PROBLEMS

△ **2.1** Solve the following problems by the simplex method:

a. maximize $3x_1 + 2x_2 + 4x_3$

subject to
$$\begin{aligned} x_1 + x_2 + 2x_3 &\le 4 \\ 2x_1 + 3x_3 &\le 5 \\ 2x_1 + x_2 + 3x_3 &\le 7 \\ x_1, x_2, x_3 &\ge 0 \end{aligned}$$

b. maximize $5x_1 + 6x_2 + 9x_3 + 8x_4$

subject to
$$\begin{aligned} x_1 + 2x_2 + 3x_3 + x_4 &\le 5 \\ x_1 + x_2 + 2x_3 + 3x_4 &\le 3 \\ x_1, x_2, x_3, x_4 &\ge 0 \end{aligned}$$

c. maximize $2x_1 + x_2$

subject to
$$\begin{aligned} 2x_1 + 3x_2 &\le 3 \\ x_1 + 5x_2 &\le 1 \\ 2x_1 + x_2 &\le 4 \\ 4x_1 + x_2 &\le 5 \\ x_1, x_2 &\ge 0. \end{aligned}$$

2.2 Use the simplex method to describe *all* the optimal solutions of the following problem:

maximize $2x_1 + 3x_2 + 5x_3 + 4x_4$

subject to
$$\begin{aligned} x_1 + 2x_2 + 3x_3 + x_4 &\le 5 \\ x_1 + x_2 + 2x_3 + 3x_4 &\le 3 \\ x_1, x_2, x_3, x_4 &\ge 0. \end{aligned}$$

3

Pitfalls and How to Avoid Them

The examples illustrating the simplex method in the preceding chapter were purposely smooth. They did not point out the dangers that can occur. The purpose of the present chapter, therefore, is to rigorously analyze the method by scrutinizing its every step.

THREE KINDS OF PITFALLS

Three kinds of pitfalls can occur in the simplex method.

(i) INITIALIZATION. We might not be able to start: How do we get hold of a feasible dictionary?

(ii) ITERATION. We might get stuck in some iteration: Can we always choose an entering variable, find the leaving variable, and construct the next feasible dictionary by pivoting?

(iii) TERMINATION. We might not be able to finish: Can the simplex method construct an endless sequence of dictionaries without ever reaching an optimal solution?

In the preceding chapter, INITIALIZATION never came up. Given a problem

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^{n} c_j x_j \\ &\text{subject to} && \sum_{j=1}^{n} a_{ij} x_j \le b_i \qquad (i = 1, 2, \ldots, m) \\ & && x_j \ge 0 \qquad (j = 1, 2, \ldots, n) \end{aligned} \tag{3.1}$$

we constructed the initial feasible dictionary by simply writing down the formulas defining the slack variables and the objective function,

$$\begin{aligned} x_{n+i} &= b_i - \sum_{j=1}^{n} a_{ij} x_j \qquad (i = 1, 2, \ldots, m) \\ \hline z &= \sum_{j=1}^{n} c_j x_j \, . \end{aligned}$$

In general, this dictionary is feasible if and only if each right-hand side, b_i, in (3.1) is nonnegative. This is the case if and only if

$$x_1 = 0, \quad x_2 = 0, \ldots, x_n = 0$$

is a feasible solution of (3.1). Since the set of zero values is sometimes called the "origin," problems (3.1) with each right-hand side b_i nonnegative are referred to as problems with a *feasible origin*. For the moment, we shall avoid the pitfalls of INITIALIZATION by default: we shall restrict ourselves to problems with a feasible origin. Problems with an infeasible origin are discussed on pages 39–42.

Iteration

Given some feasible dictionary, we have to select an entering variable, to find a leaving variable, and to construct the next feasible dictionary by pivoting.

Choosing an entering variable. The entering variable is *a nonbasic variable x_j with a positive coefficient $\bar{c}_j$ in the last row of the current dictionary*. This rule is ambiguous in the sense that it may provide more than one candidate for entering the basis, or no candidate at all. The latter alternative implies that the current dictionary describes an optimal solution, at which point the method may terminate. More precisely, consider the last row of our current dictionary,

$$z = z^* + \sum_{j \in N} \bar{c}_j x_j$$

with N standing for the set of subscripts j of nonbasic variables x_j. Our current solution, with $x_j = 0$ whenever $j \in N$, gives the objective function the numerical value of z^*. If $\bar{c}_j \le 0$ whenever $j \in N$, then every feasible solution, with $x_j \ge 0$ whenever $j \in N$, gives the objective function a numerical value of at most z^*; hence the current solution is optimal. On the other hand, if there is more than one candidate for entering the basis, then any of these candidates may serve. (In hand calculations involving small problems, it is customary to choose the candidate x_j that has the largest coefficient $\bar{c}_j$. In most computer implementations of the simplex method, however, this practice is abandoned. More on this subject in Chapter 7.)

