbest* (if necessary), recalculating each particle fitness and deciding the movement of each particle will continue until a desired condition has been reached.

In equation 1 there is as is also the incorporation of two random variables (r_1 & r_2) and two constant values (c_1 & c_2), which represent the particle and global increment (discussed at length in section) in the calculation of a particles new velocity.

$$v_i(t+1) = v_i(t) + c_1 r_1 (p_i - x_i(t)) + c_2 r_2 (g_i - x_i(t))$$
$$x_i(t+1) = x_i(t) + v_i(t+1)$$

Equation 1 - The Velocity and Locational Update Function of a Particle in a PSO (Without Inertia weight)

$$v_i(t+1) = \boxed{w} v_i(t) + c_1 r_1 (p_i - x_i(t)) + c_2 r_2 (g_i - x_i(t))$$
$$x_i(t+1) = x_i(t) + v_i(t+1)$$

Equation 2 - The Velocity and Locational Update Function of a Particle in a PSO (note Inertia weight *w*)

Equation 1 represents an early and basic approach of the PSO update function. Later works, particularly [2], develop the idea of an inertia function that would take account of the significance of a particles previous velocity when calculating its a new velocity.

For the purpose of this paper, equation 2 will be used as the update function of each particle.

3.1 Particle Increment and Global Increment

In [1] the authors established that the degree of collaboration and individual determination of each particle is an essential balance, which will dictate the effectiveness of the particles in converging on the best solution (or near best) solution in an efficient way.

The variables c_1 & c_2, are one of the cornerstones of the PSO thesis. c_1 symbolises the weight or significance placed on the cognitive aspect of each particle, which is represented in each particle as its all time personal best position (*pbest*). c_2 is a similar weighting mechanism which focuses on the global best of the entire swarm, which dictates the degree of collaboration the particle engages with.

[1] Presents interesting findings on the influence of c_1 and c_2 in its “Cornfield Vector” simulation, in which the authors shows that the weight given to each of the factors will have pivotal effect on the velocity of a particle and how long it takes the swarm to converge on a possible solution. From the point of view of a dynamic environment, where the location of ideal fitness is constantly changing, this may lead to a rethink the idea of the particle weightings; it may be the case that a slower convergence on a solution may help the swarm to adapt better to the environments dynamism.

3.2 Inertia Values

The idea conceptualized for the incorporation of the inertia value is outlined in [2]; it was brought into the velocity update function in order to help encourage the local searching capacity of a particle. [2] Discusses how the final solution of a swarm is deeply dependant on the initial location of particles in the search space. The author highlights how the constant velocity tends to polarize the particles in the search space and may at times, encourage particles to converge at too quick a rate and fall victim to a local maxima (expressed in the search space as *gbest*) of a particular particle, Hence the role of *w* is to lead to a less high speed velocity update and encourage a greater exploration of each particles locality.

The idea of local searching in an environment holds particular relevance in a dynamic environment, as it may often be the case that with limited number of particles, a swarm may find a greater degree of exploration more beneficial than simple exploitation of already known regions of high fitness.

4 A Dynamic Environment

A dynamic environment is one in which the peak of optimal fitness is constantly shifting its position in the search area, there are a multitude of influences which can be factored into in order to define the type of environment which is being understood.

The number of areas in an environment of high fitness is important when searching for peaks, Multimodal environments are ones in which there are more than one area of high fitness [8]. These could be conceptualised as a series of hills in a landscape. Obviously, this type of environment will have a far greater degree of complexity [8], when compared to an environment in which there is one main peak to be discovered.

The rate of change of any environment will be an important factor when assessing a swarm’s ability to track the peak of greatest fitness. Works such as [6] have concluded that the optimal peak should not move in the search space faster than the particles can fly, as slow particles may not be able to converge on the peak because is moving too fast.

4.1 Three-Dimensional Parabolic Function

For the purpose of this paper a 3-dimensional parabolic function will be used, this function has been widely implemented in experiments of many papers [6] [9]. The singular peak of optimal fitness moves around the environment based on an offset function, (equation 4).

The offset function uses the combination of the max velocity of the swarm multiplied by a value, set as *.1* for my experiments (as used by [6]), to define v_k, which is then multiplied by a Gaussian random number between 0 and 1(*rand(0,1)*). The offset is then completed by factoring in the offset of previous evolutions.

