IE535 – LINEAR PROGRAMMING – FALL 2017

PROGRAMMING PROJECT REPORT

NAME: GOPI VARMA MANTHENA   PUID: 0030191751

PROBLEM 12:

Model 12: Alloy (Blending)

A company desires to blend a new alloy of 40 percent tin, 35 percent zinc, and 25 percent lead from several available allows having the following properties:

	Alloy				
Property	**1**	**2**	**3**	**4**	**5**
Percentage tin	60	25	45	20	50
Percentage zinc	10	15	45	50	40
Percentage lead	30	60	10	30	10
Cost ($/lb)	19	17	23	21	25

The objective is to determine the proportions of these alloys that should be blended to produce the new alloy at a minimum cost. Formulate the linear programming model for this problem.

LP formulation:

$$Minimize \quad 19x_1 + 17x_2 + 23x_3 + 21x_4 + 25x_5$$
$$s.t \quad 60x_1 + 25x_2 + 45x_3 + 20x_4 + 50x_5 = 40$$
$$10x_1 + 15x_2 + 45x_3 + 50x_4 + 40x_5 = 35$$
$$30x_1 + 60x_2 + 10x_3 + 30x_4 + 10x_5 = 25$$
$$x_1, x_2, x_3, x_4, x_5 \geq 0$$
$$where\ x_i\ is\ the\ amount\ of\ alloy\ i\ in\ the\ blend\ \ \forall\ i \epsilon 1,2,3,4,5$$

