PART 1

Programmation

1

Linear Programming

1.1. Introduction

Linear programming can be defined as a mathematical technique for solving management problems. For example, suppose that a manager faced with various options wishes to determine the optimal way to use the resources of a company to achieve a specific objective, such as maximizing the utility or minimizing the cost. The company's problems can usually be modeled as a linear program (LP) consisting of a certain number of resources [HIL 90]. For instance, the labor, raw materials and capital are all resources available in limited quantities that must be distributed optimally over various manufacturing processes. The approach to solving this type of problem is divided into two key steps:

– model the problem with linear equations or inequalities that allow us to properly identify and structure the constraints satisfied by the variables of the model. The contribution of each variable to the company's overarching objective must also be defined as a function that will be optimized;

– find the mathematical optimum using specific linear programming techniques. We will study three methods for solving the various types of linear programming problems. The first is based on graphical solving and is therefore limited to two or three variables. The second method is more algebraic; it motivates the third method presented in this chapter, which is known as the simplex method (or algorithm).

1.2. Definitions

DEFINITION 1.1.– *An LP is in canonical form if it is expressed as follows:*

$$\begin{cases} \max\limits_{x \in \mathbb{R}^n} < c, x > \\ s.t. \quad Ax \leq b \\ \qquad x \geq 0 \end{cases} \qquad [1.1]$$

$A \in \mathcal{M}_{p,n}(\mathbb{R})$, $b \in \mathbb{R}^p$, *and* $x \in \mathbb{R}^n$. *We write* $x = (x_1, \ldots, x_n)^T \geq 0$ *if* $x_1, \ldots, x_n \geq 0$.

An LP is in standard form if it is expressed as follows:

$$\begin{cases} \max\ c^T x \\ \text{s.t.}\ \ Ax = b \\ \qquad x \geq 0 \end{cases}$$

Every linear problem can be expressed in canonical form: $\min c^T x = -\max c^T x$.

THEOREM 1.1.– Every LP in standard form can be expressed in canonical form and vice versa.

Terminology

– The function $c^T x$ is the *objective function*, *cost function* or *loss function*.

– The components of the vector x are called *decision variables*.

– If the vector x satisfies all constraints, then x is an *admissible solution*.

– The set of all admissible solutions is the *admissible set* or *admissible region*.

– An admissible solution x^* that maximizes the loss function (i.e. $c^T x^* \geq c^T x$ for every admissible x) is called an *optimal admissible solution* or *optimal solution*.

EXAMPLE 1.1.– A factory manufactures two products P_1 and P_2 while consuming certain resources: equipment, labor and raw materials. The needs are listed in the following table. Each resource is available in limited quantities.

	P_1	P_2	Availability
Equipment	3	9	81
Labor	4	5	55
Raw materials	2	1	20

The two products P_1 and P_2 yield effective profits of 6 dhs and 4 dhs per unit. The goal is to determine which (possibly non-integer) quantities of the products P_1 and P_2 the factory should produce to maximize the total profit from selling these two products, subject to the availability of the resources.

Let x_1 and x_2 be the quantities of the products P_1 and P_2, respectively.

The LP may be expressed as follows: maximize $z(x_1, x_2) = 6x_1 + 4x_2$ subject to:

– availability of resources:

$$3x_1 + 9x_2 \leq 81$$
$$4x_1 + 5x_2 \leq 55$$
$$2x_1 + x_2 \leq 20$$

– positivity of variables: $x_1,\ x_2 \geq 0$

1.3. Geometry of the linear program

1.3.1. *Polyhedra*

DEFINITION 1.2.– *A polyhedron is a set that can be described as*

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\},$$

where A is an $m \times n$ matrix and b is a vector in $\mathbb{R}^m$. Note that the admissible set of an LP is a polyhedron.

DEFINITION 1.3.– *A polyhedron P is said to be bounded if there exists a constant c such that $\|x\| \leq c$ for every $x \in P$.*

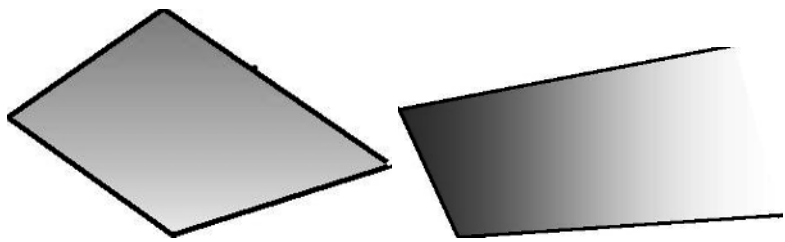

Figure 1.1. *Bounded polyhedron (left) and unbounded polyhedron (right). For a color version of this figure, see www.iste.co.uk/radi/optimizations.zip*

DEFINITION 1.4.– *Let a be a non-zero vector of $\mathbb{R}^n$, and let b be a scalar.*

– The set $\{x \in \mathbb{R}^n | a^T x = b\}$ is said to be a hyperplane.

– The set $\{x \in \mathbb{R}^n | a^T x \geq b\}$ is said to be a half-space.

Note that a hyperplane is the boundary of the corresponding half-space and the vector a is perpendicular to the hyperplane that it defines [TEG 12].

1.3.2. *Extreme points and vertices*

Let P be a polyhedron and $x^* \in P$. The point x^* is an extreme point if and only if x^* is a vertex and x^* is a basic admissible solution.

DEFINITION 1.5.– *Let P be a polyhedron. A vector $x \in P$ is an extreme point of P if it cannot be expressed as a convex combination of two other points of P.*

DEFINITION 1.6.– *Let P be a polyhedron. A vector $x \in P$ is a vertex of P if there exists c such that*

$$c^T x < c^T y$$

for every y in P not equal to x.

THEOREM 1.2.– In linear programming, if the admissible domain is neither empty nor the whole of $\mathbb{R}^n$, it is a convex polytope with finitely many vertices that can either be bounded or unbounded. If an extreme point exists, it is attained at one of the vertices of the polytope. A point in the interior of the domain is never an extreme point if $f \neq 0$. If the polytope is bounded, f attains both a minimum and a maximum on it.

This theorem gives us a graphical solving method.

1.4. Graphical solving of a linear program

The graphical method works by plotting lines and searching for a solution as follows:

– identify the admissible domain;

– identify the contours;

– contours perpendicular to the vector c, and therefore mutually parallel;

– each value of z is associated with a contour;

– the value of z increases in the direction of c.

EXAMPLE 1.2.– Consider, again, the example from earlier. Its mathematical model is defined by the following linear program:

$$\begin{cases} \max\limits_{(x_1,x_2)\in\mathbb{R}^2} & 6x_1 + 4x_2 \\ s.t. & 3x_1 + 9x_2 \leq 81 \\ & 4x_1 + 5x_2 \leq 55 \\ & 2x_1 + x_2 \leq 20 \\ & x_1,\, x_2 \geq 0 \end{cases}$$

The intersection of the half-planes determined by the lines corresponding to each constraint represents the set of solutions that satisfy the constraints (shaded area). The direction of the cost function is given by

$$z(x_1, x_2) = 6x_1 + 10x_2 = \text{constant}.$$

This gives us the optimal solution $x = \left(\frac{15}{2}, 5\right)$ by sliding the line with this direction upwards, while keeping it parallel, until its intersection with the y-axis is maximal. Graphical solutions are of course only applicable with programs of, at most, three variables (see Figure 1.2).

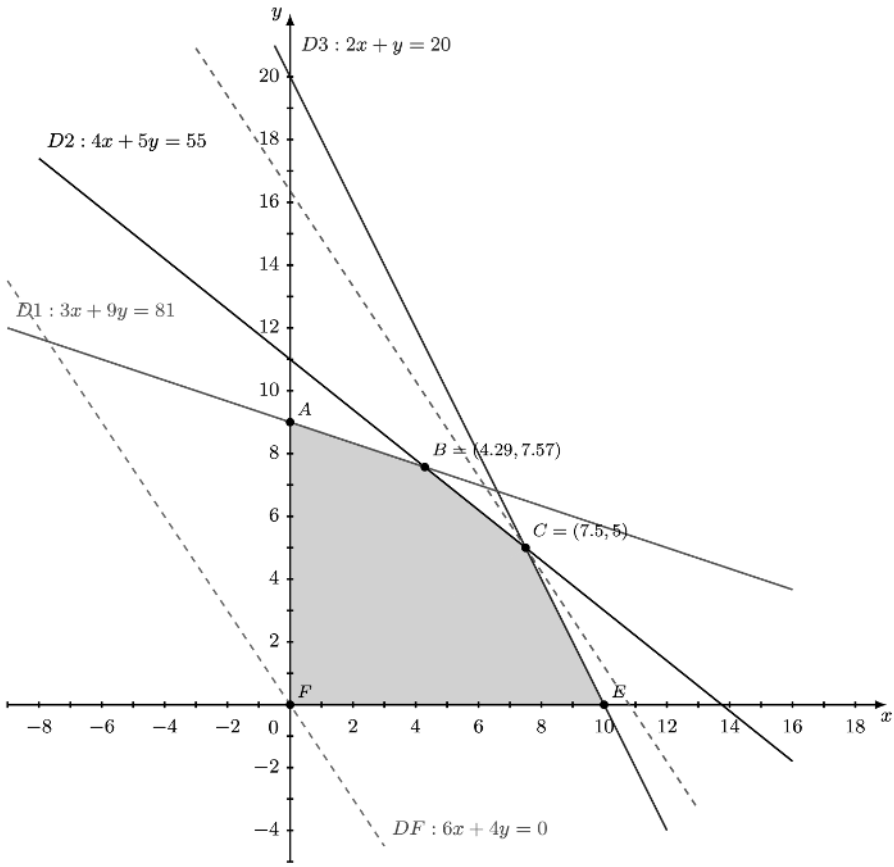

Figure 1.2. *Graphical solution. For a color version of this figure, see www.iste.co.uk/radi/optimizations.zip*

EXAMPLE 1.3.– Consider the linear program:

$$\begin{aligned} \min\ & z = -x_1 - x_2 \\ \text{s.t.}\ & x_1 + 2x_2 \leq 3 \\ & 2x_1 + x_2 \leq 3 \\ & x_1,\ x_2 \geq 0 \end{aligned} \qquad [1.2]$$

The graphical solution is shown in Figure 1.3. We have $z^* = -2$, with $x_1^* = 1$ and $x_2^* = 1$.

