IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

A STUDY OF NONLINEAR UNCONSTRAINED OPTIMIZATION METHODS USING MATLAB

[1]Mr.Viral Chavan, [2]MR.Gaurishankar Gupta, [3]Mrs. Nikita Patel
[1]Lecturer, [2]Lecturer,[3]Lecturer
Applied Science and Humanities,
Parul University, Waghodia, Vadodara, India

Abstract: In this paper we had made MATLAB programs of elimination methods for single variable nonlinear optimization problems and interpolation methods for single variable nonlinear optimization problems. It also includes comparison of different elimination methods as well as different interpolation methods. By studying results of Elimination methods using this MATLAB programs we can observed that the Fibonacci method achieve any specified accuracy in least number of iterations. So, we can conclude that Fibonacci method is very efficient than other methods. By studying results of Interpolation methods using this MATLAB programs we can observed that the Newton method achieve any specified accuracy in least number of iterations. So, we can conclude that Newton method is very efficient than other methods.

Key words: Elimination methods, interpolation methods, Fibonacci method, Newton method, non-linear optimization

I. INTRODUCTION

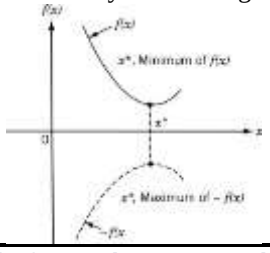
Optimization is the act of obtaining the best result under given circumstances. In design, construction and maintenance of any system, one have to take many technological and managerial decisions is to either minimize the effort required or maximize the desired benefit. Since the effort required or the benefit desired in any practical solution can be expressed as a function of certain decision variables, optimization can be defined as process of finding the values of such decision variables that gives optimum value of a function so constructed and which fulfill the restrictions in the situation under consideration .Optimization can be defined as the process of finding conditions that give the maximum or minimum values of function.

It can be seen from the below Figure that if a point x^* corresponds to the minimum value of function $f(x)$, the same point also corresponds to the maximum value of the negative of the function $-f(x)$.

Thus without loss of generality, optimization can be taken to mean minimization since the maximum of a function can be found by seeking the minimum of the negative of the same function.

Elimination method is the process of simplification of an expression or a function by eliminating radicals without changing the value of the expression or the root of the function. These methods eliminate the radicals and remove the corresponding partition of interval. This process gives the optimized point and corresponding value of the function. The assumption of unimodality is made in all the elimination techniques. If a function is known to be multimodal (i.e., having several valleys or peaks), the range of the function can be subdivided into several parts and the function treated as a unimodal function in each part. Some of the elimination methods are Unrestricted search method, Exhaustive search method, Dichotomous search method, Interval halving method, Fibonacci method and Golden selection method.

The efficiency of an elimination method can be measured in terms of the ratio of the final and the initial intervals of uncertainty L_n / L_0. The values of this ratio achieved in various methods for a specified number of experiments ($n = 5$ and $n = 10$) are compared. It can be seen that the Fibonacci method is the most efficient method, followed by the golden section method, in reducing the interval of uncertainty. A similar observation can be made by considering the number of experiments (or function evaluations)



needed to achieve a specified accuracy in various methods. The results are compared for maximum permissible errors of 0.1 and 0.01.It can be seen that to achieve any specified accuracy, the Fibonacci method requires the least number of experiments, followed by the golden section method.