This offset is updated upon each evolution of the swarm and is used within the fitness function (equation 3) to help evaluate each particle within the swarm.

$$f(x) = \sum_{i=1}^{n} (x_i + s_i)^2$$

Equation 3 - The Dynamic Fitness Function

$$s_i(t+1) = s_i(t) + v_k * rand(0,1)$$

Equation 4 - The Dynamic Offset Generation Function

5 Experiments

The experiments conducted in this paper have a common implementation; they utilize a modified version of the JSwarm-PSO java package constructed by Pablo Cingolani (version 1.2). The package was suited to the required tasks of the experiments, as it is modular and the core classes such as `FitnessFunction.java` and the `Particle.java` can be extended to include the dynamic fitness function and 3 dimensional particles.

For the calculation of random numbers the experiments use the standard java `java.util.Random`, This is used for generating both normal and Gaussian random numbers (required by the dynamic environment offset).

For each experiments result, a swarm of a standard size (25 or 10 for certain experiments) is initiated, which is designed to work within a search space of a fixed size (1000 by 1000, *x* and *y* axis respectively). Each modified swarm completed a run of 300 evolutions. This process is repeated over a series of 15 runs of each varying set of parameters. As such all values presented in the following experiments are based on averages from these runs.

As outlined earlier the acceleration of the location of the dynamic value is important, and as such the speed has been restricted in this experiment to be no greater than the maximum speed of a particles.

In general, the amount of runs and evolutions used gave each swarm a sufficient amount of time to solve the required problem and that extending the swarms evolutions beyond 300 is not necessary.

5.1 Experiment 1 - Varying Inertia Values in Regular sized and Small Swarm

This experiment is made up of two sub sections, both of which involving the testing of various inertia values using different size swarms. All other variables and influences have been set to standard values as defined in section 5.

5.1.1 Regular Sized Swarm

The first set of tests conducted test the inertia values of *.9*, *.95*, *1*, *1.05*, and *1.1* on the standard swarm; other values such as *.85* were tested but yielded no information of significance. The most obvious feature can be seen in the results of the average fitness of all particles in the swarm (Chart 1).

