Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a population-based, stochastic optimization technique. This optimization technique is based off of the movement of a swarm of particles and is heavily influenced by the behaviour of biological beings. The swarm of particles mimics the movement of a school of fish or a flock of birds, with each member of the swarm communicating information amongst each other. Each individual within said swarm mainly function to move towards its own personal best position and move towards the best position within a neighbourhood. In this implementation of PSO, the solution is formed from the position and the fitness is calculated using the Rastrigin Function. The Rastrigin Function appears as follows:

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

Ultimately, the goal is to minimize the value produced by the Rastrigin function and the optimal value is 0.

For this experiment, the PSO was exclusively initialized to have a swarm of 30 particles and a dimension of 30. The parameters of inertia weight (ω), cognitive acceleration coefficient (c_1), and social acceleration coefficient (c_2) were tested at four different combinations and their results were recorded after 1000 iterations of the PSO. A global-best neighbourhood was assumed and individual particles were permitted to leave the feasible region, however, updating of the personal best only took place when individuals were within the aforementioned region. Additionally, the PSO undergoes a synchronous iteration strategy, meaning the neighbourhood best position is updated after all particles have updated their position.

After running the PSO through 1000 iterations, it was clear that the PSO performed optimally under the configuration PSO-1 found in Table 1. The final data also displayed optimization was possible through the PSO-2 configuration, even though it was not nearly as efficient as PSO-1. In contrast, the configurations of PSO-3 and PSO-4 resulted in no improvement of fitness from their initial states. These results suggested that optimization within the PSO is only possible with specific parameter configurations. The difference in having an oppositely-signed inertia weight resulted in no change in the outcome between PSO-3 and PSO-4 in that they both could not optimize their solutions. Looking at the line trends of Figure 1 – 6 it was interesting to see PSO-1's configuration lead to the most up-down fluctuation of best-fit values when optimizing its solution, though the other tests resulted in lines that would only improve gradually over time or remain unchanged.

When contrasting results from performing PSO and Random Optimization, it was clear PSO was superior in its optimization when given specific parameters. Intriguingly, the Random Search 2 from Table 1 produced results that nearly rivaled the results obtained in a PSO-2. The main difference between the two Random Search configurations being in the upper and lower bounds used when determining a particle's velocity. For this experiment, Random Search 1 used a lower bound of -1 and an upper bound of 1. Whereas, Random Search 2 used a lower range of -0.1 and an upper range of 0.1 and was able to obtain a better optimization.

In conclusion, the results obtained from this experiment suggest the parameter settings are crucial to performance when implementing and contrasting a PSO and a RO. The data demonstrates that while under certain circumstances, the Random Optimization was able to do an adequate job optimizing its solution, Particle Swarm Optimization proved to be superior at finding an optimal solution.

Table 1
Results: Best Fitness

Configuration	Mean	Std. Dev.	Median
Random Search 1	290.7032888	12.54967362	287.7190564
Random Search 2	178.70969	5.766397307	179.1209352
PSO-1	89.52797301	23.70002922	89.54606959
PSO-2	130.3111555	17.11263599	122.3796951
PSO-3	429.170371	24.05520792	437.6598996
PSO-4	455.4120178	11.59089533	450.2375783