Table 1: Final intervals of uncertainty

Method	**Formula**	**n = 5**	**n = 10**
Exhaustive search	$L_n = \frac{2}{n} + 1 L_0$	$0.33333L_0$	$0.18182L_0$
Dichotomous search ($\delta = 0.01$ and $n =$ $even$)	$L_n = \frac{L_0}{2^{n/2}} + \delta\left(1 - \frac{1}{2^{n/2}}\right)$	$\frac{1}{4}L_0 + 0.0075$ with n=4, $\frac{1}{8}L_0 + 0.00875$ with n=6	$0.03125L_0$ $+ 0.0096875$
Interval halving ($n \geq 3$ and odd)	$L_n = \frac{1}{2^{\frac{n-1}{2}}} L_0$	$0.25L_0$	$0.0625L_0$ with n=9, $0.03125L_0$ with n=11
Fibonacci	$L_n = \frac{1}{F_n} L_0$	$0.125L_0$	$0.01124L_0$
Golden section	$L_n = 0.618^{n-1}L_0$	$0.1459L_0$	$0.0.01315L_0$

Table 2: Number of experiments for a specified accuracy

Method	**Error:** $\frac{1}{2}\frac{L_n}{L_0} \leq 0.1$	**Error:** $\frac{1}{2}\frac{L_n}{L_0} \leq 0.01$
Exhaustive search	$n \geq 9$	$n \geq 99$
Dichotomous search ($\delta = 0.01$ *and even* n)	$n \geq 6$	$n \geq 14$
Interval halving ($n \geq 3$ and odd)	$n \geq 7$	$n \geq 13$
Fibonacci	$n \geq 4$	$n \geq 9$
Golden section	$n \geq 5$	$n \geq 10$

In the mathematical field of numerical analysis, interpolation is a type of estimation, a method of constructing new data points within the range of a discrete set of known data points. It is often required to interpolate, i.e., estimate the value of that function for an intermediate value of the independent variable. The necessary condition for $f(\lambda)$ to have a minimum of λ^* is that $f'(\lambda^*) = 0$. The direct root methods seek to find the root (or solution) of the equation, $f'(\lambda) = 0$. Three root finding methods - the Newton, the quasi- Newton, and the secant methods are discussed in this section.

II. MATLAB PROGRAMMING FOR ELIMINATION METHODS

In this chapter we have developed the MATLAB programs for the elimination methods for the following functions and compare the efficiency of the methods.

1. $f_1(x) = x^2 - 4$
2. $f_2(x) = x^3 + x^2 - x - 2$
3. $f_3(x) = x^5 - 5x^3 - 20x + 5$
4. $f_2(x) = \frac{x^3}{16} - \frac{27x}{4}$
5. $f_5(x) = 0.65 - 0.75/(1+x^2) - 0.65x\tan^{-1}(1/x)$

2.1 EXHAUSTIVE SEARCH:

```
%exhaustive search
method clear all
close all clc
%f=inline('x.*x-4');
%g=inline ('x.*x-4');
%A=-1;
%B=1;
f=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
g=inline('realpow(x,3.0)+realpow(x,2.0)-x-2'); A=-1;
B=1;

%f=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%g=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%A=0;
%B=5;
```

```
%f=inline('(realpow(x,3.0)/16)-(27*x/4)');
%g=inline('(realpow(x,3.0)/16)-(27*x/4)');
%A=0;
%B=10;

%f=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
%g=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
%A=0;
%B=3;

n=input('Enter the number of iterations:');
h=(B-A)/n;
x=A:h:B;
y=A:h:B;
for
i=1:length(y) plot(y(i),g(y(i))) hold on
end
fprintf('\ni\tx(i)\tf(x(i))'); for i=1:n
fprintf('\n%d\t%f\t%f',i,x(i),f(x(i))); end
min=f(x(1));
minp=x(1);
for i=2:n
if min>f(x(i))
min=f(x(i)); minp=x(i); z=i;
end
end
ans=(x(z-1)+x(z+1))/2;
fprintf('\nthe point is %f \t its function value is %f',ans,f(ans));
plot(ans,f(ans),'*r')
```