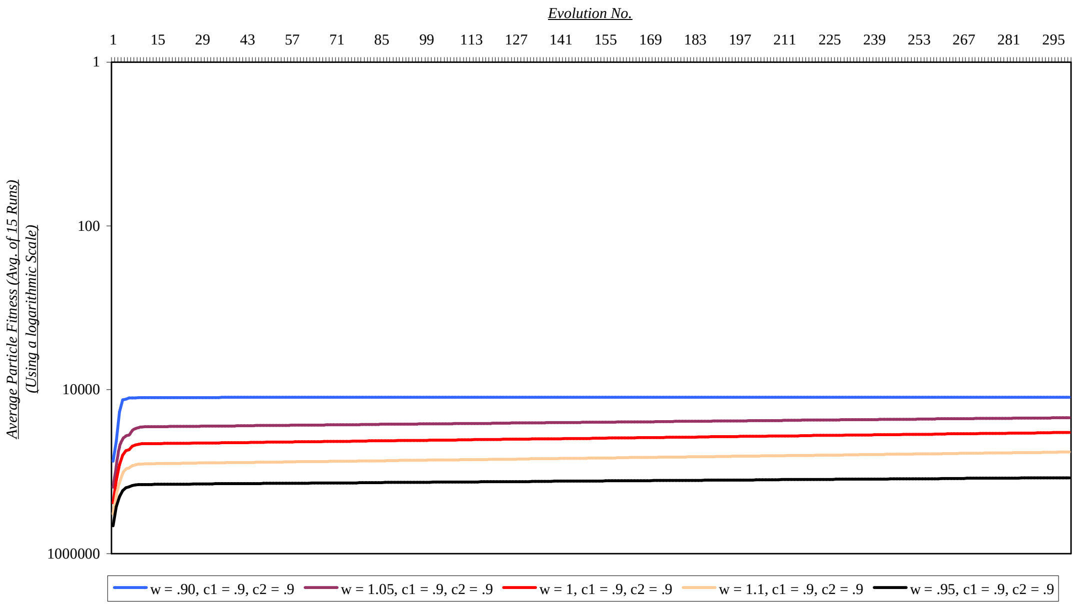


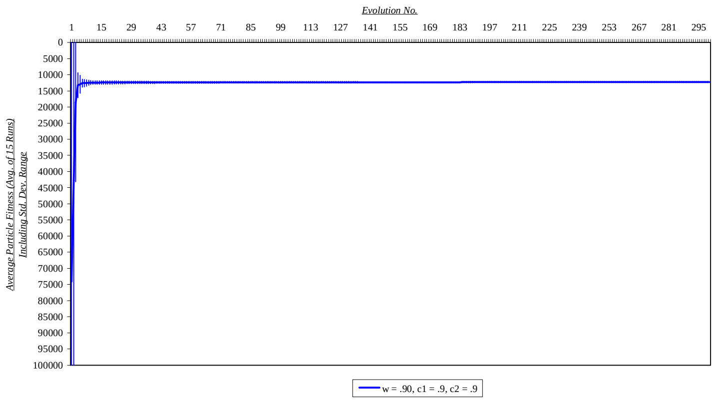
Chart 1 - Impact of Inertia on PSO fitness in a Dynamic Environment

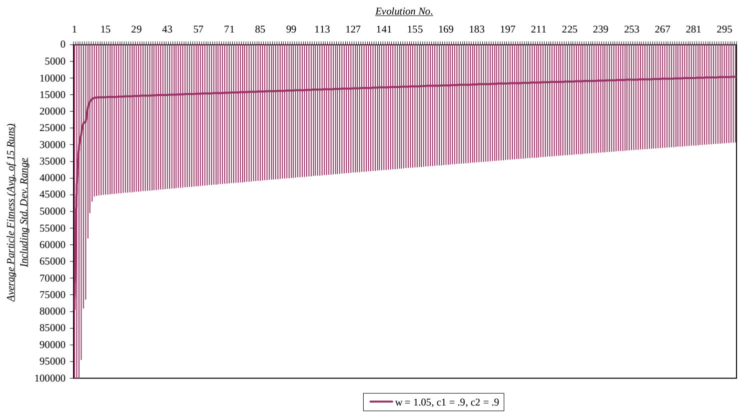
As seen in Chart 1, an inertia value of *.9* appeared to have the greatest benefit to the swarm and is markedly different from the next best value. Other inertia values are at times quite far off the best solution.

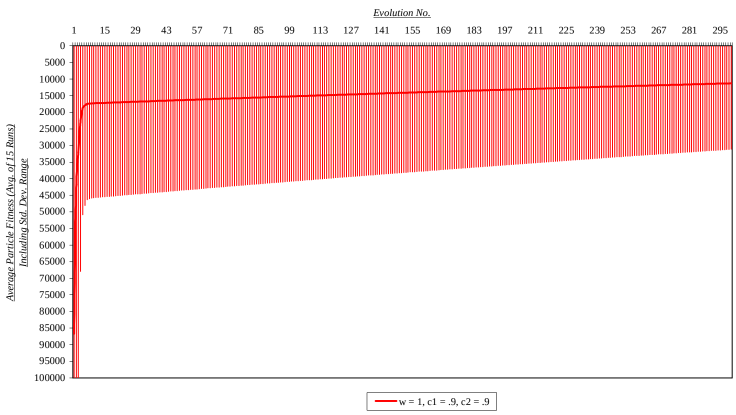
The inertia values played an important role in encouraging particles within the swarm to explore their locality and arrive at areas of better fitness. The graphs presented in Chart 2 expand on the data presented in Chart 1, by including standard deviation information. This information is interesting as it shows that in many instances there are solutions available that were of a better fitness, but due to the shift in inertia values there was not enough velocity present in each particle for them to cluster, or simply that the swarm is being pulled in a multitude of directions and was unable to form a consensus on what the best position available is.

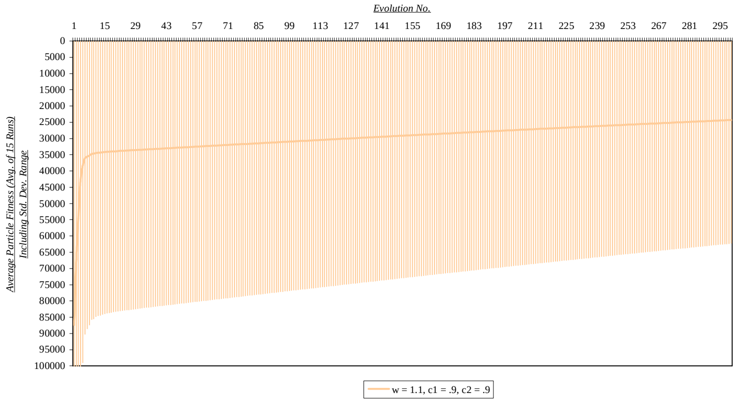
Interestingly enough the inertia value of the curve which showed the greatest fitness, showed little deviation from its best average position, this would give the impression of a swarm which has essentially stabilised in some way and is clustered at a certain point, which is, over the course of the runs, proved to be the highest fitness achievable.

