FFR105
Stochastic Optimization Algorithms
HP2

Manish Suvarna + smanish

October 14, 2020

Problem 2.1: The Traveling Salesman Problem

a

The salesman can choose to start from N different cities as the starting point for his path which makes the possible number of paths to travel as $N!$ which in its expanded is as shown in Equation 1. This gives the number of equivalent paths there are. For distinct paths, it does not matter if the salesman starts his journey from city1 or city 2 or city 3 or city N in general, which makes the number of available options wrt the distinct paths as $\frac{N!}{N}$ as shown in Equation 2.

$$N! = N \cdot (N-1) \cdot (N-2) \cdot (N-3)! \tag{1}$$

$$\frac{N!}{N} = (N-1)! \tag{2}$$

As these $(N-1)!$ paths can be traced in both the directions, i.e forwards and backwards by the salesman, the number of distinct paths in one particular direction for the salesman will then simply be $\frac{(N-1)!}{2}$.

b

The code for GA for the TSP can be found in MATLAB script named as `GA21b.m`. Since GAs are stochastic global search algorithms, each individual of the population performs different trajectories across the search space during the execution. Thus the population converges to multiple sub optimal solutions which requires one to run this algorithm multiple times in order to get best result out of it.

c

The Ant Colony Optimization (ACO) algorithm can be found in the MATLAB script file named as `AntSystem.m`.

d

The nearest-neighbour path length given by the function `NNPathlLengthCalculator.m` with the starting city index of 28 (which is selected randomly by this function) was 128.94. On the other hand, the best path length achieved by the GA after running for 10000 generations for this TSP was around in the range of 135-165. As mentioned in section (b), due to its tendency to getting stuck at various local optima, the GAs give sub-optimal solutions, such as in this case. Hence, relying on ACO for solving the TSP seems to be a better option as they employ the method of finding the nearest-neighbour path length as shown in the `NNPathlLengthCalculator.m`.

e

The plots of the path given by the GA and ACO are as shown in the Figures 1 and 2 respectively. As mentioned in the figure, the shortest path length given by GA and ACO are 156.88 and 121.8 respectively. Whereas the nearest neighbour path length given by `NNPathLengthCalculator.m` is 142.93 with city number 29 as the starting city. Clearly, the shortest pathlength here is given by ACO and the corresponding path recommended by ACO can be found by running the `BestResultFOund.m` MATLAB file.

