Particle Swarm Optimization

Cosc 3P71

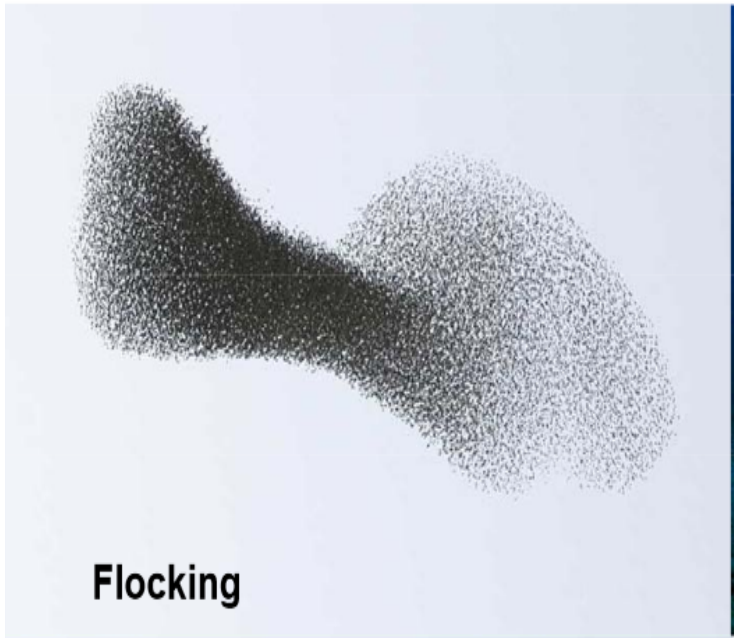
Fall 2015

Contents

Swarm Intelligence

- Collective intelligence of groups of simple individuals
- Interact to accomplish a common goal
- Often nature inspired
  - Bird flocks
  - Ant colonies
  - Fish schooling

Swarm Intelligence



Main Principles

- 1) The swarm is composed of many individuals, some of which may make mistakes
- 2) The swarm can solve complex problems that a single individual could not
- 3) Individuals in a swarm rely on their personal experience and the globally best individual(s).

Swarm Intelligence: Application Areas

- Biological and social modelling
- Movie effects
- Swarm robotics
- Dynamic optimization
  - Routing optimization
  - Structure optimization
  - Data clustering
  - Data mining

Particle Swarm Optimization

- Particle swarm optimization [1] (PSO) is an optimization algorithm
- Modelled after the real-world flocking behaviour observed in bird species.
- Designed to tackle problems with one objective (although multi-objective variants exist)

Particle Swarm Optimization

- Similarities to Genetic Algorithms:
  - Both are population-based algorithms designed to tackle optimization problems
  - Both are metaheuristic methods adept at overcoming local minima
  - Over time, individuals become similar to the "elite" members of the population

Particle Swarm Optimization

- Differences from Genetic Algorithms:
  - Inherently designed to tackle continuous domains
  - Steady-state population of individuals which move position rather than recombine
  - The most elite individual(“global best”) always participates in leading the entire population

Particle Swarm Optimization

- **Metaphor:** A bird flock is searching for an area with the highest concentration of food

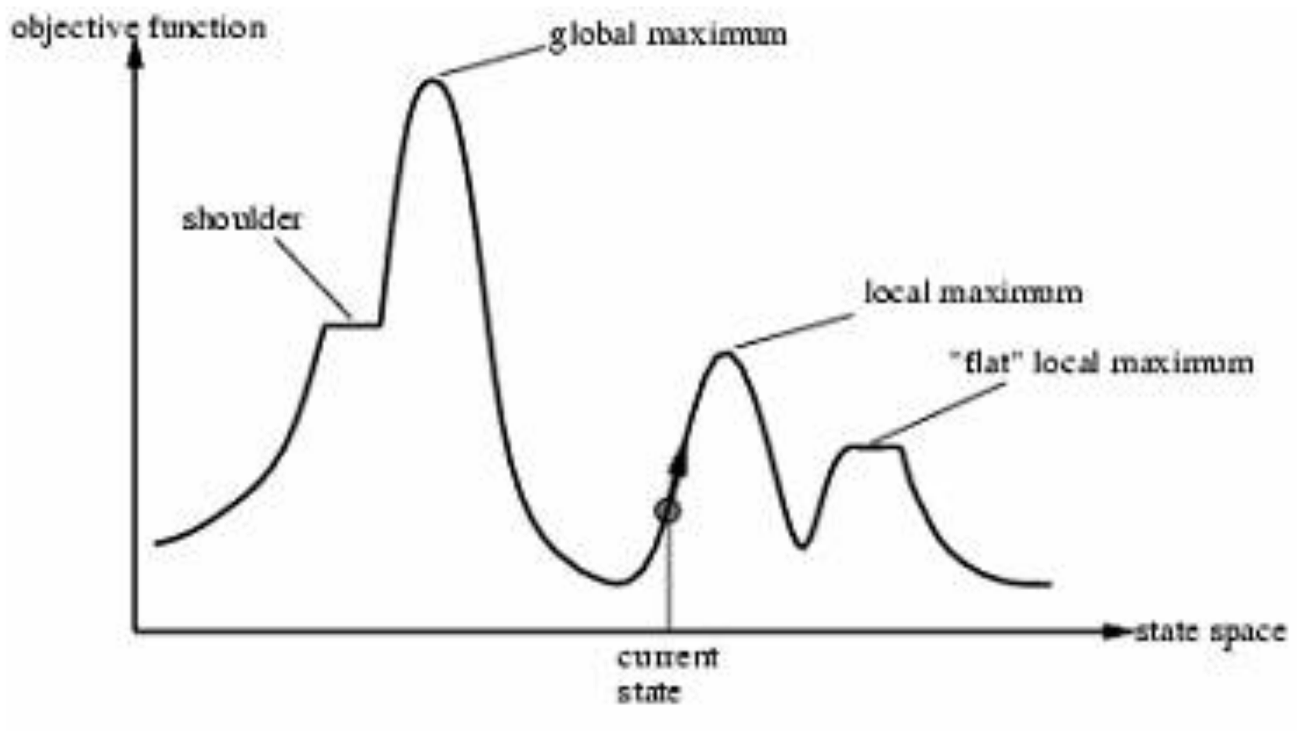
- Birds do not initially know where that area is
- Birds can communicate with the entire flock to determine the globally best location
- Birds also remember their own personal best locations

Particle Swarm Optimization

- In this example, the food concentration describes the search space
- Birds represent candidate solutions to a problem, referred to as *particles*
- A particles desirability is determined using a fitness function for the problem at hand
- In our bird example, the fitness function of a position would be the concentration of food in the immediate area

Particle Swarm Optimization

Particles collaborate to find the global maxima:



Particle Swarm Optimization

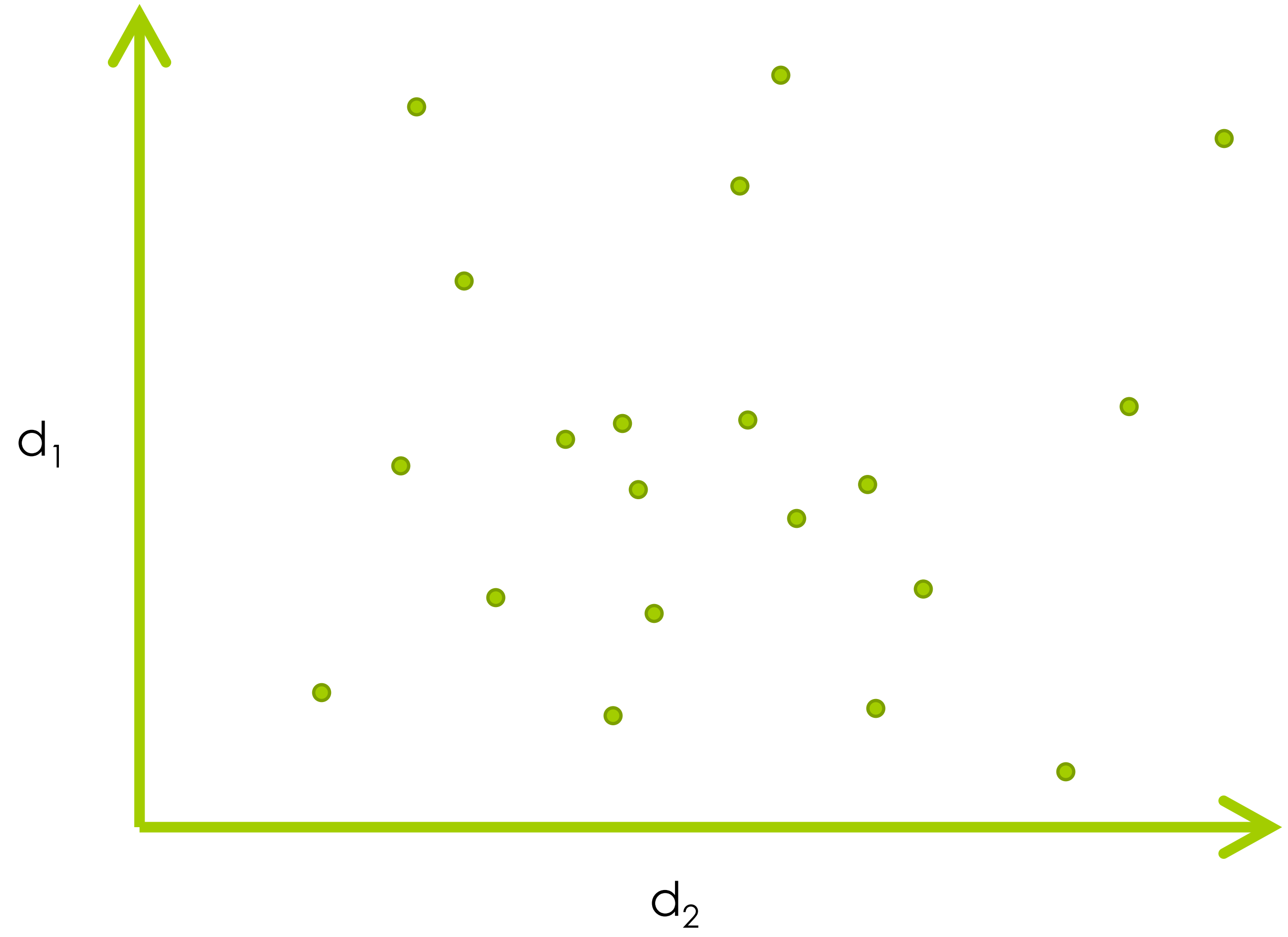
- A particle maintains two things:
  - A **position** in the search space
  - A **velocity** indicating each step size
- Throughout the search, the position and velocity of each particle in the swarm is continuously updated in an attempt to find the global optima

Particle Swarm Optimization

- Over a number of **iterations**, particles move towards two positions:
  - The highest quality position among all positions that the particle has encountered. Referred to as the **personal best**.
  - The highest quality position among all positions that the entire swarm has encountered. Referred to as the **global best.**

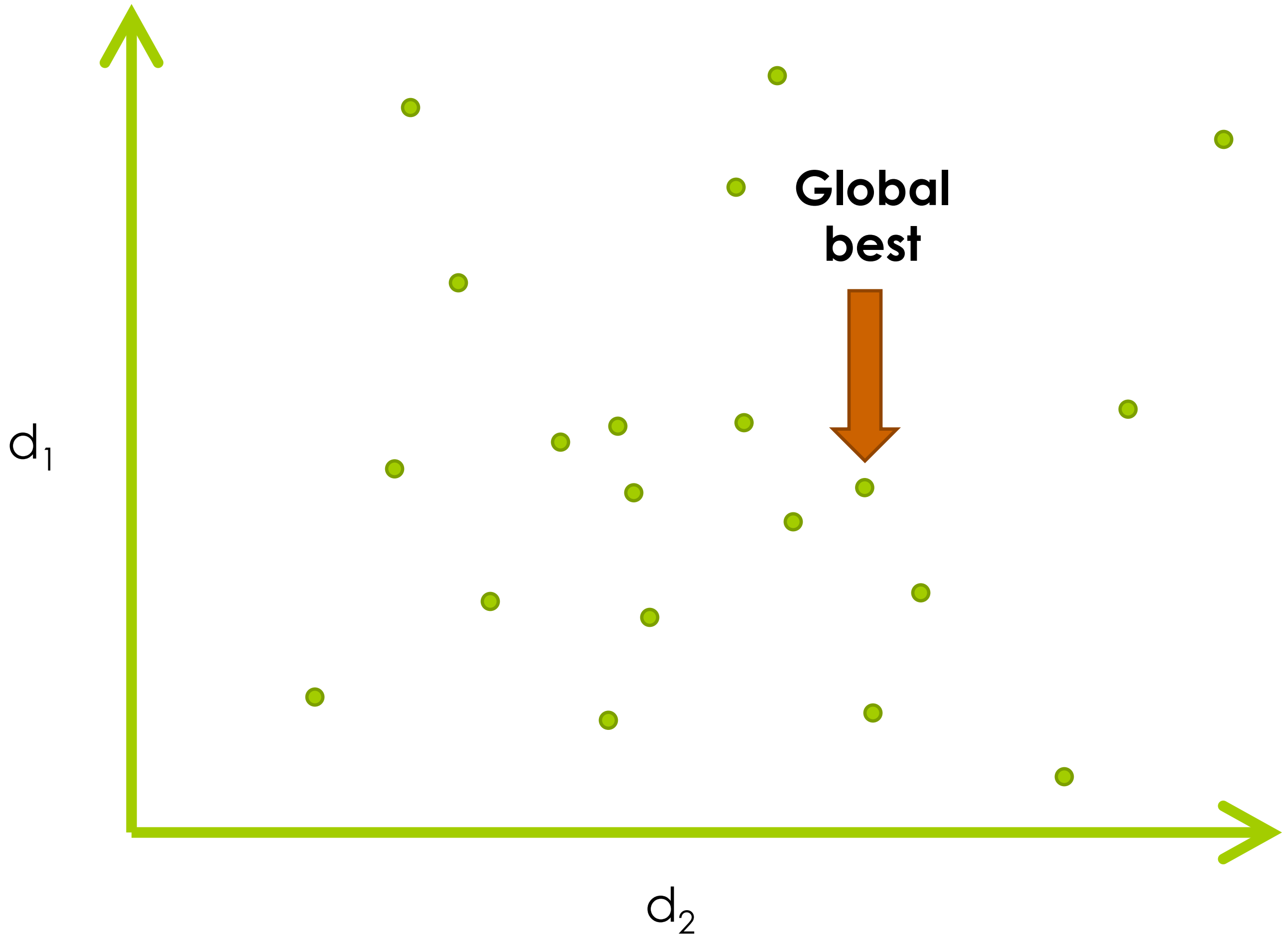
Particle Swarm Optimization

Random initialization:



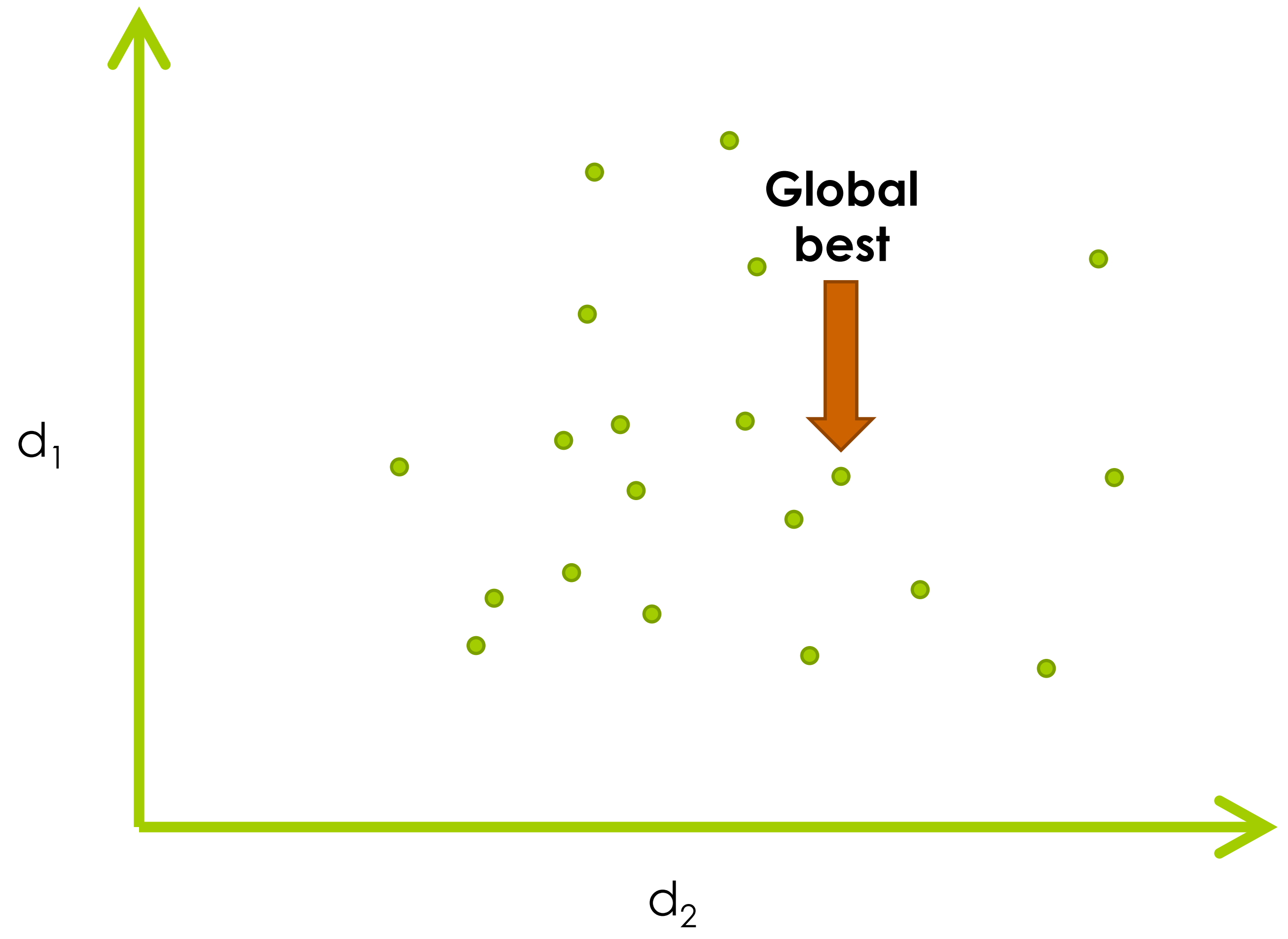
Particle Swarm Optimization

Random initialization:



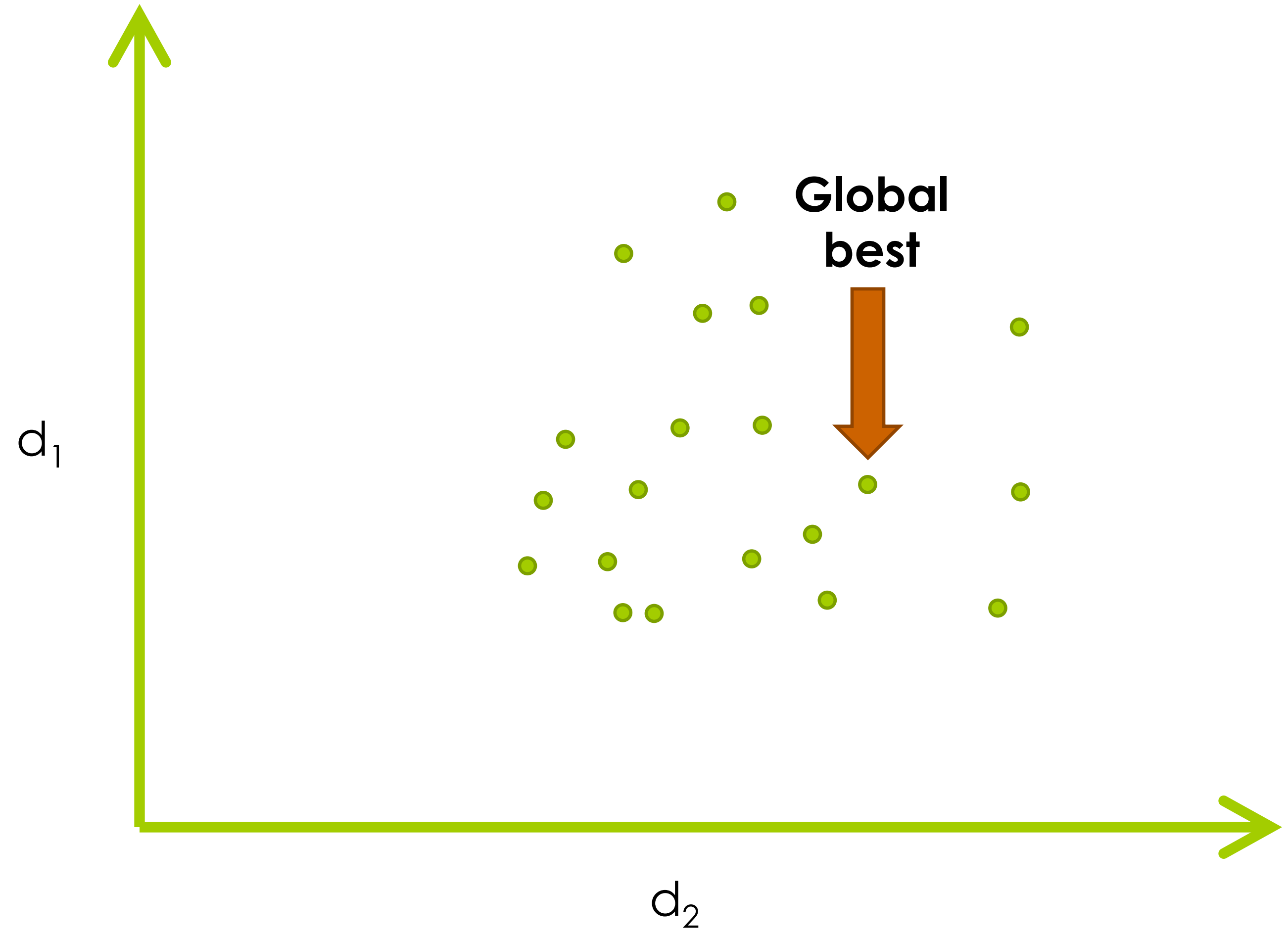
Particle Swarm Optimization

Iteration 1:



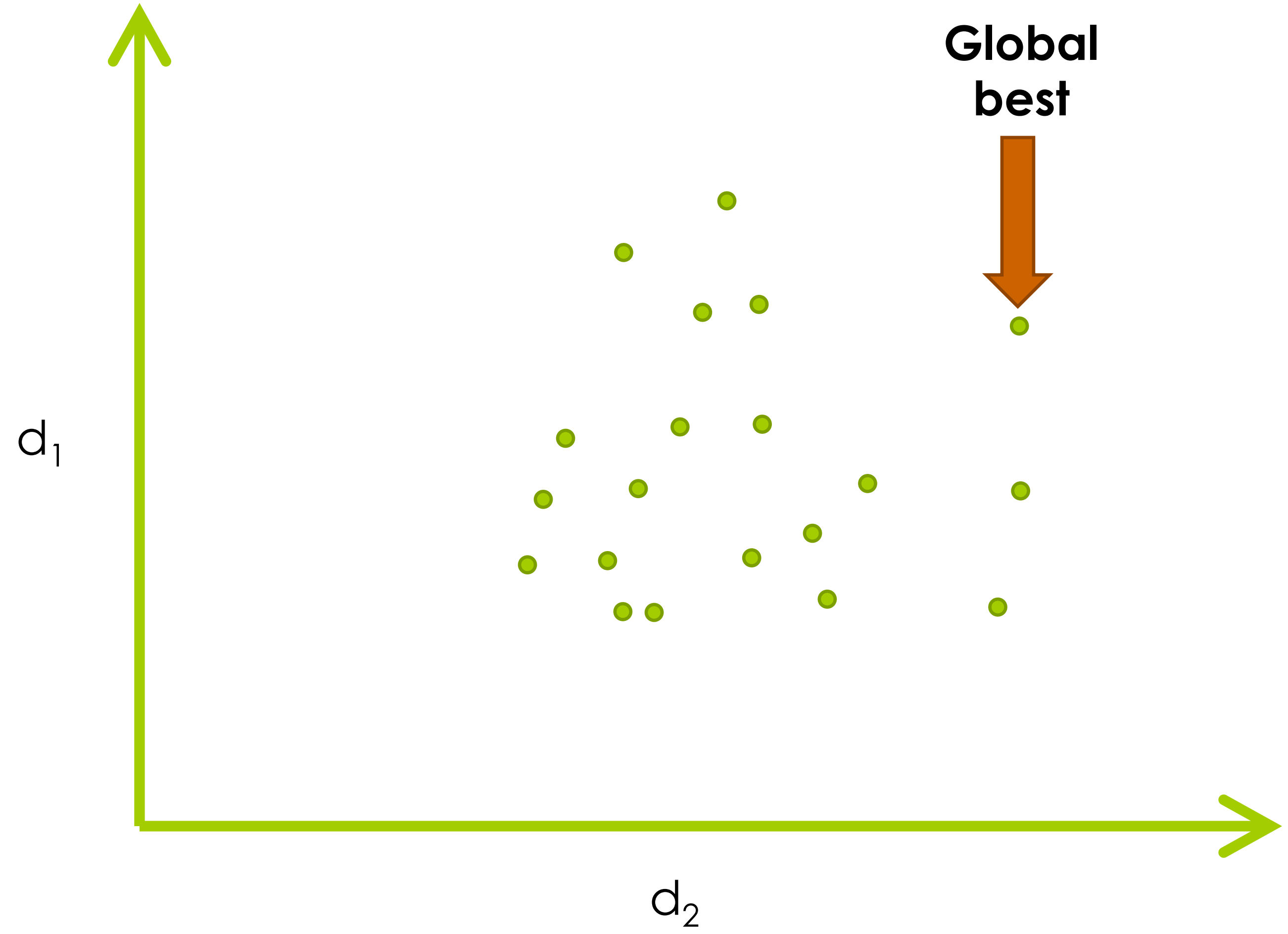
Particle Swarm Optimization

Iteration 2:



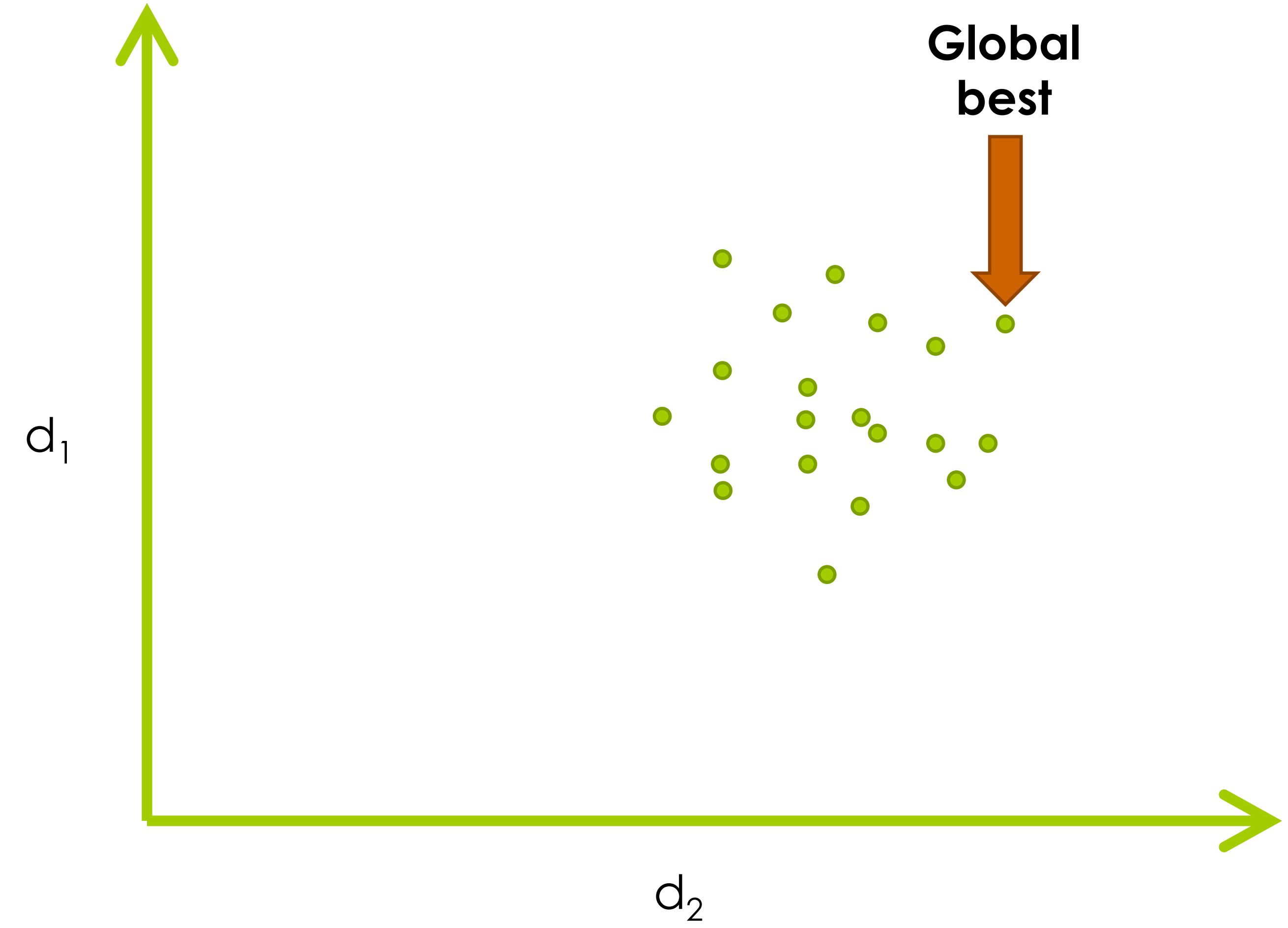
Particle Swarm Optimization

Iteration 2:



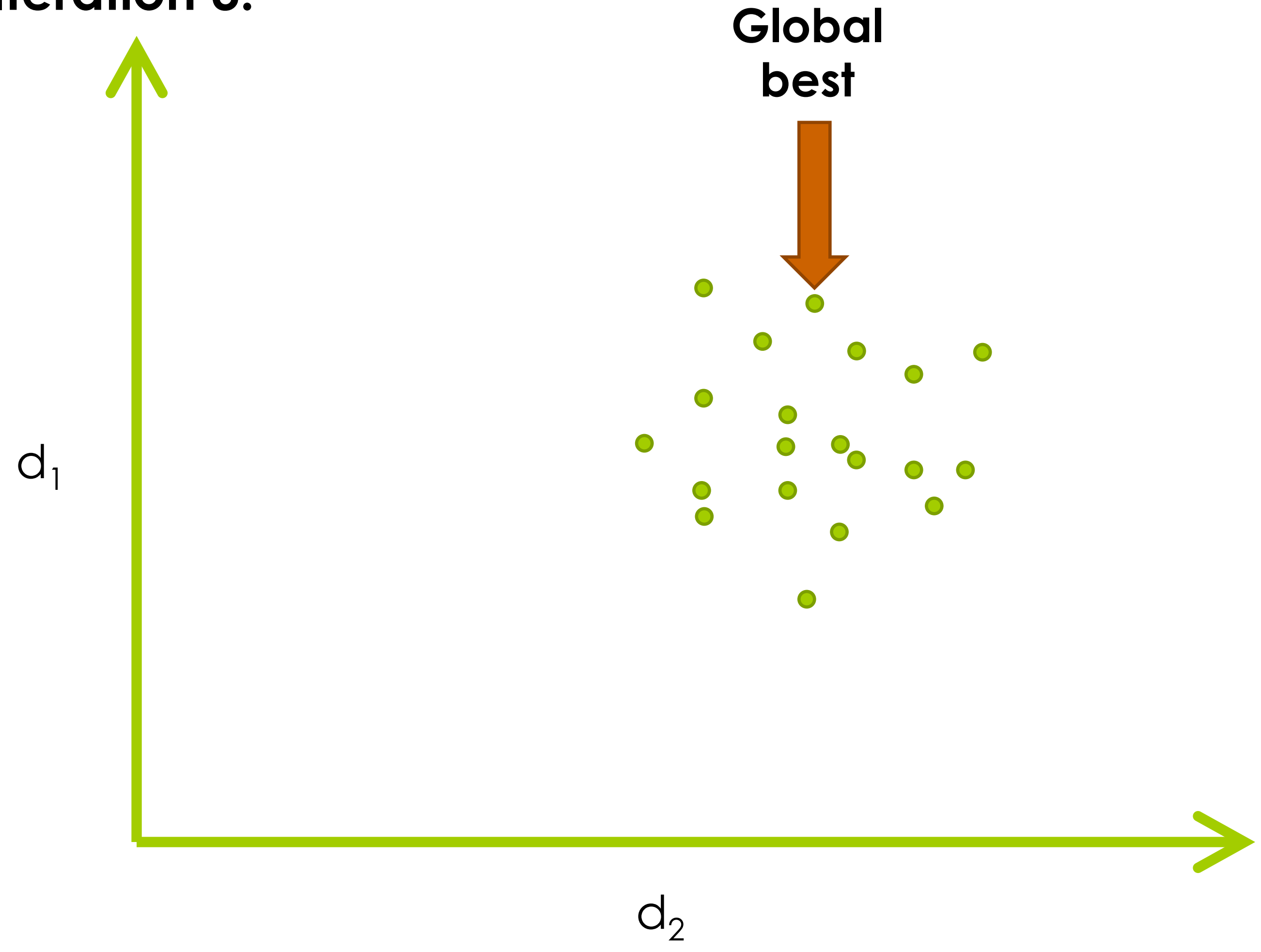
Particle Swarm Optimization

Iteration 3:



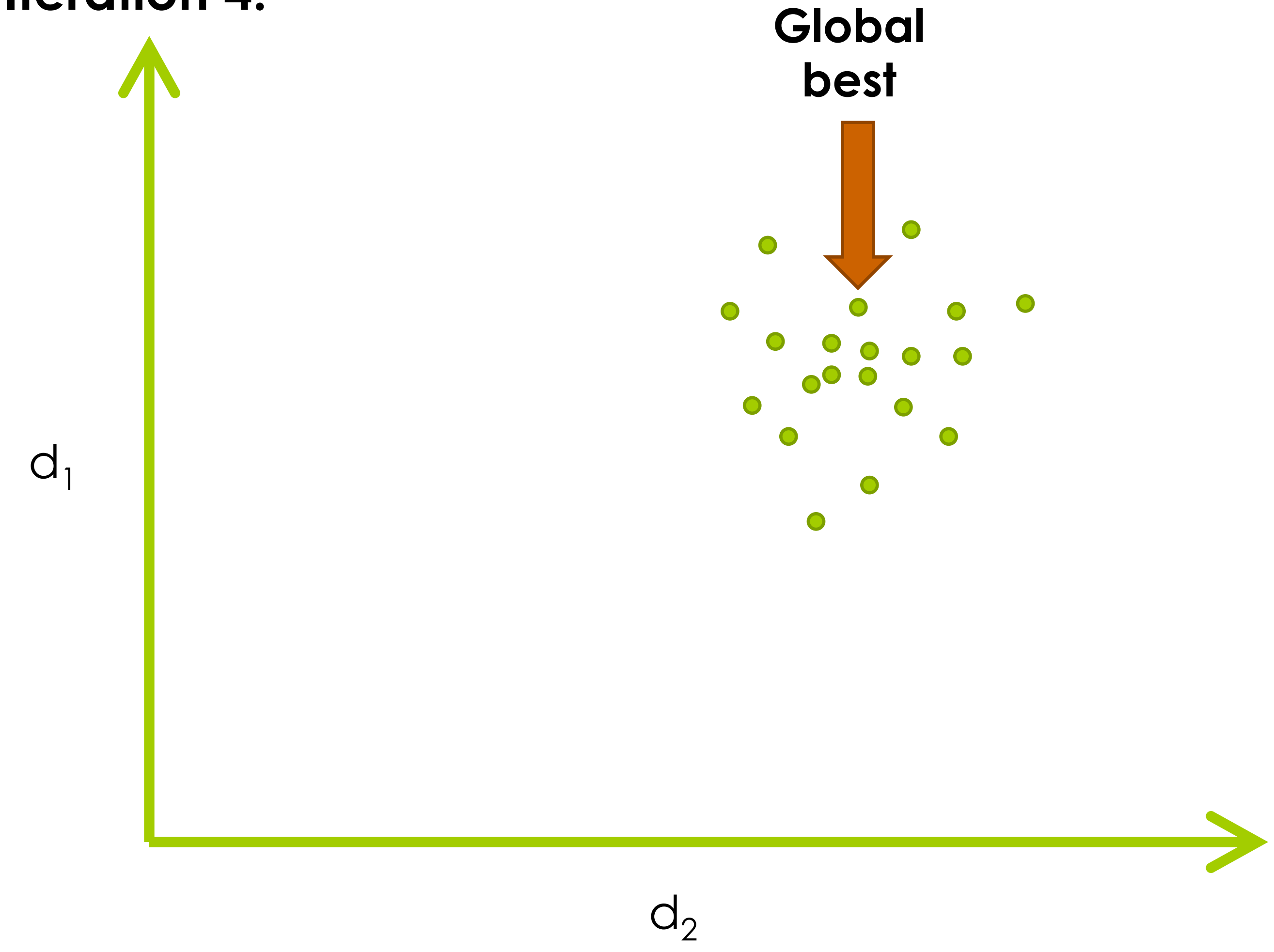
Particle Swarm Optimization

Iteration 3:



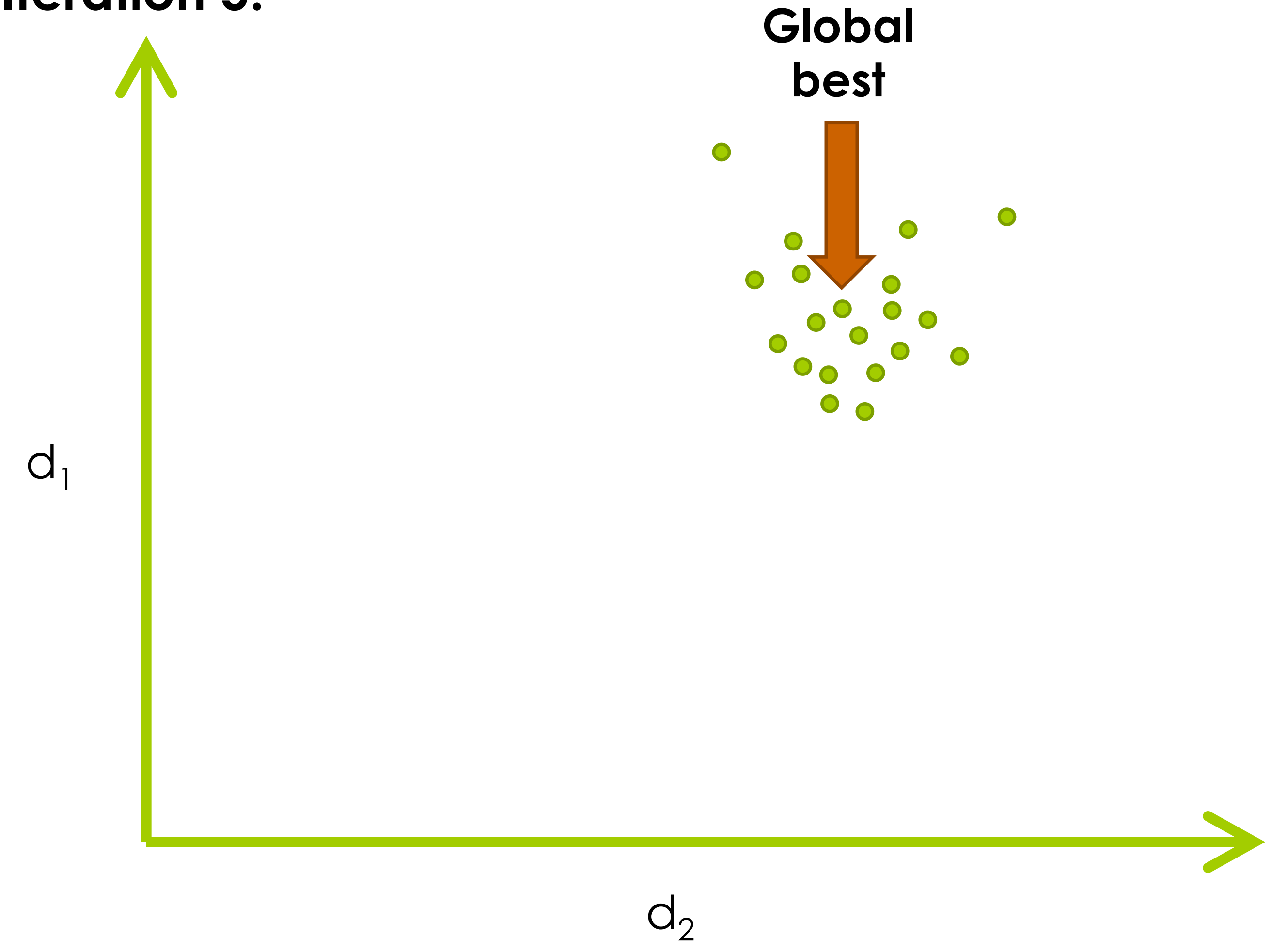
Particle Swarm Optimization

Iteration 4:



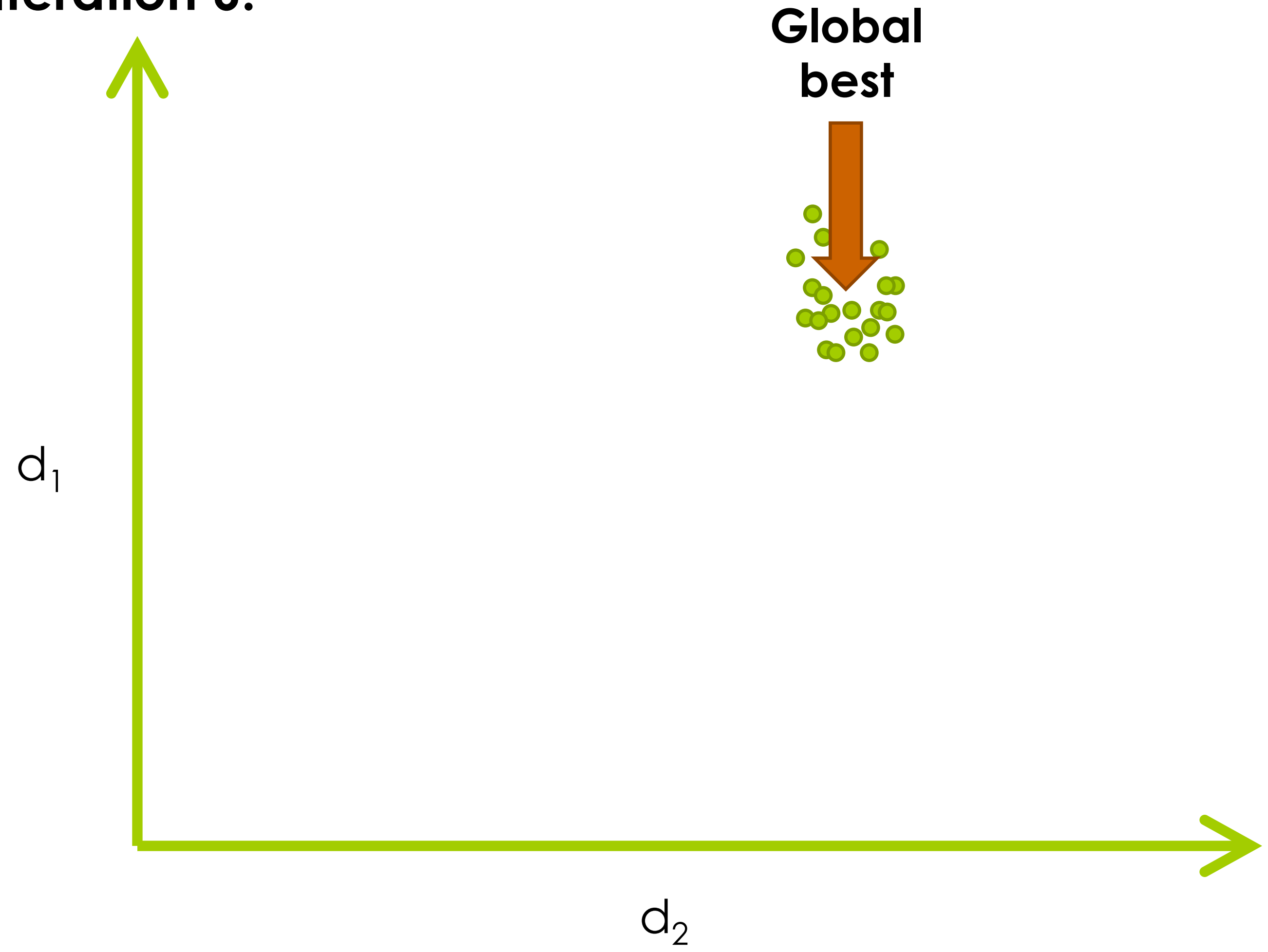
Particle Swarm Optimization

Iteration 5:



Particle Swarm Optimization

Iteration 6:



Particle Swarm Optimization

- At each iteration, particles alter their position by first calculating a step-size(velocity)

- For this purpose, two equations are used:
  - Velocity update equation
  - Position update equation

Velocity Update Equation

$$S.\vec{v}_i(t+1) =$$

$$\big\uparrow$$

The velocity at time t+1

Velocity Update Equation

$$S.\vec{v}_i(t+1) = \quad \underbrace{S.\vec{v}_i(t)}_{\text{The velocity at time t}}$$

Velocity Update Equation

$$S.\vec{v}_i(t+1) = \quad S.\vec{v}_i(t) + \quad \underset{\substack{\Uparrow \\ \textbf{Influence of the Personal Best}}}{(S.\vec{y}_i(t) - S.\vec{x}_i(t))} + \quad \underset{\substack{\Uparrow \\ \textbf{Influence of the Global Best}}}{(S.\vec{\hat{y}}(t) - S.\vec{x}_i(t))}$$

Velocity Update Equation

$$S.\vec{v}_i(t+1) = \quad S.\vec{v}_i(t) + \quad \underset{\uparrow}{\vec{r}_1}(S.\vec{y}_i(t) - S.\vec{x}_i(t)) + \quad \underset{\uparrow}{\vec{r}_2}(S.\vec{\hat{y}}(t) - S.\vec{x}_i(t))$$

Random float in the range [0,1]

Random float in the range [0,1]

“Inertial Weight” value typically in the range [0,2]

$$S.\vec{v}_i(t+1) = \omega S.\vec{v}_i(t) + c_1\vec{r}_1(S.\vec{y}_i(t) - S.\vec{x}_i(t)) + c_2\vec{r}_2(S.\vec{\hat{y}}(t) - S.\vec{x}_i(t))$$

“Cognitive Weight” value typically in the range [0,2]

“Social Weight” value typically in the range [0,2]

Position Update Equation

$$S.\vec{x}_i(t+1) =$$

$\uparrow$

Position at time t+1

Position Update Equation

$$S.\vec{x}_i(t+1) = S.\vec{x}_i(t)$$

Position at time t

Position Update Equation

$$S.\vec{x}_i(t+1) = S.\vec{x}_i(t) + S.\vec{v}_i(t+1)$$

Velocity at time t+1

$$S.\vec{v}_i(t+1) = \omega S.\vec{v}_i(t) + c_1\vec{r}_1(S.\vec{y}_i(t) - S.\vec{x}_i(t)) + c_2\vec{r}_2(S.\vec{\hat{y}}(t) - S.\vec{x}_i(t))$$

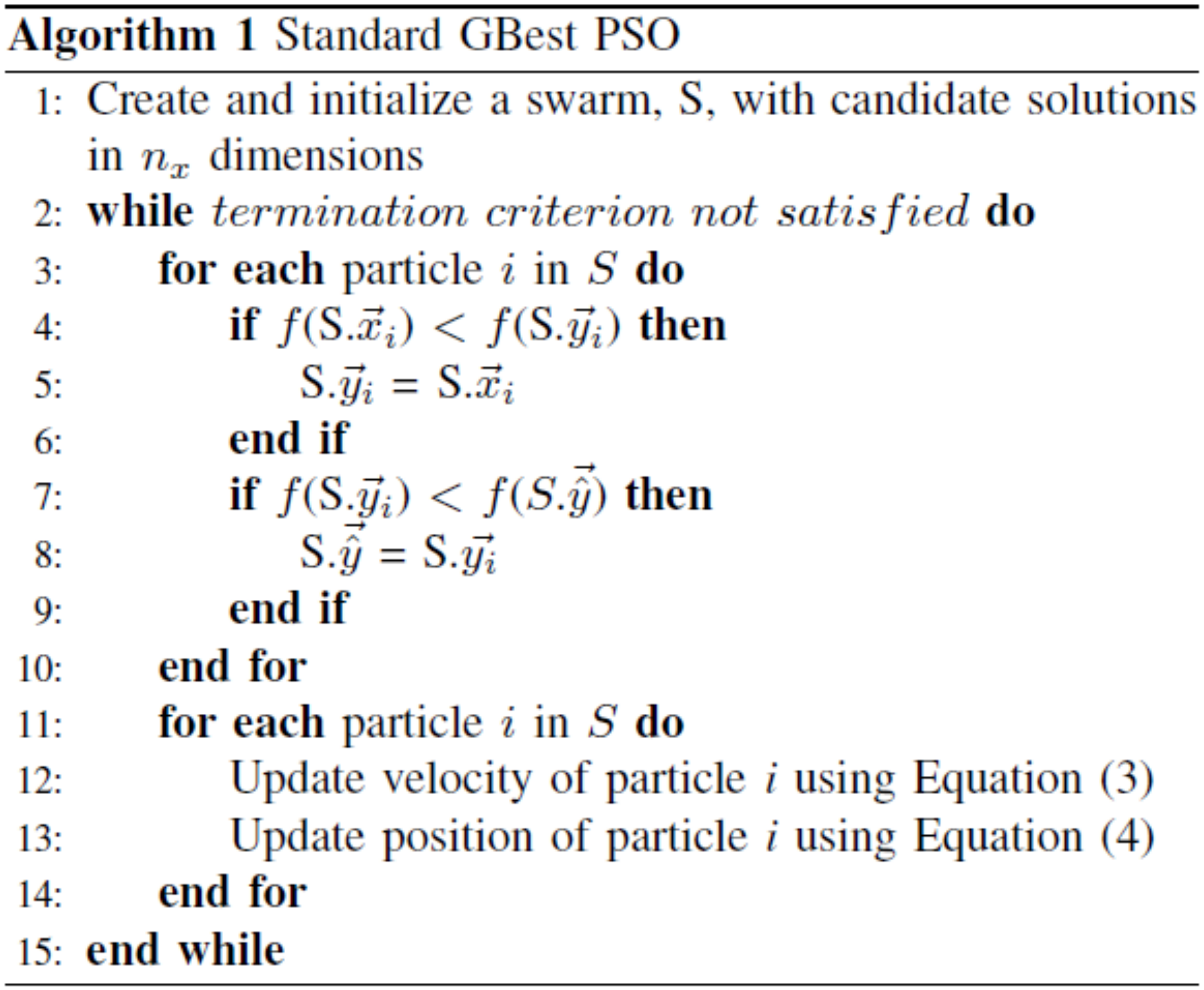
$$S.\vec{x}_i(t+1) = S.\vec{x}_i(t) + S.\vec{v}_i(t+1)$$

- Cognitive weight(c1) – influence of the personal best position found (pbest)
- Social weight(c2) – influence of the swarm collective via the global best position found (gbest)
- Inertial weight(ω) – influence of the previous computed velocity Random, stochastic component

PSO Algorithm

The basic high-level PSO algorithm is:

```
**While** not at MAX_ITERATION **do**
        Update personal bests
        Update global best
        Update particle positions
        iterations++;
**End**
```



Algorithm 1 Standard GBest PSO

```
1: Create and initialize a swarm, S, with candidate solutions
   in $n_x$ dimensions
2: while $termination\ criterion\ not\ satisfied$ do
3:     for each particle $i$ in $S$ do
4:         if $f(\mathrm{S}.\vec{x}_i) < f(\mathrm{S}.\vec{y}_i)$ then
5:             $\mathrm{S}.\vec{y}_i = \mathrm{S}.\vec{x}_i$
6:         end if
7:         if $f(\mathrm{S}.\vec{y}_i) < f(S.\vec{\hat{y}})$ then
8:             $\mathrm{S}.\vec{\hat{y}} = \mathrm{S}.\vec{y}_i$
9:         end if
10:    end for
11:    for each particle $i$ in $S$ do
12:        Update velocity of particle $i$ using Equation (3)
13:        Update position of particle $i$ using Equation (4)
14:    end for
15: end while
```

