Particle Swarm Optimization

- a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality.
- solves a problem by having a population of candidate solutions, called particles, and moving these particles around in the search-space according to simple mathematical formula over the particle's position and velocity.

Each particle's movement

- influenced by its local best known position, but also guided toward the best known positions (updated as better positions found by other particles)
- expected to move the swarm toward the best solutions

- Inspired by Nature, **such as flocking and schooling patterns of birds and fish**
- a flock of birds circling over an area where they can smell a hidden source of food
- The one who is closest to the food chirps the loudest and the other birds swing around in his direction
- If any of the other circling birds comes closer to the target than the first, it chirps louder and the others veer over toward him
- This tightening pattern continues until one of the birds finds the food.

- originally attributed to **Kennedy, Eberhart in 1995**
- in 1997, Kennedy first simulated social behavior as a stylized representation of the movement of particles in a bird flock or fish school.
- a **Modified version by Eberhart & Shi** (the modification introduced a social factor to speed up the algorithm) **in 1998**

- computer software simulates of birds flocking around food sources,
- over a number of iterations, a group of variables have their values adjusted closer to the member whose value is closest to the target at any given moment.
- a simple and easy to implement algorithm

- Kennedy, J.; Eberhart, R.C. (2001) in *Swarm Intelligence* first describes many philosophical aspects of PSO and swarm intelligence.
- PSO is a **metaheuristic** which makes *few or no assumptions* about the problem being optimized and can **search very large spaces** of candidate solutions.

keeps track of three global variables:

- Target value or condition
- Global best (gBest) value indicating which particle's data is currently closest to the Target
- Stopping value indicating when the algorithm should stop if the Target isn't found

Each particle consists of:

- **Data** representing a possible solution
- A **Velocity** value indicating how much the Data can be changed
- A personal best (pBest) value indicating the closest the particle’s data has ever come to the Target

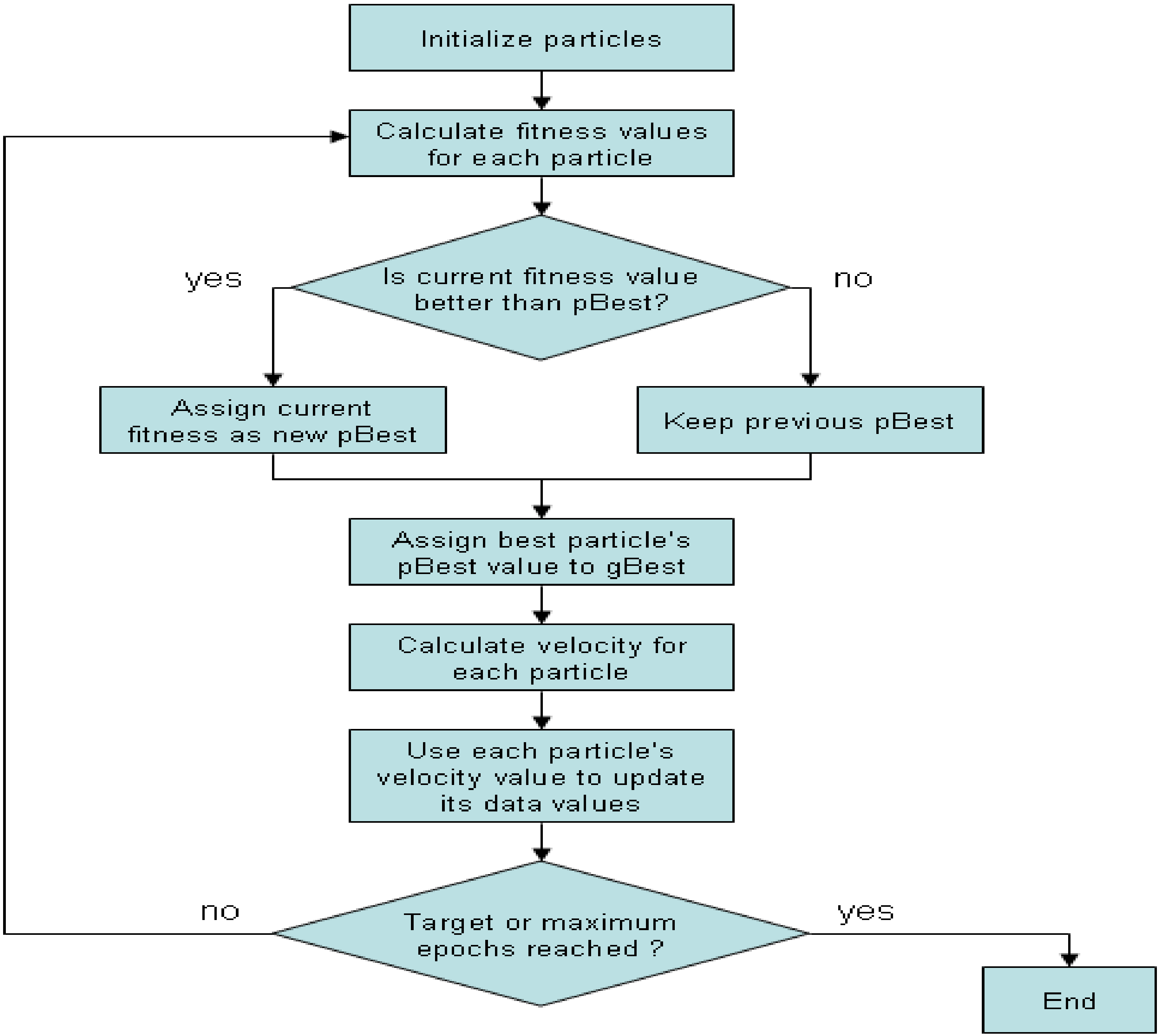
- The particles' data could be anything. In the flocking birds example above, the data would be the X, Y, Z coordinates of each bird.
- The individual coordinates of each bird would try to move closer to the coordinates of the bird which is closer to the food's coordinates (gBest).
- If the data is a pattern or sequence, then individual pieces of the data would be manipulated until the pattern matches the target pattern.

