Assignment 2

1 NEURAL NETWORKS

I used Bank Marketing problem from Assignment 1. With one hidden layer and size of the hidden layer equal to 5, I had shown that the training error was 9.0116% and testing error was 9.1425%. The total number of weights is equal to 260. The cost function used was Standard minimization of the sum of the squared distances. Finally I evaluated the performance of the neural network, from test data used in earlier assignment. The weights were randomly initialized using MATLAB function rand.

Following were the results obtained

Neural Nets Algorithm	Function Value	Time	Testing Error	Functions Evaluated
Random Hill Climbing	64.9721	429.444	11.8853	4691
Simulated Annealing	72.5193	409.2656	11.8853	4577
Genetic Algorithm	64.5561	98.040131	11.8853	1040

Test Error obtained was exact same for all the algorithms. This might be because the objective function values were very close and hence the classifiers might be very similar. They performed equally well for all the algorithms. The objective function value was less for GA, hence training error might be less. SA evaluates more functions then GA, but RHC requires maximum evaluations. Gradient descent performed better than all the algorithms tried. This might be because of no cross-validation used while performing, RHC, SA and GA

2 Optimization Problems

Following are the three optimization problems on which I applied all four local random search algorithms. Rastrigin Function and TSP is minimization problem while, 4 Peaks is maximization problem. Both Rastrigin and TSP can converted to maximization by changing the sign of the function.

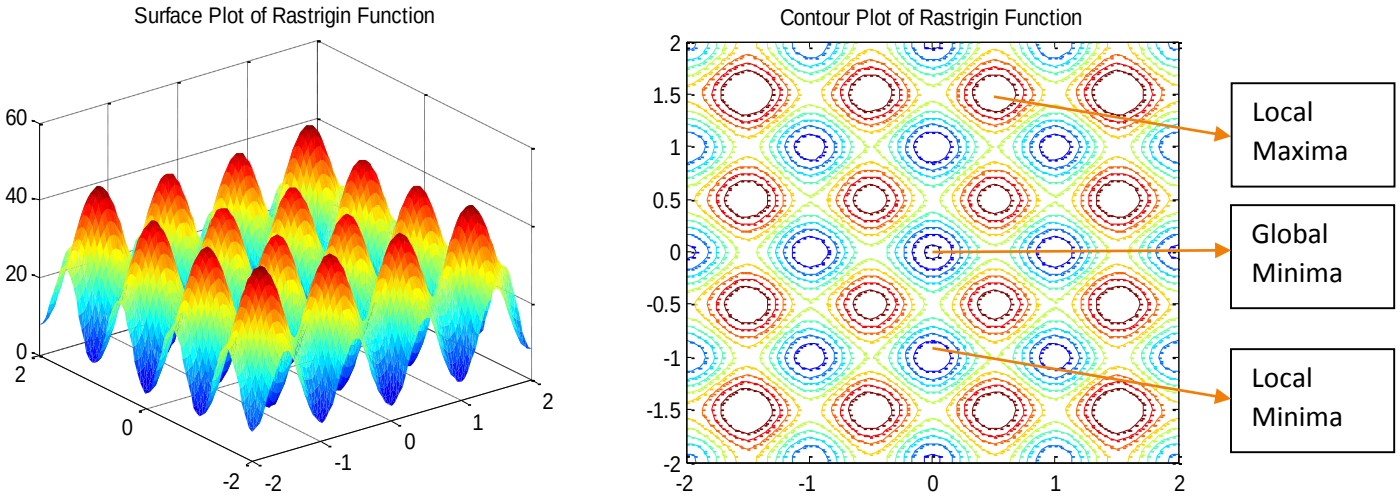
2.1 Rastrigin Function

The Rastrigin function is a non-convex function used as a performance test problem for optimization algorithms. For two independent variables, Rastrigin function is defined as

$$Ras(x) = 20 + x_1^2 + x_2^2 - 10(cos2\pi x_1 + cos2\pi x_2)$$

Following figure shows the surface and contour plots of the function

s



As we can see from the surface plot, finding the minimum of this function is a fairly difficult problem due to its large search space and its large number of local minima. Algorithms which don't exploit their local search space are likely to get stuck at any of the local minima. The function has just one global minimum, which occurs at the point [0 0] in the x-y plane, where the value of the function is 0. At any local minimum other than [0 0], the value of Rastrigin function is greater than 0. The farther the local minimum is from the origin, the larger the value of the function is at that point. Although this is a minimization problem it can be changed to a maximization problem by just reversing the sign of the function. Representation of x as binary vectors, makes this problem a discrete one; which is basically a continuous problem.

