Computational Intelligence

Unit # 16

Differential Evolution

- Differential evolution (DE) is a population-based search algorithm.
- The algorithm draws inspiration from the field of evolutionary computation, as it embeds implicit concepts of mutation, recombination and fitness-based selection to evolve good solutions to a problem of interest by manipulating a population of solution encodings.
- An individual in DE is generally comprised of a real-valued chromosome.

Working of DE

- At the start of the algorithm, a population of N, d-dimensional vectors $X_j = (x_{i1}, x_{i2}, \ldots, x_{id})$, $j = 1, \ldots, N$, each of which encode a solution, is randomly initialized and evaluated using a fitness function f.
- During the search process, each individual (j) is iteratively refined.

Three Major Steps

- The modification process has three steps:
  - Create a variant vector which encodes a solution, using randomly selected members of the population (mutation step).
  - Create a trial vector, by combining the variant vector with j (crossover step).
  - Perform a selection process to determine whether the newly-created trial vector replaces j in the population.

Mutation

- Under the mutation operator, for each vector $X_j(t)$ a variant vector $V_j(t+1)$ is obtained:
  - $V_j(t + 1) = X_m(t) + F(X_k(t) - X_l(t))$
  - where k, l,m ∈ 1, . . .,N are randomly selected indices, and all the indices ≠ j (X_m is referred to as the base vector, and $X_k(t) - X_l(t)$ is referred to as a difference vector).

Crossover

- Selecting the three indices randomly implies that all members of the current population have the same chance of being selected, and therefore influencing the creation of the difference vector.
- The difference between vectors X_k and X_l is multiplied by a scaling parameter F (typically F ∈ (0, 2]).
- The scaling factor controls the amplification of the difference between X_k and X_l, and is used to avoid stagnation of the search process.

Self-Scaling Mutation

- A notable attribute of the mutation step in DE is that it is self-scaling.
- The size/rate of mutation along each dimension stems solely from the location of the particles in the current population.
- The mutation step self-adapts as the population converges leading to a finer-grained search.
- In contrast, the mutation process in the canonical GA is typically based on draws from a separately defined (fixed) probability density function.

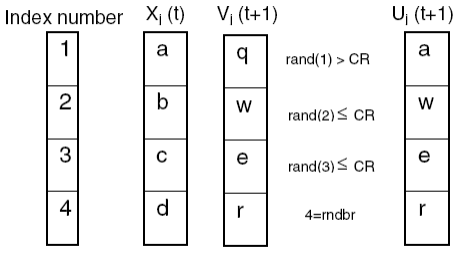
Cross Over

- Following the creation of the variant vector, a trial vector $U_j(t + 1)$ = (uj1, uj2, . . . , ujd) is obtained:

$$U_{jk}(t+1) = \begin{cases} V_{jk}(t+1), \text{ if } (rand \leq CR) \text{ or } (j = rnbr(ind)) \ ; \\ X_{jk}(t), \text{ if } (rand > CR) \text{ and } (j \neq rnbr(ind)). \end{cases}$$

- where k = 1, 2, . . . , d, rand is a random number generated in the range (0,1), CR is the user-specified crossover constant from the range (0,1), and rnbr(ind) is a randomly chosen index chosen from the range (1, 2, . . . , d).
- The random index is used to ensure that the trial solution differs by at least one component from $X_j(t)$.

Cross Over (Cont'd)



Mutation and Cross Over

Parents

a	b	c
4.435	1.222	9.735
6.567	3.986	1.503
2.456	2.325	7.908
4.567	1.254	3.420
3.976	5.124	5.394

i=1

First selection (i_4)

4.567	1.254	3.420

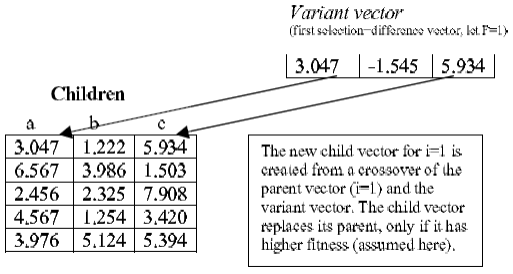
Second and third selections (i_3 & i_5)

3.976	5.124	5.394
2.456	2.325	7.908

Difference vector (i_5-i_3)

-1.520	-2.799	2.514

Mutation and Cross Over (Cont'd)



- The resulting trial (child) solution replaces its parent if it has higher fitness (a form of selection), otherwise the parent survives unchanged into the next iteration of the algorithm

Summary of DE

- Initialize Population
- Evaluate fitness of population members
- Do
  - For each member of the population
    - Perform mutation operator by creating a variant vector with the help of three randomly selected individuals
    - Perform crossover operator by combining the variant vector with the individual. The resultant vector is called trial vector
    - Using a selection mechanism (Binary Tournament, etc.), decide if the trial vector replaces the original individual or not.
  - Next
- While (Not Finished)

Artificial Bee Colony

Forging Process in Honey Bees

- The foraging process begins in a colony by *scout bees* being sent to search for promising flower patches.
- Scout bees move randomly from one patch to another.
- During the harvesting season, a colony continues its exploration, keeping a percentage of the population as scout bees.
- When they return to the hive, those scout bees that found a patch which is rated above a certain quality threshold (measured as a combination of some constituents, such as sugar content) deposit their nectar or pollen and go to the "dance floor" to perform a dance known as the waggle dance.
- http://www.youtube.com/watch?v=-7ijI-g4jHg

Waggle Dance

- This dance is essential for colony communication, and contains three pieces of information regarding a flower patch:
  - the direction in which it will be found,
  - its distance from the hive and
  - its quality rating (or fitness).
- This information helps the colony to send its bees to flower patches precisely, without using guides or maps. This dance enables the colony to evaluate the relative merit of different patches according to both the quality of the food they provide and the amount of energy needed to harvest it.
- After waggle dancing inside the hive, the dancer (i.e. the scout bee) goes back to the flower patch with follower bees that were waiting inside the hive.
- More follower bees are sent to more promising patches.

