Home assignment 2 (ETS061) - *Simulation*

Niclas Lövdahl <dic13nlo@student.lu.se>

Task 1

(a) See LP_1b.m.

(b) Results from LP_1b.m in Matlab.

```
x =

   16.0000
   14.8000
    2.0000
         0
   28.0000

fval =

 -140.4000

exitflag =

     1

output =

  struct with fields:

        iterations: 5
    constrviolation: 3.5527e-15
           message: 'Optimal solution found.'
         algorithm: 'dual-simplex'
     firstorderopt: 1.0658e-14

lambda =

  struct with fields:

      lower: [5×1 double]
      upper: [5×1 double]
      eqlin: []
    ineqlin: [7×1 double]
```

The total profit $Z = 1404000$ SEK and the optimal amount of packages in table 1.

	PI	**PII**	**PIII**	**PIV**	**PV**
Amount	16 000	14 800	2 000	0	28 000

Table 1: Optimal amount of packages to be sold.

(c) For implementation see 1_cd.xlsx.

	Macro lens	**Prime lens**	**Wide lens**	**PI**	**PIII**	**PI+PII+PIII**	**PIV+PV**
SP	0	0	3	3.4	2	0	1

Table 2: Shadow prices for each constraint.

(d) The shadow price changes three times in total. First at 3.8, second at 17.6 and third time at 18.2 when it becomes flattened. The conclusion is that if the demand reaches above 18.2 there is no increased profit even if it is decided to produce and sell more of this item.

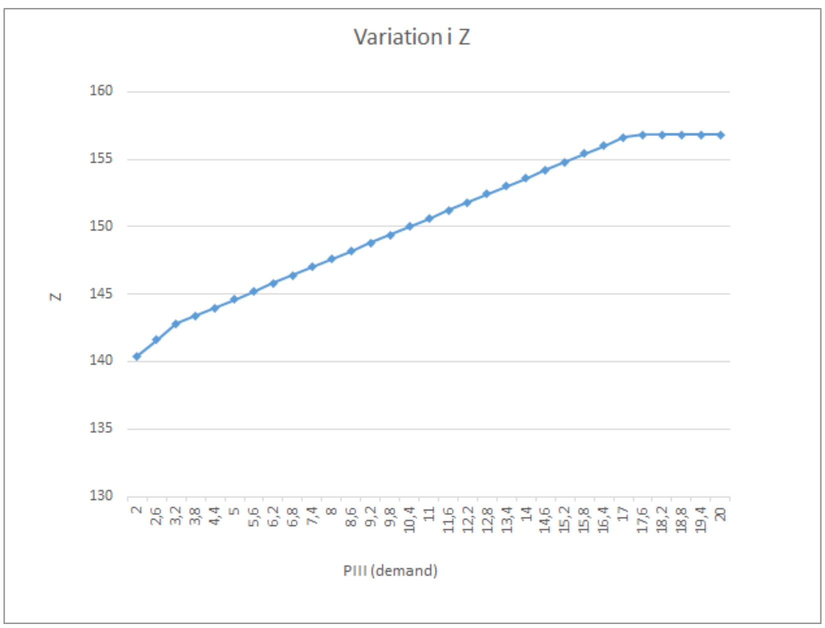

Figure 1: Relationship between the change of maximum demand for PIII and the total profit.

(e) Alpha value found to change the optimal solution but when increasing the price there is no change in the optimal solution. The conclusion is that increasing the price will not change the optimal solution since amount of PIII is prohibited by the constraints. Implementation in file LP_1e.m.

$$\alpha \leq 4 \leq \beta \text{ where } \alpha = 0.6 \text{ and } \beta = \infty$$

Task 2

(a) Results from IP_2a.m in Matlab.

```
x =

    3.0000
    2.0000

fval =

   -13

exitflag =

     1

output =

  struct with fields:

        relativegap: 0
        absolutegap: 0
      numfeaspoints: 1
           numnodes: 0
    constrviolation: 0
            message: 'Optimal solution found. Intlinprog stopped at
                the root node because the objective value is within a
                gap tolerance of the optimal value,
                options.AbsoluteGapTolerance = 0 (the default value).
                The intcon variables are integer within tolerance,
                options.IntegerTolerance = 1e-05 (the default value).'
```