2.2 Travelling Salesman Problem

As Wikipedia says, travelling salesman problem (TSP) asks the following question:

Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?

Following are some of the applications of TSP

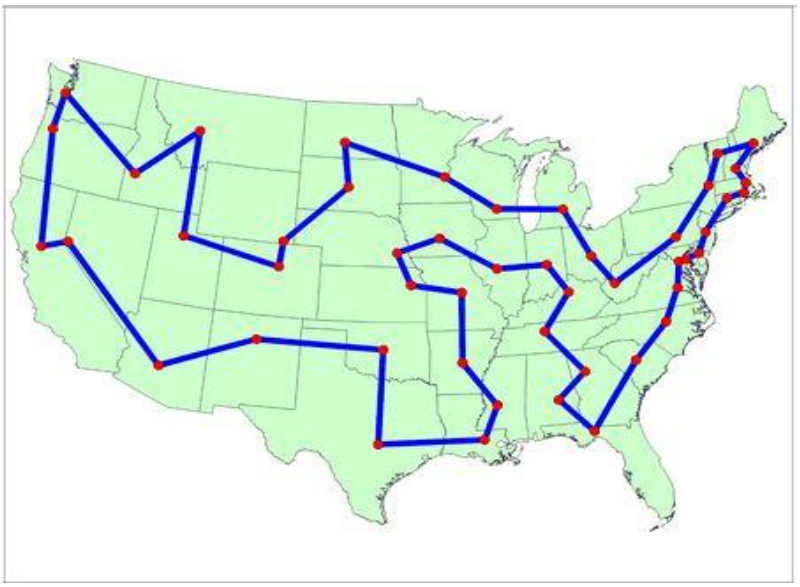
- Planning routes between cities

- Logistics companies
- Manufacture of microchips
- Wiring, and
- DNA Sequencing

In these applications, the concept city represents, for example, customers, soldering points, circuit nodes or DNA fragments, and the concept distance represents travelling times or cost, or a similarity measure between DNA fragments.

For the purpose of this assignment I considered the example of TSP through US capital cities. Given n cities, if we don't consider the direction in which we are travelling there will be $(n-1)!/2$ possibilities for a round trip.

Following is the optimal route and was taken from [1]



From [1] it was proved that the optimal route was from Montgomery to Atlanta, and the total mileage needed to optimally traverse the capital cities is 33,523.75 miles.

This problem is NP- hard.

For n= 48, number of possible routes are 1.2931e+59 and it becomes difficult to find the most optimum one.

2.3 Four Peaks Problem

The four peaks problem was taken from the reading [2]. Given an N- dimensional vector X, the four peaks evaluation function is defined as:

$$f(X,T) = \max[tail(0,X), head(1,X)] + R(X,T)$$

where

$$tail(b,X) = number\ of\ trailing\ b's\ in\ X$$

$$head(b, X) = number\ of\ leading\ b's\ in\ X$$

$$R(X,T) = \begin{cases} N & if\ tail(0,X) > T\ and\ head(1,X) > T \\ 0 & otherwise \end{cases}$$

There are two global maxima for this function,

- When there are T+1 leading 1's followed by all 0's
- When there are T+1 trailing 0's preceded by all 1's

And there are two local maxima when

- All 1's
- All 0's

For example, let N = 40 and T be 10% of N. Therefore the value of function at global maxima will be 35+40 = 75 and that at local maxima will be 40 + 0 = 40. For purpose of this assignment T was always 10% of N

3 IMPLEMENTATION

3.1 RANDOM HILL CLIMBING

I used MATLAB to implement this algorithm. Along with MATLAB's Global Optimization toolbox, function *patternsearch* was used. At each step, *patternsearch* searches a set of points, called a mesh (Neighbours), for a point that improves the objective function. *patternsearch* forms the mesh by

- Generating a set of vectors $\{d_i\}$ by multiplying each pattern vector v_i by a scalar ∂m. ∂m is called the mesh size.
- Adding the $\{d_i\}$ to the current point—the point with the best objective function value found at the previous step.