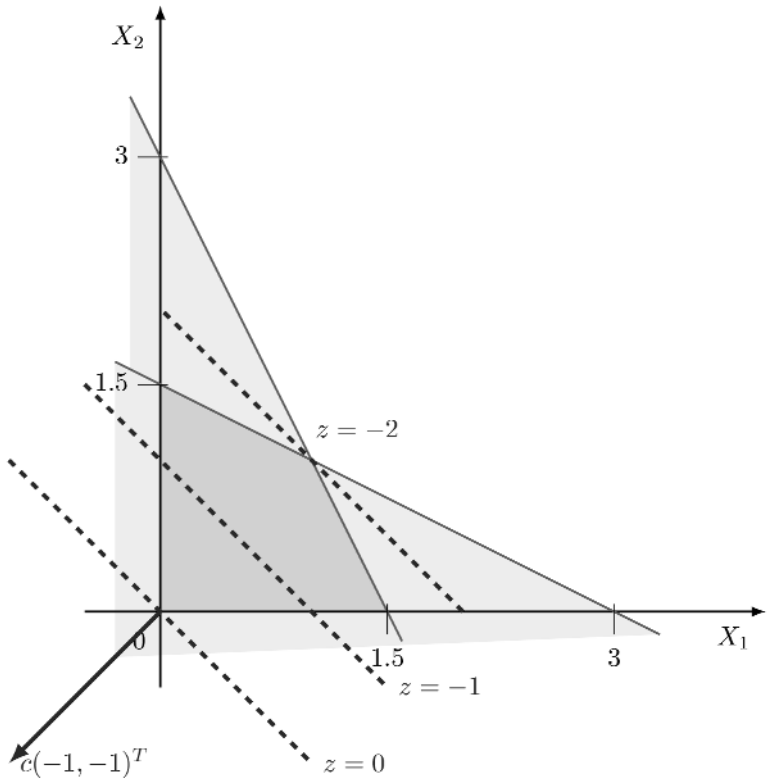

Figure 1.3. *Graphical solution with isoprofit lines. For a color version of this figure, see www.iste.co.uk/radi/optimizations.zip*

1.5. Simplex algorithm

If a linear programming problem in standard form has an optimal solution, then a basic admissible solution that is optimal exists. The simplex algorithm is an algorithm for solving linear optimization problems. Introduced by George Dantzig in 1947, it was probably the earliest algorithm for minimizing functions on sets defined by inequalities. The algorithm moves from one basic admissible solution to the next while reducing the cost.

The idea of the method is as follows:

– Let x_0 be a basic admissible solution.

– For $k = 0, \ldots,$ find an adjacent basic admissible solution x_{k+1} such that $c^T x_{k+1} < c^T x_k$.

– Until no adjacent basic admissible solution improves the objective.

This gives a local minimum, which is in fact a global minimum in linear programming.

1.5.1. *Basic solutions and basic feasible solutions*

Consider a system of m linear equations $Ax = b$ with n variables $(m \leq n)$.

DEFINITION 1.7.– *A basic solution of the system of equations $Ax = b$ is obtained by setting $n - m$ variables to zero and solving the system for the remaining p variables. The solution of the system of p equations in m unknowns is assumed to be unique. The $n - p$ variables set to zero are said to be non-basic variables, and the p remaining variables are said to be basic variables. An LP admits, at most, C_n^m basic solutions. If the basic solution also satisfies the non-negativity constraints, it is said to be a basic feasible solution.*

DEFINITION 1.8.– *A basic solution is said to be degenerate if at least one basic variable is zero. This type of solution is obtained when the number of lines passing through an extreme point is greater than the number of decision variables. A basic feasible solution whose m basic variables are positive is said to be a non-degenerate basic feasible solution.*

REMARK.– Each basic feasible solution corresponds to an extreme point. However, there can be more than one basic feasible solution for the same extreme point. This occurs when the basic feasible solution is degenerate.

REMARK.– The number of basic solutions quickly becomes very large, even in modestly sized models. For example, a model in standard form with 12 constraints and 25 variables can have up to $C_{25}^{12} = 5\,200\,300$ basic solutions.

1.5.2. *Simplex tableau*

Suppose that we wish to solve

$$\begin{cases} \max\ z(x_1,x_2,x_3,x_4) = 4x_1 + 3x_2 \\ \text{s.t.}\ \ 3x_1 + 4x_2 + x_3 = 12 \\ \qquad 7x_1 + 2x_2 + x_4 = 14, \quad x_i \geq 0\ (i = 1,\ldots,4) \end{cases}$$

Let $c = (4,3,0,0)^T$, $b = (12,14)^T$, $(A, I_2) = \begin{pmatrix} 3 & 4 & 1 & 0 \\ 7 & 2 & 0 & 1 \end{pmatrix}$. It is easy to see that $x = (x_1,x_2,x_3,x_4)^T = (0,0,12,14)^T$ satisfies the constraints. Thus, x is a basic feasible solution with basis $J = \{3,4\}$ (the third and fourth components of x), whose basic variables are $x_3 = 12$ and $x_4 = 14$.

Note that x is not optimal: $z(x) = c^T x = 0$. The optimization procedure constructs a sequence of tables, called simplex tableaus. The first tableau summarizes the data c, b, A' and the basic variables. The last row is equal to $-c^T$.

x_1	x_2	x_3	x_4	
3	4	1	0	$x_1^J = x_3 = 12$
7	2	0	1	$x_2^J = x_4 = 14$
-4	-3	0	0	$z(x) = 0$

Table 1.1. *First simplex tableau*

General case

Consider the problem

$$(P)\begin{cases} \max\ z(x) = c^T x \\ \text{s.t.}\ \ Ax = b, \\ \qquad x \geq 0 \end{cases}$$

where $x \in \mathbb{R}^n_+$, $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and A is an $m \times n$ matrix. We may assume without loss of generality that rank $A = m < n$.

DEFINITION 1.9.– *Let $A = (a^1, a^2, \ldots, a^n)$ (where a^j is the jth column of A), and, for $J \subset \{1,2,\ldots,n\}$, let $A^J = \{a^j,\ j \in J\}$.*

– $J \subset \{1,2,\ldots,n\}$ is a basis of (P) if $|J| = m$ and if rank A^J = rank $(a^j,\ j \in J) = m$;

– let $x = (x_1, \ldots, x_n)^T \in \mathbb{R}^n_+$. Then x_j is a basic variable (respectively, non-basic variable) if $j \in J$ (respectively, $j \notin J$). We write $x^J = (x_j,\ j \in J)$;

– basic feasible solution: $x = (x_1, \ldots, x_n)^T \in \mathbb{R}^n_+$ such that $x_j = 0$ for $j \notin J$ and such that $Ax = A^J x^J = b$.

REMARK.– The advantage of passing from the canonical form to the standard form is that we immediately obtain a basic feasible solution that can be used as a starting point for the simplex algorithm. The basic variables are the slack variables.

1.5.3. *Change of feasible basis*

Let x be a basic feasible solution of (P). Then

$$z(x) = c^T x = c^J x^J \quad \text{where } c^J = (c_j,\ j \in J).$$

Our goal is to find another feasible basis $\bar{J}$ and a basic feasible solution $\bar{x}$ such that $z(\bar{x}) > z(x)$ (meaning that $\bar{x}$ is better than x). The simplex method proceeds by replacing one of the basic variables x_r with a non-basic variable x_k. We say that

– the variable x_r enters the basis J: $x_r \to \bar{x}_r = 0$;

– the variable x_k leaves the basis $\bar{J}$: $x_k = 0 \to \bar{x}_k > 0$.

Thus, $\bar{J} = (J - \{r\}) \cup \{k\}$. We need rules to choose r and k. These rules are as follows:

– Choose r such that

$$\frac{x^J_r}{a_{rk}} = \min\left\{\frac{x^J_j}{a_{ik}},\ a_{ik} > 0,\ i \in J\right\}. \qquad [1.3]$$

– Choose k such that

$$z_k - c_k = \min_{j=1,\ldots,n}\{z_j - c_j,\ z_j < c_j\ \exists i \in J \mid a_{ij} > 0\}. \qquad [1.4]$$

EXAMPLE 1.4.– Returning to the previous example: $J = \{3, 4\}$, $c_3 = c_4 = 0$, and so $z_j = 0$ for $j = 1, 2, 3, 4$. Thus, $z_j - c_j = -c_j$, $j = 1, 2, 3, 4$. Hence, $k = 1$.