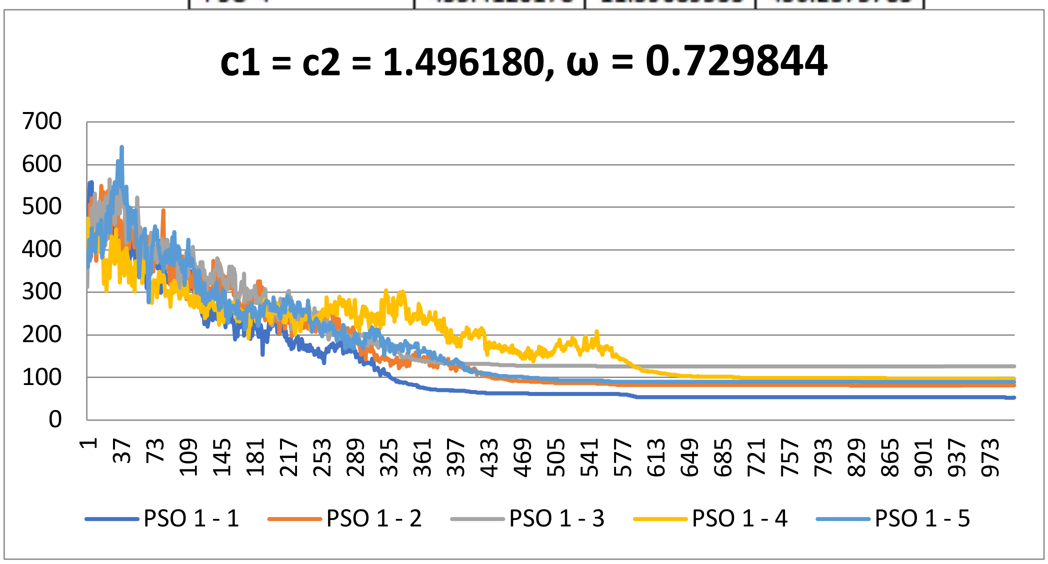

Figure 1. Applying PSO using **ω = 0.729844 and $c_1 = c_2$ = 1.496180**

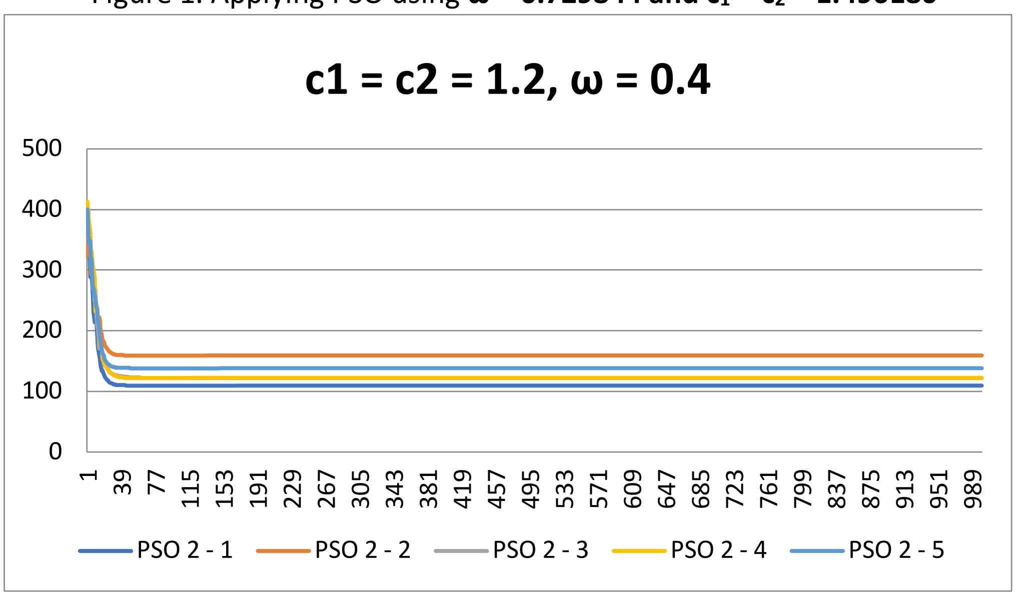

Figure 2. Applying PSO using **ω = 0.4 and $c_1 = c_2$ = 1.2**

