Part II Swarm Intelligence

Chapter 6
Particle Swarm Optimization

CSI Overview

- Computational Swarm Intelligence（CSI）
- A swarm can be defined as a group of (generally mobile) agents that communicate with each other(either directly or indirectly), by acting on their local environment.
- The interactions between agentsresult in distributive collective problem-solving strategies.

- Swarm intelligence (SI)refers to the problem-solving behavior that emerges from the interaction of such agents, andcomputational swarm intelligence (CSI)refers to algorithmic models of such behavior.
  - Swarm intelligence is the property of a system whereby the collective behaviors of unsophisticated agents interactinglocally with their environmentcause coherent functional global patterns to emerge.

CSI Overview

- Studies of social animals and social insectshave resulted in a number of computational models of swarm intelligence.
  - Biological swarm systems: ants, termites, bees, spiders, fish schools, and bird flocks.
- Within these swarms,individuals' structureare relatively simple, but theircollective behavioris usually very complex.The complex behavior of a swarm is a result of the pattern of interactions between the individuals of the swarm over time.
  - This complex behavior is not a property of any single individual, and is usually not easily predicted or deduced from the simple behaviors of the individuals.This is referred to as emergence.

计算群体智能概述

- 一些群体智能计算模型来源于社会性昆虫和动物的研究。
  - 生物群体系统：蚂蚁、白蚁、蜜蜂、蜘蛛、鱼群和鸟群等。
- 这些群体中的个体结构相对都比较简单，但它们的集体行为往往比较复杂。群体的复杂行为是群体中个体之间长期交互而形成模式的结果。
  - 这种复杂行为绝不是任何个体能具备的属性，并且通常也不能根据个体的简单行为来推断或预测，这叫做涌现（emergence）。

Emergence

- Emergence is the process of deriving some new and coherent structures, patterns and properties (or behaviors) in a complex system.
  - These structures, patterns and behaviors come to existence without any coordinated control system,but emerge from the interactions of individuals with their local (potentially adaptive)environment.

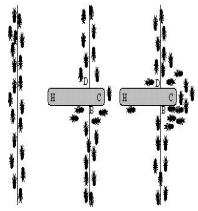
CSI Overview

- The collective behavior of a swarm of social organisms therefore emerges in a nonlinear manner from the behaviors of the individuals of that swarm.
- There exists a tight coupling between individual and collective behavior.
  - The collective behavior of individuals shapes and dictates the behavior of the swarm.
  - Swarm behavior has an influence on the conditions under which each individual performs its actions.

- Behaviors of that individual and its neighbors may change environment – which again may change the collective swarm behavior.
- The most important ingredient of swarm intelligence, and facilitator of emergent behavior, is interaction, or cooperation. Interaction among individuals aids in refining experiential knowledge about the environment.

CSI Overview

- Interaction in biological swarm systems happens in a number of ways, of which social interaction is the most prominent.
  - Here, interaction can be direct (by means of physical contact,or by means of visual, audio, or chemical perceptual inputs) or indirect (via local changes of the environment).



计算群体智能概述

- 大自然中涌现行为的例子数不胜数：
  - 白蚁构建的大型巢穴结构，其复杂性远远超过出任何单只白蚁的理解和能力。
  - 蚁群中的任务是动态分配的，没有任何管理者和协调者。
  - 蜜蜂摆动舞蹈募集（Recruitment），这导致了最优化的寻食行为。作为一种简单示踪行为的结果，寻食行为同样出现在蚁群中。
  - 鸟群和鱼群自组织形成最优的空间模式。鱼群基于小部分邻居个体决定它们的行为（如游行方向和速度）。鸟群的空间布局源于通过声音和视觉感知达到的沟通。
  - 肉食动物，例如一群狮子，展现的猎食策略比它们的猎食对象聪明。
  - 细菌通过分子沟通（可比作信息素）以跟踪环境中的变化。
  - 粘液菌类由非常简单的、能力有限的细胞有机体组成。然而，在食物短缺的时候，它们聚合形成移动“鼻涕虫”，以把个体运输到新的有食物的区域。

CSI Overview

- **The objective of computational swarm intelligence models: Modelling the simple behaviors of individuals, and the local interactions with the environment and neighboring individuals, in order to obtain more complex behaviors that can be used to solve complex problems.**

- For example, particle swarm optimization (PSO) models two simple behaviors:
  ① Each individual moves toward its closest best neighbor.
  ② Each individual moves back to the state that the individual has experienced to be best for itself.
  - As a result, the collective behavior that emerges is that of all individuals converging on the environment state that is best for all individuals.

- On the other hand, ant colony optimization models the very simple pheromone(信息素) trail-following behavior of ants.
  - Each ant perceives pheromone concentrations in its local environment and acts by probabilistically selecting the direction with the highest pheromone concentration.

Contents of this chapter

- 6-1 Basic Particle Swarm Optimization
- 6-2 Social interaction and social networks
- 6-3 Basic variations of the PSO
- 6-4 PSO parameters
- 6-5 Single-Solution Particle Swarm Optimization

6-6 Advanced topics

- 6-7 Applications

- Kennedy J，Eberhart R.
- Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks . 1995. 1942~1948.

Particle Swarm Optimization

- The particle swarm optimization(PSO) algorithm is a population-based search algorithm based on the simulation of the social behavior of birds within a flock.
- The initial intent of the particle swarm concept was to graphically simulate the graceful and unpredictable choreography of a bird flock, with the aim of discovering patterns that govern the ability of birds to fly synchronously, and to suddenly change direction with a regrouping in an optimal formation.
  - From this initial objective, the concept evolved into a simple and efficient optimization algorithm.

- In PSO, individuals, referred to as particles, are "flown" through hyperdimensional search space. Changes to the position of particles within the search space are based on the social-psychological tendency of individuals to emulate the success of other individuals
  - The changes to a particle within the swarm are therefore influenced by the experience, or knowledge, of its neighbors. The search behavior of a particle is thus affected by that of other particles within the swarm.
  - PSO is therefore a kind of symbiotic cooperative algorithm共生协同算法

Basic Particle Swarm Optimization

- 6.1.1 Global Best PSO
- 6.1.2 Local Best PSO
- 6.1.3 gbest versus lbest PSO
- 6.1.4 Velocity Components
- 6.1.5 Geometric Illustration
- 6.1.6 Algorithm Aspects

Basic Particle Swarm Optimization

- Individuals in a particle swarm follow a very simple behavior: emulate the success of neighboring individuals and their own successes.
  - The collective behavior that emerges from this simple behavior is that of discovering optimal regions of a high dimensional search space.

- A PSO algorithm maintains a swarm of particles, where each particle represents a potential solution.
  - In analogy with evolutionary computation paradigms, a swarm is similar to a population, while a particle is similar to an individual.
- The particles are "flown" through a multidimensional search space, where the position of each particle is adjusted according to its own experience and that of its neighbors.

Basic Particle Swarm Optimization

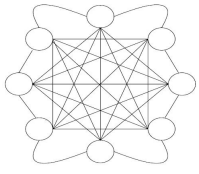
- Let $x_i(t)$ denote the position of particle i in the search space at time step t; t denotes discrete time steps.
- The position of the particle is changed by adding a velocity $v_i(t)$ to the current position, i.e.

$$x_i(t+1) = x_i(t) + v_i(t+1)$$
$$With,\ x_i(0) \sim U(\mathbf{x}_{\min}, \mathbf{x}_{\max})$$

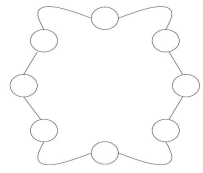
- It is the velocity vector $v_i(t)$ that drives the optimization process.
  - Velocity vector reflects both the experiential knowledge of the particle and socially exchanged information from the particle's neighborhood.
  - The experiential knowledge of a particle is generally referred to as the cognitive component, which is proportional to the distance of the particle from its own best position (referred to as the particle's personal best position).
  - The socially exchanged information is referred to as the social component of the velocity equation.