(b) Results from LP_2b.m in Matlab.

```
x =

    1.6000
    2.6000

fval =

  -14.6000

exitflag =
```

```
     1

output =

  struct with fields:

        iterations: 2
    constrviolation: 0
          message: 'Optimal solution found.'
         algorithm: 'dual-simplex'
      firstorderopt: 2.2204e-16

lambda =

  struct with fields:

      lower: [2×1 double]
      upper: [2×1 double]
      eqlin: []
    ineqlin: [3×1 double]
```

Among the 9 total feasible solutions found in the given problem the maximum value for the objective function occur when $x_1 = 3$ and $x_2 = 2$ which yields $Z = 13$. This confirms that our solution in (a) is the best feasible solution for the given integer problem.

x_1	x_2	Z
0	0	0
0	1	5
1	0	1
1	1	6
1	2	11
2	0	2
2	1	7
2	2	12
3	2	13

Table 3: List of all feasible solutions.

(c) The optimal integer solution was found using branch and bound. Below in figure 2 the steps leading to the solution is depicted. Green nodes are feasible solutions that fulfills the requirement of being integer, yellow nodes are feasible solutions yet not integer and red nodes are when there was no feasible solution found. Branches were terminated on the rules of not being a feasible solution or a feasible integer solution was found. Each node were solved using structure of LP_2c.m.

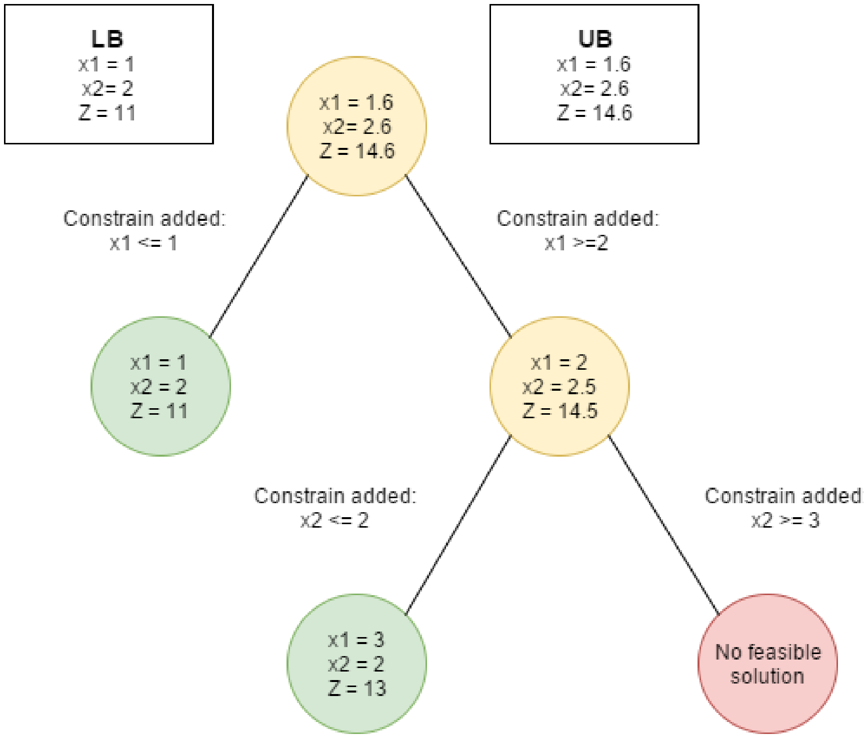

Figure 2: Branch-and-bound tree.