It is important to maintain that although some solutions were better than others, they were all considerably inefficient at dealing with the problem proposed.


Evolution No.
1 15 29 43 57 71 85 99 113 127 141 155 169 183 197 211 225 239 253 267 281 295
0 5000 10000 15000 20000 25000 30000 35000 40000 45000 50000 55000 60000 65000 70000 75000 80000 85000 90000 95000 100000
Average Particle Fitness (Avg. of 15 Runs)
Including Std. Dev. Range
w = .90, c1 = .9, c2 = .9


Evolution No.
1 15 29 43 57 71 85 99 113 127 141 155 169 183 197 211 225 239 253 267 281 295
0 5000 10000 15000 20000 25000 30000 35000 40000 45000 50000 55000 60000 65000 70000 75000 80000 85000 90000 95000 100000
Average Particle Fitness (Avg. of 15 Runs)
Including Std. Dev. Range
w = 1.05, c1 = .9, c2 = .9


Evolution No.
1 15 29 43 57 71 85 99 113 127 141 155 169 183 197 211 225 239 253 267 281 295
0 5000 10000 15000 20000 25000 30000 35000 40000 45000 50000 55000 60000 65000 70000 75000 80000 85000 90000 95000 100000
Average Particle Fitness (Avg. of 15 Runs)
Including Std. Dev. Range
w = 1, c1 = .9, c2 = .9



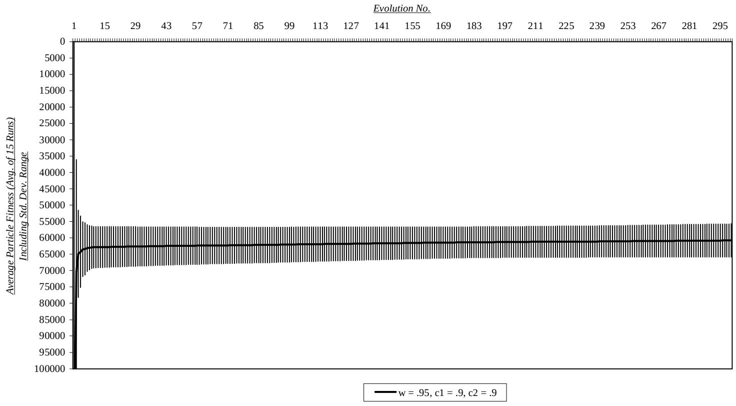


Chart 2 - Impact of Inertia on PSO fitness in a Dynamic Environment (With Standard Deviation)

5.1.2 Small Sized Swarm

In order to test the impact of inertia change on a smaller swarm, parameters were kept the same as in the previous test, however with a swarm of population size 10. As outlined in section 3.2, the idea of using a smaller swarm with various inertia values is to determine whether or not a small swarm will benefit from a greater degree of exploration or simply behave as per usual.

The average particle fitness for each run plotted for various inertia values shows quite interesting and different results when compared with the larger swarm presented in Chart 1. The most obvious feature presented by the chart, is that a smaller swarm seems to show a far greater degree of optimization ability when using a higher inertia value, from experiments it would appear that an inertia weighting of *1.05* as being the optimal level, with a considerable gain over alternative values.

When comparing the average fitness values generated from the use of a small swarm, it is no surprise that in the vast majority of cases, due to the lesser amount of particles in the swarm, it is unable to find as great a fitness value. However, from the results of the experiment it is clear that by altering the inertia value, a small swarm can offer equal or better solutions.

