Genetic Algorithm: An Approach for Optimization (Using MATLAB)

Subhadip Samanta
Department of Applied Electronics and Instrumentation Engineering.
Greater Kolkata College of Engineering and Management
Kolkata, West Bengal, India

Abstract: In this paper we have gone through a very brief idea on Genetic Algorithm, which is a very new approach for problems related to Optimization. There are many techniques used to optimize a function but in case of optimizing Multimodal Functions most of these techniques face a common problem of robustness. This can be overcome by using Genetic algorithm. Through this paper we will learn how the Genetic Algorithm actually works with proper explanation and with some real time examples based on MATLAB.

I. INTRODUCTION

Nature follows a very interesting path to select an optimum solution of any problem. By generations it chooses and keeps the best and fittest solutions as survivor and discards the rest other options. The fittest solutions again evolve continuously as the test ground of the nature gets more difficult. This is a continuous and almost flawless process to find out the most optimum result. We all know this process under the heading "Survival of the Fittest". In the year 1975 a person named John Holland, from the University of Michigan, first attempted to apply this natural selection technique to optimize artificial problems. His goal was to find out one robust technique to optimize all kind of functions, one robust technique which can provide global optimization to any function. His approach was the building steps of Genetic Algorithm.

II. CONCEPT

The Genetic Algorithm is an example of a search procedure that uses a random choice as a tool to guide a highly exploitative search through a coding of a parameter space.
The principle and procedure of Genetic algorithm can be summarized under the following,

1. Outline of the Algorithm.
2. Initial population.
3. Creating the Next Generation.
4. Plots of Later Generation.
5. Stopping Condition for the Algorithm.

Let's have a brief idea on each section.

1. Outline of the Algorithm: The following outlines summarized how the Genetic Algorithm works.
   a. The algorithm begins by creating a random initial population.
   b. The algorithm then creates a new generation or population by using the individuals of the current population.
   c. To create new generation the algorithm performs crossover and mutation in between the individuals of the current generation.
   d. Replace the current generation by the children of the next generation.
   e. The children of the next generation are chosen on the basis of their best fitness value.
   f. The algorithm stops when one of the stopping criteria is met.
2. Initial Population: The algorithm begins by creating a random initial population for a given problem.
3. Creating Next Generation: At each step, the genetic algorithm uses the current population to create the children that makes up the next generation. The algorithm selects a group of individuals in the current population, called 'Parents', who contribute their genes (The Entries Of Their Vectors) to their children. The parents are selected on the basis of 'Fitness' to create better children. Genetic Algorithm creates three types of children, which are,

a. Elite Children: The individual in the current generation with best fitness value will automatically survive to the next generation.

b. Crossover Children: The children created by combining the genes from a pair of parents selected on count of Fitness.

c. Mutation Children: The children created by introducing random change or mutation in the gene of a single parent.

As the number of generation increases the average fitness value of the entire generation increases and the individuals in the population get closer to the global minimum point [0,0] of optimization.

4. Plots of Later Generation: As the algorithm goes on the graphical plot of each successive generation shows the increasing closeness of the outcome to the global optimum point.
5. Stopping Condition for the Algorithm:

   The Genetic algorithm uses the following five conditions to determine when to stop.

   i. Maximum Number of Generation.
   ii. Maximum Time Limit.
   iii. Maximum Fitness limit.
   iv. Stall Generations.
   v. Stall Time Limit.

   The Algorithm stops as soon as any one of these five conditions met.

III. TECHNIQUES

There are two ways we can use the Genetic Algorithm in MATLAB (7.11.0) for optimization.

1. Calling the Genetic Algorithm Function 'ga' at the command line.
2. Using the Genetic Algorithm Tool, a graphical interface to the genetic algorithm.

Let's have a brief idea on both.

1. Calling the Genetic Algorithm Function: To use the Genetic Algorithm at the command line, call the algorithm function 'ga' with the below mentioned syntax

   [x fval]=ga(@fitnessfun, nvars, options)

   Where,

   - '@fitnessfun' is a handle to the Fitness Function.
   - 'nvars' is the number of independent variables for the Fitness Function.
   - 'options' is a structure containing options for the genetic algorithm.