Basic Particle Swarm Optimization

- Two PSO algorithms have been developed which differ in the size of their neighborhoods
  - gbest PSO, global best PSO
  - lbest PSO, local best PSO



gbest PSO uses the star topology



lbest PSO uses a ring social network topology

6.1.1 gbest PSO

- For the global best PSO(gbest PSO), the neighborhood for each particle is the entire swarm. The social network employed by the gbest PSO reflects the star topology.
- For the star neighborhood topology, the social component of the particle velocity update reflects information obtained from all the particles in the swarm.
  - In this case, the social information is the best position found by the swarm, referred to as $\hat{y}(t)$.

gbest PSO

- For gbest PSO, the velocity of particle i is calculated as:

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2j}(t)\left[\hat{y}_j(t) - x_{ij}(t)\right]$$

Where,

$v_{ij}(t)$ is the velocity of particle i at time step t in dimension j=1,…,n_x;

$x_{ij}(t)$ is the position of particle i at time step t in dimension j,

c_1 and c_2 are positive acceleration constants used to scale the contribution of the cognitive and social components respectively,

and r_{1j} and r_{2j} are random values in the range [0, 1], sampled from a uniform distribution.

gbest PSO

- The personal best position $\mathbf{y}_i$, associated with particle i is the best position the particle has visited since the first time step
- Considering minimization problems, the personal best position $\mathbf{y}_i$ at the next time step t+1, is calculated as :

$$\mathbf{y}_i(t+1) = \begin{cases} \mathbf{y}_i(t) & \text{if } f(\mathbf{x}_i(t+1)) \geq f(\mathbf{y}_i(t)) \\ \mathbf{x}_i(t+1) & \text{if } f(\mathbf{x}_i(t+1)) < f(\mathbf{y}_i(t)) \end{cases}$$

Where f: $R^{n_x} \rightarrow \mathrm{R}$ *is* the fitness function.

- As with Evolutionary Algorithms, the fitness function measures how close the corresponding solution is to the optimum, i.e. the fitness function quantifies the performance, or quality, of a particle (or solution).

gbest PSO

- The global best position $\hat{\mathbf{y}}$, *at time step t*, is defined as:

$$\hat{\mathbf{y}}(t) \in \{\mathbf{y}_0(t), \cdots, \mathbf{y}_{n_s}(t)\}$$
$$|f(\hat{\mathbf{y}}(t)) = \min\{f(\mathbf{y}_0(t)), \cdots, f(\mathbf{y}_{n_s}(t))\}$$

Where, n_s is the total number of particles in the swarm.

- It notes that the definition, $\hat{\mathbf{y}}$ is the best position discovered by any of the particles so far.
  - It is usually calculated as the best personal best position.
- The global best position can also be selected from the particles of the current swarm, in which case:

$$\hat{\mathbf{y}}(t) \in \{x_0(t), \cdots, x_{n_s}(t) | f(\hat{\mathbf{y}}(t)) = \min\{f(x_0(t)), \cdots, f(x_{n_s}(t))\}\}$$

gbest PSO algorithm

```
1.  Create and initialize an n_x-dimensional swarm;
2.  repeat
3.    for each particle i=1,…,n_s do
4.      if f(x_i) < f(y_i) then //set the personal best position
5.        y_i = x_i;
6.      end
7.      if f(y_i) < f(ŷ) then //set the global best position
8.        ŷ = y_i;
9.      end
10.   end
11.   for each particle i=1,…,n_s do
12.     update the velocity using equation v_i(t);
13.     update the position using equation x_i(t);
14.   end
15.  until stopping condition is true;
```

6.1.2 lbest PSO

- The local best PSO (lbest PSO), uses a ring social network topology , where smaller neighborhoods are defined for each particle.
  - The social component reflects information exchanged within the neighborhood of the particle, reflecting local knowledge of the environment
- With reference to the velocity equation, the social contribution to particle velocity is proportional to the distance between a particle and the best position found by the neighborhood of particles.

lbest PSO

- The velocity is calculated as:

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2j}(t)\left[\hat{y}_{ij}(t) - x_{ij}(t)\right]$$

Where, $\hat{y}_{ij}$ is the best position , found by the neighborhood of particle i in dimension j.

lbest PSO

- The local best particle position $\hat{\mathbf{y}}_i$, i.e. the best position found in the neighborhood N_i, is defined as:

$$\hat{\mathbf{y}}_i(t+1) \in \left\{N_i \middle| f\left(\hat{\mathbf{y}}_i(t+1)\right) = \min\{f(x)\}, \forall \mathbf{x} \in N_i\right\}$$

- with the neighborhood N_i defined as:

$$N_i = \left\{\mathbf{y}_{i-n_{N_i}}(t), \mathbf{y}_{i-n_{N_i}+1}(t), \cdots, \mathbf{y}_{i-1}(t), \mathbf{y}_i(t), \mathbf{y}_{i+1}(t), \cdots, \mathbf{y}_{i+n_{N_i}}(t)\right\}$$

lbest PSO

- For neighborhoods of size n_{Ni}, the local best position will also be referred to as the neighborhood best position
- Note: for the basic PSO, particles within a neighborhood have no relationship to each other. Selection of neighborhoods is done based on particle indices.
  - However, strategies have been developed where neighborhoods are formed based on spatial similarity.

lbest PSO

- There are mainly two reasons why neighborhoods based on particle indices are preferred:
  - ① It is computationally inexpensive, since spatial ordering of particles is not required.
    - For approaches where the distance between particles is used to form neighborhoods, it is necessary to calculate the Euclidean distance between all pairs of particles, which is of $O(n_s^2)$ complexity.
  - ② It helps to promote the spread of information regarding good solutions to all particles, irrespective of their current location in the search space.

lbest PSO

- Neighborhoods overlap: A particle takes part as a member of a number of neighborhoods.
  - This interconnection of neighborhoods also facilitates the sharing of information among neighborhoods, and ensures that the swarm converges on a single point, namely the global best particle.
- The gbest PSO is a special case of the lbest PSO with $n_{Ni}=n_s$。

lbest PSO

```
1.  Create and initialize an n_x-dimensional swarm;
2.  repeat
3.    for each particle i=1,…,n_s do
4.      if f(x_i) < f(y_i) then //set the personal best position
5.        y_i = x_i;
6.      end
7.      if f(y_i) < f(ŷ_i) then //set the neighborhood best position
8.        ŷ_i = y_i;
9.      end
10.   end
11.   for each particle i=1,…,n_s do
12.     update the velocity using equation v_i(t) ;
13.     update the position using equation x_i(t) ;
14.   end
15.  until stopping condition is true;
```

Basic Particle Swarm Optimization

- Global Best PSO
- Local Best PSO
- gbest versus lbest PSO
- Velocity Components
- Geometric Illustration
- Algorithm Aspects

6.1.3 gbest versus lbest PSO

- gbest PSO and lbest PSO aresimilar:
  - The social component of the velocity updates causes both to move towards the global best particle.
    - For lbest PSO, this is due to the overlapping neighborhoods.
- There are two main differences between gbest PSO and lbest PSO with respect to convergence characteristics:
  - Due to the larger particle interconnectivity of the gbest PSO,it converges faster than the lbest PSO.
    - However, this faster convergence comes at the cost of less diversity than the lbest PSO.
  - As a consequence of its larger diversity,lbest PSO is less susceptible to being trapped in local minima.
    - In general (depending on the problem), neighborhood structures(such as the ring topology used in lbest PSO)improves performance.

Basic Particle Swarm Optimization

- Global Best PSO
- Local Best PSO
- gbest versus lbest PSO
- Velocity Components
- Geometric Illustration
- Algorithm Aspects

6.1.4 Velocity Components

- The velocity calculation as given in equations consists of three terms:
  - Previous velocity, cognitive component, social component.
- Previous velocity $v_i(t)$: serve as a memory of the previous flight direction, i.e. movement in the immediate past.
  - This memory term can be seen as a momentum, which prevents the particle from drastically changing direction,and to bias towards the current direction.This component is referred to asthe inertia component

Velocity Components

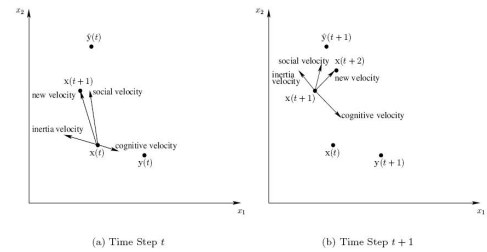
- Cognitive component $c_1r_1(y_i-x_i)$: quantify the performance of particle i relative to past performances. In a sense, the cognitive component resembles individual memory of the position that was best for the particle.
  - This term is that particles are drawn back to their own best positions, resembling the tendency of individuals to return to situations or places that satisfied them most in the past.
  - Kennedy and Eberhart referred to the cognitive component as the "nostalgia" of the particle.

