





Guide: Respected Mahipal Reddy

EXPERIMENT 1

# Graphical optimization using linpro for 2 variables

(i) Application of experiment:

Modern science

Economics

Management

**LPP.method**

1 .A patient in a hospital is required to have at least 84 units of dug A and 120 units of dug B each day. Each gram of substance M contains 10 units of drug A and 8 units of B, and each gram of substance N contains 2 units of drug A and 4 units of drug B. Now suppose that both M and N contain an undesirable drug C, 3 units per gram in M and 1 unit per gram in N. How many grams of substance M and N should be mixed to meet the minimum daily requirement at the same time minimize the intake of drug C? How many units of the desirable drag C will be in this mixture.

# INPUT

%

%

%L31-32

clc clear all tic

C=[3 1];

A=[-10 -2;-8 -4];

B=[-84;-120];

[X Fmin]=linprog(C,A,B) Toc

# OUTPUT

>> LPP

Optimal solution found. X =

4.0000 //required M

22.0000 //required N Fmin = 34 //C units

Elapsed time is 1.410230 seconds.

EXPERIMENT 2

(i) Application of experiment:

Modern science

Economics

Management

Simplex Algorithm in resource allocation in Industry

1. A manufacturer produces two products A and B. Both the products are processed on two different machines. The available capacity of first machine is 12 hours and that of second machine is 9 hours per day. Each unit of product A requires 3 hours on both machines and each unit of product B requires 2 hours on first machine and 1 hour on second machine. Each unit of product A is sold at Rs 7 profit and B at a profit of Rs 4. Find the production level per day for maximum profit.

# INPUT

%

%

%

clc clear all

tic % time complexity, here if we give input in command window the time complexity increases so given in the code itself.

A = [3 2 1 0 ; 3 1 0 1 ];%coefficient matrix b = [12;9];%constant matrix

cm=[7 4 0 0 0];%coefficients of objective function

cb = [0 0];% coefficients of basic variables in objective function b\_var = [ 3; 4];%basic variables representation

disp('initial matrix');

Au = [b A]

[m n]=size(Au); for i = 1:10

for j = 2:n

zc(j-1) = cb\*Au(:,j)-cm(j-1); end

disp('zc'); disp(zc); if(zc>=0)

disp('optimal');

disp( cm(b\_var) \*Au(:,1)); break;%end the program

else

% [k pivotcol]=min(zc); pivotcol =1;

for v = 2:n-1 if(zc(pivotcol)>zc(v))

pivotcol =v; end

end

%displaying entering variable disp('entering variable:'); disp(pivotcol);

%finding ratio

ratio = Au(:,1)./Au(:,pivotcol+1);

% ratio = abs(ratio); disp('ratio'); disp(ratio);

temp = 0; for b = 1:m

if(ratio(b)>0) temp = temp +1;

end end

if(temp==0) disp('unbounded'); break;

end

%[pivotrow l] = min(ratio);

%finding minimum index in ratio vector min=1000;

pivotrow = 0; for v = 1:

if(min>ratio(v) && ratio(v)>=0) pivotrow=v;

min = ratio(v); end

end

%displaying leaving variable disp('leaving variable'); disp(pivotrow);

pivotelement = Au(pivotrow,pivotcol+1); disp('pivot element');

disp(pivotelement);

%displaying pivot element

Au(pivotrow,:) = Au(pivotrow,:)/pivotelement; newpivotrow = Au(pivotrow,:);%new pivot row for q = 1:m

if(q~=pivotrow)

Au(q,:) = Au(q,:)-newpivotrow\*Au(q,pivotcol+1); end

end

%displaying new Augumented matrix disp(Au);

%changing coeffcients of basic varibles cb(pivotrow) = cm(pivotcol);

%displaying cb

disp('cb'); disp(cb);

%displaying b matrix disp('b');

b = Au(:,1);

disp(b);

b\_var(pivotrow,1) = pivotcol;