To choose r,

$$\frac{x^J_r}{a_{r1}} = \min\left\{\frac{x^J_i}{a_{i1}}\ ;\ a_{i1} > 0,\ i \in J\right\} = \min\left\{\frac{12}{3}, \frac{14}{7}\right\} = 2.$$

Thus, $r = 2$, x_4 leaves the basis, and x_1 enters the basis. The new basis is $\bar{J} = \{3, 1\}$. We have

$$\bar{x}_1^{\bar{J}} = \bar{x}_3 = x_1^J - \frac{a_{11}}{a_{21}} x_2^J = 12 - \frac{3}{7}14 = 6 \bar{x}_2^{\bar{J}} = \bar{x}_1 = \frac{x_2^J}{a_{21}} = 2.$$

Hence, $\bar{x} = (2, 0, 6, 0)^T$ and $z(\bar{x}) = c_1^J \bar{x}_1^{\bar{J}} + c_2^J \bar{x}_2^{\bar{J}} = c_3 \bar{x}_3 + c_1 \bar{x}_1 = 8$. Passing from $J = \{3, 4\}$ to $\bar{J} = \{3, 1\}$ increases the value of z to 8, but this value is not yet maximal.

Calculating the new tableau

We now apply the transformation $x \to \bar{x}$ that increased the value of z. Since the value $z(\bar{x})$ ($> z(x)$) is not necessarily maximal in general, we may need to repeat the steps for choosing r and k several times until we find a basic feasible solution that is also a maximum of z.

To do this, we need a second tableau where x^J is replaced by $\bar{x}^{\bar{J}}$, interpreted in the same way as the first. The same linear transformation that allowed us to pass from x to $\bar{x}$ is applied to the columns of A.

The matrix $A = (a_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n)$ is replaced by $\bar{A} = (\bar{a}_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n)$ as follows:

$$a^j = \begin{pmatrix} a_{1j} \\ \vdots \\ a_{mj} \end{pmatrix} \to \bar{a}^j = \begin{pmatrix} \bar{a}_{1j} \\ \vdots \\ \bar{a}_{mj} \end{pmatrix}, \text{where } \bar{a}_{ij} = a_{ij} - \frac{a_{ik} a_{rj}}{a_{rk}} (\text{ if } i \neq r) \text{ and } \bar{a}_{rj} = \frac{a_{rj}}{a_{rk}}. \quad [1.5]$$

This gives

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \Leftrightarrow \sum_{j=1}^{n} \bar{a}_{ij} x_j = \bar{b}_i, \quad [1.6]$$

where $\bar{b}_i$, $i = 1, \ldots, m$, are the new basic variables $\bar{x}_i^{\bar{J}}$.

The last row of the new tableau is computed in the same way:

$$\bar{z}_j - c_j = z_j - c_j - \frac{a_{rj}}{a_{rk}} (z_k - c_k). \quad [1.7]$$

EXAMPLE 1.5.– If we apply the above procedure to our example, we obtain Table 1.2.

x_1	x_2	x_3	x_4	
0	22/7	1	-3/7	$x_1^{\bar{J}} = x_3 = 6$
1	2/7	0	1/7	$x_2^{\bar{J}} = x_1 = 2$
0	-13/7	0	4/7	$z(\bar{x}) = 8$

Table 1.2. *Second simplex tableau ($\bar{J} = \{3, 1\}$)*

As we saw above, the value of z increases from 0 to 8, but there are still negative values in the last row of the tableau, so we need to perform another change of basis by applying the formulas [1.4] and [1.3] after substituting $\bar{J}$. This gives:

$$k = 2,\ \min\left\{\frac{\bar{x}_i^{\bar{J}}}{\bar{a}_{i2}}\ ;\ i = 1, 2\right\} = \min\left\{\frac{2}{2/7},\ \frac{6}{22/7}\right\} = \min\left\{7, \frac{21}{11}\right\} = \frac{21}{11},$$

$r = 1,\ x_3$ leaves the basis. $\tilde{J} = \{2, 1\}$ is the new basis.

The computation with this new basis is presented in Table 1.3.

x_1	x_2	x_3	x_4	
0	1	7/22	-3/22	$\tilde{x}_1^{\tilde{J}} = x_2 = 21/11$
1	0	-1/11	2/11	$\tilde{x}_2^{\tilde{J}} = x_1 = 16/11$
0	0	13/22	7/22	$z(\tilde{x}) = 127/11$

Table 1.3. *Third simplex tableau ($\tilde{J} = \{2, 1\}$)*

The value of z now increases from 8 to $\frac{127}{11}$. This value is maximal because every value in the final row is non-negative. The optimal solution is therefore $\tilde{x} = (\tilde{x}_1, \tilde{x}_2) = (\frac{16}{11}, \frac{21}{11})$. This solution is unique because no further change of basis is possible.

EXAMPLE 1.6.– Consider the linear problem:

$$\begin{aligned} \min\ & z = -800x_1 - 300x_2 \\ \text{s.t.}\ & 2x_1 + x_2 \leq 400 \\ & x_1 \leq 150 \\ & x_2 \leq 200 \\ & x_1, x_2 \geq 0 \end{aligned} \qquad [1.8]$$

Let us introduce slack variables x_3, x_4 and x_5:

$$\begin{array}{lrcrcrcrcl} \min\ z = & -800x_1 & - & 300x_2 & & & & & & \\ \text{s.t.} & 2x_1 & + & x_2 & + & x_3 & & & = & 400 \\ & x_1 & & & & & + & x_4 & = & 150 \\ & & & x_2 & & & & + x_5 & = & 200 \\ & x_1, & & x_2, & & x_3, & & x_5 & \geq & 0 \end{array} \qquad [1.9]$$

This gives the following initial tableau with the basis (x_3, x_4, x_5):

$$T_0 = \begin{array}{|ccccc|c|l} \multicolumn{5}{c}{x_1\ \ x_2\ \ x_3\ x_4\ x_5} & & \theta \\ \hline 2 & 1 & 1 & 0 & 0 & 400 & 200 \\ 1 & 0 & 0 & 1 & 0 & 150 & 150 \leftarrow \\ 0 & 1 & 0 & 0 & 1 & 200 & \\ \hline \text{-800} & \text{-300} & 0 & 0 & 0 & 0 & \\ \hline \end{array}$$

The new basis is (x_3, x_1, x_5) given as:

$$T_1 = \begin{array}{|ccccc|c|l} \multicolumn{5}{c}{x_1\ \ x_2\ \ x_3\ \ x_4\ x_5} & & \theta \\ \hline 0 & 1 & 1 & \text{-2} & 0 & 100 & 100 \leftarrow \\ 1 & 0 & 0 & 1 & 0 & 150 & \\ 0 & 1 & 0 & 0 & 1 & 200 & 200 \\ \hline 0 & \text{-300} & 0 & 800 & 0 & 120000 & \\ \hline \end{array}$$

The next basis is (x_2, x_1, x_5) given as:

$$T_2 = \begin{array}{|ccccc|c|} \multicolumn{5}{c}{x_1\ x_2\ \ x_3\ \ \ x_4\ x_5} & \\ \hline 0 & 1 & 1 & \text{-2} & 0 & 100 \\ 1 & 0 & 0 & 1 & 0 & 150 \\ 0 & 0 & \text{-1} & 2 & 1 & 100 \\ \hline 0 & 0 & 300 & 200 & 0 & 150000 \\ \hline \end{array}$$

Since the reduced costs are all positive, this tableau is optimal. The optimal solution is $x_1 = 150$ and $x_2 = 100$, giving an optimal value of $z = -1,500,000$ for the objective function.

1.5.4. *Existence and uniqueness of an optimal solution*

After writing out the first simplex tableau and adding $z_j = -c_j$ to the last row, there are three possible cases:

1) $z_j - c_j \geq 0$ for $j = 1, \ldots, n$. The basic feasible solution x is already optimal. This optimal solution may or may not be unique;

2) there exists $j \in \{1, \ldots, n\}$ such that $z_j - c_j < 0$ and $a_{ij} \leq 0$ for every $i \in J$. In this case, the domain of realizable solutions is unbounded and the problem is ill-posed, since $\max z(x) = +\infty$;

3) the usual case: there exists $j \in \{1, \ldots, n\}$ such that $z_j - c_j < 0$, and there exists $i \in J$ such that $a_{ij} > 0$. The change in basis described earlier is now possible and should be performed, possibly several times, until case (1) is reached.

Could the simplex algorithm ever fail to terminate if case (3) leads to a loop? The answer is yes, and examples have been successfully constructed. However, they are very rare in practice.

Let us therefore state two important theorems about the simplex method.

THEOREM 1.3.– Let $x \in \mathbb{R}^n_+$ be a basic realizable solution of (P) with respect to a basis J ($|J| = m = rank(A)$). Let $z_j = \sum_{i \in J} c_i a_{ij}$. If $z_j - c_j \geq 0$ for every $j = 1, \ldots, n$, then x is an optimal basic realizable solution.