Velocity Components

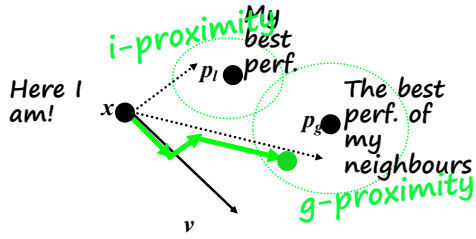
- Social component: each particle is drawn to the best solution in its neighborhood.
  - In the case of gbest PSOis $c_2r_2(\hat{y}-x_i)$
  - In the case of lbest PSO is$c_2r_2(\hat{y}_i-x_i)$
  - It quantifies the performance of particle i relative to a group of particles,or neighbors.
  - Conceptually, the social component resembles a group norm or standard that individuals seek to attain.
- The contribution of the cognitive and social components are weighed by a stochastic amountc_1r_1 or c_2r_2. These weights can effect the performance of algorithm.

Geometric Illustration

- The effect of the velocity equation can easily be illustrated in a two-dimensional vector space.
- Consider a single particle in a two-dimensional search space. The new position, x(t+1) moves closer towards the global best $\hat{y}(t)$.



(a) Time Step t (b) Time Step $t+1$



Basic Particle Swarm Optimization

- Global Best PSO
- Local Best PSO
- gbest versus lbest PSO
- Velocity Components
- Geometric Illustration
- Algorithm Aspects

6.1.5 Geometric Illustration

- Mainly due to the momentum term, it is possible for a particle to overshoot the global best position. This results in two scenarios:
  1. The new position, as a result of overshooting the current global best, may be a better position than the current global best. In this case the new particle position will become the new global best position, and all particles will be drawn towards it.
  2. The new position is worse than the current global best particle. In subsequent time steps the cognitive and social components will cause the particle to change direction back towards the global best.
- The cumulative effect of all the position updates of a particle is that each particle converges to a point on the line that connects the global best position and the personal best position of the particle.
  - A formal proof !

Multi-particle gbest PSO Illustration

- Visualizing the position updates uses gbest PSO with the task of minimizing a two-dimensional function with variables x_1 and x_2. The optimum is at the origin, indicated by the symbol X.
  - Figure(a) illustrates the initial positions of eight particles, with the global best position. Since the contribution of the cognitive component is zero for each particle at time step t=0, only the social component has an influence on the position adjustments. It is assumed that $v_i(0)=0$, for all particles.
  - Figure(b) shows the new positions of all the particles after the first iteration. A new global best position has been found with particles moving towards the new global best position.

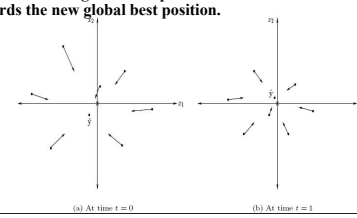


Illustration of lbest PSO

- The figures below shows how particles are influenced by their immediate neighbors.

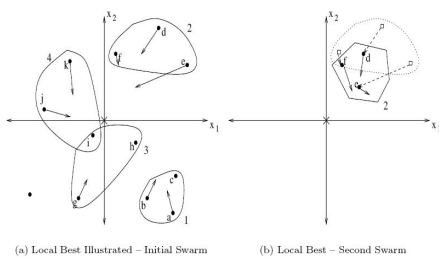
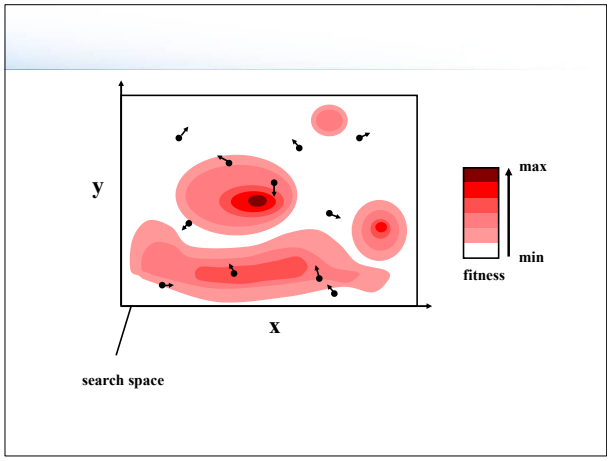
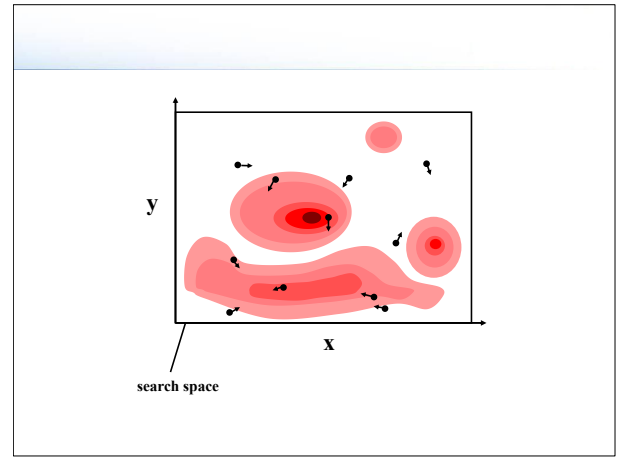
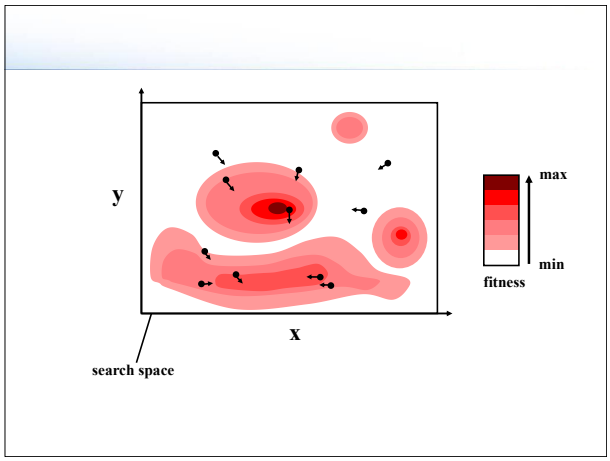


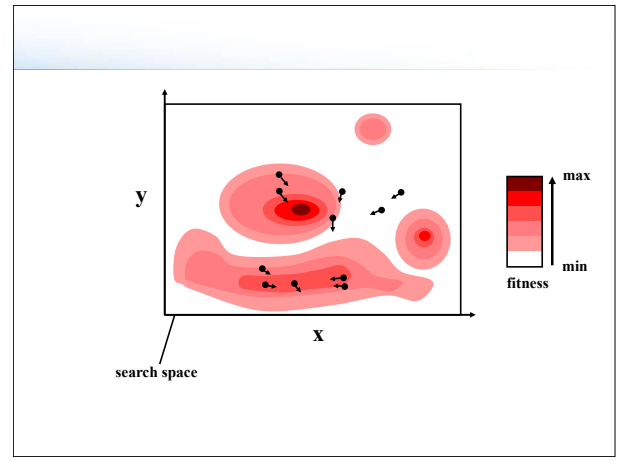
Illustration of lbest PSO

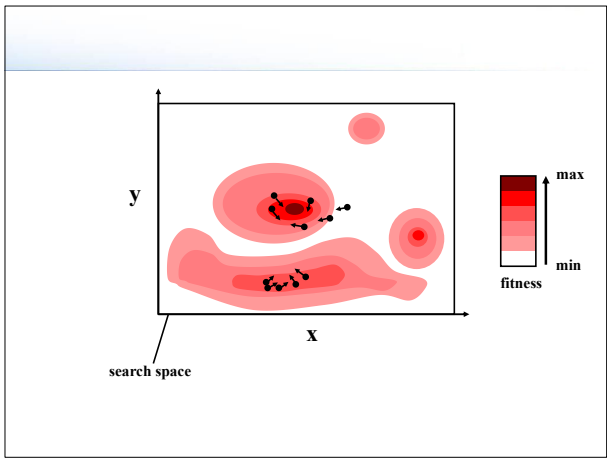
- The graph only indicates the aggregate velocity direction.
- First iteration :
  - In neighborhood 1, both particles a and b move towards particle c, which is the best solution within that neighborhood.
  - In neighborhood 2, particle d and e move to f.
- Next iteration :
  - e will be the best solution for neighborhood 2. Particles d and f move to e.