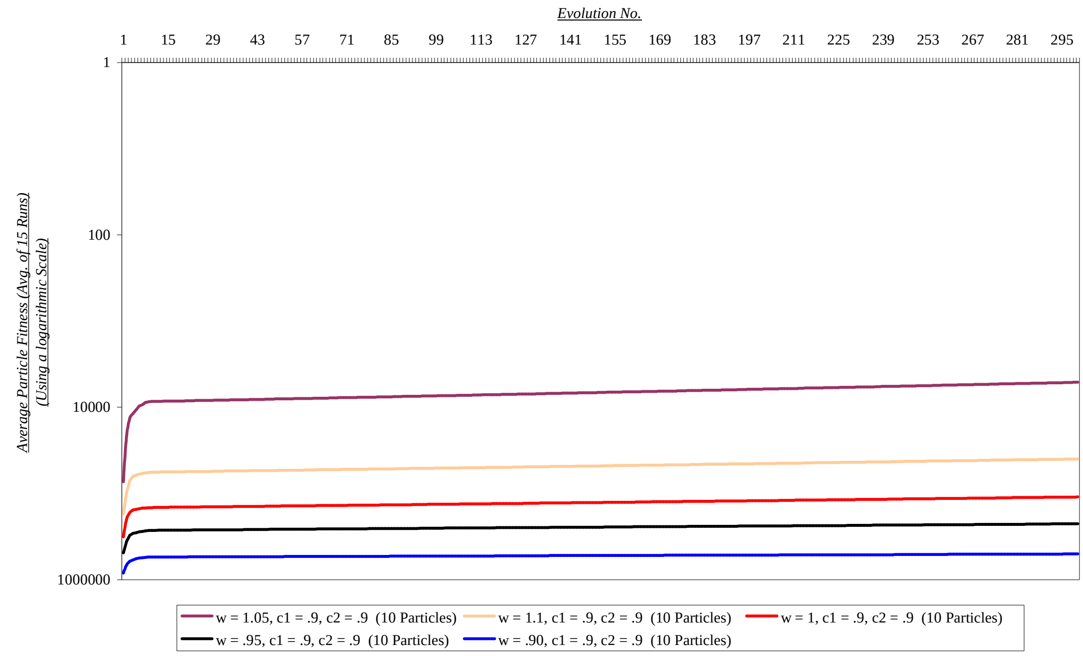
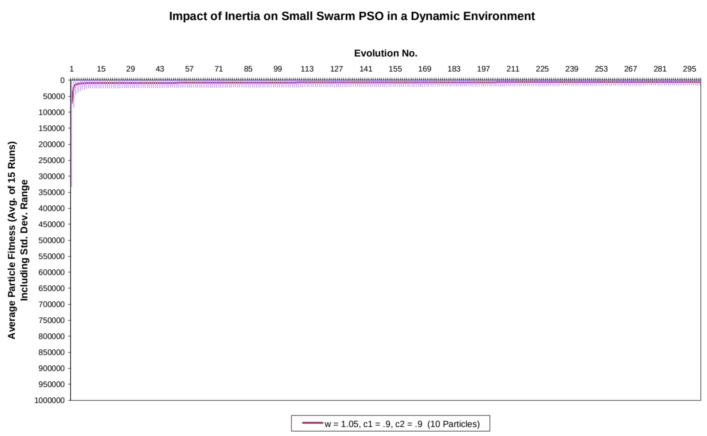
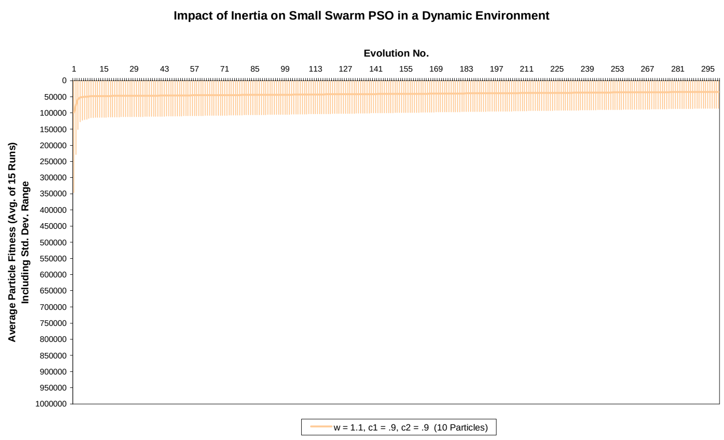
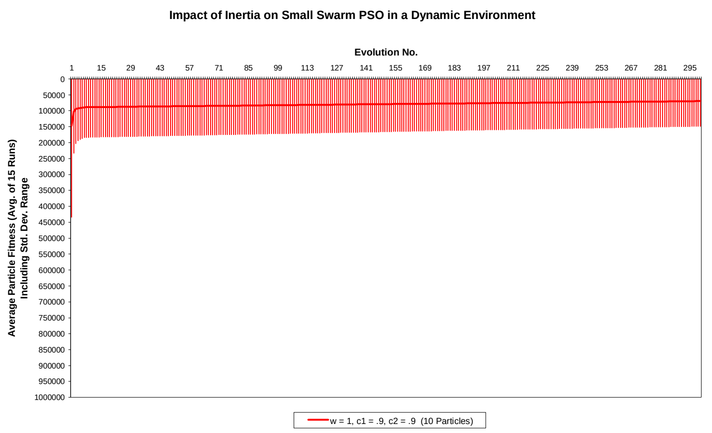