   The syntax returns the following things.

   - fval – Final Value of the Fitness Function.
   - X – Point at which the final value is attained.

   Additional output Arguments: To get more information about the performance of the algorithm, call 'ga' with the following syntax.

[x fval reason output population scores] = ga(@fitnessfun, nvars)

Including 'x' and 'fval' the syntax returns the following,

- reason – Reason for which the algorithm terminates.
- output - Structure containing information about the performance of the algorithm.
- population – Final population.
- scores – Final scores.

Setting Options: We can create the options structure by using the function,

options = gaoptimset

This returns the structure of options with the default values for its fields. To edit an option structure with a different value of an option such as 'Elite Count', we can use the function-

options = gaoptimset('Elite Count', 1)

This will return an options structure with an 'Elite Count' of 1 instead if its default value 2.

2. Using The Genetic Algorithm Tool: The genetic algorithm tool is a graphical user interface that enables us to use the genetic algorithm without working at the command line. To open this tool, enter,

   'gatool'

This will open the gatool window. To use this tool we must first enter the following information,

- Fitness Function- The objective function that we want to minimize. Enter the fitness function in the form of '@fitnessfun', where 'fitnessfun.m' is an M file that computes the fitness function.
- Number of variables- This is the length of the input vector of the fitness function.

To run the genetic algorithm, click the 'start' button. The tool displays the result of the optimization process in the 'Status and Result' pane.

The 'plot' pane enables us to display various plots that provide information about the genetic algorithm while it is running. This information can help us to improve the performance of the algorithm.

Example:

- Writing an M-File

  $f(x_1, x_2) = x_1^2 - 2x_1x_2 + 6x_1 + x_2^2 - 6x_2$

  The M-File that will compute the above function must accept as row vector x of length 2, corresponding to the variables x_1 and x_2, and return a scalar equal to the value of the function at x. To write the M-file write the following steps,

  1. Select and open a new M-File.
  2. In the M-File, enter the following lines of code

     ```
     function z = my_fun(x)
     z = x(1)^2-2*x(1)*x(2)+6*x(1)+x(2)^2-6*x(2)
     ```
  3. Save the M-File in a directory on the MATLAB path.

  To check that the M-File returns the correct value enter

  ```
  my_fun([4 7])
  ans = -9
  ```

- Optimizing a function using Genetic Algorithm.

  To optimize the following functions we have to write the M-File for each function as explained above and then proceed as mentioned below.

1. Rastrigin's Function-

$$Ras(x) = 20 + x1^2 + x2^2 - 10(cos2\pi x1 + cos2\pi x2)$$

- Using Command Line:

  Programme

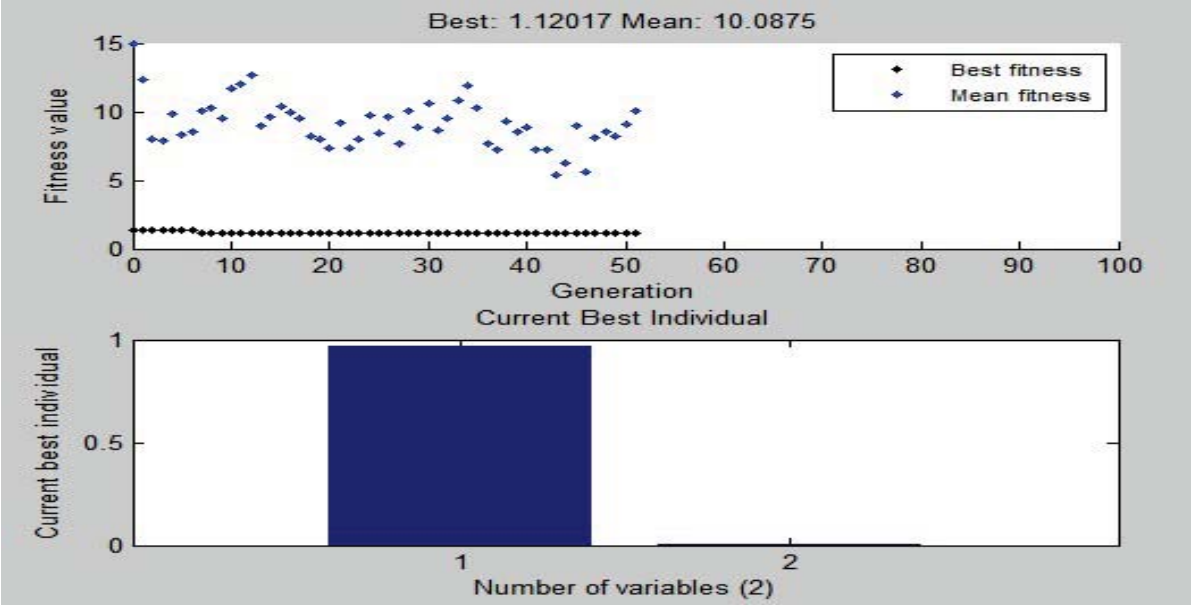
  ```
  function y = rast(x)
  % The default value of n = 2
  n = 2
  s = 0
  for j = 1:n
  s = s+(x(j)^2-10*cos(2*pi*x(j)));
  end
  y= 10*n+s
  ```

  Output

  ```
  X = 0.0027  -0.0052
  fval = 0.0068
  ```

- Using GA TOOL:
  1. Enter 'gatool' at the command line.
  2. In the 'fitness function' field enter '@rastriginsfcn'.
  3. In the 'number of variable' field enter '2'.
  4. Click the 'start' button.
  5. When the algorithm ends, the 'Status and Result' pane shows the result.

Output (For 'best fitness' and 'best individuals')



2. Ackley Function-

$f(x,y)=\frac{1}{20}\{-20e[-0.2\sqrt{(\frac{1}{2}(x^2+y^2))}]-e[\frac{1}{2}(\cos cx+\cos cy)]+20+e+5.7\}$

- Using Command Line:

  Programme

```
function z = ft_ackley(in)
a = 20;
b = 0.2;
c = 2*pi;
d = 5.7;
f = 0.8;
n = 2;
x = in(:,1);
y = in(:,2);
z=(1/f)*(-a*exp(-b*sqrt((1/n)*(x^2+y^2)))-exp((1/n)*(cos(c*x)+cos(c*y)))+a+exp(1)+d);
```

  Output

  (for x=0.0072 and y=0.0260)