%displaying basic variables disp('basic variables'); disp(b\_var);

end end

Toc

# OUTPUT

initial matrix Au =

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 12 | 3 | 2 | 1 | 0 |
| 9 | 3 | 1 | 0 | 1 |
| zc |  |  |  |  |  |
|  | -7 | -4 | 0 | 0 |  |

entering variable:

1

ratio

4

3

leaving variable 2

pivot element 3

3.0000 0 1.0000 1.0000 -1.0000

3.0000 1.0000 0.3333 0 0.3333

cb

0 7

b

3

basic variables 3

1

zc

0 -1.6667 0 2.3333

entering variable:

2

ratio

3

9

leaving variable 1

pivot element 1

3.0000 0 1.0000 1.0000 -1.0000

2.0000 1.0000 0 -0.3333 0.6667

cb

4 7

b

3

2

basic variables 2

1

zc

0 0 1.6667 0.6667

optimal 26

Elapsed time is 0.012465 seconds

Q3) **Applications of Big M Method**

**Maximize: z=-2x1 -1x2**

**Subject to constraints:**

**3x1+2x2<=0**

**4x1+3x2>=0**

**x1+2x2=0**

A=[3 2 0 0 1 0; 4 3 -1 0 0 1; 1 2 0 1 0 0];

B=[3; 6; 4];

CB=[-1000000, -1000000, 0];

CJ=[-2 -1 0 0 -1000000 -1000000];

Au=[B A];

[m n]=size(Au);

display("initial table")

for i=1:100

disp("Iteration "+ i)

Au

Z=[CB\*Au(:,2); CB\*Au(:,3)]

for j=1:2

X(j)=Z(j,1)-CJ(j)

end

[Q pc]=min(X)

for k=1:m

ratio(k)=Au(k,1)/Au(k,pc+1)

end

[S pcc]=min(ratio)

for ppp=1:size(ratio)

if S<0

ratio(pcc)=100000

[S pcc]=min(ratio)

end

end

div=Au(pcc,pc+1)

Au;

Au(pcc,:)=Au(pcc,:)/div

for l=1:pcc-1

Au(l,:)=Au(l,:)-(Au(l,pc+1)\*Au(pcc,:))

end

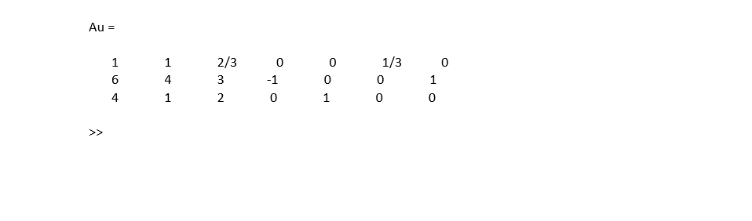
end

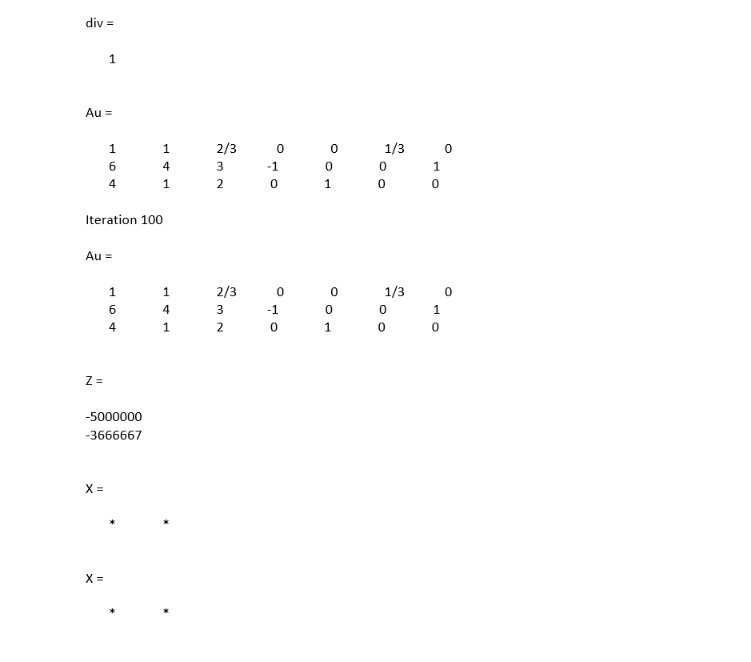
Input: A=[3 2 0 0 1 0; 4 3 -1 0 0 1; 1 2 0 1 0 0];

