NTNU

Norwegian University of Science and Technology

Lecture 8

Particle Swarm Optimization (PSO)

Kazi Shah Nawaz Ripon

ksripon@ntnu.com

Recap: Swarm Intelligence

bird flocking

ants

fish schooling

animal herding

bacteria molding

NTNU

Swarm Intelligence

- "any attempt to design algorithms or distributed problem-solving devices inspired by the **collective behavior** of social insect colonies and other animal societies" [Bonabeau, Dorigo, Theraulaz: Swarm Intelligence]
- Social interactions (*locally shared knowledge*) provides the basis for unguided problem solving.
- Solves optimization problems.

Swarm Intelligence

- **Collective system** capable of accomplishing difficult tasks in dynamic and varied environments without any external guidance or control and with no central coordination.
- An artificial intelligence (AI) technique based on the collective behavior in decentralized, self-organized systems.
- Achieving a collective performance which could not normally be achieved by an individual acting alone.
- Generally made up of agents who interact with each other and the environment.
- Based on group behavior found in nature.

NTNU

Recap: *Swarm Intelligence*

NTNU

What is a Swarm?

- A loosely structured collection of interacting agents.
  - Agents:
    - Individuals that belong to a group (but are not necessarily identical) .
    - They contribute to and benefit from the group.
    - They can recognize, communicate, and/or interact with each other.
- A swarm is better understood if thought of as agents exhibiting a **collective behavior**.

Swarming – The Definition

- Aggregation of similar animals/insects, generally cruising in the same direction.
    - Termites swarm to build colonies.
    - Birds swarm to find food.
    - Bees swarm to reproduce.

Swarming is Powerful

- Swarms can achieve things that an individual cannot (**Collective Bahaviour**).

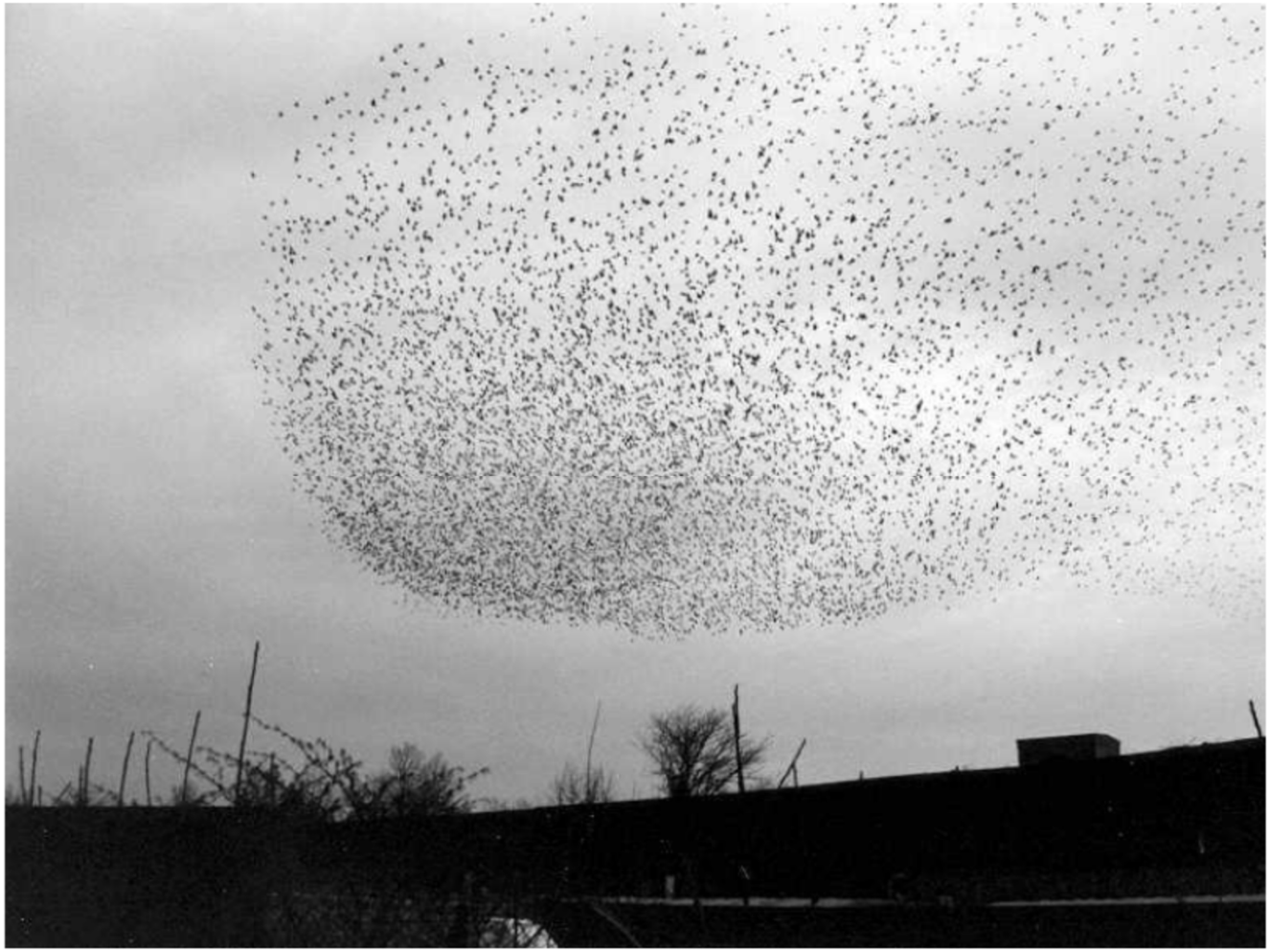
NTNU

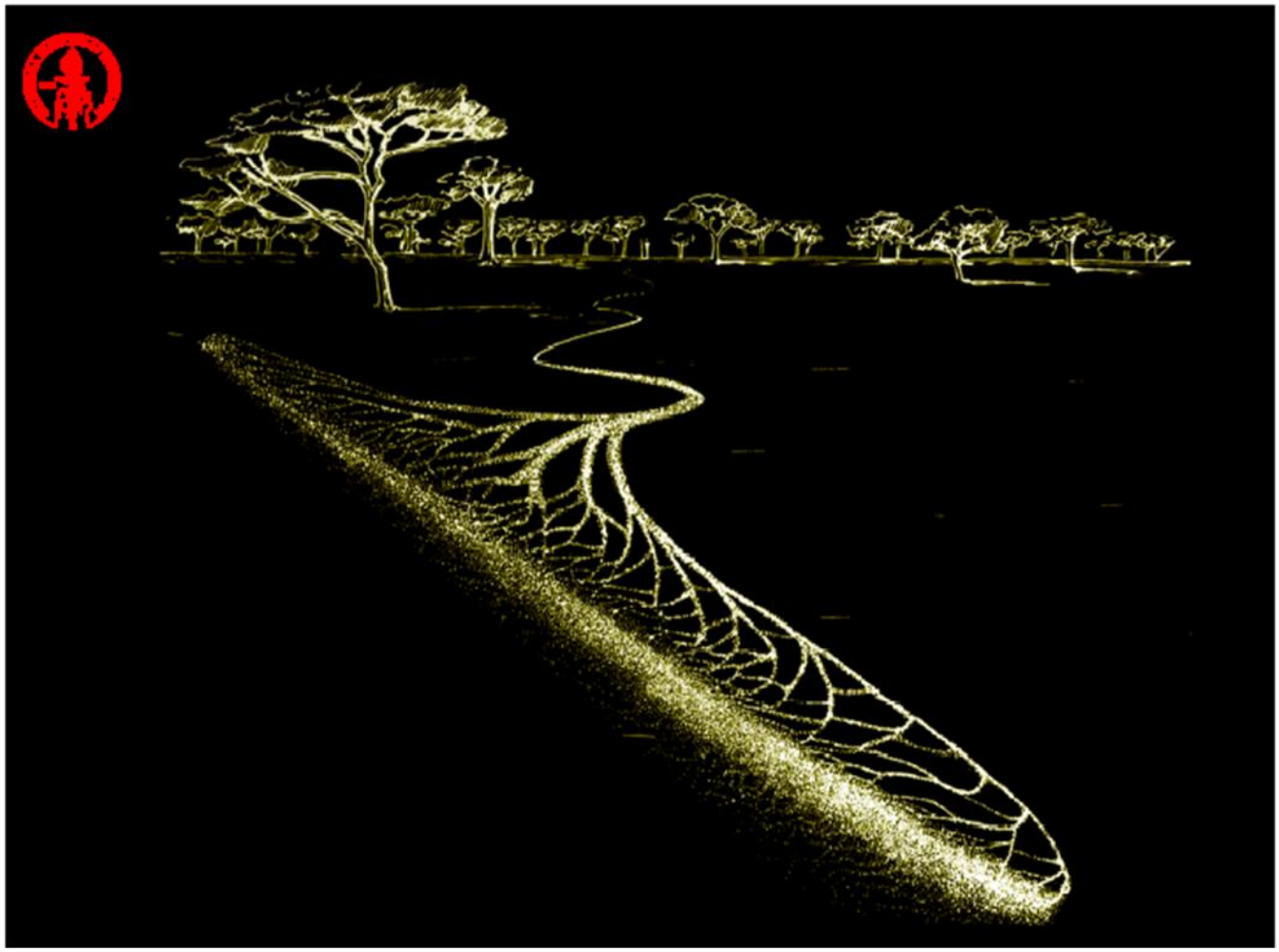
Powerful … but simple

- Simple rules for each individual
- No central control
  - Decentralized and hence robust
- Emergent
  - Performs complex functions
- Self-organization

NTNU

NTNU

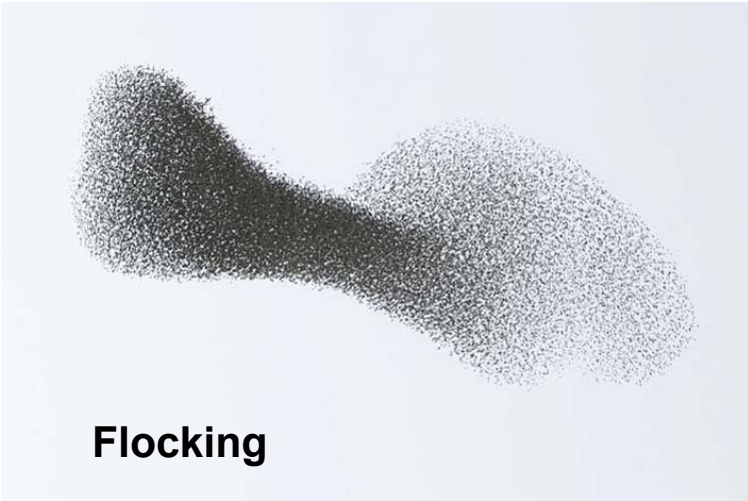




NTNU

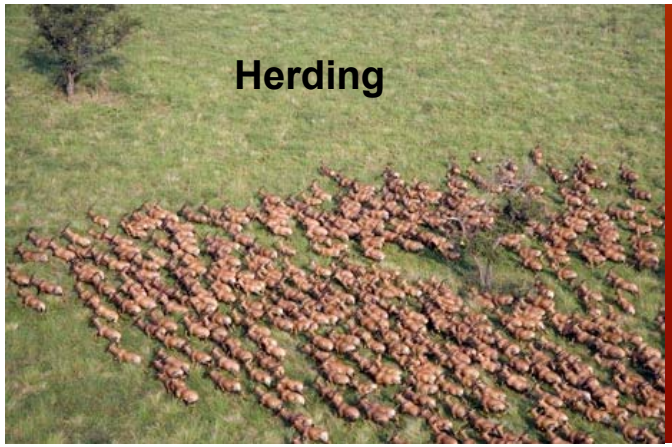
Emergent Collective Behavior

- Some animal societies display **coordinated and purposeful navigation** of several individuals (from tens to thousands).
- Each individual uses **only local information** about the presence of other individuals and of the environment.
- There is no predefined group leader.



Emergent Collective Behavior

In some cases there is a leader and more restrictive rules on relative motion, but individuals still use local information to decide how to move.



Emergent Collective Behavior

- individual swarm (ex: ant)
  - stereotyped,
  - unreliable,
  - unintelligent,
  - simple.

- Collective swarm (ex: ant colony)
  - flexible,
  - reliable,
  - intelligent,
  - complex level performance

NTNU

Coordinated Navigation of Swarms

NTNU

Swarming - Characteristics

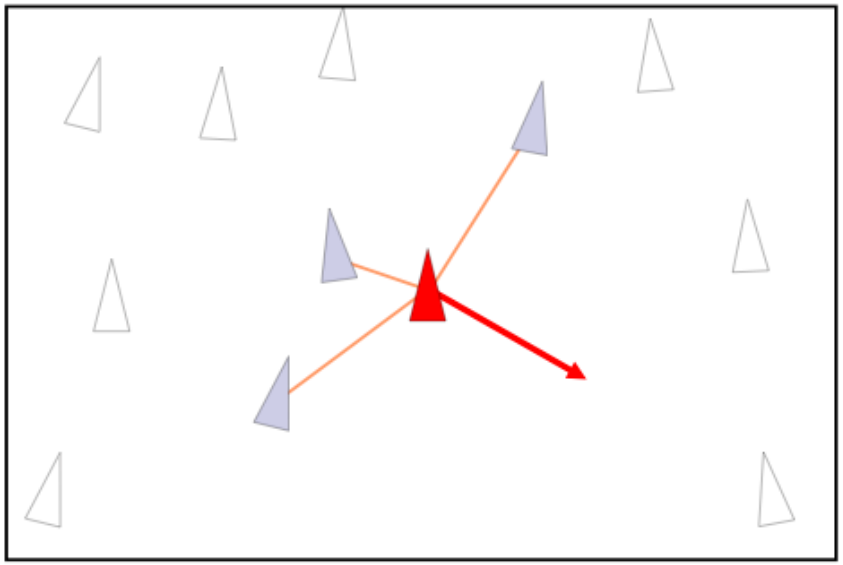
- No central control or data source (**distributed and hence robust**).
- Perception of environment (sensing).
- Ability to change environment.
- Simple creatures following simple rules.
  - **Each one acting on local information.**

Swarming

- Only three simple rules (Example: Birds flocking)

