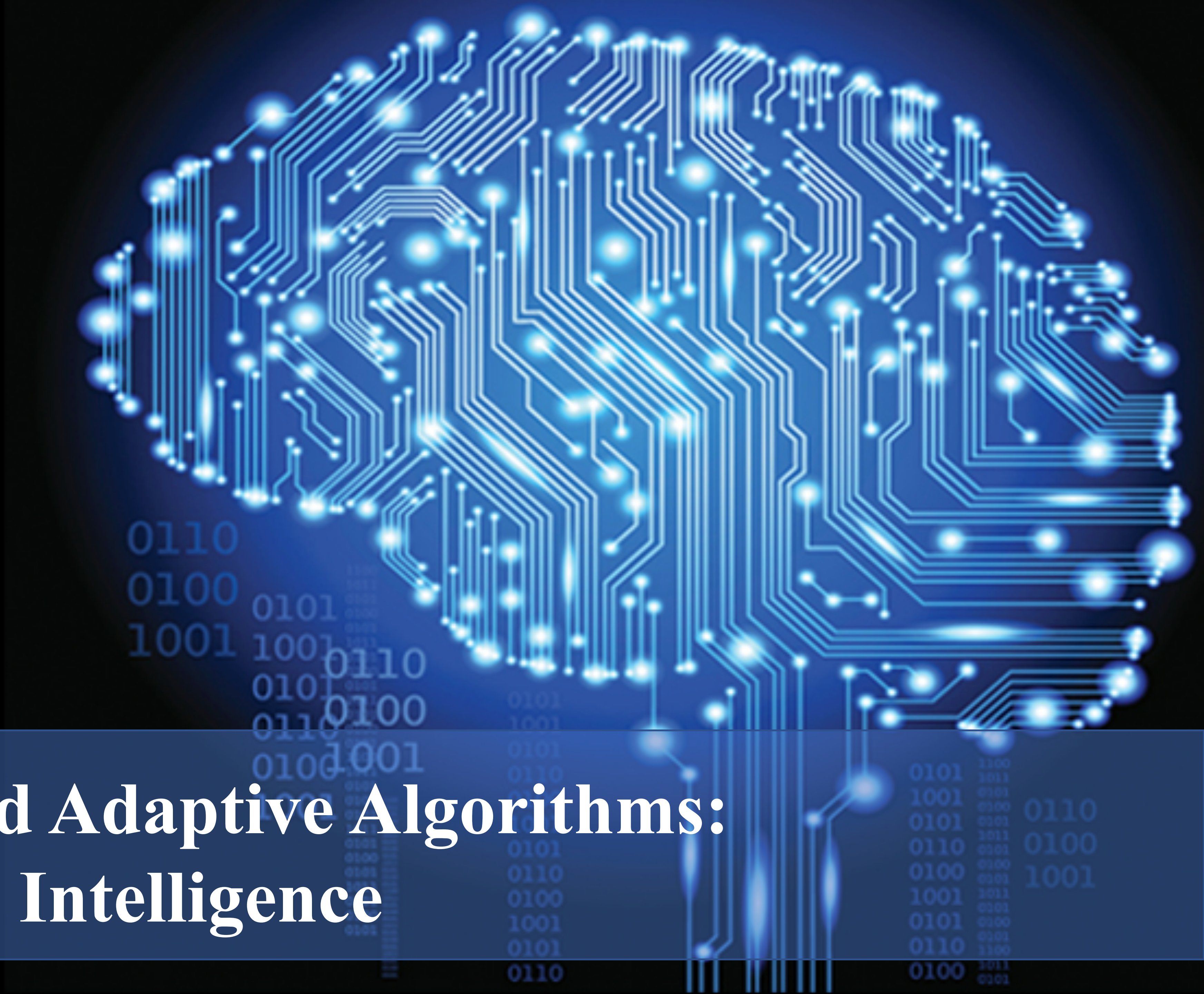
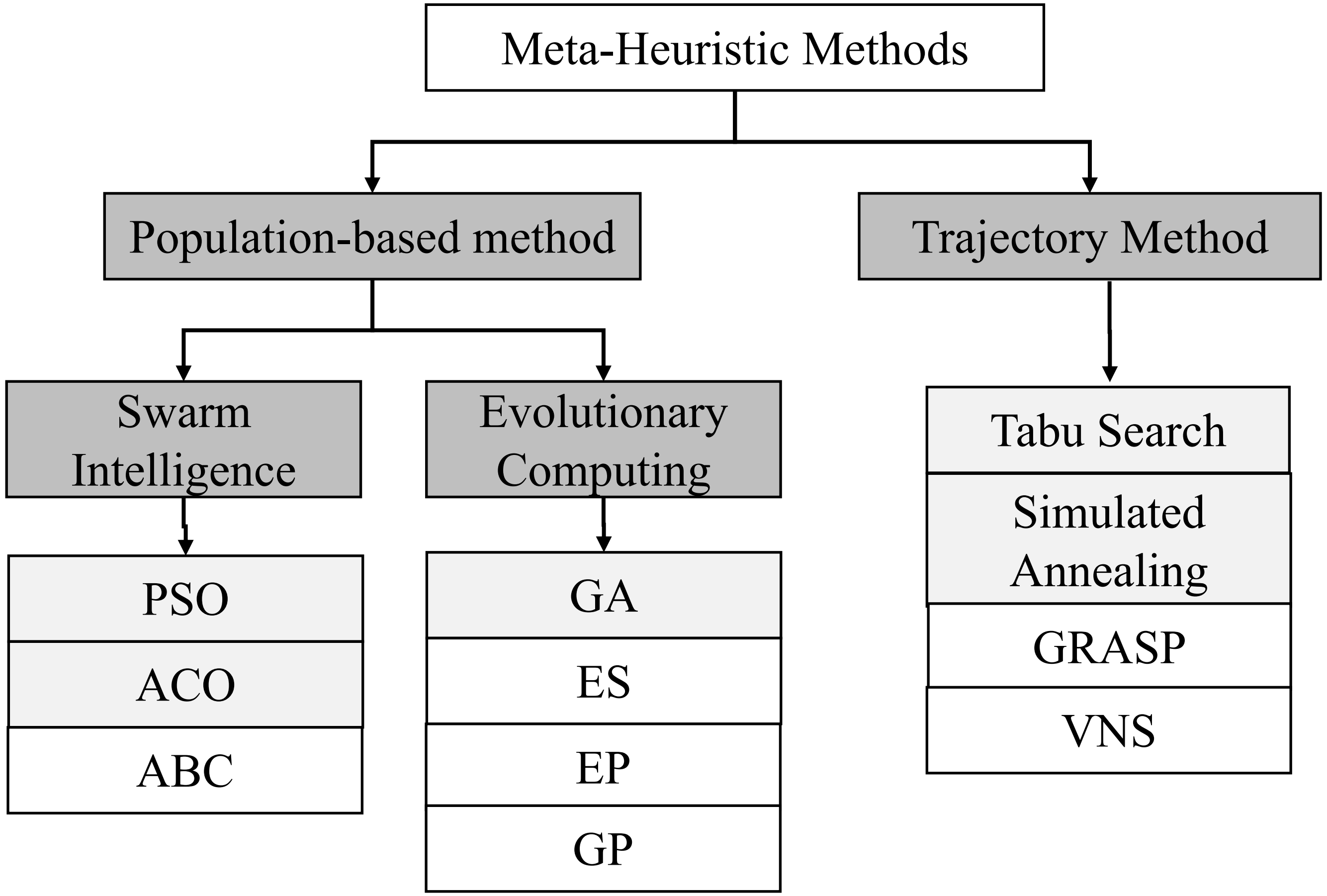
Cooperative and Adaptive Algorithms: Particle Swarm Intelligence



