Sarthak Patidar (204103420)

Introduction:

Generic algorithm (GA) is population based search and optimization technique which works based on the mechanism of natural genetics and Darwin's principal of natural selection.

Initial solution is not single. It may be 50, 100 or more. In population based search number of solutions are searched at a time. In a GA solutions are represented in a coded form like chromosomes, similarly in GA solutions consists of number of genes called as string which can consist of real or binary number.

We used binary coded GA, the string here only consist of 0 and 1. The length of string depends on the accuracy of desired solution.

Methodology:

The problem given is a minimisation problem, so GA works only with maximisation so converted minimisation to maximisation by using (1/(f(x)^2). The initial population array is generated by using numpy random function. The string length considered is 24 with s1 of size 12 and s2 of 12. The generated population is break down in length of corresponding constrain and the binary string is converted to a decimal number, This decoded value is used to calculate corresponding variable value here it is x1 and x2.

The array consist of x1 and x2 is send to fitness function to get corresponding fitness value of the generation. This fitness value is then send to reproduction function where Roulette wheel reproduction is used, the probability of each solution is calculated and cumulative sum is calculated for each solution and new array is made and n (population size) random numbers are selected between 0 and 1 and corresponding solution is selected in mating pool. This mating pool is then feed to crossover function for crossover where two point crossover method is used in which n/2 pairs of solution are selected for each pair a random number is generated between 0 and 1 if it is smaller than Pc then crossover is performed on that pair in which two random number is selected and the string between those two numbers is swapped with the corresponding pair solution and for each pair two children solution is prepared and new mating pool is obtained. This new mating pool is feed to mutation function where bitwise mutation operation is performed in each solution for each bit a random number is generated if it is less than Pm then mutation is done on that bit i.e. if it is 1 then it is converted to 0 and vice-versa. This new mutated mating pool is our new population for next generation.

Result:

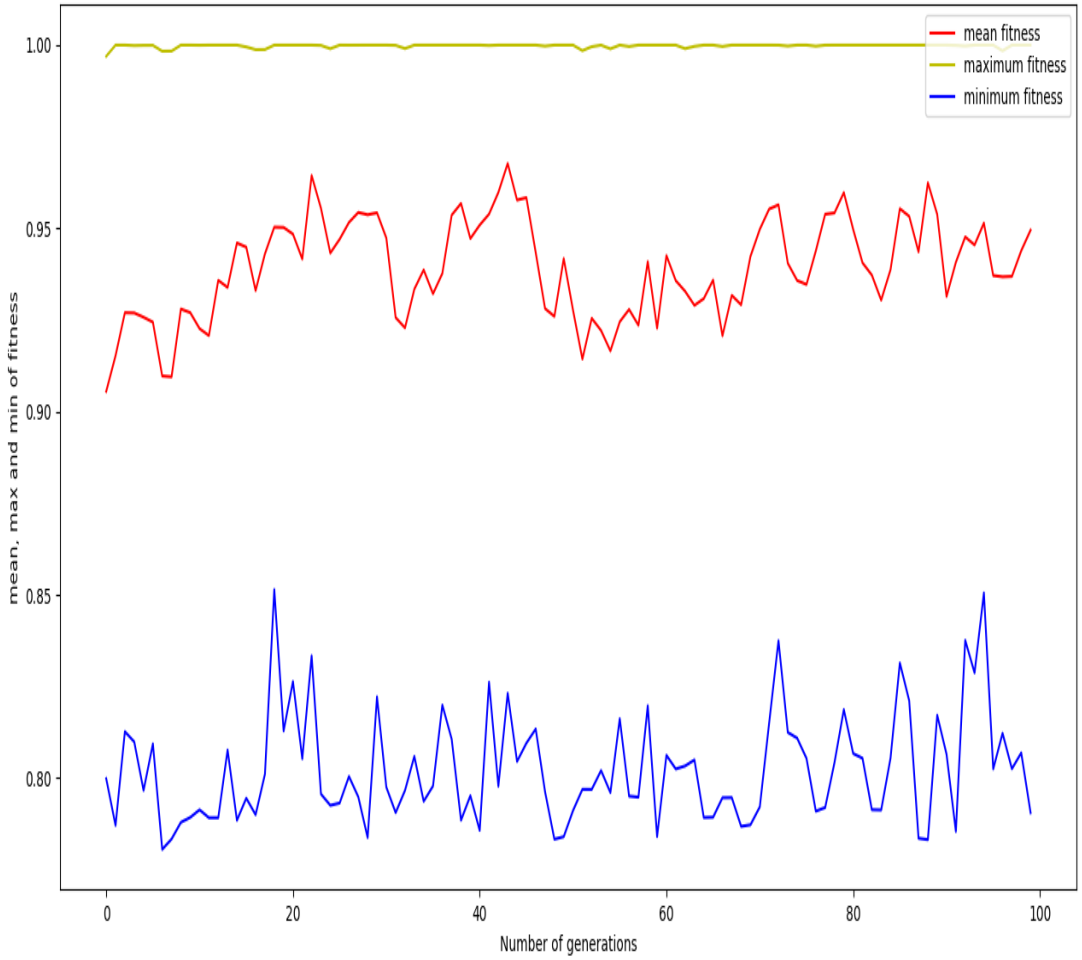
The objective function is x1 + x2 – 2*x1^2 - x2^2 + x1*x2

Number of generations = 100

Population size = 50

Crossover probability = 0.9

Mutation probability = 0.1



Conclusion:

After 100 generations the final optimal solutions is x1 = 0.0 and x2 = 0.0 and another optimal solution is x1 = 0.5 and x2 = 0.0.