Chapter 4

Linear Programming

Recall that a general instance of the LINEAR PROGRAMMING problem is described as follows.

LINEAR PROGRAMMING

$$
\begin{array}{rc}
\text{minimize} & c_1x_1 + \cdots + c_nx_n \\
\text{subject to} & a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \geq a_1 \\
& \cdots\cdots \\
& a_{r1}x_1 + a_{r2}x_2 + \ldots + a_{rn}x_n \geq a_r \\
& b_{11}x_1 + b_{12}x_2 + \ldots + b_{1n}x_n \leq b_1 \\
& \cdots\cdots \\
& b_{s1}x_1 + b_{s2}x_2 + \ldots + b_{sn}x_n \leq b_s \\
& d_{11}x_1 + d_{12}x_2 + \ldots + d_{1n}x_n = d_1 \\
& \cdots\cdots \\
& d_{t1}x_1 + d_{t2}x_2 + \ldots + d_{tn}x_n = d_t
\end{array}
\tag{4.1}
$$

where c_i, a_{ji}, a_j, b_{ki}, b_k, d_{li}, and d_l are all given real numbers, for $1 \leq i \leq n$, $1 \leq j \leq r$, $1 \leq k \leq s$, and $1 \leq l \leq t$; and x_i, $1 \leq i \leq n$, are unknown variables.

The LINEAR PROGRAMMING problem is characterized, as the name implies, by linear functions of the unknown variables: the objective function is linear in the unknown variables, and the constraints are linear equalities or linear inequalities in the unknown variables.

For many combinatorial optimization problems, the objective function and the constraints on a solution to an input instance are linear, i.e., they can be formulated by linear equalities and linear inequalities. Therefore, optimal solutions for these combinatorial optimization problems can be derived from optimal solutions for the corresponding instance in the LINEAR PROGRAMMING problem. This is one of the main reasons why the LINEAR PROGRAMMING problem receives so much attention.

For example, consider the MAXIMUM FLOW problem. Let G be an instance of the MAXIMUM FLOW problem. Thus, G is a flow network. Without loss of generality, we can assume that the vertices of G are named by the integers 1, 2, ..., n, where 1 is the source and n is the sink. Each pair of vertices i and j in G is associated with an integer c_{ij}, which is the capacity of the edge $[i, j]$ in G (if there is no edge from i to j, then $c_{ij} = 0$). To formulate the instance G of the MAXIMUM FLOW problem into an instance of the LINEAR PROGRAMMING problem, we introduce n^2 unknown variables f_{ij}, $1 \le i, j \le n$, where the variable f_{ij} is for the amount of flow from vertex i to vertex j. By the definition of flow in a flow network, the flow value f_{ij} must satisfy the capacity constraint, the skew symmetry constraint, and the flow conservation constraint. These constraints can be easily formulated into linear relations:

$$\begin{array}{rll} \text{capacity constraint:} & f_{ij} \le c_{ij} & \text{for all } 1 \le i, j \le n \\ \text{skew symmetry:} & f_{ij} = -f_{ji} & \text{for all } 1 \le i, j \le n \\ \text{flow conservation:} & \sum_{j=1}^{n} f_{ij} = 0 & \text{for } i \ne 1, n \end{array}$$

Finally, the MAXIMUM FLOW problem is to maximize the flow value, which by definition is given by $f_{11} + f_{12} + \cdots + f_{1n}$. This is equivalent to minimizing the value $-f_{11} - f_{12} - \cdots - f_{1n}$. Therefore, the instance G of the MAXIMUM FLOW problem has been formulated into an instance of the LINEAR PROGRAMMING problem as follows.

$$\begin{array}{lcl} \text{minimize} & -f_{11} - f_{12} - \cdots - f_{1n} & \\ \text{subject to} & f_{i,j} \le c_{ij} & \text{for all } 1 \le i, j \le n \\ & f_{ij} + f_{ji} = 0 & \text{for all } 1 \le i, j \le n \\ & \sum_{j=1}^{n} f_{ij} = 0 & \text{for } i \ne 1, n \end{array}$$

An efficient algorithm for the LINEAR PROGRAMMING problem implies an efficient algorithm for the MAXIMUM FLOW problem.

In this chapter, we introduce the basic concepts and efficient algorithms for the LINEAR PROGRAMMING problem. We start by introducing the basic concepts and preliminaries for the LINEAR PROGRAMMING problem. An algorithm, the “simplex method”, is then described. The simplex method is, though not a polynomial time bounded algorithm, very fast for most practical instances of the LINEAR PROGRAMMING problem. We will also discuss the idea of the *dual* LINEAR PROGRAMMING problem, which can be used to solve the original LINEAR PROGRAMMING problem more efficiently than by simply applying the simplex method to the original problem. Finally, polynomial time algorithms for the LINEAR PROGRAMMING problem will be briefly introduced.

We assume in this chapter the familiarity of the fundamentals of linear algebra. In particular, we assume that the readers are familiar with the definitions of vectors, matrices, linear dependency and linear independency, and know how a system of linear equations can be solved. All these can be found in any introductory book in Linear Algebra. To avoid confusions, we will use little bold letters such as $\mathbf{x}$ and $\mathbf{c}$ for vectors, and use capital bold letters such as $\mathbf{A}$ and $\mathbf{B}$ for matricies. For a vector $\mathbf{x}$ and a real number c, we write $\mathbf{x} \geq c$ if all elements in $\mathbf{x}$ are larger than or equal to c.

4.1 Basic concepts

First note that in the constraints in a general instance in (4.1) of the LINEAR PROGRAMMING problem, there is no strict inequalities. Mathematically, any bounded set defined by linear equalities and non-strict linear inequalities is a "compact set" in the Euclidean space, in which the objective function can always achieve its optimal value, while strict linear inequalities define a non-compact set in which the objective function may not be able to achieve its optimal value. For example, consider the following instance:

$$\begin{array}{rc} \text{minimize} & -x_1 - x_2 \\ \text{subject to} & x_1 + x_2 + x_3 < 1 \\ & x_1 \geq 0, \;\; x_2 \geq 0, \;\; x_3 \geq 0 \end{array}$$

The set S defined by the constraints $x_1 + x_2 + x_3 < 1$, $x_1 \geq 0$, $x_2 \geq 0$, and $x_3 \geq 0$ is certainly bounded. However, no vector (x_1, x_2, x_3) in S can make the objective function $-x_1 - x_2$ to achieve the minimum value: for any $\epsilon > 0$, we can find a vector (x_1, x_2, x_3) in the set S that makes the objective function $-x_1 - x_2$ to have value less than $-1 + \epsilon$ but no vector in the set S can make the objective function $-x_1 - x_2$ to have value less than or equal to -1.

Now we show how a general instance in (4.1) of the LINEAR PROGRAMMING problem can be converted into a simpler form.

The *standard form* for the LINEAR PROGRAMMING problem is given in the following format

$$\begin{aligned} \text{minimize} \quad & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\ \text{subject to} \quad & a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1 \\ & a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2 \\ & \cdots\cdots \\ & a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n = b_m \\ & x_1 \geq 0, \quad x_2 \geq 0, \quad \ldots, \quad x_n \geq 0 \end{aligned} \tag{4.2}$$

The general form in (4.1) of the LINEAR PROGRAMMING problem can be converted into the standard form in (4.2) through the following steps.

1. **Eliminating "$\geq$" inequalities**
   Each inequality $a_{i1}x_1 + a_{i2}x_2 + ... + a_{in}x_n \geq a_i$ is replaced by the equivalent inequality $(-a_{i1})x_1 + (-a_{i2})x_2 + ... + (-a_{in})x_n \leq (-a_i)$.

2. **Eliminating "$\leq$" inequalities**
   Each inequality $b_{j1}x_1 + b_{j2}x_2 + ... + b_{jn}x_n \leq b_j$ is replaced by the equality $b_{j1}x_1 + b_{j2}x_2 + ... + b_{jn}x_n + y_j = b_j$ by introducing a new *slack variable* y_j with the constraint $y_j \geq 0$.

3. **Eliminating unconstrained variables**
   For each variable x_i such that the constraint $x_i \geq 0$ is not present, introduce two new variables u_i and v_i satisfying $u_i \geq 0$ and $v_i \geq 0$, and replace the variable x_i by $u_i - v_i$.

The above transformation rules are not strict. For example, the $\geq$ inequalities can also be eliminated using a "surplus variable". Moreover, sometimes a simple linear transformation may be more convenient and more effective than the above transformations. We illustrate these transformations and other possible transformations by an example. Consider the instance in (4.3) for the LINEAR PROGRAMMING problem.

$$\begin{aligned} \text{minimize} \quad & 2x_1 + x_2 - 3x_3 \\ \text{subject to} \quad & 2x_1 - x_2 - 7x_3 \geq 5 \\ & 2x_2 - x_3 = 3 \\ & x_2 \geq 2 \end{aligned} \tag{4.3}$$

We apply the first rule to convert the first constraint $2x_1 - x_2 - 7x_3 \geq 5$ into $-2x_1 + x_2 + 7x_3 \leq -5$. Then we apply the second rule and introduce a new slack variable x_4 with constraint $x_4 \geq 0$ to get an equality $-2x_1 + x_2 + 7x_3 + x_4 = -5$.

For the constraint $x_2 \geq 2$, we could also convert it into an equality using the first and second rules. However, we can also perform a simple linear transformation as follows. Let $x_2' = x_2 - 2$ and replace in (4.3) the variable x_2 by $x_2' + 2$. This combined with the transformations on the first constraint will convert the instance (4.3) into the form

$$\begin{array}{ll} \text{minimize} & 2x_1 + x_2' - 3x_3 \\ \text{subject to} & -2x_1 + x_2' + 7x_3 + x_4 = -7 \\ & 2x_2' - x_3 = -1 \\ & x_2' \geq 0, \quad x_4 \geq 0 \end{array} \tag{4.4}$$

Note that after the linear transformation $x_2 = x_2' + 2$, the objective function $2x_1 + x_2 - 3x_3$ should have become $2x_1 + x_2' - 3x_3 + 2$. However, minimizing $2x_1 + x_2' - 3x_3 + 2$ is equivalent to minimizing $2x_1 + x_2' - 3x_3$.

Now we need to remove the unconstrained variables in the instance (4.4). For the unconstrained variable x_1, by the third rule, we introduce two new variables x_1' and x_1'' with constraints $x_1' \geq 0$ and $x_1'' \geq 0$, and replace in (4.4) x_1 by $x_1' - x_1''$. We obtain

$$\begin{array}{ll} \text{minimize} & 2x_1' - 2x_1'' + x_2' - 3x_3 \\ \text{subject to} & -2x_1' + 2x_1'' + x_2' + 7x_3 + x_4 = -7 \\ & 2x_2' - x_3 = -1 \\ & x_1' \geq 0, \quad x_1'' \geq 0 \;\; x_2' \geq 0, \quad x_4 \geq 0 \end{array} \tag{4.5}$$

The unconstrained variable x_3 could also be eliminated using the same rule. But it can also be eliminated using a simple linear transformation. For this, we observe the constraint $2x_2' - x_3 = -1$ so $x_3 = 2x_2' + 1$. Thus, replacing x_3 in (4.5) by $2x_2' + 1$, we obtain the following standard form for the LINEAR PROGRAMMING problem.

$$\begin{array}{ll} \text{minimize} & 2x_1' - 2x_1'' - 5x_2' \\ \text{subject to} & -2x_1' + 2x_1'' + 15x_2' + x_4 = -14 \\ & x_1' \geq 0, \quad x_1'' \geq 0 \;\; x_2' \geq 0 \;\; x_4 \geq 0 \end{array} \tag{4.6}$$

It is easy to verify that if we solve the instance (4.6) and obtain an optimal solution (x_1', x_1'', x_2', x_4), then we can construct an optimal solution (x_1, x_2, x_3) for the instance (4.3), where $x_1 = x_1' - x_1''$, $x_2 = x_2' + 2$, and $x_3 = 2x_2' + 1$.

Note that the transformations do not result in an instance whose size is much larger than the original instance. In fact, to eliminate an inequality, we need to introduce at most one new variable y plus a new constraint $y \geq 0$, and to eliminate an unconstrained variable we need to introduce at most two new variables u and v plus two new constraints $u \geq 0$ and $v \geq 0$. Therefore, if the original instance consists of n variables and m constraints, then the corresponding instance in the standard form consists of at most $2n + m$ variables and $2n + 2m$ constraints.

Therefore, without loss of generality, we can always assume that a given instance of the LINEAR PROGRAMMING problem is in the standard form. Using our 4-tuple formulation, the LINEAR PROGRAMMING problem can now be formulated as follows.

LINEAR PROGRAMMING $= \langle I_Q, S_Q, f_Q, opt_Q \rangle$, where

- I_Q is the set of triples $(\mathbf{b}, \mathbf{c}, \mathbf{A})$, where $\mathbf{b}$ is an m-dimensional vector of real numbers, $\mathbf{c}$ is an n-dimensional vector of real numbers, and $\mathbf{A}$ is an $m \times n$ matrix of real numbers, for some integers n and m;
- for an instance $\alpha = (\mathbf{b}, \mathbf{c}, \mathbf{A})$ in I_Q, the solution set $S_Q(\alpha)$ consists of the set of n-dimensional vectors $\mathbf{x}$ that satisfy the constraints $\mathbf{Ax} = \mathbf{b}$ and $\mathbf{x} \geq 0$;
- for a given input instance $\alpha \in I_Q$ and a solution $\mathbf{x} \in S_Q(\alpha)$, the objective function value is defined to be the inner product $\mathbf{c}^T\mathbf{x}$ of the vectors $\mathbf{c}$ and $\mathbf{x}$;
- opt_Q is min.

We make a further assumption that the $m \times n$ matrix $\mathbf{A}$ in an instance of the LINEAR PROGRAMMING problem has its m rows linearly independent, which also implies that $m \leq n$. This assumption can be justified as follows. If the m rows of the matrix $\mathbf{A}$ are not linearly independent, then either the constraint $\mathbf{Ax} = \mathbf{b}$ is contradictory, in which case the instance obviously has no solution, or there are redundancy in the constraint. The redundancy in the constraint can be eliminated using standard linear algebra techniques such as the well-known Gaussian Elimination algorithm.