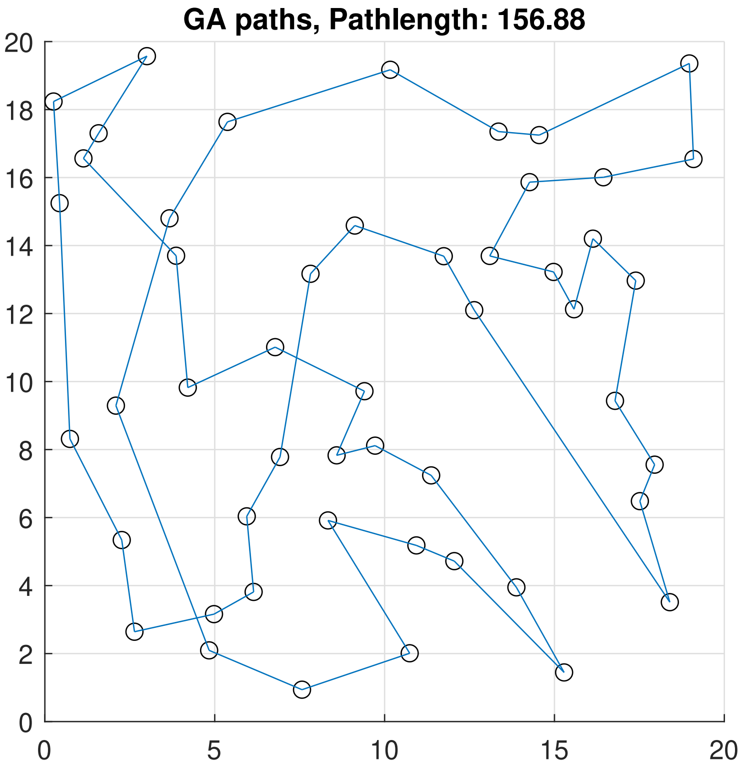

Figure 1: Path given by GA

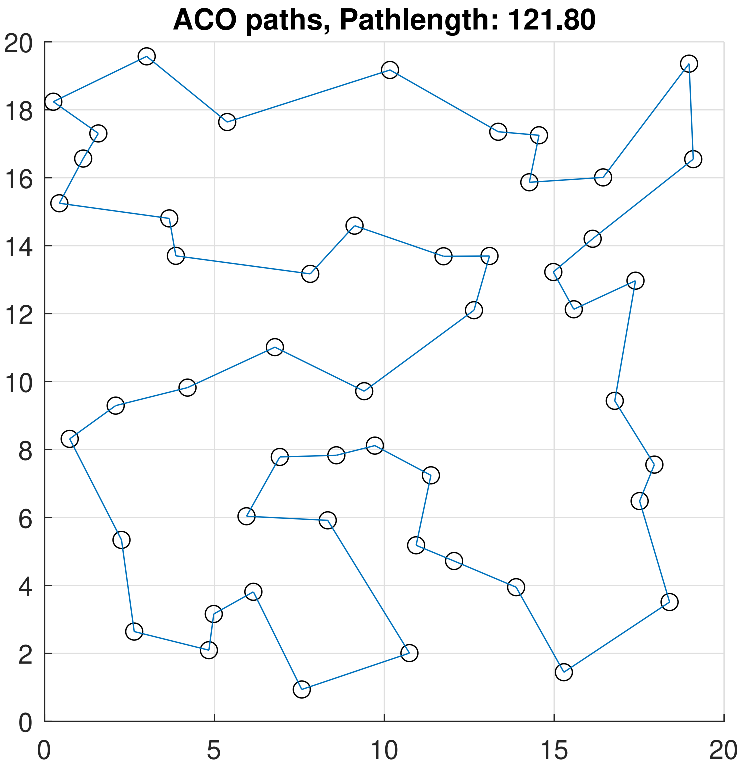

Figure 2: Path given by ACO

Problem 2.2 Particle Swarm Optimization

The given function happen to have 4 minima as shown in it's contour plot in Figure 3. These minima were found by implementing the Particle Swarm Optimization which can be found in the MATLAB file named `PSO.m`. The number of boids used for this optimization were set to 40 and was run for 1000 iterations. The inertia weight was set at 2 which varies as the iterations proceed by a factor of 0.001. The minima found by this algorithm is as shown in the Table 1