Starting points were chosen at random. For generating neighbors, I used generalized pattern search (GPS) algorithm. I ran the algorithm using two polling schemes – one that stops after finding the neighbor whose function value is less than the current, and one that uses complete polling.

3.2 SIMULATE ANNEALING

I used MATLAB to implement this algorithm. MATLAB function *Simulannealbnd* was used and exponential temperature update was used.

3.3 GENETIC ALGORITHMS

MATLAB function *ga* was used. Default crossover fraction of 0.8 was used. I used Global Optimization toolbox.

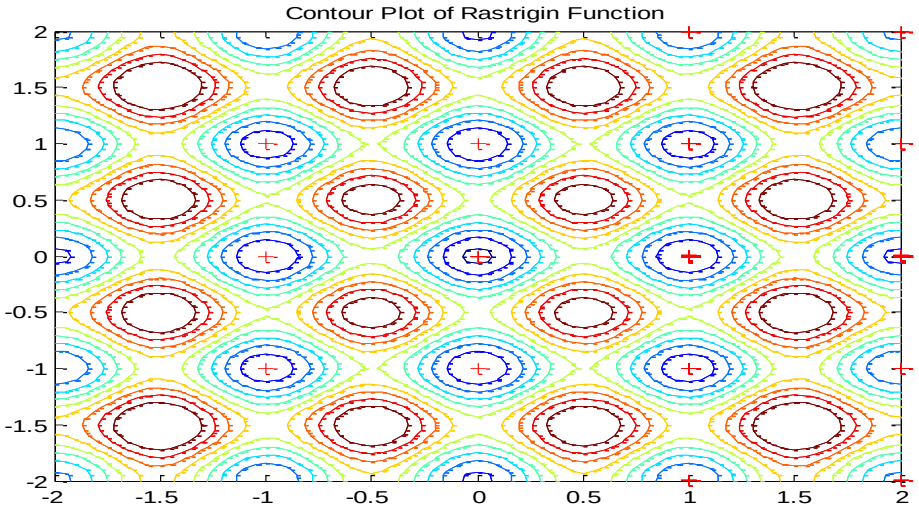
3.4 MIMIC

ABAGAIL's implementation was used.

4 RESULTS

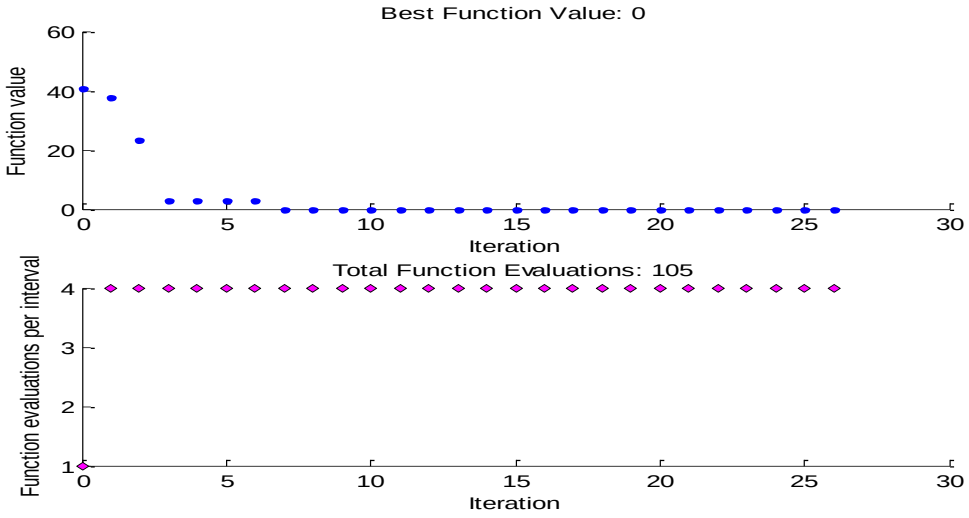
4.1 RASTRIGIN FUNCTION

I ran RHC for large number of starting points and as expected the algorithm gets stuck at local minima. The output depends on the starting point. This happens if the mesh size (Neighboring jump size) is less and is expected. If I increase the mesh size, the algorithm manages to find the global minima.