Finding the leaving variable. The leaving variable is *that basic variable whose nonnegativity imposes the most stringent upper bound on the increase of the entering variable.* Again, this rule is ambiguous in the sense that it may provide more than one candidate for leaving the basis, or no candidate at all. The latter alternative is illustrated on the dictionary

$$\begin{aligned} x_2 &= 5 + 2x_3 - x_4 - 3x_1 \\ x_5 &= 7 \qquad - 3x_4 - 4x_1 \\ \hline z &= 5 + x_3 - x_4 - x_1 . \end{aligned}$$

The entering variable is x_3, but neither of the two basic variables x_2, x_5 imposes an upper bound on its increase. Therefore, we can make x_3 as large as we wish (maintaining $x_1 = x_4 = 0$) and still retain feasibility: setting $x_3 = t$ for any positive t, we obtain a feasible solution with $x_1 = 0$, $x_2 = 5 + 2t$, $x_4 = 0$, $x_5 = 7$, and $z = 5 + t$. Since t can be made arbitrarily large, z can be made arbitrarily large. We conclude that the problem is *unbounded*: for every number M, there is a feasible solution $x_1, x_2, \ldots, x_5$ such that $x_3 - x_4 - x_1 > M$. The same conclusion can be reached in general: if there is no candidate for leaving the basis, then we can make the value of the entering variable, and therefore also the value of the objective function, as large as we wish. In that case, the problem is unbounded. On the other hand, if there is more than one candidate for leaving the basis, then any of these candidates may serve. Once the entering and leaving variables have been selected, pivoting is a straightforward matter.

Degeneracy. The presence of more than one candidate for leaving the basis has interesting consequences. For illustration, consider the dictionary

$$\begin{aligned} x_4 &= 1 \qquad\qquad - 2x_3 \\ x_5 &= 3 - 2x_1 + 4x_2 - 6x_3 \\ x_6 &= 2 + x_1 - 3x_2 - 4x_3 \\ \hline z &= \qquad 2x_1 - x_2 + 8x_3 . \end{aligned}$$

Having chosen x_3 to enter the basis, we find that each of the three basic variables x_4, x_5, x_6 limits the increase of x_3 to $\frac{1}{2}$. Hence each of these three variables is a candidate for leaving the basis. We arbitrarily choose x_4. Pivoting as usual, we obtain the dictionary

$$
\begin{aligned}
x_3 &= 0.5 \qquad\qquad - 0.5x_4 \\
x_5 &= -2x_1 + 4x_2 + 3x_4 \\
x_6 &= x_1 - 3x_2 + 2x_4 \\
\hline
z &= 4 + 2x_1 - x_2 - 4x_4.
\end{aligned}
$$

This dictionary differs from all the dictionaries we have encountered so far in one important respect: along with the nonbasic variables, the basic variables x_5 and x_6 have value zero in the associated solution. Basic solutions with one or more basic variables at zero are called *degenerate.*

Although harmless in its own right, degeneracy may have annoying side effects. These are illustrated on the next iteration in our example. There, x_1 enters the basis and x_5 leaves; because of degeneracy, the constraint $x_5 \geq 0$ limits the increment of x_1 to *zero*. Hence the value of x_1 will remain unchanged, and so will the values of the remaining variables and the value of the objective function z. This is annoying, for the motivation behind the simplex method is a desire to increase the value of z in each iteration. In this particular iteration, that desire remains unfulfilled: pivoting changes the dictionary into

$$
\begin{aligned}
x_1 &= 2x_2 + 1.5x_4 - 0.5x_5 \\
x_3 &= 0.5 \qquad - 0.5x_4 \\
x_6 &= -x_2 + 3.5x_4 - 0.5x_5 \\
\hline
z &= 4 + 3x_2 - x_4 - x_5
\end{aligned}
$$

but it does not affect the associated solution at all. Simplex iterations that do not change the basic solution are called *degenerate.* (As the reader may verify, the next iteration is degenerate again, but the one after that turns out to be nondegenerate and brings us to the optimal solution.)

In a sense, degeneracy is something of an accident: a basic variable may vanish only if the results of successive pivot operations just happen to cancel each other out. And yet degeneracy abounds in LP problems arising from practical applications. It has been said that nearly all such problems yield degenerate basic feasible solutions at some stage of the simplex method. Whenever that happens, the simplex method may stall by going through a few (and sometimes quite a few) degenerate iterations in a row. Typically, such a block of degenerate iterations ends with a breakthrough represented by a nondegenerate iteration; an example of the atypical case is presented next.

Termination: Cycling

Can the simplex method go through an endless sequence of iterations without ever finding an optimal solution? Yes, it can. To justify this claim, let us consider the initial dictionary

$$
\begin{aligned}
x_5 &= -0.5x_1 + 5.5x_2 + 2.5x_3 - 9x_4 \\
x_6 &= -0.5x_1 + 1.5x_2 + 0.5x_3 - x_4 \\
x_7 &= 1 - x_1 \\
\hline
z &= 10x_1 - 57x_2 - 9x_3 - 24x_4
\end{aligned}
$$

and let us agree on the following:

(i) The entering variable will always be the nonbasic variable that has the largest coefficient in the z-row of the dictionary.