Behavior of Honey Bee Swarm

- Employed Foragers: They are associated with a particular food source which they are currently exploiting or are "employed" at. They carry with them information about this particular source, its distance and direction from the nest, the profitability of the source and share this information with a certain probability.
- Unemployed Foragers: They are continually at look out for a food source to exploit. There are two types of unemployed foragers: scouts, searching the environment surrounding the nest for new food sources and onlookers waiting in the nest and establishing a food source through the information shared by employed foragers.

Foraging Behavior of Honey Bee

- As soon as a scout discovers a food source, this scout bee is reclassified as an employed bee.
- Onlooker bees watch numerous dances before selecting a food source.
- The probability, through which an onlooker bee selects a food source, is proportional to the nectar content of that food source. Therefore, richer the nectar content of a food source, the more onlookers it attracts.
- Once an onlooker is associated with a food source, it is reclassified as an employed bee.
- Whenever, a food source is completely exhausted, then all employed bees exploiting it leave it and become either scouts or onlookers.

Artificial Bee Colony

- A food source position represents a possible solution to the problem to be optimized.
- The amount of nectar of a food source corresponds to the quality of the solution.
- Onlookers are placed on the food sources by using a probability based selection process.

Artificial Bee Colony (Cont'd)

- As the nectar amount of a food source increases, the probability value with which the food source is preferred by onlookers increases, too.
- If a solution representing a food source is not improved by a predetermined number of trials, then that food source is abandoned and the employed bee is converted to a scout.

Initialization

- $x_{mi}=l_i + rand(0,1) * (u_i - l_i)$

  where l_i and u_i are the lower and upper bound of the parameter x_{mi} , respectively.

Employed Bee Phase

- $v_{mi} = x_{mi} + \phi_{mi}(x_{mi} - x_{ki})$

  where $\vec{x_k}$ is a randomly selected food source, i is a randomly chosen parameter index and ϕ_{mi} is a random number within the range $[-a,a]$. After producing the new food source $\vec{v_m}$, its fitness is calculated and a greedy selection is applied between $\vec{v_m}$ and $\vec{x_m}$.

Onlookers Bee Phase

- After a food source $\vec{x_m}$ for an onlooker bee is probabilistically (fitness selection) chosen, a neighborhood source $\vec{v_m}$ is determined by using equation of the employed bee phase, and its fitness value is computed.
- As in the employed bees phase, a greedy selection is applied between $\vec{v_m}$ and $\vec{x_m}$.

Scout Bee Phase

- Remember that the unemployed bees who choose their food sources randomly are called scouts.
- Employed bees whose solutions cannot be improved through a predetermined number of trials, specified by the user of the ABC algorithm and called “limit” or “abandonment criteria” herein, become scouts and their solutions are abandoned.
- Then, the converted scouts start to search for new solutions, randomly.

Summary of ABC

- Set number of patches (hence employed bees) to 10 and number of onlooker bees to 20.
- Initialize Population consisting of patches (hence population size is 10).
- Evaluate fitness of population members
- Do
  - For each member of the population (representing patch)
    - Perform mutation operator using “Employed Bees” equation (This encourages the employed bees to explore neighborhood of their assigned patch).
  - Next
  - For each member of the onlooker bees
    - Select a patch using Fitness Proportional Scheme.
    - Compute Fitness of the onlooker bees
  - Next
  - From the combined pool of employed and onlooker bees, select p best bees (p should be less than the population size n).
  - Make (n-p) bees as scout bees by randomly initializing them.
- While (Not Finished)

Cuckoo Search Algorithm

- **Cuckoo search (CS)** is an optimization algorithm developed by Xin-she Yang and Suash Deb in 2009.
- It was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds (of other species).
- Some host birds can engage direct conflict with the intruding cuckoos. For example, if a host bird discovers the eggs are not their own, it will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere.
- Some cuckoo species have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry in colors and pattern of the eggs of a few chosen host species

Cuckoo Search Algorithm (Cont'd)

- CS is based on three idealized rules:
  - Each cuckoo lays one egg at a time, and dumps its egg in a randomly chosen nest;
  - The best nests with high quality of eggs will carry over to the next generation;
  - The number of available hosts nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability . Discovering operate on some set of worst nests, and discovered solutions dumped from farther calculations.

Steps of CS Algorithm (Source: Wikipedia)

```
Objective function: f(x),   x = (x_1, x_2, ..., x_d);
Generate an initial population of n host nests;
While (t<MaxGeneration) or (stop criterion)
   Get a cuckoo randomly (say, i) and replace its solution by performing Lévy flights;
   Evaluate its quality/fitness F_i
         [For maximization, F_i ∝ f(x_i) ];
   Choose a nest among n (say, j) randomly;
   if (F_i > F_j),
          Replace j by the new solution;
   end if
   A fraction (p_a) of the worse nests are abandoned and new ones are built;
   Keep the best solutions/nests;
   Rank the solutions/nests and find the current best;
   Pass the current best solutions to the next generation;
end while
```

- The process of cuckoo random selection is done more times than the size of the population.

Levy Search

$$x_i^{(t+1)} = x_i^t \pm \alpha \oplus Lévy(\lambda)$$

- Where α is the step size which should be related
- to the scales of the problem of interest. Levy distribution is computed as

$$Lévy \sim u = t^{-\lambda}, \qquad -1 < \lambda < 3$$

- Instead of using levy distribution, one can also use a standard normal distribution with zero mean and unity standard deviation.