Computational Intelligence

Unit # 13

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.

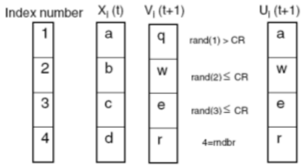
Cross Over

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

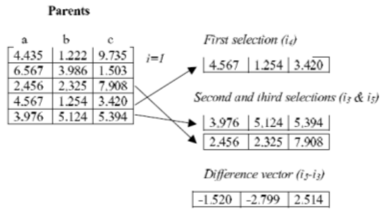
$$U_{jk}(t+1) = \begin{cases} V_{jk}(t+1), \text{ if } (rand \leq CR) \text{ or } (j = rnbr(ind)) \ ; \\ X_{jk}(t), \text{ if } (rand > CR) \text{ and } (j \neq rnbr(ind)). \end{cases}$$

- 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 $X_j(t)$.

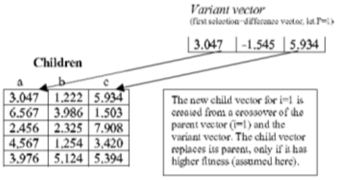
Cross Over (Cont'd)



Mutation and Cross Over



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)