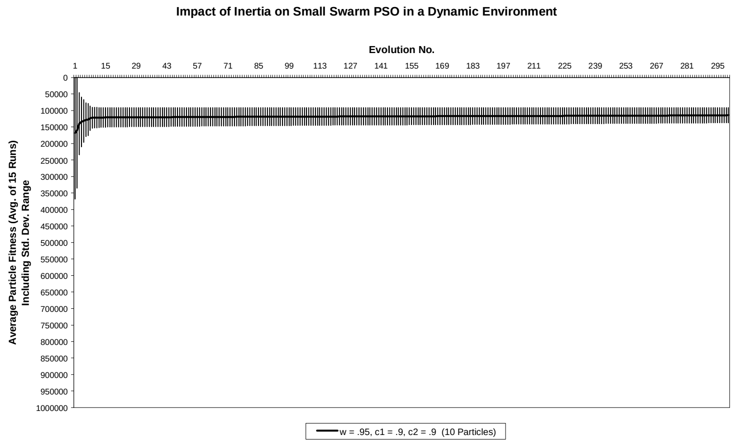
Chart 3 - Impact of Inertia on PSO fitness in a Dynamic Environment (Small Swarm - 10 Particles)

Once again the various inertia values used for the experiments present some quite interesting behavior when expressed with their standard deviation graphed (Chart 4). For instance the most effective swarm has quite low bounds for its deviation, suggesting that the particles in the swarm are not moving at too great a velocity and leading to the incorporation of irrelevant data. The deviation increases as the general fitness of the swarm is reducing.









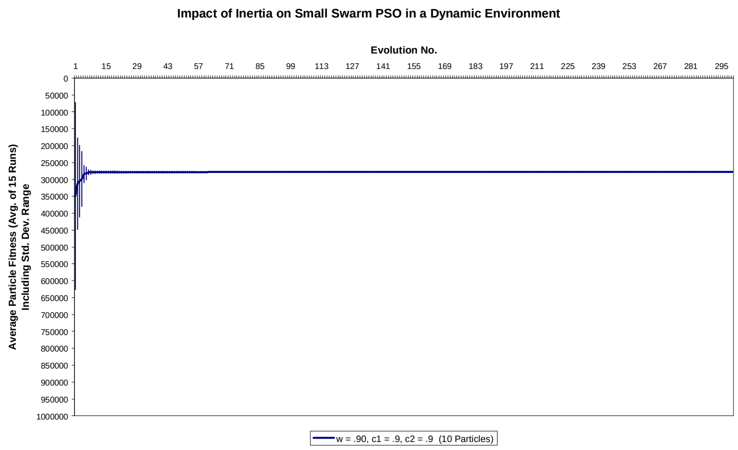


Chart 4 - Impact of Inertia on PSO fitness in a Dynamic Environment (With Standard Deviation - Small Swarm 10 Particles)

5.2 Experiment 2 – Modifying the Particle Increment and Global Increment

In this experiment I plan on testing various values of the particle and global increment in order to assess their impact upon the swarm’s ability to locate an area of high fitness within the dynamic environment.

5.2.1 – Particle Increment

Particle increment is a metric which dictates the weight of influence a particle attaches to its own *pbest* score when calculating the movement upon the next evolution. In order to test the possible benefit for the swarm, the swarm will be tested using four different values (*.85, .90, .95, and 1*).