(ii) If two or more basic variables compete for leaving the basis, then the candidate with the smallest subscript will be made to leave.

Now the sequence of dictionaries constructed in the first six iterations goes as follows.

After the first iteration:

$$
\begin{aligned}
x_1 &= 11x_2 + 5x_3 - 18x_4 - 2x_5 \\
x_6 &= -4x_2 - 2x_3 + 8x_4 + x_5 \\
x_7 &= 1 - 11x_2 - 5x_3 + 18x_4 + 2x_5 \\
\hline
z &= 53x_2 + 41x_3 - 204x_4 - 20x_5.
\end{aligned}
$$

After the second iteration:

$$
\begin{aligned}
x_2 &= -0.5x_3 + 2x_4 + 0.25x_5 - 0.25x_6 \\
x_1 &= -0.5x_3 + 4x_4 + 0.75x_5 - 2.75x_6 \\
x_7 &= 1 + 0.5x_3 - 4x_4 - 0.75x_5 - 13.25x_6 \\
\hline
z &= 14.5x_3 - 98x_4 - 6.75x_5 - 13.25x_6.
\end{aligned}
$$

After the third iteration:

$$
\begin{aligned}
x_3 &= 8x_4 + 1.5x_5 - 5.5x_6 - 2x_1 \\
x_2 &= -2x_4 - 0.5x_5 + 2.5x_6 + x_1 \\
x_7 &= 1 - x_1 \\
\hline
z &= 18x_4 + 15x_5 - 93x_6 - 29x_1.
\end{aligned}
$$

After the fourth iteration:

$$
\begin{aligned}
x_4 &= -0.25x_5 + 1.25x_6 + 0.5x_1 - 0.5x_2 \\
x_3 &= -0.5x_5 + 4.5x_6 + 2x_1 - 4x_2 \\
x_7 &= 1 - x_1 \\
\hline
z &= 10.5x_5 - 70.5x_6 - 20x_1 - 9x_2.
\end{aligned}
$$

After the fifth iteration:

$$\begin{aligned} x_5 &= 9x_6 + 4x_1 - 8x_2 - 2x_3 \\ x_4 &= -x_6 - 0.5x_1 + 1.5x_2 + 0.5x_3 \\ x_7 &= 1 - x_1 \\ \hline z &= 24x_6 + 22x_1 - 93x_2 - 21x_3. \end{aligned}$$

After the sixth iteration:

$$\begin{aligned} x_6 &= -0.5x_1 + 1.5x_2 + 0.5x_3 - x_4 \\ x_5 &= -0.5x_1 + 5.5x_2 + 2.5x_3 - 9x_4 \\ x_7 &= 1 - x_1 \\ \hline z &= 10x_1 - 57x_2 - 9x_3 - 24x_4. \end{aligned}$$

Since the dictionary constructed after the sixth iteration is identical with the initial dictionary, the method will go through the same six iterations again and again without ever finding the optimal solution (which, as we shall see later, has $z = 1$). This phenomenon is known as cycling. More precisely, we say that the simplex method *cycles* if one dictionary appears in two different iterations (and so the sequence of iterations leading from the dictionary to itself can be repeated over and over without end). Note that cycling can occur only in the presence of degeneracy: since the value of the objective function increases with each nondegenerate iteration and remains unchanged after each degenerate one, all the iterations in the sequence leading from a dictionary to itself must be degenerate. Cycling is one reason why the simplex method may fail to terminate; the following theorem shows that it is the only reason.

THEOREM 3.1. If the simplex method fails to terminate, then it must cycle.

PROOF. To begin, note that there are only finitely many ways of choosing m basic variables from all the $n + m$ variables. Thus, if the simplex method fails to terminate, then some basis must appear in two different iterations. Now it only remains to be proved that any two dictionaries with the same basis must be identical. (This fact becomes trivial as soon as one describes dictionaries in terms of matrices, as we shall do in Chapter 7. Nevertheless, we can and shall present an easy proof from scratch right now.) Consider two dictionaries

$$\begin{aligned} x_i &= b_i - \sum_{j \notin B} a_{ij}x_j \qquad (i \in B) \\ \hline z &= v + \sum_{j \notin B} c_j x_j \end{aligned} \tag{3.2}$$

and

$$\begin{aligned} x_i &= b_i^* - \sum_{j \notin B} a_{ij}^* x_j \qquad (i \in B) \\ \hline z &= v^* + \sum_{j \notin B} c_j^* x_j \end{aligned} \tag{3.3}$$

with the same set of basic variables x_i ($i \in B$). It is a defining property of dictionaries that every solution $x_1, x_2, \ldots, x_{n+m}, z$ of (3.2) is a solution of (3.3) and vice versa. In particular, if x_k is a nonbasic variable and if t is a number, then the numbers

$$x_k = t, \quad x_j = 0 \ (j \notin B \quad \text{and} \quad j \neq k), \quad x_i = b_i - a_{ik}t \ (i \in B), \quad z = v + c_k t,$$

constituting a solution of (3.2), must satisfy (3.3). Hence,

$$b_i - a_{ik}t = b_i^* - a_{ik}^* t \quad \text{for all} \quad i \in B, \quad \text{and} \quad v + c_k t = v^* + c_k^* t.$$