2.2 DICHOTOMOUS SEARCH:

```
%dichotomous search clear all
close all
clc
%f=inline('x.*x-4');
%g=inline('x.*x-4');
%A=-1;
%B=1;

%f=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
```

```
%g=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
%A=-1;
%B=4;

%f=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%g=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%A=0;
%B=5;

f=inline('(realpow(x,3.0)/16)-(27*x/4)');
g=inline('(realpow(x,3.0)/16)-(27*x/4)'); A=0;
B=10;

%f=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
%g=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
%A=0;
%B=3;
N=input('Enter the value of N:'); fprintf('\ni\tx(i)\tf(x(i))');
h=(B-A)/N;
x=A:h:B;
y=A:h:B;
for
 i=1:length(y) plot(y(i),g(y(i))) hold on
end
delta=0.0001; for i=1:N
x1=((B-A)/2)-(delta/2);
x2=((B-A)/2)+(delta/2);
f1=f(x1);
f2=f(x2);
if f2<f1
        A=x1;
 xmin=B;
 fprintf('\n%d\t%f\t%f',i,A,f(A));
 else if f2>f1
B=x2;
xmin=A;
 fprintf('\n%d\t%f\t%f',i,B,f(B)); end
end
end
xmin=(A+B)/2;
fprintf('\n The point is %f and its functional value is %f',xmin,f(xmin));
```

```
plot(xmin,f(xmin),'*r')
```

2.3 INTERVAL HALVING METHOD:

```
%inteval halving method close all
clear all clc
format long

%f=inline('x.*x-4');
%A=-1;
%B=1;

%f=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
%A=-1;
%B=4;

%f=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%A=0;
%B=5;

%f=inline('(realpow(x,3.0)/16)-(27*x/4)');
%A=0;
%B=10;

f=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))'); A=0;
B=3;
n=input('Enter the number of iterations:'); h=(B-A)/n;
x=A:h:B;
fprintf('\n\tx(i)\tf(x(i))');      for      i=1:length(x)
plot(x(i),f(x(i)))
hold on end L=B-A;
epsn = 0.0001; while L >epsn
x0=(A+B)/2;                    x1=(A+x0)/2;
x2=(x0+B)/2;
        f0=f(x0);
        f1=f(x1);
        f2=f(x2);
if f2>f0 && f0>f1
            B=x0;x0=x1;
fprintf('\n\t%f\t%f',x1,f(x1)); else if f1>f0 && f0>f2
                  A=x0;
x0=x2; fprintf('\n\t%f\t%f',x2,f(x2));
else if
f1>f0 && f2>f0
```

```
                A=x1; B=x2;
        fprintf('\n\t%f\t%f',x0,f(x0)); end
        end
                L = B-A;
        end end
        xstar=x0; plot(xstar,f(xstar),'r*')
        fprintf('\nThe point is %f and its function value is %f',xstar,f(xstar));
```

2.4 FIBONACCI METHOD:

```
%fibonacci method clear all
close all clc
format long
  %f=inline('x.*x-4');
  %g=inline('x.*x-4');
  %a=-1;
  %b=1;

    %f=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
    %g=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
    %a=-1;
    %b=1;

    %f=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
    %g=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
    %a=1;
    %b=3;

        f=inline('(realpow(x,3.0)/16)-(27*x/4)');
        g=inline('(realpow(x,3.0)/16)-(27*x/4)'); a=5;
        b=7;

  %f=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
  %g=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
  %a=0;
  %b=2;

M=input('Enter the value of M:'); N=M+1;
h=(b-a)/M;
x=a:h:b;
y=a:h:b;

for i=1:length(y) plot(y(i),g(y(i))) hold on
```

```
end

L=zeros(1,50); L(1)=2;
F=zeros(1,N);
F(1)=1;
F(2)=1;
for i=3:N
F(i)=F(i-1)+F(i-2);
end
    for i=2:N
    L(i)=(F(N-(i-1))./F(N)).*L(1);
    end


    x1=a+(F(N-2)/F(N)).*L(1);
    x2=b-(F(N-2)/F(N)).*L(1);
    f1=f(x1);
    f2=f(x2); for i=2:N-1

    Ls(i)=(F(N-i)./F(N-(i-2))).*L(i-1);
    x1=a+Ls(i); x2=b-Ls(i);
    f1=f(x1);
    f2=f(x2);

if f1<=f2
          b=x2; xstar=x1; else if f1>f2
        a=x1; xstar=x2; end
        end
    fprintf('\n%d\t%f\t%f',i,xstar,f(xstar)); end
    fprintf('\nthe point is %f \n its function value is %f',xstar,f(xstar)); plot(xstar,f(xstar),'*r')
```