- The **velocity** value is calculated according to how far an individual's data is from the target. The further it is, the larger the velocity value.
- In the birds example, the individuals furthest from the food would make an effort to keep up with the others by flying faster toward the gBest bird.
- If the data is a pattern or sequence, the velocity would describe how different the pattern is from the target, and thus, how much it needs to be changed to match the target. (Making it similar to Neural Network)

- Each particle's pBest value only indicates the closest the data has ever come to the target since the algorithm started.
- The gBest value only changes when any particle's pBest value comes closer to the target than gBest.
- Through each iteration of the algorithm, gBest gradually moves closer and closer to the target until one of the particles reaches the target.
- It's also common to see PSO algorithms using population topologies, or "neighborhoods", which can be smaller, localized subsets of the global best value.

- These neighborhoods can involve two or more particles which are predetermined to act together, or subsets of the search space that particles happen into during testing.
- The use of neighborhoods often help the algorithm to avoid getting stuck in local minima.
- Neighborhood definitions and how they're used have different effects on the behavior of the algorithm.

Flow diagram



Three Steps of PSO Algorithm

1. **Evaluate fitness of each particle**
2. **Update individual and global bests**
3. **Update velocity and position of each particle**

repeat until some stopping condition is met.

Each particle maintains:

- Current position in the search space
- Velocity
- Individual best position
- Global best position

Position : the solution performance or fitness of each individual

Velocity : the moving direction in the search space

Velocity Update

$$V_i(t+1) = wV_i(t)+c_1r_1[Pb_i(t)-X_i(t)]+c_2r_2[Gb_i(t)-X_i(t)]$$

i : particle index

t : time index

w : **inertia weight** (or inertial coefficient, usually between 0.8 and 1.2)

c_1, c_2 : **acceleration coefficients** (between 0 and 2, and usually close to 2) r_1, r_2 : random values between 0 and 1

$V_i(t)$: particle *i*'s velocity at time t

$X_i(t)$: particle *i*'s position at time t

$Pb_i(t)$: particle's individual best solution as of time t

$Gb_i(t)$: swarm's (group) best solution as of time t

More about Velocity Update- Inertia Component w

$$V_i(t+1) = wV_i(t)+c_1r_1[Pb_i(t)-X_i(t)]+c_2r_2[Gb_i(t)-X_i(t)]$$

- *w* : **inertia weight** (or inertial coefficient, usually between 0.8 and 1.2) which keeps the particle moving in the same direction it was originally heading
- Lower values speed up convergence, higher values encourage exploring the search space