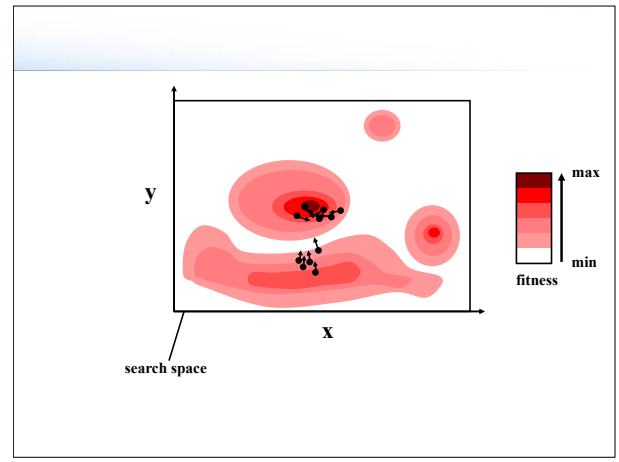

y
x
search space
max
min
fitness

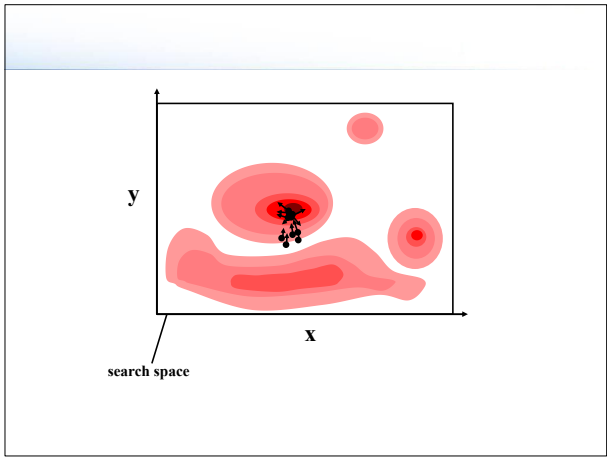

y
x
search space

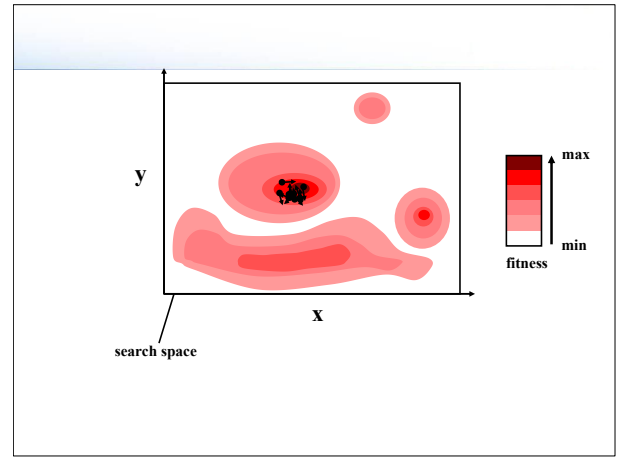

y
x
search space
max
min
fitness


y
x
search space
max
min
fitness


y
x
search space
max
min
fitness


y
x
search space
max
min
fitness





Basic Particle Swarm Optimization

- Global Best PSO
- Local Best PSO
- gbest versus lbest PSO
- Velocity Components
- Geometric Illustration
- Algorithm Aspects

Algorithm Aspects

- A few aspects of Algorithms still need to be discussed
  - Particle initialization
  - Stopping conditions
  - Defining the terms iteration
  - Function evaluation

Algorithm Aspects

- For gbest PSO and lbest PSO, the optimization process is iterative.
  - Repeat iterations of the algorithms until a stopping condition is satisfied.
- One such iteration consists of all the steps within the repeat...until loop, i.e. determining the personal best positions and the global best position, and adjusting the velocity of each particle.
- Within each iteration, a number of function evaluations are performed. A function evaluation refers to calculation of the fitness function, which characterizes the optimization problem.
  - For the basic PSO, a total of n_s function evaluations are performed per iteration , where n_s is the total number of particles in the swarm.

Initialize Swarm and Control Parameters

- The first step of the PSO algorithm is to initialize the swarm and control parameters
- In basic PSO, the acceleration constants c_1 and c_2, initial velocities, particle positions and personal best positions need to be specified.
  - In addition, lbest PSO requires the size of neighborhoods to be specified.
  - The importance of optimal values for the acceleration constants are discussed later.

Initialize Swarm and Control Parameters

- Positions of particles initialization
  - Usually, the positions of particles are initialized to uniformly cover the search space.
  - The efficiency of the PSO is influenced by the initial diversity of the swarm, i.e. how much of the search space is covered, and how well particles are distributed over the search space.
  - If regions of the search space are not covered by the initial swarm, the PSO will have difficulty in finding the optimum if it is located within an uncovered region. The PSO will discover such an optimum only if the momentum of a particle carries the particle into the uncovered area, provided that the particle ends up on either a new personal best for itself, or a position that becomes the new global best.

Initialize Swarm and Control Parameters

- Assume that an optimum located within the domain defined by the two vectors, $\mathbf{x}_{min}$ and $\mathbf{x}_{max}$, which respectively represents the minimum and maximum ranges in each dimension. Then, an efficient initialization methodfor the particle positionsis:

$$x(0) = x_{\min,j} + r_j\left(x_{\max,j} - x_{\min,j}\right),\ \ j = 1,\ldots,n_x,\ \ r_j \sim U(0,1)$$

Initialize Swarm and Control Parameters

- The initial velocitiescan be initialized to zero, i.e:

$$v_i(0) = 0,\ \ i = 1,\ldots,n_s$$

- While it is possible to alsoinitialize the velocities to random values, it is not necessary, and it must be done with care.
  - In fact, considering physical objects in their initial positions, their velocities are zero – they are stationary. If particles are initialized with nonzero velocities, this physical analogy is violated.

Initialize Swarm and Control Parameters

- Random initialization of position vectorsalready ensures random positions and moving directions.
- If, velocities are also randomly initialized, velocities should not be too large. Large initial velocities will have large initial momentum, and consequently large initial position updates. Such large initial position updates may cause particles to leave the boundaries of the search space, and may cause the swarm to take more iterations before particles settle on a single solution.

Initialize Swarm and Control Parameters

- The personal best positionfor each particle: particle's position at time step t=0, i.e.

$$\mathbf{y}_i(0) = \mathbf{x}_i(0)$$

- To ensure that the search space is uniformly covered, different initialization schemes have been used by researchers to initialize the particle positions: Sobol sequences, Faure sequences, and nonlinear simplex method.
- While it is important for application of the PSO to solve real-world problems, in that particles are uniformly distributed over the entire search space, uniformly distributed particles are not necessarily good for empirical studies of different algorithms.

Stopping Conditions

- Stopping conditionsare criteria used to terminate the iterative search process.
- When selecting a termination criterion, two important aspects have to be considered:
  ① The stopping condition should not cause the PSO to prematurely converge, since suboptimal solutions will be obtained.
  ② The stopping conditionshould protect against oversampling of the fitness If a stopping condition requires frequent calculation of the fitness function, computational complexity of the search process can be significantly increased.

Stopping Conditions

- The following stopping conditions have been used:
  - ① Terminate when a maximum number of iterations, or FEs, has been exceeded.
  - ② Terminate when an acceptable solution has been found.
  - ③ Terminate when no improvement is observed over a number of iterations.
  - ④ Terminate when the normalized swarm radius is close to zero.
  - ⑤ Terminate when the objective function slope is approximately zero.

Stopping Conditions

- **Terminate when a maximum number of iterations or FEs has been exceeded**
  - If this maximum number of iterations (or FEs) is too small, termination may occur before a good solution has been found.
  - Usually, this criterion is used in conjunction with convergence criteria to force termination if the algorithm fails to converge.
  - This criterion is useful in studies where the objective is to evaluate the best solution found in a restricted time period

Stopping Conditions

- **Terminate when an acceptable solution has been found**
  - Assume that x^* represents the optimum of the objective function f. Then, this criterion will terminate the search process as soon as a particle x_i is found such that $f(x_i) \le |f(x^*) - \varepsilon|$; that is, when an acceptable error has been reached.
  - The value of the threshold ε, has to be selected with care. If ε is too large, the search process terminates on a bad, suboptimal solution. On the other hand, if ε is too small, the search may not terminate at all.
  - This stopping condition assumes prior knowledge of what the optimum is - which is fine for problems such as training neural networks(where the optimum is usually zero). However, knowledge of the optimum is usually not available.