Observations in Previous Literature

- Can use a ring topology instead of star, resulting in neighbourhoods of particle attraction
- Instead of a global best, each particle uses a local neighbourhood best
- It is shown in [2] that setting initial particle velocity to 0 gives better performance

PSO for Travelling Salesman Problem

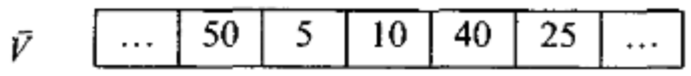
- PSO is designed for continuous domains
- Requires modification for TSP since it is a discrete permutation problem
- For TSP, standard particle position update is no longer valid – possible to have duplicate cities (illegal)
- Need to change the way that particle positions are updated

PSO for Travelling Salesman Problem

- **Idea:** Instead of using velocity as step size, use it as a probability of swapping cities
- Use the calculated probabilities to swap dimensions randomly between a particle and the swarm global best

PSO for Travelling Salesman Problem: Position Update

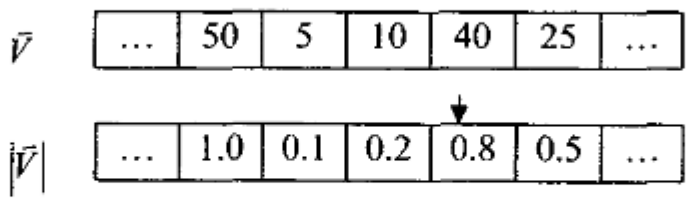
Step 1: Calculate the velocity vector using the regular PSO velocity update equations



$\bar{V}$	...	50	5	10	40	25	...

PSO for Travelling Salesman Problem: Position Update

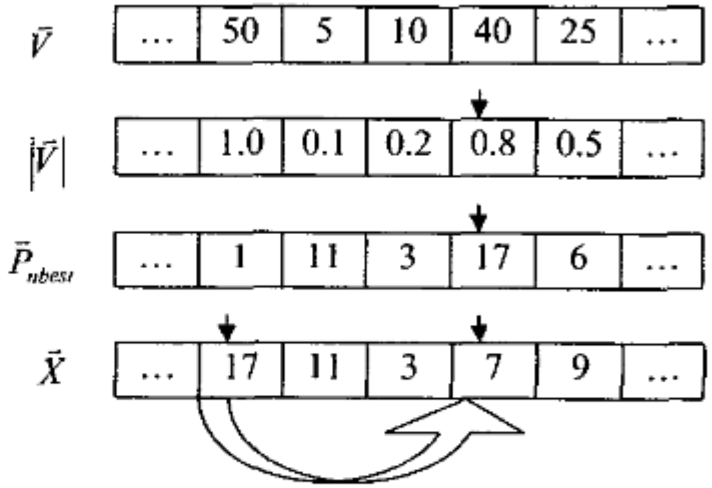
Step 2: Normalize the absolute value of the velocity by dividing by the maximum city index (50)



$\vec{V}$	…	50	5	10	40	25	…
$\lvert\vec{V}\rvert$	…	1.0	0.1	0.2	0.8	0.5	…

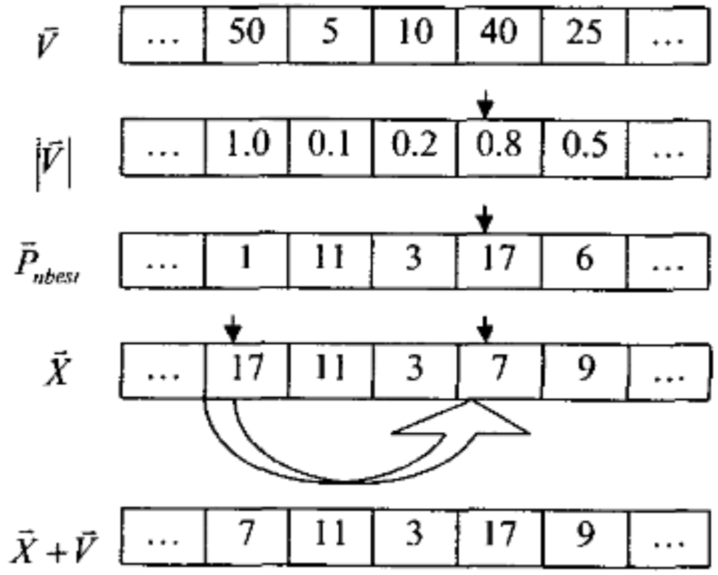
PSO for Travelling Salesman Problem: Position Update

Step 3: Swap each index to the corresponding index of the global best position with probability equal to the calculated velocities



PSO for Travelling Salesman Problem: Position Update

Step 3: Swap each index to the corresponding index of the global best position with probability equal to the calculated velocities

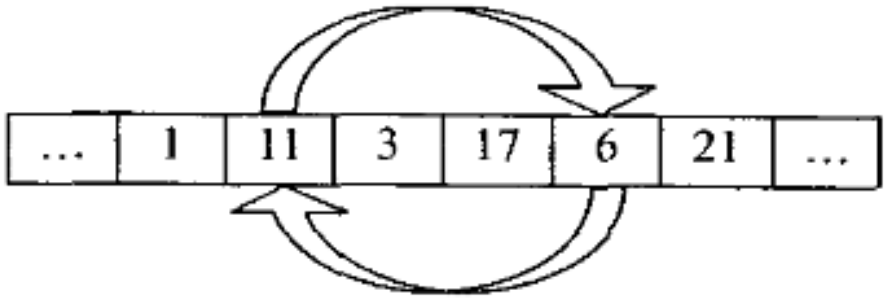


PSO for Travelling Salesman Problem

- What happens if particle is already identical to the global best?
- Swaps no longer produce any impact since particle position remains the same
- Must introduce a **mutation** factor which has a user-defined probability to swap two random indices

PSO for Travelling Salesman Problem

Mutation Example:



PSO for Travelling Salesman Problem

- Advantages of mutation:
  - Particles don’t get stuck when position is identical global best position
  - Provides additional ability to overcome local minima
  - Provides additional exploitation in the search

Binary Discrete PSO

- PSO can also be used to solved problems with binary representations
- **Binary Discrete PSO** introduced in [3] by Kennedy and Eberhart
- Uses traditional velocity equation except inertial weight is removed

$$S.\vec{v}_i(t+1) = \quad S.\vec{v}_i(t) + c_1\vec{r}_1(S.\vec{y}_i(t) - S.\vec{x}_i(t)) + c_2\vec{r}_2(S.\vec{\hat{y}}(t) - S.\vec{x}_i(t))$$

Binary Discrete PSO

- Position of particle x determined as follows:

$$x_i = \{ ■ 1 \ \mathit{if}\ \mathit{rand}() < \mathit{Sigmoid}(vi)\ 0\ \mathit{otherwise}$$

Where Sigmoid(x) is the sigmoid function defined as:

$$\mathit{Sigmoid}(x) = 1/1+e\uparrow -x$$

References

- [1] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *IEEE int'l conference on neural networks*, vol. IV, pp. 1942-1948, 1995.
- [2] A. Engelbrecht, "Particle Swarm Optimization: Velocity Initialization," *in Evolutionary Computation (CEC), 2012 IEEE Congress*. June 2012, pp. 1-8.
- [3] Kennedy, J.; Eberhart, R.C., "A discrete binary version of the particle swarm algorithm," in *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on* , vol.5, no., pp. 4104-4108 vol.5, 12-15 Oct 1997 doi: 10.1109/ICSMC.1997.637339.