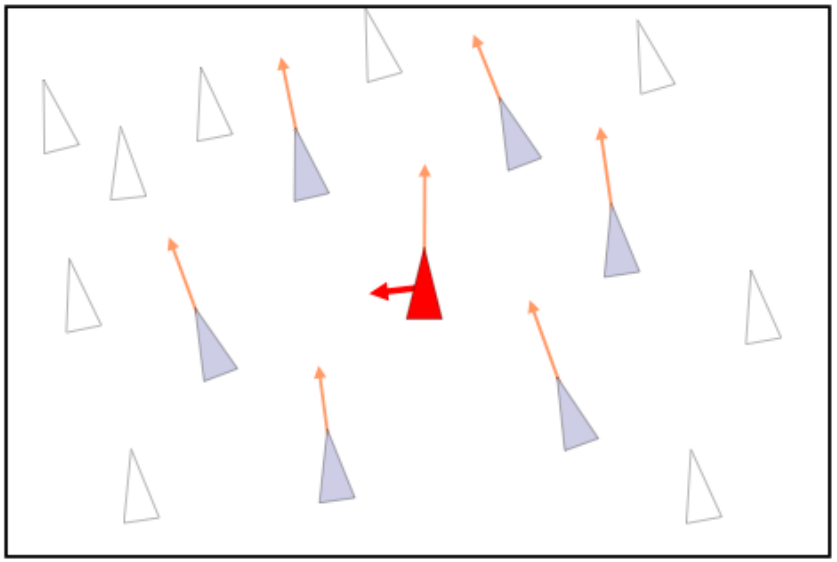
Collision Avoidance

- Rule 1: Avoid Collision with neighboring birds



Velocity Matching

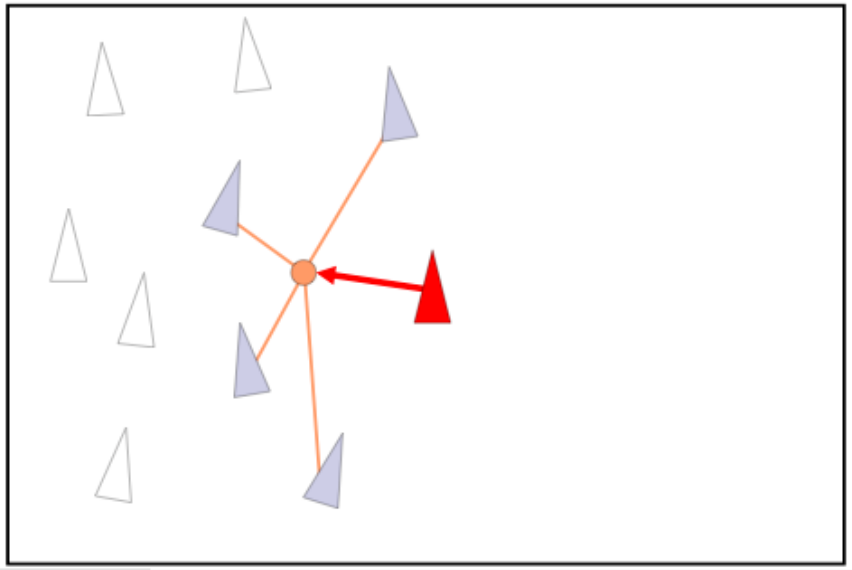
- Rule 2: Match the velocity of neighboring birds



NTNU

Flock Centering

- Rule 3: Stay near neighboring birds



Some SI Application

- U.S. military is investigating SI for controlling unmanned vehicles.

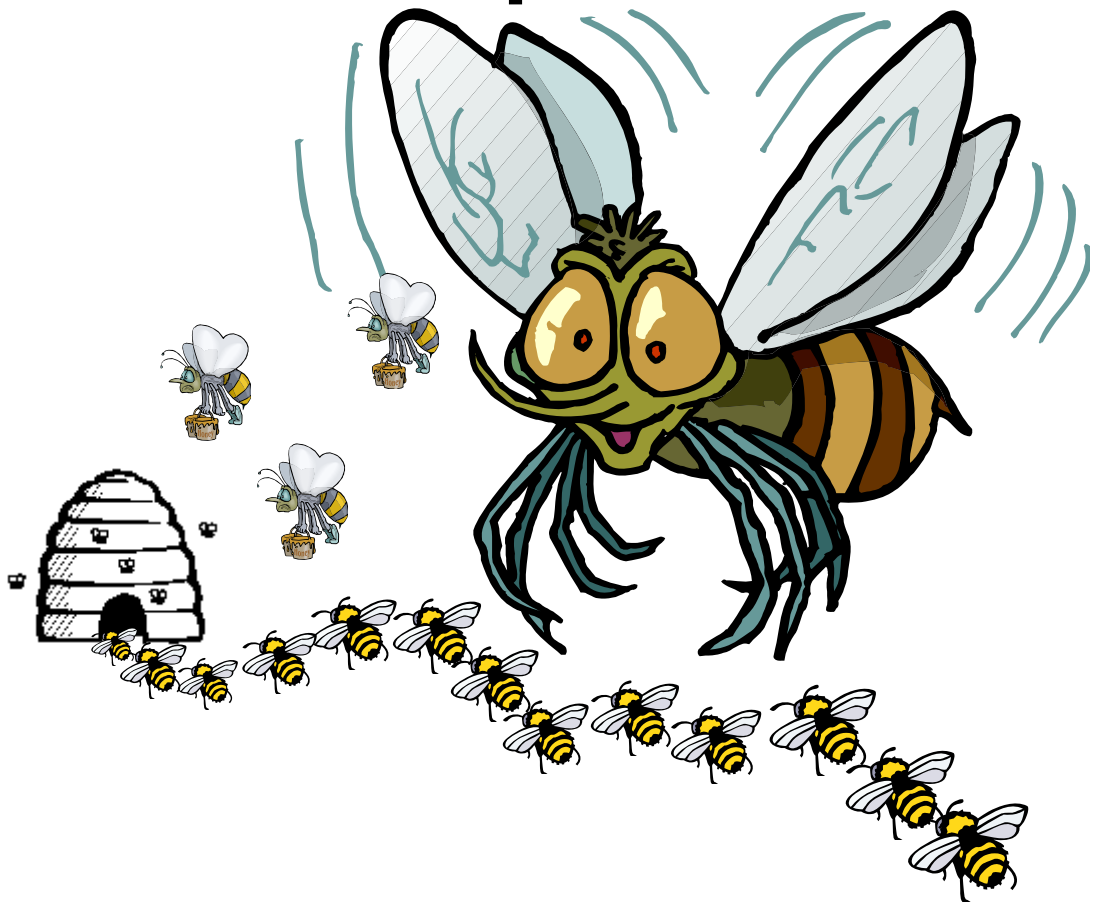
- NASA is investigating the use of SI for planetary mapping.

Some SI Application

- Not only science/engineering, but also……

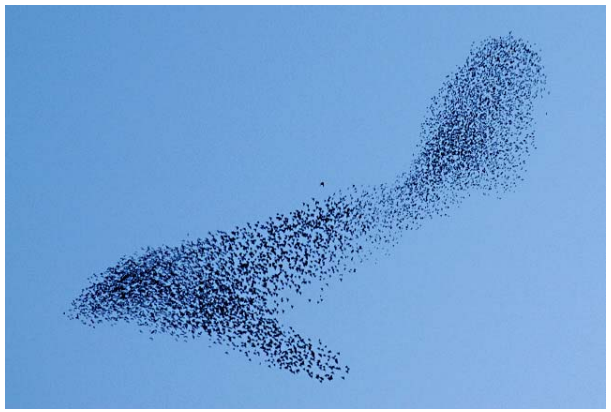
NTNU

Particle Swarm Optimization



PSO

- Russ Eberhart (engineering Prof) and James Kennedy (social scientist) in 1995.
- Inspired by social behavior of bird flocking or fish schooling.
- PSO applies the concept of social interaction to problem solving.
- Individuals interact with one another while learning from their own experience, and gradually move towards the goal.
- Finds a global optimum.



NTNU

Motivation

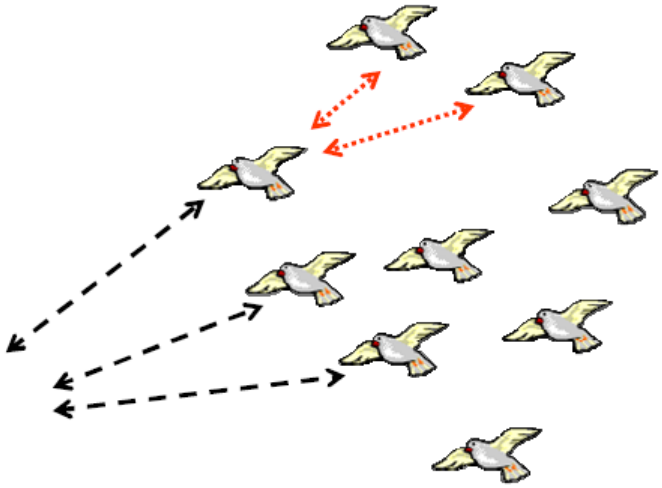
- Inspired by simulation social behavior.
- Related to bird flocking, fish schooling and swarming theory.
    - steer toward the center.
    - match neighbors' velocity.
    - avoid collisions.
- It combines local search methods with global search methods, attempting to balance exploration and exploitation.
- It is easily implemented and has proven both very effective and quick when applied to a diverse set of optimization problems.

Real-Life Example

Concept

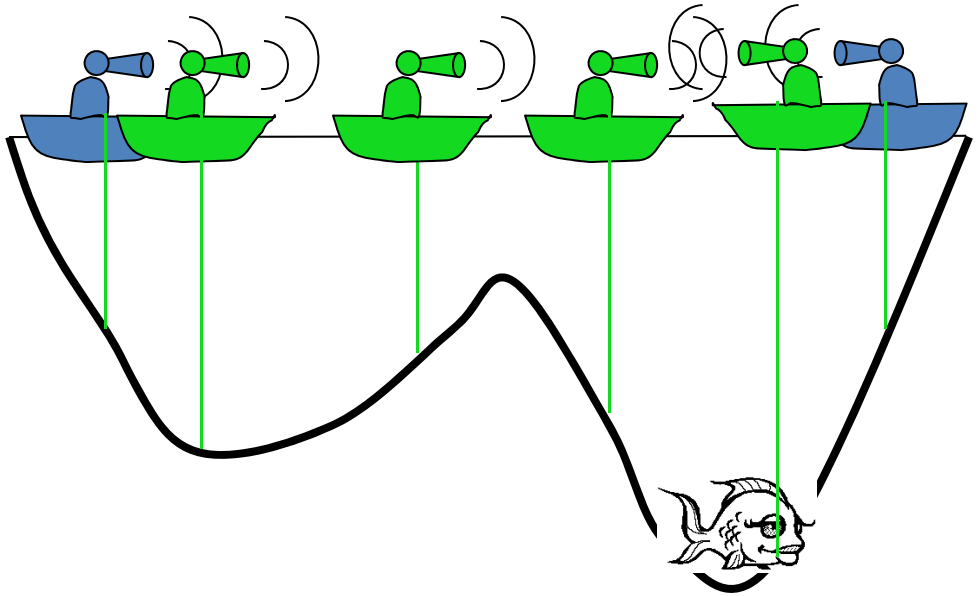
- Particles ***fly*** around in a multidimensional search space.
- During flight, each particle adjusts its position according to ***its own experience***, and according to the ***experience of a neighboring particle***,
    - making use of the best position encountered by itself and its neighbor.

NTNU



- Bird flocking is one of the best example of PSO in nature.

Cooperation example



Cooperation example

- Of course, this example is a caricatural one, but it presents the main features of a particle in basic PSO:
  - A position,
  - A velocity (or, more precisely an operator which can be applied to a position in order to modify it),
  - The ability to exchange information with its neighbors,
  - The ability to memorize a previous position, and
  - The ability to use information to make a decision.

- Let's now see more precisely these points.

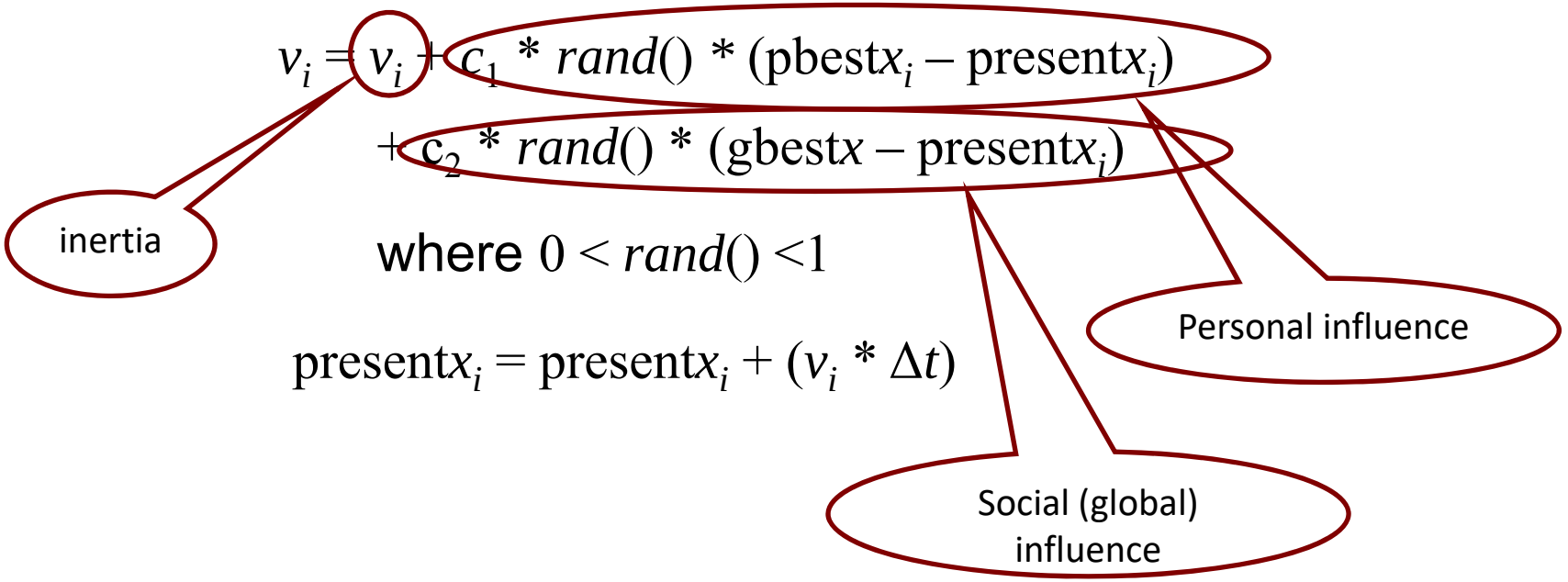
NTNU

The Basic Idea I

- Each particle (or agent) is searching for the optimum.
- Each particle evaluates the function to maximize at each point it visits in spaces.
- Each particle remembers the best value of the function found so far by it (*pbest*) and the position it was (its co-ordinates).
- Secondly, each agent know the globally best position that one member of the flock had found, and its value (*gbest*).
- Each particle is *moving* and hence has a *velocity*.