Since these identities must hold for all numbers t, we have

$$b_i = b_i^*, \ a_{ik} = a_{ik}^* \quad \text{for all} \quad i \in B, \quad \text{and} \quad v = v^*, \ c_k = c_k^*.$$

Since x_k was an arbitrary nonbasic variable, the two dictionaries are identical. ■

Cycling is a rare phenomenon. In fact, constructing an LP problem on which the simplex method may cycle is difficult. [Our example is adapted from K. T. Marshall and J. W. Suurballe (1969). The first example of this size was constructed by E. M. L. Beale (1955) and the first example ever was constructed by A. J. Hoffman (1953). Incidentally, Marshall and Suurballe (1969) proved that if the simplex method cycles off-optimum on a problem that has an optimal solution, then the dictionaries must involve at least six variables and at least three equations.] P. Wolfe (1963) and T. C. T. Kotiah and D. I. Steinberg (1978) reported having come across practical problems that cycled (in 25 and 18 iterations, respectively) but such reports are scarce. For this reason, the remote possibility of cycling is disregarded in most computer implementations of the simplex method.

There are ways of preventing the occurrence of cycling altogether. The classic *perturbation method and lexicographic method* avoid cycling by a judicious choice of the leaving variable in each simplex iteration; the more recent *smallest-subscript rule* does so by an easy choice of *both* the entering and the leaving variables. The former alternative maintains the freedom of choice among different candidates for entering the basis, but it requires extra computations to choose the leaving variable; the latter alternative requires no extra work at all, but it gives up the multitude of choices for the entering variable. We shall explain the details of both.

The perturbation method and the lexicographic method. The perturbation and the lexicographic methods are closely related. The perturbation method, suggested first by A. Orden and developed independently by A. Charnes (1952), provides an intuitive motivation for the lexicographic method of G. B. Dantzig, A. Orden, and P. Wolfe (1955). The lexicographic method can be seen as an implementation of the perturbation method.

The starting point relies on the observations that cycling can be stamped out by stamping out degeneracy and that degeneracy itself is something of an accident. To elaborate on the second observation, consider a degenerate dictionary. The basic variables currently at zero would most likely assume small nonzero values if the initial right-hand sides, b_i, were changed slightly; at the same time, if these changes were truly microscopic, then the problem could be considered unchanged for all practical purposes. One way of exploiting these observations is to add a small positive ε to each b_i, and then to apply the simplex method to the resulting problem. This trick (with $\varepsilon = 10^{-6}$ or so) is actually used in some computer implementations of the simplex method; it helps to reduce the number of degenerate iterations. Nevertheless, it does not constitute a reliable safeguard against cycling: for instance, if the simplex method is applied to the problem

$$\begin{aligned}
\text{maximize} \quad & 10x_1 - 57x_2 - 9x_3 - 24x_4 + 100x_5 \\
\text{subject to} \quad & x_5 \le 1 + \varepsilon \\
& 0.5x_1 - 5.5x_2 - 2.5x_3 + 9x_4 + x_5 \le 1 + \varepsilon \\
& 0.5x_1 - 1.5x_2 - 0.5x_3 + x_4 + x_5 \le 1 + \varepsilon \\
& x_1 + x_5 \le 2 + \varepsilon \\
& x_1, x_2, \ldots, x_5 \ge 0
\end{aligned}$$

then the degenerate dictionary

$$\begin{aligned}
x_5 &= 1 + \varepsilon - x_6 \\
x_7 &= -0.5x_1 + 5.5x_2 + 2.5x_3 - 9x_4 + x_6 \\
x_8 &= -0.5x_1 + 1.5x_2 + 0.5x_3 - x_4 + x_6 \\
x_9 &= 1 - x_1 + x_6 \\
\hline
z &= 100 + 100\varepsilon + 10x_1 - 57x_2 - 9x_3 - 24x_4 - 100x_6
\end{aligned}$$

is obtained after the first iteration and, as the reader may verify, the simplex method cycles in the next six iterations. (The cycle is essentially the same as that of the preceding example.)