B=[3; 6; 4];

CB=[-1000000, -1000000, 0];

Output





EXPERIMENT 4



# Dual method

1. Egg contains 6 units A and 7 units of B per gram and costs 12 paise per gram. Milk contains 8 units of vitamin A and 12 units of Vitamin B and costs 20 paise per gram. The daily minimum requirement of vitamin A and vitamin B are 100 units and 120 units respectively. Find the optimal product mix. (Hint: Let x1 and x2 are the number of units of egg and milk)

# INPUT:

%18bcde7178

%Mitali Chauhan

%L31-L32

clc; clear all; close all; tic

A=[-6 -8 1 0;-7 -12 0 1];

B=[-100;-120];

CM=[12 20 0 0];%cost matrix

C=[0;0];%coeff of basic variables in objective fun AM=[B A]

[m n]=size(AM) for i=1:4

for j=2:n

zc(j-1)=C'\*AM(:,j)-CM(j-1)

end

if(zc(1:n-1)<0)

fprintf("The method is fail") else

AM1 =AM(:,1); if(AM1(1:m)>0|AM1(1:m)==0)

fprintf("current solution is optimum feasible") disp(C'\*AM1);

else

[lvalue,lv]=min(AM1(:,1)); AM;

ratio=zeros(1:n-1); for k=2:n

if (AM(lv,k)<0)

ratio(k-1)=zc(k-1)/AM(lv,k); else

ratio(k-1)=10000; end

end ratio

[evalue,en]=max(ratio(1:n-1)); end

fprintf("\n Maximum ratio(entering variable)%d and pivot row(leaving variable) %d",en,lv) C(lv)=CM(en);

%replacing incoming instead of outgoing AM;

%the new pivot row AM(lv,:)=AM(lv,:)/AM(lv,en+1); for w=1:m%no of rows

if(w==(w-lv))

AM(w,:)=AM(w,:)-AM(w,en+1)\*AM(lv,:)

end end

end end

toc

# OUTPUT:

AM =

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -100 | -6 | -8 | 1 | 0 |
| -120 | -7 | -12 | 0 | 1 |