Meta-Heuristics

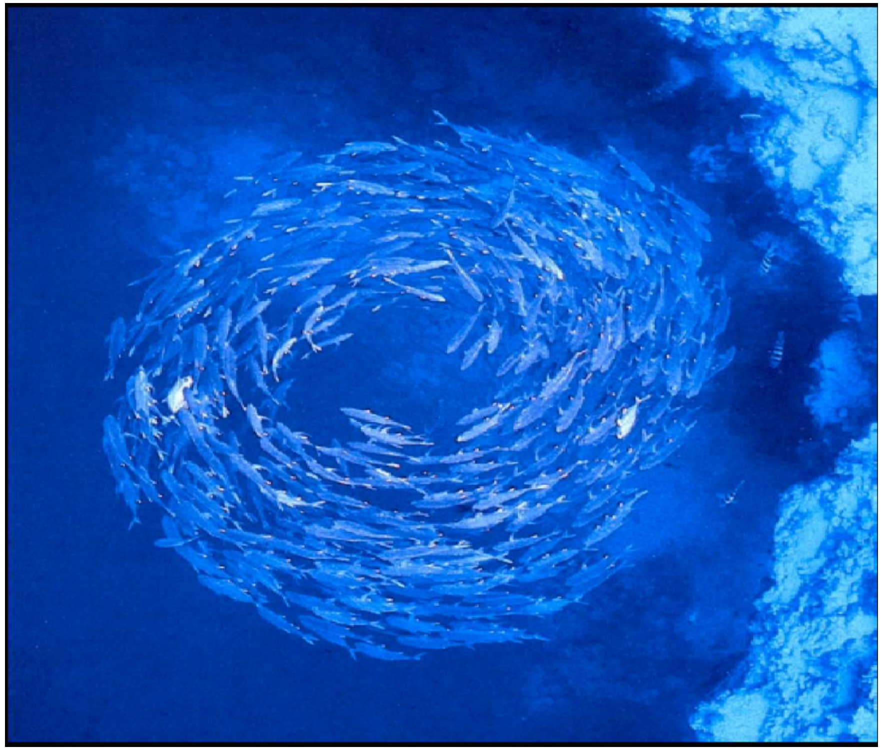


Particle Swarm Intelligence

Introduction

Introduction

Bird flocking – V formation (© Soren Breitling)



Fish schooling (© CORO, CalTech)

Introduction

- The main idea is to ***simulate*** the collective behavior of social animals
- In particular, bird flocking and fish schooling behaviors
- Unlike some animal teams where there is a leader (e.g a pride of lions or a troop of baboons), the interest here in teams that has ***no leader***
- Individuals have **no knowledge of the global behavior** of the group
- They have the ability to move together based on **social interaction** between neighbours