Under these assumptions, we can assume that there are m columns in the matrix $\mathbf{A}$ that are linearly independent. Without loss of generality, suppose that the first m columns of $\mathbf{A}$ are linearly independent and let $\mathbf{B}$ be the nonsingular $m \times m$ submatrix of $\mathbf{A}$ such that $\mathbf{B}$ consists of the first m columns of $\mathbf{A}$. Let $\mathbf{x_B} = (x_1, x_2, \ldots, x_m)^T$ be the m-dimensional vector that consists of the first m unknown variables in the vector $\mathbf{x}$. Since the matrix $\mathbf{B}$ is nonsingular, the equation

$$\mathbf{B}\mathbf{x}_{\mathrm{B}} = \mathbf{b}$$

has a unique solution $\mathbf{x}^0_{\mathbf{B}} = \mathbf{B}^{-1}\mathbf{b}$. If we let $\mathbf{x}^0 = (\mathbf{x}^0_{\mathbf{B}}, \overbrace{0, \ldots, 0}^{n-m})$, then obviously, $\mathbf{x}^0$ is a solution to the system $\mathbf{A}\mathbf{x} = \mathbf{b}$. If the vector $\mathbf{x}^0$ happens to also satisfy the constraint $\mathbf{x}^0 \geq 0$, then $\mathbf{x}^0$ is a solution to the instance of the Linear Programming problem

$$\begin{array}{ll} \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{and} \quad \mathbf{x} \geq 0 \end{array} \tag{4.7}$$

This introduces a very important class of solutions to an instance of the Linear Programming problem, formally defined as follows.

Definition 4.1.1 A vector $\mathbf{x}^0 = (x^0_1, x^0_2, \ldots, x^0_n)^T$ satisfying $\mathbf{A}\mathbf{x}^0 = \mathbf{b}$ and $\mathbf{x}^0 \geq 0$ is a *basic solution* if there are m indices $1 \leq i_1 < i_2 < \ldots < i_m \leq n$ such that the i_1th, i_2th, ..., i_mth columns of the matrix $\mathbf{A}$ are linearly independent, and $x^0_i = 0$ for all $i \notin \{i_1, \ldots, i_m\}$. These m columns of the matrix $\mathbf{A}$ will be called the *basic columns* for $\mathbf{x}^0$.

Note that we did not exclude the possibility that $x^0_{i_j} = 0$ for some index i_j in the basic solution $\mathbf{x}^0$. If any element $x^0_{i_j} = 0$ in the above basic solution $\mathbf{x}^0$, the basic solution $\mathbf{x}^0$ is called a *degenerate basic solution*.

The following theorem is fundamental in the study of the Linear Programming problem.

Theorem 4.1.1 *Let $\alpha = (\mathbf{b}, \mathbf{c}, \mathbf{A})$ be an instance for the* Linear Programming *problem. If the solution set $S_Q(\alpha)$ is not empty, then $S_Q(\alpha)$ contains a basic solution. Moreover, if the objective function $\mathbf{c}^T\mathbf{x}$ achieves the minimum value at a vector $\mathbf{x}^0$ in $S_Q(\alpha)$, then there is a basic solution $\mathbf{x}^0_b$ in $S_Q(\alpha)$ such that $\mathbf{c}^T\mathbf{x}^0_b = \mathbf{c}^T\mathbf{x}^0$.*

Proof. Suppose that $S_Q(\alpha) \neq \emptyset$. Let $\mathbf{x}_b = (x_1, x_2, \ldots, x_n)^T$ be a solution to the instance α such that $\mathbf{x}_b$ has the maximum number of 0 elements over all solutions in $S_Q(\alpha)$. We show that $\mathbf{x}_b$ must be a basic solution.

For convenience, we suppose that the first p elements x_1, x_2, ..., x_p in $\mathbf{x}_b$ are larger than 0 and all other elements in $\mathbf{x}_b$ are 0. Let the n column vectors of the matrix $\mathbf{A}$ be $\mathbf{a}_1$, $\mathbf{a}_2$, ..., $\mathbf{a}_n$. Then the equality $\mathbf{A}\mathbf{x}_b = \mathbf{b}$ can be written as $x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \cdots + x_n\mathbf{a}_n = \mathbf{b}$. Since $x_{p+1} = \cdots x_n = 0$, this equality is equivalent to

$$x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \cdots + x_p\mathbf{a}_p = \mathbf{b} \tag{4.8}$$

If $\mathbf{x}_b$ is not a basic solution, then the column vectors $\mathbf{a}_1, \ldots, \mathbf{a}_p$ are linearly dependent. Thus, there are p real numbers $y_1, \ldots, y_p$ such that at least one y_i is positive and that

$$y_1\mathbf{a}_1 + y_2\mathbf{a}_2 + \cdots + y_p\mathbf{a}_p = \mathbf{0} \tag{4.9}$$

where $\mathbf{0}$ denotes the m-dimensional vector $(0, 0, \ldots, 0)^T$. Let ϵ be a constant. Subtract ϵ times the equality (4.9) from the equality (4.8), we get

$$(x_1 - \epsilon y_1)\mathbf{a}_1 + (x_2 - \epsilon y_2)\mathbf{a}_2 + \cdots + (x_p - \epsilon y_p)\mathbf{a}_p = \mathbf{b} \tag{4.10}$$

Equality (4.10) holds for any constant ϵ. Since at least one y_i is positive, the value $\epsilon_0 = \min\{x_i/y_i \mid y_i > 0\}$ is well-defined and $\epsilon_0 > 0$ (note that $x_i > 0$ for all $1 \leq i \leq p$). Again for convenience, suppose that $y_p > 0$ and $\epsilon_0 = x_p/y_p$. With this choice of ϵ_0, we have $x_i - \epsilon_0 y_i \geq 0$ for all $1 \leq i \leq p$. Thus, in equality (4.10), if we let $z_i = x_i - \epsilon_0 y_i$ for all $1 \leq i \leq p$, we will get

$$z_1\mathbf{a}_1 + z_2\mathbf{a}_2 + \cdots + z_{p-1}\mathbf{a}_{p-1} = \mathbf{b}$$

and

$$z_1 \geq 0, \quad z_2 \geq 0, \quad \ldots, \quad z_{p-1} \geq 0$$

Now if we let $\mathbf{z} = (z_1, z_2, \ldots, z_{p-1}, 0, 0, \ldots, 0)^T$ be the n-dimensional vector with the last $n - p + 1$ elements all equal to 0, we will get

$$\mathbf{A}\mathbf{z} = \mathbf{b} \quad \text{and} \quad \mathbf{z} \geq 0$$

Thus, $\mathbf{z}$ is a solution to the instance α and $\mathbf{z}$ has at least $n-p+1$ elements equal to 0. However, this contradicts our assumption that the vector $\mathbf{x}_b$ is a solution to the instance α with the maximum number of 0 elements over all solutions to α. This contradiction shows that the vector $\mathbf{x}_b$ must be a basic solution to α.

This proves that if $S_Q(\alpha) \neq \emptyset$, then $S_Q(\alpha)$ contains basic solutions.

Now suppose that there is a solution $\mathbf{x}^0$ in $S_Q(\alpha)$ such that $\mathbf{c}^T\mathbf{x}^0$ is the minimum over all solutions in $S_Q(\alpha)$. We pick from $S_Q(\alpha)$ a solution $\mathbf{x}_b^0 = (x_1^0, \ldots, x_n^0)$ such that $\mathbf{c}^T\mathbf{x}_b^0 = \mathbf{c}^T\mathbf{x}^0$ and $\mathbf{x}_b^0$ has the maximum number of 0 elements over all solutions $\mathbf{x}$ in $S_Q(\alpha)$ satisfying $\mathbf{c}^T\mathbf{x} = \mathbf{c}^T\mathbf{x}^0$. We show that $\mathbf{x}_b^0$ is a basic solution. As we proceeded before, we assume that the first

p elements in $\mathbf{x}_b^0$ are positive and all other elements in $\mathbf{x}_b^0$ are 0. If $\mathbf{x}_b^0$ is not a basic solution, then we can find p real numbers $y_1, \ldots, y_p$ in which at least one y_i is positive such that

$$(x_1^0 - \epsilon y_1)\mathbf{a}_1 + (x_2^0 - \epsilon y_2)\mathbf{a}_2 + \cdots + (x_p^0 - \epsilon y_p)\mathbf{a}_p = \mathbf{b}$$

for any constant ϵ. Now if we let $\mathbf{y} = (y_1, y_2, \ldots, y_p, 0, \ldots, 0)^T$ be the n-dimensional vector with the last $n-p$ elements equal to 0, then $\mathbf{A}(\mathbf{x}_b^0 - \epsilon\mathbf{y})$ $= \mathbf{b}$ for any ϵ. Since $x_i > 0$ for $1 \le i \le p$ and $x_j = y_j = 0$ for $j > p$, we have $\mathbf{x}_b^0 - \epsilon\mathbf{y} \ge 0$ for small enough (positive or negative) ϵ. Thus, for any small enough ϵ, $\mathbf{z}_\epsilon = \mathbf{x}_b^0 - \epsilon\mathbf{y} \ge 0$ is a solution to the instance α. Now consider the objective function value $\mathbf{c}^T\mathbf{z}_\epsilon$. We have

$$\mathbf{c}^T\mathbf{z}_\epsilon = \mathbf{c}^T\mathbf{x}_b^0 - \epsilon\mathbf{c}^T\mathbf{y}$$

We claim that we must have $\mathbf{c}^T\mathbf{y} = \mathbf{0}$. In fact, if $\mathbf{c}^T\mathbf{y} \ne \mathbf{0}$, then pick a proper small ϵ, we will have $\epsilon\mathbf{c}^T\mathbf{y} > 0$. But this implies that the value $\mathbf{c}^T\mathbf{z}_\epsilon =$ $\mathbf{c}^T\mathbf{x}_b^0 - \epsilon\mathbf{c}^T\mathbf{y}$ is smaller than $\mathbf{c}^T\mathbf{x}_b^0$, and $\mathbf{z}_\epsilon$ is a solution in $S_Q(\alpha)$, contradicting our assumption that $\mathbf{x}_b^0$ minimizes the value $\mathbf{c}^T\mathbf{x}$ over all solutions $\mathbf{x}$ in $S_Q(\alpha)$.

Thus, we must have $\mathbf{c}^T\mathbf{y} = \mathbf{0}$. In consequence, $\mathbf{c}^T\mathbf{z}_\epsilon = \mathbf{c}^T\mathbf{x}_b^0$ for any ϵ. Now if we let $\epsilon_0 = \min\{x_i/y_i \mid y_i > 0\}$, and let $\mathbf{z}_0 = \mathbf{x}_b^0 - \epsilon_0\mathbf{y}$, then we have $\mathbf{c}^T\mathbf{z}_0 = \mathbf{c}^T\mathbf{x}_b^0 = \mathbf{c}^T\mathbf{x}^0$, $\mathbf{A}\mathbf{z}_0 = \mathbf{b}$, $\mathbf{z}_0 \ge 0$, and $\mathbf{z}_0$ has at least $n-p+1$ elements equal to 0. However, this contradicts our assumption that $\mathbf{x}_b^0$ is a solution in $S_Q(\alpha)$ with the maximum number of 0 elements over all solutions $\mathbf{x}$ satisfying $\mathbf{c}^T\mathbf{x} = \mathbf{c}^T\mathbf{x}^0$. This contradiction shows that the vector $\mathbf{x}_b^0$ must be a basic solution.

This completes the proof of the theorem. □

Theorem 4.1.1 reduces the problem of finding an optimal solution for an instance of the LINEAR PROGRAMMING problem to the problem of finding an optimal basic solution for the instance. According to the theorem, if the instance has an optimal solution, then the instance must have an optimal solution that is a basic solution. Note that in general there are infinitely many solutions to a given instance while the number of basic solutions is always finite — it is bounded by the number of ways of choosing m columns from the n columns of the matrix $\mathbf{A}$. Moreover, all these basic solutions can be constructed systematically: pick every m columns from the matrix $\mathbf{A}$, check if they are linearly independent. In case the m columns are linearly independent, a unique m-dimensional vector $\mathbf{x} = \mathbf{B}^{-1}\mathbf{b}$ can be constructed using standard linear algebra techniques, where $\mathbf{B}$ is the submatrix consisting of the m columns of $\mathbf{A}$. Now if this vector $\mathbf{x}$ also satisfies $\mathbf{x} \ge 0$, then we can expand $\mathbf{x}$ into an n-dimensional vector $\mathbf{x}_0$ by inserting properly $n - m$ 0's. The vector $\mathbf{x}_0$ is then the basic solution with these m linearly independent columns as basic columns.

Algorithmically, there can be still too many basic solutions for us to search for the optimal one — the number of ways of choosing m columns from the n columns of the matrix $\mathbf{A}$ is $\binom{n}{m}$, which is of order $\Theta(n^m)$. In the next section, we introduce the simplex method, which provides a more effective way to search for an optimal basic solution among all basic solutions.

Theorem 4.1.1 has an interesting interpretation from the view of geometry. Given an instance α of the LINEAR PROGRAMMING problem, each solution $\mathbf{x}$ to α can be regarded as a point in the n-dimensional Euclidean space $\mathcal{E}^n$. Thus, the solution set $S_Q(\alpha)$ of α is a subset in the Euclidean space $\mathcal{E}^n$. In fact, $S_Q(\alpha)$ is a *convex set* in $\mathcal{E}^n$ in the sense that for any two points $\mathbf{x}$ and $\mathbf{y}$ in $S_Q(\alpha)$, the entire line segment connecting $\mathbf{x}$ and $\mathbf{y}$ is also in $S_Q(\alpha)$. An example of convex sets in 3-dimensional Euclidean space $\mathcal{E}^3$ is a convex polyhedron. An *extreme point* in a convex set S is a point that is not an interior point of any line segment in S. For example, each vertex in a convex polyhedron P in $\mathcal{E}^3$ is an extreme point of P. It can be formally proved that the basic solutions in $S_Q(\alpha)$ correspond exactly to the extreme points in $S_Q(\alpha)$. From this point of view, Theorem 4.1.1 claims that if $S_Q(\alpha)$ is not empty then $S_Q(\alpha)$ has at least one extreme point, and that if a point in $S_Q(\alpha)$ achieves the optimal objective function value, then some extreme point in $S_Q(\alpha)$ should also achieve the optimal objective function value.