m =

2

n =

5

zc =

-12

zc =

-12 -20

zc =

-12 -20 0

zc =

-12 -20 0 0

ratio(:,:,1,1) = 1.7143 1.6667

ratio(:,:,2,1) =

10000 10000

ratio(:,:,3,1) = 0 0

ratio(:,:,1,2) = 0 0

ratio(:,:,2,2) = 0 0

ratio(:,:,3,2) = 0 0

ratio(:,:,1,3) = 0 0

ratio(:,:,2,3) = 0 0

ratio(:,:,3,3) = 0 0

ratio(:,:,1,4) = 0 0

ratio(:,:,2,4) = 0 0

ratio(:,:,3,4) = 0 0

Maximum ratio(entering variable)3 and pivot row(leaving variable) 2 zc =

NaN -20 0 0

zc =

NaN NaN 0 0

zc =

NaN NaN NaN 0

zc =

NaN NaN NaN NaN

ratio(:,:,1,1) = NaN NaN

ratio(:,:,2,1) =

10000 10000

ratio(:,:,3,1) = 0 0

ratio(:,:,1,2) =

0 0

ratio(:,:,2,2) = 0 0

ratio(:,:,3,2) = 0 0

ratio(:,:,1,3) = 0 0

ratio(:,:,2,3) = 0 0

ratio(:,:,3,3) = 0 0

ratio(:,:,1,4) = 0 0

ratio(:,:,2,4) = 0 0

ratio(:,:,3,4) = 0 0

Maximum ratio(entering variable)3 and pivot row(leaving variable) 2 zc =

NaN NaN NaN NaN

zc =

NaN NaN NaN NaN

zc =

NaN NaN NaN NaN

zc =

NaN NaN NaN NaN

ratio(:,:,1,1) = NaN NaN

ratio(:,:,2,1) =

10000 10000

ratio(:,:,3,1) = 0 0

ratio(:,:,1,2) = 0 0

ratio(:,:,2,2) = 0 0

ratio(:,:,3,2) = 0 0

ratio(:,:,1,3) = 0 0

ratio(:,:,2,3) =

0 0

ratio(:,:,3,3) = 0 0

ratio(:,:,1,4) = 0 0

ratio(:,:,2,4) = 0 0

ratio(:,:,3,4) = 0 0

Maximum ratio(entering variable)3 and pivot row(leaving variable) 1 zc =

NaN NaN NaN NaN

zc =

NaN NaN NaN NaN

zc =

NaN NaN NaN NaN

zc =

NaN NaN NaN NaN

ratio(:,:,1,1) =

NaN NaN ratio(:,:,2,1) =

10000 10000

ratio(:,:,3,1) = 0 0

ratio(:,:,1,2) =

0 0

ratio(:,:,2,2) =

0 0

ratio(:,:,3,2) = 0 0

ratio(:,:,1,3) =

0 0

ratio(:,:,2,3) = 0 0

ratio(:,:,3,3) =

0 0

ratio(:,:,1,4) = 0 0

ratio(:,:,2,4) = 0 0

ratio(:,:,3,4) = 0 0

Maximum ratio(entering variable)3 and pivot row(leaving variable) 1Elapsed time is 0.018091 seconds.

EXPERIMENT -5

Q5) **Application of Dual simplex method**

(i) Egg contains 6 units A and 7 units of B per gram and costs 12 paise per gram. Milk contains 8 units of vitamin A and 12 units of Vitamin B and costs 20 paise per gram. The daily minimum requirement of vitamin A and vitamin B are 100 units and 120 units respectively. Find the optimal product mix.

(ii) max = max 12x1+20x2

s.t.c

6X1+8x1>=100;

12x1+120x2>=120

x1,x2>=0;

(iii)Matlab Code:

clc;

clear all;

close all;

tic

A=[6 8 1 0;12 120 0 1];

B=[100;120];

CM=[12 20 0 0];%cost matrix

C=[0;0];%coeff of basic variables in objective fun

AM=[B A]

[m n]=size(AM)

for i=1:4

for j=2:n

zc(j-1)=C'\*AM(:,j)-CM(j-1)

end

if(zc(1:n-1)<0)

fprintf("The methid is fail")

else

AM1 =AM(:,1);

if(AM1(1:m)>0)

fprintf("current solution is optimum feasible")

else

[lvalue,lv]=min(AM1(:,1));

AM;

ratio=zeros(1:n-1);

for k=2:n

if (AM(lv,k)<0)

ratio(k-1)=zc(k-1)/AM(lv,k);

else

ratio(k-1)=10000;

end

end

ratio

[evalue,en]=max(ratio(1:n-1));

end

fprintf("\n Maximum ratio(entering variable)%d and pivot row(leaving variable) %d",en,lv)

C(lv)=CM(en);%replceing incoming instead of outgoing

AM;

%the new pivot row

AM(lv,:)=AM(lv,:)/AM(lv,en+1);

for w=1:m%no of rows

if(w==(w-lv))

AM(w,:)=AM(w,:)-AM(w,en+1)\*AM(lv,:)

end

end

end

end

toc

(i) Input:

A=[6 8 1 0;12 120 0 1];

B=[100;120];

CM=[12 20 0 0]

Output:





EXPERIMENT -6

Application of Transportation PROBLEM

A company has three plants P1, P2 and P3, four warehouse W1, W2, W3 and W4 The number of units available at the plants is 11, 13, 19 and the demand at warehouse W1, W2, W3 and W4 is 6, 10, 12, 15 respectively. The unit cost of the transportation is given in the following table:

W1 W2 W3 W4

Supply

P1 21 16 25 13 11

P2 17 18 14 23 13

P3 32 17 18 41 19

Demand

6 10 12 15 43

Find the allocation so that the total transportation cost is minimum

# INPUT :-

%18bce7178

%Mitali Chauhan

%L31-L32

clc

clear all;

