Computational Intelligence

Unit # 4

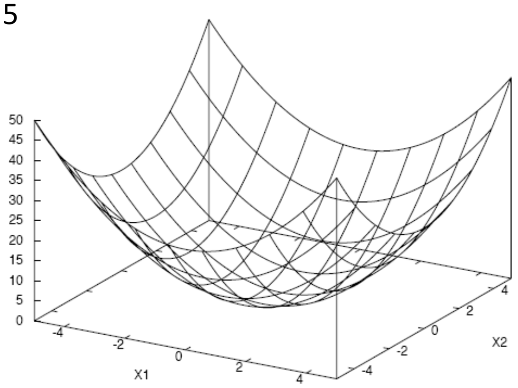
Acknowledgement

- The slides of this lecture have been taken from the lecture slides of "CSE659 – Computational Intelligence" by Dr. Sajjad Haider.

Assignment # 1 (Function 1)

$f(x, y) = x^2 + y^2$

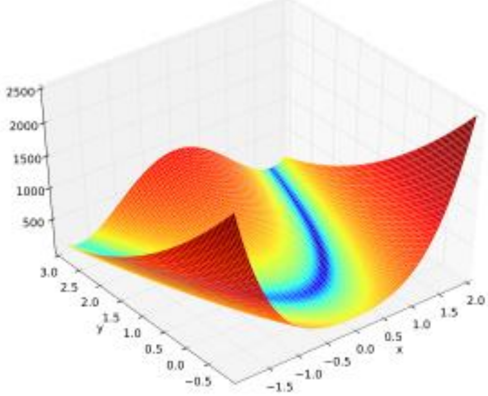
$-5 \le x, y \le 5$



Assignment # 1 (Function 2)

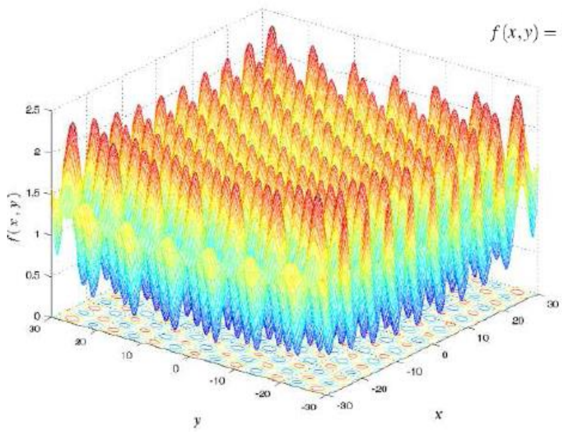
$$f(x, y) = 100 * (x^2 - y)^2 + (1 - x)^2$$

$-2 \le x \le 2, -1 \le y \le 3$



Assignment # 1 (Function 3)

$$f(x,y) = \frac{x^2 + y^2}{4000} - \cos(x)\cos\left(\frac{y}{\sqrt{2}}\right) + 1$$



A Typical Evolutionary Algorithm Cycle

- Step 1: Initialize the population randomly or with potentially good ***solutions***.
- Step 2: Compute the ***fitness*** of each individual in the population.
- Step 3: Select parents using a ***selection procedure***.
- Step 4: Create offspring by ***crossover*** and ***mutation*** operators.
- Step 5: Compute the ***fitness*** of the new offspring.
- Step 6: Select members of population to die using a ***selection procedure***.
- Step 7: Go to Step 2 until termination criteria are met.

Assignment Implementation Details

- Let's pick the population size to 10. So initialize 10 individuals of the form (x, y) randomly.
- Compute their fitness.
- Using one of the parent selection scheme, generate 10 offspring (using crossover and mutation). For mutation, you can use +/- 0.25. Remember that mutation is not applied to each gene. So you need to make it probabilistic as well.
- Compute fitness of the offspring.