THEOREM 1.4.– Let $x \in \mathbb{R}^n_+$ be a basic realizable solution of (P) with respect to a basis J ($|J| = m = rank(A)$). Let $z_j = \sum_{i \in J} c_i a_{ij}$. Suppose that $a_{ij} \leq 0$ for every $i \in J$ and for every $j \in \{1, \ldots, n\}$ such that $z_j - c_j < 0$. Then the set $\{z(x),\ x \text{ is a realizable solution}\}$ is unbounded.

REMARK.– For a comprehensive discussion of the efficiency of the simplex method, refer to [CHV 83].

1.6. Initialization of the simplex algorithm

This section presents two techniques for finding a basic realizable solution to initialize the simplex algorithm: the first is the Big M method, and the second is the Phase I method.

1.6.1. *Big M method*

Suppose that we wish to solve the LP

$$(P) \quad \begin{array}{ll} \max & z(x) = c^T x \\ \text{s.t.} & Ax = b, \\ & x \geq 0 \end{array}$$

where A is an $m \times n$ matrix, rank $A = m < n$.

By adding slack variables (with positive or negative signs), we can always reduce the LP to the form (P) described above, with $b \in \mathbb{R}^m_+$. If there is no obvious basic realizable solution to initialize the simplex algorithm, we proceed by adding artificial variables $y_i \geq 0$ to the constraints:

$$Ax = b \text{ is replaced by } Ax + y = b,\ y = (y_1, y_2, \ldots, y_m)^T \in \mathbb{R}^m_+.$$

The new constraints are not equivalent to the initial constraints. The $y_i > 0$ are penalized by replacing the objective function $z(x)$ with

$$z' = c^T x - M \sum_{i=1}^{m} y_i,$$

where M is some very large positive value. We will choose $y_i = b_i$, $i = 1, \ldots, m$, as a basic feasible solution to serve as the initial solution.

Since the y_i significantly reduce the value of z', they will disappear from the basis over the course of the simplex algorithm. As they are non-basic variables, their value will be equal to zero, and the LP that is ultimately solved is the same as the program (P).

EXAMPLE 1.7.– Consider the LP

$$\begin{aligned} \max\ & z(x) = -x_1 + 2x_2 - 2x_3 \\ \text{s.t.}\ & x_1 + x_2 - x_3 = 3 \\ & -x_1 + 3x_2 = -4 \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

The new program is given as

$$\begin{aligned} \max\ & z'(x, y) = (-1 + 2M)x_1 + (2 - 2M)x_2 + (-2 - M)x_3 \\ \text{s.t.}\ & x_1 + x_2 - x_3 + y_1 = 3 \\ & -x_1 + 3x_2 + y_2 = -4 \\ & x_1, x_2, x_3, y_1, y_2 \geq 0. \end{aligned}$$

The first tableau is as follows:

c	-1+2M	2-2M	-2-M	0	0
Basic variables	x_1	x_2	x_3	y_1	y_2
$y_1 = 3$	1	1	-1	1	0
$y_2 = 4$	1	-3	0	0	1
$z'(x) = -7M$	1-2M	-2+2M	2+M	0	0

The second tableau is as follows:

c	-1+2M	2-2M	-2-M	0	0
Basic variables	x_1	x_2	x_3	y_1	y_2
$x_1 = 3$	1	1	-1	1	0
$y_2 = 1$	0	-4	1	-1	1
$z'(x) = -3 - M$	0	-3+4M	3-M	-1+2M	0

The third tableau is as follows:

c	-1+2M	2-2M	-2-M	0	0
Basic variables	x_1	x_2	x_3	y_1	y_2
$x_1 = 4$	1	-3	0	0	1
$x_3 = 1$	0	-4	1	-1	1
$z'(x) = -6$	0	9	0	2+M	-3+M

The unique optimal solution is given by $x_1 = 4$, $x_2 = 0$, $x_3 = 1$, and $z(x_1, x_2, x_3) = -6$.

1.6.2. *Auxiliary program or Phase I*

Suppose that the LP that we wish to solve is in the standard form:

$$\begin{cases} \max\ z(x) = c^T x \\ \text{s.t.}\quad Ax = b,\ x \geq 0. \end{cases}$$

The auxiliary program method proceeds by letting $M \to +\infty$ in (P_M) and solving the program obtained in the limit. Let

$$z''(x, y) = \frac{1}{M} z'(x, y) = \frac{c}{M} x - \sum_{i=1}^{m} y_i.$$

Maximizing $z'(x, y)$ is equivalent to maximizing $z''(x, y)$, or maximizing

$$\lim_{M \to +\infty} z''(x, y) = -\sum_{i=1}^{m} y_i.$$

We therefore solve the auxiliary program (P_A) given by

$$(P_A) \begin{cases} \max\ z''(x, y) = -\sum_{i=1}^{m} y_i \\ \text{s.t.}\quad Ax + y = b, \\ \qquad x \in \mathbb{R}^n_+,\ y \in \mathbb{R}^m_+,\ b \in \mathbb{R}^m_+. \end{cases}$$

The simplex algorithm can now be applied to (P_A) starting from the basic realizable solution $y = b$.

– *First case*: $\max z''(y) < 0$. There exists $y_i > 0$, $i \in \{1, 2, \ldots, m\}$. In this case, (P) does not have any realizable solutions.

– *Second case*: $\max z''(y) = 0$. The optimal solution of (P_A) can be used as an initial basic realizable solution for applying the simplex algorithm to (P). The objective function of (P) is expressed in terms of non-basic variables, then the simplex algorithm is applied to (P).

REMARK.– For a minimization problem, we add $\sum y_i$ instead.

Let us go into more detail for a minimization problem. The initial tableau of the auxiliary problem is as follows:

A	b
$-c_B^T A \mid O$	$-c_B^T b$

where $B = I$, $c_B = (1, 1, 1, \ldots)^T$, and the matrices A and b relate to the auxiliary program. For the original variables i, we have $c_i = 0$.

If (x^*, y^*) is a zero-cost optimal solution of the auxiliary problem, then the artificial variable in the basis is necessarily zero, so the solution is degenerate.

If we assume that the kth basic variable is artificial, consider the kth row of the tableau and select the element of this row in column j, where j is the index of a variable in the original problem. If the element is non-zero: k leaves the basis and j enters the basis, and we pivot the tableau. But what if no such element exists? This

would mean that the matrix A does not have full rank, and the row in question corresponds to a redundant constraint.

Once the optimal tableau of the auxiliary problem has been obtained and all artificial variables have left the basis, we obtain the initial tableau of the original problem P1 by deleting any columns that relate to the artificial variables and calculating the reduced initial costs: $c^T - c_B^T B^{-1} A$ and $-c_B^T B^{-1} b$ in the box reserved for the cost function. The following example shows how this method works.

EXAMPLE 1.8.– Consider the following linear programming problem:

$$\begin{array}{lrcrcrcrcl} \min & z = & x_1 & + & x_2 & + & x_3 & - & 2x_4 & \\ \text{s.c.} & & x_1 & + & x_2 & + & x_3 & & & = 1 \\ & & & & x_2 & & & + & x_4 & = 1 \\ & & x_1, & & x_2, & & x_3, & & x_4 & \geq 0 \end{array} \qquad [1.10]$$

We will first perform Phase I of the simplex algorithm. After adding artificial variables, we obtain the tableau as follows:

$$T_0 = \begin{array}{c|cccccc|c|c} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & & \theta \\ \hline & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \leftarrow \\ & 0 & 1 & 0 & 1 & 0 & 1 & 1 & \\ \hline & -1 & -2 & -1 & -1 & 0 & 0 & -2 & \\ \hline & \uparrow & & & & & & & \end{array}$$

$$T_1 = \begin{array}{c|cccccc|c|c} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & & \theta \\ \hline & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \leftarrow \\ & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ \hline & 0 & -1 & 0 & -1 & 1 & 0 & -1 & \\ \hline & & \uparrow & & & & & & \end{array}$$

$$T_2 = \begin{array}{c|cccccc|c|c} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & & \theta \\ \hline & 1 & 1 & 1 & 0 & 1 & 0 & 1 & \\ & -1 & 0 & -1 & 1 & -1 & 1 & 0 & 0 \leftarrow \\ \hline & 1 & 0 & 1 & -1 & 2 & 0 & 0 & \\ \hline & & & \uparrow & & & & & \end{array}$$

$$T_3 = \begin{array}{c|cccccc|c} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & \\ \hline & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ & -1 & 0 & -1 & 1 & -1 & 1 & 0 \\ \hline & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ \hline \end{array}$$

There are no longer any artificial variables in the basis. We have found the optimal tableau of the auxiliary problem, and therefore an admissible basic solution for the initial problem. Since Phase I is now complete, delete the columns corresponding to the artificial variables and compute the reduced costs to obtain the following tableau:

$$T_0 = \begin{array}{c|cccc|c|c} & x_1 & x_2 & x_3 & x_4 & & \theta \\ \hline & 1 & 1 & 1 & 0 & 1 & 1 \leftarrow \\ & -1 & 0 & -1 & 1 & 0 & \\ \hline & -2 & 0 & -2 & 0 & -1 & \\ & \uparrow & & & & & \end{array}$$

$$T_1 = \begin{array}{c|cccc|c} & x_1 & x_2 & x_3 & x_4 & \\ \hline & 1 & 1 & 1 & 0 & 1 \\ & 0 & 1 & 0 & 1 & 1 \\ \hline & 0 & 2 & 0 & 0 & 1 \end{array}$$

This tableau is optimal. The optimal solution is $x_1 = 1$, $x_2 = 0$, $x_3 = 0$ and $x_4 = 1$, giving an optimal value of $z = -1$.