2.5 GOLDEN SELACTION METHOD:

```
%golden section method clear all
close all clc
f=inline('x.*x-4');
g=inline('x.*x-4'); A=-1;
B=1;
%f=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
%g=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
%A=-1;
%B=4;


%f=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%g=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%A=1;
%B=3;
```

```
%f=inline('(realpow(x,3.0)/16)-(27*x/4)');
%g=inline('(realpow(x,3.0)/16)-(27*x/4)');
%A=5;
%B=7;

%f=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
%g=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
%A=0;
%B=2;

n=input('Enter the number of iterations:'); h=(B-A)/n;
fprintf('\ni\tx(i)\tf(x(i))\n'); r=0.382;
x=A:h:B;
for i=1:length(x) plot(x(i),g(x(i))) hold on
end i=1;

while(i<=n) x1=A+r.*(B-A);
x2=B-r.*(B-A);
f1=f(x1);
 f2=f(x2); if f1<=f2
B=x2;
fprintf('%d\t%f\t%f\n',i,x1,f(x1)); xstar=x1;
else
A=x1;
fprintf('%d\t%f\t%f\n',i,x2,f(x2)); xstar=x2;
end
  i=i+1; plot(xstar,f(xstar)) hold on
end
if  f(A)<f(B)  point=A;  else  point=B;
end
plot(point,f(point),'r*')
fprintf(' in iteration %d The point is %f and its function value is
%f',i,point,f(point))
```

Table 3: Comparison of Elimination methods

	Minimum values from above programs															**Minimum values from analytic method**	
Methods	**Exhaustive method**			**Dichotomous method**			**Interval halving method**			**Fibonacci method**			**Golden selection method**				
Function	Minimum point	Minimum value	Itr	Minimum point	Minimum value	Itr	Minimum point	Minimum value	Itr	Minimum point	Minimum value	Itr	Minimum point	Minimum value	Itr	Minimum point	Minimum value
$F(x)= x^2-4$	0.000000	-4.000000	9	0.000050	-4.000000	9	**0.000000**	**-4.000000**	**1**	0.000626	-4.000000	6	0.000193	-4.000000	8	0.000000	-4.000000
$F(x)= x^3+x^2-x-2$	0.250000	-2.171875	13	0.300730	-2.183094	15	0.333351	-2.185185	16	**0.333751**	**-2.185185**	**9**	0.332394	-2.185183	14	0.400000	-2.176000
$F(x)=x^5-5x^3 - 20x+5$	1.875000	-42.284698	8	1.666695	-38.622053	8	2.000086	-43.000000	9	**1.999999**	**-43.000000**	**5**	1.999790	-42.999780	6	2.000000	-43.000000
$F(x)=x^3/16-27x/4$	6.250000	-26.928711	12	6.666625	-26.481547	13	5.999985	-27.000000	14	**5.999998**	**-27.000000**	**8**	5.999793	-26.999995	10	6.000000	-27.000000
$F(x)= 0.65-(0.75/(1+x^2))-0.65x\tan^{-1}(1/x)$	0.562500	-0.306713	15	0.500026	-0.309823	16	0.480881	-0.310021	16	**0.480888**	**-0.310021**	**8**	0.480646	-0.310020	12	0.480900	-0.310021

3. MATLAB PROGRAMMING FOR INTERPOLATION METHODS

In this chapter we have developed the MATLAB programs for the Interpolation methods for the following functions and compare the efficiency of the methods.