What went wrong here was that the small amounts ε added to the right-hand sides cancelled each other out in the first iteration. To guarantee that such cancellations will never take place (and therefore all the dictionaries will remain nondegenerate), we shall perturb the different right-hand sides $b_1, b_2, \ldots, b_m$ by radically different amounts $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$. More precisely, we shall choose a very small ε_1 and then make each ε_{i+1} much smaller than the preceding ε_i: in symbols,

$$0 < \varepsilon_m \ll \varepsilon_{m-1} \ll \cdots \ll \varepsilon_2 \ll \varepsilon_1 \ll 1. \tag{3.4}$$

Then we shall apply the simplex method to the perturbed problem

$$\begin{aligned}
\text{maximize} \quad & \sum_{j=1}^{n} c_j x_j \\
\text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j \le b_i + \varepsilon_i \qquad (i = 1, 2, \ldots, m) \\
& x_j \ge 0 \qquad (j = 1, 2, \ldots, n).
\end{aligned}$$

This is the *perturbation method.* (The perturbation method is usually presented with $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ equal to the powers $\varepsilon, \varepsilon^2, \ldots, \varepsilon^m$ of the same small ε. Our version makes the subsequent analysis a little more transparent.) For illustration, let us return to our first example on which the simplex method cycled. There, the initial dictionary reads

$$\begin{aligned}
x_5 &= \varepsilon_1 - 0.5x_1 + 5.5x_2 + 2.5x_3 - 9x_4 \\
x_6 &= \varepsilon_2 - 0.5x_1 + 1.5x_2 + 0.5x_3 - x_4 \\
x_7 &= 1 + \varepsilon_3 - x_1 \\
\hline
z &= 10x_1 - 57x_2 - 9x_3 - 24x_4.
\end{aligned}$$

Again, the entering variable is x_1. The constraints $x_5 \ge 0$, $x_6 \ge 0$, and $x_7 \ge 0$ limit the increase of x_1 to $2\varepsilon_1$, $2\varepsilon_2$, and $1 + \varepsilon_3$, respectively. Since $2\varepsilon_2 < 2\varepsilon_1 < 1 + \varepsilon_3$, the leaving variable is x_6, and the next dictionary reads

$$\begin{aligned}
x_1 &= 2\varepsilon_2 + 3x_2 + x_3 - 2x_4 - 2x_6 \\
x_5 &= \varepsilon_1 - \varepsilon_2 + 4x_2 + 2x_3 - 8x_4 + x_6 \\
x_7 &= 1 - 2\varepsilon_2 + \varepsilon_3 - 3x_2 - x_3 + 2x_4 + 2x_6 \\
\hline
z &= 20\varepsilon_2 - 27x_2 + x_3 - 44x_4 - 20x_6.
\end{aligned}$$

Now the only candidate for the entering variable is x_3 and the only candidate for the leaving variable is x_7. The resulting dictionary,

$$\begin{aligned}
x_3 &= 1 - 2\varepsilon_2 + \varepsilon_3 - 3x_2 + 2x_4 + 2x_6 - x_7 \\
x_1 &= 1 + \varepsilon_3 - x_7 \\
x_5 &= 2 + \varepsilon_1 - 5\varepsilon_2 + 2\varepsilon_3 - 2x_2 - 4x_4 + 5x_6 - 2x_7 \\
\hline
z &= 1 + 18\varepsilon_2 + \varepsilon_3 - 30x_2 - 42x_4 - 18x_6 - x_7
\end{aligned}$$

is the optimal dictionary for the perturbed problem. It may be converted into the optimal dictionary for the original problem by simply disregarding all the terms involving $\varepsilon_1, \varepsilon_2, \varepsilon_3$.

How should we choose the numerical values of $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$? The simplest answer is that we do not have to do that at all: rather than committing ourselves to definite values of $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$, we may just think of these symbols as representing indefinite quantities, which satisfy (3.4). After several iterations of the simplex method, these symbols spread throughout the various rows of the dictionary, but they remain confined to the absolute terms in each of the $m + 1$ rows; the coefficients at the nonbasic variables in the dictionary are unaffected by the perturbation. Now when it comes to finding the leaving variable, each of the constraints $x_i \ge 0$ for a nonbasic x_i limits the increase of the entering x_j to a quantity such as $2\varepsilon_1$, $2\varepsilon_2$, $1 + \varepsilon_3$, or, more generally,

$$r = r_0 + r_1\varepsilon_1 + \cdots + r_m\varepsilon_m, \quad s = s_0 + s_1\varepsilon_1 + \cdots + s_m\varepsilon_m \tag{3.5}$$

and so on. As we are about to explain, assumption (3.4) allows us to compare the numerical values of such quantities without referring to the precise values of $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$. If r and s in (3.5) are distinct, then there is the smallest subscript k such that $r_k \ne s_k$. It is customary to say that r is *lexicographically smaller* than s if $r_k < s_k$. (The choice of the term *lexicographically* is explained by observing that, for instance, $2 + 21\varepsilon_1 + 19\varepsilon_2 + 20\varepsilon_2$ is lexicographically smaller than $2 + 21\varepsilon_1 + 20\varepsilon_2 + 20\varepsilon_3 + 15\varepsilon_4 + 14\varepsilon_5$ for the same reason that "bust" comes before "button"

in a dictionary.) It is easy to prove that r is lexicographically smaller than s if and only if r is numerically smaller than s for all values of $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ that satisfy (3.4). This statement has to be made precise by specifying just what is meant by the symbol $\ll$ in (3.4); we leave the details for problem 3.7.