1.6.3. *Degeneracy and cycling*

Suppose that the current basic admissible solution is degenerate (i.e. at least one basic variable is zero), and let us recall a possible unfavorable scenario that can occur when the simplex algorithm is iterated. Performing a change of basis yields a value for the cost function that is greater than or equal to the previous value. The method is only guaranteed to converge if $z^* < z_0$; in fact, if the previous basic solution is degenerate, it might be the case that $z^* = z_0$.

If this phenomenon occurs multiple times over consecutive iterations, the algorithm can return to a basis that was already considered earlier. In other words, it returns to the same vertex twice. This is called cycling.

Although degeneracy is a frequently encountered phenomenon in linear programming, cycling is not. Some examples have been given in the literature [GAS 75]. Despite the rarity of cycling, it would be desirable to specify a method that avoids it and therefore guarantees that the simplex method will converge. There are several possible choices for a rule that avoids the inconvenience of cycling.

In the past, the most common technique was the so-called method of small perturbations. This method applies an infinitesimal geometric modification to the vector d in order to force it to be expressible as a linear combination of m vectors, with strictly positive coefficients in some basis. Each vertex is then defined as the intersection of n hyperplanes [HIL 90].

More recently, Bland [BLA 77] suggested modifying the rules for choosing the change of basis. Bland's rule proceeds as follows:

– for the variable entering the basis, choose the smallest index j such that the reduced cost is negative;

– for the variable leaving the basis, if there is equality in θ^*, choose the variable with the lowest index.

1.6.4. *Geometric structure of realizable solutions*

Earlier, when we solved linear programs graphically, the optimal solutions were on the boundary of the convex set of realizable solutions. If there was a unique optimal solution, it was an extreme point.

Recall the canonical form of a linear program:

$$(P)\begin{cases}\max\ z(x) = c^T x\\ \text{s.t.}\quad Ax \le b,\\ \qquad x \ge 0,\end{cases}$$

where the matrix A is assumed to be an $m \times n$ matrix of rank m $(< n)$. The standard form is then given by

$$(P_s)\begin{cases}\max\qquad z(x) = c^T x\\ s.t.\ \ (A, I_m)\begin{pmatrix}x\\ e\end{pmatrix} = b,\\ \qquad x \in \mathbb{R}^n_+,\ e \in \mathbb{R}^m_+\end{cases}$$

It is possible to show the following result.

Let Γ (respectively, $\sum$) be the set of feasible solutions of (P) (respectively of (P_s)):

$$\Gamma = \{x \in \mathbb{R}^n\ ;\ Ax \le b,\ x \ge 0\},$$
$$\sum = \left\{\begin{pmatrix}x\\ e\end{pmatrix},\ (A, I_m)\begin{pmatrix}x\\ e\end{pmatrix} = 0,\ x \in \mathbb{R}^n_+,\ e \in \mathbb{R}^m_+\right\}.$$

Then Γ and $\sum$ are closed and convex sets.

Recall that the simplex method only uses the basic solutions of (P_s). The next theorem tells us that these are in fact the extreme points of $\sum$.

THEOREM 1.5.– $\begin{pmatrix} x^0 \\ e^0 \end{pmatrix}$ is a basic feasible solution of (P_s) if and only if $\begin{pmatrix} x^0 \\ e^0 \end{pmatrix}$ is an extreme point of $\sum$.

Examples where the solution is not unique show that the optimal solutions lie on the boundary:

THEOREM 1.6.– Every optimal solution of (P) (respectively of (P_s)) belongs to the boundary of Γ (respectively of $\sum$).

A question presents itself: can there be optimal solutions but no optimal basic feasible solutions? If so, the simplex method, which only considers basic solutions, would not be able to find an optimal solution. The following theorem tells us that this does not happen.

THEOREM 1.7.– If (P_s) has an optimal solution, then (P_s) has an optimal basic feasible solution.

1.7. Interior-point algorithm

The simplex algorithm moves along a sequence of vertices of the polyhedron and converges rapidly on average. However, Klee and Minty [KLE 72] found pathological examples where the simplex algorithm visits almost every vertex, which can potentially be an enormous number. The existence of theoretical linear programs that derail the simplex algorithm motivated research to find an algorithm that has polynomial complexity in m and n, even in the worst-case scenarios. Khachian [KHA 79] found the first such algorithm, based on the ellipsoid method from nonlinear optimization. Although it is faster than the simplex method in Minty's problems, it never succeeded in establishing itself because it tended to be much slower than the simplex algorithm on real problems in practice. Nevertheless, as a mathematical result, it inspired research into interior-point algorithms.

In 1984, Karmarkar [KAR 84] found a new algorithm. The computation time in the worst-case scenario is proportional to $n^{3.4}L$, where L denotes the number of bits required to encode the tableaus A, b and c that define the linear program. The algorithm is too complicated for computations by hand; known as the interior-point algorithm, it enters into the polyhedron itself to converge on the optimal vertex more quickly.

Today, some variants of this algorithm are able to compete with the simplex algorithm. Some software programs already include an interior-point algorithm in addition to the simplex algorithm. On some types of linear programs, interior-point algorithms can be 10 times faster than the simplex algorithm, but nevertheless, there are still many cases where the simplex algorithm is quicker [PRI 11].

EXAMPLE 1.9.– Consider the LP

$$\begin{aligned} \max\ & z = 2x_1 + x_2 \\ \text{s.t.}\ & x_1 + x_2 \leq 4 \\ & -x_1 + x_2 \leq 2 \\ & x_1 \leq 2 \\ & 0.5x_1 - x_2 \leq 0 \\ & x_1, x_2 \geq 0 \end{aligned}$$

The optimum is $x^* = (2, 2)^T$ with cost $z^* = 6$. Starting from the interior point $x_0 = (1, 1)^T$ and without a safety factor, the algorithm performs two iterations: it considers the point $x_1 = (2, 1.591)^T$, followed by the point x^*.

1.8. Duality

Duality is an essential notion in linear programming. The duality operation associates any so-called primal linear programming problem with another such problem, known as the dual, that is defined solely in terms of the data of the primal problem.

Duality is interesting for two reasons:

– the dual problem has an important economic interpretation that offers another perspective of the primal problem;

– there are straightforward and powerful mathematical relations between the primal and dual problems; this will allow us to develop new algorithms that will be more efficient in many situations.

Suppose that we have the following primal program:

$$(P) \begin{cases} \max\ z(x) = cx \\ \text{s.t.}\quad Ax \leq b, \\ \qquad x \in \mathbb{R}^n_+ \end{cases}$$

Its dual program is as follows:

$$(D)\begin{cases}\min\ w(y) = yb \\ \text{s.t.}\quad yA \geq c, \\ \qquad y \in \mathbb{R}_+^m\end{cases}$$

REMARK.– The primal and dual programs satisfy the following relations:

– $m =$ number of constraints of (P) = number of variables of (D),

– $n =$ number of variables of (P) = number of constraints of (D).

If (P) has two constraints, (D) has two variables and can be solved graphically, regardless of the number of variables of (P).

So, what is the form of (D) when (P) is not in canonical form?

In such a case, we can determine (D) by reducing (P) to canonical form, as shown in the following example.

EXAMPLE 1.10.– Consider the LP:

$$(P)\begin{cases}\max\ z(x) = c^T x \\ \text{s.t.}\quad Ax = b,\ x \geq 0 \Leftrightarrow \begin{pmatrix} A \\ -A \end{pmatrix} x \leq \begin{pmatrix} b \\ -b \end{pmatrix},\ x \geq 0\end{cases}$$

By the above definition, the dual of (P) is given by

$$(D)\begin{cases}\min\ w(u,v) = (u,v)\begin{pmatrix} b \\ -b \end{pmatrix} = (u-v)b \\ \text{s.t.}\quad (u,v)\begin{pmatrix} A \\ -A \end{pmatrix} = (u-v)A \geq c,\ u,v \in \mathbb{R}_+^m.\end{cases}$$

Setting $y = u - v$ gives

$(D)\ \min\ w(y) = yb$ s.t. $yA \geq c,\ y$ with no sign restriction (denoted (y)).

In general, we have:

Primal	Dual
Maximization	Minimization
Coefficient of z	Right-hand side of the constraints
Right-hand side of the constraints	Coefficient of w

$$\text{Constraint} \begin{cases} = \\ \leq \\ \geq \end{cases} \qquad \text{Variable} \begin{cases} \text{no sign constraint} \\ \leq \\ \geq \end{cases}$$

$$\text{Variable} \begin{cases} \text{no sign constraint} \\ \geq \\ \leq \end{cases} \qquad \text{Constraint} \begin{cases} = \\ \leq \\ \geq \end{cases}$$

THEOREM 1.8.– The dual of the dual is the primal.

1.8.1. *Duality theorem*

Now that we have defined the dual of a linear program, let us study the links between the solutions of programs that are dual to one another.

THEOREM 1.9.– Let (x, y) be a feasible solution of $(\bar{P})$ and (u, v) a feasible solution of $(\bar{D})$. Then:

1) $z(x, y) \leq w(u, v)$;

2) $z(x, y) = w(u, v) \Rightarrow (x, y)$ and (u, v) are optimal solutions of $(\bar{P})$ and $(\bar{D})$.