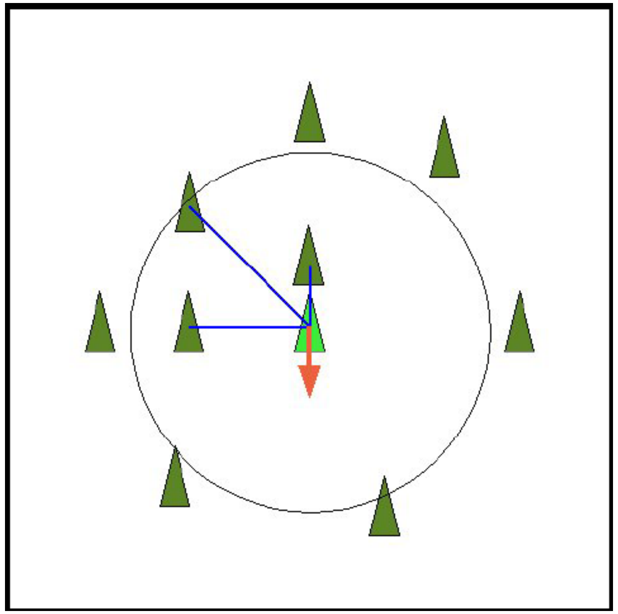
Introduction

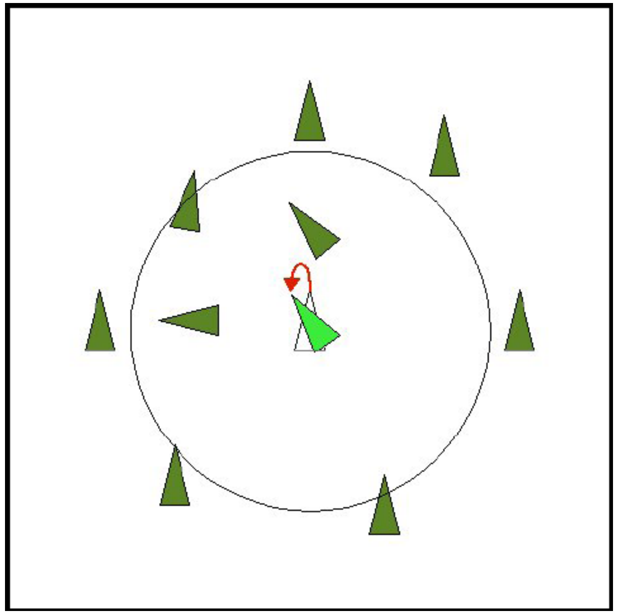
- The first computer program was written by Reynolds in 1986 [1] to simulate swarms for computer graphics and movies,

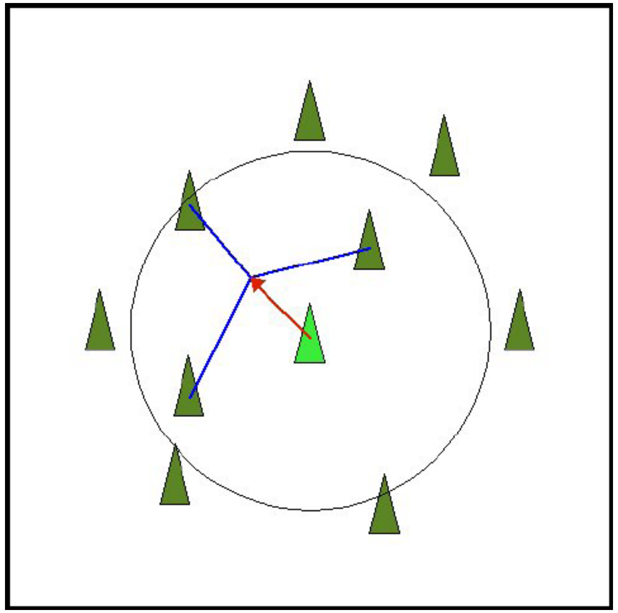
- The work took account of three behaviours:
  - Separation,
  - Alignment,
  - Cohesion.

- For online simulations, refer to http://www.red3d.com/cwr/boids/

Introduction







<u>Separation:</u> Each agent tries to move away from its nearby mates if they are too close (**Collision Avoidance**).

<u>Alignment:</u> Each agent steers towards the average heading of its nearby mates (**Velocity matching**).

<u>Cohesion:</u> Each agent tries to go towards the average position of its nearby mates (**Centering or position control**).

From [3]

Introduction

- Heppner and Grenander [4] used a similar flocking model but added a roost (place for birds to rest) as an attractor for the birds.

- The intent was to provide a computer simulation of a flock of birds to understand the underlying rules that enable synchronous flocking

- Kennedy and Eberhart in 1995 [5, 6], introduced an optimization method based on the simple behavior of emulating the success of neighboring individuals.
  - Followed the same steps taken by Reynolds and adding a ***Roost*** as proposed by Heppner and Grenander