QM=input('enter transportation matrix') S=input('enter supply column') D=input('enter demand row')

A=[QM S;D]

[m,n]=size(A); C=zeros(m,n); rsum=sum(A(1:m-1,n));

csum=sum(A(m,1:n-1)); if(rsum==csum)

for i=1:m-1 for j=1:n-1

X(i,j)=min(A(i,n),A(m,j));

A(i,n)=A(i,n)-X(i,j);

A(m,j)=A(m,j)-X(i,j);

end

end else

display("ünbalanced problem"); end

TIC=0;%initial basis feasible solution for i=1:m-1

for j=1:n-1 TIC=TIC+A(i,j)\*X(i,j);

end end TIC a=1; b=0;

u=zeros(1,m-1);

v=zeros(n-1,1); u(1)=0;

for i=1:m-1 for j=1:n-1

if(X(i,j)==0)%Occupied cells continue

else

if(j==b+1) v(j)=A(i,j)-u(i); b=j;

else

u(i)=A(i,j)-v(j); end

end end

end

for i=1:m-1 for j=1:n-1

if(X(i,j)==0)%unoccupied cells C(i,j)=A(i,j)-(u(i)+v(j));

end end

end

TFC=0 %optimal basis feasible solution for i=1:m-1

for j=1:n-1 TFC=TFC+A(i,j)\*X(i,j);

end end TFC

# OUTPUT :-

enter transportation matrix[13 16 19 17;17 19 16 25;25 27 17 16]

|  |  |  |  |
| --- | --- | --- | --- |
| QM = |  | | |
| 13 | 16 | 19 | 17 |
| 17 | 19 | 16 | 25 |
| 25 | 27 | 17 | 16 |

enter supply column[250;200;250]

S =

250

200

250

enter demand row[100 250 250 100 750]

D =

100 250 250 100 750

A =

|  |  |  |  |
| --- | --- | --- | --- |
| 13 | 16 | 19 | 17 250 |
| 17 | 19 | 16 | 25 200 |
| 25 | 27 | 17 | 16 250 |
| 100 | 250 | 250 | 100 750 |

TIC =

11350

TFC =

0

TFC =

11350

EXPERIMENT -7



Use three iterations of the golden search method inorder to maximize the function f(x)=10+x3−2x−5exp(x) in the interval(-5,5)

# INPUT

%18bce7178

%Mitali Chauhan

%L31-L32

clc clearall closeall tic symsr

fo=(10+(r\*3)-(2\*r)+(-5\*exp(r))); a=0;

b=2;

w=(r-a)/(b-a); ro=w\*(b-a)+a;

f=subs(fo,r,ro); aw=0;

bw=1; k=1;

fori=1:10 lw=bw-aw;

w1=aw+(0.681)\*lw w2=bw-(0.681)\*lw fw1=subs(f,r,w1)%function

fw2=subs(f,r,w2)%function if(w1<w2)

iffw1>fw2 aw=w1; elsefw1<fw2 bw=w2;

end

else if(fw1>fw2) bw=w1; else aw=w2;

end end i=i+1; end aw bw toc

# OUTPUT:-

fw1=

24747170039161093/2251799813685248-5\*exp(2229171902308613/2251799813685248)

fw2=

49442983940903115/4503599627370496-5\*exp(4406987667198155/4503599627370496)

aw= 0.9785

bw=

1

TIME COMPLEXITY

Elapsedtimeis2.548887seconds.

EXPERIMENT -8



Maximize the function *f(x)=-x^2+21.6\*x+3* over the interval [0,20] *.* . Via calculus the maximum is at 10.8. Let's see how the Fibonacci Search does.The interval [a,b] is [ 0.00, 20.00]and user specified tolerance level is .00100.

The first 2 experimental endpoints are x1= 7.639 and x2 = 12.361.

