FALL SEM – (2020-21)

MAT2003

SUBMITTED BY : SRIHARSHITHA DEEPALA

REG NO : 19BCD7246

LAB NO : 4

SLOT : L6

1. Maximize Z =$4x_1 + 10x_2$

   s.t.c:

   2x1 + 1x2 < = 50

   2x1 + 5x2 < =100

   2x1 + 3x2 < = 90

   x1, x2 >= 0

CODE TYPE -1:

```
clc
clear all
No_of_Variables=2;
MaxZ= input('Max z :');
A=input('Matrix A');
B=input('Matrix B');
s=eye(size(A,1));
X=[A s B];
cost=input('Cost Matrix');
BV=No_of_Variables+1:1:size(X,2)-1;
zjCj=cost(BV)*X-cost;
zcj=[zjCj;X];
SimTable=array2table(zcj);
SimTable.Properties.VariableNames(1:size(zcj,2))={'x1','x2','s1','s2','s3','Xb'}
repeat_iteration=true;
while repeat_iteration
   if any(zjCj<0)
     fprintf('Current solution is not optimal \n')
     fprintf('Next Iteration: \n')
     disp('Previous basic variables: ')
     disp(BV)
     zEntVar=zjCj(1:end-1);
     [entrnCol,pivCol]=min(zEntVar);
```

```
      fprintf("most min in zjcj is %d \n",entrnCol)
      fprintf('The Pivot column is %d \n',pivCol)
      Xb=X(:,end);
      Col=X(:,pivCol);
      if all(Col<0)
         error('Lpp is unbounded as all entries less than zero')
      else
        for i=1:size(Col,1)
          if Col(i)>0
             minratio(i)=Xb(i)./Col(i);
          else
             minratio(i)=inf;
          end
        end
       [minrat,pvtRow]=min(minratio);
       fprintf('Minimum ratio corresponding to pivot row is %d \n',pvtRow)
       fprintf('Leaving variable is %d \n',BV(pvtRow))
   end
   BV(pvtRow)=pivCol;
   disp('New Basic Variables:')
   disp(BV)
   pvtKey=X(pvtRow,pivCol);
   X(pvtRow,:)=X(pvtRow,:)./pvtKey;
   for i=1:size(X,1)
      if i~=pvtRow
         X(i,:)=X(i,:)-X(i,pivCol).*X(pvtRow,:);
     end
   end
   zjCj=zjCj-zjCj(pivCol).*X(pvtRow,:);
   zcj=[zjCj;X];
   SimTable=array2table(zcj);

SimTable.Properties.VariableNames(1:size(zcj,2))={'x1','x2','s1','s2','s3','Xb'}
   BFS=zeros(1,size(X,2));
   BFS(BV)=X(:,end);
   BFS(end)=sum(BFS.*cost);
   CurrentBFS=array2table(BFS);

CurrentBFS.Properties.VariableNames(1:size(CurrentBFS,2))={'x1','x2','s1','s2','s3
','Xb'}
   else
     repeat_iteration=false;
     fprintf('Current Basic feasible sol is optimal \n');
   end
end
```

OUTPUT:

```
Max z :
[4 10]
Matrix A
[2 1;2 5;2 3]
Matrix B
```

```
[50;100;90]
Cost Matrix
[4 10 0 0 0 0]


SimTable =
```

4×6 **table**

x1	x2	s1	s2	s3	Xb
-4	-10	0	0	0	0
2	1	1	0	0	50
2	5	0	1	0	100
2	3	0	0	1	90

```
Current solution is not optimal
Next Iteration:
Previous basic variables:
     3     4     5


most min in zjcj is -10
The Pivot column is 2
Minimum ratio corresponding to pivot row is 2
Leaving variable is 4
New Basic Variables:
     3     2     5



SimTable =
```

4×6 **table**

x1	x2	s1	s2	s3	Xb
0	0	0	2	0	200
1.6	0	1	-0.2	0	30

```
    0.4    1     0      0.2    0      20
    0.8    0     0     -0.6    1      30


CurrentBFS =


  1×6 table

    x1    x2    s1    s2    s3    Xb
    __    __    __    __    __    ___

    0     20    30    0     30    200


Current Basic feasible sol is optimal
```