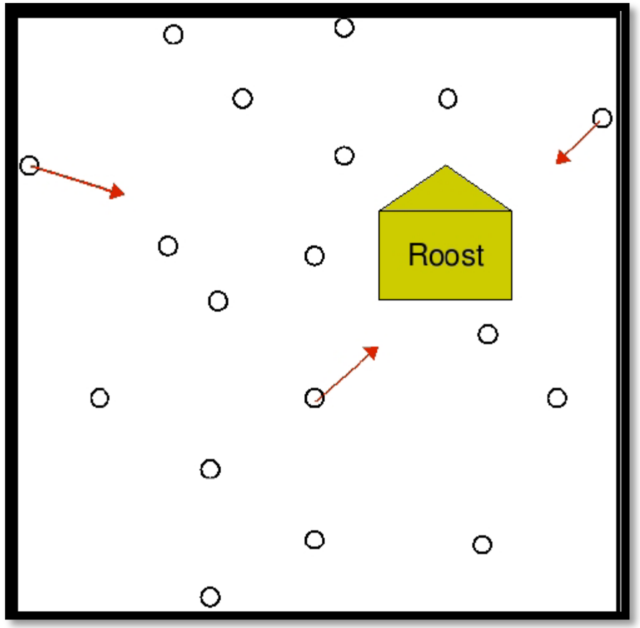
Introduction

- The ***roost*** is in the form of a memory of previous **own best and neighborhood best** positions (referred to ***cornfield***)

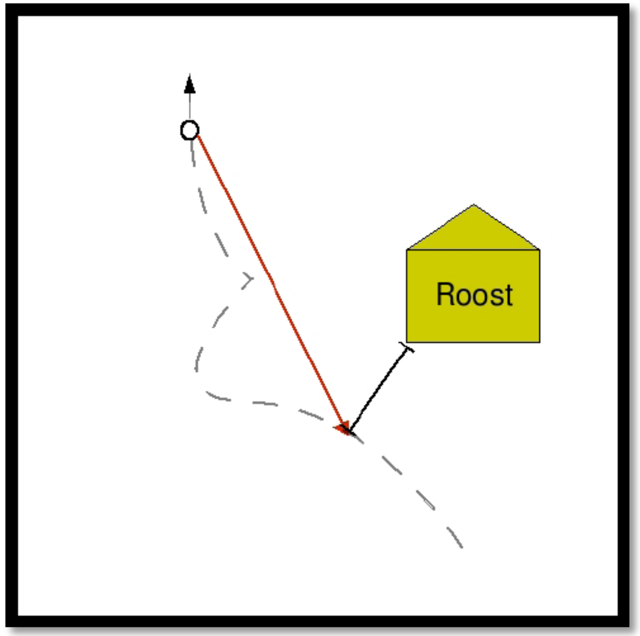
- These two best positions serve as attractor

- By adjusting the positions of the flock proportion to the distance from the best positions, they converge to the goal

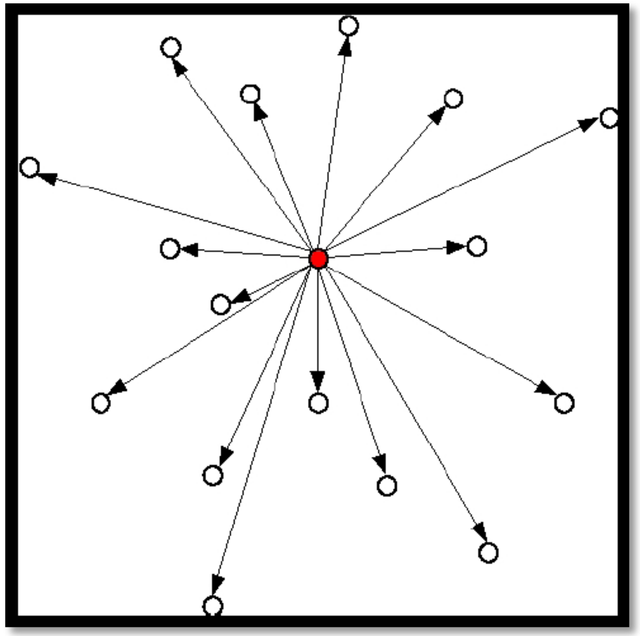
Introduction



All the individuals are attracted to the roost.



Each memorizes the position in which it was closest to the roost.



Each shares its information with all the others.

From [3]

Introduction

- At the end of the simulation, all the individuals landed on the roost,

- It was realized, this could be used to solve optimization problems,

- If the distance to the roost was changed by some unknown function, the individuals land on the minimum

Introduction

- Kennedy and Eberhart called their model **P**article **S**warm **O**ptimization (***PSO***)

- They choose the word ***particle*** to mean individual or candidate solution (in optimization terms) as they felt it is more appropriate for the use with velocity and acceleration

- As their paradigm is a simplified version of bird flocking, they preferred the use of the word ***swarm*** to indicate the population.