The *lexicographic method* is that implementation of the perturbation method in which $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ are treated as symbols, and quantities such as r and s in (3.5) are compared by the lexicographic rule. Note that it is always possible to choose the leaving variable by the lexicographic rule: in every finite set of expressions such as r and s in (3.5), there is always one that is lexicographically smaller than or equal to all the others. Even though this fact may be taken for granted intuitively, rigor requires that it be proved; we leave the details for problem 3.6. Another fine point concerns the behavior of the objective function z. The value of z, equal to some expression $v_0 + v_1\varepsilon_1 + \cdots + v_m\varepsilon_m$, remains unchanged in each degenerate iteration and increases, in the lexicographic sense, with each nondegenerate one. (In our example, the increase from 0 to $20\varepsilon_2$ in the first iteration was followed by the increase from $20\varepsilon_2$ to $1 + 18\varepsilon_2 + \varepsilon_3$ in the second iteration.) It is intuitively obvious that the total of two or more lexicographic increases is a lexicographic increase; a rigorous proof of this fact follows from the result of problem 3.5. Now it follows that, even in the generalized context of the lexicographic method, cycling is possible only in the presence of degeneracy. Finally, note that the only function of the terms involving $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ is to guide us toward the appropriate choice of a leaving variable whenever two or more candidates present themselves in the original problem. If, at any moment, these terms are deleted, then the dictionary for the perturbed problem reduces to a dictionary for the original problem.

THEOREM 3.2. The simplex method terminates as long as the leaving variable is selected by the lexicographic rule in each iteration.

PROOF. In view of the preceding remarks, we need merely prove that no degenerate dictionary will be constructed. (If all dictionaries are nondegenerate, then all iterations are nondegenerate. In that case, cycling cannot occur and the desired conclusion follows from Theorem 3.1.) Thus, we need only consider an arbitrary row

$$x_k = (r_0 + r_1\varepsilon_1 + \cdots + r_m\varepsilon_m) - \sum_{j \notin B} d_j x_j \tag{3.6}$$

of an arbitrary dictionary and to prove that at least one of the $m + 1$ numbers $r_0, r_1, \ldots, r_m$ is distinct from zero. (Actually, we shall prove that at least one of the m numbers $r_1, r_2, \ldots, r_m$ is distinct from zero.) Writing $d_k = 1$ and $d_i = 0$ for all basic variables x_i distinct from x_k, we record (3.6) as

$$\sum_{j=1}^{n+m} d_j x_j = r_0 + \sum_{i=1}^{m} r_i\varepsilon_i. \tag{3.7}$$

Since this equation has been obtained by algebraic manipulations from the definitions of the slack variables,

$$x_{n+i} = b_i + \varepsilon_i - \sum_{j=1}^{n} a_{ij}x_j \qquad (i = 1, 2, \ldots, m) \tag{3.8}$$

it must hold for all choices of numbers $x_1, x_2, \ldots, x_{n+m}$ and $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ that satisfy (3.8). Hence, the equation

$$\sum_{j=1}^{n} d_j x_j + \sum_{i=1}^{m} d_{n+i}(b_i + \varepsilon_i - \sum_{j=1}^{n} a_{ij}x_j) = r_0 + \sum_{i=1}^{m} r_i\varepsilon_i$$

which is obtained by substituting from (3.8) into (3.7), must hold for all choices of numbers x_1, $x_2, \ldots, x_n$ and $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$. Writing this identity as

$$\sum_{j=1}^{n} (d_j - \sum_{i=1}^{m} d_{n+i}a_{ij})x_j + \sum_{i=1}^{m} (d_{n+i} - r_i)\varepsilon_i = r_0 - \sum_{i=1}^{m} d_{n+i}b_i$$

we observe that the coefficient at each x_j, the coefficient at each ε_i, and the right-hand side must equal zero. Thus

$$\begin{aligned} d_{n+i} &= r_i \qquad \text{for all} \quad i = 1, 2, \ldots, m \\ d_j &= \sum_{i=1}^{m} d_{n+i}a_{ij} \qquad \text{for all} \quad j = 1, 2, \ldots, n. \end{aligned} \tag{3.9}$$

If all the numbers $r_1, r_2, \ldots, r_m$ were equal to zero, then (3.9) would imply $d_{n+i} = 0$ for all $i = 1, 2, \ldots, m$ and $d_j = 0$ for all $j = 1, 2, \ldots, n$, contradicting the fact that $d_k = 1$. ■

With the hindsight provided by Theorem 3.2, it becomes easy to prove that every LP problem in the standard form can be perturbed by adding suitable small *numbers* $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ to the right-hand sides $b_1, b_2, \ldots, b_m$ in such a way that the simplex method applied to the perturbed problem will terminate. In fact, the numbers $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ may be chosen as the powers $\varepsilon, \varepsilon^2, \ldots, \varepsilon^m$ of any sufficiently small positive ε. We leave the details for problem 3.8.