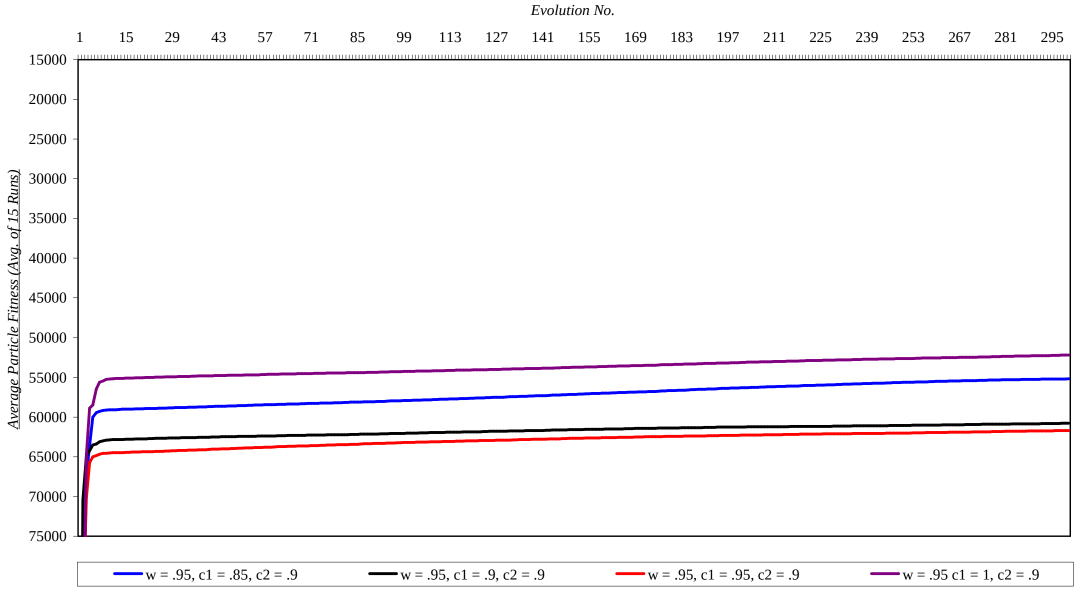


Chart 5 - Impact of Particle Increment on PSO in a Dynamic Environment

Chart 5 displays the findings of the test of the four different variants of c_1; the most obvious feature of these results is the general lack of fitness that has being gained from the variance in c_1. Of the four configurations implemented only two show any possible gain. These are the two extremes of the test values.

Without further analysis of the results, it is clear that particle increment on its own has only a small degree of leverage in the swarm's ability to locate and successfully track the global optimum in a dynamic environment.

5.2.2 – Global Increment

Global increment c_2 is responsible for the weighting the *gbest* variable in the update of a particles position. For this experiment three values (*.85*, *.9* and *.95*) are tested in order to understand role c_2 within the swarm, and the swarms' ability to navigate towards an optimum.