- PSO is a population based approach similar to GA and other EC approaches

PSO vs GA

PSO		GA
Swarm	⟷	Population
Particle	⟷	Individual
Fitness	⟷	Fitness
Less fit don't die	⟷	Survival of the fittest
Uses past experience and relationship to neighbours	⟷	Uses crossover and mutations

Particle Swarm Intelligence

Motion

PSO - Introduction

- A stochastic optimization approach that manipulates a number of candidate solutions at once,

- A solution is referred to as a ***particle***, the whole population is referred to as a ***swarm***,

- Each particle holds information essential for its movement.

PSO - Motion

- Each particle holds:
  - Its current position x_i,
  - Its current velocity v_i,
  - The best position it achieved so far, personal best, $pbest_i$ (sometimes p_i for short),
  - The best position achieved by particles in its neighbourhood *Nbest*
  - If the neighbourhood is the whole swarm, the best achieved by the whole swarm is called *global best*, $gbest_i$ (sometimes p_g for short).
  - If the neighbourhood is restricted to few particles, the best is called *local best, lbest* (or p_l)

PSO - Motion

- Each particle adjusts its velocity to move towards its personal best and the swarm neighbourhood best,

- After the velocity is updated, the particle adjusts its position.

PSO - Motion

- **Equations of motion:**

$$v_{t+1}^{id} = w * v_t^{id} + c_1 r_1^{id} \left(pbest_t^{id} - x_t^{id} \right) + c_2 r_2^{id} \left(Nbest_t^{id} - x_t^{id} \right)$$
$$x_{t+1}^{id} = x_t^{id} + v_{t+1}^{id}$$

- **Where**
  - v is the velocity of particle *id,*
  - w is the inertia weight,
  - c_1, c_2 are the acceleration coefficients,
  - r_1,r_2 are randomly generated numbers in [0, 1],
  - x is the position of the particle
  - t is the iteration number,
  - i and d are the particle number and the dimension.

PSO - Motion

$$
\begin{aligned}
v_{t+1}^{id} = \; & w * v_t^{id} && \longrightarrow \textbf{Inertia} \\
& + c_1 r_1^{id}\left(pbest_t^{id} - x_t^{id}\right) && \longrightarrow \textbf{Cognitive component} \\
& + c_2 r_2^{id}\left(Nbest_t^{id} - x_t^{id}\right) && \longrightarrow \textbf{Social component}
\end{aligned}
$$

