

## The *simplex* method

### Introduction

---

In Moodle (Practical Classes / Work 1) you will find the files that allow the implementation of the *simplex* method in Matlab. Follows a short description of each of them:

- **Method\_Simplex.m** - is the main program from which the *simplex* method is executed.  
**Input parameters:** ---  
**Output parameters:** ---
- **Get\_data.m** - is the function that gets data from user.  
**Input parameters:** ---  
**Output parameters:**  
**n** = number of variables (scalar)  
**m** = number of constraints (scalar)  
**c** = vector with the coefficients of the variables in the objective function (1xn)  
**A** = matrix with the coefficients of the variables in the constraints (mxn)  
**b** = vector with the independent terms (mx1)
- **Present\_Simplex\_tableau.m** - is the function that builds and displays the *simplex* tableau, iteration by iteration, showing the variables that get in and out of the basis in each iteration.  
**Input parameters:**  
**n** = number of variables (scalar)  
**m** = number of constraints (scalar)  
**c** = vector with the coefficients of the variables in the objective function (1xn)  
**xB** = vector with the indexes of the basic variables (mx1)  
**cB** = vector with the coefficients of the basic variables in the objective function (mx1)  
**A** = matrix that contains the values that compose the interior of *simplex* tableau (mx(n+m))  
**b** = vector with the values of the “b” column of the *simplex* tableau (mx1)  
**zjcj** = vector that contains the values of the Zj-cj row of the *simplex* tableau (1x(n+m))  
**z** = value of the objective function (scalar)  
**iteration** = number of the iteration (scalar)  
**flag** = indicates if the table is optimal (0) or not (1)  
**VN** = index of the variable that will go to the basis in the next iteration (scalar)  
**VNB** = index of variable that will leave the basis in the next iteration (scalar)  
**Output parameters:** ---

- [Present\\_final\\_results.m](#) - is the function that presents the final results in terms of  $x^*$  and  $z^*$ .

**Input parameters:**

**n** = number of variables (scalar)

**m** = number of constraints (scalar)

**z** = value of the objective function (scalar)

**SBA** = vector that contains the values of the actual basic feasible solution  $((m+n) \times 1)$

**Output parameters:** ---

## Work to do

---

### 1<sup>st</sup> Phase

---

Complete the code of the [Method\\_Simplex.m](#) file, in order to implement the *simplex* method following the algorithm used in the Operations Research practical classes.

### 2<sup>nd</sup> Phase

---

The objective of this phase is to adapt the code developed previously, in order to make it more generic, so that it can be used by other applications of interest. That is, transform the program into a function with input/output parameters. For such:

- Transform the program previously developed ([Method\\_Simplex.m](#)) into a function [MSimplex.m](#) that receives the variables **n**, **m**, **c**, **A** and **b** as parameters, instead of calling the [Get\\_data.m](#) function.
- Create a main program that calls the [Get\\_data.m](#) function to get data of a problem from the user and that subsequently calls the new [MSimplex.m](#) function to solve the problem by the *simplex* method.
- Test the new application with the following examples:

Maximize  $z = 5x_1 + 2x_2$

Subject to

$$x_1 \leq 3$$

$$x_2 \leq 4$$

$$x_1 + 2x_2 \leq 9$$

$$x_1 \geq 0, x_2 \geq 0$$

You should achieve  $x^*=(3,3)$  and  $z^*=21$

Maximize  $z = 6x_1 + 3x_2$

Subject to

$$2x_1 + 4x_2 \leq 720$$

$$4x_1 + 4x_2 \leq 880$$

$$x_1 \leq 160$$

$$x_1 \geq 0, x_2 \geq 0$$

You should achieve  $x^*=(160,60)$  and  $z^*=1140$

- Transform the function `MSimplex.m` into a new one that returns the variables `n`, `m`, `z` and `SBA` as parameters, instead of calling the `Present_final_results.m` function.
- Modify the main program to call the `Get_data.m` function to get data of a problem from the user, to call the modified `MSimplex.m` function to solve the problem by the *simplex* method and, finally, to call the `Present_final_results.m` to present the final results.
- Test the new application with the previous examples.

### 3<sup>rd</sup> Phase

- Create a new version of the function `MSimplex.m` such that the part of the code corresponding to the initialization of the variables of the Simplex table is transferred to a new function `Initialize.m`. This last should receive as input parameters the variables `n`, `m`, `c`, `A` and `b`, and should return as output parameters the variables `A`, `c`, `x`, `xB`, `cB`, `SBA` and `zjcj`.

Update the header of `MSimplex.m` as following:

```
[n,m,A,c,b,x,xB,cB,SBA,zjcj,z]
    =MSimplex(n,m,A,c,b,x,xB,cB,SBA,zjcj);
```

The referred initialization code is the following:

```
I=eye(m);
A=[A I];
cs=zeros(1,m);
c=[c cs];

xo=1:n;
xs=n+1:n+m;
x=[xo xs];

xB=xs';
cB=cs';

SBA=[zeros(n,1);b];
zjcj=zeros(1,n+m);
```

- Move the introduction lines that are at the beginning of the `MSimplex.m` file to the beginning of `Main.m`. The referred lines are the following.

```
clc
disp('-----')
disp('          Resolution of a problem by the simplex method          ')
disp('-----')
disp(' It is assumed that:                                              ')
disp(' -> The objective function is in maximization form                ')
disp(' -> All the constraints are of type "<="                            ')
disp(' -> All the variables are >=0                                       ')
disp('-----')
```

- Modify the program `Main.m` so that the following sequence is executed:

Optimization and Decision Support Methodologies  
Practical classes 2021/2022

- Get the problem data from user ([Get\\_data.m](#))
  - Initialize the variables calling the function [Initialize.m](#)  
`[A,c,x,xB,cB,SBA,zjcj]=Initialize(n,m,c,A,b) ;`
  - Solve the problem by the *simplex* method ([MSimplex.m](#))
  - Present the final results of the resolution  
([Present\\_final\\_results.m](#))
- Test the application with the previous examples.