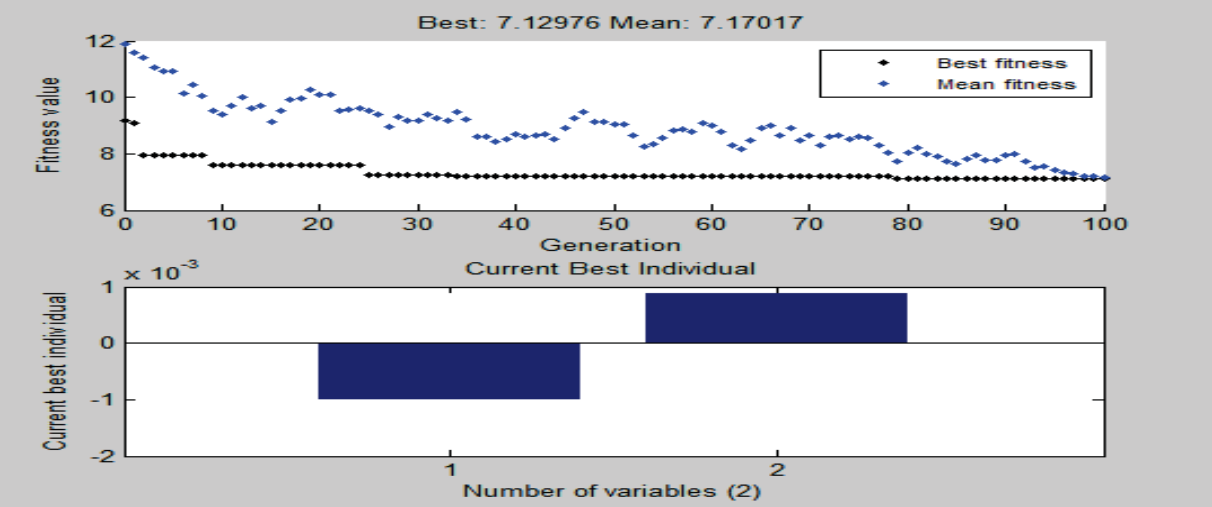
  fval = 7.2446

- Using GA TOOL:
  1. Open the gatool window.
  2. In the 'Fitness Function' field enter '@ft_ackley'.
  3. In the 'number of variable' field enter '2'.
  4. Set the following options.
     - Population Type- Double Vector.
     - Population Size- Use Default:20
     - Creation Function- Use Default:[]
     - Initial Scores- Use Default:[]

- Initial Range- Use Default:[]
- Selection Function- Stochastic Uniform.
- Elite Count- Use Default:2

5. Click the 'start' button.
6. To plot the output select the plot interval 1 and mark the 'Best Fitness'.

Output (for 'best fitness', 'best individuals' and 'stall generations'=100)



3. Easom Function-

f(x)=-cosx1*cosx2*exp(-$(x1-\pi)^2$-$(x2-\pi)^2$)

- Using Command Line:

Programme

```
function y = easom(x)
% Easom function
% The number of variables n = 2.
y = -cos(x(1))*cos(x(2))*exp(-(x(1)-                pi)^2-(x(2)-pi)^2);
```

Output

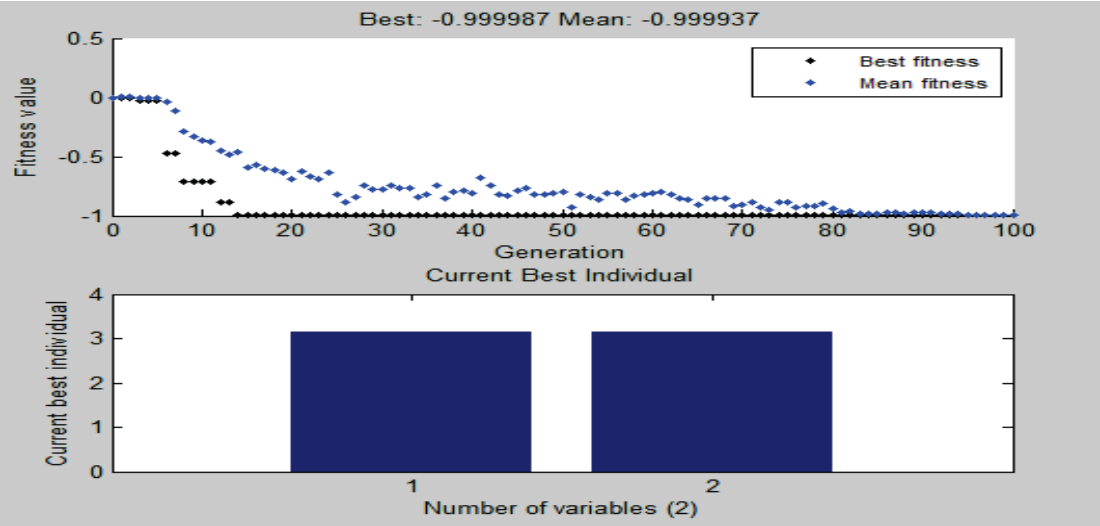
(for x1=3.1322 and x2=3.1577)

fval = -0.9995

- Using GA TOOL:

1. Open the GA TOOL window.
2. In the 'fitness function' field enter '@easom'.
3. In the 'number of variables' field enter '2'.
4. Click the 'start botton'.
5. When the algorithm ends, the 'status and result' pane shows the result.