1. $f_1x = x^2 - 4$
2. $f_2x = x^3 + x^2 - x - 2$
3. $f_3x = x^5 - 5x^3 - 20x + 5$
4. $f_4x = \frac{x^3}{16} - \frac{27x}{4}$
5. $f_5x = 0.65 - \frac{0.75}{1+x^2} - 0.65\ xtan^{-\frac{1}{1+x^2}}$

3.1 NEWTON METHOD:

```
%newton method clear all
  close all clc
f=inline('realpow(x,2.0)-4'); g=inline('2*x');
h=inline('2');
                x1=-3;

%f=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
%g=inline('3*realpow(x,2.0)+2*x-1');
%h=inline('6*x+2');
%x1=4;

%f=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%g=inline('5*realpow(x,4.0)-15*realpow(x,2.0)-20');
%h=inline('20*realpow(x,3.0)-30*x');
%x1=5;

%f=inline('(realpow(x,3.0)/16)-(27*x/4)');
%g=inline('(3*realpow(x,2.0)/16)-(27/4)');
%h=inline('(3*x/8)');
%x1=10;

%f=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
%g =inline('1.5*x/((1+x*x)*(1+x*x))+ 0.65*x/(1+x*x)- 0.65*atan(1/x)');
%h =inline('(2.8-3.2*x*x)/((1+x*x)*(1+x*x)*(1+x*x))');
%x1=0.01;

  fprintf('\nx \t\t\t\t df1 \t\t\t\t df2'); epsn = 0.01;
     itr=0;
  while abs(g(x1)) >epsn itr=itr+1;
     x2=x1-(g(x1)./h(x1));
    fprintf('\n in loop %f\t%f\t\t%f\n',x2,g(x2),h(x2)); x1=x2;
```

```
        end
        xstar = x2;
        fprintf('\n the optimum solution is %f and function value is %f in %d
         iterations',xstar,f(xstar), itr);
```

3.2 QUASI-NEWTON METHOD:

```
      %quasi newton method clear all
        close all clc
%f=inline('realpow(x,2)-4');
%g=inline('realpow((x+0.01),2)-4');
%h=inline('realpow((x-0.01),2)-4');
%k=inline('2*x');
%x1=-3;

%f=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
%g=inline('realpow((x+0.01),3.0)+realpow((x+0.01),2.0)-(x+0.01)-2');
%h=inline('realpow((x-0.01),3.0)+realpow((x-0.01),2.0)-(x-0.01)-2');
%k=inline('3*realpow(x,2.0)+2*x-1');
%x1=4;

%f=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%g=inline('realpow((x+0.01),5.0)-5*realpow((x+0.01),3.0)-20*(x+0.01)+5');
%h=inline('realpow((x-0.01),5.0)-5*realpow((x-0.01),3.0)-20*(x-0.01)+5');
%k=inline('5*realpow(x,4.0)-15*realpow(x,2.0)-20');
%x1=5;

%f=inline('(realpow(x,3.0)/16)-(27*x/4)');
%g=inline('(realpow((x+0.01),3.0)/16)-(27*(x+0.01)/4)');
%h=inline('(realpow((x-0.01),3.0)/16)-(27*(x-0.01)/4)');
%k=inline('(3*realpow(x,2.0)/16)-(27/4)');
%x1=10;
f=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
g=inline('0.65-(0.75/(1+(x+0.01)*(x+0.01)))-(0.65*(x+0.01)*atan(1/(x+0.01)))');
h=inline('0.65-(0.75/(1+(x-0.01)*(x-0.01)))-(0.65*(x-0.01)*atan(1/(x-0.01)))');
k=inline('1.5*x/((1+x*x)*(1+x*x))+ 0.65*x/(1+x*x)- 0.65*atan(1/x)'); x1=0.01;
fprintf('\nx \t\t\t\t\t f(x)'); epsn=0.01;
itr=0;
      while abs(k(x1))>epsn itr=itr+1;
          a=0.01*(g(x1)-h(x1));
          b=2*(g(x1)-2*f(x1)+h(x1));
          x2=x1-a/b;
          fprintf('\n%f\t\t%f\n',x2,f(x2)); x1=x2;
          end
```

```
xstar = x2;
    fprintf('\n the optimum solution is %f and function value is %f in %d
iteration',xstar,f(xstar),itr);
```