4.2 The simplex method

Theorem 4.1.1 claims that in order to solve the LINEAR PROGRAMMING problem, we only need to concentrate on basic solutions. This observation motivates the classical *simplex method.* Essentially, the simplex method starts with a basic solution, and repeatedly moves from a basic solution to a better basic solution until the optimal basic solution is achieved. Three immediate questions are suggested by this approach:

1. How do we find the first basic solution?
2. How do we move from one basic solution to a better one? and
3. How do we realize that an optimal basic solution has been achieved?

We first discuss the solutions to the second and the third questions. A solution to the first question can be easily obtained when the solutions to the second and the third are available.

Many arguments in the LINEAR PROGRAMMING problem are substantially simplified upon the introduction of the following assumption.

> **Nondegeneracy Assumption.** For an instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$ of the LINEAR PROGRAMMING problem, we assume that all basic solutions to α are nondegenerate.

This assumption is invoked throughout our development of the simplex method, since when it does not hold the simplex method can break down if it is not suitably amended. This assumption, however, should be regarded as one made primarily for convenience, since all arguments can be extended to include degeneracy, and the simplex method itself can be easily modified to account for it. After the whole system of methods is established, we will mention briefly how the situation of degeneracy is handled.

In the following discussion, we will fix an instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$ of the LINEAR PROGRAMMING problem, where $\mathbf{b}$ is an m-dimensional vector, $\mathbf{c}$ is an n-dimensional vector, $m \leq n$, and $\mathbf{A}$ is an $m \times n$ matrix whose m rows are linearly independent. Let the n column vectors of the matrix $\mathbf{A}$ be $\mathbf{a}_1$, $\mathbf{a}_2$, …, $\mathbf{a}_n$.

How to move to a neighbor basic solution

Let $\mathbf{x}$ be a basic solution to the instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$ such that the i_1th, i_2th, …, i_mth elements in $\mathbf{x}$ are positive and all other elements in $\mathbf{x}$ are 0. Let $\mathbf{x}'$ be another basic solution to α such that the i'_1th, i'_2th, …, i'_mth elements in $\mathbf{x}'$ are positive and all other elements in $\mathbf{x}'$ are 0. The basic solution $\mathbf{x}'$ is a *neighbor basic solution to* $\mathbf{x}$ if the index sets $\{t_1, \ldots, t_m\}$ and $\{t'_1, \ldots, t'_m\}$ have $m-1$ indices in common. For a give basic solution $\mathbf{x}$, the simplex method looks at neighbor basic solutions to $\mathbf{x}$ and tries to find one that is better than the current basic solution $\mathbf{x}$.

For the convenience of our discussion, we will suppose that the basic solution $\mathbf{x}$ has the first m elements being positive:

$$\mathbf{x} = (x_1, \ldots, x_m, 0, \ldots, 0) \tag{4.11}$$

Since $\mathbf{x}$ is a basic solution to the instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$, we have

$$x_1\mathbf{a}_1 + x_2\mathbf{x}_2 + \cdots + x_m\mathbf{a}_m = \mathbf{b} \tag{4.12}$$

By the definition, the m m-dimensional vectors $\mathbf{a}_1$, …, $\mathbf{a}_m$ are linearly independent. Therefore, every column vector $\mathbf{a}_q$ of the matrix $\mathbf{A}$ can be represented as a linear combination of the vectors $\mathbf{a}_1$, …, $\mathbf{a}_m$:

$$\mathbf{a}_q = y_{1q}\mathbf{a}_1 + y_{2q}\mathbf{a}_2 + \cdots + y_{mq}\mathbf{a}_m \quad \text{for } q = 1, \ldots, n \tag{4.13}$$

or

$$y_{1q}\mathbf{a}_1 + y_{2q}\mathbf{a}_2 + \cdots + y_{mq}\mathbf{a}_m - \mathbf{a}_q = \mathbf{0} \tag{4.14}$$

where $\mathbf{0}$ is the m-dimensional vector with all elements equal to 0. Let ϵ be a constant. Substract ϵ times the equality (4.14) from the equality (4.12),

$$(x_1 - \epsilon y_{1q})\mathbf{a}_1 + (x_2 - \epsilon y_{2q})\mathbf{a}_2 + \cdots + (x_m - \epsilon y_{mq})\mathbf{a}_m + \epsilon \mathbf{a}_q = \mathbf{b} \tag{4.15}$$

The equality (4.15) holds for all constant ϵ. In particular, when $\epsilon = 0$, it corresponds to the basic solution $\mathbf{x}$ and for ϵ being a small positive number, it corresponds to a non-basic solution (note that by the Nondegeneracy Assumption, $x_i > 0$ for $1 \leq i \leq m$). Now if we let ϵ be increased from 0, then the coefficient of the vector $\mathbf{a}_q$ in the equality (4.15) is increased, and the coefficients of the other vectors $\mathbf{a}_i$, $i \neq q$, in the equality (4.15) are either increased (when $y_{iq} < 0$), unchanged (when $y_{iq} = 0$), or decreased (when $y_{iq} > 0$). Therefore, if there is a positive y_{iq}, then we can let ϵ be the smallest positive number that makes $x_p - \epsilon y_{pq} = 0$ for some p, $1 \leq p \leq m$. This ϵ corresponds to the value

$$\epsilon_0 = x_p / y_{pq} = \min\{x_i / y_{iq} \mid y_{iq} > 0 \text{ and } 1 \leq i \leq m\}$$

Note that with this value ϵ_0, all coefficients in the equality (4.15) are non-negative, the coefficient of $\mathbf{a}_q$ is positive, and the coefficient of $\mathbf{a}_p$ becomes 0. Therefore, in this case, the vector

$$\begin{aligned} \mathbf{x}' \;=\; & (x_1 - \epsilon_0 y_{1q}, \ldots, x_{p-1} - \epsilon_0 y_{p-1,q}, 0, x_{p+1} - \epsilon_0 y_{p+1,q}, \ldots, \\ & \ldots, x_m - \epsilon_0 y_{mq}, 0, \ldots, 0, \epsilon_0, 0, \ldots, 0) \end{aligned} \tag{4.16}$$

satisfies $\mathbf{A}\ \mathbf{x}' = \mathbf{b}$ and $\mathbf{x}' \geq 0$, and has at most m nonzero elements, where the element ϵ_0 in $\mathbf{x}'$ is at the qth position. These m possibly nonzero elements in $\mathbf{x}'$ correspond to the m columns $\mathbf{a}_1$, $\ldots$, $\mathbf{a}_{p-1}$, $\mathbf{a}_{p+1}$, $\ldots$, $\mathbf{a}_m$, $\mathbf{a}_q$ of the matrix $\mathbf{A}$. By our assumption $y_{pq} > 0$, thus by equality (4.14), we have

$$\begin{aligned} \mathbf{a}_p \;=\; & (-y_{1q}/y_{pq})\mathbf{a}_1 + \cdots + (-y_{p-1,q}/y_{pq})\mathbf{a}_{p-1} + (-y_{p+1,q}/y_{pq})\mathbf{a}_{p+1} + \\ & + \cdots + (-y_{mq}/y_{pq})\mathbf{a}_m + (1/y_{pq})\mathbf{a}_q \end{aligned} \tag{4.17}$$

That is, the vector $\mathbf{a}_p$ can be represented by a linear combination of the vectors $\mathbf{a}_1$, $\ldots$, $\mathbf{a}_{p-1}$, $\mathbf{a}_{p+1}$, $\ldots$, $\mathbf{a}_m$, $\mathbf{a}_q$. Since the vectors $\mathbf{a}_1$, $\mathbf{a}_2$, $\ldots$, $\mathbf{a}_m$ are linearly independent, Equality (4.17) implies that the vectors $\mathbf{a}_1$, $\ldots$, $\mathbf{a}_{p-1}$, $\mathbf{a}_{p+1}$, $\ldots$, $\mathbf{a}_m$, $\mathbf{a}_q$ are linearly independent (see Appendix C). Hence, the vector $\mathbf{x}'$ is in fact a basic solution to the instance $\alpha = (\mathbf{b}, \mathbf{c}, \mathbf{A})$. Moreover, $\mathbf{x}'$ is a neighbor basic solution to the basic solution $\mathbf{x}$.

Let us consider how each column vector $\mathbf{a}_i$ of the matrix $\mathbf{A}$ is represented by a linear combination of this new group of linearly independent vectors $\mathbf{a}_1, \ldots, \mathbf{a}_{p-1}, \mathbf{a}_{p+1}, \ldots, \mathbf{a}_m, \mathbf{a}_q$. By equality (4.13), we have

$$\mathbf{a}_i = y_{1i}\mathbf{a}_1 + y_{2i}\mathbf{a}_2 + \cdots + y_{mi}\mathbf{a}_m \tag{4.18}$$

Replace $\mathbf{a}_p$ in (4.18) by the expression in (4.17) and reorganize the equality, we get

$$\begin{aligned} \mathbf{a}_i \;=\; & (y_{1i} - y_{pi}y_{1q}/y_{pq})\mathbf{a}_1 + \cdots + (y_{p-1,i} - y_{pi}y_{p-1,q}/y_{pq})\mathbf{a}_{p-1} \\ & +(y_{pi}/y_{pq})\mathbf{a}_q + (y_{p+1,i} - y_{pi}y_{p+1,q}/y_{pq})\mathbf{a}_{p+1} + \\ & + \cdots + (y_{mi} - y_{pi}y_{mq}/y_{pq})\mathbf{a}_m \end{aligned} \tag{4.19}$$

Thus, the column $\mathbf{a}_q$ replaces the column $\mathbf{a}_p$ and becomes the pth basic column for the basic solution.

The above transformation from the basic solution $\mathbf{x}$ to the neighbor basic solution $\mathbf{x}'$ can be conveniently managed in the form of a tableau. For the basic solution $\mathbf{x} = (x_1, \ldots, x_m, 0, \ldots, 0)$, and suppose that the last $n - m$ columns $\mathbf{a}_q$, $m + 1 \le q \le n$, of the matrix $\mathbf{A}$ are given by the linear combinations of the columns $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m$ in equality (4.13), then the tableau corresponding to the basic solution $\mathbf{x}$ is given as

$\mathbf{a}_1$	$\cdots$	$\mathbf{a}_p$	$\cdots$	$\mathbf{a}_m$	$\mathbf{a}_{m+1}$	$\cdots$	$\mathbf{a}_q$	$\cdots$	$\mathbf{a}_n$	
1	$\cdots$	0	$\cdots$	0	$y_{1,m+1}$	$\cdots$	y_{1q}	$\cdots$	y_{1n}	x_1
.		.		.	.		.		.	.
0	$\cdots$	1	$\cdots$	0	$y_{p,m+1}$	$\cdots$	y_{pq}	$\cdots$	y_{pn}	x_p
.		.		.	.		.		.	.
0	$\cdots$	0	$\cdots$	1	$y_{m,m+1}$	$\cdots$	y_{mq}	$\cdots$	y_{mn}	x_m

In order to move from the basic solution $\mathbf{x}$ to the neighbor basic solution $\mathbf{x}'$ by replacing the column $\mathbf{a}_p$ by the column $\mathbf{a}_q$ (assume that $y_{pq} > 0$ and x_p/y_{pq} is the minimum x_i/y_{iq} over all i such that $1 \le i \le m$ and $y_{iq} > 0$), we only need to perform the following row transformations on the tableau: (1) divide the pth row of the tableau by y_{pq}; and (2) for each row j, $j \ne p$, subtract y_{jq} times the pth row from the jth row. After these transformations, the tableau becomes

$\mathbf{a}_1$	$\cdots$	$\mathbf{a}_p$	$\cdots$	$\mathbf{a}_m$	$\mathbf{a}_{m+1}$	$\cdots$	$\mathbf{a}_q$	$\cdots$	$\mathbf{a}_n$	
1	$\cdots$	y'_{1p}	$\cdots$	0	$y'_{1,m+1}$	$\cdots$	0	$\cdots$	y'_{1n}	x'_1
.		.		.	.		.		.	.
0	$\cdots$	y'_{pp}	$\cdots$	0	$y'_{p,m+1}$	$\cdots$	1	$\cdots$	y'_{pn}	x'_p
.		.		.	.		.		.	.
0	$\cdots$	y'_{mp}	$\cdots$	1	$y'_{m,m+1}$	$\cdots$	0	$\cdots$	y'_{mn}	x'_m

Thus, the qth column in the tableau, which corresponds to column $\mathbf{a}_q$, now becomes a vector whose pth element is 1 and all other elements are 0.

Consider the pth column $\mathbf{a}_p$ in the tableau. We have

$$y'_{pp} = 1/y_{pq} \tag{4.20}$$

and

$$y'_{jp} = -y_{jq}/y_{pq} \quad \text{for } 1 \le i \le m \text{ and } j \ne p \tag{4.21}$$

By equality (4.17), we get

$$\mathbf{a}_p = y'_{1p}\mathbf{a}_1 + \cdots + y'_{p-1,p}\mathbf{a}_{p-1} + y'_{pp}\mathbf{a}_q + y'_{p+1,p}\mathbf{a}_{p+1} + \cdots + y'_{mp}\mathbf{a}_m$$

Therefore, the pth column in the new tableau gives exactly the coefficients of the linear combination for the column $\mathbf{a}_p$ in terms of the new basic columns $\mathbf{a}_1, \ldots, \mathbf{a}_{p-1}, \mathbf{a}_q, \mathbf{a}_{p+1}, \ldots, \mathbf{a}_m$.

