MAT 168 – Programing Project I

Missael Colin Cuevas – 914373182

October 25, 2017

Use your program to solve the following linear programs:

(1) The linear program with data

$$c = \begin{bmatrix} 1 \\ 4 \\ 1 \\ 3 \end{bmatrix}, \quad A = \begin{bmatrix} 1 & 3 & 0 & 1 \\ 2 & 1 & 0 & 0 \\ 0 & 1 & 4 & 1 \end{bmatrix}, \quad \text{and} \quad b = \begin{bmatrix} 4 \\ 3 \\ 3 \end{bmatrix}.$$

Largest Coefficient pivoting rule

A =

```
   1   3   0   1
   2   1   0   0
   0   1   4   1
```

D =

```
   1.3333   -0.3333   -0.3333         0   -0.3333
   1.6667   -1.6667    0.3333         0    0.3333
   1.6667    0.3333    0.3333   -4.0000   -0.6667
   5.3333   -0.3333   -1.3333    1.0000    1.6667
```

BA =

[x2, w2, w3]

N =

[w1, x1, x3, x4]

iter =

1

D =

```
   0.5000   -0.5000   -0.5000    2.0000    0.5000
   2.5000   -1.5000    0.5000   -2.0000   -0.5000
   2.5000    0.5000    0.5000   -6.0000   -1.5000
   9.5000    0.5000   -0.5000   -9.0000   -2.5000
```

BA =

[x2, w2, x4]

N =

[w1, w3, x1, x3]

iter =

2

D =

```
   1.0000   -2.0000   -1.0000    4.0000    1.0000
   1.0000    3.0000    2.0000   -8.0000   -2.0000
   3.0000   -1.0000         0   -4.0000   -1.0000
  10.0000   -1.0000   -1.0000   -7.0000   -2.0000
```

BA =

[x1, w2, x4]

N =

[w1, w3, x2, x3]

iter =

3

Bland’s pivoting rule

```
A =
    1    3    0    1
    2    1    0    0
    0    1    4    1

D =
    2.5000    0.5000   -2.5000         0   -1.0000
    1.5000   -0.5000   -0.5000         0         0
    3.0000         0   -1.0000   -4.0000   -1.0000
    1.5000   -0.5000    3.5000    1.0000    3.0000
BA =
[ w1, x1, w3]
N =
[ w2, x2, x3, x4]
iter =
     1

D =
    1.0000    0.2000   -0.4000         0   -0.4000
    1.0000   -0.6000    0.2000         0    0.2000
    2.0000   -0.2000    0.4000   -4.0000   -0.6000
    5.0000    0.2000   -1.4000    1.0000    1.6000
BA =
[ x2, x1, w3]
N =
[ w1, w2, x3, x4]
iter =
     2

D =
    1.3333   -0.3333   -0.3333         0   -0.3333
    1.6667   -1.6667    0.3333         0    0.3333
    1.6667    0.3333    0.3333   -4.0000   -0.6667
    5.3333   -0.3333   -1.3333    1.0000    1.6667

BA =
[ x2, x1, w3]
N =
[ w1, w2, x3, x4]
iter =
     3

D =
    1.3333   -0.3333   -0.3333         0   -0.3333
    1.6667   -1.6667    0.3333         0    0.3333
    0.4167    0.0833    0.0833   -0.2500   -0.1667
    5.7500   -0.2500   -1.2500   -0.2500    1.5000
BA =
[ x2, x1, x3]
N =
[ w1, w2, w3, x4]
iter =
     4

D =
    0.5000   -0.5000   -0.5000    0.5000    2.0000
    2.5000   -1.5000    0.5000   -0.5000   -2.0000
    2.5000    0.5000    0.5000   -1.5000   -6.0000
    9.5000    0.5000   -0.5000   -2.5000   -9.0000
BA =
[ x2, x1, x4]
 N =
[ w1, w2, w3, x3]
iter =
     5

D =
    1.0000   -2.0000   -1.0000    1.0000    4.0000
    1.0000    3.0000    2.0000   -2.0000   -8.0000
    3.0000   -1.0000         0   -1.0000   -4.0000
   10.0000   -1.0000   -1.0000   -2.0000   -7.0000
BA =
[ x1, x1, x4]
N =
[ w1, w2, w3, x2, x3]
iter =
     6
```

(2) The n-variable linear program

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} 10^{n-j} x_j \\ \text{s.t.} \quad & 2\sum_{j=1}^{i-1} 10^{i-j} x_j + x_i \le 100^{i-1}, \quad i = 1, 2, \ldots, n \\ & x_j \ge 0, \quad j = 1, 2, \ldots, n \end{aligned}$$

for the cases $n = 3, 5, 8$.