More about Velocity Update- Cognitive and Social Components

- $V_i(t+1) = wV_i(t)+c_1r_1[Pb_i(t)-X_i(t)]+c_2r_2[Gb_i(t)-X_i(t)]$

- c_1 is **Cognitive** Component which causes the particle to return to its individual best regions of the search space
- c_2 is **Social** Component which causes the particle to move to the best regions the swarm has found so far

Position(Solution) Update

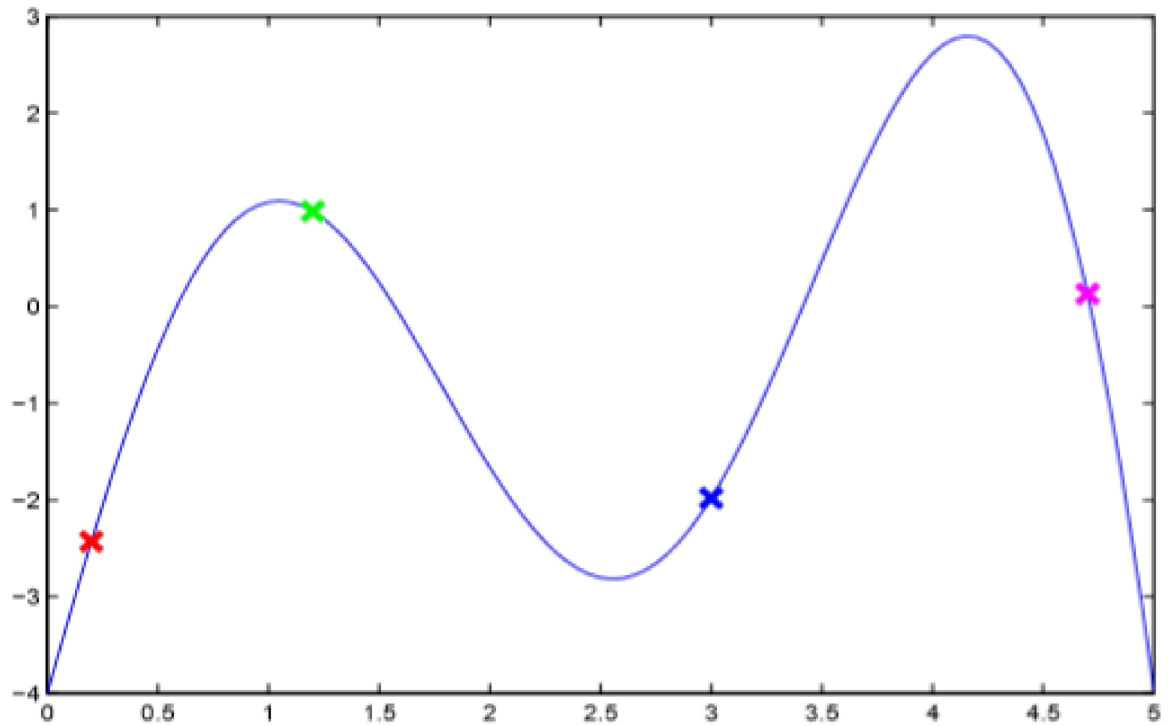
$X_i(t+1) = X_i(t) + V_i(t+1)$

where

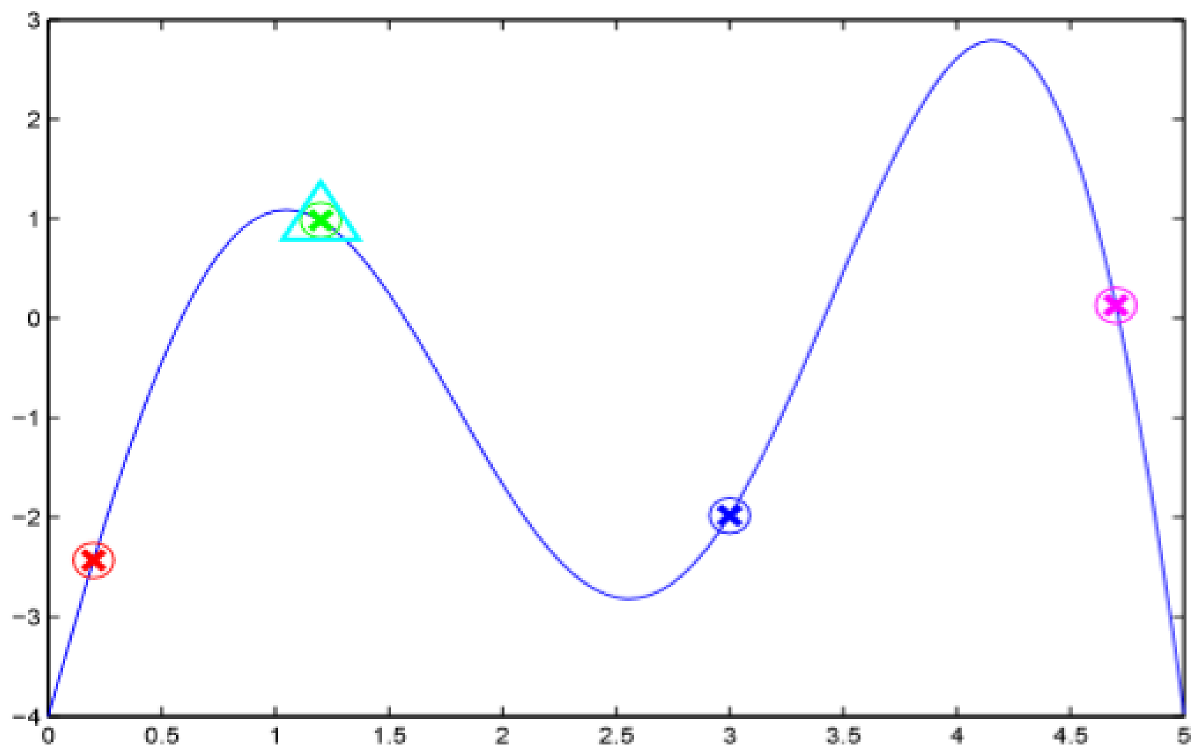
$X_i(t)$: particle *i*'s position at time t

PSO Example (James BloJandin, ArmstrongAtlanticStateUniversity)

Fitness Evaluation (t=1)

