Particle Swarm Optimization Assignment 03

Sarah Childs
Brock University
Email: sc15fv@brocku.ca

***Abstract*—This report focuses on the use of the vanilla particle swarming optimization for the Rastrigin function, and the outcomes based on different control parameters.**

I. Introduction

Particle swarm optimization (PSO) is a population-based technique that has multiple different algorithmic implementations. For this assignment I used the vanilla PSO as required. This algorithm is very basic and focuses on the particles being attracted to the most fit values that they personally have discovered and their neighbourhood's global best.
This algorithm was used to attempt optimization of the Rastrigin function (Fig. 1) while gathering data on how different control parameters affected the outcome of these optimizations.

$$f(x) = 10n_x + \sum_{i=1}^{n_x}(x_i^2 - 10cos(2\pi x_i)) \tag{1}$$

There was also the need to implement a random search algorithm to aid the comparison of parameter configurations, and how they allow the PSO to perform better than a random search.

II. Parameters

As mentioned the assignment required multiple experiments to be run for each parameter configuration, and then for the random search algorithm. These control parameters can be seen in Fig. I

TABLE I
Control parameter configurations

Experiment	Inertia	Cognitive/Social
Random Search	-	-
PSO-1	0.729844	1.496180
PSO-2	0.4	1.2
PSO-3	1.0	2.0
PSO-4	-1.0	2.0

For these parameter configurations the cognitive and social acceleration coefficients are set to be the same. This means that the particle will consider it's personal best position to the same degree as it considered the neighbourhood's best position. These coefficients determine how much influence the particle will search in the local sense when comparing to it's personal best, and how much it will search in the global space when comparing to the neighbourhood's best.

The inertia of the particles determines how much influence the previous velocity will have on the updated velocity. This can help overcome local optima as it attempts to push the particle in the same direction it was moving in.

Alongside the designated control parameters there was also the size of the swarm and the iteration count to be considered. Through experimentation I found that around 550-600 the optimization stopped and the swarm's positions converged into a local optima. They continued to improve past this iteration count however those improvements were only in the decimal range. As such I have used 600 as the maximum number of iterations.

For the population size of the swarm I used 30 particles. I determined this number was the best suited as it provided results with a lower standard deviation compared to 20 or 40 particles in the swarm. A size of 40 did provide with the best solution for 1 seed of an experiment, however the others did not perform any better than a size of 30. As such I left the population at 30 so as to have less deviation amongst my results.

The sampling used for the random search was calculated using the iteration count multiplied by the swarm size. This provided the search with an equal amount of positions to be checked, much like each particle of the PSO will sample a single position for each iteration.

The position dimension and range of the search space were provided in the assignment outline. The dimensions of the position were to be 30, so each particle held a 30 element array which represented its position. As for the search space, that was to be in the range [-5.12, 5.12]. These configurations provided us with a baseline for the optimal value of the Rastrigin function; this value was 0 should all values of x be 0.

III. Results

For the experiments I ran each parameter configuration 7 times and compiled their average fitness, standard deviation and median values in Table II. These values give a brief idea of how the parameters control the algorithm's output.

Additionally, PSO has the potential to perform worse than a random search should the parameter configurations be bad. These configurations in Fig. I allowed me to see the difference between the best configuration and others.

TABLE II
SUMMARY OF FITNESS VALUES FOR 7 SIMULATIONS

Experiment	Average	Std. Dev.	Median
Random Search	344.73	10.88	344.59
PSO-1	64.67	17.21	66.662
PSO-2	135.57	25.75	124.08
PSO-3	440.06	26.88	430.15
PSO-4	440.06	26.88	430.15

As can be seen from Table II the first PSO experiment with inertia set to 0.729844 and the coefficients being 1.496180 is the experiment that performed the best. These configurations had been previously found to perform well as they satisfy the convergence criteria for PSO, meaning that the particles have entered an equilibrium state around an optima. For all 7 experiments they did not deviate much from the average while some performed better and others worse, their standard deviation was measured to be 17.21 coming in as the best for all PSO experiments. The deviation was higher for PSO than random search as PSO was reliant on the random function to provide good starting positions for the particles; whereas random search will continue to produce different positions that aren't reliant on the previous position and velocity updates.