Stopping Conditions

- **Terminate when no improvement is observed over a number of iterations**
  - There are different ways in which improvement can be measured.
  - For example, if the average change in particle positions is small, the swarm can be considered to have converged. Alternatively, if the average particle velocity over a number of iterations is approximately zero, only small position updates are made, and the search can be terminated.
  - The search can also be terminated if there is no significant improvement over a number of iterations.
  - Shortcoming: Stopping conditions introduce two parameters for which sensible values need to be found.
    - ① The window of iterations(or function evaluations) monitors performance;
    - ② A threshold to indicate what constitutes unacceptable performance.

Stopping Conditions

- **Terminate when the normalized swarm radius is close to zero**
  - Normalized swarm radius, calculated as:

$$R_{norm} = \frac{R_{\max}}{\text{diameter}\ (S)}$$

where, diameter (S) is the diameter of the initial swarm, and the maximum radius $R_{\max}$ is

$$R_{\max} = \|x_m - \hat{y}\|,\ m \in 1,\ldots, n_s$$

$$\|x_m - \hat{y}\| \ge \|x_i - \hat{y}\|,\ \forall i = 1,\ldots, n_s$$

  - In the equations above, $\|\bullet\|$ is a suitable distance norm, e.g. Euclidean distance.
  - If the the diameter is close to zero, the swarm has little potential for improvement, unless the global best is still moving.

Stopping Conditions

- **Terminate when the normalized swarm radius is close to zero**
  - The algorithm is terminated when $R_{norm} < \varepsilon$. If ε is too large, the search process may stop prematurely before a good solution is found; Alternatively, if ε is too small, the search may take excessively more iterations for the particles to form a compact swarm, tightly centered around the global best position.
  - A more aggressive version: particles are clustered in the search space.
    - After particle clustering, there is a single cluster C (including $\hat{y}$).
    - If $|C|/n_s > \delta$, the swarm is considered to have converged. For example, $\delta = 0.7$, if at least 70% of the particles are centered around the global best position, the search will terminate.

Stopping Conditions

- Terminate when the normalized swarm radius is close to zero
  - A more aggressive version: particles are clustered in the search space.

```
//Particle Clustering Algorithm
1. Initialize cluster C={ŷ};
2. for about 5 times do
3.    Calculate the centroid of cluster, x̄ = (Σ_{i=1,x_i∈C}^{|C|} x_i) / |C|
4.    for ∀x_i∈S do
5.       if ||x_i − x̄|| < ε then
6.          C ← C∪{x_i}
7.       end
8.    end
9. end
```

Stopping Conditions

- Terminate when the objective function slope is approximately zero
  - The stopping conditions above consider the relative positions of particles in the search space, and do not take into consideration information about the slope of the objective function.
  - To base termination on the rate of change in the objective function, consider the ratio

$$f'(t) = \frac{f(\hat{\mathbf{y}}(t)) - f(\hat{\mathbf{y}}(t-1))}{f(\hat{\mathbf{y}}(t))}$$

for a number of iterations, if $f'(t) < \varepsilon$, the swarm is assumed to have converged.

  - This approximation to the slope of the objective function is superior to the methods above.
  - Because: it actually determines if the swarm is still making progress using information about the search space.

Stopping Conditions

- Terminate when the objective function slope is approximately zero
  - The problem of objective function slope approach: if some of the particles are attracted to a local minimum, the search will be terminated, irrespective of whether other particles may still be busy exploring other parts of the search space.
    - These exploring particles could have found a better solution had the search not terminated.
    - To solve this problem, the objective function slope method can be used in conjunction with the radius or cluster methods to test if all particles have converged to the same point before terminating the search process.
- Here, convergence does not imply that the swarm has settled on an optimum(local or global). The term convergence is meant that the swarm has reached an equilibrium, i.e. just that the particles converged to a point, which is not necessarily an optimum.

Contents of this chapter

- Basic Particle Swarm Optimization
- Social interaction and social networks
- Basic variations of the PSO
- PSO parameters
- Single-Solution Particle Swarm Optimization Advanced topics
- Applications

6.2 Social interaction and social networks

- The feature that drives PSO is social interaction.
  - Particles within the swarm learn from each other and, on the basis of the knowledge obtained, move to become more similar to their "better" neighbors.
- The social structure for PSO is determined by the formation of overlapping neighborhoods, where particles within a neighborhood influence one another.
  - This is in analogy with observations of animal behavior: an organism is most likely to be influenced by others in its neighborhood, and organisms that are more successful will have a greater influence on members of the neighborhood than the less successful.

Social interaction and social networks

- Within the PSO, particles in the same neighborhood communicate with one another by exchanging information about the success of each particle in that neighborhood. All particles then move towards some quantification of what is believed to be a better position.
  - The performance of the PSO depends strongly on the social network structure .

Social interaction and social networks

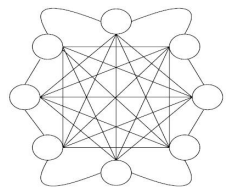
- The flow of information through a social network depends on:
  ① the degree of connectivity among nodes (members) of the network;
  ② the amount of clustering(clustering occurs when a node's neighbors are also neighbors to one another);
  ③ the average shortest distance from one node to another.
- With a highly connected social network, most of the individualscan communicate with one another.Consequently, information about the perceived best member quickly filters through the social network.
  - In terms of optimization, this means faster convergence to a solution than for less connected networks.
  - However, the faster convergence comes at the price of susceptibility to local minima, mainly due to the fact thatthe extent of coverage in the search space is less than for less connected social networks.

Social interaction and social networks

- For sparsely connected networks with a large amount of clustering in neighborhoods, it can also happen: the search space is not covered sufficiently to obtain the best possible solutions.
  - Each cluster contains individuals in a tight neighborhood covering only a part of the search space.
  - Within these network structures there usually exist a few clusters, with a low connectivity between clusters.
  - Consequently information on only a limited part of the search space is shared with a slow flow of information between clusters.
- For PSO, different social network structures have been developed.
  - star social structure, ring social structure, wheel social structure
  - pyramid social structure, four clusters social structure
  - Von Neumann social structure, and so on

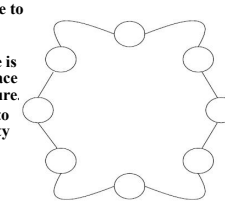
Star social structure

- The star social structure:all particles are interconnected. Each particle can therefore communicate with every other particle.
  - In this case each particle is attracted towards the best solution found by the entire swarm.
- The star network structure is the first implementation of the PSO, with the resulting algorithm generally being referred to as the gbest PSO.
- Gbest PSO performs best for unimodal problems.



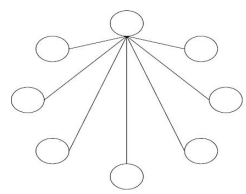
Ring social structure

- The ring social structure:each particle communicates with its n_N immediate neighbors. In the case ofn_N=2, a particle communicates with its immediately adjacent neighbors. Each particle attempts to imitate its best neighbor by moving closer to the best solution found within the neighborhood.
- Note:
  - The neighborhoods overlapin figure, which facilitates the exchange of information between neighborhoods, and convergence to a single solution in the end.
  - Since information flows at a slower rate through the social network, convergence is slower, but larger parts of the search space are covered compared to the star structure.
  - This behavior allows the ring structure to provide better performance in the quality of solutions found formulti-modal problems than the star structure. The resulting PSO algorithm is generally referred to as the lbest PSO.



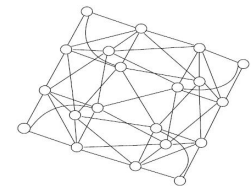
Wheel social structure

- The wheel social structure: individuals in a neighborhood are isolated from one another. One particle serves as thefocal point, and all information is communicated through the focal particle.
  - The focal particle compares the performances of all particles in the neighborhood, and adjusts its position towards the best neighbor.
  - If the new position of the focal particle results in better performance, then the improvement is communicated to all the members of the neighborhood.
- The wheel social networkslows down the propagation of good solutions through the swarm



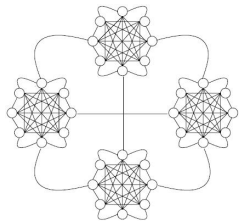
Pyramid social structure

- The pyramid social structure: forms a three-dimensional wire-frame.



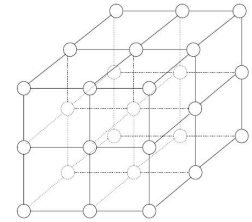
Four clusters social structure

- **The four clusters social structure**: four clusters (or cliques) are formed with two connections between clusters in this network structure.