Assignment Implementation Details (Cont'd)

- Now you got 10 parents and 10 offspring (20 individuals.
- Using one of the survival selection scheme, pick 10 individuals that survive to the next generation. Discard the remaining individuals.
- Store best survived individual's fitness and average fitness of the whole survived individual.

Assignment Implementation Details (Cont'd)

- Suppose you run this process for 40 generations. You will have recorded the following observations.

Generation #	Best Fitness	Average Fitness
1		
2		
..		
..		
..		
40		

Average Best-So-Far

- You need to repeat this exercise (for a single combination of parent and survival selection schemes) 10 times).
- Each run will start with new randomly initialized population.

Generation #	Run # 1 BSF	Run # 2 BSF	..	..	Run # 10 BSF	Average BSF
1						
2						
..						
..						
..						
40						

Average Average-Fitness

- You need to repeat this exercise (for a single combination of parent and survival selection schemes 10 times) for average fitness values.
- Each run will start with new randomly initialized population.

Generation #	Run # 1 Avg. Fit.	Run # 2 Avg. Fit.	..	..	Run # 10 Avg. Fit.	Average Avg. Fit.
1						
2						
..						
..						
..						
40						

Assignment Implementation Details (Cont'd)

- Once you have gotten (a) average best-so-far values and average average-fitness, you need to plot the values against generation # (separate graphs).
- The scheme described in the previous slides will be repeated for each combination of parent and survival selections, i.e.,
  - FPS and Truncation
  - RBS and Truncation
  - Binary Tournament and Truncation
  - FPS and Binary Tournament
  - RBS and Binary Tournament
  - Binary Tournament and Binary Tournament

Performance Indicator

- **Best-so-far (BSF) -** We record the best solution found by the algorithm thus far for each generation in every run.
- **Average-of-current-population (ACP) -** We record the average solution in each generation in every run.
- **Worst-of-current-population (WCP) -** We record the worst solution in each generation in every run.

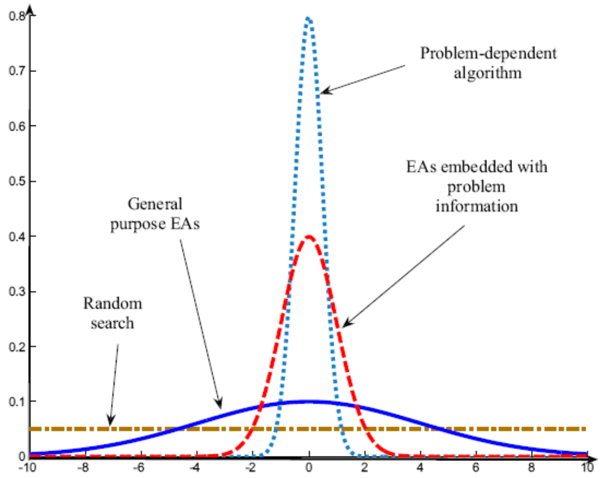
No Free Lunch Theorem

- Wolpert and Macready published a paper with a very strong title: “No Free Lunch Theorems for Optimization”. The key contents of the paper can be quoted as follows:
  - For both static and time dependent optimization problems, the average performance of any pair of algorithms across all possible problems is identical.

NFL Theorem (Cont'd)

- The more we understand the problem, the more specific technique we could design for solving it, and the better performance it will have, but the less robust it will be for other problems.
- We need to demonstrate that our algorithms are better than random search on the problem we face.
- General purpose EAs are reliable methods when you are doing a blind or near blind search in most cases.
- If problem information could be embedded into the encoding and decoding process and into operators, together with a problem-dependent local search method, the performance of the algorithm would be improved at the expense of lower adaptability for other problems.

One Possible Interpretation of NFL Theorem

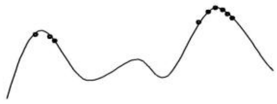


Typical behaviour of an EA

Phases in optimising on a 1-dimensional fitness landscape

Early phase:
quasi-random population distribution

Mid-phase:
population arranged around/on hills



Late phase:
population concentrated on high hills