LP code:

```
clc
clear all
C = [19 17 23 21 25];
A = [60 25 45 20 50;10 15 45 50 40;30 60 10 30 10];
b = [40;35;25];
% m - number of constraints
% n - number of variables
[m n] = size(A);
% Checking whether the constraint matrix has redundant rows or not
if rank(A)==m
    fprintf('The constraint matrix has full row rank \n')
% If it has redundant rows, the code below will remove them
Else
      fprintf('The constraint matrix doesnt have full row rank and there are
      redundant rows \n')
    for i=1:m
        for j=i+1:m
            if rank([A(i,:);A(j,:)]) < m-1
                % Combining the rows before and after the redundant row
                % This is same as removing the redundant row
                A = A([1:j-1,j+1:end],:);
            end
```

```
        end
    end
% The constraint matrix is updated after removing redundant rows
fprintf('The updated constrained matrix after removing redundant rows is \n')
    disp(A)
end
found = 0;
%In this code we don't need to check initial basis
%because we are anyway doing phase 1
%to check for feasibility irrespective of finding initial basis
% Code for Phase-1
% This is executed only if initial basis not found
if found == 0
    fprintf('initial basis not found \n\n')
    % Adding artificial variables
    A = [A eye(m)];
    % Cost function for phase 1
    c1 = [ zeros(1,n) -max(eye(m))];
    % Initial basis
    basis=[];
        for u=1:m
         basis = [basis n+u];
    end
    fprintf('The current variables in the basis are \n')
    display(basis)
    % creating the tableau format
    tab1 = [c1 0];
    for i=1:m
        tab1 = [ tab1;A(i,:) b(i,:)];
    end
    fprintf('initial tableau for phase 1 iteration 1 is')
    display(tab1)
    % making the z values of the basic variables 'zero' initially
    for j=2:m+1
        tab1(1,:)= tab1(1,:)+tab1(j,:);
    end
    optimal = 0;
    iter = 0;
    % Loop for running iterations until the optimal solution is reached
    while(optimal==0)
        iter = iter+1;
        % Choosing the maximum z value which will enter the basis
        % Here bland's rule automatically applied
        % Since 'max' will return the value with least index incase of tie
        [M,I] = max(tab1(1,1:m+n));
        % ratio test for choosing the minimum positive value
        % choosing which variable is leaving the basis
        ratiot = tab1(2:m+1,m+n+1)./tab1(2:m+1,I);
  ratiotc=ratiot;
        if isempty(ratiot(ratiot>0))==1
            fprintf('The LP is unbounded \n')
            break;
        end
        % Here bland's rule automatically applied
        % Since 'min' will return the value with least index incase of tie
        [H] = min(ratiot(ratiot>0));
        G=find((ratiot==H),1);
```

```
            basis(G)=I;
        fprintf('The current variables in the basis are \n')
        display(basis)
        % Row transformation of the pivot variable to make it '1'
        tab1(G+1,:)=tab1(G+1,:)./tab1(G+1,I);
        % Row transformations to make other rows '0' using the pivot row
        for i=1:m+1
            % Transforming rows other than pivot row
            % Since pivot row is already transformed
            if(i~=G+1)
                tab1(i,:)=tab1(i,:)-(tab1(G+1,:)*tab1(i,I));
            end
        end
        fprintf('The tableau for iteration %d',iter+1)
        display(tab1)
        % Checking if all 'z' are negative to obtain an optimal solution
        if (isempty(tab1(tab1(1,1:m+n)>0.000000001))==1)||(tab1(1,m+n+1)<=0)
            optimal=1;
        end
    end
    % If at optimal solution, RHS not equal to zero
    % It becomes infeasible
    if tab1(1,m+n+1)~=0
        fprintf('The LP is infeasible')
    % Otherwise, we can proceed to Phase-2
    else
        fprintf('The LP is feasible and proceed to phase 2')

        % If we have an initial basis we will directly come to Phase-2
        % If we have a feasible solution after Phase-1, we come to Phase-2
        % Beginning of Phase-2
        fprintf('The current variables in the basis are \n')
        display(basis)
        fprintf('initial tableau for phase 2 iteration 1 is')
        % Creating the initial tableau for Phase-2
        d = tab1(2:end,m+n+1);
        A = tab1(2:end,1:n);
        tab2 = [-C 0];
        for i=1:m
            tab2 = [ tab2;A(i,:) d(i,:)];
        end
        % Making the z-values of initial basis as 0
        for j=1:m
            tab2(1,:)=tab2(1,:)-tab2(j+1,:)*tab2(1,basis(j));
        end
        display(tab2)
        optimal = 0;
        iter = 0;
        % Checking if the solution is optimal before proceeding
        % This is done by checking if all z values are positive
        if isempty(tab2(tab2(1,1:n)>0.000000001))==1
            optimal=1;
        end
        % The loop goes on until an optimal solution is reached
        while(optimal==0)
            iter = iter+1;
            % Choosing the maximum z value which will enter the basis
```

```
            % Here bland's rule automatically applied
           % Since 'max' will return the value with least index incase of tie
            [M,I] = max(tab2(1,1:n));
            % Ratio test for choosing the minimum positive value
            ratiot = tab2(2:m+1,n+1)./tab2(2:m+1,I);
          % If there is no positive value in ratio test, the LP is unbounded
            if isempty(ratiot(ratiot>0))==1
                fprintf('The LP is unbounded \n')
                break;
            end
            % Here bland's rule automatically applied
          % Since 'min' will return the value with least index incase of tie
            [H,G] = min(ratiot(ratiot>0));
            basis(G)=I;
            fprintf('The current variables in the basis are \n')
            display(basis)
            % Row transformation of the pivot variable to make it '1'
            tab2(G+1,:)=tab2(G+1,:)./tab2(G+1,I);
            % Row transformations to make other rows '0' using the pivot row
            for i=1:m+1
                 % Transforming rows other than pivot row
                 % Since pivot row is already transformed
                if(i~=G+1)
                    tab2(i,:)=tab2(i,:)-(tab2(G+1,:)*tab2(i,I));
                end
            end
            fprintf('The tableau for iteration %d',iter+1)
            display(tab2)
            % Checking if all 'z' are negative to obtain an optimal solution
            if isempty(tab2(tab2(1,1:n)>0.000000001))==1
                optimal=1;
            end
        end
   % If we reach an optimal solution, printing the optimal obj function value
        if optimal==1
            fprintf('The optimal objective function value is \n')
            display(tab2(1,n+1))
            fprintf('The optimal solution \n')
            % To print the optimal solution
            for i=1:n
                u=0;
                fprintf('x%d = ',i)
                for j=1:m
                    if i==basis(j)
                        fprintf('%d \n',tab2(j+1,n+1))
                        u=1;
                    end
                end
                if u==0
                    fprintf('%d \n',u)
                end
            end
        % Printing the extreme direction incase of unboundedness
        else
         fprintf('The extreme direction for the LP which is unbounded is \n')
            for k=1:n
                v=0;
```

```
                for j=1:m
                    if k==basis(j)
                        dir(k,1)= -tab2(j+1,I);
                        v=1;
                    end
                end
                if v==0
                    if k==I
                        dir(k,1)=1;
                    else
                        dir(k,1)=0;
                    end
                end
            end
        end
    end
end
```

