Particle Swarm Optimization

PSO in one page

PSO is an evolutionary computation for optimization, particularly useful when the objective function $f : \Re^N \to \Re$ is a black box, in other words, we dont know function f explicitly, but we know output y of function f, given an input vector $\mathbf{x} \in \Re^N$. Typical examples include multilayer neural network, and objective function provided by a third party static or shared C++ library. The objective function f can be non differentiable, there may be constraints. PSO is an iterative algorithm that maintains a population of M particles (likes the genes in genetic algorithm), the m^{th} particle contains its current location $\mathbf{x}_m(t)$, current velocity $\mathbf{v}_m(t)$, its historical best location $\mathbf{x}_{m,best}(t)$ together with its historical best fitness $y_{m,best}(t)$. Besides, PSO should record global best location $\mathbf{g}(t)$ and global best fitness h(t), please be recalled that bolded letter means vector. Here are the PSO structures.

```
struct particle
{
    parameter_space_value_type        current_location;
    parameter_space_difference_type   current_velocity;
    parameter_space_value_type        my_best_location;
    double                            my_best_fitness;
};

struct swarm // i.e. global information
{
    parameter_space_value_type        best_location;
    double                            best_fitness;
};
```

Here is the algorithm : for each iteration and for each particle, calculate fitness at current location, update individual best and global best, finally, update velocity and find next location, repeat until stopping conditions are satisfied, such as predefined number of iterations, predefined minimum global best fitness, convergence of global best fitness etc. The updating formulae for velocity and location are :

$$\begin{aligned}
\vec{v}_m(t+1) &= \underbrace{w_0\vec{v}_m(t)}_{term_1} + \underbrace{w_1r_1[\vec{x}_{m,best}(t)-\vec{x}_m(t)]}_{term_2} + \underbrace{w_2r_2[\vec{g}(t)-\vec{x}_m(t)]}_{term_3} \\
\vec{x}_m(t+1) &= \vec{x}_m(t)+\vec{v}_m(t+1)
\end{aligned}$$

where $0.8 \le w_0 \le 1.2$ and $w_1, w_2 \sim 2$ are user defined constants

while $0 \le r_1, r_2 \le 1$ are values randomised in each iteration (evolutionary computation)

The velocity formula is the key of PSO, in which term 1, 2 and 3 play different roles, they are called inertia component, cognitive component and social component respectively. Inertial coefficient w_0 is smaller than one for deceleration or damping, which allows a faster convergence, it is greater than one for acceleration, which tends to explore a wider search space. Cognitive component acts as a particle's learning experience, while social component acts like a swarm behaviour. Finally, we can also add some constrains on velocity, so as to avoiding the particles from going too fast.

Reference

Partical Swarm Optimization : a Tutorial, by James Blondin, in 2009.