For the ith column $\mathbf{a}_i$ in the tableau, where $m+1 \le i \le n$ and $i \ne q$, we have

$$y'_{pi} = y_{pi}/y_{pq} \tag{4.22}$$

and

$$y'_{ji} = y_{ji} - y_{jq}y_{pi}/y_{pq} \quad \text{for } 1 \le j \le m \text{ and } j \ne p \tag{4.23}$$

By eqality (4.19), the ith column in the tableau gives exactly the coefficients of the linear combination for the column $\mathbf{a}_i$ in terms of the new basic columns $\mathbf{a}_1, \ldots, \mathbf{a}_{p-1}, \mathbf{a}_q, \mathbf{a}_{p+1}, \ldots, \mathbf{a}_m$.

Finally, let us consider the last column in the tableau. We have

$$x'_p = x_p/y_{pq} = \epsilon_0$$

and

$$x'_j = x_j - y_{jq}x_p/y_{pq} = x_j - \epsilon_0 y_{jq} \quad \text{for } 1 \le j \le m \text{ and } j \ne p$$

Thus, the last column of the tableau gives exactly the values for the new basic solution $\mathbf{x}'$.

Therefore, the row transformations performed on the tableau for the basic solution $\mathbf{x}$ result in the tableau for the new basic solution $\mathbf{x}'$.

We should point out that in the above discussion, the basic columns for a basic solution are not ordered by their indices. Instead, they are ordered by the positions of the element 1 in the corresponding columns in the tableau. For example, the column $\mathbf{a}_q$ becomes the pth basic column because in the qth column of the new tableau, the pth element is 1 and all other elements are 0. Hence, the pth row in the new tableau corresponds to the coefficients for the column $\mathbf{a}_q$, i.e., y'_{pi} is the coefficient of $\mathbf{a}_q$ in the linear combination for $\mathbf{a}_i$ in terms of $\mathbf{a}_1, \ldots, \mathbf{a}_{p-1}, \mathbf{a}_q, \mathbf{a}_{p+1}, \ldots, \mathbf{a}_m$, and x'_p is the value of the qth element in the basic solution $\mathbf{x}'$.

In general case, suppose that we have a basic solution $\mathbf{x}$ in which the i_1th, i_2th, ..., i_mth elements $x_{i_1}, x_{i_2}, \ldots, x_{i_m}$ are positive, and the tableau $\mathcal{T}$ for $\mathbf{x}$ such that (1) for each j, $1 \leq j \leq m$, the i_jth column of $\mathcal{T}$ has the jth element equal to 1 and all other elements equal to 0; (2) for each j, $1 \leq j \leq m$, the jth element in the last column of $\mathcal{T}$ is x_{i_j}; and (3) for each i, $1 \leq i \leq n$, the ith column of $\mathcal{T}$ is $(y_{1i}, y_{2i}, \ldots, y_{mi})^T$ if the ith column $\mathbf{a}_i$ of the matrix $\mathbf{A}$ is represented by the linear combination of the columns $\mathbf{a}_{i_1}$, $\mathbf{a}_{i_2}, \ldots, \mathbf{a}_{i_m}$ as

$$\mathbf{a}_i = y_{1i}\mathbf{a}_{i_1} + y_{2i}\mathbf{a}_{i_2} + \cdots + y_{mi}\mathbf{a}_{i_m}$$

In order to replace the column $\mathbf{a}_{i_p}$ in the basic solution $\mathbf{x}$ by a new column $\mathbf{a}_q$ to obtain a new basic solution $\mathbf{x}'$, we first require that the element y_{pq} in the qth column of the tableau $\mathcal{T}$ be positive, and that x_p/y_{pq} be the minimum over all x_j/y_{jq} with $y_{jq} > 0$. With these conditions satisfied, perform the following row transformation on the tableau $\mathcal{T}$: (1) divide the pth row by y_{pq}; and (2) for each j, $1 \leq j \leq m$ and $j \neq p$, subtract y_{jq} times the pth row from the jth row. The resulting tableau by these row transformations is exactly the tableau for the new basic solution $\mathbf{x}'$ obtained by adding the qth column and deleting the i_pth column from the basic solution $\mathbf{x}$.

Example 4.2.1 Consider the following instance $\alpha = (\mathbf{b, c, A})$ of the Linear Programming problem

$$\begin{aligned} \text{minimize} \quad & x_6 \\ \text{subject to} \quad & 3x_1 + 5x_2 + x_3 = 24 \\ & 4x_1 + 2x_2 + x_4 = 16 \\ & x_1 + x_2 - x_5 + x_6 = 3 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{aligned} \tag{4.24}$$

Let the six column vectors of the matrix $\mathbf{A}$ be $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{a}_3$, $\mathbf{a}_4$, $\mathbf{a}_5$, $\mathbf{a}_6$. The vector $\mathbf{x}= (0, 0, 24, 16, 0, 3)$ is obviously a basic solution to the instance α with the basic columns $\mathbf{a}_3$, $\mathbf{a}_4$, and $\mathbf{a}_6$. The other columns of $\mathbf{A}$ can be represented by linear combinations of the columns $\mathbf{a}_3$, $\mathbf{a}_4$, and $\mathbf{a}_6$ as follows.

$$\begin{aligned} \mathbf{a}_1 &= 3\mathbf{a}_3 + 4\mathbf{a}_4 + \mathbf{a}_6 \\ \mathbf{a}_2 &= 5\mathbf{a}_3 + 2\mathbf{a}_4 + \mathbf{a}_6 \\ \mathbf{a}_5 &= -\mathbf{a}_6 \end{aligned}$$

Thus, the tableau for the basic solution $\mathbf{x}$ is

$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$	
3	5	1	0	0	0	24
4	2	0	1	0	0	16
1	1	0	0	−1	1	3

It should not be surprising that the tableau for the basic solution $\mathbf{x}$ consists of the columns of the matrix $\mathbf{A}$ plus the vecter $\mathbf{b}$. This is because that the three columns $\mathbf{a}_3$, $\mathbf{a}_4$, and $\mathbf{a}_6$ are three linearly independent unit vectors in the 3-dimensional Euclidean space $\mathcal{E}^3$.

Now suppose that we want to construct a new basic solution by replacing a column for the basic solution $\mathbf{x}$ by the second column $\mathbf{a}_2$ of the matrix $\mathbf{A}$. All elements in the second column of the tableau are positive. Thus, we only need to check the ratios. We have

$$x_1/y_{12} = 24/5 = 4.8 \qquad x_2/y_{22} = 16/2 = 8 \qquad x_3/y_{32} = 3/1 = 3$$

Thus, we will replace the column $\mathbf{a}_6$ by the column $\mathbf{a}_2$ (note that the 3rd row of the tableau corresponds to the 3rd basic column for $\mathbf{x}$, which is $\mathbf{a}_6$). Dividing the third row of the tableau by y_{32} does not change the tableau since $y_{32} = 1$. Then we subtract from the second row by 2 times the third row, and subtract from the first row by 5 times the third row. We obtain the final tableau

$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$	
−2	0	1	0	5	−5	9
2	0	0	1	2	−2	10
1	1	0	0	−1	1	3

The new basic solution $\mathbf{x}'$ corresponds to the columns $\mathbf{a}_3$, $\mathbf{a}_4$, and $\mathbf{a}_2$ (again note that though $\mathbf{a}_2$ has the smallest index, it is the 3rd basic column for $\mathbf{x}'$). The value of $\mathbf{x}'$ can be read directly from the last column of the tableau, which is $\mathbf{x}' = (0, 3, 9, 10, 0, 0)$. The coefficients of the linear combinations of the columns $\mathbf{a}_1$, $\mathbf{a}_5$, and $\mathbf{a}_6$ in terms of the columns $\mathbf{a}_2$, $\mathbf{a}_3$, and $\mathbf{a}_4$ can also read directly from the tableau:

$$\mathbf{a}_1 = -2\mathbf{a}_3 + 2\mathbf{a}_4 + \mathbf{a}_2 = \mathbf{a}_2 - 2\mathbf{a}_3 + 2\mathbf{a}_4$$
$$\mathbf{a}_5 = 5\mathbf{a}_3 + 2\mathbf{a}_4 - \mathbf{a}_2 = -\mathbf{a}_2 + 5\mathbf{a}_3 + 2\mathbf{a}_4$$
$$\mathbf{a}_6 = -5\mathbf{a}_3 - 2\mathbf{a}_4 + \mathbf{a}_2 = \mathbf{a}_2 - 5\mathbf{a}_3 - 2\mathbf{a}_4$$

All these can be verified easily in the original instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$.

How to move to a better neighbor basic solution

We have described how the basic solution $\mathbf{x} = (x_1, \ldots, x_m, 0, \ldots, 0)$ for the instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$ of the LINEAR PROGRAMMING problem can be converted to a neighbor basic solution

$$\begin{aligned}\mathbf{x}' \;=\; & (x_1 - \epsilon_0 y_{1q}, \ldots, x_{p-1} - \epsilon_0 y_{p-1,q}, 0, x_{p+1} - \epsilon_0 y_{p+1,q}, \ldots, \\ & \ldots, x_m - \epsilon_0 y_{mq}, 0, \ldots, 0, \epsilon_0, 0, \ldots, 0)\end{aligned}$$

by replacing the column $\mathbf{a}_p$ by the column $\mathbf{a}_q$, where $y_{pq} > 0$ and $\epsilon_0 = x_p/y_{pq}$ is the minimum over all x_i/y_{iq} with $y_{iq} > 0$. Since we want to minimize the value of the objective function $\mathbf{c}^T\mathbf{x}$, we would like to have $\mathbf{x}'$ to give a smaller objective function value. Consider the objective function values on these two basic solutions:

$$\mathbf{c}^T\mathbf{x} = c_1x_1 + c_2x_2 + \cdots + c_mx_m$$

and

$$\begin{aligned}\mathbf{c}^T\mathbf{x}' \;=\; & c_1(x_1 - \epsilon_0 y_{1q}) + \cdots + c_{p-1}(x_{p-1} - \epsilon_0 y_{p-1,q}) + \\ & +c_{p+1}(x_{p+1} - \epsilon_0 y_{p+1,q}) + \cdots + c_m(x_m - \epsilon_0 y_{mq}) + c_q\epsilon_0 \\ =\; & (\sum_{j=1}^{m} c_jx_j) - c_px_p + \epsilon_0(c_q - \sum_{j=1}^{m} c_jy_{jq}) + \epsilon_0 c_p y_{pq}\end{aligned}$$

Since $\epsilon_0 = x_p/y_{pq}$, we have $\epsilon_0 c_p y_{pq} = c_p x_p$. Thus, the last equality gives

$$\mathbf{c}^T\mathbf{x}' = \mathbf{c}^T\mathbf{x} + \epsilon_0(c_q - \sum_{j=1}^{m} c_jy_{jq})$$

Thus, the basic solution $\mathbf{x}'$ gives a better (i.e., smaller) objective function value $\mathbf{c}^T\mathbf{x}'$ than $\mathbf{c}^T\mathbf{x}$ if and only if $\epsilon_0(c_q - \sum_{j=1}^{m} c_jy_{jq}) < 0$. This thus gives us a guideline for choosing a column to construct a better neighbor

$\mathbf{a}_1$	$\cdots$	$\mathbf{a}_p$	$\cdots$	$\mathbf{a}_m$	$\mathbf{a}_{m+1}$	$\cdots$	$\mathbf{a}_q$	$\cdots$	$\mathbf{a}_n$	
1	$\cdots$	0	$\cdots$	0	$y_{1,m+1}$	$\cdots$	y_{1q}	$\cdots$	y_{1n}	x_1
.		.		.	.		.		.	.
0	$\cdots$	1	$\cdots$	0	$y_{p,m+1}$	$\cdots$	y_{pq}	$\cdots$	y_{pn}	x_p
.		.		.	.		.		.	.
0	$\cdots$	0	$\cdots$	1	$y_{m,m+1}$	$\cdots$	y_{mq}	$\cdots$	y_{mn}	x_m
0	$\cdots$	0	$\cdots$	0	$r_m + 1$	$\cdots$	r_q	$\cdots$	r_n	z_0

Figure 4.1: The general tableau format for the basic solution $\mathbf{x}$

basic solution. The constant $c_q - \sum_{j=1}^{m} c_j y_{jq}$ plays such a central role in the development of the simplex method, it is convenient to introduce somewhat abbreviated notation for it. Denote by r_q the constant $c_q - \sum_{j=1}^{m} c_j y_{jq}$ for $1 \le q \le n$, and call them the *reduced cost coefficients*. The above discussion gives us the following lemma.

Lemma 4.2.1 *Let $\mathbf{x}$ and $\mathbf{x}'$ be the basic solutions as given above. The basic solution $\mathbf{x}'$ gives a better (i.e., smaller) objective function value $\mathbf{c}^T\mathbf{x}'$ than $\mathbf{c}^T\mathbf{x}$ if and only if the reduced cost coefficient*

$$r_q = c_q - \sum_{j=1}^{m} c_j y_{jq}$$

is less than 0.

The nice thing is that the reduced cost coefficients r_i as well as the objective function value $\mathbf{c}^T\mathbf{x}$ can also be made a row in the tableau for the basic solution $\mathbf{x}$ and calculated by formal row transformations of the tableau. For this, we create a new row, the $(m+1)$st row, in the tableau so that the element corresponding to the vector $\mathbf{a}_i$ in this row is r_i (note that by the formula if $1 \le i \le m$ then $r_i = 0$), and the element in the last column of this row is the value $z_0 = -\mathbf{c}^T\mathbf{x}$. The new tableau format is given in Figure 4.1.

Now suppose that we replace the basic column $\mathbf{a}_p$ for the basic solution $\mathbf{x}$ by the column $\mathbf{a}_q$ to construct the basic solution $\mathbf{x}'$. Then in addition to the row transformations described before to obtain the coefficients y'_{ji} and $\mathbf{x}'$, we also (after dividing the pth two by y_{pq}) subtract r_q times the pth row from the $(m+1)$st row. We verify that this row transformation converts the $(m+1)$st row to give exactly the reduced cost coefficients and the objective function value for the new basic solution $\mathbf{x}'$.