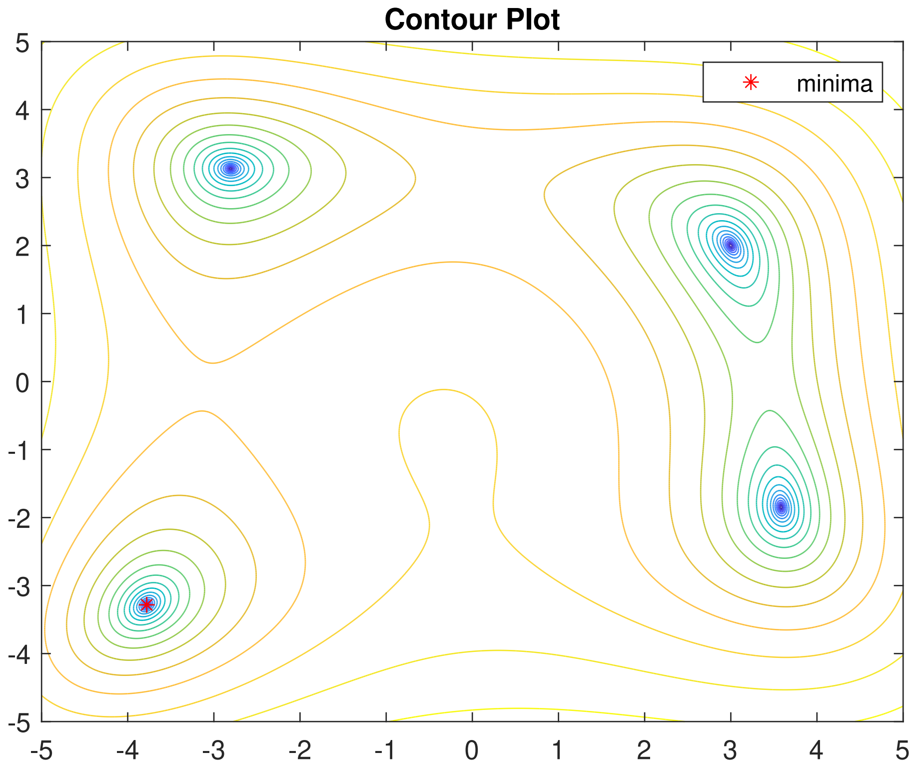

Figure 3: Contour plot for the function $f(x, y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2$, along with one of its 4 minima

Table 1: Table with 4 minima

x	y	f
3	2	0
-2.81	3.13	0
3.58	-1.85	0
-3.78	-3.28	0

Problem 2.4: LGP

The best fit to the given dataset was found on running the LGP for 6 variable registers and 1 constant set equal to 1 and the algorithm was executed for 50000 iterations. Since the initialized population has varying length of chromosomes in each individual of the population, an average of these lengths were computed whose inverse value was set as the mutation rate for that particular population. The idea was to have the algorithm explore various trajectories for the first 80 percent of the generations and become deterministic there onwards, hence the mutation rate is varied for the first 80 percent of the generations and has no variation in the mutation rate there onwards. The chromosomes violating the limit of maximum number of instructions i.e 50 were penalized. On execution, the estimated function given by this algorithm is as shown in the Equation 3 for which the function fit plot is as shown in the Figure 4 with an error value of $2.79 * 10^{-9}$ respectively.

$$g(x) = \frac{x^3 - x^2 + 1}{x^4 - x^2 + 1} \tag{3}$$

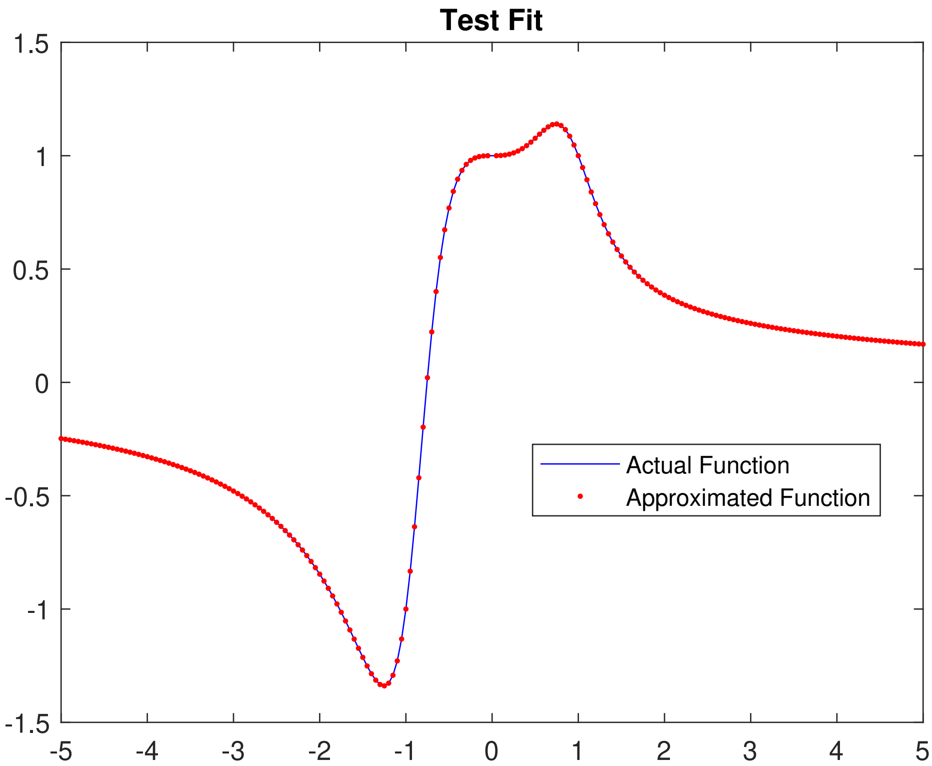

Figure 4: Result found by the LGP program