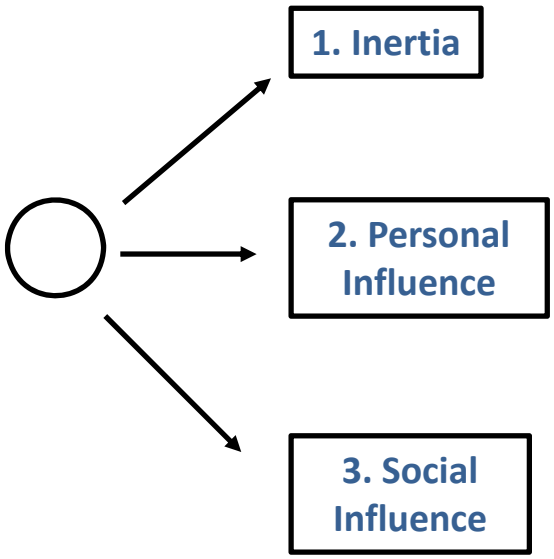
Algorithm – Phase 1 (1D)

- Using the co-ordinates of *pbest* and *gbest*, each agent calculates its new velocity as:



Velocity

$$v_i = v_i + c_1 * rand() * (\text{pbest}x_i - \text{present}x_i) + c_2 * rand() * (\text{gbest}x - \text{present}x_i)$$



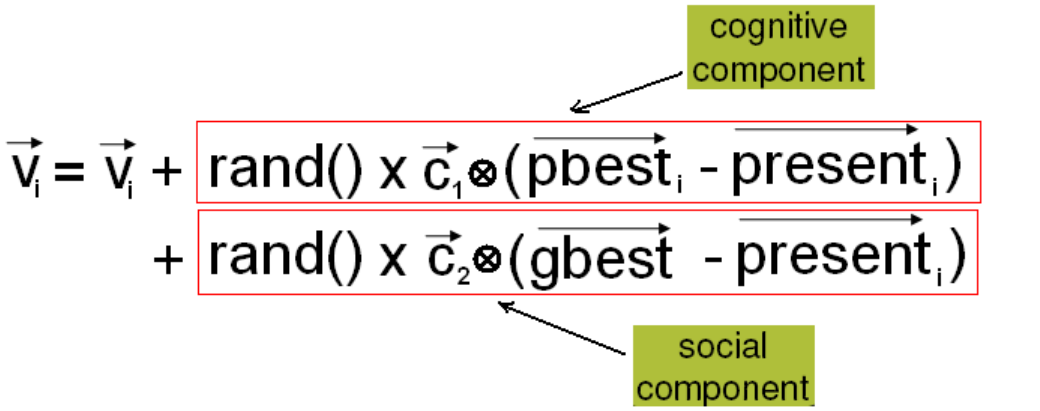
- Makes the particle move in the same direction and with the same velocity

- Improves the individual
- Makes the particle return to a previous position, better than the current
- Conservative

- Makes the particle follow the best neighbors direction

Algorithm – Phase 2 (*n*-dimensions)

- In *n*-dimensional space :



Note that the symbol $\otimes$ denotes a point-wise vector multiplication.

The basic idea II

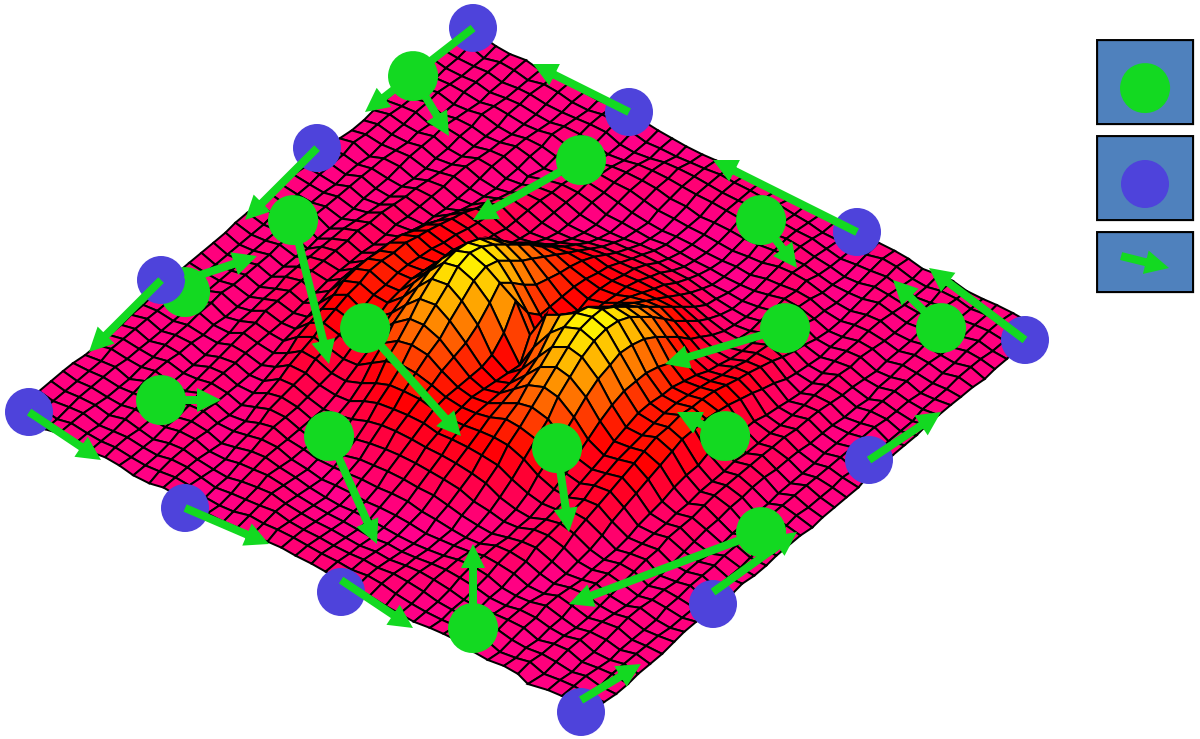
- The particles in the swarm *co-operate*.
  - They exchange information about what they've discovered in the places they have visited.
- The co-operation is very simple. In basic PSO it is like this:
  - A particle has a *neighborhood* associated with it.
  - A particle knows the *fitnesses* of those in its neighborhood, and uses the *position* of the one with *global best fitness*.
  - This position is simply used to adjust the particle's velocity.

Algorithm

```
Randomly generate an initial population
repeat
    for i = 1 to population_size do
        if f(present_i) < f(pbest)
            then pbest = present_i;
        gbest = best(pbest);
        for d =1 to dimensions do
            velocity_update();
            position_update();
        end
    end
until termination criterion is met.
```

NTNU

Initialization: Positions and Velocities



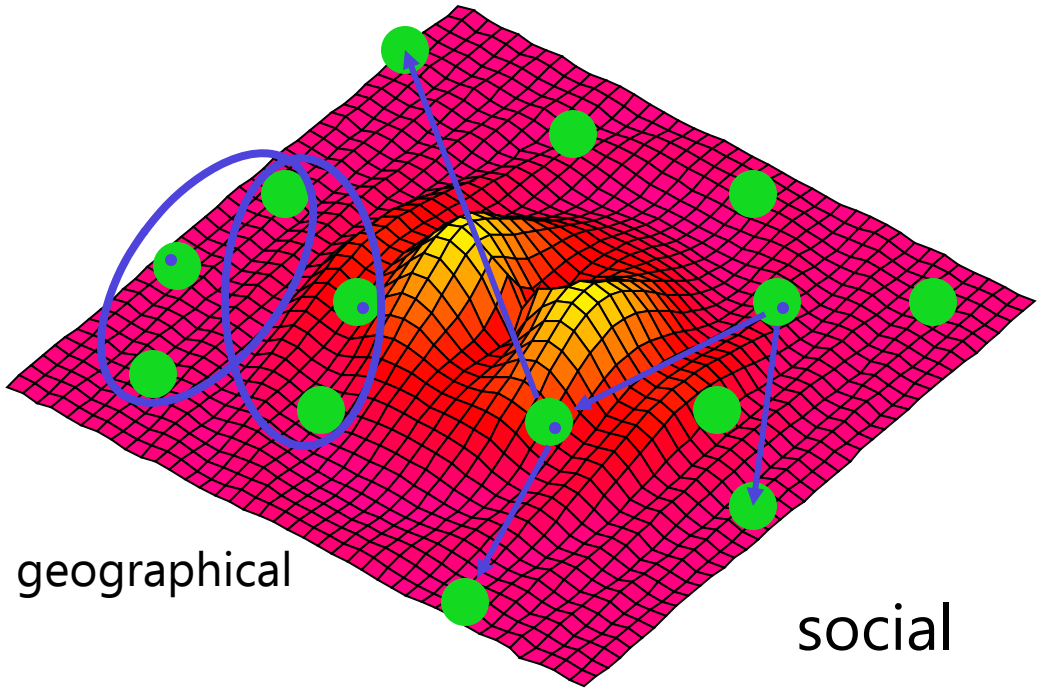
What a Particle Does

- In each timestep, a particle has to move to a new position.
- It does this by adjusting its *velocity*. The adjustment is essentially this:
  - The current velocity PLUS
  - A weighted random portion in the direction of its personal best PLUS
  - A weighted random portion in the direction of the neighborhood best.

$$v_i = v_i + c_1 * rand() * (\text{pbest}x_i - \text{present}x_i) + c_2 * rand() * (\text{gbest}x - \text{present}x_i)$$
$$\text{where } 0 < rand() < 1$$
$$\text{present}x_i = \text{present}x_i + (v_i * \Delta t)$$

- *Having worked out a new velocity, its position is simply its old position plus the new velocity.*

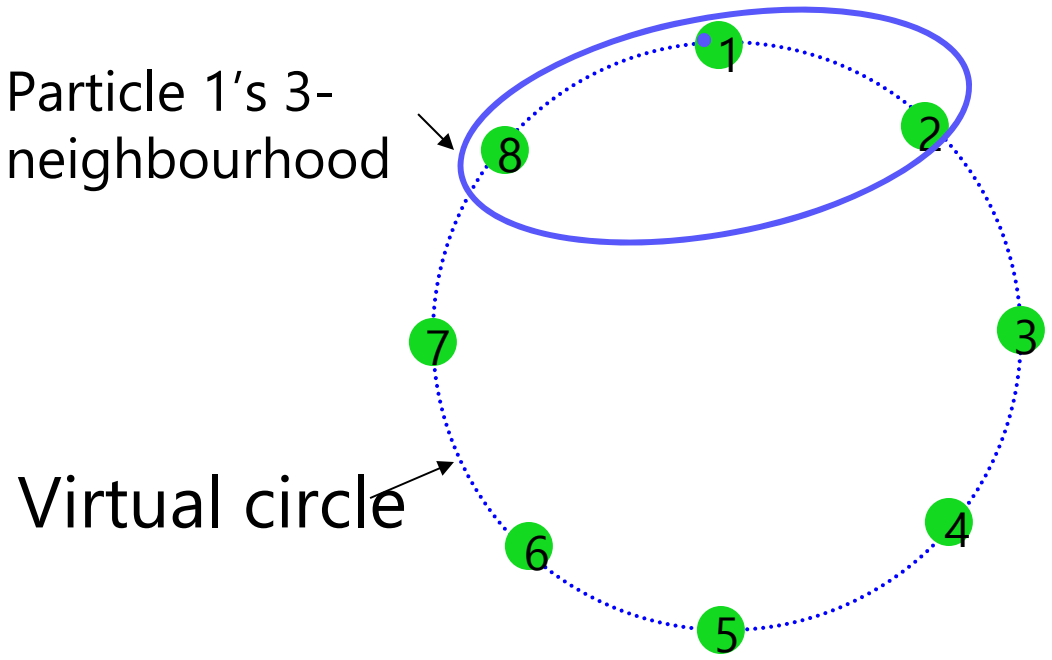
Neighbourhoods



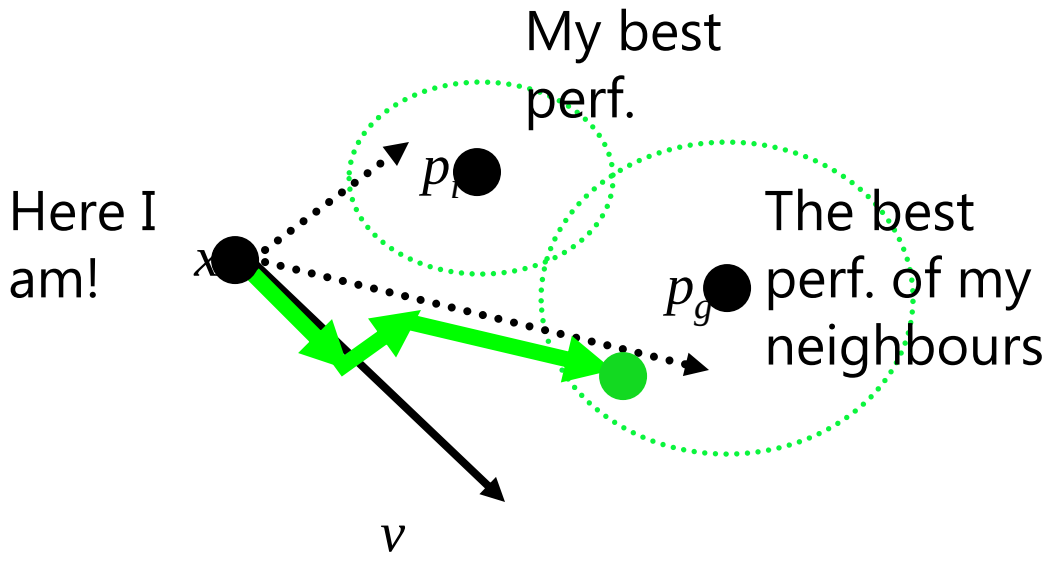
Neighbourhoods: *Size*

- Now, the size of the neighborhood could be a problem.
- Fortunately, PSO is not very sensitive to this parameter and most of users just take a value of 3 or 5 with good results.
- Unlike for the swarm size, there is no mathematical formula,
  - but like for the swarm size, there are some adaptive variants.