3.3 SECANT METHOD:

```
%secant method clear all
close all clc
format long
%f=inline('realpow(x,2)-4');
%g=inline('2*x');

%f=inline('realpow(x,3.0)+realpow(x,2.0)-x-2');
%g=inline('3*realpow(x,2.0)+2*x-1');

%f=inline('realpow(x,5.0)-5*realpow(x,3.0)-20*x+5');
%g=inline('5*realpow(x,4.0)-15*realpow(x,2.0)-20');

%f=inline('(realpow(x,3.0)/16)-(27*x/4)');
%g=inline('(3*realpow(x,2.0)/16)-(27/4)');

f=inline('0.65-(0.75/(1+x*x))-(0.65*x*atan(1/x))');
g=inline('1.5*x/((1+x*x)*(1+x*x))+ 0.65*x/(1+x*x)- 0.65*atan(1/x)');

A=0.0;
s=0.1;
epsn=0.001;
maxitr=1000;
while g(A)>0 A=A+s;
end s=0.1;
if g(s)>=0 B=s;
gB=g(s);
A=0.0;
end
while g(s)<0 A=s; B=A+s;
gA=g(s);
s=2*s;
end itr=0;
for i=1 :
 maxitr itr=itr+1;
x2 = A-( g(A)*(B-A)/(g(B)-g(A)));
```

Table 4: Comparison of Interpolation methods

	Minimum values from above programs									**Minimum values from analytic method**	
Methods	**Newton**			**Quasi-newton**			**Secant**				
Functions	minimum point	minimum value	Itr	minimum point	minimum value	Itr	minimum point	minimum value	Itr	minimum point	minimum value
$F(x)=x^2-4$	**0.000000**	**-4.000000**	**1**	**0.000000**	**-4.000000**	**1**	**0.000000**	**-4.000000**	**1**	0.000000	-4.000000
$F(x)=x^3+x^2-x-2$	0.333373	-2.185185	5	0.333374	-2.185185	5	**0.333317**	**-2.176000**	**3**	0.400000	-2.176000
$F(x)= x^5-5x^3 -20x+5$	**2.000000**	**-43.000000**	7	1.999965	-43.000000	7	1.999993	-43.000000	27	2.000000	-43.000000
$F(x)=(x^3/16)-(27x/4)$	**6.000180**	**-27.000000**	**3**	6.000183	-27.000000	3	5.999877	-27.000000	2	6.000000	-27.000000
$F(x)=0.65-(0.75/(1+x^2))$ $-0.65x\tan^{-1}(1/x)$	**0.480196**	**-0.310020**	**3**	0.480261	-0.310020	3	0.481056	-0.310020	4	0.480900	-0.310020

Conclusion:

By studying results of Interpolation methods using above programs we can observed that the Newton method achieve any specified accuracy in least number of iterations. So we can conclude that **Newton method** is very efficient than other methods

REFERENCES:

1. Harvey M. Wagner, "Principles of Operations Research with applications to managerial decisions", Prentice-Hall of India
2. Mohan .C. Joshi and Kannan M. Moudgalya, "Optimization: Theory and Practice" 2006.
3. Singiresu S. Rao, "Engineering Optimization Theory and Practice", Fourth Edition, Wiley India Pvt. Ltd.
4. Stephen J. Chapman, "MATLAB Programming for Engineers".
5. William J. Palm III, "Introduction to MATLAB for Engineers".