Natural Computing Tutorial 2: Particle Swarm Optimisation (week 4)

Exercises

1. Check your understanding of the PSO algorithm:
   a. Why do we call the terms with α_1, α_2 "forces" and ω itself "inertia"?
   b. Would the algorithm work with negative values for α_1 and/or α_2?
   c. How well would the algorithm work for $\alpha_1 >> \alpha_2 > 0$ or for $0 < \alpha_1 << \alpha_2$?
   d. What is the benefit from using ω close to 1? What is the downside of this?
   e. Would the algorithm necessarily diverge if $\omega \geq 1$?
   f. Would it work with negative values for ω?
   g. Discuss how diversity can be maintained in a particle swarm.

2. Consider a particle "swarm" consisting of a single particle (i.e. we just have a personal best).
   a. How a deterministic PSO particle move in a one-dimensional search space? Assume the random factors are constant and equal to 1, and that the personal best never changes. Try to solve the problem analytically.
      Hint: Consider a matrix equation for the 2D vector $(v,x)^T$
   b. What would happen in this case in higher dimensions?
   c. Without aiming for the maths, discuss the effect or the noise in the original algorithm.
   d. If you like you can again discuss the PSO search biases here.

3. [**Numerical exercise**] Compare your findings from the previous questions with a simulation of a PSO algorithm. Try also to solve an actual optimisation problem such as the minimization of $f(x)=x^2$ or of a more complex function. The algorithm will be available before the tutorial, but it should not be to difficult to try writing (or finding) one yourself.

4. We have mentioned adding a repulsion term to the velocity rule of PSO.
   a. What happens when the particle is repelled from the globally best particle by which it is also attracted?
   b. What other terms could you add in order to adapt the PSO algorithm better to a particular problem?
   c. Each of these terms comes with one or more parameters. How can you use a genetic algorithm to choose for you the parameters for the new PSO algorithm with inertia, attractive forces, repulsion, alignment of velocities, …, …?

5. The particles in PSO interact only via the global best which is determined over all particles, compare this with the interaction of individuals in biological swarms, and discuss resulting options for the design of metaheuristic optimisation algorithms.

6. How would you adapt the standard particle swarm optimization algorithm to the travelling salesperson problem?

7. Differential evolution (DE) is a metaheuristic optimisation algorithm similar to PSO. However, for each member of the population in DE three other vectors are randomly chosen from this population in each step. The difference between two of them is multiplied by a parameter F and then added to the third vector as a kind of mutation. The new individual is obtained by a combination of some randomly chosen components from the mixture of three parent and the remaining components are kept from the original vector. The result is accepted or not, in a similar way as in Simulated Annealing. Compare the function of the algorithms for particle swarm optimisation and differential evolution.

This page by C. Reynolds www.red3d.com/cwr/ibm.html is of historic interest (not many links left)