By the above described procedure, the new value r'_i of the ith element in the $(m+1)$st row in the tableau is

$$\begin{aligned} r'_i &= r_i - r_q y_{pi}/y_{pq} \\ &= (c_i - \sum_{j=1}^{m} c_j y_{ji}) - (c_q - \sum_{j=1}^{m} c_j y_{jq}) y_{pi}/y_{pq} \\ &= c_i - \sum_{j=1}^{m} c_j y_{ji} - c_q y_{pi}/y_{pq} + \sum_{j=1}^{m} c_j y_{jq} y_{pi}/y_{pq} \\ &= c_i - \sum_{j=1}^{m} c_j (y_{ji} - y_{jq} y_{pi}/y_{pq}) - c_q y_{pi}/y_{pq} \end{aligned}$$

By equalities (4.22) and (4.23), and note $y'_{pi} = y_{pi} - y_{pq} y_{pi}/y_{pq} = 0$, we get

$$\begin{aligned} r'_i &= c_i - \sum_{j=1}^{m} c_j y'_{ji} - c_q y'_{pi} \\ &= c_i - (c_1 y'_{1i} + \cdots + c_{p-1} y'_{p-1,i} + c_{p+1} y'_{p+1,i} + \cdots + c_m y'_{mi} + c_q y'_{pi}) \end{aligned}$$

Therefore, the value r'_i is exactly the reduced cost coefficient for the column $\mathbf{a}_i$ in the new basic solution $\mathbf{x}'$.

Consider the new value z'_0 in the last column of the $(m+1)$st row. By the procedure, the new value is equal to

$$\begin{aligned} z'_0 &= z_0 - r_q x_p/y_{pq} \\ &= -\mathbf{c}^T\mathbf{x} - (c_q - \sum_{j=1}^{m} c_j y_{jq}) x_p/y_{pq} \\ &= -\sum_{j=1}^{m} c_j x_j + \sum_{j=1}^{m} c_j y_{jq} x_p/y_{pq} - c_q x_p/y_{pq} \end{aligned}$$

Let $\epsilon_0 = x_p/y_{pq}$ and note that $x_p - \epsilon_0 y_{pq} = 0$, we get

$$\begin{aligned} z'_0 &= -\sum_{j=1}^{m} c_j x_j + \sum_{j=1}^{m} c_j \epsilon_0 y_{jq} - c_q \epsilon_0 \\ &= -(\sum_{j=1}^{m} c_j (x_j - \epsilon_0 y_{jq}) + c_q \epsilon_0) \\ &= -(c_1(x_1 - \epsilon_0 y_{1q}) + \cdots + c_{p-1}(x_{p-1} - \epsilon_0 y_{p-1,q}) + \\ &\qquad + c_{p+1}(x_{p+1} - \epsilon_0 y_{p+1,q}) + \cdots + c_m(x_m - \epsilon_0 y_{mq}) + c_q \epsilon_0) \end{aligned}$$

According to equality (4.16), z'_0 gives exactly the value $-\mathbf{c}^T\mathbf{x}'$

Therefore, after the row transformations of the tableau, the $(m+1)$st row of the tableau gives exactly the reduced cost coefficients and the objective function value for the new basic solution $\mathbf{x}'$.

Example 4.2.2. Let us reconsider the instance $\alpha = (\mathbf{b}, \mathbf{c}, \mathbf{A})$ of the Linear Programming problem given in Example 4.2.1.

$$\begin{array}{rc} \text{minimize} & x_6 \\ \text{subject to} & 3x_1 + 5x_2 + x_3 = 24 \\ & 4x_1 + 2x_2 + x_4 = 16 \\ & x_1 + x_2 - x_5 + x_6 = 3 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{array}$$

The extended tableau for the basic solution $\mathbf{x}= (0, 0, 24, 16, 0, 3)$, which has an objective function value 3, is as follows.

$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$	
3	5	1	0	0	0	24
4	2	0	1	0	0	16
1	1	0	0	−1	1	3
−1	−1	0	0	1	0	−3

If we replace the column $\mathbf{a}_6$ by the column $\mathbf{a}_2$ (note $r_2 < 0$), then after the row transformations, we obtain the tableau

$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$	
−2	0	1	0	5	−5	9
2	0	0	1	2	−2	10
1	1	0	0	−1	1	3
0	0	0	0	0	1	0

Thus, we obtained an improved basic solution $\mathbf{x}' = (0, 3, 9, 10, 0, 0)$ that has an objective function value 0.

We summarize the above discussion on tableau transformations into the algorithm given in Figure 4.2. Thus, suppose that $\mathcal{T}$ is the tableau for the basic solution $\mathbf{x}$ with $\mathcal{T}[m+1, q] < 0$ and let p be the index such that $\mathcal{T}[p, q] > 0$ and the ratio $\mathcal{T}[p, n+1]/\mathcal{T}[p, q]$ is the minimum over all ratios $\mathcal{T}[j, n+1]/\mathcal{T}[j, q]$ with $\mathcal{T}[j, q] > 0$, then according to Lemma 4.2.1, the

Algorithm. TableauMove$(\mathcal{T}, p, q)$
INPUT: an $(m+1) \times (n+1)$ tableau $\mathcal{T}$, $1 \le p \le m$, $1 \le q \le n$, $\mathcal{T}[p, q] \ne 0$

1. $y_{pq} = \mathcal{T}[p, q]$;
2. **for** $i = 1$ **to** $n + 1$ **do** $\mathcal{T}[p, i] = \mathcal{T}[p, i]/y_{pq}$;
3. **for** $(1 \le j \le m + 1)$ and $(j \ne p)$ **do**
   $y_{jq} = \mathcal{T}[j, q]$;
   for $i = 1$ **to** $n + 1$ **do** $\mathcal{T}[j, i] = \mathcal{T}[j, i] - \mathcal{T}[p, i] * y_{jq}$;

Figure 4.2: Tableau transformation

algorithm **TableauMove**$(\mathcal{T}, p, q)$ will result in the tableau for a neighbor basic solution $\mathbf{x}'$, by replacing the pth basic column for $\mathbf{x}$ by the qth column, such that $\mathbf{c}^T\mathbf{x}' < \mathbf{c}^T\mathbf{x}$.

When an optimal solution is achieved

Suppose that we have the basic solution $\mathbf{x} = (x_1, \ldots, x_m, 0, \ldots, 0)$ and the tableau $\mathcal{T}$ in Figure 4.1 for $\mathbf{x}$. By Lemma 4.2.1, if there is a column q in $\mathcal{T}$ such that $r_q < 0$ and there is a positive element y_{pq} in the qth column, then we can perform the row transformations to obtain a better basic solution $\mathbf{x}'$ with $\mathbf{c}^T\mathbf{x}'$, thus achieving an improvement. What if no such a column q exists in the tableau?

If no such a column exists in the tableau, then either we have $r_q \ge 0$ for all q, $1 \le q \le n$, or we have $r_q < 0$ for some q but $y_{pq} \le 0$ for all p, $1 \le p \le m$. We consider these two cases separately below.

CASE 1. all values $r_q \ge 0$.

We prove that in this case, the basic solution $\mathbf{x}$ is an optimal solution.

Let $\mathbf{x}' = (x'_1, x'_2, \ldots, x'_n)$ be any solution to the instance $\alpha = (\mathbf{b}, \mathbf{c}, \mathbf{A})$. Thus, we have $x'_i \ge 0$, $1 \le i \le n$, and

$$x'_1\mathbf{a}_1 + x'_2\mathbf{a}_2 + \cdots + x'_n\mathbf{a}_n = \mathbf{b} \tag{4.25}$$

Since $\mathbf{x} = (x_1, \ldots, x_m, 0, \ldots, 0)$ is a basic solution, each column $\mathbf{a}_i$ of $\mathbf{A}$ can be represented by a linear combination of $\mathbf{a}_1$, $\mathbf{a}_2$, $\ldots$, $\mathbf{a}_m$:

$$\mathbf{a}_i = \sum_{j=1}^{m} y_{ji}\mathbf{a}_j, \qquad m + 1 \le i \le n \tag{4.26}$$

Replace each $\mathbf{a}_i$ in (4.25), $m+1 \leq i \leq n$, by the equality (4.26), we obtain

$$\begin{aligned} x'_1\mathbf{a}_1 + \cdots + x'_m\mathbf{a}_m + x'_{m+1}\sum_{j=1}^{m} y_{j,m+1}\mathbf{a}_j + \\ + x'_{m+2}\sum_{j=1}^{m} y_{j,m+2}\mathbf{a}_j + \cdots + x'_n\sum_{j=1}^{m} y_{jn}\mathbf{a}_j &= \mathbf{b} \end{aligned}$$

Regrouping the terms gives

$$\begin{aligned} (x'_1 + \sum_{i=m+1}^{n} x'_i y_{1i})\mathbf{a}_1 + (x'_2 + \sum_{i=m+1}^{n} x'_i y_{2i})\mathbf{a}_2 + \\ + \cdots + (x'_m + \sum_{i=m+1}^{n} x'_i y_{mi})\mathbf{a}_m &= \mathbf{b} \end{aligned} \tag{4.27}$$

Since $\mathbf{x} = (x_1, \ldots, x_m, 0, \ldots, 0)$ is a basic solution, we also have

$$x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \cdots + x_m\mathbf{a}_m = \mathbf{b} \tag{4.28}$$

Compare equalities (4.27) and (4.28). Since the columns $\mathbf{a}_1, \ldots, \mathbf{a}_m$ are linearly independent, the vector $\mathbf{b}$ has a unique representation in terms of the linear combination of $\mathbf{a}_1, \ldots, \mathbf{a}_m$. Thus, we must have

$$\begin{aligned} x_1 &= x'_1 + \sum_{i=m+1}^{n} x'_i y_{1i} \\ x_2 &= x'_2 + \sum_{i=m+1}^{n} x'_i y_{2i} \\ &\cdots\cdots \\ x_m &= x'_m + \sum_{i=m+1}^{n} x'_i y_{mi} \end{aligned}$$

Bringing these values for $x_1, x_2, \ldots, x_m$ to the inner product $\mathbf{c}^T\mathbf{x}$, we get

$$\begin{aligned} \mathbf{c}^T\mathbf{x} &= \sum_{j=1}^{n} c_j x_j = \sum_{j=1}^{m} c_j x_j \\ &= \sum_{j=1}^{m} c_j (x'_j + \sum_{i=m+1}^{n} x'_i y_{ji}) \\ &= \sum_{j=1}^{m} c_j x'_j + \sum_{j=1}^{m} \sum_{i=m+1}^{n} c_j x'_i y_{ji} \end{aligned}$$

$$
\begin{aligned}
&= \sum_{j=1}^{m} c_j x'_j + \sum_{i=m+1}^{n} x'_i (\sum_{j=1}^{m} c_j y_{ji}) \\
&= \sum_{j=1}^{n} c_j x'_j - \sum_{i=m+1}^{n} c_i x'_i + \sum_{i=m+1}^{n} x'_i (\sum_{j=1}^{m} c_j y_{ji}) \\
&= \mathbf{c}^T \mathbf{x}' - \sum_{i=m+1}^{n} x'_i (c_i - \sum_{j=1}^{m} c_j y_{ji}) \\
&= \mathbf{c}^T \mathbf{x}' - \sum_{i=m+1}^{n} x'_i r_i
\end{aligned}
$$

Since $\mathbf{x}'$ is a solution to the instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$, $x'_i \geq 0$ for $m+1 \leq i \leq n$, and by our assumption, $r_q \geq 0$ for $m+1 \leq q \leq n$, we get

$$\mathbf{c}^T \mathbf{x} = \mathbf{c}^T \mathbf{x}' - \sum_{i=m+1}^{n} x'_i r_i \leq \mathbf{c}^T \mathbf{x}';$$

Since $\mathbf{x}'$ is an arbitrary solution to the instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$, we conclude that $\mathbf{x}$ is an optimal solution to $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$. This conclusion is summarized in the following lemma.

Lemma 4.2.2 *Let* $\mathbf{x}$ *be a basic solution to the instance* $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$ *with the tableau given in Figure 4.1. If all reduced cost coefficients* $r_q \geq 0$, $1 \leq q \leq n$, *then* $\mathbf{x}$ *is an optimal solution.*

Example 4.2.3. Recall that in Example 4.2.2, we obtained the basic solution $\mathbf{x}' = (0, 3, 9, 10, 0, 0)$, which has the objective function value 0, such that all reduced cost coefficients are larger than or equal to 0 (see the last tableau in Example 4.2.2). By Lemma 4.2.2, the solution $\mathbf{x}'$ is an optimal solution to the given instance.

CASE 2. There is a q such that $r_q < 0$ but no element in the qth column of the tableau is positive.

In this case, consider the equalities

$$y_{1q}\mathbf{a}_1 + y_{2q}\mathbf{a}_2 + \cdots + y_{mq}\mathbf{a}_m - \mathbf{a}_q = 0$$

and

$$x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \cdots + x_m\mathbf{a}_m = \mathbf{b}$$

Subtract from the second equality by ϵ times the first equality, where ϵ is any positive number, we get

$$(x_1 - \epsilon y_{1q})\mathbf{a}_1 + (x_2 - \epsilon y_{2q})\mathbf{a}_2 + \cdots + (x_m - \epsilon y_{mq})\mathbf{a}_m + \epsilon \mathbf{a}_q = \mathbf{b}$$

Since $x_i > 0$ for $1 \leq i \leq m$, and $y_{jq} \leq 0$ for all $1 \leq j \leq m$, we have $x_j - \epsilon y_{jq} > 0$ for all $1 \leq j \leq m$. Thus,

$$\mathbf{x}_\epsilon = (x_1 - \epsilon y_{1q}, \ldots, x_m - \epsilon y_{mq}, 0, \ldots, 0, \epsilon, 0, \ldots, 0)$$

where the element ϵ is in the qth position, is a solution to $\alpha = (\mathbf{b}, \mathbf{c}, \mathbf{A})$ for all positive value ϵ.