Code

1b

LP_1b.m

```
clc
clear all

c = [-4;
    -3;
    -2;
    -2;
    -1];

A = [2 0 0 0 0;
    0 2 2 2 1;
    0.2 1 0 0.5 0;
    1 0 0 0 0;
    0 0 1 0 0;
    1 1 1 0 0;
    0 0 0 1 1];

 b = [36;
     216;
     18;
     16;
     2;
     34;
     28];

 lb = [0;
      0;
      0;
      0;
      0];

 %options = optimoptions('linprog', 'Algorithm', 'interior-point',
     'Display', 'iter');
 %options = optimoptions('linprog', 'Algorithm', 'interior-point',
     'Display', 'off');
 %options = optimoptions('linprog', 'Algorithm', 'dual-simplex',
     'Display', 'iter');
 options = optimoptions('linprog', 'Algorithm', 'dual-simplex',
     'Display', 'off');
 [x,fval,exitflag,output,lambda] = linprog(c, A, b, [], [], lb, [], [],
     options)
```

1e

LP_1e.m

```
clc
clear all

c = [-4;
    -3;
    -2;
    -2;
    -1];

A = [2 0 0 0 0;
    0 2 2 2 5;
    0.2 1 0 0.5 0;
    1 0 0 0 0;
    0 0 1 0 0;
    1 1 1 0 0;
    0 0 0 1 1];

 b = [36;
     216;
     18;
     16;
     2;
     34;
     28];

 lb = [0;
      0;
      0;
      0;
      0];

 %options = optimoptions('linprog', 'Algorithm', 'interior-point',
     'Display', 'iter');
 %options = optimoptions('linprog', 'Algorithm', 'interior-point',
     'Display', 'off');
 %options = optimoptions('linprog', 'Algorithm', 'dual-simplex',
     'Display', 'iter');
 options = optimoptions('linprog', 'Algorithm', 'dual-simplex',
     'Display', 'off');
 [x,fval,exitflag,output,lambda] = linprog(c, A, b, [], [], lb, [], [],
     options)

 xStart = x;

 while xStart == x
    c(1,1) = c(1,1) + 0.01;
```

```
    display(c(1,1));
    [x,fval,exitflag,output,lambda] = linprog(c, A, b, [], [], lb, [],
        [], options)
end

alpha = -(c(1,1) - 0.01);
c(1,1) = -4;
x = xStart;

while xStart == x
    if c(1,1) < -10
        break
    end
    c(1,1) = c(1,1) - 0.01;
    display(c(1,1));
    [x,fval,exitflag,output,lambda] = linprog(c, A, b, [], [], lb, [],
        [], options)
end

beta = -(c(1,1) + 0.01);

disp("Alpha is: " + alpha);
disp("Beta is: " + beta);
```

2a

IP_2a.m

```
clc
clear all

c = [-1;
    -5];

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

 b = [4;
     1;
     12];

 lb = [0;
      0];

 %options = optimoptions('intlinprog', 'Display', 'iter');
 options = optimoptions('intlinprog', 'Display', 'off');
```

```
 intcon = [1;
           2];

[x, fval, exitflag, output] = intlinprog(c', intcon, A, b, [], [], lb,
    [], options)
```

2b

LP_2b.m

```
clc
clear all

c = [-1;
     -5];

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

b = [4;
     1;
     12];

lb = [0;
      0];
%options = optimoptions('linprog', 'Algorithm', 'interior-point',
    'Display', 'iter');
%options = optimoptions('linprog', 'Algorithm', 'interior-point',
    'Display', 'off');
%options = optimoptions('linprog', 'Algorithm', 'dual-simplex',
    'Display', 'iter');
options = optimoptions('linprog', 'Algorithm', 'dual-simplex',
    'Display', 'off');
[x,fval,exitflag,output,lambda] = linprog(c', A, b, [], [], lb, [], [],
    options)
```

2c

LP_2c.m

```
clc
clear all

c = [-1;
     -5];
```

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

b = [4;
     1;
     12;
     -3];

lb = [0;
      0];
%options = optimoptions('linprog', 'Algorithm', 'interior-point',
    'Display', 'iter');
%options = optimoptions('linprog', 'Algorithm', 'interior-point',
    'Display', 'off');
%options = optimoptions('linprog', 'Algorithm', 'dual-simplex',
    'Display', 'iter');
options = optimoptions('linprog', 'Algorithm', 'dual-simplex',
    'Display', 'off');
[x,fval,exitflag,output,lambda] = linprog(c', A, b, [], [], lb, [], [],
    options)
```