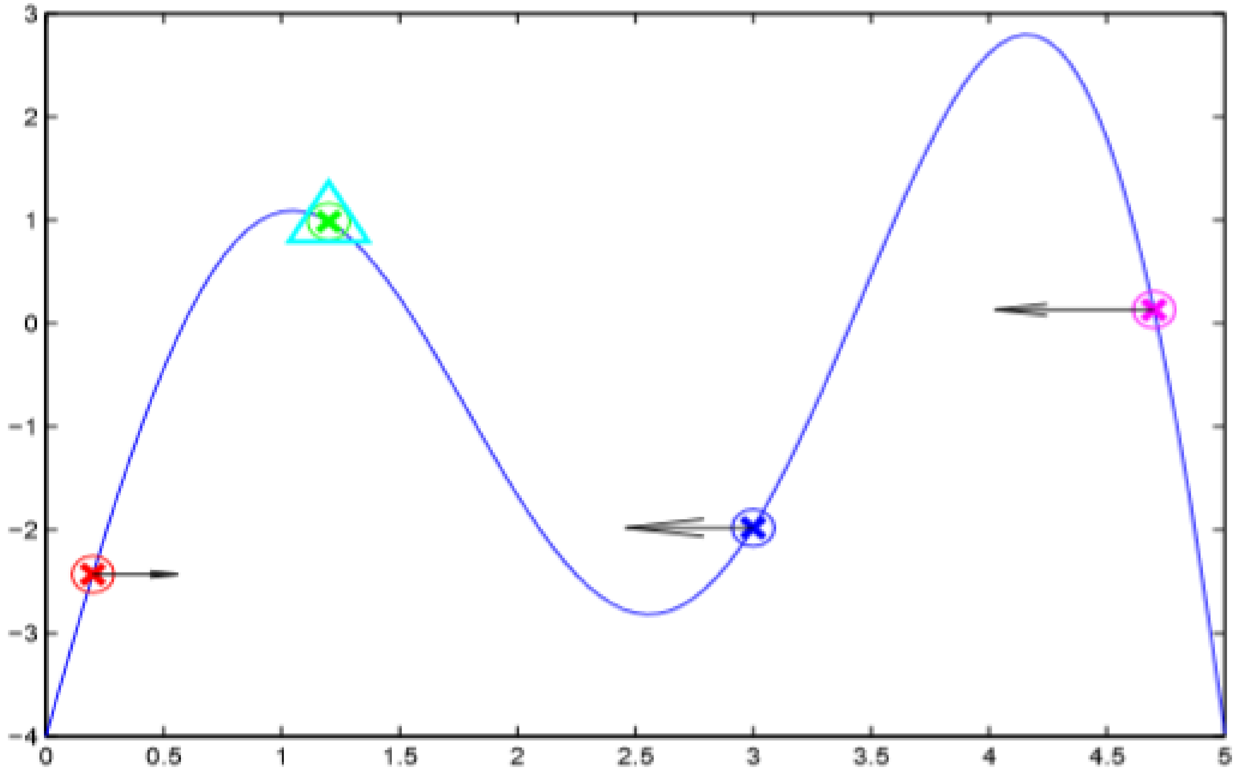


Update Individual / Global Bests (t=1)