Largest Coefficient pivoting & bland's rule n=3

A =

1 0 0

20 1 0

200 200 1

D =

1 -1 0 0

80 20 -1 0

9800 200 -200 -1

100 -100 10 1

BA =

[x1, w2, w3]

N =

[w1, x2, x3]

iter =

1

D =

1.0000 -1.0000 0 0

31.0000 19.0000 0.0050 0.0050

49.0000 1.0000 -0.0050 -0.0050

590.0000 -90.0000 -0.0500 0.9500

BA =

[x1, w2, x2]

N =

[w1, w3, x3]

iter =

2

D =

1 -1 0 0

80 20 0 -1

9800 200 -1 -200

9900 100 -1 -190

BA =

[x1, w2, x3]

N =

[w1, w3, x2]

iter =

3

D =

1 -1 0 0

100 -20 0 -1

10000 -200 -1 -200

10000 -100 -1 -190

BA =

[x1, w2, x3]

N =

[w1, w3, x2]

iter =

4

Largest Coefficient pivoting & bland's rule n=5

A =

1	0	0	0	0
20	1	0	0	0
200	200	1	0	0
2000	2000	2000	1	0
20000	20000	20000	20000	1

D =

1	-1	0	0	0	0
100	-20	-1	0	0	0
10000	-200	-200	-1	0	8.4703e-20
1e+06	-2000	-2000	-2000	0	-1
1e+08	-20000	-20000	-20000	-1	-20000
1e+08	-10000	-19000	-19900	-1	-19990

BA =

[x1, w2, x2, x3, x5]

N =

[w1, w3, w4, w5, x4]

iter =

8

Largest Coefficient pivoting & bland's rule n=7

A =

1	0	0	0	0	0	0
20	1	0	0	0	0	0
200	200	1	0	0	0	0
2000	2000	2000	1	0	0	0
20000	20000	20000	20000	1	0	0
200000	200000	200000	200000	200000	1	0
2000000	2000000	2000000	2000000	2000000	2000000	1

D =

1	-1	0	0	0	0	0	0
100	-20	-1	0	0	0	0	0
10000	-200	-200	-1	8.4703e-20	0	0	0
1e+06	-2000	-2000	-2000	-1	0	0	0
1e+08	-20000	-20000	-20000	-20000	-1	0	0
1e+10	-2e+05	-2e+05	-2e+05	-2e+05	-2e+05	0	-1
1e+12	-2e+06	-2e+06	-2e+06	-2e+06	-2e+06	-1	-2e+06
1e+12	-1e+06	-1.9e+06	-1.99e+06	-1.999e+06	-1.9999e+06	-1	2e+06

BA =

[x1, w2, x2, x3, x4, x5, x7]

N =

[w1, w3, w4, w5, w6, w7, x6]

iter =

12

MATLAB Code- Problem (a)

```
n=4;
m=3;
ini=0;
c = [1,4,1,3];
A = [1 3 0 1;2 1 0 0;0 1 4 1]
b = [4,3,3]';
iter = 0;
BA = sym('w',[1 3]); % Variables Basis
NA = sym('x',[1 4]); % Variables Constrain
NB = [NA BA];
A=-A;
while max(c) > 0,
[cj, col] = max(c);%choose largest coefficient
%||| [cj, col] =  find(c > 0,1,'first') |||
% If bland's pivoting rule
Acol = A(:,col);
[i, row] = max(-Acol./b); %select leaving variable
if i < 0;
   opt = -1; %unbounded
   'unbounded'
   break;
end
Arow = A(row,:);%A matrix
a = A(row,col);
A = A - Acol*Arow/a;
A(row,:) = -Arow/a;
A(:,col) = Acol/a;
A(row,col) = 1./a; %A after pivoting

brow = b(row);% b matrix
b = b - Acol*(brow)./a;
b(row) = -brow./a; %b after pivoting

P = b(row)*c(col) + ini;% P constraint add constant
value.
ini = P;

ccol = c(col);%c matrix
c = c - ccol*Arow./a;
c(col) = ccol./a;

D = horzcat(B,D) %Print D matrix
BA(row) = NA(col)  %Print Basis
N = setdiff(NB, BA)  %Print Constrain

iter = iter+1

end
```

MATLAB Code – Problem (b)

```
n=3; %(n=3,5,7)
A=zeros(n);
ini = 0;
for x = 1:n
   for y = 1:n
      if x-y > 0;
         A(x,y) = 2*(10.^ [x-1]);
      else if x == y;
         A(x,y) =   1;
      end  end
end  end
b= 100.^[0:n-1]';
c= 10.^[0:n-1];
c=fliplr(c);
iter = 0;
BA = sym('w',[1 n]); % Variables Basis
NA = sym('x',[1 n]); % Variables Constrain
NB = [NA  BA];
A=-A;
while max(c) > 0,%choose largest coefficient
[cj, col] = max(c);%choose largest coefficient
%||| [cj, col] =  find(c > 0,1,'first') |||
% If bland's pivoting rule
Acol = A(:,col);
[i, row] = max(-Acol./b); %select leaving variable
if i < 0;
   opt = -1; %unbounded
   'unbounded'
   break;
end
Arow = A(row,:);%A matrix
a = A(row,col);
A = A - Acol*Arow/a;
A(row,:) = -Arow/a;
A(:,col) = Acol/a;
A(row,col) = 1./a; %A after pivoting
brow = b(row);% b matrix
b = b - Acol*(brow)./a;
b(row) = -brow./a; %b after pivoting
b(row)*c(col);
P = b(row)*c(col) + ini; % P constraint add constant
value.
ini = P;
ccol = c(col);%c matrix
c = c - ccol*Arow./a;
c(col) = ccol./a;
B = b; %Create D (dictionary) matrix
B(n+1) = P;
D = vertcat(A,c);
D = horzcat(B,D) %Print D matrix
BA(row) = NA(col)  %Print Basis
N = setdiff(NB, BA)  %Print Constrain
iter = iter+1
end
```