# 1 Particle Swarm Optimization

PSO can be done for constrained, non-constrained, multi-objective, etc. Essentially the idea is to get a group (swarm) of semi-independent searchers (particles) looking for the solution.

## 1.1 Particles

Each particle is an agent that is cooperatively searching the solution space. The particles, numbered $i \in \{1 : n\}$ at step $k$ store:

1. Current fitness, $f_{i,k} = f(x_i(k))$, and location, $x_i(k)$
2. Individual best fitness, $F_{i,k} = \max_{j\in\{1:k\}} f_{i,j}$, and the associated location, $X_{i,k} = x_i(j)$ such that $j = \arg\max_{j\in\{1:k\}} f_{i,j}$, encountered so far
3. Global best fitness, $F_{g,k} = \max_{j\in\{1:k\},i\in\{1:n\}} f_{i,j}$, and location, $X_{g,k} = \arg\max_{j\in\{1:k\},i\in\{1:n\}} f_{i,j}$
4. Current velocity, $v_{i,k}$

evaluation of the max and arg max is trivial in this case since it is done iteratively and we can just compare to the current fitness and position.

## 1.2 Constraints

Constraints are handled a variety of ways:

**Feasibility preserving** make a function that transform feasible points into new feasible points - typically assume linear/convex and a feasible start. These essentially replace the velocity and position generation. Searching the boundary is difficult with inequality and/or non-linear constraints is very difficult. Additionally the cost to force feasibility should be much higher than taking a feasibility step, ala superiorization.

**Penalty functions** incorporate the constraints into the fitness by adding a penalty function to heavily penalize not meeting the constraints.

**Distinguish feasible and infeasible** treat feasible and infeasible points differently. Some turn off feasibility constraints till a certain number of particles work and seek to increase the feasibility constraints that are on and work for the point. Note on (evaluated) and off (not evaluated) are different than active (on boundary) and inactive (in interior).

**hybrid methods** combine some other method with swarm. Often these are numerical methods, of mixed integer optimization.

### 1.3 Parameters

There are three basic parameters used to control the behavior of the algorithm and essentially control the weighting of the velocity weighting, which is how it gets a new guess. The parameters are $\omega$ (weight to give current velocity), $\phi_p$ (the weight to give the particles best), and $\phi_g$ (the weight to give the global best). These parameters can be thought of as learning parameters.

### 1.4 Algorithm

Each member of the swarm does the following at each iteration:

1. Calculate the fitness of its current guess/location in the solution space. The fitness function is the cost function we are trying to minimize. *This is expensive for many constraint methods.*

2. If a particle's current location has the best fitness it has personally encountered then it updates its personal best field with the location and fitness. *Trivial binary comparison.*

3. If a particle's current location has the best fitness any particle has encountered then it broadcasts the location and fitness and fitness, and updates its global best field. emph Asynchronous network broadcast.

4. Calculate the next guess by:

   (a) Calculate new velocity, by $v_{i,k+1} = \omega \cdot v_{i,k} + \phi_p \cdot rand() \cdot (X_{i,k} - x_i(k)) + \phi_g \cdot rand() \cdot (X_{g,k} - x_i(k))$

   (b) Calculate new position, by $x_i(k+1) = x_i(k) + v_i(k+1)$

   *Feasibility preserving modifies this part for meeting the constraints.*

## 2 Particle Swarm Superiorization

Each member of the swarm does the following at each iteration:

1. Calculate the fitness of its current guess/location in the solution space according to the unconstrained optimization problem.

2. If a particle's current location has the best fitness it has personally encountered then it updates its personal best field with the location and fitness.

3. If a particle's current location has the best fitness any particle has encountered then it broadcasts the location and fitness and fitness, and updates its global best field.

4. Calculate the next guess by:

(a) Calculate new velocity, by $v_{i,k+1} = \omega \cdot v_{i,k} + \phi_p \cdot rand() \cdot (X_{i,k} - x_i(k)) + \phi_g \cdot rand() \cdot (X_{g,k} - x_i(k)) + \psi \cdot d_{feas}$

(b) Calculate new position, by $x_i(k+1) = x_i(k) + v_i(k+1)$

Where $d_{feas}$ is the feasibility step from superiorization, and $\psi$ is its corresponding weight.

This method would be classified as hybrid, as intermediates are not guaranteed to be feasible, there is no penalty, and no distinction is made. The parameter $\psi$ can incorporate relative weighting of the unconstrained optimization to the feasibility update ($\phi$ terms).