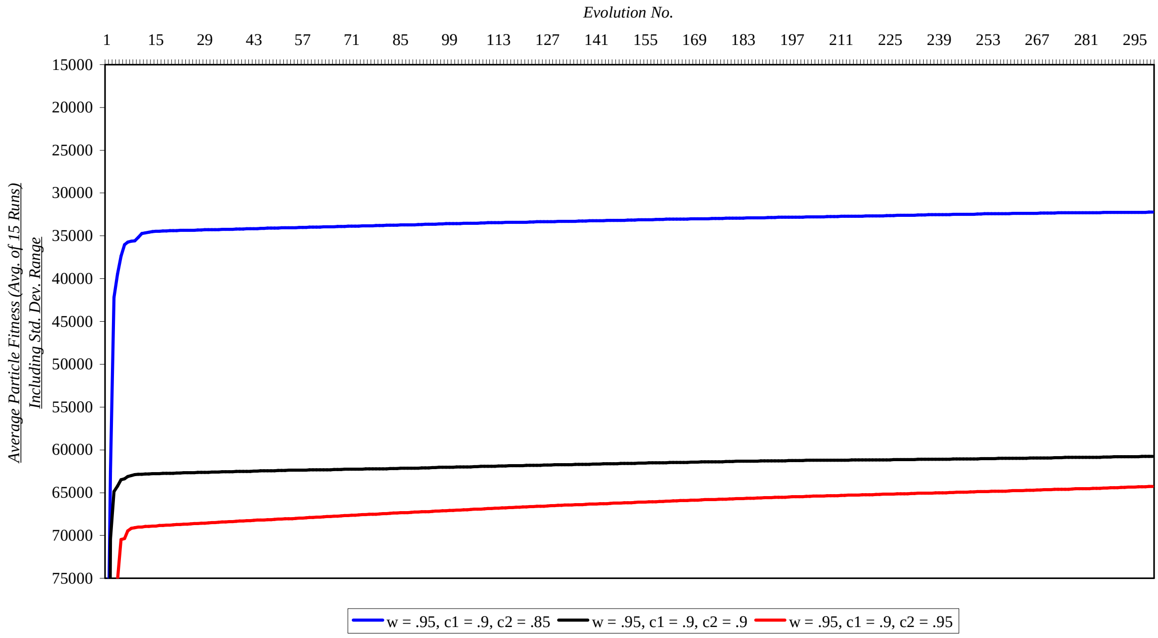


Chart 6 - Impact of Global Increment on PSO in a Dynamic Environment

Chart 6 shows the difference in average fitness upon each evolution, between the various increment values, there is evidence of a considerable gain attributed to usage of a lower $c_{2,}$ This is quite interesting as it displays how a swarm reacts differently based on the weight it gives to the global best solution.

The rise in fitness could be attributed to the fact that due to the dynamic nature of the environment; the swarm is not as attracted to the global best so as new global best is constantly changing particles are discovering better solutions in the locality of the global best.

From information provided by the chart, it is sufficient to say that in a dynamic environment a lower global increment value results in a greater fitness value for the swarm. However it is once again important to observe that the swarm with the best fitness value was far from optimum.

6 Conclusions

From the experiments undertaken in this paper it is obvious that a basic PSO algorithm is not extremely effective at locating optimum results within a dynamic environment using a parabolic function, However, the use of inertia (*w*), particle increment (c_1) and global increment (c_2) do play a key role in the swarms evolution and although when altered on their own they may not induce a significant gain. It is possible, that by the combination of each of the factors a swarm may become better at resolving problems.

To conclude, from research undertaken by various authors [4] [7] [8] [9] , it has being demonstrated that the alteration of the PSO algorithm as a whole will produce a greater degree of efficiency and allow the swarm to wok effectively in a greater range of problems [7], with a vastly more complex combination of dimensions and multi modal designs.

7 Future Works

Due to the constraints placed on this paper, it was not possible to investigate other areas of interest; these include the inter-relation of inertia, particle increment and global increment, and how a combination of modification to each of the parameters may cause the swarm to behave.

The dynamic environment was also an area which could have being expanded, the possibility of working with more than simply a three dimension, single optima curve would have being explore.

Overall it must be said that work achieved, and research undertook in this project both served as stimuli for interest in the field of PSO and consequently further study of the area.

References

[1] Kennedy, J. and Eberhart, R. 1995. *Particle Swarm Optimization*, IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, IV: 1942-1948.

[2] Shi, Y. and Eberhart, R. 1998. *A Modified Particle Swarm Optimizer*. Proceedings of the IEEE Congress on Evolutionary Computation (CEC 1998), Piscataway, NJ. pp. 69-73,

[3] Zhang, L., Yu, H., and Hu, S. 2003. *A New Approach to Improve Particle Swarm Optimization*. Genetic and Evolutionary Computation – GECCO 2003, Springer

[4] Carlisle, A. and Dozier, G. 2000. A*dapting Particle Swarm Optimization to Dynamic environments*. Proceedings of International Conference on Artificial Intelligence (ICAI 2000), LV, Nevada, USA. pp. 429-434, 2000

[5] Kennedy, J. 1997. *The Particle Swarm: Social Adaptation of Knowledge*, IEEE International Conference on Evolutionary Computation NJ, USA

[6] Cui, X., Hardin, C.T., Ragade R.K., T., Potok, E., and Elmaghraby A.S., 2005. *Tracking non-Stationary Optimal Solution by Particle Swarm Optimizer.* Sixth International Conference on Software Engineering,

[7] Blackwell, T., and Branke, J., 2006. *Multiswarms, Exclusion, and Anti-Convergence in Dynamic Environments*. IEEE Transactions on Evolutionary Computation, Vol. 10 No. 4

[8] Parrott, D. and Li, X., 2004. *A Particle Swarm Model for Tracking Multiple.* School of Computer Science and Information Technology, RMIT University, Australia.

[9] Angeline, P., 1997. *Tracking Extrema in Dynamic Environments*, Proceedings of the Sixth annual Conference on Evolutionary Programming VI, IN, USA.

[10] Surowiecki, J. 2006 *the Wisdom of Crowds, Why the Many Are Smarter Than the Few.* abacus, London