Von Neumann social structure

- **The Von Neumann social structure**: particles are connected in a grid structure.
  - The Von Neumann social network has been shown in a number of empirical studies to outperform other social networks in a large number of problems.



Social interaction and social networks

- While many studies have been done using the different topologies, there is no outright best topology for all problems.
- In general, problem-solving performance depending on the degree of particle interconnection.
  - The fully connected structures perform best for unimodal problems, while the less connected structures perform better on multimodal problems.
- Neighborhoods are usually determined on the basis of particle indices.
  - For example, for the lbest PSO with n_N=2, the neighborhood of a particle with index i includes particles i-1 and i+1.
- While indices are usually used to determine neighborhoods, but some are based on the Euclidean distance between particles.

Contents of this chapter

- Basic Particle Swarm Optimization
- Social interaction and social networks
- Basic variations of the PSO
- PSO parameters
- Single-Solution Particle Swarm Optimization Advanced topics
- Applications

6.3 Basic Variations

- The basic PSO has been applied successfully to a number of problems, including standard function optimization problems, solving permutation problems, and training multi-layer neural networks.
- While the empirical results illustrated the ability of the PSO to solve optimization problems, these results also showed that the basic PSO has problems with consistently converging to good solutions.
- A number of basic modifications to the basic PSO have been developed to improve speed of convergence and the quality of solutions found by the PSO.
  - These modifications include the introduction of an inertia weight, velocity clamping, velocity constriction, different ways of determining the personal best and global best (or local best) positions, and different velocity models.

Basic Variations

- Velocity Clamping
- Inertia Weight
- Constriction Coefficient
- Synchronous versus Asynchronous Updates
- Velocity Models

6.3.1 Velocity Clamping

- Velocity Clamping
- An important aspect that determines the efficiency and accuracy of an optimization algorithm is the exploration–exploitation trade-off.
  - Exploration is a search algorithm.
  - Explore different regions of the search space in order to locate agood optimum.
  - Exploitation is the ability to concentrate the search around a promising area in order to refine a candidate solution.
- A good optimization algorithm optimally balances these contradictory objectives.
- Within the PSO, these objectives are addressed by the velocity update equation.

Velocity Clamping

gbest PSO:

$$v_{ij}(t+1)=v_{ij}(t)+c_1r_{1j}(t)\left[y_{ij}(t)-x_{ij}(t)\right]+c_2r_{2j}(t)\left[\hat{y}_j(t)-x_{ij}(t)\right]$$

lbest PSO:

$$v_{ij}(t+1)=v_{ij}(t)+c_1r_{ij}(t)\left[y_{ij}(t)-x_{ij}(t)\right]+c_2r_{2j}(t)\left[\hat{y}_{ij}(t)-x_{ij}(t)\right]$$

- The velocity updates consist of three terms that contribute to the step size of particles.
- In the early applications of the basic PSO, it was found that the velocity quickly explodes to large values, especially for particles far from the neighborhood best and personal best positions.
- Consequently, particles have large position updates, which result in particles leaving the boundaries of the search space-the particles diverge.

Velocity Clamping

- To control the global exploration of particles, velocities are clamped to stay within boundary constraints
  - If a particle's velocity exceeds a specified maximum velocity, the particle's velocity is set to the maximum velocity.
- Let $V_{max,j}$ denote the maximum allowed velocity in dimension j, Particle velocity is then adjusted before the position update using,

$$v_{ij}(t+1)=\begin{cases} v'_{ij}(t+1) & \text{如果} v'_{ij}(t+1)<V_{\max,j} \\ V_{\max,j} & \text{如果} v'_{ij}(t+1)\geq V_{\max,j} \end{cases}$$

where, v'_{ij} is calculated using equation.

- The value of $V_{max,j}$ is very important, since it controls the granularity of the search by clamping escalating velocities. Large values of $V_{max,j}$ facilitate global exploration, while smaller values encourage local exploitation.

Velocity Clamping

- If $V_{max,j}$ is too small, the swarm may not explore sufficiently beyond locally good regions.
  - Too small values for $V_{max,j}$ increase the number of time steps to reach an optimum.
- If $V_{max,j}$ is too large, it will meet a risk that the possibility of missing a good region. The particles may jump over good solutions, and continue to search in fruitless regions of the search space.
  - While large values of $V_{max,j}$ have the disadvantage that particles may jump over optima, particles are moving faster.
- Problem: how to find a good value in order to balance the following two aspects:
  - ① Moving too fast or too slow;
  - ② Exploration and exploitation.

Velocity Clamping

- Usually, $V_{max,j}$ values are selected to be a fraction of the domain of each dimension of the search space. That is:

$$V_{\max,j}=\delta\left(x_{\max,j}-x_{\min,j}\right)$$

where $x_{\max,j}$ and $x_{\min,j}$ are respectively the maximum and minimum values of the domain in dimension j, and $\delta\in(0,1]$.

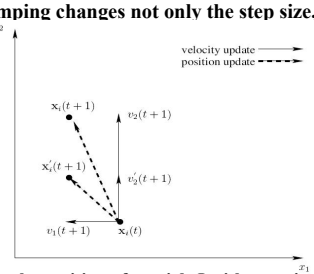
- Empirical studies found δ is problem-dependent.
  - For each different problem, the best value should be found using empirical techniques such as cross-validation.

Velocity Clamping

- There are two important aspects of the velocity clamping approach above:
  - ① Velocity clamping does not confine the positions of particles, only the step sizes as determined from the particle velocity.
  - ② A maximum velocity is associated with each dimension, proportional to the domain of that dimension.
- For the sake of the argument, assume that all dimensions are clamped with the same constant V_{max}.
  - If a dimension j, exists such that $x_{max,j}-x_{min,j}\ll V_{max}$, particles may still overshoot an optimum in dimension j.
- While velocity clamping has the advantage that explosion of velocity is controlled, it also has disadvantages that is in addition to the problem-dependent nature of $V_{max,i}$.
  - Velocity clamping changes not only the step size, but also the direction in which a particle moves
  - Problem occurs when all velocities are equal to the maximum velocity

Velocity Clamping

- Velocity clamping changes not only the step size, but also the direction.



- $x_i(t+1)$ denotes the position of particle I without using velocity clamping
- The position $x'_i(t+1)$ is the result of velocity clamping on the second dimension.

Velocity Clamping

- Problem occurs when all velocities are equal to the maximum velocity.
- If no measures are implemented to prevent this situation, particles remain to search on the boundaries of a hypercube defined by $[x_i(t) - V_{\max}, x_i(t) + V_{\max}]$
- A particle may stumble upon the optimum, but in general the swarm will have difficulty in exploiting this local area.
- Solution:
  - Reduce $V_{\max,i}$ over time
  - Introduce an inertia weight

Reducing $V_{\max,l}$ over time

- Reducing Vmax, I over time: To start with large values, allowing the particles to explore the search space; then to decrease the maximum velocity over time.
  - The decreasing maximum velocity constrains the exploration ability in favor of local exploitation at mature stages of the optimization process.
  - Approach 1: Change the maximum velocity when no improvement in the global best position:

$$V_{\max,j}(t+1) = \begin{cases} \gamma V_{\max,j}(t) & \text{if } f(\hat{y}(t)) \geq f(\hat{y}(t-t')) \ \forall t' = 1,\cdots,n_{t'} \\ V_{\max,j}(t) & \text{otherwise} \end{cases}$$

where γ decreases from 1 to 0.01 (the decrease can be linear or exponential).

Reducing $V_{\max,l}$ over time

- Approach 2: Exponentially decay the maximum velocity, using:

$$V_{\max,j}(t+1) = \left(1 - \left(\frac{t}{n_t}\right)^{\alpha}\right) \cdot V_{\max,j}(t)$$

where α is a positive constant, found by trial and error, or cross-validation methods; n_t is the maximum number of time steps (or iterations).

- Approach 3: the sensitivity of PSO to the value of δ can be reduced by constraining velocities using the hyperbolic tangent function:

$$v_{ij}(t+1) = V_{\max,j}(t) \cdot \tanh\left(\frac{v'_{ij}(t+1)}{V_{\max,j}(t)}\right)$$

where $v'_{ij}(t+1)$ is calculated from equation with gbest PSO or lbest PSO.

Basic Variations

- Velocity Clamping
- Inertia Weight
- Constriction Coefficient
- Synchronous versus Asynchronous Updates
- Velocity Models

6.3.2 Inertia Weight

- Inertia Weight is a mechanism to control the exploration and exploitation abilities of the swarm and to eliminate the need for velocity clamping.
  - The inertia weight was successful in addressing the first objective, but could not completely eliminate the need for velocity clamping.
- The inertia weight w controls the momentum of the particle by weighing the contribution of the previous velocity—basically controlling how much memory of the previous flight direction will influence the new velocity.