The circular neighbourhood



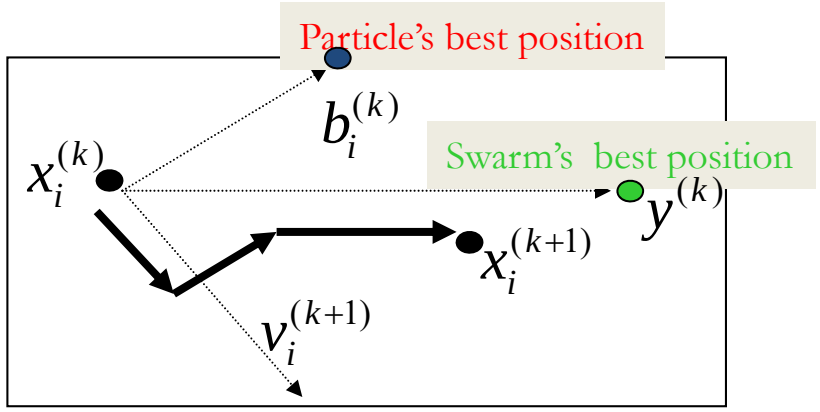
Particles Adjust their positions according to a ``Psychosocial compromise" between *what an individual is comfortable with*, and what society reckons



PSO - *Sheeplike, Going More or Less Towards Your Best Neighbour*

$$v_i^{(k+1)} = v_i^{(k)} + c_1 * (b_i^{(k)} - x_i^{(k)}) + c_2 * (y^{(k)} - x_i^{(k)})$$

$$x_i^{(k+1)} = x_i^{(k)} + v_i^{(k+1)}$$



Where:

$x_i^{(k)}$ is the position of the i^{th} particle at step k

$v_i^{(k+1)}$ is its velocity

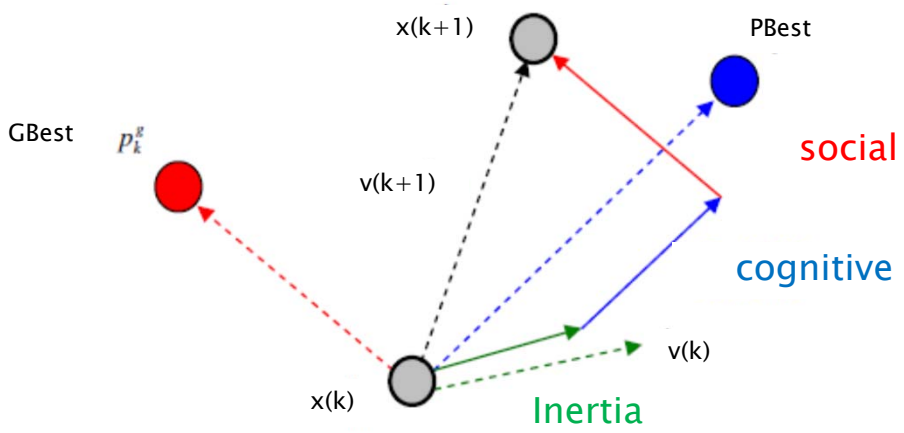
$b_i^{(k)}$ is the best position visited by the i^{th} particle

$y^{(k)}$ is the overall best position ever visited

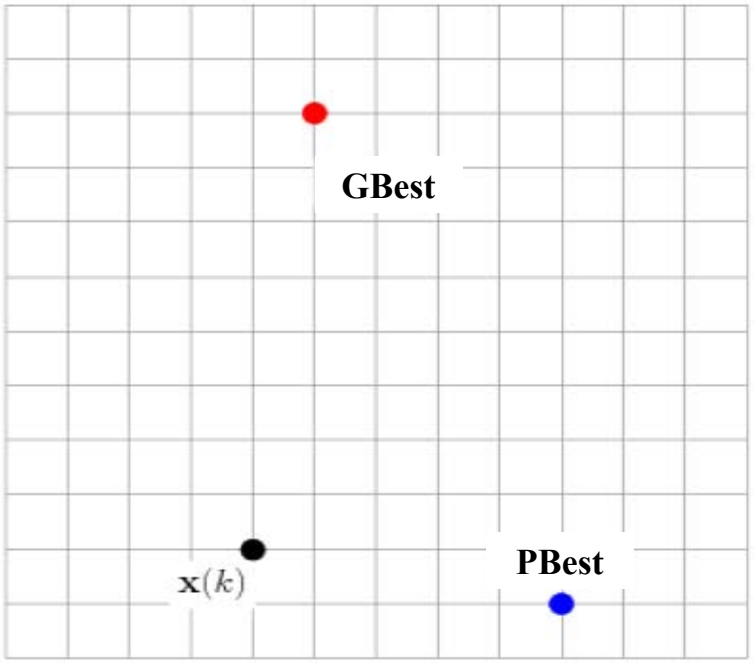
PSO Algorithm

- Particle's velocity

$$\mathbf{v}_i(k+1) = \text{Inertia} + \text{cognitive} + \text{social}$$

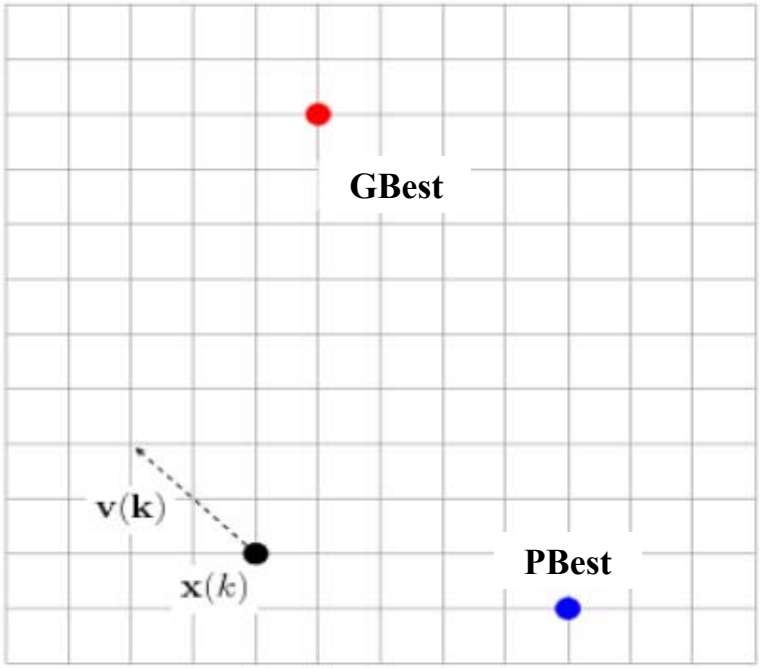


PSO solution update in 2D



- $\mathbf{x}(k)$ - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

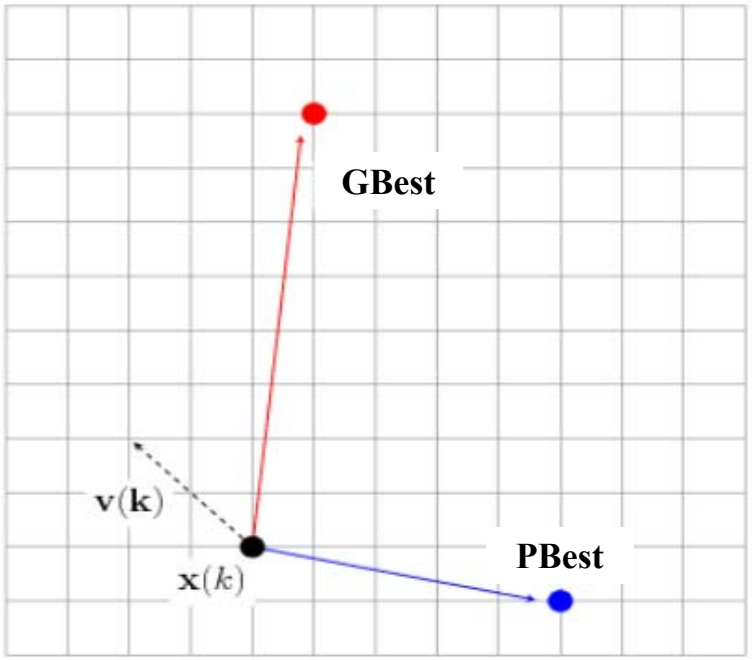
PSO solution update in 2D



Inertia: $\mathbf{v}(k)=(-2, 2)$

- $\mathbf{x}(k)$ - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

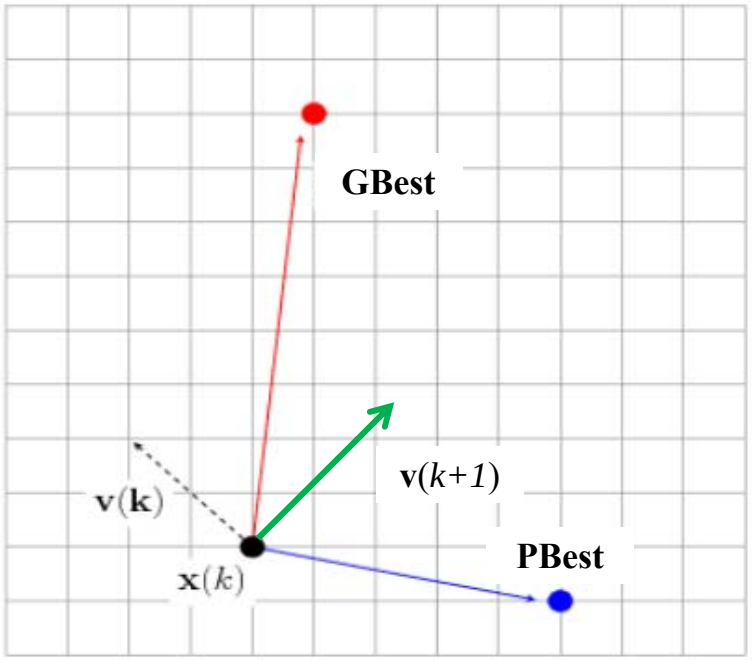
PSO solution update in 2D



- Inertia: $\mathbf{v}(k)=(-2,2)$
- Cognitive:
  PBest-$\mathbf{x}(k)=(9,1)-(4,2)=(5,-1)$
- Social:
  GBest-$\mathbf{x}(k)=(5,10)-(4,2)=(1,8)$

- $\mathbf{x}(k)$ - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

PSO solution update in 2D

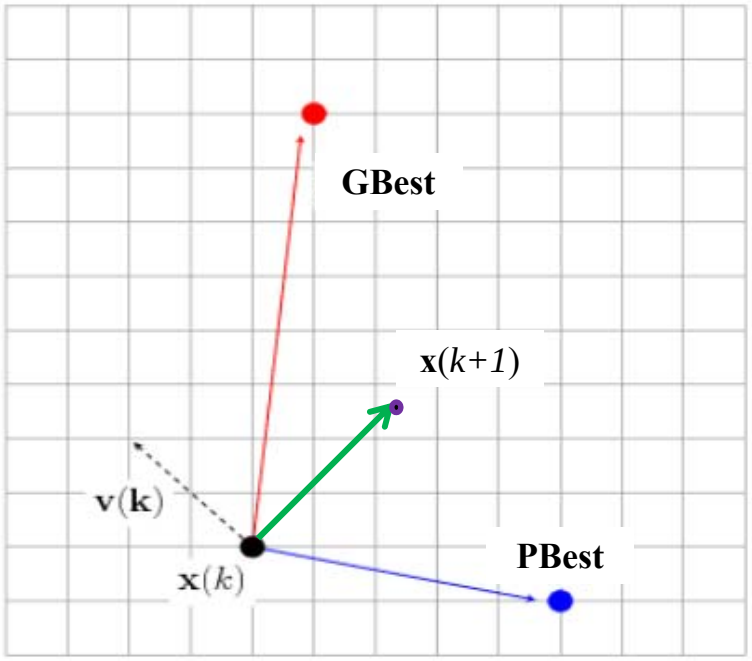


- Inertia: **v**(*k*)=(-2,2)
- Cognitive:
  PBest-**x**(*k*)=(9,1)-(4,2)=(5,-1)
- Social:
  GBest-**x**(*k*)=(5,10)-
  (4,2)=(1,8)

v(*k+1*)=(-2,2)+0.8*(5,-1)
+0.2*(1,8) = (2.2,2.8)

- **x**(*k*) - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

PSO solution update in 2D



- Inertia: $\mathbf{v}(k)=(-2,2)$
- Cognitive:
  PBest-$\mathbf{x}(k)=(9,1)-(4,2)=(5,-1)$
- Social:
  GBest-$\mathbf{x}(k)=(5,10)-(4,2)=(1,8)$
- $\mathbf{v}(k+1)=(2.2,2.8)$

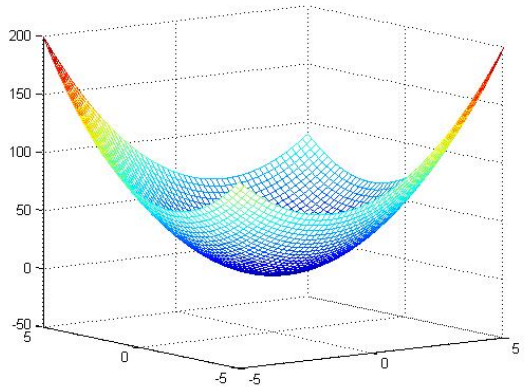
$\mathbf{x}(k+1)=\mathbf{x}(k)+\mathbf{v}(k+1)=(4,2)+(2.2,2.8)=(6.2,4.8)$

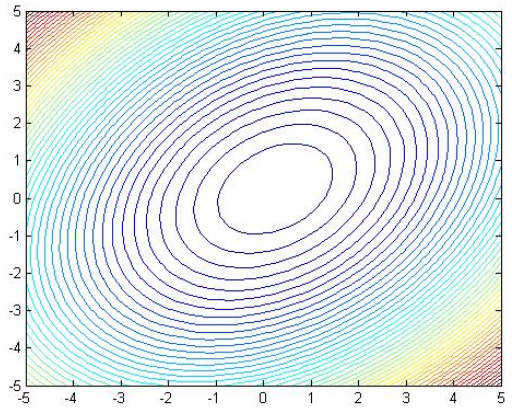
- $\mathbf{x}(k)$ - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

Example

- Find the minimum of this function

$$f(\mathbf{x}) = 3x_1^2 - 2x_1x_2 + 3x_2^2 - x_1 - x_2$$





Example

$$\mathbf{x}_1 = \begin{bmatrix} 2.2824 & 0.6238 & 4.0005 & 3.1717 & -4.0058 \\ -0.4894 & -2.7580 & -2.7043 & -3.3118 & 1.5771 \end{bmatrix}$$

$$\mathbf{v}_1 = \begin{bmatrix} -0.6321 & 0.1712 & 0.6942 & 0.0264 & 0.2207 \\ 0.2133 & -0.5598 & -0.2500 & 0.6079 & 0.3122 \end{bmatrix}$$

$$\mathbf{x}_2 = \begin{bmatrix} 1.7767 & 1.4300 & 2.5656 & 2.2018 & 3.3541 \\ -0.3187 & -2.2903 & -0.3385 & 0.3199 & -0.5338 \end{bmatrix}$$

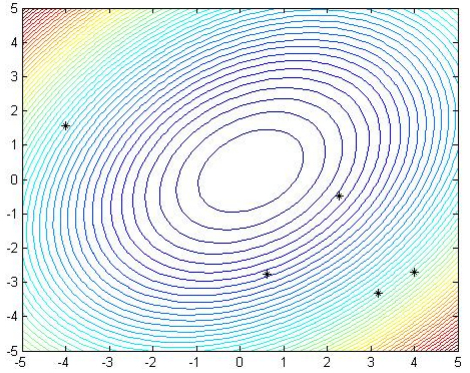
$$\mathbf{v}_2 = \begin{bmatrix} -0.5057 & 0.8063 & -1.4349 & -0.9700 & 7.3599 \\ 0.1706 & 0.4677 & 2.3657 & 3.6317 & -2.1109 \end{bmatrix}$$