OUTPUT from code (MATLAB):

```
The constraint matrix has full row rank
initial basis not found

The current variables in the basis are

basis =

     6     7     8

initial tableau for phase 1 iteration 1 is
tab1 =

     0     0     0     0     0    -1    -1    -1     0
    60    25    45    20    50     1     0     0    40
    10    15    45    50    40     0     1     0    35
    30    60    10    30    10     0     0     1    25

The current variables in the basis are

basis =

     1     7     8

The tableau for iteration 2
tab1 =

         0   58.3333   25.0000   66.6667   16.6667   -1.6667         0         0   33.3333
    1.0000    0.4167    0.7500    0.3333    0.8333    0.0167         0         0    0.6667
         0   10.8333   37.5000   46.6667   31.6667   -0.1667    1.0000         0   28.3333
         0   47.5000  -12.5000   20.0000  -15.0000   -0.5000         0    1.0000    5.0000

The current variables in the basis are

basis =

     1     7     4

The tableau for iteration 3
tab1 =

         0 -100.0000   66.6667         0   66.6667    0.0000         0   -3.3333   16.6667
    1.0000   -0.3750    0.9583         0    1.0833    0.0250         0   -0.0167    0.5833
         0 -100.0000   66.6667         0   66.6667    1.0000    1.0000   -2.3333   16.6667
         0    2.3750   -0.6250    1.0000   -0.7500   -0.0250         0    0.0500    0.2500
```

```
The current variables in the basis are

basis =

     1     3     4

The tableau for iteration 4
tab1 =

         0    0.0000         0         0   -0.0000   -1.0000   -1.0000   -1.0000         0
    1.0000    1.0625         0         0    0.1250    0.0106   -0.0144    0.0169    0.3437
         0   -1.5000    1.0000         0    1.0000    0.0150    0.0150   -0.0350    0.2500
         0    1.4375         0    1.0000   -0.1250   -0.0156    0.0094    0.0281    0.4063

The LP is feasible and proceed to phase 2The current variables in the basis are

basis =

     1     3     4

initial tableau for phase 2 iteration 1 is
tab2 =

         0   -1.1250         0         0   -2.2500   20.8125
    1.0000    1.0625         0         0    0.1250    0.3437
         0   -1.5000    1.0000         0    1.0000    0.2500
         0    1.4375         0    1.0000   -0.1250    0.4063

The optimal objective function value is

ans =

   20.8125

The optimal solution
x1 = 3.437500e-01
x2 = 0
x3 = 2.500000e-01
x4 = 4.062500e-01
x5 = 0
```

OUTPUT from excel and one more solver:

Alloy	1	2	3	4	5			
Amount to be blended	0.34375	0	0.25	0.40625	0			
Cost ($/lb)	19	17	23	21	25			
Total Cost	20.8125							
Constraints								
			Alloy					Limit
Property	1	2	3	4	5			
Percentage of tin	60	25	45	20	50	40		40
Percentage of zinc	10	15	45	50	40	35		35
Percentage of lead	30	60	10	30	10	25		25

Linear Programming Calculator

```
min
19 17 23 21 25
```

```
60 25 45 20 50 0 40
10 15 45 50 40 0 35
30 60 10 30 10 0 25
1 0 0 0 0 1 0
0 1 0 0 0 1 0
0 0 1 0 0 1 0
```

Calculate | Reset | Example

```
Solution vector x:
  0.3438
  0.0000
  0.2500
  0.4063
  0.0000
```

Number of decimal places can be 1 to 15, default is 4

Output in Scientific/Fixed Format

PROBLEM 7*: (difficult one)

Model 7: Electronics Company

An electronics company has a contract to deliver 21,475 radios within the next four weeks. The client is willing to pay $20 for each radio delivered by the end of the first week, $18 for those delivered by the end of the second week, $16 by the end of the third week, and $14 by the end of the fourth week. Since each worker can assemble only 50 radios per week, the company cannot meet the order with its present labor force of 40; hence it must hire and train temporary help. Any of the experienced workers can be taken off the assembly line to instruct a class of three trainees; after one week of instruction, each of the trainees can either proceed to the assembly line or instruct additional new classes.