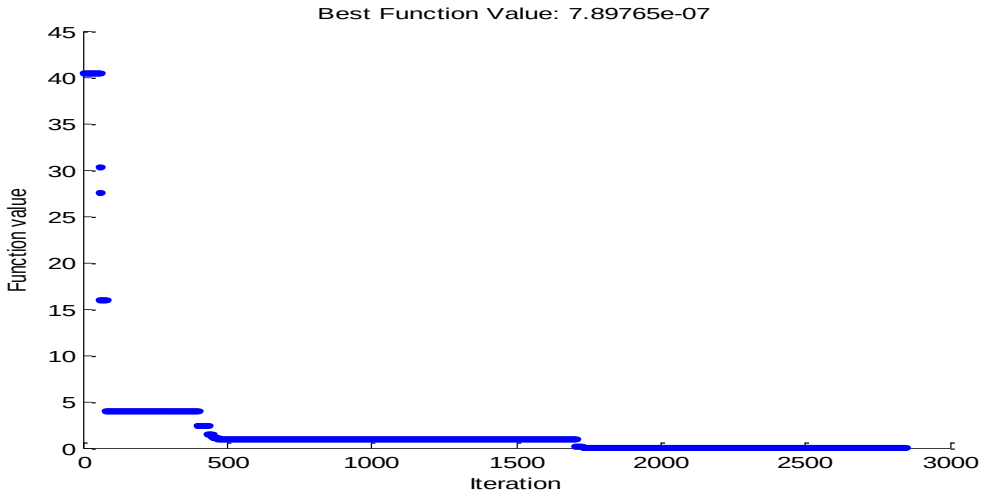
The red marks are the best points obtained. As we can see RHC gets stuck at local minima. It does find global optima but it depends upon the starting point.

Following is the plot of total number of function evaluations and function values versus iterations where starting point was [0.5, 0.5] and mesh size was 0.25 and complete polling was on. This is the case when RHC finds global minima



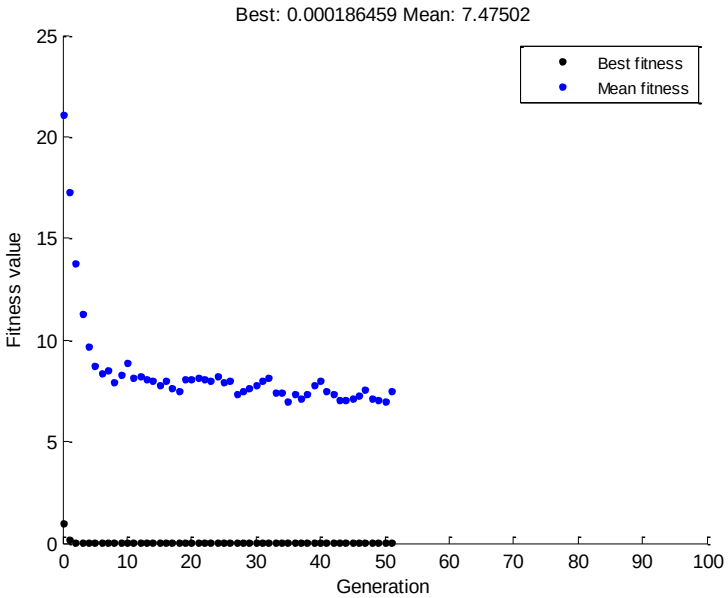
Complete polling requires more number of evaluations and is expected. I ran the algorithm for same initial starting point with complete polling off, and found out it that function evaluations were 109, instead of 105. Also the number of iterations required when complete polling was off was 30 as compared to 26 as seen in above graph. This should be the case, since complete polling selects the best neighbor after evaluating all the neighbors and hence requires more f-evals. Graph was almost the same & hence omitted

Following is the plot for simulated annealing. Same starting point was used. The best function value was 7.89 * 10^-7, which is very close to 0. Initial temperature was set to 100 and exponential temperature update was used.