# INPUT:-

%18bcde7178

%Mitali Chauhan

%L31-L32

clc clear all

FIBSearch:=proc(f::procedure,a::numeric,b::numeric,T::numeric) local L, M, j, C, x1, x2, N, val;

L:=(b-a)/T;

M:=round(L); j:=0;

label\_2; C:=fib(j); if C<L then j:=j+1;

goto(label\_2); else

N:=j;

end if;

x1:=evalf(a+(fib(N-2)/fib(N))\*(b-a));

x2:=evalf(a+(fib(N-1)/fib(N))\*(b-a));

printf("The interval [a,b] is [% 4.2f,% 4.2f]and user specified tolerance level is% 6.5f.\n",a,b,T); printf("The first 2 experimental endpoints are x1= % 6.3f and x2 = % 6.3f. \n",x1,x2);

printf(" \n");

printf(" \n");

printf(" Iteration x(1) x(2) f(x1) f(x2) Interval \n"); iterate(f,a,b,N,x1,x2);

val:=f(mdpt);

printf(" \n");

printf(" \n");

printf("The midpoint of the final interval is% 9.6f and f(midpoint) = % 7.3f. \n",mdpt, val); printf(" \n");

printf(" \n");

printf("The maximum of the function is % 7.3f and the x value = % 9.6f \n",fkeep,xkeep); printf(" \n");

printf(" \n"); end:

iterate:=proc(f::procedure,a::numeric,b::numeric, N::posint,x1::numeric,x2::numeric) local x1n,x2n,an,bn,i,fx1,fx2,j,f1,f2,fmid;

global mdpt,fkeep,xkeep,fib; i:=1;

x1n(1):=x1;

x2n(1):=x2;

an(1):=a;

bn(1):=b; i:=1;

for j from 1 to N do fx1(i):=evalf(f(x1n(i)));

fx2(i):=evalf(f(x2n(i))); if fx1(i)<=fx2(i) then an(i+1):=x1n(i);

bn(i+1):=bn(i);

x1n(i+1):=x2n(i);

x2n(i+1):=an(i+1)+(fib(N-i-1)/fib(N-i))\*(bn(i+1)-an(i+1)); else

an(i+1):=an(i);

bn(i+1):=x2n(i);

x2n(i+1):=x1n(i);

x1n(i+1):=an(i+1)+(fib(N-i-2)/fib(N-i))\*(bn(i+1)-an(i+1)); fi;

i:=i+1;

printf(" % 3.0f % 11.4f % 10.4f % 10.4f %10.4f [% 6.4f, %

6.4f]\n",i,x1n(i),x2n(i),fx1(i-1),fx2(i-1),an(i),bn(i)); mdpt := (an(i) + bn(i))/2;

if ((i+2)=N) then

if evalf(f(an(i))) > evalf(f(bn(i))) or evalf(f(an(i))) > evalf(f(mdpt)) then fkeep := f(an(i)); xkeep := an(i);

else

if evalf(f(bn(i))) > evalf(f(mdpt)) then fkeep := f(bn(i)); xkeep := bn(i);

else

fkeep := f(mdpt); xkeep := mdpt; end if;

end if; end if;

if(N-i-2)<0 then: goto(label\_3):end if; end do;

label\_3; end:

fib:=proc(n::numeric) option rememer; fib(0):=1:fib(1):=1;

if n<2 then n;

else

fib(n-1)+fib(n-2); end if;

end

# OUTPUT :-

f:= x->-x^2+21.6\*x+3;

FIBSearch(f,0,20,.001);

Iteration x(1) x(2) f(x1) f(x2) Interval

2 12.3607 15.2786 109.6501 117.2043 [ 7.6393, 20.0000]