At present, the company has no other contracts; hence some workers may become idle once the delivery is completed. All of them, whether permanent or temporary, must be kept on the payroll 'til the end of the fourth week. The weekly wages of a worker, whether assembling, instructing, or being idle, are $200; the weekly wages of a trainee are $100. The production costs, excluding the worker's wages, are $5 per radio. The company's aim is to maximize the total net profit. Formulate this as an LP problem (not necessarily in the standard form).

LP formulation:

$$Maximize \quad 550a_1 + 450a_2 + 350a_3 + 250a_4 - 500t_1 - 500t_2 - 500t_3 - 500t_4 - 200x_1 - 200x_2 - 200x_3 - 200x_4$$

$$s.t \qquad a_1 + t_1 + x_1 = 40$$

$$a_2 + t_2 + x_2 = a_1 + t_1 + x_1 + 3t_1$$

$$a_3 + t_3 + x_3 = a_2 + t_2 + x_2 + 3t_2$$

$$a_4 + t_4 + x_4 = a_3 + t_3 + x_3 + 3t_3$$

$$50a_1 + 50a_2 + 50a_3 + 50a_4 = 21475$$

$$t_4 = 0$$

$$a_1, a_2, a_3, a_4, t_1, t_2, t_3, t_4, x_1, x_2, x_3, x_4 \geq 0$$

$$where\ a_i\ are\ the\ number\ of\ workers\ assembling\ in\ i^{th} week\ \forall\ i \epsilon 1,2,3,4$$

$$t_i\ are\ the\ number\ of\ instructers\ in\ i^{th} week\ \forall\ i \epsilon 1,2,3,4$$

$$x_i\ are\ the\ number\ of\ idle\ workers\ in\ i^{th} week\ \forall\ i \epsilon 1,2,3,4$$