Consider the objective function value $\mathbf{c}^T\mathbf{x}_\epsilon$ on the solution $\mathbf{x}_\epsilon$, we have

$$\begin{aligned} \mathbf{c}^T\mathbf{x}_\epsilon &= \sum_{j=1}^{m} c_j(x_j - \epsilon y_{jq}) + c_q\epsilon \\ &= \sum_{j=1}^{m} c_j x_j + \epsilon(c_q - \sum_{j=1}^{m} c_j y_{jq}) \\ &= \mathbf{c}^T\mathbf{x} + \epsilon r_q \end{aligned}$$

By our assumption, $r_q < 0$ and ϵ can be any positive number, the value $\mathbf{c}^T\mathbf{x}_\epsilon$ can be arbitrarily small, i.e., the instance $\alpha = (\mathbf{b}, \mathbf{c}, \mathbf{A})$ has no optimal solution. We summarize this in the following lemma.

Lemma 4.2.3 *Let* $\mathbf{x}$ *be a basic solution to the instance* $\alpha = (\mathbf{b}, \mathbf{c}, \mathbf{A})$ *with the tableau given in Figure 4.1. If there is a reduced cost coefficients* $r_q < 0$*, and all elements in the* q*th column of the tableau are less than or equal to 0, then the objective function can have arbitrarily small value and the instance* α *has no optimal solution.*

Degeneracy

It is possible that in the course of the simplex method described above, a degenerate basic solution occurs. Often they can be handled as a non-degenerate basic solution. However, it is possible that after a new column $\mathbf{a}_q$ is selected to replace a current basic column $\mathbf{a}_p$, the ratio x_p/y_{pq} is 0, implying that the basic column $\mathbf{a}_p$ is the one to go out. This means that the new variable x_q will come into the new basic solution at value 0, the objective function value will not decrease, and the new basic solution will also be degenerate. Conceivably, this process could continue for a series of steps and even worse, some degenerate basic solution may repeat in the series, leading to an endless process without being able to achieve an optimal solution. This situation is called *cycling*.

Degeneracy often occurs in large-scale real-world problems. However, cycling in such instances is very rare. Methods have been developed to avoid cyclings. In practice, however, such procedures are found to be unnecessary. When degenerate solutions are encountered, the simplex method generally does not enter cycling. However, anticycling procedures are simple, and many codes incorporate such a procedure for the sake of safety.

How to obtain the first basic solution

Lemmas 4.2.1, 4.2.2, and 4.2.3 completely describe how we can move from a basic solution to a better basic solution and when an optimal basic solution is achieved. To describe the simplex method completely, the only thing remaining is how the first basic solution can be obtained.

A basic solution is sometimes immediately available from an instance of the LINEAR PROGRAMMING problem. For example, suppose that the instance of the LINEAR PROGRAMMING problem is given in the form

$$\begin{array}{rc} \text{minimize} & c_1x_1 + c_2x_2 + \cdots c_nx_n \\ \text{subject to} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \le b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \le b_2 \\ & \cdots\cdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \le b_m \\ & x_1 \ge 0,\ x_2 \ge 0,\ \ldots,\ x_n \ge 0 \end{array}$$

with $b_i \ge 0$ for all i. Then in the elimination of the $\le$ signs we introduce m slack variables $y_1, \ldots, y_m$ and convert it into the standard form

$$\begin{array}{rc} \text{minimize} & c_1x_1 + c_2x_2 + \cdots c_nx_n \\ \text{subject to} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + y_1 = b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + y_2 = b_2 \\ & \cdots\cdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n + y_m = b_m \\ & x_1 \ge 0,\ x_2 \ge 0,\ \ldots,\ x_n \ge 0 \\ & y_1 \ge 0,\ y_2 \ge 0,\ \ldots,\ y_m \ge 0 \end{array}$$

Obviously, the $(n+m)$-dimensional vector $(0, \ldots, 0, b_1, b_2, \ldots, b_m)$ is a basic solution to this new instance, from which the simplex method can be initiated. In fact, this method can be applied to general instances for the LINEAR PROGRAMMING problem, as described below.

Given an instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$ of the LINEAR PROGRAMMING problem, by multiplying an equality by -1 when necessary, we can always assume that $\mathbf{b} \geq 0$. In order to find a solution to α, consider the auxiliary instance α' for the LINEAR PROGRAMMING problem

$$\begin{aligned} \text{minimize} \quad & y_1 + y_2 + \cdots + y_m \\ \text{subject to} \quad & \mathbf{Ax} + \mathbf{y} = \mathbf{b} \\ & \mathbf{x},\ \mathbf{y} \geq 0 \end{aligned} \tag{4.29}$$

where $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ is an m-dimensional vector of *artificial variables.* Note that the $(n+m)$-dimensional vector $\mathbf{w} = (0, 0, \ldots, 0, b_1, b_2, \ldots, b_m)$ is clearly a basic solution to the instance α' in (4.29). If there is a solution $(x_1, \ldots, x_n)$ to the instance α, then it is clear that the instance α' in (4.29) has an optimal solution $(x_1, \ldots, x_n, 0, \ldots, 0)$ with optimal objective function value 0. On the other hand, if the instance α has no solution, then the optimal objective function value for the instance α' is larger than 0 (note that the solution set $S_Q(\alpha')$ for the instance α' is always nonempty).

Now starting with the basic solution $\mathbf{w} = (0, 0, \ldots, 0, b_1, b_2, \ldots, b_m)$ for the instance α', we can apply the simplex method to find an optimal solution for α'. Note that the tableau for the basic solution $\mathbf{w}$ is also immediate — the first m rows of the tableau have the form $[\mathbf{A}, \mathbf{I}, \mathbf{b}]$, where $\mathbf{I}$ is the m-dimensional identity matrix, the reduced cost coefficient r_j for $1 \leq j \leq n$ is equal to $-v_j$, where v_j is the sum of the elements in the jth column in the tableau, and the last element in the $(m+1)$st row is equal to $-b_1 - \cdots - b_m$.

Suppose that the simplex method finds an optimal basic solution $\mathbf{w}_0 = (w_1, w_2, \ldots, w_{n+m})$ for the instance α' in (4.29). If $\mathbf{w}_0$ does not have objective function value 0, then the original instance α has no solution. If $\mathbf{w}_0$ has objective function value 0, then we must have $w_j = 0$ for all $n+1 \leq j \leq n+m$. In the second case, we let $\mathbf{x} = (w_1, w_2, \ldots, w_n)$. We claim that the vector $\mathbf{x}$ is a basic solution for the instance α. First of all, it is clear that $\mathbf{x} \geq 0$ and $\mathbf{Ax} = \mathbf{b}$. Moreover, suppose that w_{i_1}, w_{i_2}, $\ldots$, w_{i_k} are the positive elements in $\mathbf{x}$, then they are also positive elements in $\mathbf{w}_0$. Thus, the columns $\mathbf{a}_{i_1}$, $\mathbf{a}_{i_2}$, $\ldots$, $\mathbf{a}_{i_k}$ of the matrix $\mathbf{A}$ are basic columns for $\mathbf{w}_0$ thus are linearly independant. Thus, if we extend the k columns $\mathbf{a}_{i_1}$, $\mathbf{a}_{i_2}$, $\ldots$, $\mathbf{a}_{i_k}$ of the matrix $\mathbf{A}$ (arbitrarily) into m linearly independent columns of $\mathbf{A}$, then the solution $\mathbf{x}$ is a basic solution with these m linearly independent columns as its basic columns. In case $k = m$, $\mathbf{x}$ is a non-degenerate basic solution, and in case $k < m$, $\mathbf{x}$ is a degenerate basic solution.

To summarize, we use artificial variables to attack a general instance of the LINEAR PROGRAMMING problem. Our approach is a *two-phase method.* This method consists of the first phase in which artificial variables are introduced to construct an auxiliary instance α' with an obvious starting basic solution, and an optimal solution $\mathbf{w}_0$ for α' is constructed using the simplex method; and the second phase in which, a basic solution $\mathbf{x}$ for the original instance α is constructed from the vector $\mathbf{w}_0$ obtained in the first phase, and an optimal solution for α is constructed using the simplex method.

Putting all these together

We summarize the above procedures to give the complete simplex method. See Figure 4.3. For a given column q in a tableau $\mathcal{T}$ of $m+1$ rows and $n+1$ columns, an element $\mathcal{T}[p,q]$ of $\mathcal{T}$ is said to have the *minimum ratio* in the qth column if $\mathcal{T}[p,q] > 0$ and the ratio $\mathcal{T}[p,n+1]/\mathcal{T}[p,q]$ is the minimum over all ratios $\mathcal{T}[j,n+1]/\mathcal{T}[j,q]$ with $1 \le j \le n$ and $\mathcal{T}[j,q] > 0$.

Note that in Phase I, step 3, we do not have to check whether the qth column of the tableau $\mathcal{T}_1$ has an element with minimum ratio — it must have one. This is because if it does not have one, then by Lemma 4.2.3, the objective function of the instance α' would have had arbitrary small value. On the other hand, 0 is obviously a lower bound for the objective function values for the instance α'.

The correctness of the algorithm **Simplex Method** is by Lemmas 4.2.1, 4.2.2, and 4.2.3. Under the Nondegeneracy Assumption, each procedure call **TableauMove**$(\mathcal{T}, p, q)$ results in a basic solution with a smaller objective function value. Since the number of basic solutions is finite, and since no basic solution repeats because of the strictly decreasing objective function values, the algorithm **Simplex Method** will eventually stop, either with an optimal solution to the instance α, or with a claim that there is no optimal solution to the instance α. In the case of degeneracy, incorporated with an anticycling procedure, we can also guarantee that the algorithm **Simplex Method** terminates in a finite number of steps with a correct conclusion.

Moving from one basic solution to a neighbor basic solution, using the tableau format, can be done in time $O(nm)$. Thus, the time complexity of the algorithm **Simplex Method** depends on how many basic solution moves are needed to achieve an optimal basic solution. Extensive experience with the simplex method applied to problems from various fields, and having various of the number n of variables and the number m of constraints, have indicated that the method can be expected to converge to an optimal solution in $O(m)$ basic solution moves. Therefore, practically, the algorithm **Simplex Method** is pretty fast. However and unfortunately, there are instances for the problem for which the algorithm **Simplex Method** requires a large number of basic solution moves. These instances show that the algorithm **Simplex Method** is not a polynomial time algorithm.

Algorithm. Simplex Method
INPUT: an instance $\alpha = (\mathbf{b},\ \mathbf{c},\ \mathbf{A})$ with $\mathbf{b} \geq 0$

Phase I.
1. construct a new instance α': $\mathbf{Ax} + \mathbf{y} = \mathbf{b}$; $\mathbf{x}, \mathbf{y} \geq 0$
2. construct the basic solution $\mathbf{w}$ for α' and the tableau $\mathcal{T}_1[1..m+1, 1..m+n+1]$ for $\mathbf{w}$;
3. **while** $(\mathcal{T}_1[m+1, q] < 0$ for some $1 \leq q \leq n+m)$ **do**
   let $\mathcal{T}_1[p, q]$ have the minimum ratio in column q; call **TableauMove**$(\mathcal{T}_1, p, q)$;
4. **if** $(\mathcal{T}_1[m+1, m+n+1] \neq 0)$ **then** stop: instance α has no solution;

Phase II.

5. let $\mathcal{T}_2[1..m+1, 1..n+1]$ be $\mathcal{T}_1$ with the $(n+1)$st $\ldots$, $(n+m)$th columns deleted;
   \\ $\mathcal{T}_2$ is the tableau for a basic solution $\mathbf{x}$ for α.
   $\mathcal{T}_2[m+1, n+1] = -\mathbf{c}^T\mathbf{x}$;
6. **while** $(\mathcal{T}_2[m+1, q] < 0$ for some $1 \leq q \leq n)$ **do**
   if no element in the qth column of $\mathcal{T}_2$ is positive
   then stop: the instance α has no optimal solution
   else let $\mathcal{T}_2[p, q]$ have the minimum ratio in column q; call **TableauMove**$(\mathcal{T}_2, p, q)$;
7. stop: the tableau $\mathcal{T}_2$ gives an optimal solution $\mathbf{x}$ to α.

Figure 4.3: The Simplex Method algorithm

4.3 Duality

Associated with every instance $(\mathbf{b},\ \mathbf{c},\ \mathbf{A})$ of the LINEAR PROGRAMMING problem is a corresponding *dual instance*. Both instances are constructed from the vectors $\mathbf{b}$ and $\mathbf{c}$ and the matrix $\mathbf{A}$ but in such a way that if one of these instances is one of a maximization problem then the other is a minimization problem, and that the optimal objective function values of the instances, if finite, are equal. The variables of the dual instance are also intimately related to the calculation of the reduced cost coefficients in the simplex method. Thus, a study of duality sharpens our understanding of the simplex method and motivates certain alternative solution methods. Indeed, the simultaneous consideration of a problem form both the primal and dual viewpoints often provides significant computational advantage.

Dual instance

We first depart from our usual strategy of considering instance in the standard form, since the duality relationship is most symmetric for instances expressed solely in terms of inequalities.

Given an instance α of the LINEAR PROGRAMMING problem

$$\begin{array}{rc} & \underline{\text{Primal Instance } \alpha} \\ \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} \geq \mathbf{b},\ \ \mathbf{x} \geq \mathbf{0} \end{array} \tag{4.30}$$

where $\mathbf{A}$ is an $m \times n$ matrix, $\mathbf{c}$ is an n-dimensional vector, $\mathbf{b}$ is an m-dimensional vector, and $\mathbf{x}$ is an n-dimensional vector of variables, the corresponding dual instance α' is of the form

$$\begin{array}{rc} & \underline{\text{Dual Instance } \alpha'} \\ \text{maximize} & \mathbf{y}^T\mathbf{b} \\ \text{subject to} & \mathbf{y}^T\mathbf{A} \leq \mathbf{c},\ \ \mathbf{y} \geq \mathbf{0} \end{array} \tag{4.31}$$

where $\mathbf{y}$ is an m-dimensional vector of variables.

