Computational Intelligence

Unit # 10

Particle Swarm Optimization

- Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling.
- The system is initialized with a population of random solutions and searches for optima by updating generations.
- Unlike EA, PSO has no evolution operators such as crossover and mutation.

Concept

- In PSO, each single solution is a "bird" in the search space. We call it "particle".
- All of particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles.
- The particles fly through the problem space by following the current optimum particles.
- PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations.

Algorithm Description

- Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called *pbest*.
- Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called *lbest*.
- When a particle takes all the population as its topological neighbors, the best value is a global best and is called *gbest*.

Algorithm Description (Cont'd)

- At each step of the algorithm, particles are displaced from their current position by applying a velocity (gradient) vector to them.
- The magnitude and direction of their velocity at each step is influenced by their velocity in the previous iteration of the algorithm, simulated momentum, and the location of the a particle relative to the location of its pbest and the gbest.
- Therefore, at each step, the size and direction of each particle's move is a function of its own history (experience), and the social influence of its peer group.

Algorithm Description (Cont'd)

- After finding the two best values, the particle updates its velocity and positions with following equation (a) and (b).
- v[ ] = v[ ] + c1 * rand() * (pbest[ ] - present[ ]) + c2 * rand() * (gbest[ ] - present[ ]) (a)
  present[] = persent[ ] + v[ ] (b)

  v[ ] is the particle velocity, persent[] is the current particle (solution). pbest[ ] and gbest[ ] are defined as stated before. rand () is a random number between (0,1). c1, c2 are learning factors. usually c1 = c2 = 2.

Basic Theme

- The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its p*best* and *lbest* locations (local version of PSO).
- Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward *pbest* and *lbest* locations.

Role of pbest and gbest

- At the start of the algorithm, the pbest for each particle is set at its initial location, and gbest is set to the location of the best of the pbests.
- In each iteration of the algorithm, a particle is stochastically accelerated towards its previous best position and towards a neighborhood (global) best position, thereby forcing particles to continually search in the most-promising regions found so far in the solution space.

Role of c1 and c2

- The weight coefficient c1 and c2 control the relative impact of the pbest and gbest locations on the velocity of a particle.
- Low values for c1 and c2 allow each particle to explore far away from already uncovered good points, high values of the parameters encourage more intensive search of regions close to these points.

Comparing PSO with the EA

- Like the EA, PSO is population-based, it is typically initialized with a population (swarm) of random encodings of solutions, and search proceeds by updating these encodings over a series of generations (iterations).
- Both systems do not guarantee success.
- Both algorithms start with a group of a randomly generated population, both have fitness values to evaluate the population.
- Unlike the EA, PSO has no explicit selection process as all particles persist over time. Instead a *memory* in the form of gbest/lbest is substituted for selection.
- PSO also does not have genetic operators like crossover and mutation.

Comparing PSO with the EA (Cont'd)

- Compared with evolutionary algorithms (EAs), the information sharing mechanism in PSO is significantly different.
- In EAs, chromosomes share information with each other. So the whole population moves like a one group towards an optimal area.
- In PSO, only gBest (or lBest) gives out the information to others. It is a one -way information sharing mechanism. The evolution only looks for the best solution.
- Compared with EA, all the particles tend to converge to the best solution quickly even in the local version in most cases.

Working of the Algorithm for Assignment Part 1(c)

- Create 10 particles randomly
- Set c1 and c2 as 2.
- Initial velocity in each dimension is 0.
- Since both functions of Assignment 1(c) have two dimensions, your particles would also have two dimensions in each case.
- You would update velocity in each direct. For example V_{1x} and V_{1y} represents the probability of particle 1 in x and y directions, respetively.

Working of the Algorithm for Assignment Part 1(c) (Cont'd)

- You need to store the personal best (x and y) values for each particle.
- Similarly you need to store global best (x and y) for a particular generation.
- Keep in mind that you wouldn't change gbest during a generation. For example, suppose while updating particle # 3, you find a value which is better than the current gbest value. But you wouldn't treat this new value as gbest immediately.
- The new gbest value would be in effect from the next iteration (new generation).