From the second PSO configuration onwards there is a drastic jump in average fitness found and the standard deviation from that average. For PSO-2 the median value is below the average indicating that most of the fitness found are greater than 124.08; leaving the potential that 124 was found more so because of a good random seed than the parameter configurations themselves. The difference in performance from PSO-1 to PSO-2 is rather drastic with only a slight change to the parameter values. However, 0.4 is on the lower end of the suggested inertia weight range, compared to PSO-1 which has an inertia on the higher end. The recommended range for inertia is [0.4, 0.8]. This provides the velocity of a particle with a general direction to move in without simply speeding up the particle as an inertia of 1 would do. An inertia value of 0.4 seems to provide the velocity with a general direction but slows the particle down more than necessary. This configuration still led to better results than that of PSO-3 and PSO-4 which both were outperformed by random search; however, it doesn't optimize as well as PSO-1's configuration.

Random search performed badly in comparison to the first two parameter configurations for PSO, it did optimize values however the best position it could find for each experiment held very close to the same average. There was a standard deviation amongst experiments of only 10.88, this measure means that all fitness values produced were within 11 counts of the average fitness for all. This goes to show that random search is very inefficient for optimization as with 18,000 iterations it could not produce fitness values closer to zero like PSO-1 and PSO-2. It may have been able to optimize more with a larger number of iterations, however given the equal iteration count it was unable to succeed over these two PSO configurations because it does not take into account any details of the surrounding search space, it can only search at random.

PSO-3 and PSO-4 performed exactly the same, producing the same fitness values amongst all experiments, and the same global best position was found by each. They were identical, despite their parameters being different (Fig. I), PSO-3 had a 1.0 inertia value, whereas PSO-4 has a -1.0 inertia value. This suggests that a negative inertia value does not change the overall performance of a PSO, however this may change with the type of PSO algorithm implemented, additionally it could change should different coefficient values be used. The convergence curve graph for equal configurations (Cognitive coefficient = Social coefficient) in the COSC 3P71 slides suggests that if the sum of the two terms was less than 3 then their outputs may have varied despite having the same inertia value. However, this graph suggests that with the configurations used their convergence was much faster and stuck to a local optima that both could achieve. Additionally, both inertia values are outside of the recommended [0.4, 0.8] range which indicates that their outputs would most likely not be lead to an optimal solution.

When considering the coefficient values it seems that the closer to a sum of 3 is best for equal configurations. Above that threshold provides more influence on the particle when searching locally and globally and could lead them to premature convergence as their influence takes over. This influence in addition to the strong inertia for PSO-3/4 which left the previous velocities in tact I believe was what made those configurations perform so poorly. When looking at PSO-3/4 in comparison to PSO-1 the difference in performance is staggering when only minor adjustments were made to the parameter values. The standard deviation for PSO-3 and PSO-4 was also the greatest; although PSO-2 also had a similarly large standard deviation amongst it's produced fitnesses it still performed better than PSO-3/4.

IV. CONCLUSION

When looking at the comparison of each PSO experiment, and the random search, it's impressive to see that such a minor difference in parameter configurations such as that between PSO-1 and PSO-3/4 could produce widely contrasting results. This brings to attention how important these control parameters are for the PSO to perform optimally, with slight adjustments the results could change dramatically. It's noticeable the importance of having strong inertia while still not allowing it to overpower the velocity equation, as seen with PSO-1. It provides a stronger inertia and balances that with the coefficient values to provide the most optimal

outcomes for the parameter configurations.

The coefficient values also need to be balanced so that they do not overwhelmingly influence the particle for ever best solution found, the stronger these coefficients are the more exploration and exploitation of the search space is occurring. If they are too strong then each new best solution found will influence the particle to an entirely different area, not allowing those particles to gradually converge to an optima, but rather forcing them to converge quickly. This suggests that a faster convergence amongst the particles leads to worse overall performance.