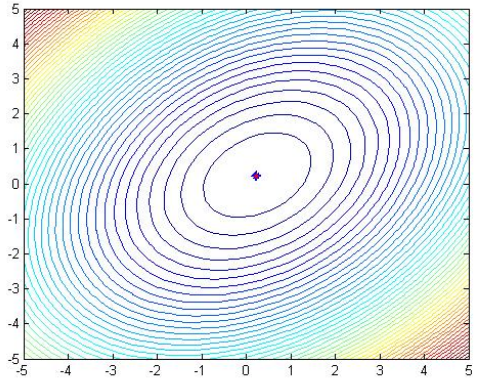
$$\mathbf{x}_3 = \begin{bmatrix} 1.3721 & 2.4464 & 1.0728 & 1.1350 & 7.9656 \\ -0.1822 & 0.1959 & 1.5627 & 2.7884 & -2.0485 \end{bmatrix}$$

$$\mathbf{v}_3 = \begin{bmatrix} -0.4046 & 1.0163 & -1.4928 & -1.0667 & 4.6114 \\ 0.1365 & 2.4862 & 1.9012 & 2.4685 & -1.5146 \end{bmatrix}$$

$$\vdots$$

$$\mathbf{x}_t = \begin{bmatrix} 0.2230 & 0.2197 & 0.2400 & 0.2293 & 0.2167 \\ 0.2056 & 0.2436 & 0.2378 & 0.2156 & 0.2106 \end{bmatrix}$$





$$GBest = \begin{bmatrix} 0.2227 \\ 0.2057 \end{bmatrix} \quad fitness = -0.25$$

Pseudocode

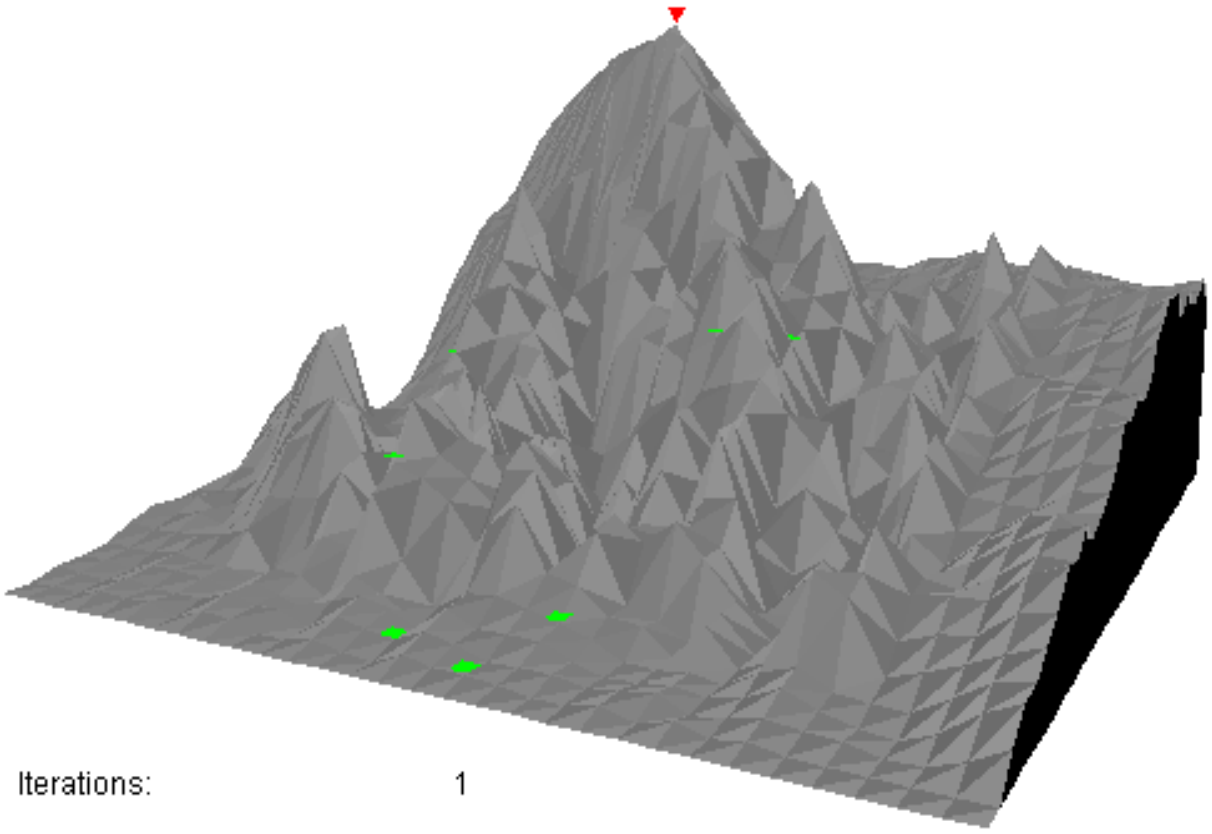
http://www.swarmintelligence.org/tutorials.php

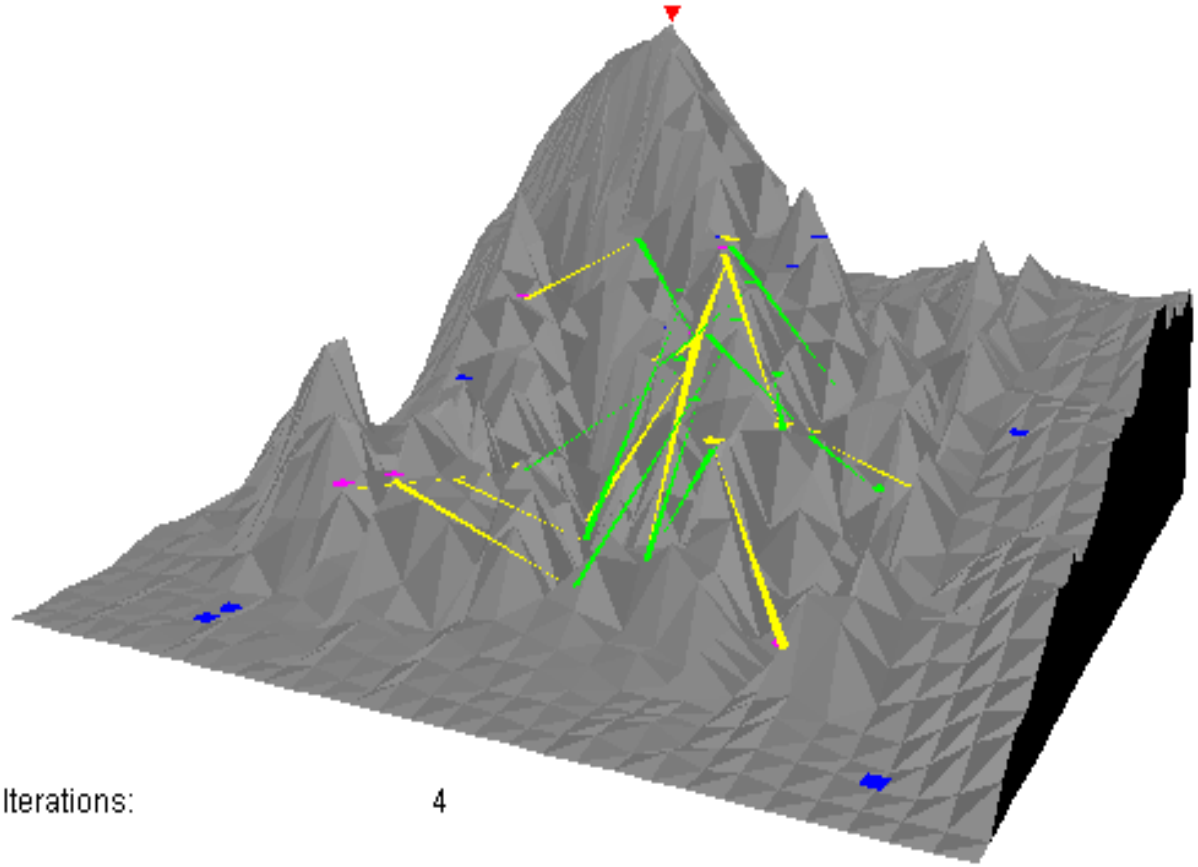
```
For each particle
    Initialize particle
END

Do
    For each particle
        Calculate fitness value
        If the fitness value is better than its peronal best
      set current value as the new pBest
    End

  Choose the particle with the best fitness value of all as gBest

    For each particle
        Calculate particle velocity
        Update particle position
    End

While maximum iterations or minimum error criteria is not attained
```

Pseudocode

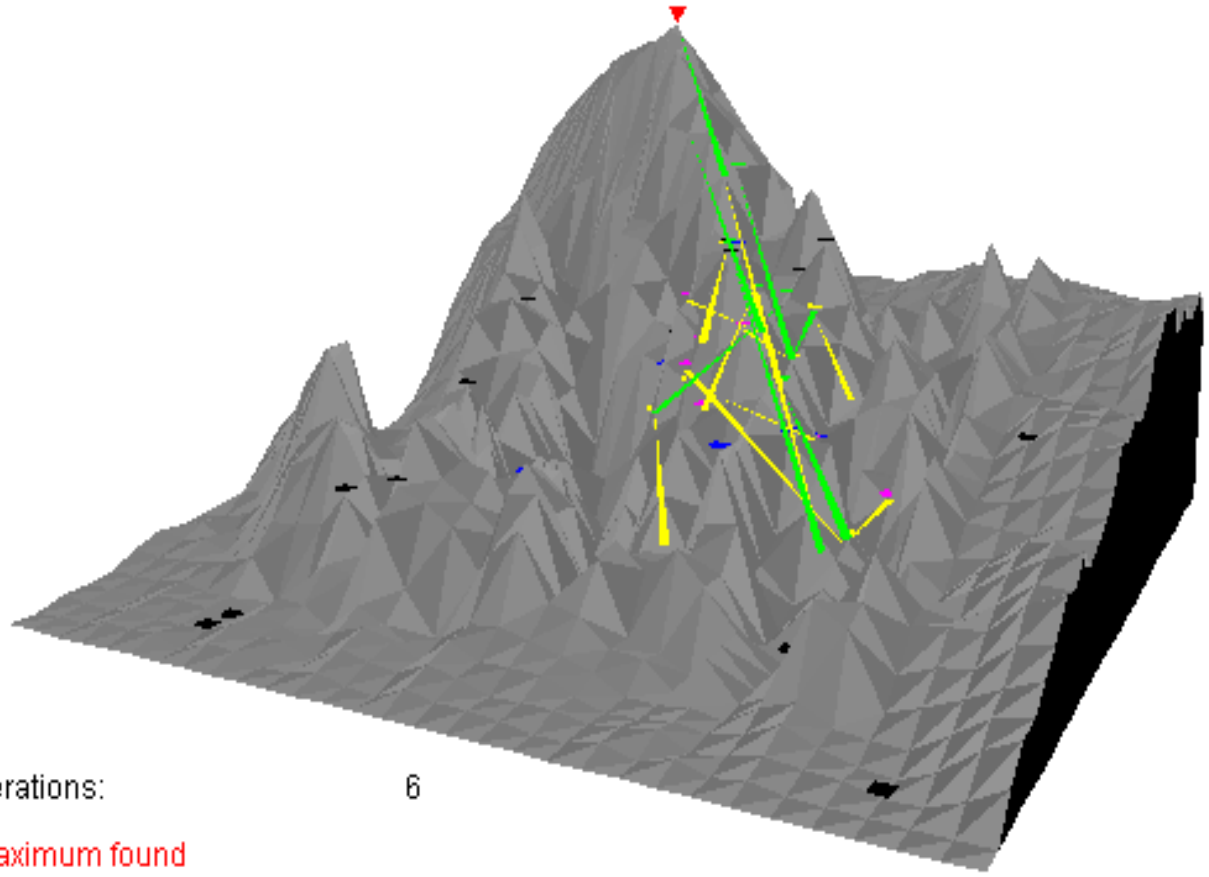
http://www.swarmintelligence.org/tutorials.php

- Particles' velocities on each dimension are clamped to a maximum velocity V_{max} (a parameter specified by the user).
- If the sum of accelerations would cause the velocity on that dimension to exceed V_{max}
  - Then the velocity on that dimension is limited to V_{max}.


Iterations: 1



Iterations: 4



Iterations: 6

Maximum found

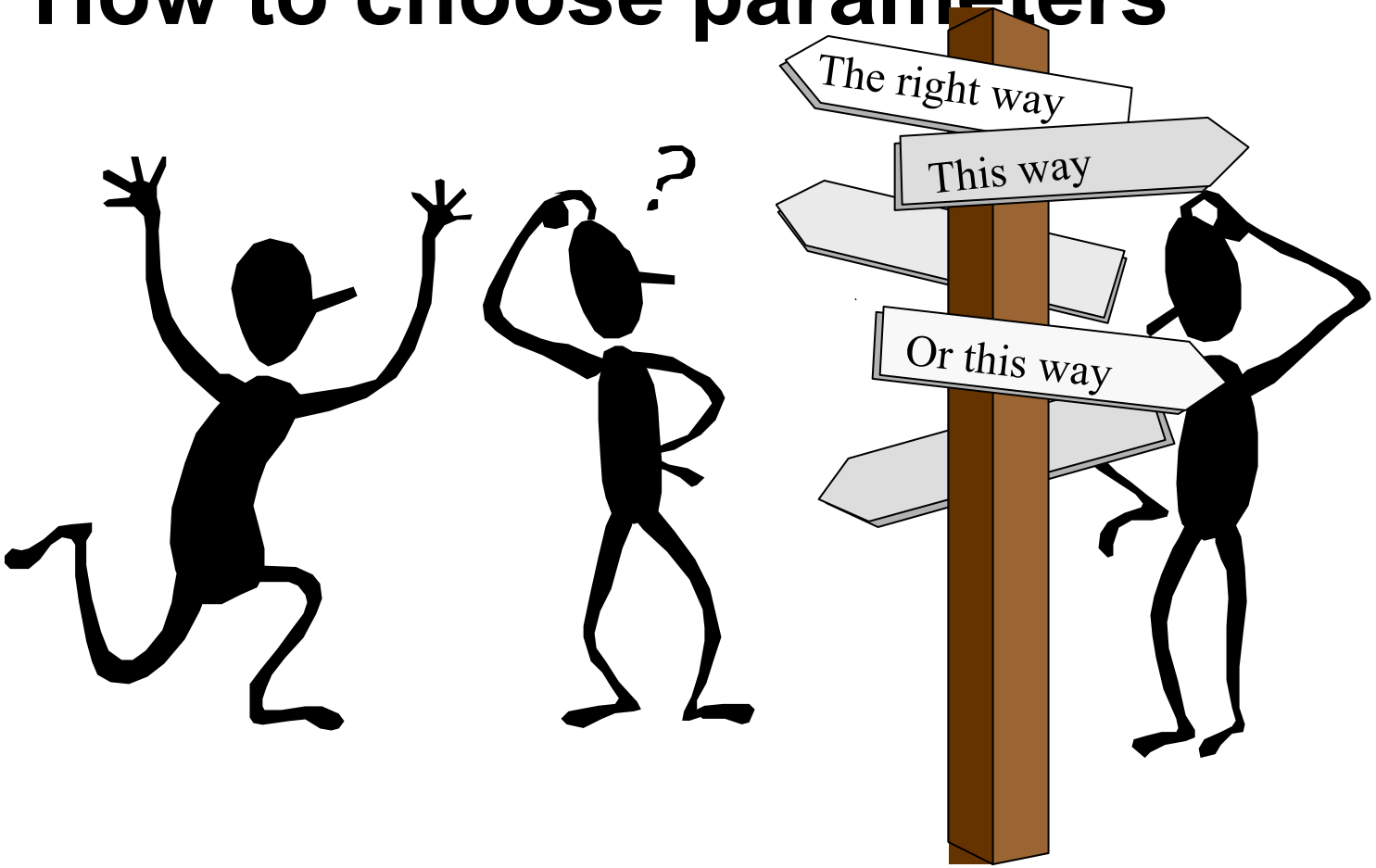
NTNU

Parameters

- Number of particles (swarm-size)
- C_1: (importance of personal best)
- C_2: (importance of neighborhood best)