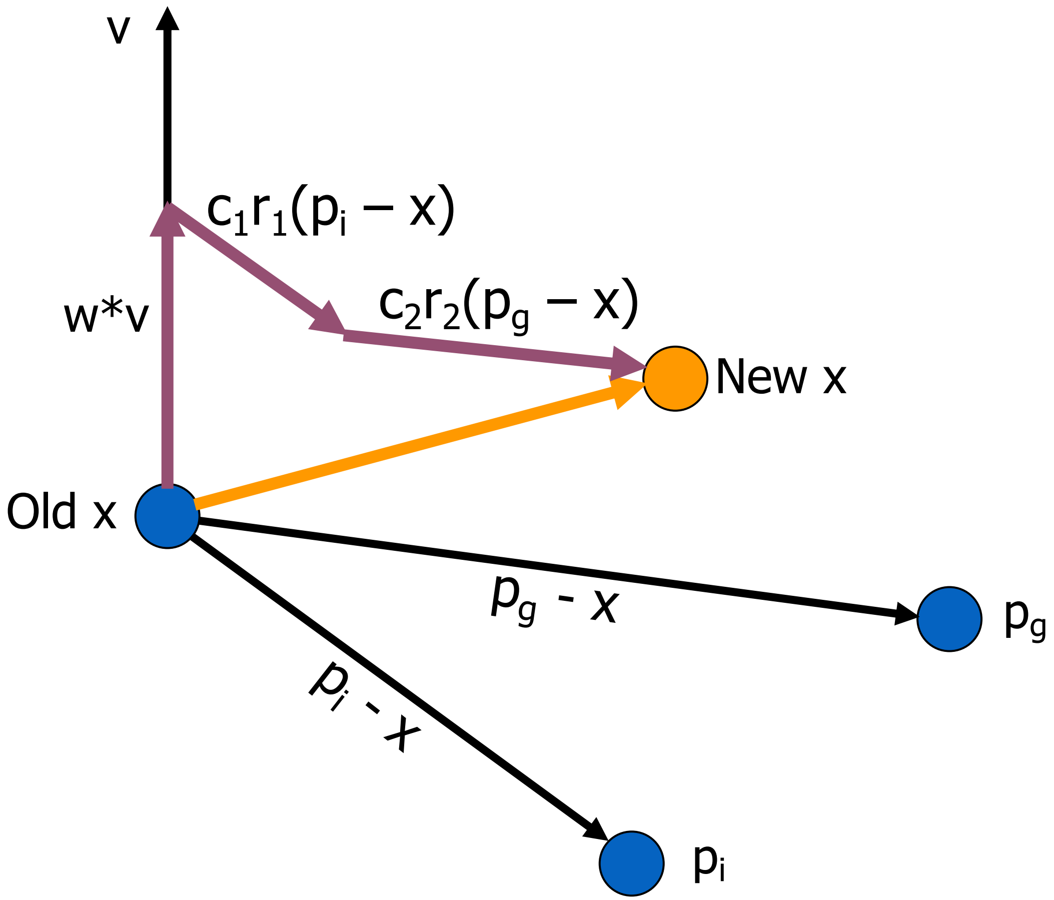
- The inertia component accommodates the fact that a bird (particle) cannot suddenly change its direction of movement,
- The c_1 and c_2 factors balance the weights in which each particle:
  - Trusts its own experience, *cognitive component*,
  - Trusts the swarm experience, *social component*.

PSO - Motion

- Note that the random numbers are generated for each dimension and not for each particle,
  - if the function you are optimizing has 3 variables, the particle will have 3 dimensions

- If the numbers are generated for each particle, the algorithm is referred to as ***linear PSO***, which usually produces sub-optimal solutions in comparison with PSO.

PSO - Motion



- An important factor to set is the maximum velocity allowed for the particles V_{max} :
  - If too high, particles can fly past optimal solutions,
  - If too low, particles can get stuck in local optima.

- Usually set according to the domain of the search space.

PSO - Motion

- After that, each particle updates its own personal best (assuming a minimization problem):

$$pbest_{t+1}^{i} = \begin{cases} x_{t+1}^{i} & ,\text{if } f\left(x_{t+1}^{i}\right) \le f\left(pbest_{t}^{i}\right) \\ pbest_{t}^{i} & ,\text{otherwise} \end{cases}$$

- After that, each swarm updates its global best (assuming a minimization problem):

$$\mathrm{Nbest}_{t+1}^{i} = \arg \min_{pbest_{t+1}^{i} \in N} \mathrm{f}\left(\mathrm{pbest}_{\mathrm{t+1}}^{\mathrm{i}}\right)$$

Particle Swarm Intelligence

Algorithms

PSO – A simple algorithm (Synchronous update)

- Initialize the swarm,
- While *termination criteria* is not met
  - For each particle
    - Update the particle's velocity,
    - Update the particle's position,
    - Update the particle's personal best,

    end for
  - Update the Nbest,

  end while

PSO – A different algorithm (Asynchronous update)

- Initialize the swarm,
- While *termination criteria* is not met
  - For each particle
    - Update the particle's velocity,
    - Update the particle's position,
    - Update the particle's personal best,
    - Update the Nbest,

    end for

  end while

The neighbourhood best update is moved into the particles update loop

PSO Algorithms

- ***Synchronous*** version, if the neighbourhood best is updated after all the population has been updated as well,
- ***Asynchronous*** version, if the neighbourhood best is updated after every particle,
- Asynchronous version usually produces better results as it causes the particles to use a more up-to-date information.
- Termination Criteria can be:
  - Max number of iterations
  - Max number of function evaluations
  - Acceptable solution has been found
  - No improvement over a number of iterations.