CHAPTER 3-1
SWARM INTELLIGENCE AND PSO

Siamak Sarmady, UUT

Swarm Intelligence

Swarm Intelligence

- **Swarm Intelligence:** is the emergent collective intelligence of groups of simple agents (Bonabeau et al, 1999)
  - Is used to solve optimization problems, simulations etc.

- **Characteristics:**
  - **Population based:** composed of many individuals
  - **Homogeneous Individuals:** population is made of similar individuals
  - **Local interaction:** agents behave based on simple rules (e.g. try to follow the rest of swarm)
  - **Self-organization (No centralized Control):** individuals follow specific rules that results in emergent organization

Swarm Intelligence Algorithms

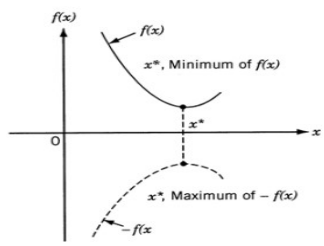
- Examples of swarm intelligence optimization:
  - Particle Swarm Optimization (PSO)
  - Ant Colony Optimization
  - Artificial Bee Colony Algorithm
  - Artificial Immune Systems Algorithm

- Swarm intelligence is considered a stochastic or random search techniques
  - Swarm Intelligence
  - Genetic Algorithm
  - Simulated Annealing

Heuristics

- **Heuristic:** algorithms that aim to find a good solution in a reasonable amount of time
  - Provide no guarantee of "goodness" or "efficiency"

- **Example applications:**
  - Can be used to solve NP-Complete problems
  - Decision problems which are optimization problem themselves



Minimum of $f(x)$ is same as maximum of $-f(x)$.

Metaheuristics

- **Metaheuristics:** are (roughly) high-level strategies that combining lower-level techniques for exploration and exploitation of the search space.
- Metaheuristic algorithms are:
  - Strategies that guide the search process
  - Approximate and usually non-deterministic
  - Not problem-specific
  - Used to efficiently explore the search space in order to find near-optimal solutions

- **Examples:**
  - Evolutionary Algorithms (Differential Evolution, Evolutionary programing, Genetic Algorithm)
  - Swarm Intelligence
  - Simulated Annealing
  - Tabu Search

Particle Swarm Optimization (PSO)

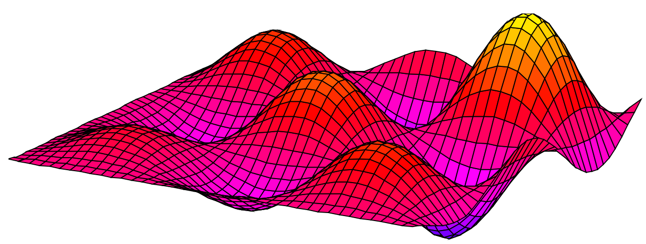
Particle Swarm Optimization (PSO)

- **Particle Swarm Intelligence (PSO):** was proposed as an optimization technique by Kennedy and Eberhart (1995)
- **Inspiration:** from the movements and social behavior of insects, birds and fish (e.g. in finding food)
  - **Solution:** is represented by the position of a particle
  - **Swarm of particles (solutions):** act as search agents or population of evolving solutions
- **Objective function:** like other optimization methods the function is used to guide the search
  - **Cost function:** used to find a global minimum solutions
  - **Fitness function:** used to find a global optimum solution
- **Note:** PSO doesn't use Gradient Descent, so it can be used to non linear problems once it doesn't require that the problem have to be differentiable.

Alpine function

- **Alpine function:** has many optima and we can easily compute the value for every pair of parameters

$$f(x_1, \dots, x_D) = \sin x_1 \dots \sin x_D \sqrt{x_1 \dots x_D}$$



Particle fly and search for the highest peak in the search space

PSO Search Idea and Scheme

- A swarm of particles (agents) move in the search space and look for the best solution.
- Each particle (candidate solution) moves in N-dimensional space.
  - It accelerates toward p_{best} and the g_{best} locations, with a random weighted acceleration at each time step (initial speed, i.e. inertia has some effect too)
    - $V(t)$: its current velocity (inertia)
    - X_{pbest}: its own movement experience (individual best). The best solution (fitness) a particle has achieved so far (Cognitive Term)
    - X_{gbest}: the movement experience of other particles (swarm best). The global best solution of all particles within the swarm (Social Term)

Particle Swarm Optimization (PSO)