- V_{max}: limit on velocity

How to choose parameters



Parameters

- Number of particles
  - (10—50) are reported as usually sufficient.
- C_1 (importance of personal best)
- C_2 (importance of neighbourhood best)
  - Usually $C_1 + C_1 = 4$.
  - No good reason other than empiricism

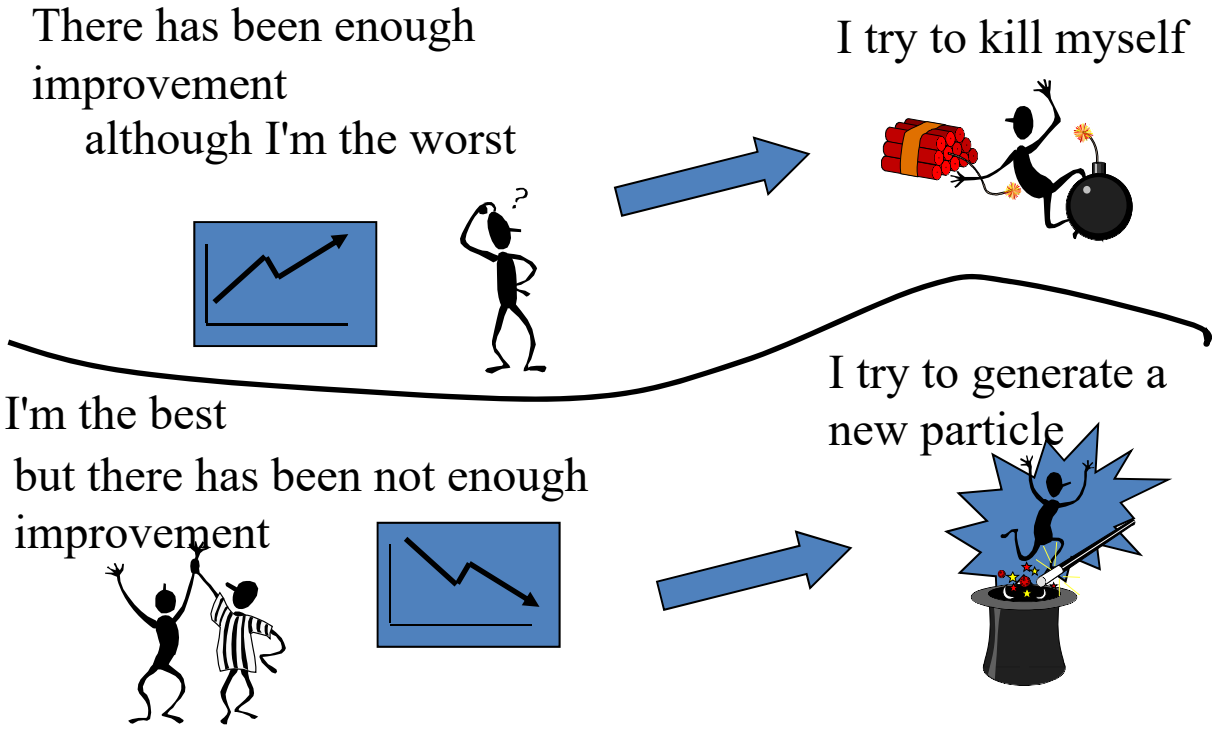
V_{max}

- An important parameter in PSO; typically the only once adjusted.
- Clamps particles velocities on each dimension
- Determines “fineness” with which regions are searched
  - if too high, can fly past optimal solutions
  - if too low, can get stuck in local minima

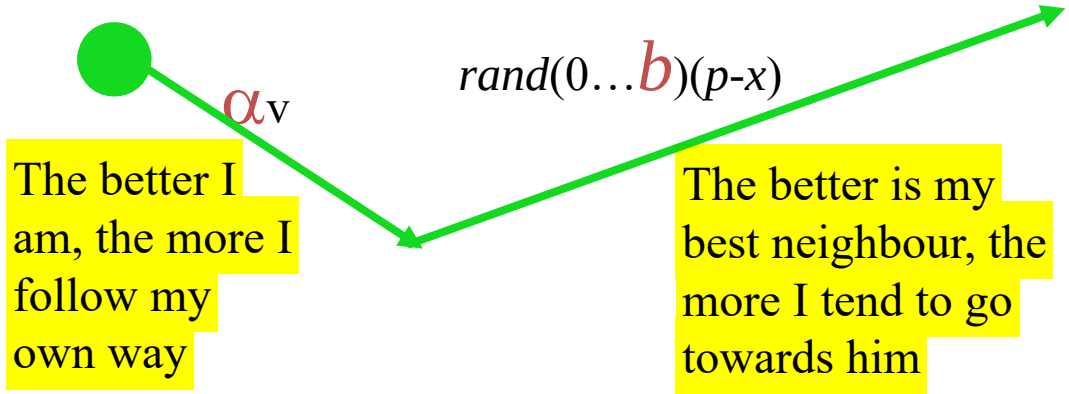
Swarm Size

- There is no selection in PSO
    - All particles survive for the length of the run.
    - PSO is the only EA that does not remove candidate population members.
- In PSO, topology is constant; a neighbor is a neighbor.
- You can perfectly use a swarm size of 20 and a neighbourhood of 3 for a large range of problems with good results.

Adaptive swarm size



Adaptive coefficients



How and when should an excellent algorithm terminate?

How and when should an excellent algorithm terminate?

Like this

PSO Has A memory

- Not "**what**" that best solution was, but "**where**" that best solution was
- **Quality**: population responds to quality factors *pbest* and *gbest.*
- **Diverse** response: responses allocated between *pbest* and *gbest.*
- **Stability**: population changes state only when *gbest* changes.
- **Adaptability**: population does change state when *gbest* changes.

69

PSO - Summary

- The method allows the motion of particles to explore the space of interest.
- Each particle updates its position in discrete unit time steps.
- The velocity is updated by a linear combination of two terms:
  - The first along the direction pointing to the best position discovered by the particle.
  - The second towards the overall best position.

NTNU

Advantages

- PSO have no overlapping and mutation calculation.
- The calculation in PSO is very simple.
- Easily parallelized for concurrent processing.
- Derivative free.
- Very few algorithm parameters.
- Very efficient global search algorithm

NTNU

Disadvantages

- Tendency to a fast and premature convergence in mid optimum points.
- The method can not work out the problems of scattering.
- The method can not work out the problems of non-coordinate system, such as the solution to the energy field and the moving rules of the particles in the energy field.
- Tendency to a fast and premature convergence in mid optimum points.

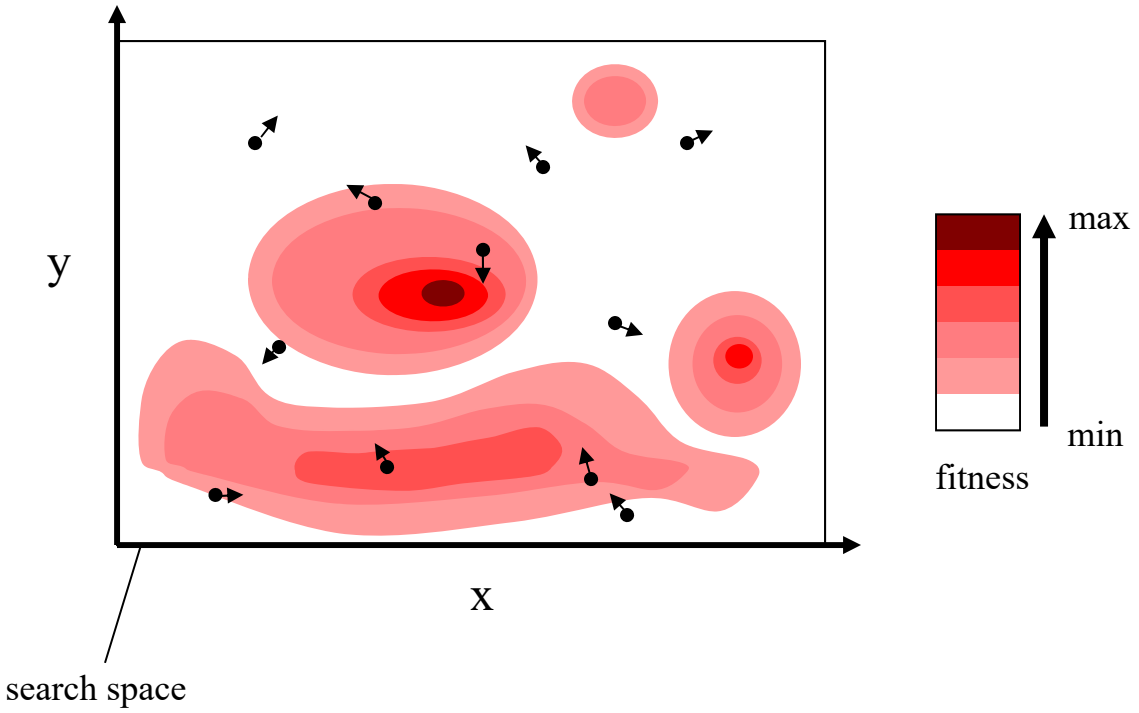
PSO and GA Comparison

- **Commonalities**
  - PSO and GA are both population based stochastic optimization.
  - both algorithms start with a group of a randomly generated population.
  - both have fitness values to evaluate the population.
  - Both update the population and search for the optimium with random techniques.
  - Both systems do not guarantee success.

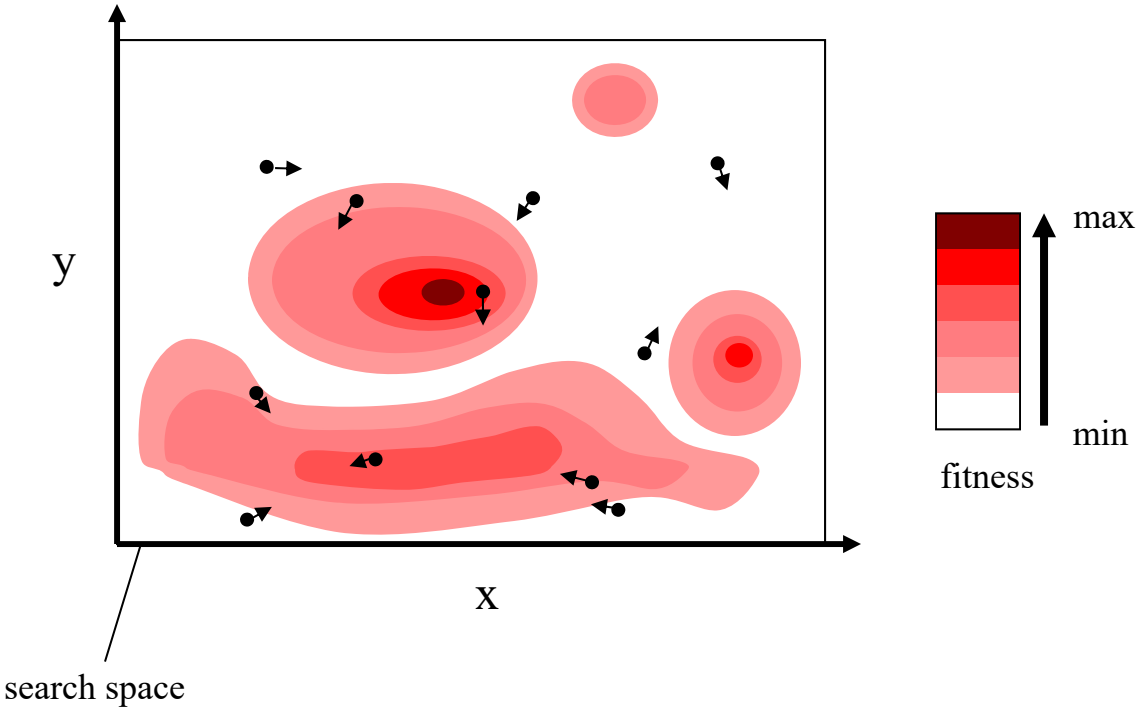
PSO and GA Comparison

- **Differences**
  - PSO does not have genetic operators like crossover and mutation. Particles update themselves with the internal velocity.
  - They also have memory, which is important to the algorithm.
  - Particles do not die.
  - the information sharing mechanism in PSO is significantly different.
    - Info from best to others, GA population moves together