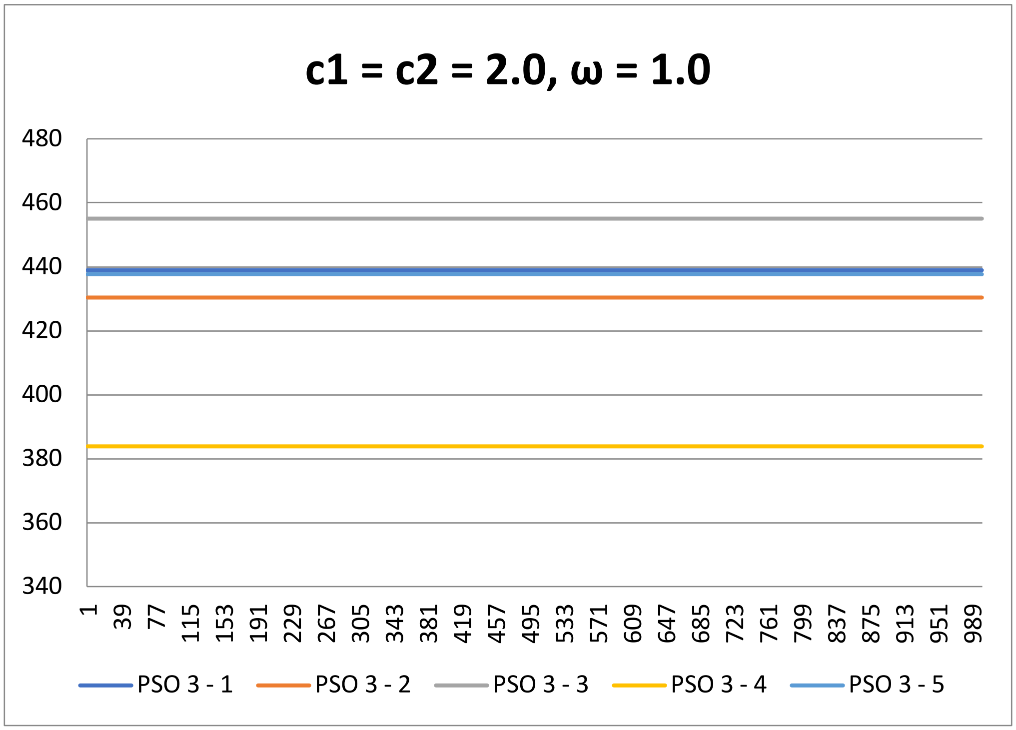

Figure 3. Applying PSO using **ω = 1.0 and $c_1 = c_2 = 2.0$**

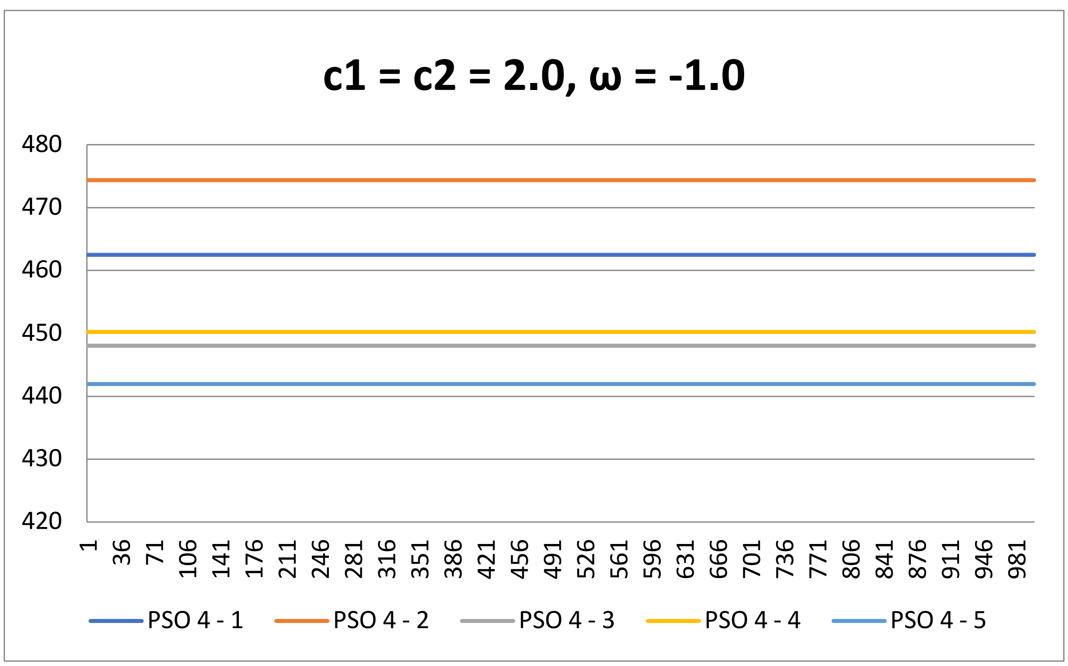

Figure 4. Applying PSO using **ω = -1.0 and $c_1 = c_2 = 2.0$**