LP code: (exactly same as previous code except for C, A, b values which are the inputs)

```
clc
clear all
C = [-550 -450 -350 -250 500 500 500 500 -200 -200 -200 -200];
A = [1 0 0 0 1 0 0 0 1 0 0 0;
    -1 1 0 0 -4 1 0 0 -1 1 0 0;
     0 -1 1 0 0 -4 1 0 0 -1 1 0;
     0 0 -1 1 0 0 -4 1 0 0 -1 1;
     50 50 50 50 0 0 0 0 0 0 0 0;
     0 0 0 0 0 0 0 1 0 0 0 0];
b = [40;0;0;0;21475;0];
% m - number of constraints
% n - number of variables
[m n] = size(A);
% Checking whether the constraint matrix has redundant rows or not
if rank(A)==m
    fprintf('The constraint matrix has full row rank \n')
```

```
% If it has redundant rows, the code below will remove them
Else
      fprintf('The constraint matrix doesnt have full row rank and there are
      redundant rows \n')
    for i=1:m
        for j=i+1:m
            if rank([A(i,:);A(j,:)]) < m-1
                % Combining the rows before and after the redundant row
                % This is same as removing the redundant row
                A = A([1:j-1,j+1:end],:);
            end
        end
    end
% The constraint matrix is updated after removing redundant rows
fprintf('The updated constrained matrix after removing redundant rows is \n')
    disp(A)
end
found = 0;
%In this code we don't need to check initial basis
%because we are anyway doing phase 1
%to check for feasibility irrespective of finding initial basis
% Code for Phase-1
% This is executed only if initial basis not found
if found == 0
    fprintf('initial basis not found \n\n')
    % Adding artificial variables
    A = [A eye(m)];
    % Cost function for phase 1
    c1 = [ zeros(1,n) -max(eye(m))];
    % Initial basis
    basis=[];
        for u=1:m
         basis = [basis n+u];
    end
    fprintf('The current variables in the basis are \n')
    display(basis)
    % creating the tableau format
    tab1 = [c1 0];
    for i=1:m
        tab1 = [ tab1;A(i,:) b(i,:)];
    end
    fprintf('initial tableau for phase 1 iteration 1 is')
    display(tab1)
    % making the z values of the basic variables 'zero' initially
    for j=2:m+1
        tab1(1,:)= tab1(1,:)+tab1(j,:);
    end
    optimal = 0;
    iter = 0;
    % Loop for running iterations until the optimal solution is reached
    while(optimal==0)
        iter = iter+1;
        % Choosing the maximum z value which will enter the basis
        % Here bland's rule automatically applied
        % Since 'max' will return the value with least index incase of tie
        [M,I] = max(tab1(1,1:m+n));
        % ratio test for choosing the minimum positive value
```

```
        % choosing which variable is leaving the basis
        ratiot = tab1(2:m+1,m+n+1)./tab1(2:m+1,I);
ratiotc=ratiot;
        if isempty(ratiot(ratiot>0))==1
            fprintf('The LP is unbounded \n')
            break;
        end
        % Here bland's rule automatically applied
        % Since 'min' will return the value with least index incase of tie
        [H] = min(ratiot(ratiot>0));
        G=find((ratiot==H),1);
                basis(G)=I;
        fprintf('The current variables in the basis are \n')
        display(basis)
        % Row transformation of the pivot variable to make it '1'
        tab1(G+1,:)=tab1(G+1,:)./tab1(G+1,I);
        % Row transformations to make other rows '0' using the pivot row
        for i=1:m+1
            % Transforming rows other than pivot row
            % Since pivot row is already transformed
            if(i~=G+1)
                tab1(i,:)=tab1(i,:)-(tab1(G+1,:)*tab1(i,I));
            end
        end
        fprintf('The tableau for iteration %d',iter+1)
        display(tab1)
        % Checking if all 'z' are negative to obtain an optimal solution
        if (isempty(tab1(tab1(1,1:m+n)>0.000000001))==1)||(tab1(1,m+n+1)<=0)
            optimal=1;
        end
    end
    % If at optimal solution, RHS not equal to zero
    % It becomes infeasible
    if tab1(1,m+n+1)~=0
        fprintf('The LP is infeasible')
    % Otherwise, we can proceed to Phase-2
    else
        fprintf('The LP is feasible and proceed to phase 2')

        % If we have an initial basis we will directly come to Phase-2
        % If we have a feasible solution after Phase-1, we come to Phase-2
        % Beginning of Phase-2
        fprintf('The current variables in the basis are \n')
        display(basis)
        fprintf('initial tableau for phase 2 iteration 1 is')
        % Creating the initial tableau for Phase-2
        d = tab1(2:end,m+n+1);
        A = tab1(2:end,1:n);
        tab2 = [-C 0];
        for i=1:m
            tab2 = [ tab2;A(i,:) d(i,:)];
        end
        % Making the z-values of initial basis as 0
        for j=1:m
            tab2(1,:)=tab2(1,:)-tab2(j+1,:)*tab2(1,basis(j));
        end
        display(tab2)
```

```
    optimal = 0;
    iter = 0;
    % Checking if the solution is optimal before proceeding
    % This is done by checking if all z values are positive
    if isempty(tab2(tab2(1,1:n)>0.000000001))==1
        optimal=1;
    end
    % The loop goes on until an optimal solution is reached
    while(optimal==0)
        iter = iter+1;
        % Choosing the maximum z value which will enter the basis
        % Here bland's rule automatically applied
       % Since 'max' will return the value with least index incase of tie
        [M,I] = max(tab2(1,1:n));
        % Ratio test for choosing the minimum positive value
        ratiot = tab2(2:m+1,n+1)./tab2(2:m+1,I);
      % If there is no positive value in ratio test, the LP is unbounded
        if isempty(ratiot(ratiot>0))==1
            fprintf('The LP is unbounded \n')
            break;
        end
        % Here bland's rule automatically applied
      % Since 'min' will return the value with least index incase of tie
        [H,G] = min(ratiot(ratiot>0));
        basis(G)=I;
        fprintf('The current variables in the basis are \n')
        display(basis)
        % Row transformation of the pivot variable to make it '1'
        tab2(G+1,:)=tab2(G+1,:)./tab2(G+1,I);
        % Row transformations to make other rows '0' using the pivot row
        for i=1:m+1
             % Transforming rows other than pivot row
             % Since pivot row is already transformed
            if(i~=G+1)
                tab2(i,:)=tab2(i,:)-(tab2(G+1,:)*tab2(i,I));
            end
        end
        fprintf('The tableau for iteration %d',iter+1)
        display(tab2)
        % Checking if all 'z' are negative to obtain an optimal solution
        if isempty(tab2(tab2(1,1:n)>0.000000001))==1
            optimal=1;
        end
    end
% If we reach an optimal solution, printing the optimal obj function value
    if optimal==1
        fprintf('The optimal objective function value is \n')
        display(tab2(1,n+1))
        fprintf('The optimal solution \n')
        % To print the optimal solution
        for i=1:n
            u=0;
            fprintf('x%d = ',i)
            for j=1:m
                if i==basis(j)
                    fprintf('%d \n',tab2(j+1,n+1))
                    u=1;
```

```
                    end
                end
                if u==0
                    fprintf('%d \n',u)
                end
            end
        % Printing the extreme direction incase of unboundedness
        else
         fprintf('The extreme direction for the LP which is unbounded is \n')
            for k=1:n
                v=0;
                for j=1:m
                    if k==basis(j)
                        dir(k,1)= -tab2(j+1,I);
                        v=1;
                    end
                end
                if v==0
                    if k==I
                        dir(k,1)=1;
                    else
                        dir(k,1)=0;
                    end
                end
            end
        end
    end
end
```