As we have observed, the terms involving $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ are needed only when a tie has to be broken between two or more candidates for leaving the basis. Thus we might just as well wait until such a need arises, and only then introduce an ad hoc perturbation. This idea was developed by P. Wolfe (1963); its lexicographic counterpart comes from G. B. Dantzig (1960).

Smallest-subscript rule. This term will refer to breaking ties in the choice of the entering and leaving variables by always choosing the candidate x_k that has the smallest subscript k. The motivation for this elegant concept is provided by the following result.

THEOREM 3.3. [R. G. Bland (1977).] The simplex method terminates as long as the entering and leaving variables are selected by the smallest-subscript rule in each iteration.

PROOF. By virtue of Theorem 3.1, we need only show that cycling is impossible when the smallest-subscript rule is used. We shall do this by deriving a contradiction from the assumption that the smallest-subscript rule leads from some dictionary D_0 to itself in a sequence of degenerate iterations. For definiteness, let us say that this sequence of iterations produces dictionaries D_1, $D_2, \ldots, D_k$ such that $D_k = D_0$. A variable will be called *fickle* if it is nonbasic in some of these dictionaries and basic in others. Among all the fickle variables, let x_t have the largest subscript. In the sequence $D_0, D_1, \ldots, D_k$, there is a dictionary D with x_t leaving (basic in D but nonbasic in the next dictionary), and some other fickle variable x_s entering (nonbasic in D but basic in the

next dictionary). Further along in the sequence $D_0, D_1, \ldots, D_k, D_1, D_2, \ldots, D_k$, there must be a dictionary D^* with x_t entering. Let us record D as

$$\begin{aligned} x_i &= b_i - \sum_{j \notin B} a_{ij} x_j \qquad (i \in B) \\ \hline z &= v + \sum_{j \notin B} c_j x_j \,. \end{aligned}$$

Since all the iterations leading from D to D^* are degenerate, the objective function z must have the same value v in both dictionaries. Thus, the last row of D^* may be recorded as

$$z = v + \sum_{j=1}^{n+m} c_j^* x_j$$

with $c_j^* = 0$ whenever x_j is basic in D^*. Since this equation has been obtained from D by algebraic manipulations, it must be satisfied by every solution of D. In particular, it must be satisfied by $x_s = y$, $x_j = 0$ ($j \notin B$ but $j \neq s$), $x_i = b_i - a_{is} y$ ($i \in B$) and $z = v + c_s y$ for every choice of y. Thus we have

$$v + c_s y = v + c_s^* y + \sum_{i \in B} c_i^* (b_i - a_{is} y)$$

and, after simplification,

$$\left(c_s - c_s^* + \sum_{i \in B} c_i^* a_{is} \right) y = \sum_{i \in B} c_i^* b_i$$

for every choice of y. Since the right-hand side of the last equation is a constant independent of y, we conclude that

$$c_s - c_s^* + \sum_{i \in B} c_i^* a_{is} = 0. \tag{3.10}$$

The rest is easy. Since x_s is entering in D, we have $c_s > 0$. Since x_s is not entering in D^* and yet $s < t$, we have $c_s^* \leq 0$. Hence (3.10) implies that

$$c_r^* a_{rs} < 0 \qquad \text{for some} \quad r \in B. \tag{3.11}$$

Since $r \in B$, the variable x_r is basic in D; since $c_r^* \neq 0$, the same variable is nonbasic in D^*. Hence, x_r is fickle and we have $r \leq t$. Actually, x_r is different from x_t: since x_t is leaving in D, we have $a_{ts} > 0$ and so $c_t^* a_{ts} > 0$. Now $r < t$ and yet x_r is not entering in D^*. Thus, we cannot have $c_r^* > 0$. From (3.11), we conclude that

$$a_{rs} > 0.$$

Since all the iterations leading from D to D^* are degenerate, the two dictionaries describe the same solution. In particular, the value of x_r is zero in both dictionaries (x_r is nonbasic in D^*) and so $b_r = 0$. Hence x_r was a candidate for leaving the basis of D—yet we picked x_t, even though $r < t$. This contradiction completes the proof. ■

One further point: termination of the simplex method can be guaranteed even without abiding by the smallest-subscript rule in every single iteration. We might resort to the smallest-subscript rule, for instance, only when the last fifty or so iterations were degenerate, and abandon it after the next nondegenerate iteration in favor of any other way of choosing the entering and leaving variables. Although cycling might conceivably take place in this case, each block of consecutive degenerate iterations would be followed by a nondegenerate iteration, and so each dictionary would be constructed only a finite number of times. □