Performance of simulated annealing was comparable to the RHC, but the local search space of RHC is less as compared to Simulate annealing. This was evident from the fact that, when I changed the initial starting point to [1, 1] RHC gave me the function value of 1.9899, while simulated annealing gave 5.888E-7. I further increased the starting point to [2, 2] to see whether simulated annealing always finds the global minima, but this was not the case.

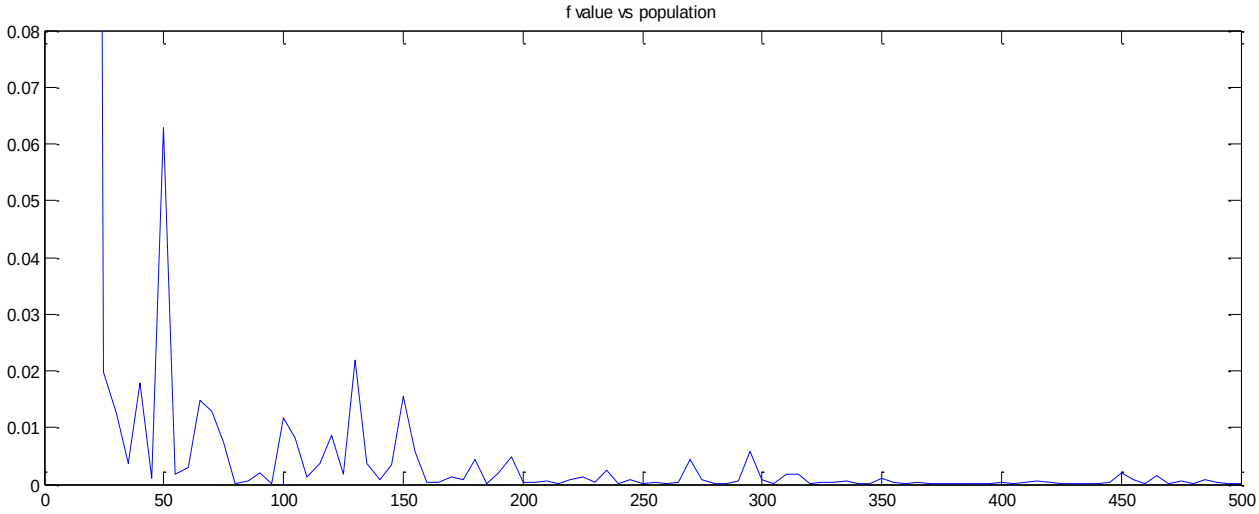
For Genetic algorithms, following is the plot when starting population is 500. As the population goes on increasing, genetic algorithms perform well and is shown in below plot of best value of function versus population. Notice the graph goes on decreasing as population goes on increasing.

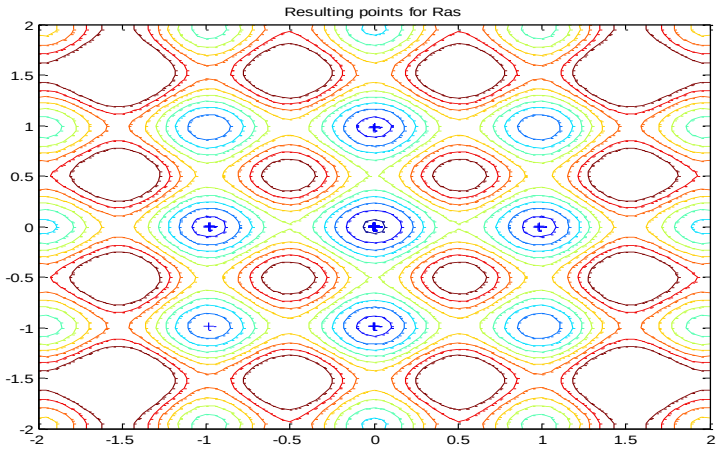


Algorithm	elapsed time
RHC (Complete Polling - OFF)	0.523948
RHC (Complete Polling - ON)	0.577599
GA	0.129014
SA	19.715042
MIMIC	18