THEOREM 1.10.– One of the following three cases is satisfied:

1) $(\bar{P})$ and $(\bar{D})$ have optimal solutions and $\max z(x, y) = \min w(u, v)$;

2) one of $(\bar{P})$ and $(\bar{D})$ has a feasible solution, but not the other;

3) neither $(\bar{P})$ nor $(\bar{D})$ have feasible solutions.

THEOREM 1.11.– Let (x, y) (respectively, (u, v)) be feasible solutions of $(\bar{P})$ (respectively, $(\bar{D})$). Then (x, y) and (u, v) are optimal solutions of $(\bar{P})$ and $(\bar{D})$ if and only if the following statements hold:

– if one constraint is strictly an inequality in $(\bar{P})$ (respectively, $(\bar{D})$), then the corresponding variable of $(\bar{D})$ (respectively, of $(\bar{P})$) is zero;

– if the value of a restricted variable (i.e. a positive variable or a negative variable) in one of the programs $(\bar{P})$ or $(\bar{D})$ is not zero, then the corresponding constraint in the other program is an equality.

This theorem can alternatively be stated as follows:

THEOREM 1.12.– Let x and p be admissible solutions of the primal and dual programs, respectively. Then the vectors x and p are, respectively, optimal solutions of the two problems if and only if:

$$p_i(a_i^T x - b_i) = 0 \ \forall i$$
$$(c_j - p^T A_j)x_j = 0 \ \forall j$$

Application

Consider the LP

$$\begin{aligned} \min \ & 13x_1 + 10x_2 + 6x_3 \\ \text{s.t.} \ & 5x_1 + x_2 + 3x_3 = 8 \\ & 3x_1 + x_2 = 3 \\ & x_i \geq 0 \end{aligned}$$

The dual of this program is:

$$\begin{aligned} \max \ & 8p_1 + 3p_2 \\ \text{s.t.} \ & 5p_1 + 3p_2 \leq 13 \\ & p_1 + p_2 \leq 10 \\ & 3p_1 \leq 6 \end{aligned}$$

The solution of the primal is $x^* = (1, 0, 1)^T$.

The optimal dual solution can be constructed from additional slack conditions:

– the condition $p_i(a_i^T x^* - b_i) = 0$ is satisfied because x^* is admissible;

– the condition $(c_j - p^T A_j)x_j = 0$ is satisfied for $j = 2$:

- for $j = 1$, this condition becomes:

$$5p_1 + 3p_2 = 13.$$

- for $j = 3$, it becomes:

$3p_1 = 6$.

These two conditions give $p_1 = 2$ and $p_2 = 1$. This solution is dual admissible. The total dual cost is 19, and so is the primal cost.

1.9. Relaxation

In mathematical programming, good performance essentially depends on the program's ability to construct useful bounds (lower bounds for maximization and upper bounds for minimization). These bounds can be obtained by relaxing the linear program, i.e.

– either increasing the set of solutions to optimize over a larger space (that is easier to optimize); this includes constraint relaxation (and in particular continuous relaxation);

– or replacing the objective function with an upper bound (in the case of maximization), for example, Lagrangian relaxation.

For a mathematical program, the following continuous relaxations are possible:

– (simple) continuous maximization, which is very effective for LPs (with the simplex algorithm, the interior-point method, or the volume algorithm); however, we will see that continuous relaxation is rarely the best option;

– in the specific context of LPs, we can improve the value of linear relaxation by adding constraints – reinforcement;

– we can also add variables – reformulation;

– Lagrangian relaxation is another approach.

1.9.1. *Lagrangian relaxation*

Lagrangian duality provides a particularly fruitful framework for relaxation. It works as follows:

– choose an inequality in the LP and remove it from the LP;

– add this inequality to the objective function with a multiplier that penalizes the objective function if the solution found does not satisfy the inequality.

This relaxation often produces a very good relaxation value, far better than continuous relaxation.

Consider the following LP:

$$\begin{aligned} \min\ & f(x) \\ \text{s.t.}\ & g_i(x) \leq 0 \quad i \in I \\ & x \in K \subset \mathbb{R}^n \end{aligned} \tag{1.11}$$

To each constraint $i \in I$, we assign a real number $\lambda_i \geq 0$ called the Lagrange multiplier. We also define the vector $\lambda = (\lambda_i, 0\ i \in I)$. The Lagrangian function is then defined as

$$\mathcal{L}(x, \lambda) = f(x) + \sum_{i \in I} \lambda_i gi(x).$$

This function has the two vector arguments x and λ and takes values in $\mathbb{R}$. Suppose that the PL is such that the problem $\min_{x \in S} \mathcal{L}(x, \lambda)$ has an optimal solution for every $\lambda \geq 0$.

More details are given in Chapter 7. Now consider the following LP [FLE 10]:

$$\begin{aligned} \min\ & z = 4x_1 + 2x_2 + x_3 \\ \text{s.t.}\ & x_1 + x_2 \geq 1.1 \\ & x_2 + x_3 \geq 3.2 \\ & x_1 + x_3 \geq 4.4 \\ & x_i \geq 0 \end{aligned} \tag{1.12}$$

We obtain the optimal solution $(0, 1.1, 4.4)^T$, which gives the optimal cost as $z^* = 6.6$.

By relaxing the third constraint ($x_1 + x_2 \geq 4.4$) and keeping the integrality constraints on the variables x_i, we obtain the following LP by adding the expression $\lambda(4.4 - x_1 - x_3)$ to the objective function, where λ is the Lagrangian relaxation coefficient, which is updated at each iteration of the method:

$$\begin{aligned} \min\ & f_\lambda(x_i) = 4x_1 + 2x_2 + x_3 + \lambda(4.4 - x_1 - x_3) \\ \text{s.t.}\ & x_1 + x_2 \geq 1.1 \\ & x_2 + x_3 \geq 3.2 \\ & x_i \geq 0 \\ & x_i \leq 10 \end{aligned} \tag{1.13}$$

The last constraint is added to bound the variables x_i. This guarantees that the problem is bounded for every $\lambda > 0$. The coefficient λ is updated as follows: $\lambda^{k+1} = \max\left(0, \lambda^k + \frac{1}{k} \times (4.4 - x_1 - x_3)\right)$. It can be shown that λ tends to 1 and z tends to 8.4.

1.10. Postoptimal analysis

Consider the LP:

$$\begin{aligned} \min\ & z = c^T x \\ \text{s.t.}\ & Ax \leq b \\ & x \geq 0 \end{aligned} \qquad [1.14]$$

We will now perform a postoptimal analysis of the optimal solution in terms of the problem data, also known as a sensitivity analysis. In other words, how does the solution or the objective function vary if we modify one of the entries of the vectors c or b, or the matrix A?

To conduct a postoptimal analysis, we need to be able to express the solution of the simplex algorithm in terms of the initial problem data instead of simplex tableaus, which change at each iteration.

As always, we introduce slack variables to the problem [1.14], writing A for the resulting matrix of size $m \times (m+n)$. We also assume that the basic variables B of the optimal solution x_B are known. This information can be read off from the final tableau of the simplex algorithm, which is why this procedure is called postoptimal analysis.

Let $B = \{x_{j_1}, x_{j_2}, \ldots, x_{j_m}\}$ be the list of basic variables, and let N be its complement with respect to the index set $\{1, 2, \ldots, m+n\}$. For example, if $m = n = 3$ and $B = \{x_1, x_5, x_2\}$, then $N = \{x_3, x_4, x_6\}$. Based on the sets B and N, we can extract the following submatrices of A:

– A_B is the submatrix formed by the columns corresponding to the variables of B. We take them in the order specified by B.

– A_N is the submatrix formed by the columns corresponding to the variables of N in the order specified by N.

EXAMPLE 1.11.– Consider the problem

$$\begin{aligned} \min\ & z = -1,100x_1 - 1,400x_2 - 1,500x_3 \\ \text{s.t.}\ & x_1 + x_2 + x_3 \leq 340 \\ & 2x_1 + 3x_2 + x_3 \leq 2,400 \\ & x_1 + 2x_2 + 3x_3 \leq 560 \\ & x_1, x_2, x_3 \geq 0 \end{aligned} \qquad [1.15]$$

The matrix system $Ax = b$ with slack variables is given as

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 2 & 3 & 1 & 0 & 1 & 0 \\ 1 & 2 & 3 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_6 \end{Bmatrix} = \begin{Bmatrix} 340 \\ 2,400 \\ 560 \end{Bmatrix}.$$

The final tableau of the simplex algorithm is written as

x_1	x_2	x_3	x_4	x_5	x_6	
1	0	-1	2	0	-1	120
0	0	-3	-1	1	-1	1,500
0	1	2	-1	0	1	220
0	0	200	800	0	300	440

We therefore indeed have $B = \{x_1, x_5, x_2\}$ and $N = \{x_3, x_4, x_6\}$. The submatrices A_B and A_N can be written as:

$$A_B = \begin{bmatrix} 1 & 0 & 1 \\ 2 & 1 & 3 \\ 1 & 0 & 2 \end{bmatrix} \quad A_N = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 3 & 0 & 1 \end{bmatrix}.$$

Using a block partition based on the index sets B and N, we can write

$$Ax = b \leftrightarrow [A_B \; A_N] \begin{bmatrix} x_B \\ x_N \end{bmatrix} = b \leftrightarrow A_B x_B + A_N x_N = b.$$

The solution $x_B : x_B = A_B^{-1}(b - A_N x_N)$ can be computed algebraically. The matrix A_B is invertible because B is a basis. The optimal solution is given by the formula $x_B = A_B^{-1} b$, since the non-basic variables must be zero, i.e. $x_N = 0$. In our example, we have

$$x_B = \begin{Bmatrix} x_1 \\ x_5 \\ x_2 \end{Bmatrix} = \begin{bmatrix} 2 & 0 & -1 \\ -1 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix} \begin{Bmatrix} 340 \\ 2,400 \\ 560 \end{Bmatrix} = \begin{Bmatrix} 120 \\ 1,500 \\ 220 \end{Bmatrix},$$

which is the correct optimal solution according to the final simplex tableau. The other columns of the final tableau correspond to

$$A_B^{-1} A_N = \begin{bmatrix} 2 & 0 & -1 \\ -1 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 3 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 2 & -1 \\ -3 & -1 & -1 \\ 2 & -1 & 1 \end{bmatrix},$$

which are indeed the columns that correspond to $N = \{x_3, x_4, x_6\}$.