The pair (α, α') of instances is called the *synmmetric form* of duality. We explain below how the symmetric form of duality can be used to define the dual of any instance of the LINEAR PROGRAMMING problem. We first note that the role of primal and dual can be reversed. In fact, if the dual instance α' is transformed, by multiplying the objective function and the constraints by -1 so that it has the format of the primal instance in (4.30) (but is still expressed in terms of $\mathbf{y}$), then its corresponding dual will be equivalent to the original instance α in the format given in (4.30).

Consider an instance of the LINEAR PROGRAMMING in the standard form

$$\begin{array}{rc} & \underline{\text{Primal Instance } \alpha} \\ \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0} \end{array} \tag{4.32}$$

Write it in the equivalent form

$$\begin{array}{rc} \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} \geq \mathbf{b},\ -\mathbf{A}\mathbf{x} \geq -\mathbf{b},\ \mathbf{x} \geq \mathbf{0} \end{array}$$

which is now in the format of the primal instance in (4.30), with coefficient matrix

$$\begin{bmatrix} \mathbf{A} \\ \text{-}\mathbf{A} \end{bmatrix}$$

Now we will let $\mathbf{z}$ be the $(2m)$-dimensional vector of variables for the dual instance, and write $\mathbf{z}$ as $\mathbf{z}^T = (\mathbf{u},\ \mathbf{v})^T$ where both $\mathbf{u}$ and $\mathbf{v}$ are m-dimensional vectors of variables, the corresponding dual instance has the format

$$\begin{array}{rc} \text{maximize} & \mathbf{u}^T\mathbf{b} - \mathbf{v}^T\mathbf{b} \\ \text{subject to} & \mathbf{u}^T\mathbf{A} - \mathbf{v}^T\mathbf{A} \le \mathbf{c}^T,\ \mathbf{u},\ \mathbf{v} \ge \mathbf{0} \end{array}$$

Letting $\mathbf{y} = \mathbf{u} - \mathbf{v}$ we simplify the representation of the dual problem into the following format

$$\begin{array}{rc} & \underline{\text{Dual Instance } \alpha'} \\ \text{maximize} & \mathbf{y}^T\mathbf{b} \\ \text{subject to} & \mathbf{y}^T\mathbf{A} \le \mathbf{c}^T \end{array} \tag{4.33}$$

The pair (α, α') in (4.32) and (4.33) gives the *asymmetric form* of the duality relation. In this form the dual vector $\mathbf{y}$ is not restricted to be nonegative.

Similar transformation can be worked out for any instance of the Linear Programming problem by first converting the primal instance into the format in (4.30), calculating the dual, and then simplifying the dual to account for a special structure.

The Duality Theorem

So far the relation between a primal instance α and its dual instance α' for the Linear Programming problem has been simply a formal definition. In the following, we reveal a deeper connection between a primal instance and its dual. This connection will enable us to solve the Linear Programming problem more efficiently than by simply applying the simplex method.

Lemma 4.3.1 *Let $\mathbf{x}$ be a solution to the primal instance α in* (4.32) *and let $\mathbf{y}$ be a solution to the dual instance in* (4.33)*. Then $\mathbf{c}^T\mathbf{x} \ge \mathbf{y}^T\mathbf{b}$.*

Proof. Since $\mathbf{x}$ is a solution to the instance α, we have $\mathbf{y}^T\mathbf{b} = \mathbf{y}^T\mathbf{A}\mathbf{x}$. Now since $\mathbf{y}$ is a solution to the dual instance α', $\mathbf{y}^T\mathbf{A} \le \mathbf{c}^T$. Note that $\mathbf{x} \ge \mathbf{0}$, thus, $\mathbf{y}^T\mathbf{A}\mathbf{x} \le \mathbf{c}^T\mathbf{x}$. This gives $\mathbf{c}^T\mathbf{x} \ge \mathbf{y}^T\mathbf{b}$. □

Note that the instance α in (4.32) looks for a minimum value $\mathbf{c}^T\mathbf{x}$ while the instance α' in (4.33) looks for a maximum value $\mathbf{y}^T\mathbf{b}$. Thus, Lemma 4.3.1 shows that a solution to one problem yields a finite bound on the objective function value for the other problem. In particular, this lemma can be used to test whether the primal instance or the dual instance has solution. We say that the primal instance α has an *unbounded* solution if for any negative value $-M$ there is a solution $\mathbf{x}$ to α such that $\mathbf{c}^T\mathbf{x} \leq -M$, and that the dual instance α' has an *unbounded* solution if for any positive value M there is a solution $\mathbf{y}$ to α' such that $\mathbf{y}^T\mathbf{b} \geq M$.

Theorem 4.3.2 *If the primal instance α in* (4.32) *has an unbounded solution then the dual instance α' in* (4.33) *has no solution, if the dual instance α' has an unbounded solution then the primal instance has no solution.*

PROOF. Suppose that α has an unbounded solution but α' has a solution $\mathbf{y}$. Fix $\mathbf{y}$. Given any negative number $-M$, by defintion, there is a solution $\mathbf{x}$ to α such that $\mathbf{c}^T\mathbf{x} \leq -M$. By Lemma 4.3.1, $\mathbf{c}^T\mathbf{x} \geq \mathbf{y}^T\mathbf{b}$. Thus, $\mathbf{y}^T\mathbf{b} \leq -M$. But this is impossible since $-M$ can be any negative number.

The second statement can be proved similarly. □

If $\mathbf{c}^T\mathbf{x} = \mathbf{y}^T\mathbf{b}$ for a solution $\mathbf{x}$ to α and a solution $\mathbf{y}$ to α', then by Lemma 4.3.1, $\mathbf{x}$ must be an optimal solution for the instance α and $\mathbf{y}$ must be an optimal solution for the instance α'. The following theorem indicates that, in fact, this is a necessary and sufficient condition for $\mathbf{x}$ to be an optimal solution for α and for $\mathbf{y}$ to be an optimal solution for α'.

Theorem 4.3.3 *Let $\mathbf{x}$ be a solution to the primal instance α in* (4.32). *Then $\mathbf{x}$ is an optimal solution to α if and only if $\mathbf{c}^T\mathbf{x} = \mathbf{y}^T\mathbf{b}$ for some solution $\mathbf{y}$ to the dual instance α' in* (4.33).

PROOF. Suppose that $\mathbf{c}^T\mathbf{x} = \mathbf{y}^T\mathbf{b}$. By Lemma 4.3.1, for every solution $\mathbf{x}'$ to the primal instance α, we have $\mathbf{c}^T\mathbf{x}' \geq \mathbf{y}^T\mathbf{b} = \mathbf{c}^T\mathbf{x}$. Thus, $\mathbf{x}$ is an optimal solution to the primal instance α.

Conversely, suppose that $\mathbf{x}$ is an optimal solution to the primal instance α. We show that there is a solution $\mathbf{y}$ to the dual instance α' such that $\mathbf{c}^T\mathbf{x} = \mathbf{y}^T\mathbf{b}$.

Since all optimal solutions to the primal instance α give the same objective function value, we can assume, without loss of generality, that the solution $\mathbf{x}$ is a basic solution to the instance α. Furthermore, we assume for convenience that $\mathbf{x} = (x_1, x_2, \ldots, x_m, 0, \ldots, 0)^T$, and that the first m columns $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m$ of the matrix $\mathbf{A}$ are the basic columns for $\mathbf{x}$. Then the tableau for the basic solution **x** is of the following form.

$\mathbf{a}_1$	$\mathbf{a}_2$	$\cdots$	$\mathbf{a}_m$	$\mathbf{a}_{m+1}$	$\mathbf{a}_{m+2}$	$\cdots$	$\mathbf{a}_n$	
1	0	$\cdots$	0	$y_{1,m+1}$	$y_{1,m+2}$	$\cdots$	y_{1n}	x_1
0	1	$\cdots$	0	$y_{2,m+1}$	$y_{2,m+2}$	$\cdots$	y_{2n}	x_2
·	·		·	·	·		·	·
0	0	$\cdots$	1	$y_{m,m+1}$	$y_{m,m+2}$	$\cdots$	y_{mn}	x_m
0	0	$\cdots$	0	r_{m+1}	r_{m+2}	$\cdots$	r_n	z_0

where for each j, $m+1 \le j \le n$, we have

$$\mathbf{a}_j = \sum_{i=1}^{m} y_{ij}\mathbf{a}_i \quad \text{and} \quad r_j = c_j - \sum_{i=1}^{m} c_i y_{ij}$$

Since **x** is an optimal solution, by Lemmas 4.2.1 and 4.2.2, we must have $r_j \ge 0$ for all $m+1 \le j \le n$. That is,

$$c_j \ge \sum_{i=1}^{m} c_i y_{ij} \quad \text{for } m+1 \le j \le n \tag{4.34}$$

Let $\mathbf{B} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m]$ be the $m \times m$ nonsingular submatrix of the matrix $\mathbf{A}$ and let $\mathbf{B}^{-1}$ be the inverse matrix of $\mathbf{B}$. Note that for each i, $1 \le i \le m$, $\mathbf{B}^{-1}\mathbf{a}_i$ is the ith unit vector of dimension m (i.e., the m-dimensional vector whose ith element is 1 while all other elements are 0):

$$\mathbf{B}^{-1}\mathbf{a}_i = (0, \ldots, 0, 1, 0, \ldots, 0)^T, \quad i = 1, \ldots, m \tag{4.35}$$

Therefore, for $j = m+1, \ldots, n$, we have

$$\mathbf{B}^{-1}\mathbf{a}_j = \mathbf{B}^{-1}\sum_{i=1}^{m} y_{ij}\mathbf{a}_i = \sum_{i=1}^{m} y_{ij}\mathbf{B}^{-1}\mathbf{a}_i = (y_{1j}, y_{2j}, \ldots, y_{mj})^T \tag{4.36}$$

The last equality is because of the equality (4.35).

Now we let $\mathbf{y}^T = (c_1, c_2, \ldots, c_m)\mathbf{B}^{-1}$. Then $\mathbf{y}$ is an m-dimensional vector. We show that $\mathbf{y}$ is a solution to the dual instance α' and satisfies the condition $\mathbf{c}^T\mathbf{x} = \mathbf{y}^T\mathbf{b}$.

First consider $\mathbf{y}^T\mathbf{A}$. We have

$$\begin{aligned}
\mathbf{y}^T\mathbf{A} &= \mathbf{y}^T[\mathbf{B}, \mathbf{a}_{m+1}, \ldots, \mathbf{a}_n] \\
&= (c_1, \ldots, c_m)\mathbf{B}^{-1}[\mathbf{B}, \mathbf{a}_{m+1}, \ldots, \mathbf{a}_n] \\
&= (c_1, \ldots, c_m)[\mathbf{I}, \mathbf{B}^{-1}\mathbf{a}_{m+1}, \ldots, \mathbf{B}^{-1}\mathbf{a}_n] \\
&= (c_1, \ldots, c_m, c'_{m+1}, \ldots, c'_n)
\end{aligned}$$

where (note the equality (4.36))

$$c'_j = (c_1, \ldots, c_m)\mathbf{B}^{-1}\mathbf{a}_j = (c_1, \ldots, c_m)(y_{1j}, \ldots, y_{mj})^T = \sum_{i=1}^{m} c_i y_{ij}$$

for $j = m+1, \ldots, n$. By the inequality (4.34), we have $c'_j \leq c_j$ for $j = m+1, \ldots, n$. This thus proves $\mathbf{y}^T\mathbf{A} \leq \mathbf{c}^T$. That is, $\mathbf{y}$ is a solution to the dual instance α' in (4.33).

Finally, we have

$$\mathbf{y}^T\mathbf{b} = (c_1, \ldots, c_m)\mathbf{B}^{-1}\mathbf{b} = (c_1, \ldots, c_m)(x_1, \ldots, x_m)^T = \sum_{i=1}^{m} c_i x_i = \mathbf{c}^T\mathbf{x}$$

Therefore, $\mathbf{y}$ is a solution to the dual instance α' that satisfies $\mathbf{c}^T\mathbf{x} = \mathbf{y}^T\mathbf{b}$. This completes the proof of the theorem. □

The dual simplex method

Suppose that we have an instance $\alpha = (\mathbf{b}, \mathbf{c}, \mathbf{A})$ (in the standard form) of the LINEAR PROGRAMMING problem. Often available is a vector $\mathbf{x}$ such that $\mathbf{x}$ satisfies $\mathbf{A}\mathbf{x} = \mathbf{b}$ but not $\mathbf{x} \geq \mathbf{0}$. Moveover, $\mathbf{x}$ "optimizes" the objective function value $\mathbf{c}^T\mathbf{x}$ in the sense that no reduced cost efficient is negative (cf. Lemma 4.2.2). Such a situation may arise, for example, if a solution to a certain instance $\beta = (\mathbf{b}', \mathbf{c}, \mathbf{A})$ of the LINEAR PROGRAMMING problem is calculated and then the instance α is constructed such that α differs from β only by the vector $\mathbf{b}$. In such situations a basic solution to the dual instance α' of the instance α is available and hence, based on Theorem 4.3.3, it may be desirable to approach the optimal solution for the instance α in such a way as to optimize the dual instance α'.

Rather than constructing a tableau for the dual instance α', it is more efficient to work on the dual instance from the tableau for the primal instance α. The complete technique based on this idea is the *dual simplex method.* In terms of the primal instance α, it operates by maintaining the optimality condition of the reduced cost coefficients while working toward a solution $\mathbf{x}$ to α that satisfies $\mathbf{x} \geq \mathbf{0}$. In terms of the dual instance α', however, it maintains a solution to α' while working toward optimality.