- Each particle tries to modify its position X using the following formula:

$$X(t+1) = X(t) + V(t+1)$$

$$V(t+1) = w \times V(t) + c_1 \times r_1 \times (X_{pbest} - X(t)) + c_2 \times r_2 \times (X_{gbest} - X(t))$$

Inertia Term
Cognitive Term
Social Term

$V(t)$	velocity of the particle at time t
$X(t)$	Particle position at time t
w	Inertia weight
c_1 , c_2	learning factor or accelerating factor
rand	uniformly distributed random number between 0 and 1
X_{pbest}	particle's best position
X_{gbest}	global best position

PSO Algorithm

```
FOR each particle i
   FOR each dimension d
      Initialize position X_id randomly within permissible range
      Initialize velocity V_id randomly within permissible range
   END FOR
END FOR
Iteration k=1
DO
    FOR each particle i
       Calculate fitness value
       IF the fitness value is better than p_best_i in history
          Set current fitness value as the p_best_i
       END IF
    END FOR
    Choose the particle having the best fitness value as the g_best
    FOR each particle i
       FOR each dimension d
          Calculate velocity according to the equation
          V_id(k+1)= W V_id(k) + C_1 r_1(p_id - x_id) + C_2 r_2 (p_gd - x_id)
          Update particle position according to the equation
          X_id(k+1) = X_id(k) + V_id(k+1)
       END FOR
   END FOR
k=k+1
WHILE maximum iterations or minimum error criteria not reached
```

Initialization
Fitness Evaluation + Best Selection
Search (Moves)
Main Loop

PSO Algorithm

- In the first part of the algorithm we build a swarm of "I" particles each with "D" dimensions
  - Each dimension is filled with a uniform random number in the permissible range of that dimension
  - Velocity in each dimension is also filled with uniform random values

```
FOR each particle i
   FOR each dimension d
      Initialize position X_id randomly within permissible range
      Initialize velocity V_id randomly within permissible range
   END FOR
END FOR
```

PSO Algorithm

- In the main iterations loop (which takes either k loops or until minimum error is reached):
  - Evaluate particles (determine fitness of each, update the best position seen)
  - Select the best particle seen
  - Update velocities of particles to new values

```
Iteration k=1
DO
        ---- Evaluation of Particles ----
        ---- Selection of Global Best ----
        ---- Update Velocities ----
k=k+1
WHILE maximum iterations or minimum error criteria not reached
```

PSO Algorithm

- In the evaluation part
  - Calculate fitness/cost for each particle
  - Update the best position seen for each particle, if current position is better than all past
  - Update the best position ever by all particles in g_best_i

```
FOR each particle i
    Calculate fitness value
    IF the fitness value is better than p_best_i in history
       Set current fitness value as the p_best_i
    END IF
END FOR
Choose the particle having the best fitness value as the g_best
```

PSO Algorithm

- In the velocity update part, for each dimension of each particle:
  - For each dimension of each particle
  - Calculate velocity in each dimension by combining
    - Current speed in that dimension
    - Vector between current position and best position seen by this particle
    - Vector between current position and best position seen by all particle
  - Add the velocity to current position and find next position

FOR each particle i

FOR each dimension d

Calculate velocity according to the equation

$V_{id}(k+1) = W\ V_{id}(k) + C_1\ r_1\ (p_{id} - x_{id}) + C_2\ r_2\ (p_{gd} - x_{id})$

Update particle position according to the equation

$X_{id}(k+1) = X_{id}(k) + V_{id}(k+1)$

END FOR

END FOR

PSO Algorithm

- In the velocity update equation:

$$V_{id}(k+1)= W\, V_{id}(k) + C_1\, r_1\, (p_{id} - x_{id}) + C_2\, r_2\, (p_{gd} - x_{id})$$

  - **W:** is inertia constant that determines how much of the velocity is determined by current velocity
  - **C1 and C2:** are acceleration parameters. These two determine how much of change in speed is towards "best position seen by particle P_{id}" and "best position seen by all particles P_{gd}".
    - Of course random values $rand_1$ and $rand_2$ randomly affect these constants.
  - **P_{id} - X_{id}:** determines the vector towards "the best position seen by particle"
  - **P_{gd} - X_{id}:** determines the vector towards "the best position seen by all particles"

PSO Algorithm

- In the location update equation:

$$X_{id}(k+1) = X_{id}(k) + V_{id}(k+1)$$

  - **$X_{id}(k+1)$:** is the next position of a particle
  - **$X_{id}(k)$:** is the current position of the particle
  - **$V_{id}(k+1)$:** is the movement velocity calculated from previous equation