Output (for 'best fitness', 'best individuals' and 'stall generations'=100)



4. Sum Squares Function

f(x)=$\sum_{i=1}^{n} x_i^2$

- Using Command Line:

  Programme

```
function y = sum2(x)
% The default value of n = 15.
n = 15;
s = 0;
for j = 1:n
s = s+j*x(j)^2 ;
end
y = s;
```

  Output

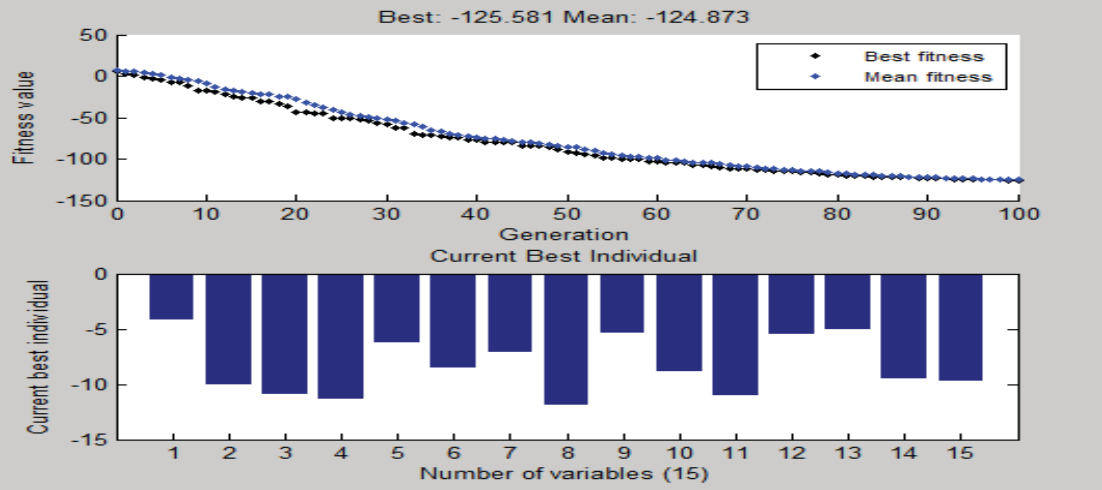
```
Optimization terminated: maximum number of generations exceeded.

x =
Columns 1 through 5
-10.3379   -4.7037  -10.6112   -8.1426   -8.1926
Columns 6 through 10
-8.1024   -8.9803   -4.9453   -8.8101  -11.5099
Columns 11 through 15
-7.8621   -4.7255   -7.0088  -11.0725  -11.7873

fval =
-126.7920
```

- Using GA TOOL:
  1. Open the GA TOOL window.
  2. In the ‘fitness function’ field enter ‘@sum’.
  3. In the ‘number of variables’ field enter ‘15’.
  4. Click the ‘start’ button.
  5. When the algorithm ends, the ‘status and result’ pane shows the result.

Output (for 'best fitness', 'best individuals' and 'stall generations'=100)



IV. CONCLUSION

To optimize an unimodal function, there are many other technique which can work efficiently and faster than 'Genetic Algorithm'. There are a lot of optimization techniques which work magnificently in dedicated problems but for complex multimodal problems with a frequent change in nature the Genetic Algorithm is the best choice for optimization. This technique may be slow but robust in nature and confirmly produce the possible best solution for optimization. We can increase the speed by introducing guided search techniques along with genetic algorithm in a combination, which is known as 'Hybrid Optimization'.

REFERENCES

[1] 'Tutorial on Genetic Algorithm'- Dr. Adel Abdennour(Electrical Engineering Department).
[2] 'Teaching Genetic Algorithm Using Matlab'- Y.Z.CAO and Q.H.WU
[3] 'A Genetic Algorithm for Function Optimization: A MATLAB IMPLEMENTATION'- Christopher.R.Houek
[4] 'Global Optimization Genetic Algorithm'- Olesya Peshks
[5] 'Genetic Algorithm For Optimization'- AndreyPopov(Hamburg 2005)
[6] 'A quick look at Genetic Algorithm Methods for optimization'- Reza Khademekbari