OUTPUT from code (MATLAB):

The constraint matrix has full row rank

initial basis not found

The current variables in the basis are

basis =

13 14 15 16 17 18

initial tableau for phase 1 iteration 1 is

tab1 =

Columns 1 through 13

```
     0     0     0     0     0     0     0     0     0     0     0     0    -1
     1     0     0     0     1     0     0     0     1     0     0     0     1
    -1     1     0     0    -4     1     0     0    -1     1     0     0     0
     0    -1     1     0     0    -4     1     0     0    -1     1     0     0
     0     0    -1     1     0     0    -4     1     0     0    -1     1     0
    50    50    50    50     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     1     0     0     0     0     0
```

Columns 14 through 19

```
    -1    -1    -1    -1    -1     0
     0     0     0     0     0    40
     1     0     0     0     0     0
     0     1     0     0     0     0
     0     0     1     0     0     0
     0     0     0     1     0 21475
     0     0     0     0     1     0
```

The current variables in the basis are

basis =

```
    13    14    15    16     4    18
```

The tableau for iteration 2

tab1 =

Columns 1 through 16

```
  -1.0000  -1.0000  -1.0000        0  -3.0000  -3.0000  -3.0000   2.0000        0        0        0   1.0000        0        0        0        0
   1.0000        0        0        0   1.0000        0        0        0   1.0000        0        0        0   1.0000        0        0        0
  -1.0000   1.0000        0        0  -4.0000   1.0000        0        0  -1.0000   1.0000        0        0        0   1.0000        0        0
        0  -1.0000   1.0000        0        0  -4.0000   1.0000        0        0  -1.0000   1.0000        0        0        0   1.0000        0
  -1.0000  -1.0000  -2.0000        0        0        0  -4.0000   1.0000        0        0  -1.0000   1.0000        0        0        0   1.0000
   1.0000   1.0000   1.0000   1.0000        0        0        0        0        0        0        0        0        0        0        0        0
        0        0        0        0        0        0        0   1.0000        0        0        0        0        0        0        0        0
```

Columns 17 through 19

```
 -1.0200         0 -389.5000
       0         0   40.0000
       0         0         0
       0         0         0
 -0.0200         0 -429.5000
  0.0200         0  429.5000
       0    1.0000         0
```

The LP is infeasible

<u>OUTPUT from MATLAB solver and online solver:</u>

```
clc
clear all
C = [-550 -450 -350 -250 500 500 500 500 200 200 200 200];
A = [1 0 0 0 1 0 0 0 1 0 0 0;
    -1 1 0 0 -4 1 0 0 -1 1 0 0;
     0 -1 1 0 0 -4 1 0 0 -1 1 0;
     0 0 -1 1 0 0 -4 1 0 0 -1 1;
     50 50 50 50 0 0 0 0 0 0 0 0;
     0 0 0 0 0 0 0 1 0 0 0 0];
b = [40;0;0;0;21475;0];
 Ae=[];
 be=[];
 x = linprog(C,Ae,be,A,b)
```

MATLAB output:

```
Exiting: One or more of the residuals, duality gap, or total relative error
 has stalled:
         the dual appears to be infeasible (and the primal unbounded).
         (The primal residual < TolFun=1.00e-08.)
```

Online solver: