Università della Svizzera italiana | Institute of Computing CI

Numerical Computing **2024**

Student: Lorenzo Galli Discussed with: Gianmarco De Vita

Solution for Project 5

Due date: Wednesday, December 11, 2024, 11:59 PM

1. Graphical solution of linear programming problems [20 points]

Linear programming (LP) is part of mathematical optimization that manage problems in which the objective function and the constraints are all linear. In other words, the goal is to maximize or minimize a linear function, subject to constraints that are also expressed as linear equations or inequalities.

The Objective Function is the function that we want to optimize (maximize or minimize).
It's usually a linear combination of decision variables. For example:

$$Z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

where $c_1, c_2, \ldots, c_n$ are constant coefficients, and $x_1, x_2, \ldots, x_n$ are decision variables.

The constraints are linear conditions that must be satisfied by the decision variables. They can be both equations (i.e, $a_1x_1 + a_2x_2 = b$) or inequalities (i.e, $a_1x_1 + a_2x_2 \leq b$).

The decision variables are the unknown variables $x_1, x_2, \ldots, x_n$ or $x, y, \ldots, z$ that must be determined to optimize the objective function. Their values must satisfy the constraints.

An example of a linear programming problem could be the following:

$$\text{Maximize } Z = 3x + 2y$$

subject to:

$$x + y \leq 4$$
$$x - y \leq 1$$
$$x, y \geq 0$$

In this case, we want to find the values of x and y that maximize Z, while satisfying the constraints listed. How do we do that?

In the case of simple linear problems with a limited number of equations, solutions can often be found through manual computation. However, for more complex and large-scale linear programming problems, iterative methods, such as the Simplex Method, are typically required to find a solution.

In this section, we are required to consider two linear programming problems:

Problem 1

Minimize

$$z = 4x + y$$

s.t.:

$$x + 2y \leq 40$$

$$x + y \geq 30$$

$$2x + 3y \geq 72$$

$$x, y \geq 0$$

Problem 2

A tailor plans to sell two types of trousers, with production costs of 25 CHF and 40 CHF, respectively. The former type can be sold for 85 CHF, while the latter for 110 CHF. The tailor estimates a total monthly demand of 265 trousers. Find the number of units of each type of trousers that should be produced in order to maximize the net profit of the tailor, if we assume that the he cannot spend more than 7000 CHF in raw materials.

We will start by writing problem (2) as a linear programming problem. Then, we will solve the systems of inequalities, we will plot the feasible region identifies by the constraint and we will find the optimal solution and the value of the objective function in that point.

In order to write the problem (2) as a linear programming problem, first of all we need to set our unknowns variables:

- Let x be the number of trousers of the first type produced.
- Let y be the number of trousers of the second type produced.

The goal is to maximize the net profit, which is the difference between the earnings from the trousers and the production costs. The net profit function is:

$$z = 85x + 110y - (25x + 40y) = 60x + 70y$$

where $85x + 110y$ represents the total revenue and $25x + 40y$ represents the total cost.

Considering the monthly demand and the upper bound of production cost, our final linear program will be the following:

Maximize

$$z = 60x + 70y$$

s.t.:

$$x + y = 265$$

$$25x + 40y \leq 7000$$

$$x, y \geq 0$$

Now, we can solve this linear system of inequalities and plot the region identfied by the constraints by using MATLAB.

If we want, we could simplify the as:

Maximize

$$z = 60x + 70y$$

s.t.:

$$y = 265 - x$$

$$y \leq 175 - \frac{5x}{8}$$

$$x, y \geq 0$$

However, since we are using MATLAB to compute the solutions, it doesn't really make a difference to use the simplified one or not, they are both fine.

Remember that what we want to do, is to maximize the objective function Z (net profit), which depends on the number of trousers produced of type 1 (x) and type 2 (y). Specifically, we want to determine the values of x and y that maximize:

$$Z = 60x + 70y$$

while ensuring that the solution satisfies the given constraints.

To achieve this, we will plot the feasible region that represents all the combinations of x and y that satisfy the constraints (such as total demand, production cost limits, and non-negativity).

Then, we need to evaluate the value of Z at the corner points: in fact, the optimal solution will be found at one of the corner (or intersection) points of the feasible region. The best solution is guaranteed to be at one of these corner points, where the constraints intersect, because the objective function is linear, and any movement inside the feasible region would reduce the value of the objective. To find the best x and y, we compute Z for each of these points. The point with the highest value of Z will be the optimal production plan for the tailor.

This is guaranteed by the Fundamental Theorem of Linear Programming: *If a linear program admits a solution (i.e., if it is neither unfeasible nor unbounded), this solution will lie on a vertex of the polytope defined by the feasible region. In the case in which two vertices are both maximizers (or minimizers) of the objective function, then also all the points lying on the line segment between them will represent optimal solutions of the linear programming problem.*

The script that computes this can be found as *ex1_2.m* and a snippet is the following:

```
Z = [-60, -70];

A_0 = [25, 40];
b_0 = 7000;

A_1 = [1, 1];
b_1 = 265;

lower_bound = [0, 0];

[x, fval, exitflag] = linprog(Z, A_0, b_0, A_1, b_1, lower_bound);

...Plot the feasible region, interceptions and optimal solution...
```

Now, let's plot the feasible region identified by the constraints, finding the optimal solution and the value of the objective function in that point.

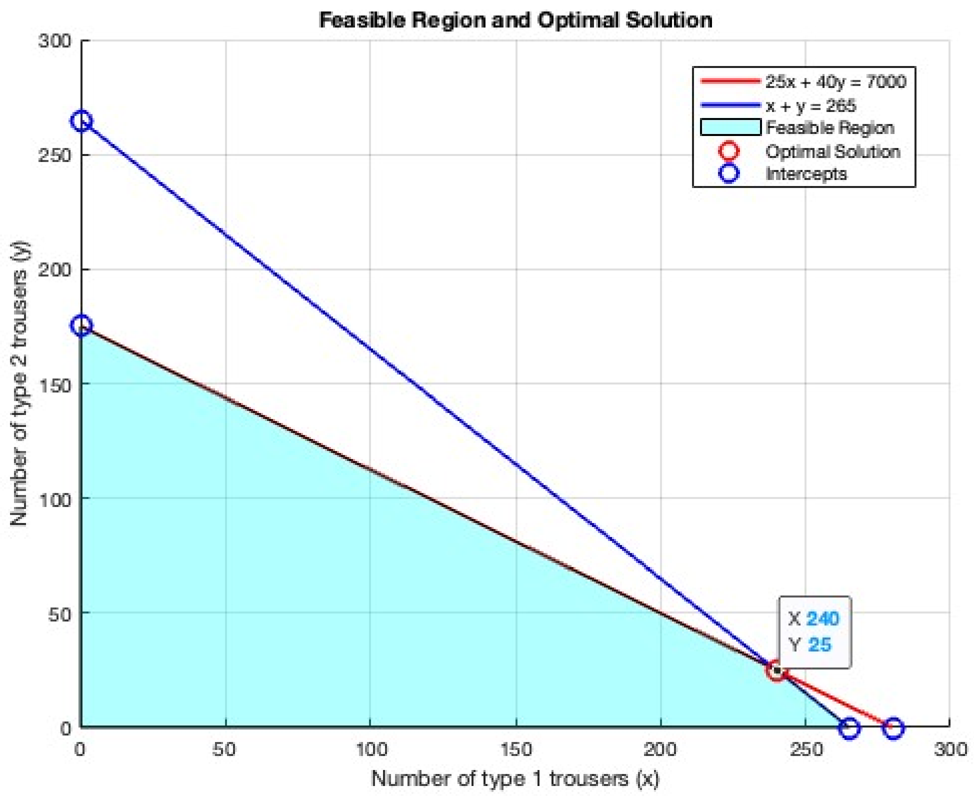


Figure 1: Plot of the optimal feasible region and best solution for the tailor problem

As we can see from the plot, there are 5 intersection points, but only 3 of them satisfy the feasible region of the plan. The optimal solution must be found among these three points, and this can be done by calculating the value of Z at each of these points:

- For $(0, 175)$:
$$Z = 60(0) + 70(175) = 0 + 12250 = 12250 \quad (CHF)$$
- For $(240, 25)$:
$$Z = 60(240) + 70(25) = 14400 + 1750 = 16150 \quad (CHF)$$
- For $(265, 0)$:
$$Z = 60(265) + 70(0) = 15900 + 0 = 15900 \quad (CHF)$$

Therefore, the optimal solution is to produce 240 trousers of type 1 and 25 trousers of type 2 to maximize the profit, with a maximum of 16.150 CHF.

We successfully found the optimal solution for problem (2), now let's get back to problem (1):

Minimize
$$z = 4x + y$$
s.t.:
$$x + 2y \leq 40$$
$$x + y \geq 30$$
$$2x + 3y \geq 72$$
$$x, y \geq 0$$

Now we know how to handle this problem, since we have just done it with a real-life problem. The only difference is that now the goal is to minimize z.

The script is pretty much the same as the problem (2), with different data. After plotting the solution, the result is the following:

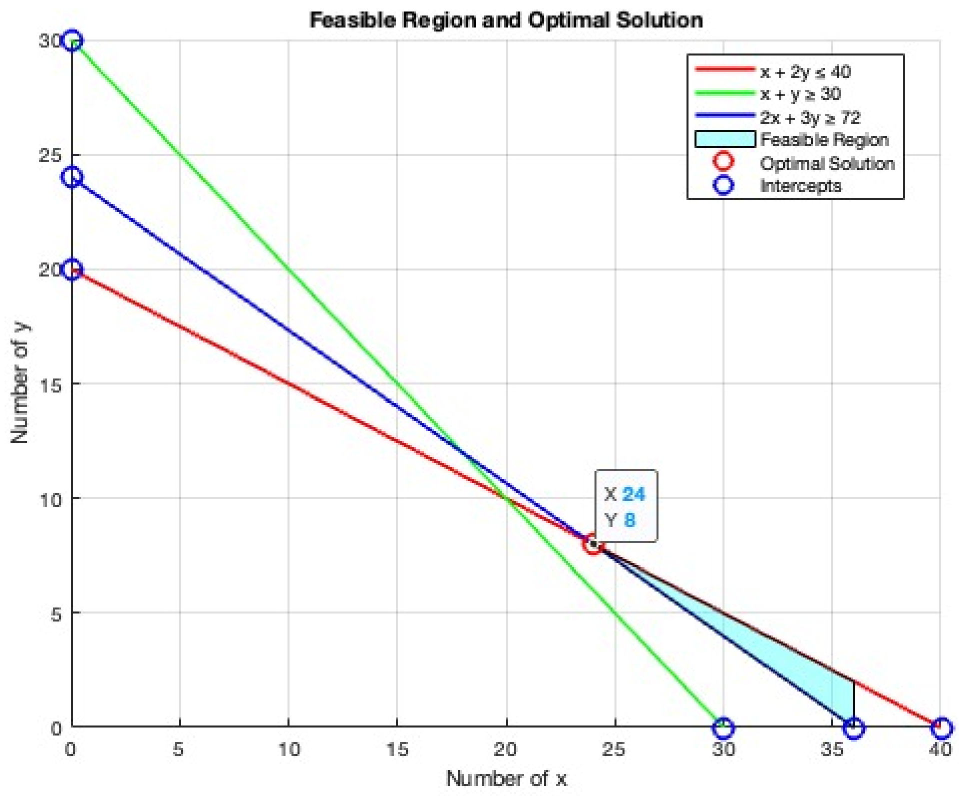


Figure 2: Plot of the optimal feasible region and best solution for problem (1)

As we can see from the plot, there are a lot of intersection points, but only 3 of them satisfy the feasible region of the plan. The optimal solution must be found among these three points, and this can be done by calculating the value of z at each of these points:

- For $(24, 8)$:
$$z = 4(24) + 8 = 96 + 8 = 104$$
- For $(36, 0)$:
$$z = 4(36) + 0 = 144 + 0 = 144$$
- For $(40, 0)$:
$$z = 4(40) + 0 = 160 + 0 = 160$$

Therefore, the optimal solution, which in this case is the minimal one, is the value 104 obtained at point (24,8).

Note: looking at the constraints, in my opinion, the region between $x = 35$ and $x = 40$ should be colored as well, but I couldn't really figure it out in MATLAB.

In this second part of the assignment, we are required to complete 2 functions which are part of a dummy implementation of the simplex method.

Before starting though, let's get quickly into a brief explanation of what the Simplex method is. The Simplex method is an iterative algorithm used to solve linear programming problems, like the ones we dealed with in the previous exercise. It's one of the most common methods for finding the optimal (maximum or minimum) solution to a linear programming problem, especially when the number of variables and constraints is large.

Now, let's understand how does this method work:

- Initially, we need to write the problem in standard form by transforming all the inequalities into equality constraints. The algorithm does that by adding slack variables (in case of a maximization problem) or surplus variables (in case of a minimization problem). These additional variables are also introduced in the objective function, with a coefficient equal to 0. Than, the Simplex method begins with a guess of a feasible basic solution, which is a solution that satisfies all the constraints.
- In each iteration, the method tries to improve the current solution by moving along the vertices (or nodes) of the feasible solution polytope (the set of all solutions that satisfy the constraints). In order to do that, it exchanges one basic and non basic variable. At this point, the gradient of the objective function is calculated, to check whether the current solution is optimal. If it's not, the method tries a new solution that is expected to improve the value of the objective function.
- This process continues until an optimal solution is reached, which is a solution that cannot be further improved. If the objective function is increasing, the method continues iterating until it reaches the maximum or minimum.

If the problem is a maximization problem, the method moves along the feasible solution polytope to find the point that maximizes the objective function, otherwise, where the point minimizes the function.

A generalized idea of the algorithm, could be the following:

```
- Algorithm Simplex Method -

Write the problem in standard form and add slack/surplus variables

Consider a feasible starting basic solution (through solution of an
   auxiliary problem)

while the optimality criterion is not satisfied do
    Apply the iterative rule by exchanging one basic and nonbasic
       variable.
end while

return the basic solution satisfying the optimality criterion
```

Before implementing the algorithm, I think it's crucial, at least it is for me, to deeply understand what is the approach of this algorithm.

Guided by pages 17-22 of *linear_programming_simplex_method.pdf*, I will provide a really clear and understandable version of an example, that helped me to understand every step. If you are already familiar with this algorithm, I invite you to skip to page 12.

The Simplex Method: How does it actually work?

Suppose we want to find the maximum value of

$$z = 4x_1 + 6x_2,$$

subject to the following constraints:

$$-x_1 + x_2 \leq 11,$$

$$x_1 + x_2 \leq 27,$$

$$2x_1 + 5x_2 \leq 90,$$

$$x_1 \geq 0, \quad x_2 \geq 0.$$

Since the left-hand side of each inequality is less than or equal to the right-hand side, there must exist non-negative numbers s_1, s_2, and s_3 that can be added to the left-hand side of each equation to produce the following system of linear equations:

$$-x_1 + x_2 + s_1 = 11$$

$$x_1 + x_2 + s_2 = 27,$$

$$2x_1 + 5x_2 + s_3 = 90.$$

The numbers s_1, s_2, and s_3 are called slack variables.

A basic solution of a linear programming problem in the standard form is basically a solution $(x_1, x_2, ..., x_n, s_1, s_2, ..., s_m)$ of the constraint equations in which at most m variables are nonzero. The variables that are nonzero are called **basic variables**. A basic solution for which all variables are non negative is called a **basic feasible solution**.

The simplex method is carried out by performing elementary row operations on a matrix that we call the **simplex tableau**. For instance, the simplex tableau for the linear programming problem in the example above is:

$$\begin{array}{cccccc c}
x_1 & x_2 & s_1 & s_2 & s_3 & b & \textbf{Basic variables} \\
\hline
-1 & 1 & 1 & 0 & 0 & 11 & s_1 \\
1 & 1 & 0 & 1 & 0 & 27 & s_2 \\
2 & 5 & 0 & 0 & 1 & 90 & s_3 \\
\hline
-4 & -6 & 0 & 0 & 0 & 0 & \leftarrow \textbf{Current z-value}
\end{array}$$

For this initial simplex tableau, the basic variables are s_1, s_2, and s_3, and the nonbasic variables (which have a value of zero) are x_1 and x_2. In fact, from the two columns that are farthest to the right, we see that the current solution is:

$$x_1 = 0, \quad x_2 = 0, \quad s_1 = 11, \quad s_2 = 27, \quad s_3 = 90.$$

This solution is usually the first basic feasible solution of the algorithm and is often written as:

$$(x_1, x_2, s_1, s_2, s_3) = (0, 0, 11, 27, 90).$$

The entry in the lower-right corner of the simplex tableau is the current value of z. In our first feasible solution the result is 0, since $z = 4x_1 + 6x_2$ and both x_1 and x_2 are 0. To check if the solution represented by a simplex tableau is optimal, we look at the entries in the bottom row of the tableau. If any of these entries are negative (as above), then the current solution is not optimal, otherwise it is optimal. In fact, if one or more entries are negative, it means the objective function will increase if that variables are added to the solution. What we are talking about is the reduced cost, which is the change in the objective function if a non-basic variable enters the solution.

Please note that the bottom-row entries under x_1 and x_2 are the negatives of the coefficients of x_1 and x_2 in the objective function, since we are dealing with maximization, in this example. That's why variables with negative entries would increase the function if added, but if we write the coefficients as they are, then the positive entries are the ones that would increase the function.

Pivoting

Once we have set up the initial simplex tableau for the problem, the simplex method checks the optimality and, if the current solution is not optimal, improves the current solution.

To improve the current solution, the algorithm brings a new basic variable (non zero) into the solution, called the **entering variable**. Since we can have up to m (3 in this case) basic variables, one of the current basic variables must leave, which is the **leaving variable**.

How do we choose them?

- The entering variable corresponds to the smallest (most negative) entry in the bottom row of the tableau.
- The leaving variable corresponds to the smallest non negative ratio of $\frac{b_i}{a_{ij}}$, in the column determined by the entering variable.
- The entry in the simplex tableau in the entering variable's column and the departing variable's row is called the *pivot*.

To form the improved solution, we apply the pivoting process, which is the Gauss-Jordan elimination to the column that contains the pivot. For now, remember that the current solution $(x_1 = 0, x_2 = 0, s_1 = 11, s_2 = 27, s_3 = 90)$ corresponds to a z-value of 0. To improve this solution, we determine that x_2 is the entering variable since it's the smallest entry in the bottom row (-6).

To find the leaving variable, we locate the b_i's that have corresponding positive elements in the entering variable's column and form the following ratios:

$$\frac{b_1}{a_{12}} = \frac{11}{1} = 11, \quad \frac{b_2}{a_{22}} = \frac{27}{1} = 27, \quad \frac{b_3}{a_{32}} = \frac{90}{5} = 18.$$

Here, the smallest positive ratio is 11, so s_1 is chosen as the departing variable.

$$\begin{array}{|cccccc|cc}
\multicolumn{1}{c}{x_1} & x_2 & s_1 & s_2 & s_3 & \multicolumn{1}{c}{b} & \textbf{Basic variables} & \\
\hline
-1 & \textcircled{1} & 1 & 0 & 0 & 11 & s_1 & \textbf{Leaving} \\
1 & 1 & 0 & 1 & 0 & 27 & s_2 & \\
2 & 5 & 0 & 0 & 1 & 90 & s_3 & \\
\hline
\multicolumn{1}{c}{-4} & -6 & 0 & 0 & 0 & \multicolumn{1}{c}{0} & \leftarrow \textbf{Current z-value} & \\
\multicolumn{1}{c}{} & \textbf{Entering} & & & & \multicolumn{1}{c}{} & &
\end{array}$$

The pivot is the entry in the first row and second column. Using Gauss-Jordan elimination, we want to have 0s in all the elements under the pivot. The solution is the following:

$$\begin{bmatrix} -1 & 1 & 1 & 0 & 0 & 11 \\ 1 & 1 & 0 & 1 & 0 & 27 \\ 2 & 5 & 0 & 0 & 1 & 90 \\ -4 & -6 & 0 & 0 & 0 & 0 \end{bmatrix} \xrightarrow{\text{Pivoting}} \begin{bmatrix} -1 & 1 & 1 & 0 & 0 & 11 \\ 2 & 0 & -1 & 1 & 0 & 16 \\ 7 & 0 & -5 & 0 & 1 & 35 \\ -10 & 0 & 6 & 0 & 0 & 66 \end{bmatrix}$$

After pivoting, the new tableau appears as follows:

x_1	x_2	s_1	s_2	s_3	b	**Basic variables**
-1	1	1	0	0	11	x_2
2	0	-1	1	0	16	s_2
7	0	-5	0	1	35	s_3
-10	0	6	0	0	66	← **Current z-value**

Note that x_2 has replaced s_1 in the basis column. The improved solution is:

$$(x_1, x_2, s_1, s_2, s_3) = (0, 11, 0, 16, 35),$$

with a z-value of:

$$z = 4x_1 + 6x_2 = 4(0) + 6(11) = 66.$$

However, the improved solution is not yet optimal since the bottom row still contains a negative entry (-10). For this reason, we apply another iteration of the simplex method.

This time, the smallest entry is x_1 (-10), which will be the entering variable. The smallest nonnegative ratio of $\frac{b_i}{a_{ij}}$ is 5, so the leaving variable will be s_3.

Now, we want to have all 0s except the pivot in the x_1 column. We can do that by using Gauss-Jordan elimination again:

$$\begin{bmatrix} -1 & 1 & 1 & 0 & 0 & 11 \\ 2 & 0 & -1 & 1 & 0 & 16 \\ 7 & 0 & -5 & 0 & 1 & 35 \\ -10 & 0 & 6 & 0 & 0 & 66 \end{bmatrix} \rightarrow \begin{bmatrix} -1 & 1 & 1 & 0 & 0 & 11 \\ 2 & 0 & -1 & 1 & 0 & 16 \\ 1 & 0 & -\frac{5}{7} & 0 & \frac{1}{7} & 5 \\ -10 & 0 & 6 & 0 & 0 & 66 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & \frac{2}{7} & 0 & \frac{1}{7} & 16 \\ 0 & 0 & \frac{3}{7} & 1 & -\frac{2}{7} & 6 \\ 1 & 0 & -\frac{5}{7} & 0 & \frac{1}{7} & 5 \\ 0 & 0 & -\frac{8}{7} & 0 & \frac{10}{7} & 116 \end{bmatrix}$$

Then, the new simplex tableau becomes:

x_1	x_2	s_1	s_2	s_3	b	**Basic variables**	
0	1	1	0	0	11	x_2	
0	0	3/7	1	-2/7	6	s_2	**Leaving**
1	0	-5/7	0	1/7	5	x_1	
0	0	-8/7	0	10/7	116	← **Current z-value**	
		Entering					

In this tableau, there is still a negative entry in the bottom row. So, we choose s_1 as the entering variable and s_2 as the leaving variable (the ratios are omitted), in order to proceed with the next iteration. We can now skip the pivoting part, showing directly the next tableau:

x_1	x_2	s_1	s_2	s_3	b	**Basic variables**
0	1	0	-2/3	1/3	12	x_2
0	0	1	7/3	-2/3	14	s_1
1	0	0	5/3	-1/3	15	x_1
0	0	0	8/3	2/3	132	← **Maximum z-value**

Notice that, in this tableau, there are no negative elements in the bottom row. Therefore, we have determined the optimal solution to be:

$$(x_1, x_2, s_1, s_2, s_3) = (15, 12, 14, 0, 0),$$

with:

$$z = 4x_1 + 6x_2 = 4(15) + 6(12) = 132.$$

Let's clarify one thing that could lead to confusion: the reduced costs are the entries in the bottom row of the simplex tableau for the non-basic variables. These entries tell us whether the current solution can be improved by including a non-basic variable in the solution. If any reduced cost is positive, it means that by bringing that variable in, we could increase the objective. So, the current solution is not optimal, and we need to continue. As we will see also in the code, if all reduced costs are non positive, then there are no variables that can still increase the objective function, and the solution is optimal. However, in the pdf and so in the previous example, at the beginning we used the opposite sign for the objective function coefficients (-4 and -6) and, for this reason, in this case we were looking for the reduced costs to be all negative instead.

As a consequence of the Fundamental Theorem of Linear Programming, the existence of a feasible solution implies also the existence of a feasible basic solution. Moreover, the existence of an optimal solution implies the existence of an optimal basic solution, which is the one that the algorithm is looking for.

The Simplex method algorithm basically find the solution of the linear programming problem by starting from a feasible basic solution and iterating through all the other basic solutions, whose number is:

$$N = \frac{(m+n)!}{m!\,n!}$$

Where n is the number of original decision variables and m is the number of constraints (equal to the number of slack/surplus variables).

In the worst-case scenario, the algorithm would achieve the optimal solution in the maximum possible number of iterations, which is exactly equal to the number of basic solutions, making the complexity of the algorithm possibly exponential.

If you think about it, in the example just shown, we have $n = 2$ and $m = 3$, which makes the formula:

$$N = \frac{(3+2)!}{3!\,2!} = \frac{120}{40} = 10$$

However, we found the optimal solution just after 4 iterations.

We have seen a maximization approach, but when it comes to minimizazion, the approach is similar, but with some differences. We form the augmented matrix for the given system of inequalities by subtracting surplus variables and by adding a bottom row consisting of the coefficients of the objective function. At this point, we form the transpose of this matrix. Finally, we form the dual maximization problem corresponding to this transposed matrix, which requires to find the maximum of the objective function given by $z = b_1y_1 + b_2y_2 + ... + b_my_m$ subject to all the constraints. In conclusion, we apply the simplex method to the dual maximization problem and the maximum value of z will be the minimum value of w. The values of $x_1, x_2, ..., x_n$ will appear in the bottom row of the final simplex tableau, in the columns corresponding to the slack variables.

One thing I didn't mention, is what we mean when we talk about solving an auxiliary problem. In a few words, in order to find a feasible starting solution (like in the example we chose s_1, s_2, s_3 as starting basic variables), we want to adopt the two-phase simplex method approach, which starts by solving an auxiliary minimization problem.

This is used especially when an initial feasible basic solution cannot be determined directly, for example in cases with inequality constraints or equality constraints, where slack variables alone are insufficient to construct a feasible starting point.

The auxiliary problem is defined as follows:

- Introduce artificial variables $a_1, a_2, \dots, a_m$ into the problematic constraints.
- Define a new objective function $W = \sum_{i=1}^{m} a_i$, which minimizes the sum of the artificial variables.
- Solve this auxiliary minimization problem using the simplex method.

If the optimal value of $W = 0$, then the artificial variables can be removed, and a feasible starting solution for the original problem has been found.

2. Implementation of the simplex method [30 points]

After having understood how the Simplex method approach works, let's get back to our implementation. We are required to complete 2 functions which are part of a dummy implementation of the simplex method. Specifically we have to complete the TODOs in:

- *standardize.m*, which writes a maximization or minimization input problem in standard form;
- *simplexSolve.m*, which solves a maximization or minimization problem using the simplex method.

We have talked a lot about the theory of the method, so understanding the implementation should be pretty straightforward. For simplicity, I will include only the updated part of the code for each implementation. The full implementations can be found in the project folder.

```
Simplex.m
...

% TODO: Compute the maximum number of basic solutions
itMax = factorial(n+m) / (factorial(n) * factorial(m));

...

% TODO: Compute the value of the objective function
z = round(c_B * x_B);

...
```

```
Standardize.m
...

% TODO: Correction on the sign of h for auxiliary problem
for i = 1:m
    if(h(i)<0)
        A(i,:) = -A(i,:);
        h(i,:) = -h(i,:);
        aug_matrix(i,:) = -aug_matrix(i,:);
    end
end
...

c_aug = [c, zeros(1,m)];
if(strcmp(type,'max'))
    % TODO: Extend matrix A by adding the slack variables
    A_aug = [A, aug_matrix];
elseif(strcmp(type,'min'))
    % TODO: Extend matrix A by adding the surplus variables
    A_aug = [A, -aug_matrix];
else
    ...
...
end
```

```
SimplexSolve.m
...
% TODO: Compute the reduced cost coefficients
r_D = c_D - c_B * BiD;
...

% TODO: Check the optimality condition, if already optimal
if(strcmp(type,'max'))
    optCheck = all(r_D <= 0);
elseif(strcmp(type,'min'))
    optCheck = all(r_D >= 0);
...

while(optCheck~=tol)
    if optCheck
        break;
    end

    % TODO: Find the index of the entering variable
    if(strcmp(type,'max'))
        [~, idxIN] = max(r_D);
    elseif(strcmp(type,'min'))
        [~, idxIN] = min(r_D);
    ...
    % TODO: Find coefficients ratio for the entering variable column
    ratio = Bih./BiD(:,idxIN);

    % TODO: Find the smallest positive ratio
    idxOUT = find(ratio == min(ratio(ratio >= 0)), 1);
    ...
    % TODO: Update the matrices by exchanging the columns
    B(:,idxOUT) = in;
    D(:,idxIN) = out;
    c_B(1,idxOUT) = c_in;
    c_D(1,idxIN) = c_out;
    index_B(1,idxOUT) = index_in;
    index_D(1,idxIN) = index_out;
    ...
    % TODO: Compute the reduced cost coefficients
    r_D = c_D - (c_B * BiD);

    % TODO: Check the optimality condition
    if(strcmp(type,'max'))
        optCheck = all(r_D <= 0);
    elseif(strcmp(type,'min'))
        optCheck = all(r_D >= 0);
    ...
    % TODO: Compute the new x_B
    x_B = Bih - (BiD * x_D);
...
end
```

All the tests were successfully passed by running the *testSimplex.m* algorithm.

3. Applications to a real-life example: Cargo aircraft [25 points]

In this second part of the assignment, we are required to use the simplex method implementation to solve a real-life problem taken from economics (constrained profit maximization).

A cargo aircraft has 4 compartments (indicated simply as $S_1, ..., S_4$) used to store the goods to be transported. Details about the weight capacity and storage capacity of the different compartments can be inferred from the data reported in the following table:

Compartment	**Weight Capacity (t)**	**Storage Capacity (m^3)**
$\mathbf{S_1}$	18	11830
$\mathbf{S_2}$	32	22552
$\mathbf{S_3}$	25	11209
$\mathbf{S_4}$	17	5870

The following four cargos are available for shipment during the next flight:

Cargo	**Weight (t)**	**Volume (m^3/t)**	**Profit (CHF/t)**
$\mathbf{C_1}$	16	320	135
$\mathbf{C_2}$	32	500	200
$\mathbf{C_3}$	40	630	410
$\mathbf{C_4}$	28	125	520

Any proportion of the four cargos can be accepted, and the profit obtained for each cargo is increased by 10% if it is put in S2, by 20% if it is put in S3 and by 30% if it is put in S4, due to the better storage conditions. The objective of this problem is to determine which amount of the different cargos will be transported and how to allocate it among the different compartments, while maximizing the profit of the owner of the cargo plane. Specifically we have to:

1. Formulate the problem above as a linear program: what is the objective function? What are the constraints? Write down all equations, with comments explaining what you are doing.

Knowing that we can split the cargo weights among the different aircraft compartments means that we have to take care of several aspects when we write the constraints. First of all, let's write down the objective function, which is to maximize the profit:

Maximize

$$\begin{aligned} z = &\big(135x_{11} + 200x_{12} + 410x_{13} + 520x_{14}\big) \\ &+ 1.1\big(135x_{21} + 200x_{22} + 410x_{23} + 520x_{24}\big) \\ &+ 1.2\big(135x_{31} + 200x_{32} + 410x_{33} + 520x_{34}\big) \\ &+ 1.3\big(135x_{41} + 200x_{42} + 410x_{43} + 520x_{44}\big) \end{aligned}$$

Where The x_{ij} values in the equation represent the weight (in tons) of cargo C_j allocated in aircraft compartment S_i.

Subject to the following constraints:

- **Compartment Weight Constraints** (the sum of cargo weights in each compartment must not exceed the compartment weight limit):

$$\begin{aligned}x_{11} + x_{12} + x_{13} + x_{14} &\leq 18\\ x_{21} + x_{22} + x_{23} + x_{24} &\leq 32\\ x_{31} + x_{32} + x_{33} + x_{34} &\leq 25\\ x_{41} + x_{42} + x_{43} + x_{44} &\leq 17\end{aligned}$$

- **Compartment Volume Constraints** (the sum of cargo volumes in each compartment must not exceed the compartment volume limit):

$$\begin{aligned}320x_{11} + 500x_{12} + 630x_{13} + 125x_{14} &\leq 11830\\ 320x_{21} + 500x_{22} + 630x_{23} + 125x_{24} &\leq 22552\\ 320x_{31} + 500x_{32} + 630x_{33} + 125x_{34} &\leq 11209\\ 320x_{41} + 500x_{42} + 630x_{43} + 125x_{44} &\leq 5870\end{aligned}$$

- **Cargo Weight Constraints** (the sum of weights belonging to cargo j allocated to different compartments i must be equal to (or less than) the total weight of cargo j):

$$\begin{aligned}x_{11} + x_{21} + x_{31} + x_{41} &\leq 16\\ x_{12} + x_{22} + x_{32} + x_{42} &\leq 32\\ x_{13} + x_{23} + x_{33} + x_{43} &\leq 40\\ x_{14} + x_{24} + x_{34} + x_{44} &\leq 28\end{aligned}$$

- **Non negative value Constraints** (the value of the weight or volume of the cargos can't be a negative value):

$$x_{ij} \geq 0 \quad (0 \leq i, j \leq 4)$$

Now that we have defined the objective function and all the constraints, we can move on the next point, that will deal with the actual calculation of the maximum profit.

2. Create a script *exercise*2*.m* which uses the simplex method implemented in the previous exercise to solve the problem. What is the optimal solution? Visualize it graphically and briefly comment the results obtained (are you surprised of this outcome on the basis of your data?).

The script *exercise*2*.m* is the following:

```
type = 'max';

% Right-hand side of constraints
h = [18; 32; 25; 17; 11830; 22552; 11209; 5870; 16; 32; 40; 28];

% Prices
prices = [135, 200, 410, 520];

% Objective coefficients modified by the factors
c = [];
factors = [1, 1.1, 1.2, 1.3];
for i=1:length(factors)
    c = [c; factors(1, i) * prices];
end
c = reshape(c, [], 1);
c = c';

% Constraints coefficients inizialization
A = zeros(12, 16);

% Compartment Weight Constraints
CWC = kron([1, 1, 1, 1], eye(4));
A(1:4, :) = CWC;

% Compartment Volume Constraints
CVC = kron([320 500 630 125], eye(4));
A(5:8, :) = CVC;

% Cargo Weight Constraints
A(9, 1:4) = 1;
A(10, 5:8) = 1;
A(11, 9:12) = 1;
A(12, 13:16) = 1;

% Indicates the direction of the inequalities in the constraints
sign = -1 * ones(1, 12);
[z, x_B, index_B] = simplex(type, A, h, c, sign);
```

The result of optimal solution has been calculated through the implemented function *simplex.m*. The result is shown in the following table.

	S_1	**S_2**	**S_3**	**S_4**	**Weight for each Cargo**
C_1	0	0	0	0	0 (<16)
C_2	18	6	0	0	24 (<32)
C_3	0	26	14	0	40 (=40)
C_4	0	0	11	17	28 (=28)
Compartment weight filling	18 (=18)	32 (=32)	25 (=25)	17 (=17)	

The result is very interesting. According to the optimal solution found by the algorithm, the maximum profit reachable is **41890 CHF**.

As we can see from the table, all the compartments have been fullfilled and all the weight limits for both the cargos and the compartments have been respected. To reach the maximum profit, we found the following combination to be the best one:

- Cargo 1 was completely ignored, not even a little weight.
- 24 tons (out of 32) of Cargo 2 were transported: 18 tons in compartment S_1 and 6 tons in S_2.
- Cargo 3 was fully transported (40 tons): 26 tons in compartment S_2 and 14 tons in S_3.
- Cargo 4 was fully transported (28 tons): 11 tons in compartment S_3 and 14 tons in S_4.

As I could expect, the cargo with the higher profit/weight ratio was placed in the compartment with the larger profit increase factor. Furthermore, since the total compartment capacity was visibly lower than the total amount of cargo weights present, cargo 1 (which guarantees the lower profit), was completely left out. Cargo 2 also have a pretty low profit/weight ratio (maybe the lower) and, for this reason, was partially left out.

4. Cycling and degeneracy [10 points]

In the last part of the project, we consider the following linear programming problem:

Maximize

$$z = 3x_1 + 4x_2$$

subject to:

$$4x_1 + 3x_2 \leq 12$$
$$4x_1 + x_2 \leq 8$$
$$4x_1 + 2x_2 \leq 8$$
$$x_2, x_2 \geq 0$$

1. Create a script *exercise*3.*m* which uses the simplex method implemented above to solve this problem. Do you achieve convergence within the maximum number of iterations (given by the maximum number of possible basic solutions)? Do you notice any strange behaviour? (hint: check, e.g., the indices of the entering and departing variables).
2. Look at the number of constraints and at the number of unknowns: what can you notice about the underlying system of equations? Represent them graphically and try to use this information to explain the behaviour of your solver in the previous point.

The problem can be (and must be) written in its vector and matrix forms in order to be solved with simplex method:

$$\mathbf{c} = \begin{bmatrix} 3 \\ 4 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 4 & 3 & 1 & 0 & 0 \\ 4 & 1 & 0 & 1 & 0 \\ 4 & 2 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{h} = \begin{bmatrix} 12 \\ 8 \\ 8 \end{bmatrix}$$

This form is basically the input form for the script. If we were to visualize it by using a simplex tableau, as in the previous examples, the problem would appear as follows:

x_1	x_2	s_1	s_2	s_3	h	**Basic variables**
4	3	1	0	0	12	...
4	1	0	1	0	8	...
4	2	0	0	1	8	...
3	4	0	0	0	0	← **objective function coefficients**

The code snippet for *exercise*3.*m* is the following:

```
type = "max";

c = [3, 4, 0, 0, 0];
A = [4, 3, 1, 0, 0; 4, 1, 0, 1, 0; 4, 2, 0, 0, 1];
h = [12; 8; 8];

sign = [-1, -1, -1];

[z, x_B, index_B] = simplex(type, A, h, c, sign);
```

However, if we run the script, we get the following result:

```
Maximum number of iterations: 56
     4
     2
     0
    ...
     4
     2
     0
Incorrect loop, more iterations than the number of basic solutions
```

What does that mean? We notice that at each iteration, the vector x_B, which is the vector of the solution values for the variables currently in the basis, never changes, so in conclusion the simplex method algorithm doesn't converge.

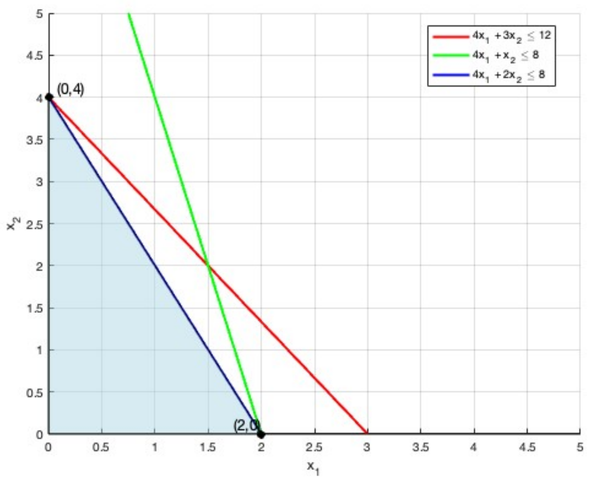
The issue we are encountering is a classic problem in the simplex algorithm, which is called cycling. Cycling occurs when the simplex algorithm finds the same solution multiple times without progressing toward optimality. In fact, The output of $x_B = [4 \quad 2 \quad 0]$ repeats across iterations, indicating that the simplex method revisits the same basic feasible solution, leading to divergence.

As we said earlier, the maximum number of iteration represents the worst-case scenario, in which the algorithm would achieve the optimal solution in the maximum possible number of iterations, which is exactly equal to the number of basic solutions, making the algorithm complexity possibly exponential. In this case, apparently, the number of all possible solutions in 56, but the algorithm doesn't converge to optimality since it's stuck with the same solution repeating many times.

The cycling behavior can happen for several reasons:

- Degeneracy: this occurs when multiple basic feasible solutions have the same objective function value, and it can cause the algorithm to get "stuck" in a loop, revisiting solutions without making progress. In particular, in linear programming, degeneracy occurs when a basic feasible solution has one or more basic variables equal to zero.
- Pivoting: the way the algorithm selects the entering and leaving variables can lead to cycling sometimes. For example, this might happen when we encounter ties in the reduced cost coefficients or in the minimum ratio comparison.

We can represent the constraints graphically, to have a better understanding of the reasons behind this cycling behavior:



What is the problem with our constraints? Since there are 3 constraints and only 2 variables, there is a chance that some of the constraints are linearly dependent, meaning they don't define distinct regions but rather intersect along the same line or point. However, we can easily check with Gauss elimination that the system has full rank, so the constraints are actually linear independent.

In fact, the problem seems to be another one: our constraints might be degenerate because they share the same coefficients for x_1. This can cause several constraints intersecting at the same points, which leads to multiple basic feasible solutions with identical z-values, where multiple constraints are satisfied at the same vertex.. In fact, it's probably that the algorithm repeatedly selects the same basic variables x_1, x_2 and one of the slack variables, creating the cycling behavior. We can see that $x_B = [4 \quad 2 \quad 0]$ represents a degenerate basic feasible solution. In fact, at this solution, $x_1 = 4, x_2 = 2,$ and one slack variable is 0 (for example $s_1 = 0$).

We could solve this problem by modifying the pivot selection strategy, for example selecting the entering and leaving variables with the smallest index when there are ties. Otherwise, we could add a small perturbation (ϵ) to the right-hand side (h) values to prevent ties and degenerate solutions from repeating.

Sources:

The Art of Linear Programming

Intro to Simplex Method - Solve LP - Simplex Tableau

4.2: Maximization By The Simplex Method

linear_programming_simplex_method.pdf