Q4. Global Optimisation

a. Particle Swarm Methods

a..1 Summary

The original Particle Swarm Optimisation (PSO) was originally introduced by Dr. Kennedy and Dr. Eberhart [1] in 1995 as a method of finding global optima, at the time for use in neural network optimisation. The algorithm was built on the basis of swarm intelligence, as is found in bird flocks or similar. The main attributes of a swarm that were also found to be the main attributes of PSO algorithms are:

1. The swarm must have the ability to carry out simple space and time calculations. In the case of PSO this means that the particles must be able to change direction and velocity in response to stimuli.
2. The swarm must be able to respond to its surrounding environmental quality. A particle in PSO must be able to assess its objective function value.
3. The swarm has some diversity in its response to stimuli and so there is some variation in movement.
4. The swarm must have some stability and therefore not respond to all stimuli equally/ maintain some sort of its own motion.
5. The opposite to 4, is that the swarm must also adapt its behaviour when the gains made by this change are considered worthwhile.

These concepts are all expressed in the general PSO algorithm.

a..2 Method

The general idea of the algorithm is that a population of particles begins at random positions with random velocities in the space. At each iteration the position of each particle is updated by calculating the displacement of the particle due to a calculated velocity. The velocity of the particle is a function of time and is influenced by three parameters: ω the inertia weight constant, C_1 the cognitive coefficient, and C_1 the social coefficient. The tuning of these parameters impacts the speed and accuracy of the PSO algorithm [2].

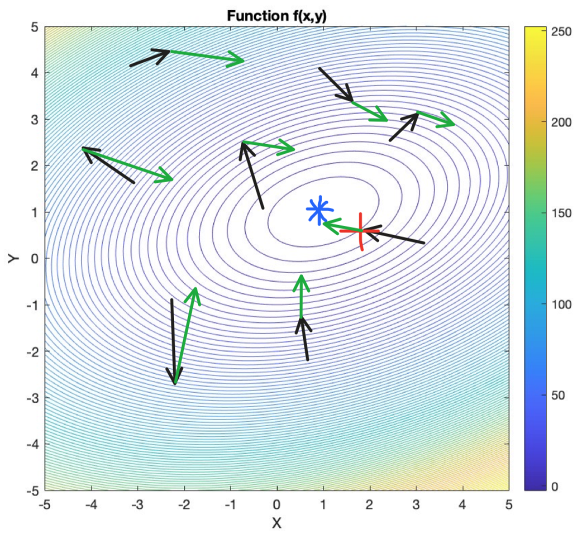

Figure 1: A rough demonstration of how PSO would act on a simple optimisation problem.

At each iteration, each particle updates its best value of the objective function pbest, and the global best gbest is also updated from all the particles. These variables give the swarm a 'memory' and allow locations of the optimal values (so far) to be stored. This allows the swarm intelligence to slowly discover new local optima while centering around a global optima. As shown in Figure 1, the black arrows indicate the start of the algorithm and the green show the actions of the particles on the second iteration. The red cross is the first iteration global best and the blue star is the true optimum. The particles velocities are impacted by their previous velocity (ω inertia) and both the location of their own optimal and the swarm optimal from the previous iteration. This relationship is described in the following:

$$X^i(t+1) = X^i(t) + V^i(t+1) \tag{1}$$

$$V^i(t+1) = \omega V^i(t) + c_1 r_1(pbest^i - X^i(t)) + c_2 r_2(gbest - X^i(t)) \tag{2}$$

These two equations govern the actions of all the particles in the algorithm [3]. Equation 1 simply determines new positions at each iteration in the same way as physics particles. Equation 2 determines the direction and magnitude of the velocity through comparison of the current location to the local best and the global best.

r_1, r_2 are random numbers given from a distribution giving the algorithm a stochastic quality. The influence of the local versus the global best is determined by the C_1 the cognitive coefficient and C_2 the social coefficient as mentioned above. A greater C_1 increases the exploration of new space by the particles whereas a greater C_2 increases exploitation, i.e. how much each particle is influenced by the discoveries of the rest of the swarm. ω is used to vary the inertia of a particle.

A greater inertia causes PSO to take more iterations to find a global optimum (if found) but also increases the likelihood it is not found. Whereas a smaller ω means that the PSO algorithm finds a global optimum quicker (if found) but also increases the likelihood that it is not found.

$$W_i = 1.1 - \left(\frac{gbest_i}{pbest_i}\right) \tag{3}$$

The solution to this is varying the inertia with time [2]. It is found that varying it non-linearly by method of Global Local best Inertia Weight (GLbestIW), in equation 3 has the best results. Generally this means the inertia increases the closer each particle is to the global optimum meaning that they are kept in that area.

a..3 Pros

1. The PSO method,first of all, is a simple algorithm with very basic calculation at each iteration. This means that it is computationally inexpensive and is therefore generally insensitive to the scaling of variables [4]. Similarly this means that the memory and processor requirements for implementing the method are minimal. In terms of computational efficiency the method can also be parallelized meaning each particle can be calculated simultaneously meaning that the algorithms are generally quick to run.
2. Another highly important feature is that the PSO method is derivative free and can therefore be applied to real world non-smooth objective functions [4].
3. By using the PSO method, the algorithm benefits from few parameters which can be tuned for optimal performance [5], as described in section a..2.

a..4 Cons

1. One of the main drawbacks of the PSO method is that it has a tendency to result in early convergence on sub optimal points. In other words, the method will likely converge in proximity to the true global optimum quickly [4]. For some applications this makes this optimisation method not suitable.
2. The convergence of this method is also not guaranteed for some functions [4].
3. For use in refined search areas, the PSO method is often slower when finding a true optimum in a mid optimal area.

Many of these drawbacks can be limited and controlled through the use of the method parameters, allowing PSO to be used in many different applications successfully.

References

[1] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948 vol.4, 1995.

[2] Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, pages 69–73, 1998.

[3] M Senthil Arumugam and M V C Rao. On the performance of the particle swarm optimization algorithm with various inertia weight variants for computing optimal control of a class of hybrid systems. *Discrete Dynamics in Nature and Society*, 2006:79295, 2006.

[4] Mohd Nadhir Ab Wahab, Samia Nefti-Meziani, and Adham Atyabi. A comprehensive review of swarm optimization algorithms. *PloS one*, 10(5):e0122827, 2015.

[5] Ahmad Rezaee Jordehi and Jasronita Jasni. Particle swarm optimisation for discrete optimisation problems: a review. *Artificial Intelligence Review*, 43(2):243–258, 2015.