1.10.1. *Effect of modifying* b

Let us analyze the effect of modifying the vector b. In other words, we shall study the behavior of the solution of the modified problem when b is replaced by $\tilde{b} = b + \Delta b$.

$$\begin{aligned} \min\ & z = c^T x \\ \text{s.t.}\ & Ax \leq \tilde{b} \\ & x \geq 0 \end{aligned} \qquad [1.16]$$

Let x_B be the basic variables of the solution. Our goal is to determine a condition that guarantees that the basis B will remain optimal. In fact, this is easy. The vector b only appears in the optimality condition [1.16]. Therefore, the basic variables x_B remain optimal for the modified problem if

$$A_B^{-1}\tilde{b} \geq 0 \leftrightarrow A_B^{-1}\Delta b \geq -A_B^{-1}b. \qquad [1.17]$$

EXAMPLE 1.12.– Consider the LP:

$$\min \quad z = x_1 + \tfrac{3}{2}x_2 + 3x_3 \qquad [1.18]$$

$$\text{s.t.} \begin{cases} x_1 + x_2 + 2x_3 \geq 6 \\ x_1 + 2x_2 + x_3 \geq 10 \\ x_1, x_2, x_3 \geq 0 \end{cases} \qquad [1.19]$$

In matrix form with slack variables, this can be written as:

$$A = \begin{bmatrix} -1 & -1 & -2 & 1 & 0 \\ -1 & -2 & -1 & 0 & 1 \end{bmatrix}, \quad b = \begin{Bmatrix} -6 \\ -10 \end{Bmatrix}, \quad c = \begin{Bmatrix} 1 \\ 3/2 \\ 3 \\ 0 \\ 0 \end{Bmatrix}.$$

Suppose that the optimal basis is $B = \{x_1, x_2\}$. Then

$$A_B = \begin{bmatrix} -1 & -1 \\ -1 & -2 \end{bmatrix} \Rightarrow A_B^{-1} = \begin{bmatrix} -2 & 1 \\ 1 & -1 \end{bmatrix}.$$

Compute x_B:

$$x_B = A_B^{-1}b = \begin{bmatrix} -2 & 1 \\ 1 & -1 \end{bmatrix} \begin{Bmatrix} -6 \\ -10 \end{Bmatrix} = \begin{Bmatrix} 2 \\ 4 \end{Bmatrix} \geq 0.$$

Compute the reduced costs $c_N^T - c_B^T A_B^{-1} A_N$:

$$c_N^T - c_B^T A_B^{-1} A_N = [\frac{3}{2}, \frac{1}{2}, \frac{1}{2}] \geq 0.$$

The solution $x_B = (2, 4)^T$ is therefore optimal, i.e. $x = (2, 4, 0, 0, 0)^T$.

If $-b \mapsto -\tilde{b} = \begin{Bmatrix} -6 - a \\ -10 \end{Bmatrix}$, i.e. if we consider a change in b_1 only, then:

$$\tilde{x}_B = A_B^{-1}(-\tilde{b}) = \begin{bmatrix} -2 & 1 \\ 1 & -1 \end{bmatrix} \begin{Bmatrix} -6 - a \\ -10 \end{Bmatrix} = \begin{Bmatrix} 2 + 2a \\ 4 - a \end{Bmatrix}.$$

We will have $\tilde{x}_B \geq 0$ if and only if

$$2 + 2a \geq 0 \quad \text{and} \quad 4 - a \geq 0.$$

This gives an interval for the parameter b_1: $-1 \leq a \leq 4$.

What is the interval for b_2?

If $-b \mapsto -\tilde{b} = \begin{Bmatrix} -6 \\ -10 - a \end{Bmatrix}$, then:

$$\tilde{x}_B = A_B^{-1}(-\tilde{b}) = \begin{bmatrix} -2 & 1 \\ 1 & -1 \end{bmatrix} \begin{Bmatrix} -6 \\ -10 - a \end{Bmatrix} = \begin{Bmatrix} 2 - a \\ 4 + a \end{Bmatrix}.$$

We will have $\tilde{x}_B \geq 0$ if and only if

$$2 - a \geq 0 \quad \text{and} \quad 4 + a \geq 0.$$

This gives the interval for the parameter b_2: $-4 \leq a \leq 2$.

What is the effect on the minimum value of the objective function?

The effect is:

$$z(\tilde{b}) = c_B^T \tilde{x}_B + c_N^T \tilde{x}_N = c_B^T \tilde{x}_B = [1, \frac{3}{2}] \begin{Bmatrix} 2 - a \\ 4 + a \end{Bmatrix} = 8 + \frac{a}{2}.$$

1.10.2. *Effect of modifying* c

Next, let us find a condition that guarantees that the basis B will remain optimal under $c \mapsto \tilde{c} = c + \Delta c$. First, observe that the solution $x_B = A_B^{-1} b$ remains

unchanged. However, the simplex tableau might no longer be optimal. To keep it optimal, we must impose the stability condition

$$\tilde{c}_N^T - \tilde{c}_B^T A_B^{-1} A_N \geq 0 \leftrightarrow (c + \Delta x)_N^T - (c - \Delta c)_B^T A_B^{-1} A_N \geq 0. \quad [1.20]$$

In general, let us determine the effect of modifying a single variable x_i:

$$c_i \mapsto \tilde{c}_i = c_i + \Delta c_i$$

There are two cases to analyze depending on whether x_i is a basic or non-basic variable.

Case of a non-basic variable

Write e_i for the canonical basis vector of $\mathbb{R}^n$, i.e. $e_i = (0, 0, \ldots, 1, 0, \ldots, 0)^T$ with 1 at position i. Then:

$$\begin{aligned} \Delta c &= \Delta c_i e_i \\ (\Delta c)_B &= 0 \\ (\Delta c)_N &= \Delta c_i e_i \end{aligned}$$

The ith component of the vector $\tilde{c}_N^T - \tilde{c}_B^T A_B^{-1} A_N$ is

$$\begin{aligned} &c_i + \Delta c_i - (c_B^T A_B^{-1} A_N)_i \geq 0 \\ &\Delta c_i \geq -(c_i - (c_B^T A_B^{-1} A_N)_i) = -r_i \ , \\ &\tilde{c}_i \geq c_i - r_i \end{aligned}$$

where the r_i are the components of the final row of the simplex tableau. In other words, we have all the information that we need to compute the stability intervals of the coefficients c_i.

EXAMPLE 1.13.– Again, consider the previous example. The final simplex tableau is:

x_1	x_2	x_3	x_4	x_5	
1	0	3	-2	1	2
0	1	-1	1	-1	4
0	0	3/2	1/2	1/2	-8

The last row gives us the vector of reduced costs $r = (0, 0, \frac{3}{2}, \frac{1}{2}, \frac{1}{2})^T$. The basic variables are $B = \{x_1, x_2\}$, and the non-basic variables are $N = \{x_3, x_4, x_5\}$. The

objective function is of the form $z = c_1x_1 + c_2x_2 + c_3x_3$ with $c_1 = 1$, $c_2 = \frac{3}{2}$, and $c_3 = 3$. It does not make sense to perturb the coefficients c_4 and c_5. Accordingly, we can only perturb x_3. Let us compute the stability interval around c_3 associated with the non-basic variable x_3. The stability condition is

$$\tilde{c}_3 \geq c_3 - r_3 = \frac{3}{2} \leftrightarrow \tilde{c}_3 \geq \frac{3}{2}.$$

The optimal solution $x = (2, 4, 0, 0, 0)^T$ is the same for every value $\tilde{c}_3 \geq \frac{3}{2}$. Furthermore, the value of the objective function ($z = 8$) remains unchanged because the variable is non-basic.

1.11. Application to an inventory problem

Consider the problem of an automobile manufacturer that offers two models for sale, a small automobile and a large automobile. Suppose that there is sufficient demand that the manufacturer is certain to sell whatever is produced at the current sales price of 16,000 euros for large automobiles, and 10,000 euros for small ones. The manufacturer's only problem is the limited supply of two raw materials, rubber and steel. Manufacturing a small automobile requires one unit of rubber and one unit of steel, whereas manufacturing a large automobile requires one unit of rubber and two units of steel. If the manufacturer has 400 units of rubber and 600 units of steel in stock, how many small and large vehicles should be produced from this inventory to maximize the turnover?