Genetic algorithms took less time as compared to SA, but the function value

obtained was 1.864591560334361E-4 for GA as compared to 7.89E-7 of SA. So GA had more error than SA.





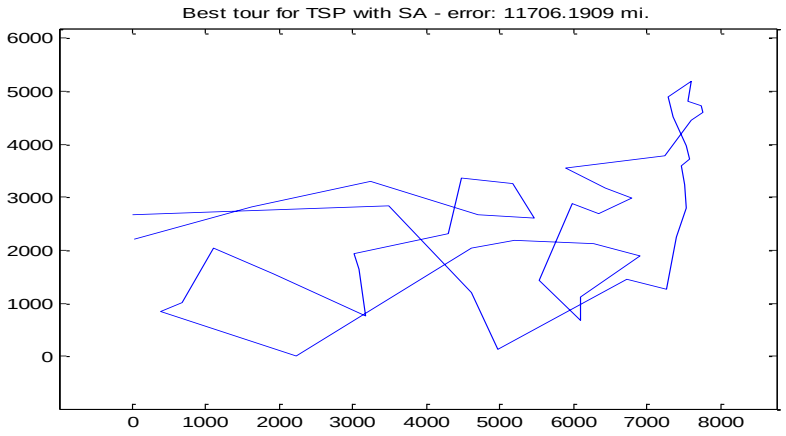
MIMIC was performed using ABAGAIL and its best points are plotted. The problem of local minima is still there in MIMIC and can be seen in figure plotted

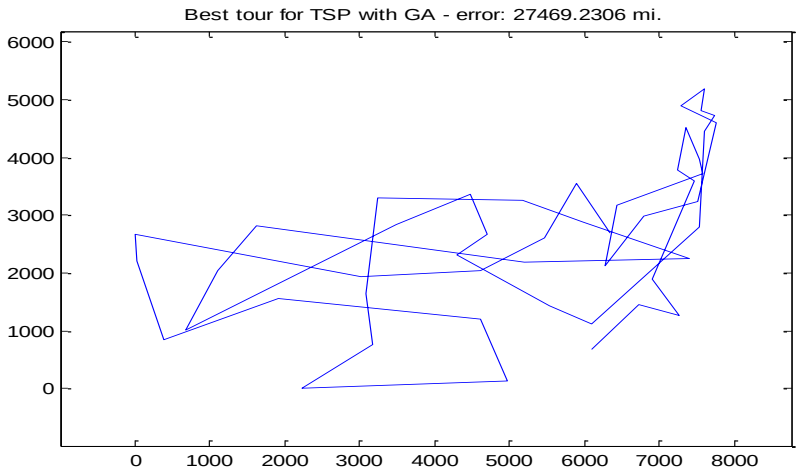
4.2 TRAVELLING SALESMAN PROBLEM

I ran all the algorithms for 10 times. Likewise Rastrigin's function, GA was run for varying population sizes varying from 5 to 50. Following are the results obtained

Algorithm	elapsed time	mean f-evals	average f-val
RHC (Complete Polling - OFF)	1407.995	4534	17598.3194
RHC (Complete Polling - ON)	1422.523	4248	17270.1803
GA	365.613	4112	47634.3624
SA	61.9431	6909.4	17932.7313
MIMIC	81	20000	33076.9576

The objective function is the distance to the true minimal function value given by the optimal tour and hence for optimal solution f-val should be evaluated to zero. Clearly from the results, none of the algorithms perform very well, the route they compute is in average 50% too long and is plotted for SA and GA.