6.3.2 Inertia Weight

- For gbest PSO，the velocity equation changes from equation

$$v_{ij}(t+1)=v_{ij}(t)+c_1r_{1j}(t)\left[y_{ij}(t)-x_{ij}(t)\right]+c_2r_{2j}(t)\left[\hat{y}_j(t)-x_{ij}(t)\right]$$

to：

$$v_{ij}(t+1)=w\cdot v_{ij}(t)+c_1r_{1j}(t)\left[y_{ij}(t)-x_{ij}(t)\right]+c_2r_{2j}(t)\left[\hat{y}_j(t)-x_{ij}(t)\right]$$

Inertia Weight

- *The value of w* is extremely important to ensure convergent behavior and to optimally tradeoff exploration and exploitation.
- For $w\geq 1$, velocities increase over time, accelerating towards the maximum velocity (assuming velocity clamping is used), and the swarm diverges. Particles fail to change direction in order to move back towards promising areas.
- For $w<1$, particles decelerate until their velocities reach zero (depending on the values of the acceleration coefficients).
- Large values for *w* facilitate exploration, with increased diversity. A small *w* promotes local exploitation.
- Too small values eliminate the exploration ability of the swarm. Little momentum is then preserved from the previous time step, which enables quick changes in direction. The smaller*w*, the more do the cognitive and social components control position updates.

Inertia Weight

- As with the maximum velocity, the optimal value for the inertia weight is problemdependent.
- Problem: How to set inertia weight?
- Initial implementations of the inertia weight:
  ① Early, the inertia weight used a static value for the entire search duration, for all particles for each dimension.
  ② Later, use dynamically changing inertia values
- These approaches usually start with large inertia values, which decreases over time to smaller values.
  – particles are allowed to explore in the initial search steps, while favoring exploitation as time increases.

Inertia Weight

- Relationship between inertia weight w and the acceleration constants
  – Van de Bergh and Engelbrecht showed that

$$w>\frac{c_1+c_2}{2}-1,$$

guarantees convergent particle trajectories. If this condition is not satisfied, divergent or cyclic behavior may occur.

- Approaches to dynamically varying the inertia weight including:
  – Random adjustments
  – Linear decreasing
  – Nonlinear decreasing
  – Fuzzy adaptive inertia
  – Increasing inertia

Dynamical varying of the inertia weight

- Random adjustments: A different inertia weight is randomly selected at each iteration.
  – For example, one approach is to sample from a Gaussian distribution:

$$w\sim N(0.72,\sigma)$$

where σ is small enough to ensure that wis not predominantly greater than one.

  – Alternatively,

$$w=c_1r_1+c_2r_2$$

Dynamical varying of the inertia weight

- Linear decreasing: An initially large inertia weight (usually 0.9) is linearly decreased to a small value (usually 0.4).
  – For example,

$$w(t)=(w(0)-w(n_t))\frac{n_t-t}{n_t}+w(n_t)$$

where, n_t is the maximum number of time steps,$w(0)$ is the initial inertia weight, $w(n_t)$ is the final inertia weight, and $w(t)$ is the inertia at time step*t*, $w(0)>w(n_t)$。

Dynamical varying of the inertia weight

- **Nonlinear decreasing:** An initially large value decreases nonlinearly to a small value.
  - Nonlinear decreasing methods allow a shorter exploration time than the linear decreasing methods, with more time spent on refining solutions (exploiting).
  - Nonlinear decreasing methods will be more appropriate for smoother search spaces.
  - The nonlinear decreasing methods 1:

$$w(t+1)=\frac{w(t)-0.4}{n_t+0.4}(n_t-t),\quad \text{且} w(0)=0.9$$

Dynamical varying of the inertia weight

- The nonlinear decreasing methods 2:

$$w(t+1)=\alpha\cdot w(t')$$

where, α=0.975, and t' is the time step when the inertia last changed.

- The initial inertia weight $w(0)$=1.4 is used with a lower bound of $w(n_t)$=0.35. The initial $w(0)$=1.4 ensures that a large area of the search space is covered before the swarm focuses on refining solutions.
- The inertia is only changed when there is no significant difference in the fitness of the swarm.
- For example, Measure the variation in particle fitness of a 20% subset of randomly selected particles. If this variation is too small, the inertia is changed.

Dynamical varying of the inertia weight

- The nonlinear decreasing methods 3: An adaptive inertia weight approach where the amount of change in the inertia value is proportional to the relative improvement of the swarm.

$$w(t+1)=w(0)+(w(n_t)-w(0))\frac{e^{m_i(t)}-1}{e^{m_i(t)}+1}$$

where the relative improvement m_i is estimated as

$$m_i(t)=\frac{f(\hat{y}_i(t))-f(x_i(t))}{f(\hat{y}_i(t))+f(x_i(t))},\quad where\ w(n_t)\approx 0.5\ and\ w(0)<1.$$

  - Each particle has its own inertia weight based on its distance from the local best (or neighborhood best) position.
  - The local best position can just as well be replaced with the global best position.
  - Motivation: The more an individual improves upon his/her neighbors, the more he/she follows his/her own way.

Dynamical varying of the inertia weight

- Fuzzy adaptive inertia: the inertia weight is dynamically adjusted on the basis of fuzzy sets and rules.
- For example, a fuzzy system for the inertia adaptation to consist of the following components:
  - Two inputs, one to represent the fitness of the global best position, and the other the current value of the inertia weight.
  - One output to represent the change in inertia weight.
  - Three fuzzy sets, namely LOW, MEDIUM and HIGH, respectively implemented as a left triangle, triangle and right triangle membership function.
  - Nine fuzzy rules from which the change in inertia is calculated. An example rule in the fuzzy system is

```
if  normalized best fitness is LOW, and
      current inertia weight value is LOW
then  the change in weight is MEDIUM
```

Dynamical varying of the inertia weight

- Increasing inertia:
  - The inertia weight is linearly increased from 0.4 to 0.9.
- The linear and nonlinear adaptive inertia methods above are very similar to the temperature schedule of simulated annealing.

Basic Variations

- Velocity Clamping
- Inertia Weight
- Constriction Coefficient
- Synchronous versus Asynchronous Updates
- Velocity Models

6.3.3 Constriction Coefficient

- Clerc developed an approach very similar to the inertia weight to balance the exploration–exploitation trade-off, where the velocities are constricted by a constant, referred to as the constriction coefficient
- The velocity update equation changes to:

$$v_{ij}(t+1) = \chi\left[v_{ij}(t) + \phi_1\left(y_{ij}(t) - x_{ij}(t)\right) + \phi_2\left(\hat{y}_j(t) - x_{ij}(t)\right)\right]$$

where,

$$\chi = \frac{2\kappa}{\left|2 - \phi - \sqrt{\phi(\phi - 4)}\right|}$$

$$\phi = \phi_1 + \phi_2, \quad \phi_1 = c_1 r_1, \quad \phi_1 = c_1 r_1.$$

$$Note: \phi > 4 \text{ and } \kappa \in [0, 1].$$

- The above equations were derived from a formal eigenvalue analysis of swarm dynamics.

Constriction Coefficient

- The constriction approach was developed as a natural, dynamic way to ensure convergence to a stable point, without the need for velocity clamping.
  - Under the conditions that $\phi \geq 4$ and $\kappa \in [0, 1]$, the swarm is guaranteed to converge. The constriction coefficient χ evaluates to a value in the range [0, 1] which implies that the velocity is reduced at each time step.
- The parameter κ controls the exploration and exploitation abilities of the swarm.
  - For $\kappa \approx 0$, fast convergence is obtained with local exploitation. The swarm exhibits an almost hill-climbing behavior.
  - For $\kappa = 1$, results in slow convergence with a high degree of exploration.
  - Usually, κ is set to a constant value. However, an initial high degree of exploration with local exploitation in the later search phases can be achieved using an initial value close to one, decreasing it to zero.