Let x be the number of large automobiles manufactured, y the number of small automobiles manufactured and z the resulting turnover. The problem can therefore be expressed in the form:

$$\begin{aligned} \max\ & z = 16,000x + 10,000y \\ \text{s.t.}\quad & x + y \leq 400 \\ & 2x + y \leq 600 \\ & x \geq 0,\ y \geq 0 \end{aligned} \qquad [1.21]$$

1.11.1. *Optimal solution*

It is relatively easy to find a graphical solution for systems like this with only two variables. This allows us to find the optimal solution $x = 200$ and $y = 200$, which corresponds to $z = 5,200,000$. In this particular example, the optimal solution is unique, corresponding to a vertex of the region bounded by the constraints.

1.11.2. *Sensitivity to variation in stock*

Let us observe how the solution of the problem evolves when some of the starting data is modified, for example by increasing the stock of rubber or steel. Suppose that 700 units of steel are in stock instead of 600. The problem now becomes:

$$\begin{aligned} \max\ & z = 16,000x + 10,000y \\ \text{s.t.}\ & x + y \leq 400 \\ & 2x + y \leq 700 \\ & x \geq 0,\ y \geq 0 \end{aligned} \qquad [1.22]$$

By solving graphically again, we find that the optimal solution is now $x = 300$ and $y = 100$, giving $z = 5,800,000$. In other words, increasing the available steel by 100 units increases the turnover by 600,000 euros. We say that the marginal price of a unit of steel is 6,000 euros.

If the amount of steel in stock is increased to 800, the optimal solution becomes $x = 400$ and $y = 0$, and the turnover becomes $z = 6,400,000$. Increasing the steel in stock beyond 800 without also changing the rubber in stock no longer affects the optimal solution, because y is constrained to remain positive.

Suppose now that the steel in stock remains fixed at 600, but the rubber in stock is increased by 400 to 500. The new problem is

$$\begin{aligned} \max\ & z = 16,000x + 10,000y \\ \text{s.t.}\ & x + y \leq 500 \\ & 2x + y \leq 600 \\ & x \geq 0,\ y \geq 0 \end{aligned} \qquad [1.23]$$

Graphical solving shows that the optimal solution is now given by $x = 100$ and $y = 400$, which corresponds to $z = 5,600,000$. In other words, increasing the rubber by 100 units changes the turnover by 400,000 euros. We say that the marginal price of a unit of rubber is 4,000 euros.

If the rubber in stock is increased to 600, the optimal solution becomes $x = 0$ and $y = 600$, and the turnover becomes $z = 6,000,000$. Increasing the amount of rubber in stock beyond 600, without also increasing the amount of steel in stock, no longer affects the optimal solution because x is constrained to remain positive.

1.11.3. *Dual problem of the competitor*

Suppose now that the automobile manufacturer has a competitor who offers to repurchase all raw materials in stock to fulfill a large number of orders. The competitor must offer a certain price (assumed to be constant and denoted u) per unit of rubber and another price (denoted v) per unit of steel. For the offer to be accepted, the price paid by the competitor must at least be equal to the turnover that the manufacturer could achieve by manufacturing automobiles itself. The competitor's problem can therefore be written as

$$\begin{aligned} \min\ & p = 400u + 600v \\ \text{s.t.}\ & u + v \geq 10,000 \\ & u + 2v \geq 16,000 \\ & u \geq 0,\ v \geq 0 \end{aligned} \qquad [1.24]$$

Graphical analysis finds the optimal solution $u = 4,000$ and $v = 6,000$, which corresponds to a global price of $p = 5,200,000$. Observe (and we will see later that this is no coincidence) that the optimal solution of the competitor's problem (the dual problem to the manufacturer's primal problem) coincides with the marginal prices of the manufacturer's problem, and the minimum price that the competitor can offer is equal to the manufacturer's maximum turnover.

1.12. Using Matlab

Matlab offers the *linprog* solver for optimizing LPs. The *linprog* solver takes the following input arguments:

– f: the objective function;

– A: the matrix of inequality constraints;

– b: the right-hand side of the inequality constraints;

– Aeq: the matrix of equality constraints;

– beq: the right-hand side of the equality constraints;

– lb: the vector of lower bounds of the decision variables;

– ub: the vector of upper bounds of the decision variables;

– x0: the starting point of x, only used for the active-set algorithm;

– options : options created with "optimoptions".

All *linprog* arguments use the following options, defined via the "optimoptions" input:

– Algorithm: the following algorithms can be used:

1) “interior-point” (default);

2) “dual-simplex”;

3) “active-set”;

4) “simplex”.

– LargeScale: the “interior-point” algorithm is used when this option is set to “on” (default) and the primal simplex algorithm is used when it is set to “off”.

– TolCon: the feasibility tolerance for constraints. The default value is 1e-4. It takes scalar values from 1e-10 to 1e-3 (only available for the dual simplex algorithm).

The output arguments of the *linprog* solver are as follows:

– fval: the value of the objective function at the solution x;

– x: the solution found by the solver. If exitflag > 0, then x is a solution; if not, x is the value of the variables when the solver terminated prematurely;

– exitflag: the reason why the algorithm terminated:

- 1: the solver converged to the solution x;

- 0: the number of iterations exceeded the “MaxIter option”;

- 2: no feasible point was found;

- 3: the LP problem is not bounded;

- 4: a NaN value was encountered when executing the algorithm;

- 5: the primal and dual problems are infeasible;

- 7: the search direction became too small and no other progress was observed.

The following list specifies the various syntaxes that can be used to call the *linprog* solver:

– *x = linprog(f, A, b)*: solves $\min f^T x$ such that Ax $\leq$ b (used when there are only equality constraints);

– *x = linprog(f, A, b, Aeq, beq)*: solves the problem with the equality constraints Aeqx = beq. Set A=[ ] and b=[ ] if no inequalities exist;

– *x = linprog(f, A, b, Aeq, beq, lb, ub)*: defines a set of lower and upper bounds on the design variables, x, so that the solution is always in the range $lb \leq x \leq ub$. Set Aeq=[ ] and beq=[ ] if no equalities exist;

– *x = linprog(f, A, b, Aeq, beq, lb, ub, x0)*: defines the starting point at x0 (only used for the active-set algorithm);

– *x = linprog(f, A, b, Aeq, beq, lb, ub, x0, options)*: minimizes with the specified options;

– *[x, fval] = linprog(...)*: returns the value of the objective function at the solution x;

– *[x, fval, exitflag] = linprog(...)*: returns a value exitflag that describes the exit condition;

– *[x, fval, exitflag, output] = linprog(...)*: returns a structure output that contains information about the optimization process;

– *[x, fval, exitflag, output, lambda] = linprog(...)*: returns a structure output whose fields contain the Lagrange multipliers λ at the solution x.

EXAMPLE 1.14.– Consider the linear program:

$$\begin{array}{lll} \min\ z = -2x_1 - 3x_2 + x_3 + 4x_4 + x_5 & & \\ \text{s.t.}\ \ x_1 + 4x_2 - 2x_3 - x_4 & \leq & 8 \\ \quad x_1 + 3x_2 - 4x_3 - x_4 + x_5 & = & 10 \\ \quad -2x_1 - x_2 - 2x_3 - 3x_4 + x_5 & \leq & -20 \\ \quad x_1 & \geq & 5 \\ \quad x_2 & \geq & 1 \\ \quad x_j \geq 0;\ j = 1, 2, 3, 4, 5 & & \end{array} \qquad [1.25]$$

The following code can be executed to compute this example.

```
variables = {'x1', 'x2', 'x3', 'x4', 'x5'}; % construct the vector x
n = length(variables);
for i = 1:n % create the indices of x
eval([variables{i}, ' = ', num2str(i), ';']);
end
lb = zeros(1, n);
lb([x1, x2]) = [5, 1]; % add the bounds of the variables x1,x2
ub = Inf(1, n);
A = zeros(3, n); % create the matrix A
b = zeros(3, 1); % create the vector b
% define the constraints
A(1, [x1, x2, x3, x4]) = [1, 4, −2, −1]; b(1) = 8;
A(2, [x1, x2, x3, x4]) = [1, 3, 2, −1]; b(2) = 10;
A(3, [x1, x2, x3, x4, x5]) = [2, 1, 2, 3, −1]; b(3) = 20;
Aeq = zeros(1, n); % create the matrix Aeq
beq = zeros(1, 1); % create the vector beq
% define the equality constraints
Aeq(1, [x1, x2, x3, x4, x5]) = [1, 3, −4, −1, 1]; beq(1) = 7;
```

```
19 c = zeros(n, 1); % create the vector c
20 c([x1 x2 x3 x4 x5]) = [−2; −3; 1; 4; 1];
21 % call linprog solver
22 [x, objVal] = linprog(c, A, b, Aeq, beq, lb, ub);
23 for i = 1:n
24 fprintf('%s \t %20.4f\n', variables{i}, x(i))
25 end
26 fprintf(['The value of the objective function is' '%20.4f\n'],
27         objVal)
```

For a color version of this code, see www.iste.co.uk/radi/optimizations.zip

Executing this code produces the following results:

```
Optimization terminated.
x1                  5.5000
x2                  1.0000
x3                  0.7500
x4                  0.0000
x5                  1.5000
The value of the objective function is            -11.7500
```