Matlab tutorial Introduction à l'optimisation - Fall 2014 (Author: Riccardo Scarinci)

```
%% Windows
% Command window
% Workspace

%% Operations
1+(36/2)
3.2/5.4

%% Variables
% Scalar
a = 1
b = 2
OptimalSolutionOfTheProblem = b
x_opt = b

% Scalar operation
c = a + b
d = c*(a+b)
c^2
a = a + 1

%% Useful operators
;           % suppress output
clc;        % clears all input and output from the Command Window
clear;      % remove items from workspace
a           % read output in command window

%% Matrix
% Definition
A = [1 2 3]         % vector
B = [4; 5; 6]       % vector
C = [1 2; 3 4; 5 6]  % matrix

% Element
A(2)            % access a single element
C(1,2)          % access a single element
C(1,2) = 20     % modify a single element
C(:,2)          % access a column
C(2,:)          % access a row

% Matrix operation
A*B
D = A*C

B' % transpose
A.*B'           % element operation
```

```
E = A+B'

%% Script (.m file)
% Editor window
% Write
% Run all F5
% Run selection F9

%% Function
inv([1 2; 3 4]) % Matrix inverse
                % input
A = [1 2; 3 4]
B = inv(A)      % output

zeros(3)        % Create matrix of all zeros
zeros(3,5)      % different behavior for different input
ones(3)         % Create matrix of all one
ones(3,5)       %

% linprog       % Finds the minimum of linear problem
c = [-5; -4; -6]
A =  [1 -1  1
      3  2  4
      3  2  0]
b = [20; 42; 30]
lb = zeros(3,1)

linprog(c,A,b,[],[],lb)
x_opt = linprog(c,A,b,[],[],lb)
[x_opt,fval] = linprog(c,A,b,[],[],lb) % multiple output


%% Loop
% For
% fill a row vector with increasing number
for index = 1:10
    A(index) = index;
end

A

A = 1:10

% fill by row a matrix 3x10 with increasing number
value = 1;
for indexRow = 1:3
    for indexCol = 1:10
        A(indexRow,indexCol) = value;
        value = value + 1;
    end
```

```
end

A


%% Help
% Matlab
    % top right corner
    % press F1 (location activated)

% Google
```

Question 1:

```
% (a)
% (1) Define matrices
clear;
clc;
C=[-1,1,0,0]; % objective function
B=[2,1;1,0]; % Base matrices
b=[4;2];

% The first step is to inverse the Base matrix this can be done by inv(X)
% function in MATLAB
InB=inv(B);

% The obtimal solution is thus InvB.b -->
Sol=InB*b;
CTB=[-1,0];
ValObj=CTB*Sol;


% (b) one of the basic ariables equals to zero. Therefore, the solution is
% degenerate.

% (c) Now, we determine the reduced costs of variables at the optimal solution:
% x1 and e1 are basic solutions therefore their associated reduced costs
% equl to zero:
N=[1,0;1,1];
CTN=[1,0];
RedCost=CTN-CTB*InB*N;
```

Question 2:

```
% Based on the values of columns associated with the slack variables in the optimal table, we
have, B^-1 defined as :
clc;
```

```
clear;
W=[1/2,1/5,-1;-1,0,1/2;5,-3/10,2];
InW=inv(W);

% by having B, we can obtain the coefficients of  all variables for all constraints in the original
mathematical model:
A1=InW*[1;0;0];
A2=InW*[0;1;0];
A3=InW*[0;0;1];
A4=InW*[-1;2;-1];
A5=InW*[0;1;-2];
A6=InW*[1/2;-1;5];
A7=InW*[1/5;0;-3/10];
A8=InW*[-1;1/2;2];

%Now, we calculate the original RHS that is vector (b):
RHS_Original=InW*[3;1;7];
% Here, we find the original coefficients of the variables in the objective function, first we start
with the system of linear equations
%associated with the three basic variables in the optimal solution:
A=[1/2,-1,5;1/5,0,-3/10;-1,1/2,2];
c=[2;1/10;2];
COB=A\c;
% This is the coefficient of the base variables in the model

% by having found the coefficients of the basic variables, we obtain coefficients of the other
variables in the objective function.
COBT=transpose(COB);
COFX4=COBT*[-1;2;-1];
COFX4=-2+COFX4;
COFX5=COBT*[0;1;-2];
COFX5=COFX5;
```

Question 3:

```
% Guide:
%in the matrices in matlab outputs: xa=optimal solution of part a),
%xb=optimal solution of part b) and so on. fvala=optimal objective function
%of part a) fvalb=optimal value of the objective function in part b) and
%son.
clc;
clear;

% (a)
f = [15; 5; 4];
%First we write the constraits in (<=) format then we build the matrix of
%coefficients: Relaxation problem:
```

```
A =  [-4, -1.5, -1; -1,0,0; 0,-1,0; 0,0,-1];
b = [-41000;-1300;-1250;-4000];
%the lower bounds of variables are expressed as follows:
lb = zeros(3,1);
[xa,fvala]=linprog(f,A,b,[],[],lb);

% by solving this problem we obtain the following solution
% (1300,21200,4000), fval=1.4150*e05

% (b)
% if the production exactly equals to 1000 tricycles then we must replace constraint #5
%with the following constraint X_T=1000 That is X_T<=1000 and X_T>=1000 then:
Ab =  [-4, -1.5, -1; 1,0,0; ; -1,0,0; 0,-1,0; 0,0,-1];
bb = [-41000;1000;-1000; -1250;-4000];
[xb,fvalb]=linprog(f,Ab,bb,[],[],lb);

% (c)
% in this case we must change the RHS vector , for X_T the value will be
% XT=1300-(0.12*1300)=1144
% XP=4000-(0.12*4000)=3520
bc= [-41000;-1144;-1250;-3520];
[xc,fvalc]=linprog(f,A,bc,[],[],lb);

% (d) in this case the objective function will change to the following:
fd=[15*1.04,5*1.04,4*1.04];
[xd,fvald]=linprog(fd,A,b,[],[],lb);

% (e)
% here only the price of T is affected in the objective function.
fe=[15*1.04,5,4];
[xe,fvale]=linprog(fe,A,b,[],[],lb);

%Note: even though the objective function is changed by changing one of
%costs, the optimal solution remains unchanged.
% (f)
bf= [-39800;-1300;-1250;-4000];
[xf,fvalf]=linprog(f,A,bf,[],[],lb);
```

We have kept question number four for next TP 2. Therefore, I strongly encourage students to think about this question and compare their results to the codes during the next lab session on November 7th.