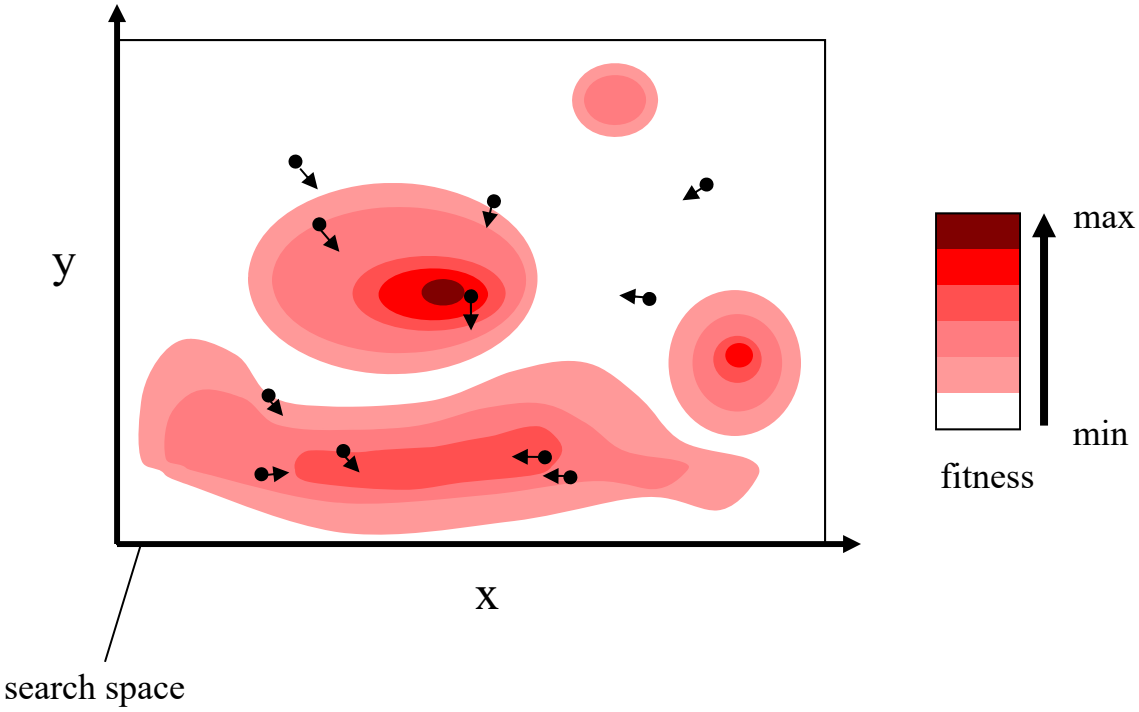
Simulation 1



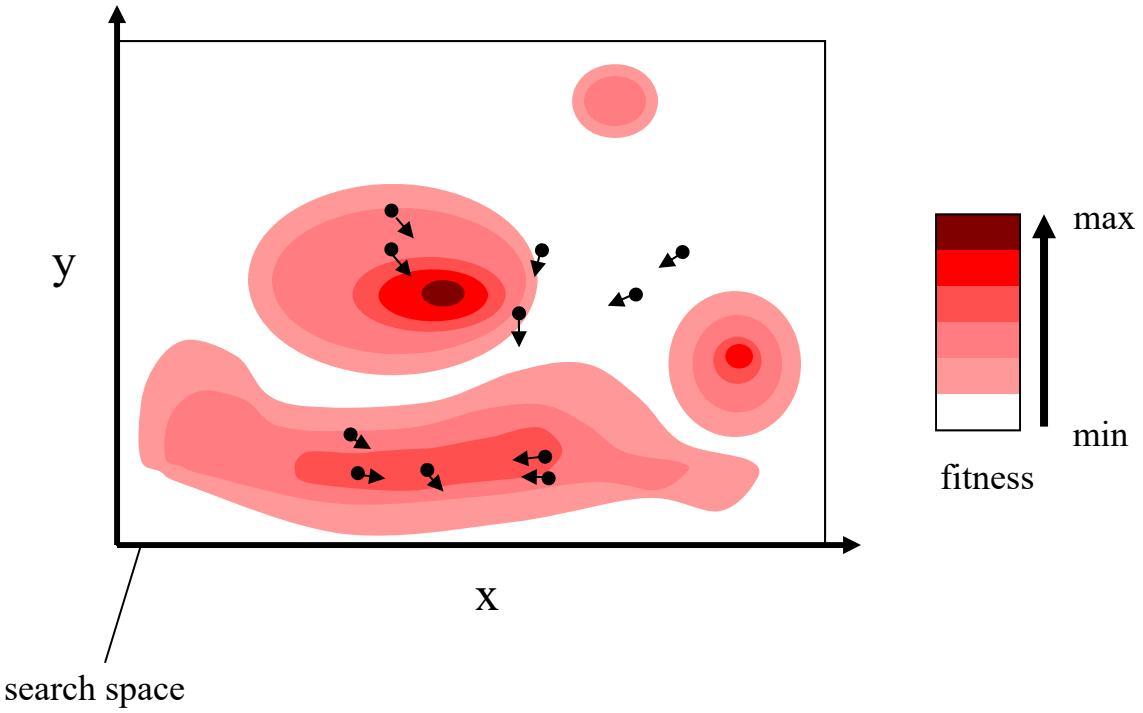
simulation 2



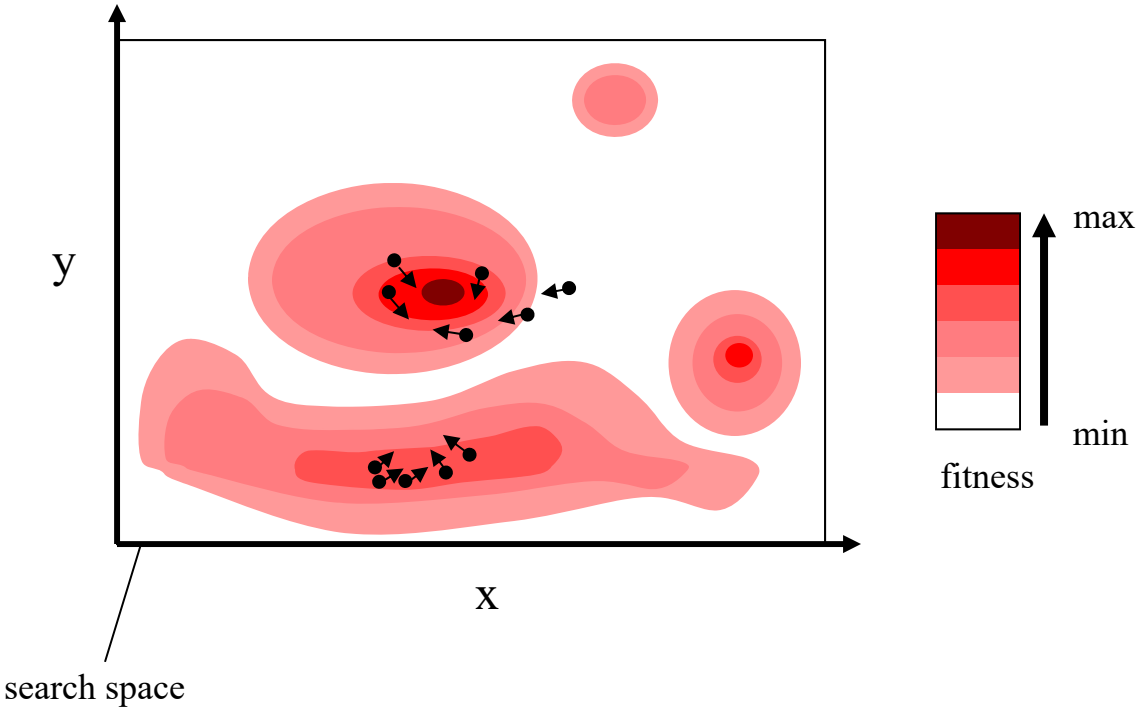
simulation 3



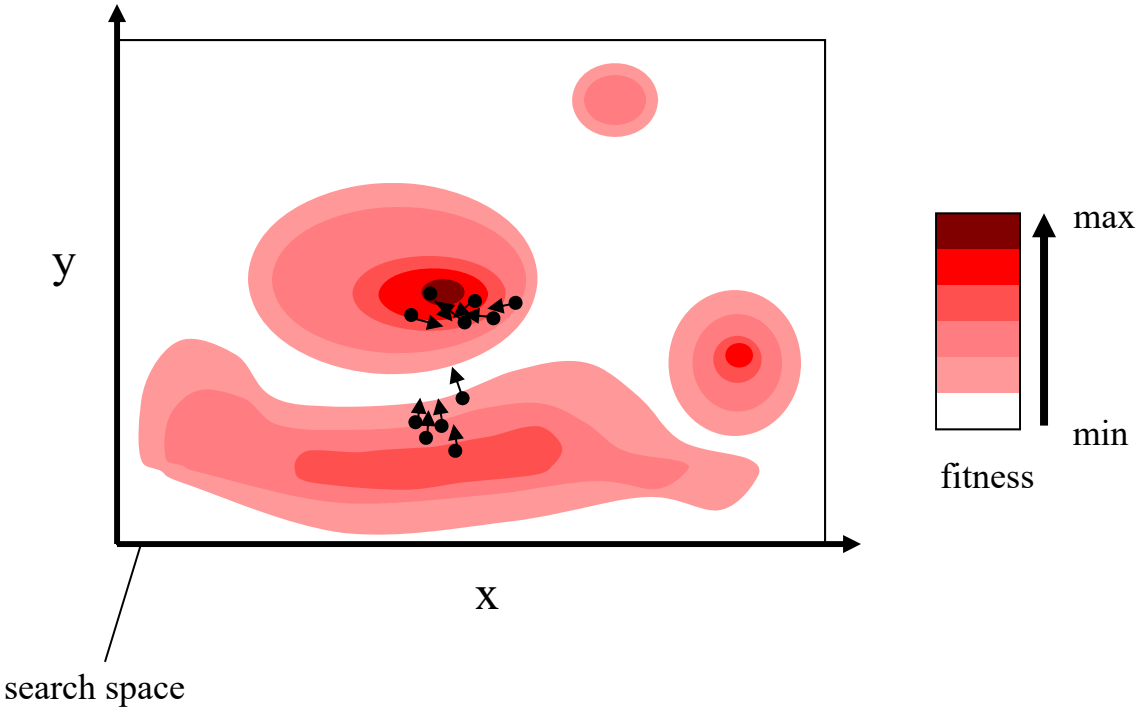
simulation 4



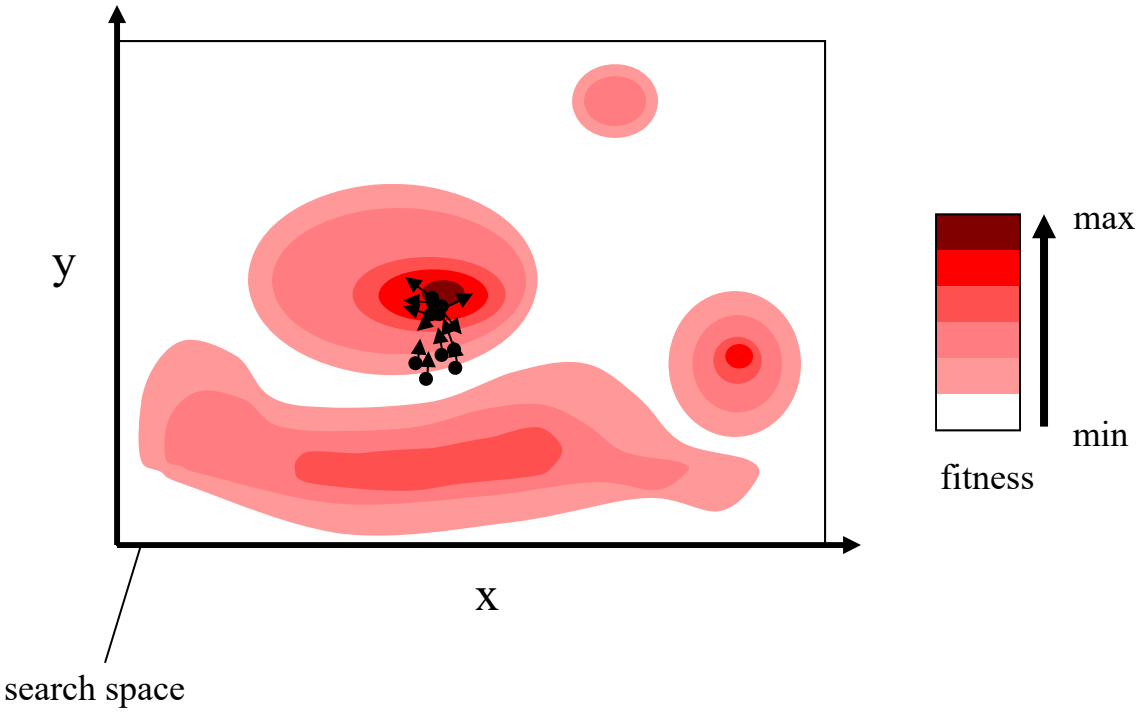
simulation 5



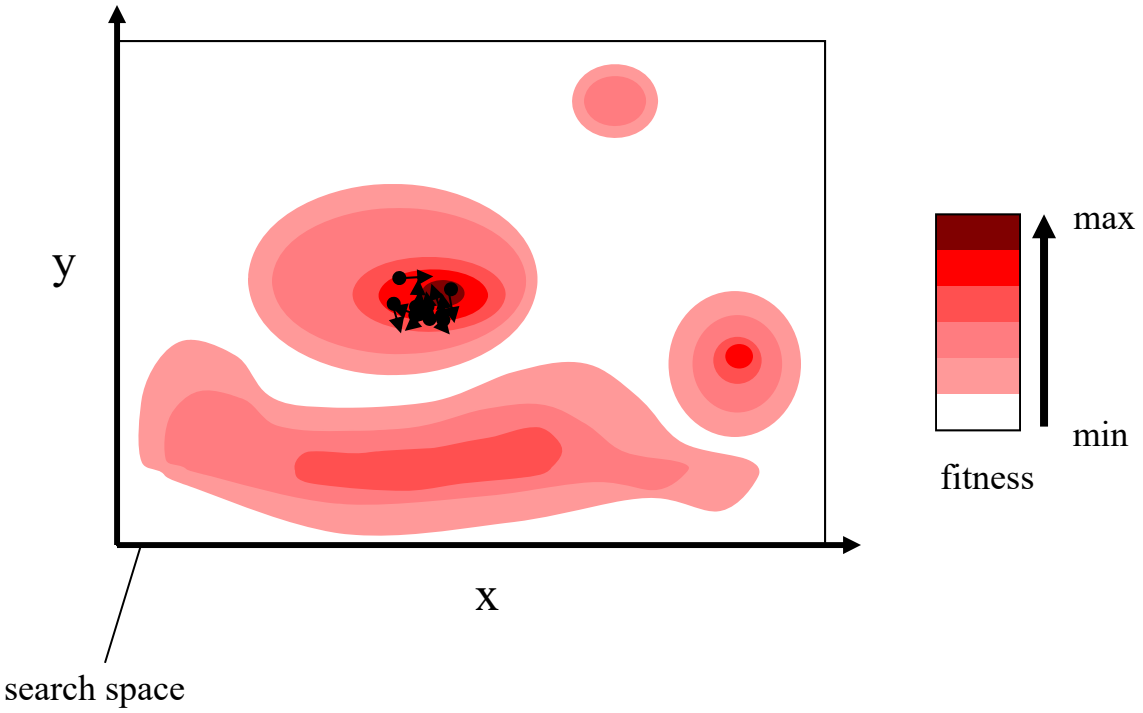
simulation 6



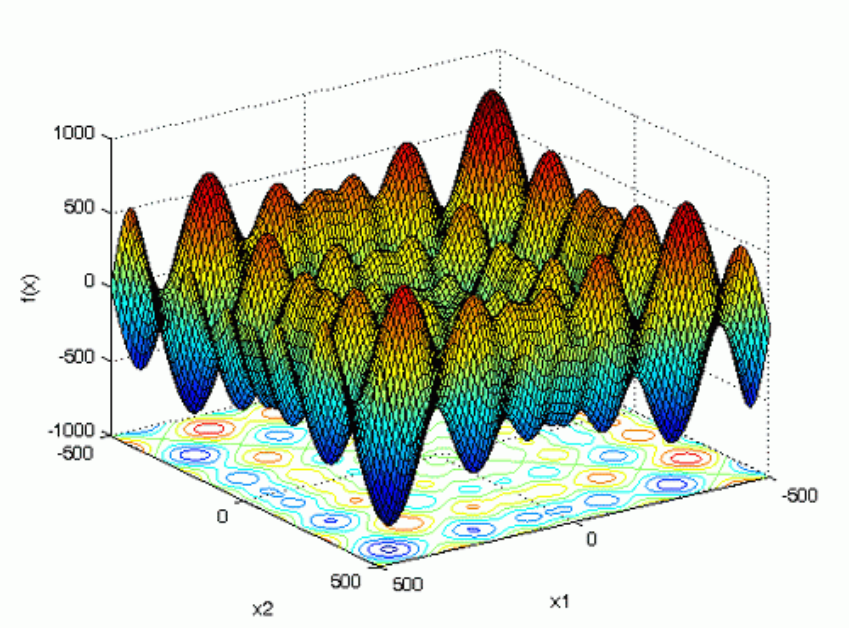
simulation 7



simulation 8



Schwefel's function



$$f(x) = \sum_{i=1}^{n} (-x_i) \cdot \sin(\sqrt{|x_i|})$$

where

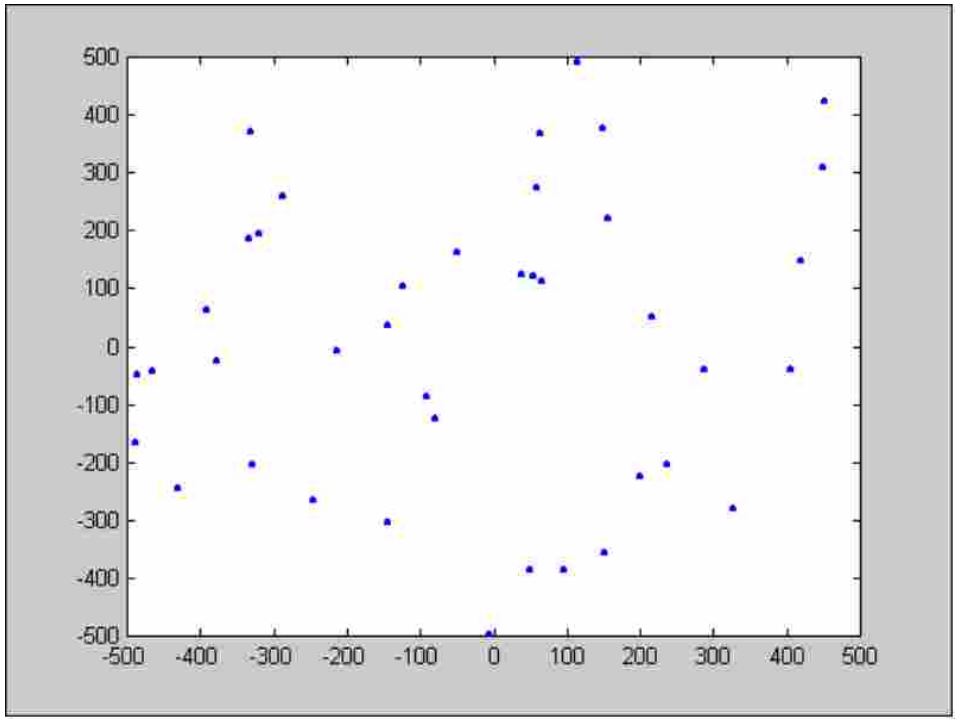
$$-500 \le x_i \le 500$$