CODE TYPE- 2:

```
clear
clc
format rational
warning off
textstr1 = 'Set up augumented matrix Ab = [A b]';
disp(textstr1);
Ab = input('Enter the matrix Ab');
disp(Ab);
[m1, n1] = size(Ab);
m = m1-1;
n = n1-1;
fprintf('No.of variables [n](Including slack and syrplus variables): %3i\n
No.of constraints [m]: %3i\n\n',n,m);
nmax = 100;
tablenumber = 1;
% Creating column symbols
coltxt=[] ;
for i = 1:n1
stri = num2str(i);
catstr = strcat(' x',stri);
if(i==n1)
coltxt = [coltxt,' b'];
else
coltxt = [coltxt,catstr];
end
end
```

```
disp(coltxt);
iprintsol = 0;
%printing the table including columns
for ii =1:nmax
textstr1 = strcat('Simplex Table -',num2str(tablenumber));
fprintf('-----------------------------------------------------\n');
fprintf('%s',textstr1);
fprintf('\n-----------------------------------------------------');
fprintf('\n\n');
disp(coltxt);
for i = 1:m1
fprintf ('%10.3f',Ab(i,:));
fprintf('\n');
end
fprintf('\n\n')
   %deciding whether iterations are required or not
   [minv,ebv] = min(Ab(m1,1:n));
   ebvint = uint8(ebv);
   if(minv<0.0)
    iprintsol = 0;
    fprintf('-----------------------------------------------\n');
    fprintf('Another Iteration is Necessary!!!\n');
    fprintf('-----------------------------------------------\n');
   else
    iprintsol = 1;
    fprintf('-----------------------------------------------\n');
    fprintf('Another iteration is not possible\n');
    fprintf('Check for the solution\n');
    fprintf('-----------------------------------------------\n');
   end
   if(iprintsol == 0)
    fprintf('\nEntering Basic Variable (EBV) =
%s\n',strcat('x',num2str(ebv)));
    dummyvec(:,1) = (Ab(:,n1)./Ab(:,ebvint)).*(Ab(:,ebvint) > 0);
    %identifying the pivot row
    minval = +inf;
    pivot = 1;
      for j=1:m
        if(dummyvec(j,1)<minval && dummyvec(j,1)>0)
            minval = dummyvec(j,1);
            pivotrow = j;
        end
      end
    fprintf('Pivot Row : %3i\n',pivotrow);
    %Assuming leaving basic variable is row -1
    lbv = 1;
    %Creating unit vector to identify lbv
    unitvec = zeros(m1,1);
    unitvec(pivotrow,1)=1;
    %checking the columns and updating lbv
      for i=1:n
        if(Ab(:,i)==unitvec)
            lbv = i;
        end
```

```
      end
    fprintf('Leaving Basic Variable (LBV) =
%s\n',strcat('x',num2str(lbv)));
    Ab(pivotrow,:)=Ab(pivotrow,:)/Ab(pivotrow,ebv);
      for j = 1:m1
        if (j~=pivotrow)
            Ab(j,:)=Ab(j,:)-Ab(j,ebv)*Ab(pivotrow,:);
        end
      end
    tablenumber = tablenumber + 1;
  end
  if(iprintsol == 1)
      %Recognizing Solution
    isol = [];
    sol = [];
    if(Ab([1:m],n1)>0)
        %Creating unit vector to identify columns
        for k = 1:m
            unitvec = zeros(m1,1);
            unitvec(k,1)=1;
            for i = 1:n
                %identifying solution and variables
                if(Ab(:,i)==unitvec)
                    isol=[isol i];
                    sol = [sol Ab(k,n1)];
                end
            end
        end
        for ip = 1:m
            fprintf('\nx(%2i)=%7.3f',isol(ip),sol(ip));
        end
        fprintf('\n');
        fprintf('Optimal Value : %10.3f',Ab(m1,n1));
        fprintf("\n\n");
        return
    else
        fprintf('Infeasible Solution solution exists \n')
    end
  end
end
```

OUTPUT :

```
Set up augumented matrix Ab = [A b]
Enter the matrix Ab
[2 1 1 0 0 50;2 5 0 1 0 100;2 3 0 0 1 90;-4 -10 0 0 0 0]
```

```
  Columns 1 through 5

       2             1             1             0             0
       2             5             0             1             0
       2             3             0             0             1
      -4           -10             0             0             0

  Column 6

      50
     100
      90
       0

No.of variables [n](Including slack and syrplus variables):   5
 No.of constraints [m]:   3

        x1        x2        x3        x4        x5        b
----------------------------------------------------
Simplex Table -1
----------------------------------------------------

        x1        x2        x3        x4        x5        b
     2.000     1.000     1.000     0.000     0.000    50.000
     2.000     5.000     0.000     1.000     0.000   100.000
     2.000     3.000     0.000     0.000     1.000    90.000
    -4.000   -10.000     0.000     0.000     0.000     0.000


-----------------------------------------------
Another Iteration is Necessary!!!
-----------------------------------------------

Entering Basic Variable (EBV) = x2
Pivot Row :   2
Leaving Basic Variable (LBV) = x4
----------------------------------------------------
Simplex Table -2
----------------------------------------------------

        x1        x2        x3        x4        x5        b
     1.600     0.000     1.000    -0.200     0.000    30.000
     0.400     1.000     0.000     0.200     0.000    20.000
     0.800     0.000     0.000    -0.600     1.000    30.000
```

```
     0.000     0.000     0.000     2.000     0.000   200.000


---------------------------------------------
Another iteration is not possible
Check for the solution
---------------------------------------------

x( 3)= 30.000
x( 2)= 20.000
x( 5)= 30.000
Optimal Value :    200.000
```