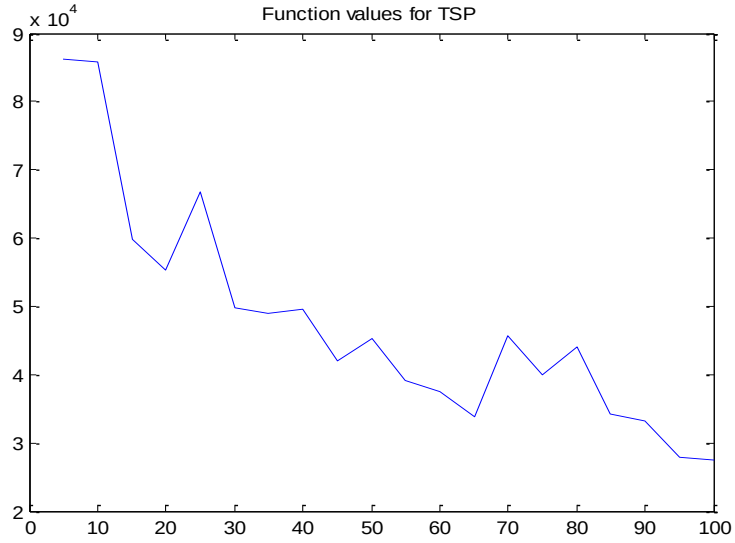




Unlike with Rastrigin's function the TSP problem is not smooth. Due to its 'discontinuities' MIMIC and GA perform badly.

RHC and SA perform equally well. I think both algorithms perform pretty good, cause they explore a huge search space and are likely to find good routes. Although RHC performs equally well to SA, it is way slower. Also RHC expands its search radius, it stays in its current neighborhood and searches within it. The restart from a different random point may change this. SA evaluates not only points in the current neighborhood but also other points. That might be why it finds a 'good' route faster than RHC

Like with Rastrigin's function GA improves with increasing population and is plotted below. MIMIC was performed using ABAGAIL



4.3 Four Peaks Problem

I performed this experiment using the code implemented in ABAGAIL. I performed for N = 40 and N = 100. For both the cases, T was 10% of N. Following were the results obtained

	for N = 40, T = 4		for N= 100, T = 10		
Algorithm	**elapsed time**	**f-val**	**elapsed time**	**f-val**	**Iterations**
RHC	56	40	89	100	200000
SA	167	75	198	100	20000
GA	824	74	899	143	10000
MIMIC	1612	75	5230	189	1000

For N = 40, global maxima is at 75 and local maxima is at 40. While for N = 100, global maxima is at 189 and minima is 100. Clearly MIMIC performs better. RHC and SA get stuck at local maxima. For both the cases MIMIC finds global maxima, although it takes lot of time. SA and RHC only check neighbors and hence their search area is less, falling to the local maxima. GA performs better than RHC and SA, but requires more time. GA does crossover and mutations, hence has lot better choices while selecting neighbors. Lastly, MIMIC performs the best. It does well with the structure problems and was evident here. Although MIMIC requires fewer iterations, time required is more. This is expected. MIMIC can be used where the evaluation of cost function is costly and hence number of iteration is an important factor.

5 Conclusion

Random Hill Climbing gets stuck at local minima and was evident from all three examples. Complete polling requires more evaluation but performs better. Simulated Annealing performs better than RHC as it not only exploits but also explores. Simulated Annealing does not always find global minima. Genetic Algorithms perform better when population goes on increasing. They perform well for structured problems, because its random crossover operation is able to capture some of the underlying structure. Genetic algorithm performed worst for TSP, because of the discontinuities of TSP's function. MIMIC performed best for Four Peaks Problem as it could model the structure perfectly. MIMIC requires few iterations, but every iteration takes more time. Thus, it can be used when evaluating a cost function is high.

6 References

[1]http://support.sas.com/documentation/cdl/en/ornoaug/65289/HTML/default/viewer.htm#ornoaug_optnet_examples07.html

[2]De Bonet, Jeremy S., Charles L. Isbell, and Paul Viola. "MIMIC: Finding optima by estimating probability densities." Advances in neural information processing systems (1997): 424-430.