3 10.5573 12.3607 117.2043 99.5818 [ 7.6393, 15.2786]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 4 | 9.4427 | 10.5573 | 119.5811 | 117.2043 | [ 7.6393, 12.3607] | |
| 5 | 10.5573 | 11.2461 | 117.7978 | 119.5811 | [ 9.4427, 12.3607] | |
| 6 | 10.1316 | 10.5573 | 119.5811 | 119.4410 | [ 9.4427, 11.2461] | |
| 7 10.5573 | | 10.8204 | 119.1932 | 119.5811 | [ 10.1316, | 11.2461] |
| 8 10.8204 | | 10.9830 | 119.5811 | 119.6396 | [ 10.5573, | 11.2461] |
| 9 10.7199 | | 10.8204 | 119.6396 | 119.6065 | [ 10.5573, | 10.9830] |
| 10 10.8204 | | 10.8825 | 119.6336 | 119.6396 | [ 10.7199, | 10.9830] |
| 11 10.7820 | | 10.8204 | 119.6396 | 119.6332 | [ 10.7199, | 10.8825] |
| 12 10.7583 | | 10.7820 | 119.6397 | 119.6396 | [ 10.7199, | 10.8204] |
| 13 10.7820 | | 10.7967 | 119.6383 | 119.6397 | [ 10.7583, | 10.8204] |
| 14 10.7967 | | 10.8057 | 119.6397 | 119.6400 | [ 10.7820, | 10.8204] |
| 15 10.7911 | | 10.7967 | 119.6400 | 119.6400 | [ 10.7820, | 10.8057] |
| 16 10.7967 | | 10.8002 | 119.6399 | 119.6400 | [ 10.7911, | 10.8057] |
| 17 10.8002 | | 10.8022 | 119.6400 | 119.6400 | [ 10.7967, | 10.8057] |
| 18 10.7988 | | 10.8002 | 119.6400 | 119.6400 | [ 10.7967, | 10.8022] |
| 19 10.8002 | | 10.8009 | 119.6400 | 119.6400 | [ 10.7988, | 10.8022] |
| 20 10.7995 | | 10.8002 | 119.6400 | 119.6400 | [ 10.7988, | 10.8009] |
| 21 10.8002 | | 10.8002 | 119.6400 | 119.6400 | [ 10.7995, | 10.8009] |

The midpoint of the final interval is 10.800154 and f(midpoint) = 119.640. The maximum of the function is 119.640 and the x value = 10.799805

EXPERIMENT -9



PROBLEM

3 \* x1 + 4 \* x2 + 5 \* x3 = 49 (1)

4 \* x1 + 7 \* x2 + 3 \* x3 = 51 (2)

5 \* x1 + 6 \* x2 + 5 \* x3 = 61 (3)

eg: %%

mx = [ 3 4 5 ; consts = [49;51;61]; guess = [5 ;5; 5];

4 7 3 ;

5 6 5 ];

user has to apply the values for mx, const and guess

the system assigns the x1, x2, xn variables and sloves

the system of equations the input mx, consts and guess matrices can be easily fed in using simple file i/o commands.

Maximum error and maximum number of iterations can be set also, the default values are maxerr = 0.001 and maxit = 10;

# INPUT

% Newton Raphson Method

%18bce7178

%Mitali Chauhan

%L31-L32

clear all close all clc

format short g; mx = [ 3 4 5 ;

4 7 3 ;

5 6 5 ]; %matrix for Q1

consts = [49;51;61];

guess = [5 ;5; 5];

x = sym('x',[length(mx) 1]); m = mx \* x;

m = m - consts;

J = jacobian(m,x); iterations = 0;

maxerr = 0.001; % Maximum error setting.

maxit = 10; % Maximum number of iterations setting. herrx = inf;

current\_x = subs(x, x,guess);

%% Iteration

while herrx > maxerr && iterations < maxit iterations = iterations + 1;

c = subs(m,x,current\_x);

xelta= (J^-1)\* c; % can use the "\" also xref = current\_x;

partx = eval(-xelta);

current\_x = current\_x + partx;

herrx = max(abs((current\_x - xref) ./ current\_x)\* 100); %highest error precentage

err\_log(:,iterations) = herrx; result\_log(:,iterations) = current\_x;

end

%% Results

%% Results matrix will have the form: [iteration, x1, x2, x3,. ,xn, error ]

%%master\_log = [1:iterations; result\_log; err\_log]';

%%disp(master\_log);

# OUTPUT:

[ 1, 5/2, 7/2, 11/2, 100]

[ 2, 5/2, 7/2, 11/2, 0]

>>Newton Raphson Method