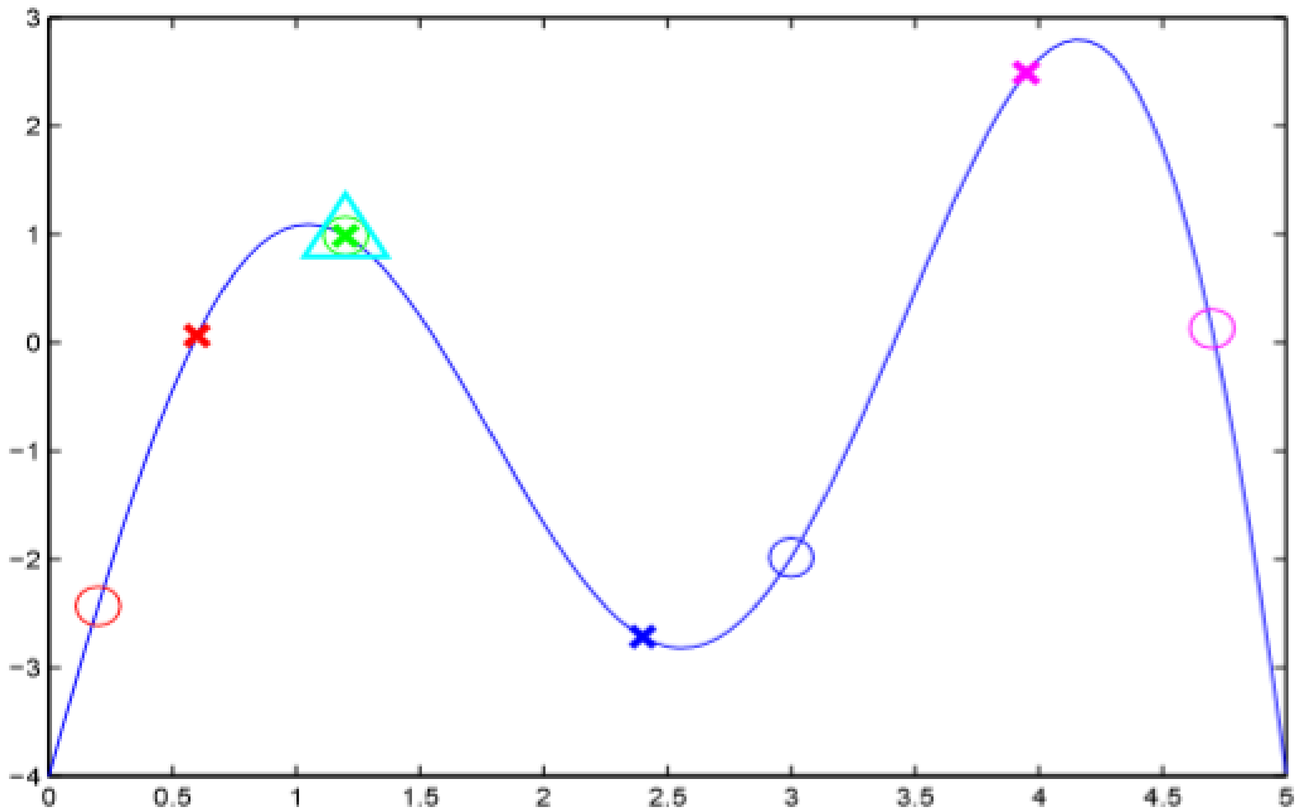


Update Velocity and Position (t=1)

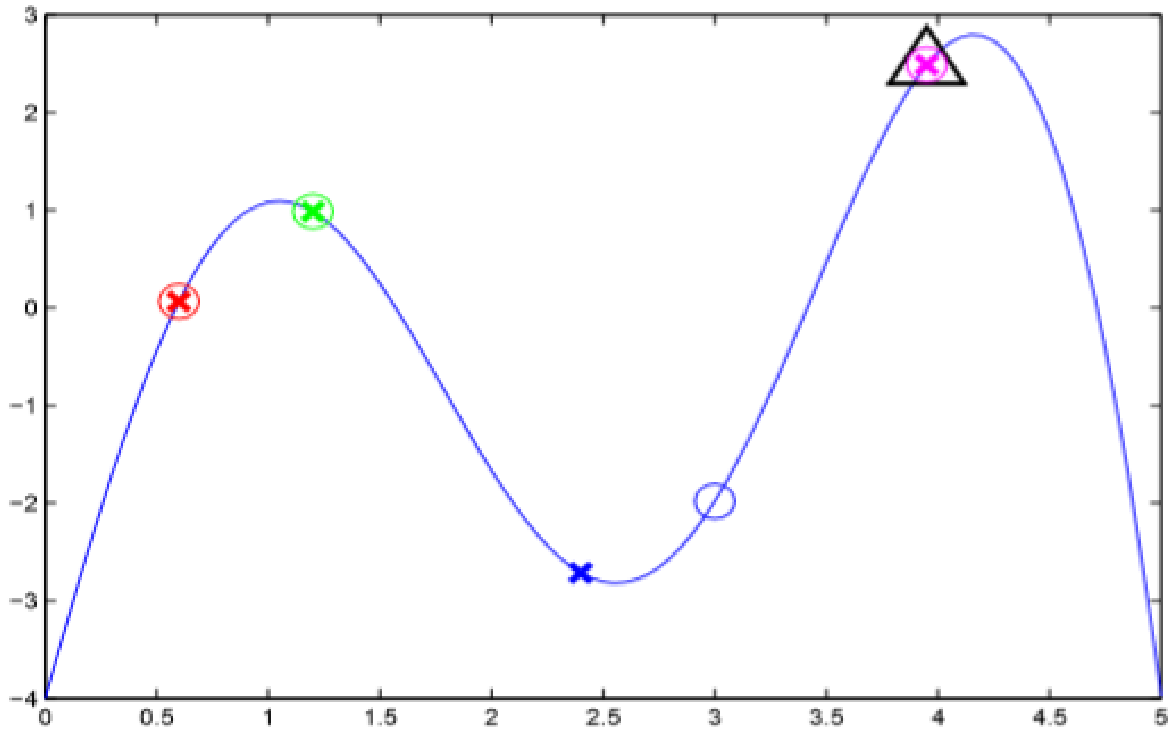
$$v_i(t+1) = wv_i(t)+c_1r_1[\hat{x}_i(t)-x_i(t)]+c_2r_2[g(t)-x_i(t)]$$



Fitness Evaluation (t=2)

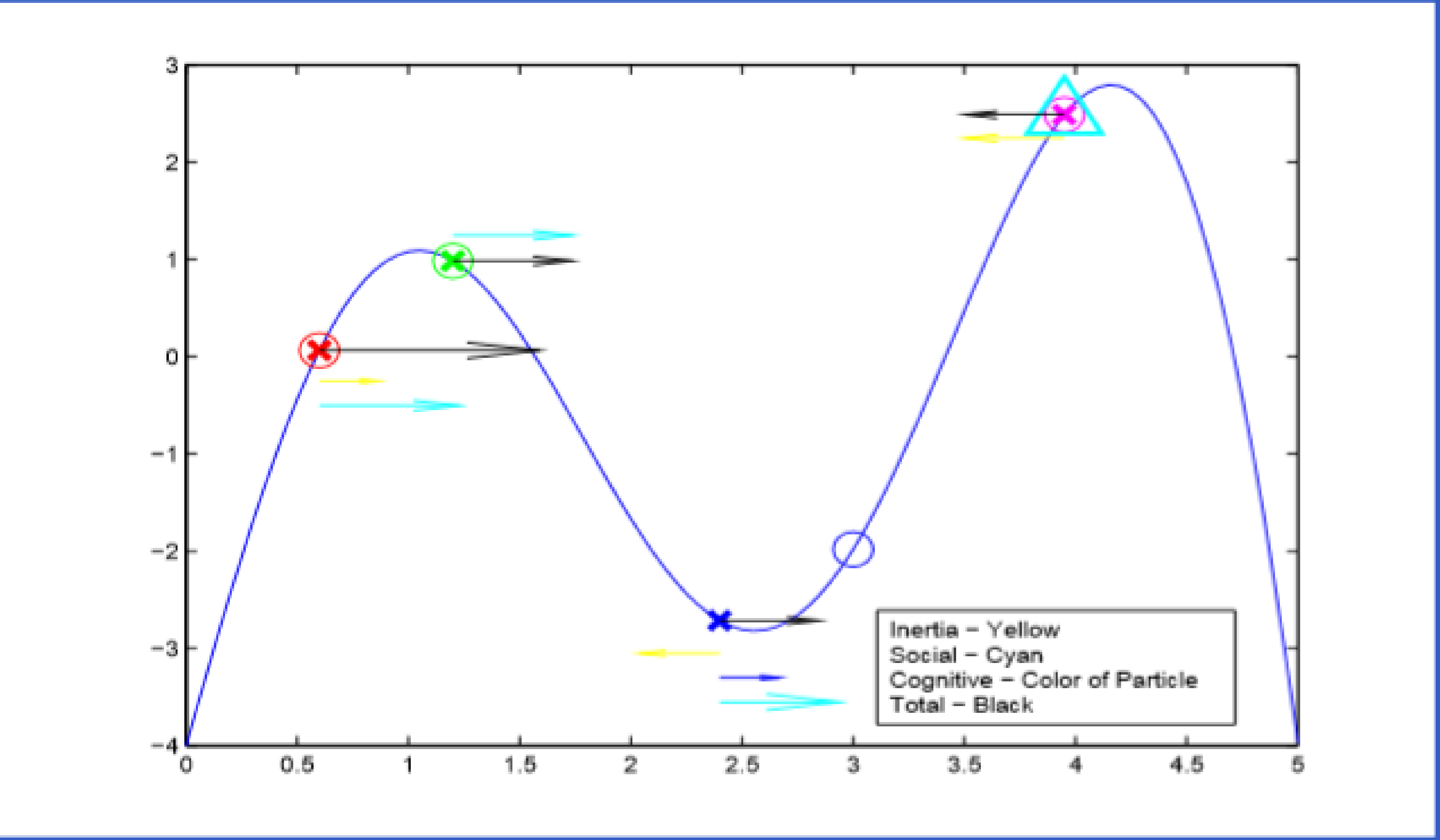


Update Individual / Global Bests (t=2)

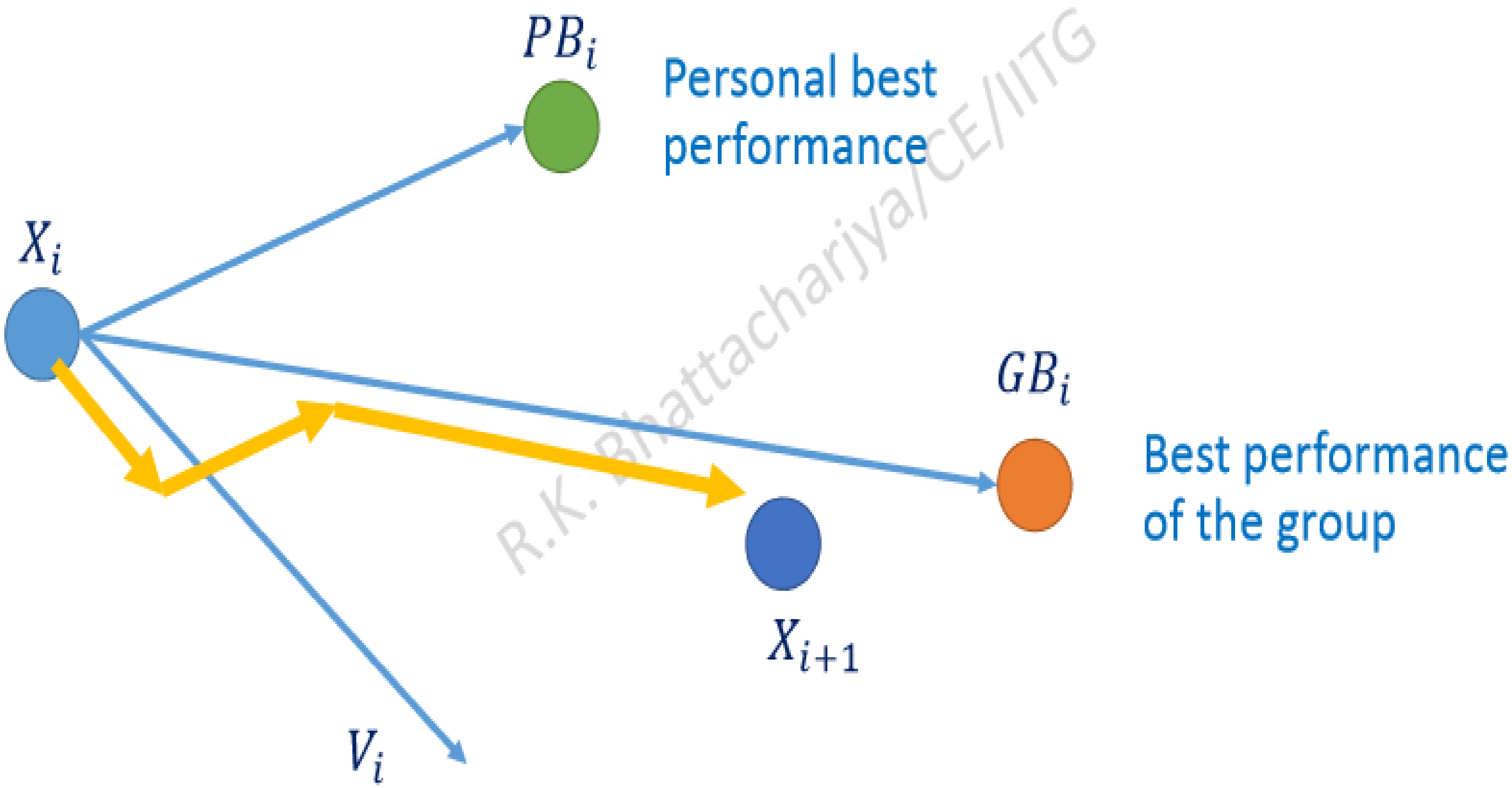


Update Velocity and Position (t=2)

$$v_i(t+1) = wv_i(t)+c_1r_1[\hat{x}_i(t)-x_i(t)]+c_2r_2[g(t)-x_i(t)]$$



Particle Swarm Algorithm



Particle Swarm Algorithm

Velocity is updated

$$V_{i+1} = \omega V_i + C_1 * rand() * (PB_i - X_i) + C_2 * rand() * (GB_i - X_i)$$

Position is updated

$$X_{i+1} = X_i + V_{i+1}$$

C_1 and C_2 are the learning factor
ω is the inertia weight

An Animation of Particle Swarm Optimization