global minimum

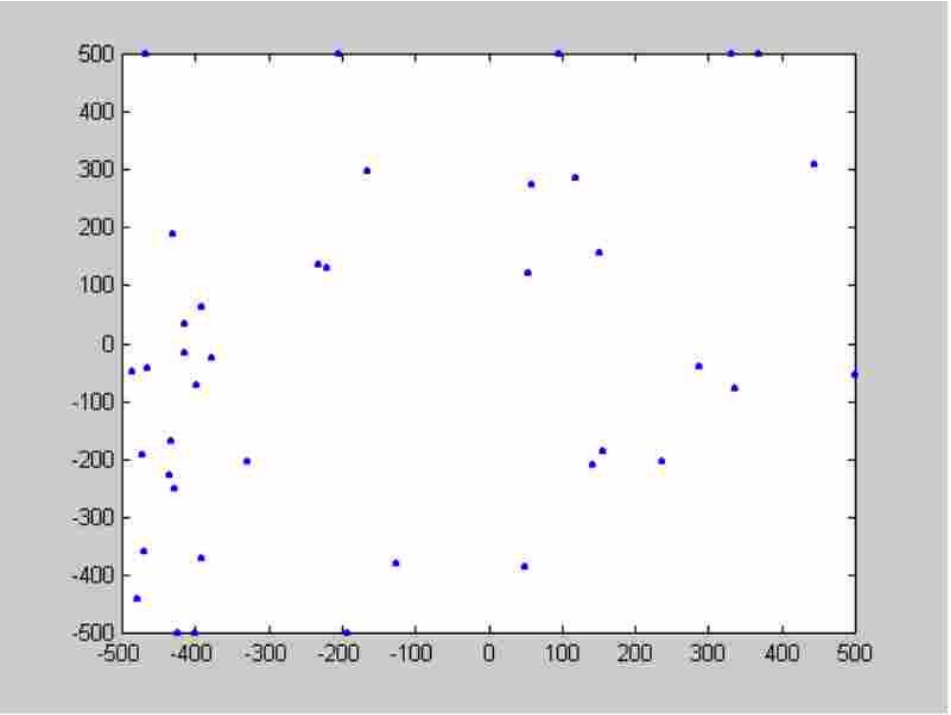
$$f(x) = n \cdot 418.9829;$$

$$x_i = -420.9687, \mathrm{i} = 1 : \mathrm{n}$$

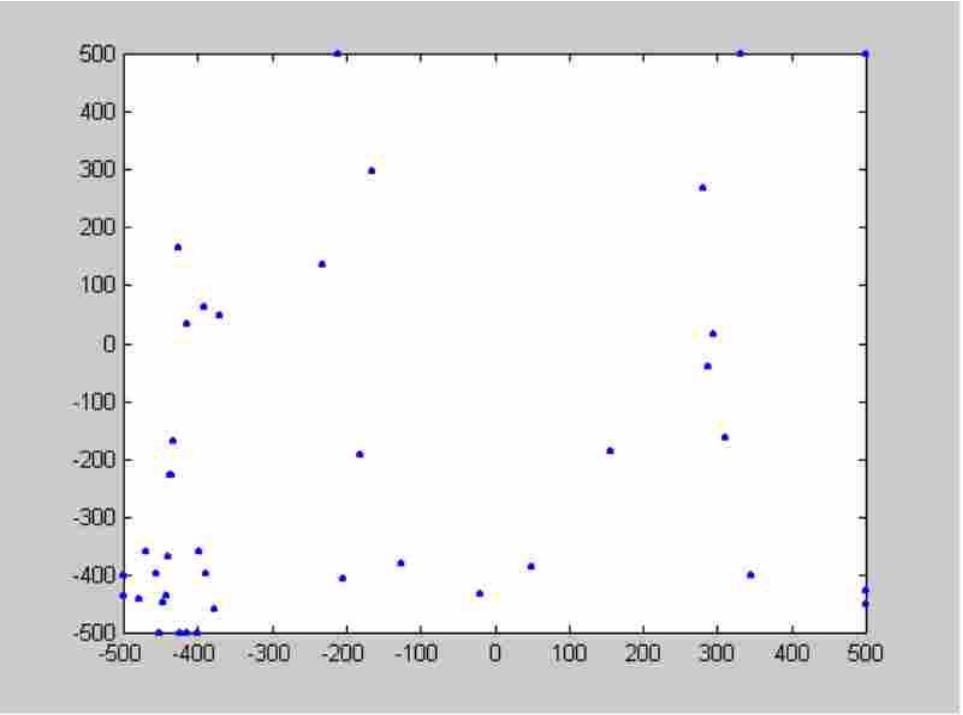
Evolution—Initialization



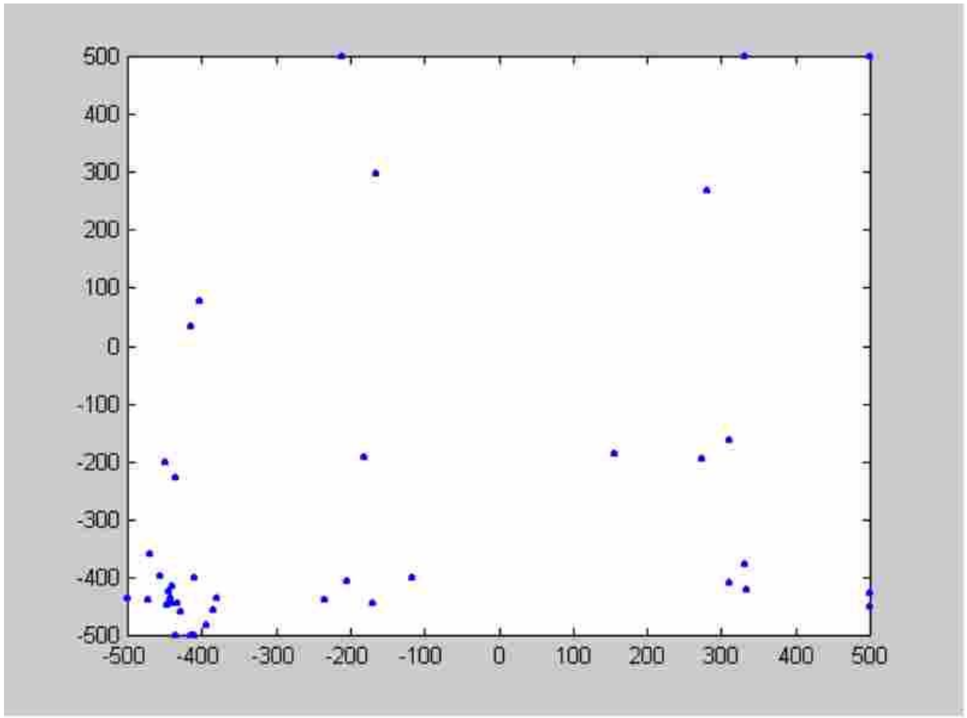
Evolution—5 iteration



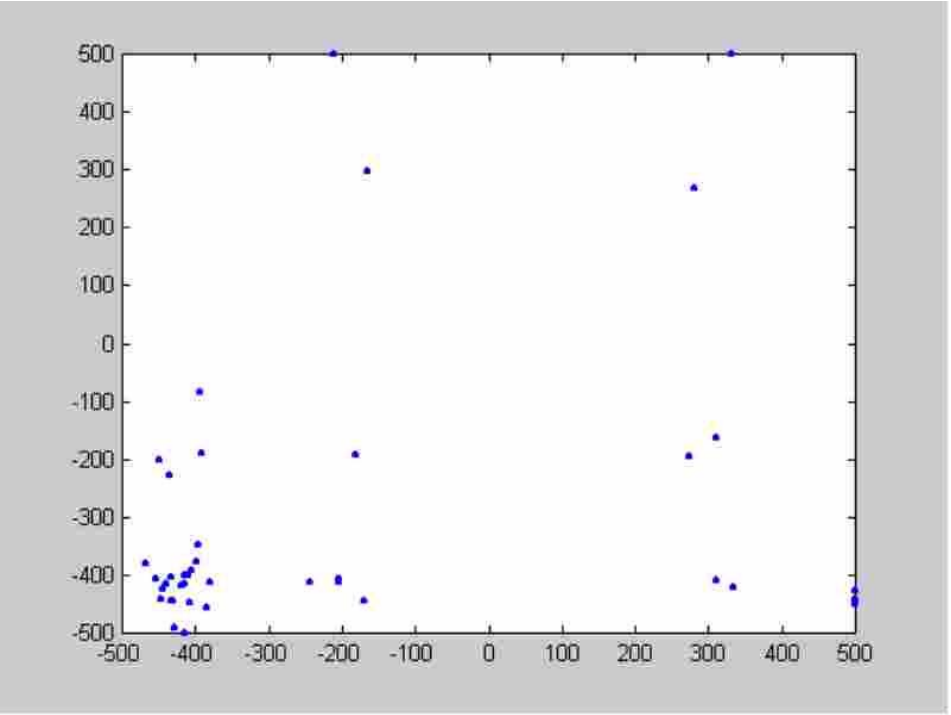
Evolution－10 iteration



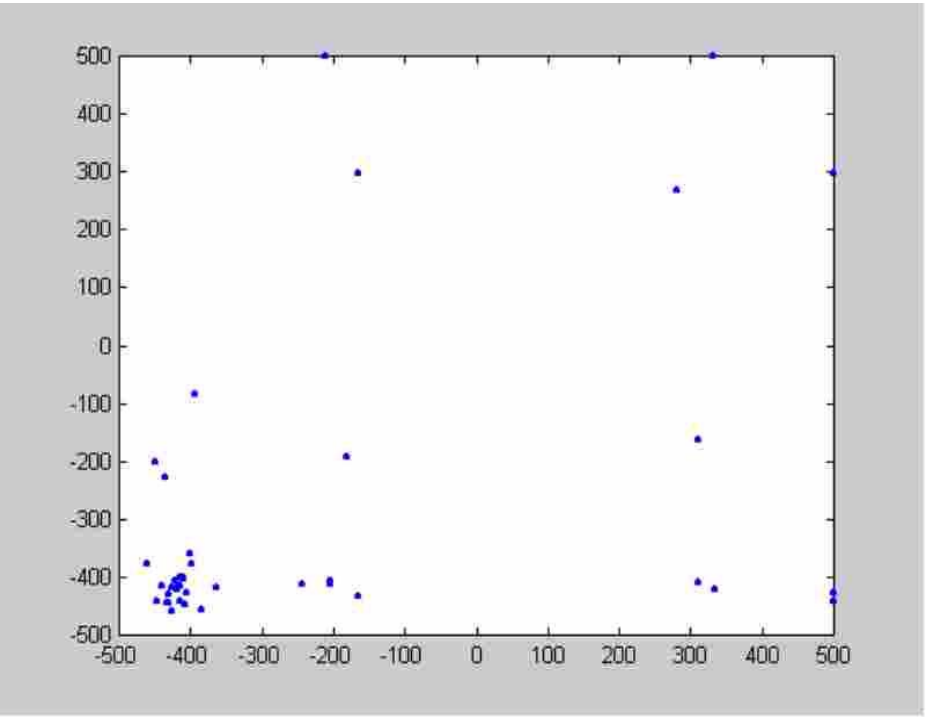
Evolution—15 iteration



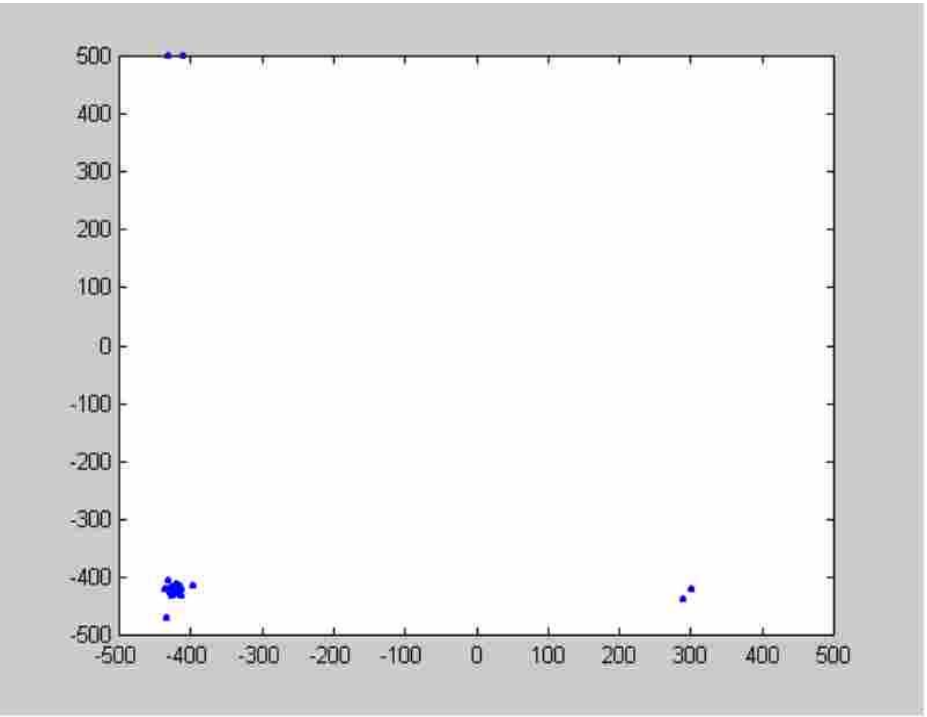
Evolution—20 iteration



Evolution—25 iteration



Evolution－100 iteration



Evolution—500 iteration