Constriction Coefficient

- The constriction approach is effectively equivalent to the inertia weight approach.
  - Both approaches have the objective of balancing exploration and exploitation, and in doing so of improving convergence time and the quality of solutions found.
- For w and χ, low values result in exploitation with little exploration, while large values result in exploration with difficulties in refining solutions.
  - For a specific χ, the equivalent inertia model can be obtained by simply setting $w = \chi$, $\phi_1 = \chi c_1 r_1$, $\phi_2 = \chi c_2 r_2$
- The differences in the two approaches
  - Velocity clamping is not necessary for the constriction model.
  - The constriction model guarantees convergence under the given constraints.
  - Via the constants ϕ_1 and ϕ_2, the constriction model has the ability to regulate the change in direction of particles

Basic Variations

- Velocity Clamping
- Inertia Weight
- Constriction Coefficient
- Synchronous versus Asynchronous Updates
- Velocity Models

6.3.4 Synchronous versus Asynchronous Updates

- The gbest and lbest PSO algorithms perform synchronous updates of the personal best and global (or local) best positions.
- Note: Synchronous updates of the new best positions and the gbest(or lbest) are done separately from the particle position updates.
- Asynchronous updates: Each particle calculates the new best positions after position updates.
  - Advantage of asynchronous updates: immediate feedback is given about the best regions of the search space, while feedback with synchronous updates is only given once per iteration.
- Carlisle and Dozier reason that asynchronous updates are more important for lbest PSO where immediate feedback will be more beneficial in loosely connected swarms, while synchronous updates are more appropriate for gbest PSO.

Synchronous versus Asynchronous Updates

- Selection of the global (or local) best positions is usually done by selecting the absolute best position found by the swarm (or neighborhood).
- Kennedy proposed to select randomly the best positions from the neighborhood.
  - This is done to break the effect that one, potentially bad, solution drives the swarm.
  - The random selection was specifically used to address the difficulties that the gbest PSO experience on highly multi-modal problems.

Synchronous versus Asynchronous Updates

- The best positions (gbest or lbest) are selected from the particle positions of the current iterations, or from the personal best positions of all particles, which strongly influenced the performance of the basic PSO.
  - The latter includes a memory component(the best positions are the best positions found over all iterations), and the former approach neglects the temporal experience of the swarm.
  - Selection from the personal best positions is similar to the "hall of fame" concept used within evolutionary computation.

Basic Variations

- Velocity Clamping
- Inertia Weight
- Constriction Coefficient
- Synchronous versus Asynchronous Updates
- Velocity Models

6.3.5 Velocity Models

- According to the overview of Kennedy, a number of variations to the full PSO models differ in: the components included in the velocity equation, and how best positions are determined
- This model includes mainly:
  - Cognition-Only Model
  - Social-Only Model
  - Selfless Model

Cognition-Only Model

- The cognition-only model from the original velocity equation as given in equation excludes the social component
  - For the cognition-only model, the velocity update changes to:

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)\left(y_{ij}(t) - x_{ij}(t)\right)$$

- Illustration:
  - Can introduce the inertia weight to above velocity models.
  - Only cognition-only model can be likened to nostalgia, and illustrates a stochastic tendency for particles to return toward their previous best position.
- From empirical work, Kennedy reported that the cognition-only model is slightly more vulnerable to failure than the full model.
  - It tends to locally search in areas where particles are initialized. The cognition-only model is slower in the number of iterations it requires to reach a good solution, and fails when velocity clamping and the acceleration coefficient are small.

Social-Only Model

- The social-only model excludes the cognitive component from the velocity equation.
  - For gbest PSO:

$$v_{ij}(t+1) = v_{ij}(t) + c_2 r_{2j}(t)\left(\hat{y}_j(t) - x_{ij}(t)\right)$$

  - For lbest PSO:

$$v_{ij}(t+1) = v_{ij}(t) + c_2 r_{2j}(t)\left(\hat{y}_{ij}(t) - x_{ij}(t)\right)$$

- For the social-only model, particles have no tendency to return to previous best positions. All particles are attracted towards the best position of their neighborhood.
- Kennedy empirically illustrated: the social-only model is faster and more efficient than the full and cognitive models.

Selfless Model

- The selfless model is basically the social model, but with the neighborhood best solution only chosen from a particle's neighbors.
  - In other words, the particle itself is not allowed to become the neighborhood best.
- Kennedy showed the selfless model to be faster than the social-only model for a few problems.

Contents of this chapter

- Basic Particle Swarm Optimization
- Social interaction and social networks
- Basic variations of the PSO
- Basic PSO Parameters
- Single-Solution Particle Swarm Optimization
- Advanced topics
- Applications

6.4 Basic PSO Parameters

- The basic PSO is influenced by a number of control parameters，including:
  - the dimension of the problem
  - number of particles
  - acceleration coefficients
  - inertia weight
  - neighborhood size
  - number of iterations
  - the random values that scale the contribution of the cognitive and social components
- If velocity clamping or constriction is used, the maximum velocity and constriction coefficient also influence the performance of the PSO.

Contents of this chapter

- Basic Particle Swarm Optimization
- Social interaction and social networks
- Basic variations of the PSO
- Basic PSO Parameters
- Single-Solution Particle Swarm Optimization
- Advanced topics
- Applications

6.5 Single-Solution PSO

- While PSO is an efficient optimization approach, the performance of the PSO issensitive to the values of control parameters. Also the basic PSO has a serious defect that may cause stagnation.
  - A variety of PSO variations have been developed, mainly to improve the accuracy of solutions, diversity and convergence behavior.
- This section reviews some of these variations for locating a single solution to unconstrained, single-objective, static optimization problems.
  1. Guaranteed Convergence PSO
  2. Social-Based Particle Swarm Optimization
  3. Hybrid Algorithms
  4. Sub-Swarm Based PSO
  5. Multi-Start PSO Algorithms
  6. Repelling Methods
  7. Binary PSO

Contents of this chapter

- Basic Particle Swarm Optimization
- Social interaction and social networks
- Basic variations of the PSO
- PSO parameters
- Single-Solution Particle Swarm Optimization
- Advanced topics
- Applications

6.6 Advanced topics

- A few PSO variations for solving constrained problems, multiobjective optimization problems, problems with dynamically changing objective functions and to locate multiple solutions.
  - Constraint Handling Approaches
  - Multi-Objective Optimization
  - Dynamic Environments
  - Niching PSO

Contents of this chapter

- Basic Particle Swarm Optimization
- Social interaction and social networks
- Basic variations of the PSO
- PSO parameters
- Single-Solution Particle Swarm Optimization
- Advanced topics
- Applications

6.7 Applications

- Neural Networks Training
- Architecture Selection
- Game Learning
- ……

Summary

- The basic framework of PSO
- Neighborhood structure of PSO algorithm(social network structure)
- Variations of algorithms and their characteristics
- PSO parameters and PSO application etc

Velocity Updating

- Why rand()?
  - As with GAs, to stop the swarm converging too quickly.
- Why c1 & c2?
  - These constants can be used to change the weighting between personal and population experience.（改变其大小可以控制个体知识和群体知识对当前个体的影响
- Why pbest?
  - This is the cognitive component which draws individuals back to their previous best situations
- Why gbest?
  - This is the social component where individuals compare themselves to others in their group

离散PSO

基本PSO是用于实值连续空间的优化技巧，然而许多实际问题是组合优化问题，因而文献提出一种离散形式的PSO方法，在该方法中每一个粒子在状态空间的每一维被严格限制为 0 或者 1，而每个v_{id}代表x_{id}取值是 1 的可能性。离散二进制PSO的速度更新公式为：

$$v_{id} = v_{id} + c_1 r_1 (pbest_{id} - x_{id}) + c_2 r_2 (gbest_{id} - x_{id}) \qquad (3.1)$$

而粒子的位置更新公式变化如下：

$$x_{id} = \begin{cases} 0 & if \ \ r_3 \geq S(v_{id}) \\ 1 & if \ \ r_3 < S(v_{id}) \end{cases} \qquad (3.2)$$

其中$S(v_{id}) = \dfrac{1}{1+\exp(-v_{id})}$为sigmoid函数，$r_3$为0到1之间的随机数；分量$v_{id}$决定了位置分量取1或0的概率，$v_{id}$越大，则$x_{id}$取1的概率越大，反之越小。

PSO vs GA

- 仿生算法
- 优化算法
- 随机并行搜索
- 由适应度引导搜索
- 有可能陷入局部最优，不能保证全局最优

PSO vs GA

- EC 模拟生物进化，强调适者生存，有选择、交叉变异操作。
- SI 模拟社会中个体相互协同，强调协同合作，跟随操作。

- GA是信息相互共享，PSO是信息单向流动。
- GA是全部种群改变，PSO有记忆。
- PSO更容易实现，参数少，速度比GA快。