Formally, let the primal instance α be of the form

	Primal Instance α
minimize	$\mathbf{c}^T\mathbf{x}$
subject to	$\mathbf{A}\mathbf{x} = \mathbf{b}, \ \mathbf{x} \geq \mathbf{0}$

The dual instance α' of the instance α is of the form

$$\begin{array}{rc} & \underline{\text{Dual Instance } \alpha'} \\ \text{maximize} & \mathbf{y}^T\mathbf{b} \\ \text{subject to} & \mathbf{y}^T\mathbf{A} \leq \mathbf{c}^T \end{array}$$

Suppose that m linearly independent columns of the matrix $\mathbf{A}$ have been identified such that with these m columns as "basic columns", no reduced cost coefficient is negative. Again for convenience of our discussion, suppose these m columns are the first m columns of $\mathbf{A}$ and let $\mathbf{B}$ be the $m \times m$ nonsingular submatrix consisting of these m columns. Therefore, the corresponding tableau should have the form (following our convention, $\mathbf{a}_1, \ldots, \mathbf{a}_n$ denote the columns of the matrix $\mathbf{A}$).

$\mathbf{a}_1$	$\mathbf{a}_2$	$\cdots$	$\mathbf{a}_m$	$\mathbf{a}_{m+1}$	$\mathbf{a}_{m+2}$	$\cdots$	$\mathbf{a}_n$	
1	0	$\cdots$	0	$y_{1,m+1}$	$y_{1,m+2}$	$\cdots$	y_{1n}	x_1
0	1	$\cdots$	0	$y_{2,m+1}$	$y_{2,m+2}$	$\cdots$	y_{2n}	x_2
·	·		·	·	·		·	·
0	0	$\cdots$	1	$y_{m,m+1}$	$y_{m,m+2}$	$\cdots$	y_{mn}	x_m
0	0	$\cdots$	0	r_{m+1}	r_{m+2}	$\cdots$	r_n	z_0

where for each j, $m+1 \leq j \leq n$, we have

$$\mathbf{a}_j = \sum_{i=1}^{m} y_{ij}\mathbf{a}_i \quad \text{and} \quad r_j = c_j - \sum_{i=1}^{m} c_i y_{ij} \geq 0 \tag{4.37}$$

and

$$z_0 = -(c_1x_1 + c_2x_2 + \cdots + c_mx_m) \quad \text{and} \quad (x_1, \ldots, x_m)^T = \mathbf{B}^{-1}\mathbf{b} \tag{4.38}$$

We show how we find an optimal solution for the primal instance α, starting from this tableau.

If $x_i \geq 0$ for all $1 \leq i \leq m$, then $\mathbf{x} = (x_1, \ldots, x_m, 0, \ldots, 0)$ is an optimal basic solution to the instance α so we are done.

Thus we suppose that there is an $x_p < 0$ for some $1 \leq p \leq m$. Fix this index p.

Let $\mathbf{y}^T = (c_1, \ldots, c_m)\mathbf{B}^{-1}$. Then $\mathbf{y}^T\mathbf{A} = (c_1, \ldots, c_m, c'_{m+1}, \ldots, c'_n)$, where $c'_j = \sum_{i=1}^{m} c_i y_{ij} \leq c_j$, for $j = m+1, \ldots, n$ (see the proof for Theorem 4.3.3). Thus, $\mathbf{y}^T\mathbf{A} \leq \mathbf{c}^T$ and $\mathbf{y}^T$ is a solution to the dual instance α', with the objective function value

$$\mathbf{y}^T\mathbf{b} = (c_1, \ldots, c_m)\mathbf{B}^{-1}\mathbf{b} = (c_1, \ldots, c_m)(x_1, \ldots, x_m)^T = -z_0$$

Thus, it is proper to call the vector $\mathbf{x}= (x_1, \ldots, x_m, 0, \ldots, 0)$ the *dual basic solution* to the instance α (distinguish it from the a basic solution to the dual instance α'), and the columns $\mathbf{a}_1, \ldots ,\mathbf{a}_m$ the *dual basic columns*.

Our intention is to replace a dual basic column by a new column so that the dual basic solution corresponds to an improved solution to the dual instance α'.

Note that $\mathbf{B}^{-1}\mathbf{a}_i = (0, \ldots, 0, 1, 0, \ldots, 0)^T$ is the ith unit vector of dimension m for $i = 1, \ldots, m$ (see equality (4.35)), and $\mathbf{B}^{-1}\mathbf{a}_j = (y_{1j}, y_{2j}, \ldots, y_{mj})^T$ for $j = m+1, \ldots, n$ (see equality (4.36)). Therefore, if we let $\mathbf{u}_p$ be the row vector given by the pth row of the matrix $\mathbf{B}^{-1}$, we will have

$$\mathbf{u}_p\mathbf{a}_i = \begin{cases} 0 & \text{if } i \neq p \\ 1 & \text{if } i = p \end{cases} \quad \text{for } i = 1, \ldots, m \tag{4.39}$$

and

$$\mathbf{u}_p\mathbf{a}_j = y_{pj} \quad \text{for } j = m+1, \ldots, n \tag{4.40}$$

Let $\mathbf{y}_\epsilon^T = \mathbf{y}^T - \epsilon\mathbf{u}_p$. We show that with properly selected $\epsilon > 0$, $\mathbf{y}_\epsilon$ is an improved solution to the dual instance α'.

First consider

$$\begin{aligned} \mathbf{y}_\epsilon^T\mathbf{A} &= (\mathbf{y}^T - \epsilon\mathbf{u}_p)\mathbf{A} = \mathbf{y}^T\mathbf{A} - \epsilon\mathbf{u}_p\mathbf{A} \\ &= \mathbf{y}^T\mathbf{A} - \epsilon\mathbf{u}_p[\mathbf{a}_1, \ldots, \mathbf{a}_n] \\ &= \mathbf{y}^T\mathbf{A} - \epsilon[\mathbf{u}_p\mathbf{a}_1, \ldots, \mathbf{u}_p\mathbf{a}_m, \mathbf{u}_p\mathbf{a}_{m+1}, \ldots, \mathbf{u}_p\mathbf{a}_n] \\ &= (c_1, \ldots, c_m, c'_{m+1}, \ldots, c'_n) - \epsilon(0, \ldots, 0, 1, 0, \ldots, 0, y_{p,m+1}, \ldots, y_{p,n}) \\ &= (c_1, \ldots, c_{p-1}, c_p - \epsilon, c_{p+1}, \ldots, c_m, c'_{m+1} - \epsilon y_{p,m+1}, \ldots, c'_n - \epsilon y_{pn}) \end{aligned} \tag{4.41}$$

The fifth equality is from the equalities (4.39) and (4.40).

If all $y_{pj} > 0$ for $j = m+1, \ldots, n$, then $\mathbf{y}_\epsilon$ is a solution to the dual instance α' for any $\epsilon \geq 0$ with the objective function value

$$\mathbf{y}_\epsilon^T\mathbf{b} = (\mathbf{y}^T - \epsilon\mathbf{u}_p)\mathbf{b} = \mathbf{y}^T\mathbf{b} - \epsilon\mathbf{u}_p\mathbf{b} = \mathbf{y}^T\mathbf{b} - \epsilon x_p$$

(note $\mathbf{B}^{-1}\mathbf{b} = (x_1, \ldots, x_m)^T$ thus $\mathbf{u}_p\mathbf{b} = x_p$). Since $x_p < 0$, $\mathbf{y}_\epsilon^T\mathbf{b}$ can be arbitrarily large. That is, the solution to the dual instance α' is unbounded. By Theorem 4.3.2, the primal instance α has no solution. Thus, again, we are done.

So we assume that $y_{pq} < 0$ for some q, $m+1 \leq q \leq n$. Select the index q such that $c'_q - \epsilon y_{pq}$ is the first that meets c_q when ϵ increases from 0. Therefore, the index q should be chosen as follows.

$$\epsilon_0 = -\frac{r_q}{y_{pq}} = \min_{m+1 \leq j \leq n} \{-\frac{r_j}{y_{pj}} | y_{pj} < 0\}$$

(note $y_{pq} < 0$ and $r_q \geq 0$ so $\epsilon_0 \geq 0$). We verify that $\mathbf{y}_0^T = \mathbf{y}^T - \epsilon_0 \mathbf{u}_p$ is a solution to the dual instance α', i.e., $\mathbf{y}_0^T \mathbf{A} \leq \mathbf{c}^T$.

Consider the equality (4.41). Since $\epsilon_0 \geq 0$, we have $c_p - \epsilon_0 \leq c_p$. Moreover, for each $c'_j - \epsilon_0 y_{pj}$ with $j = m+1, \ldots, n$, if $y_{pj} \geq 0$, then of course $c'_j - \epsilon_0 y_{pj} \leq c'_j \leq c_j$; while for $y_{pj} < 0$, by our choice of q we have

$$-\frac{r_q}{y_{pq}} \leq -\frac{r_j}{y_{pj}} \quad \text{or} \quad \frac{r_q}{y_{pq}} \geq \frac{r_j}{y_{pj}}$$

and we have (note $y_{pj} < 0$)

$$c'_j - \epsilon_0 y_{pj} = c'_j + \frac{r_q}{y_{pq}} y_{pj} \leq c'_j + \frac{r_j}{y_{pj}} y_{pj} = c'_j + r_j = c_j$$

This proves that $\mathbf{y}_0^T \mathbf{A} \leq \mathbf{c}^T$ and $\mathbf{y}_0$ is a solution to the dual instance α'.

Algorithm. Dual Simplex Method
INPUT: an instance α with a tableau $\mathcal{T}$ for a dual basic solution $\mathbf{x}$ to α

1. **while** ($\mathcal{T}[p, n+1] < 0$ for some $1 \leq p \leq m$) **do**
   if (no $\mathcal{T}[p, j] < 0$ for any $1 \leq j \leq n$)
   then stop: the instance α has no solution
   else let $\mathcal{T}[p, q]$ have the minimum ratio in row p;
   call **TableauMove**($\mathcal{T}, p, q$);
2. stop: the tableau $\mathcal{T}$ gives an optimal solution $\mathbf{x}$ to α.

Figure 4.4: The Dual Simplex Method algorithm

We evaluate the objective function value for $\mathbf{y}_0$:

$$\mathbf{y}_0^T \mathbf{b} = (\mathbf{y}^T - \epsilon_0 \mathbf{u}_p)\mathbf{b} = \mathbf{y}^T \mathbf{b} - \epsilon_0 \mathbf{u}_p \mathbf{b} = \mathbf{y}^T \mathbf{b} - \epsilon_0 x_p$$

Since $x_p < 0$, we have $\mathbf{y}_0^T \mathbf{b} \geq \mathbf{y}^T \mathbf{b}$. In particular, if $\epsilon_0 > 0$, then $\mathbf{y}_0$ is an improvement over the solution $\mathbf{y}$ for the dual instance α'. The case $\epsilon_0 = 0$, i.e., $r_q = 0$, is the *degenerate situation* for the dual simplex method. As we have discussed for the regular simplex method, the dual simplex method in general works fine with degenerate situations, and special techniques can be adopted to handle degenerate situations.

Therefore, the above procedure illustrates how we obtain an improved solution for the dual instance α' by replacing a dual basic column $\mathbf{a}_p$ by a new column $\mathbf{a}_q$. Note that this replacement is not done based on the tableau for the solution $\mathbf{y}$ to the dual instance α'. Instead, it is accomplished based on the tableau for the dual basic solution $\mathbf{x}$ for the primal instance α. This replacement can be simply done by calling the algorithm **TableauMove**$(\mathcal{T}, p, q)$ in Figure 4.2 when the indices p and q are decided. We summarize this method in Figure 4.4, where we say that an element $\mathcal{T}[p, q]$ in the pth row in $\mathcal{T}$ has the *minimum ratio* if the ratio $-r_q/y_{pq}$ is the minimum over all $-r_j/y_{pj}$ with $y_{pj} < 0$.

Example 4.3.1. Consider the following instance α for the LINEAR PROGRAMMING problem.

$$\begin{aligned} \text{minimize} \quad & 3x_1 + 4x_2 + 5x_3 \\ \text{subject to} \quad & -x_1 - 2x_2 - 3x_3 + x_4 = -5 \\ & -2x_1 - 2x_2 - x_3 + x_5 = -6 \\ & x_1, x_2, x_3, x_4, x_5 \geq 0 \end{aligned}$$

The dual basic solution $\mathbf{x} = (0, 0, 0, -5, -6)^T$ to α has the tableau

$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	
-1	-2	-3	1	0	-5
$\boxed{-2}$	-2	-1	0	1	-6
3	4	5	0	0	0

Pick $x_5 = -6$. To find a proper element in the second row, we compute the ratios $-r_q/y_{2q}$ and select the one with the minimum ratio. This makes us to pick $y_{21} = -2$ (as indicated by the box). Applying the algorithm **TableauMove**$(\mathcal{T}, 2, 1)$ gives us

$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	
0	$\boxed{-1}$	$-5/2$	1	$-1/2$	-2
1	1	$1/2$	0	$-1/2$	3
0	1	$7/2$	0	$3/2$	-9

Now pick $x_4 = -2$ and choose the element $y_{12} = -1$ (as indicated in the box). The algorithm **Tableau**$(\mathcal{T}, 1, 2)$ results in

$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	
0	1	$5/2$	-1	$1/2$	2
1	0	-2	1	-1	1
0	0	1	1	1	-11

The last tableau yields a dual basic solution $\mathbf{x}_0 = (1, 2, 0, 0, 0)^T$ with $\mathbf{x}_0 \geq 0$. Thus it must be an optimal solution to the instance α. The objective fucntion value on $\mathbf{x}_0$ is 11.

4.4 Polynomial time algorithms

It was an outstanding open problem whether the Linear Programming problem could be solved in polynomial time, until the spring of 1979, the Russian mathematician L. G. Khachian published a proof that an algorithm, called *the Ellipsoid Algorithm*, solves the Linear Programming problem in polynomial time [85]. Despite the great theoretical value of the Ellipsoid Algorithm, it is not clear at all that this algorithm can be practically useful. The most obvious among many obstacles is the large precision apparently required.

Another polynomial time algorithm for the Linear Programming problem, called the *Projective Algorithm*, or more generally, the *Interior Point Algorithm*, was published by N. Karmarkar in 1984 [79]. The Projective Algorithm, and its derivatives, have great impact in the study of the Linear Programming problem.