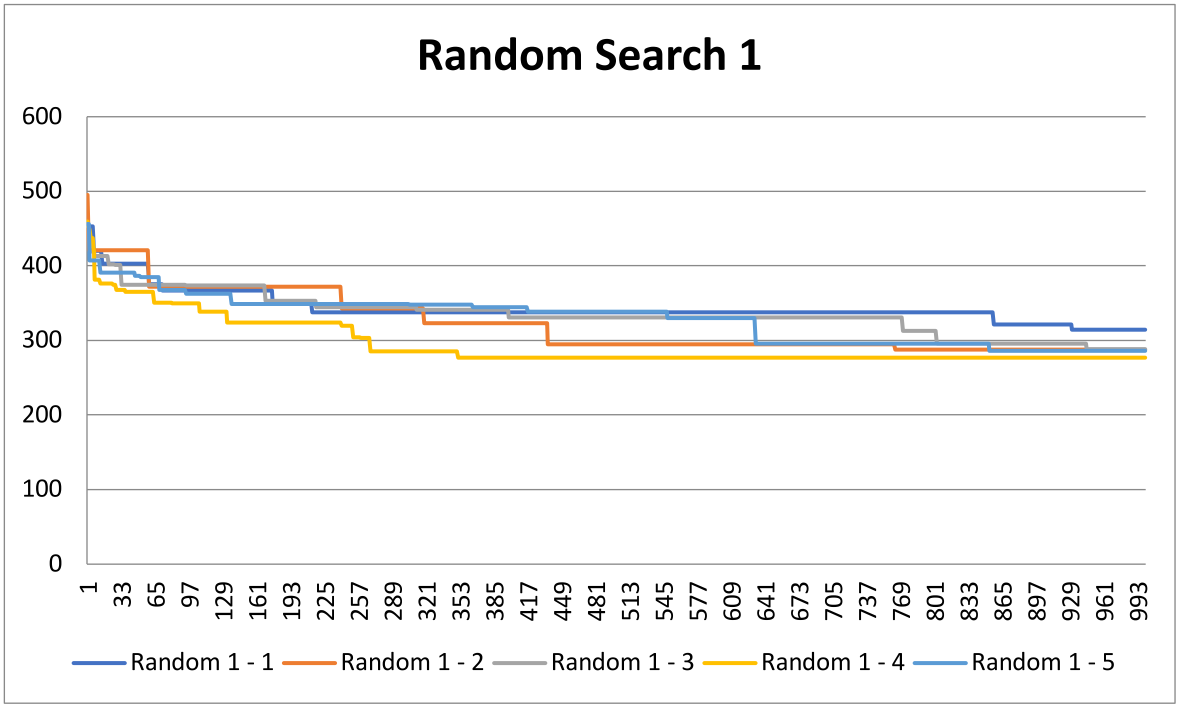

Figure 5. Applying RO using **velocity lower bound = -1 and upper bound = 1**

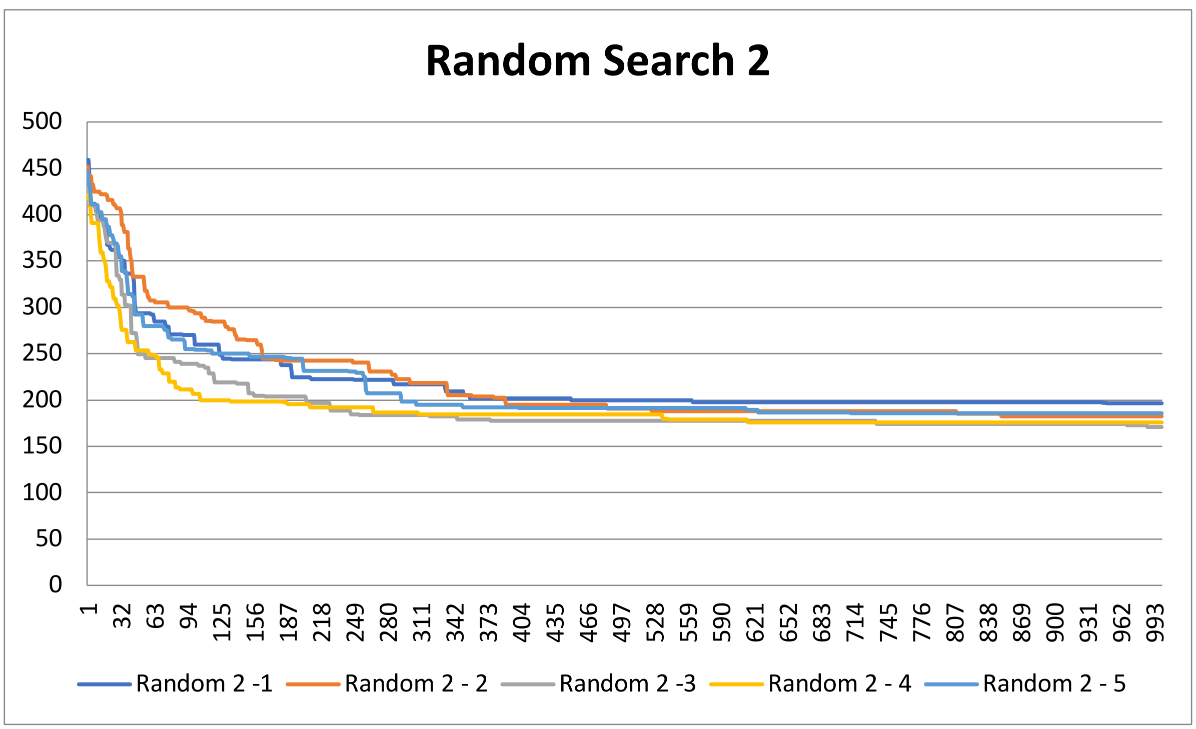

Figure 6. Applying RO using **velocity lower bound = -0.1 and upper bound = 0.1**