Initialization

The only remaining point that needs to be explained is getting hold of the initial feasible dictionary in a problem

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^{n} c_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j \leq b_i \qquad (i = 1, 2, \ldots, m) \\ & x_j \geq 0 \qquad (j = 1, 2, \ldots, n) \end{aligned}$$

with an infeasible origin. The trouble with an infeasible origin is twofold. First, it may not be clear that our problem has any feasible solutions at all. Second, even if a feasible solution is apparent, a feasible dictionary may not be. One way of getting around both obstacles uses a so-called *auxiliary problem*,

$$\begin{aligned} \text{minimize} \quad & x_0 \\ \text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j - x_0 \leq b_i \qquad (i = 1, 2, \ldots, m) \\ & x_j \geq 0 \qquad (j = 0, 1, \ldots, n). \end{aligned}$$

A feasible solution of the auxiliary problem is readily available: it suffices to set the value of each x_j with $1 \leq j \leq n$ at zero and make the value of x_0 sufficiently large. Furthermore, it is easy to see that the original problem has a feasible solution *if and only if* the auxiliary problem has a feasible solution with $x_0 = 0$. To put it differently, the original problem has a feasible solution if and only if the optimum value of the auxiliary problem is zero. Hence our plan is to solve the auxiliary problem first; the technical details are illustrated on the problem

$$\begin{aligned} \text{maximize} \quad & x_1 - x_2 + x_3 \\ \text{subject to} \quad & 2x_1 - x_2 + 2x_3 \leq 4 \\ & 2x_1 - 3x_2 + x_3 \leq -5 \\ & -x_1 + x_2 - 2x_3 \leq -1 \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

To avoid unnecessary confusion, we write the auxiliary problem in its maximization form:

$$\begin{aligned} \text{maximize} \quad & -x_0 \\ \text{subject to} \quad & 2x_1 - x_2 + 2x_3 - x_0 \leq 4 \\ & 2x_1 - 3x_2 + x_3 - x_0 \leq -5 \\ & -x_1 + x_2 - 2x_3 - x_0 \leq -1 \\ & x_0, x_1, x_2, x_3 \geq 0. \end{aligned}$$

Writing down the formulas defining the slack variables x_4, x_5, x_6 and the objective function w, we obtain the dictionary

$$\begin{aligned} x_4 &= 4 - 2x_1 + x_2 - 2x_3 + x_0 \\ x_5 &= -5 - 2x_1 + 3x_2 - x_3 + x_0 \\ x_6 &= -1 + x_1 - x_2 + 2x_3 + x_0 \\ \hline w &= -x_0 \end{aligned}$$

which is infeasible. Nevertheless, this infeasible dictionary can be transformed into a feasible one by a single pivot, with x_0 entering and x_5 leaving the basis:

$$\begin{aligned} x_0 &= 5 + 2x_1 - 3x_2 + x_3 + x_5 \\ x_4 &= 9 - 2x_2 - x_3 + x_5 \\ x_6 &= 4 + 3x_1 - 4x_2 + 3x_3 + x_5 \\ \hline w &= -5 - 2x_1 + 3x_2 - x_3 - x_5. \end{aligned}$$

In general, the auxiliary problem may be written as

$$\begin{aligned} &\text{maximize} && -x_0 \\ &\text{subject to} && \sum_{j=1}^{n} a_{ij}x_j - x_0 \le b_i \qquad (i = 1, 2, \ldots, m) \\ & && x_j \ge 0 \qquad (j = 0, 1, \ldots, n). \end{aligned}$$

Writing down the formulas defining the slack variables $x_{n+1}, x_{n+2}, \ldots, x_{n+m}$ and the objective function w gives us the dictionary

$$\begin{aligned} x_{n+i} &= b_i - \sum_{j=1}^{n} a_{ij}x_j + x_0 \qquad (i = 1, 2, \ldots, m) \\ \hline w &= -x_0 \end{aligned}$$

which is infeasible. Nevertheless, this infeasible dictionary can be transformed into a feasible one by a single pivot, with x_0 entering and the "most infeasible" x_{n+i} leaving the basis. More precisely, the leaving variable is that x_{n+k} whose negative value, b_k, has the largest magnitude among all the negative numbers b_i. After pivoting, the variable x_0 assumes the positive value of $-b_k$, whereas each basic x_{n+i} assumes the nonnegative value of $b_i - b_k$. Now we are set to solve the auxiliary problem by the simplex method. In our illustrative example, the computations go as follows.

After the first iteration, with x_2 entering and x_6 leaving:

$$\begin{aligned} x_2 &= 1 + 0.75x_1 + 0.75x_3 + 0.25x_5 - 0.25x_6 \\ x_0 &= 2 - 0.25x_1 - 1.25x_3 + 0.25x_5 + 0.75x_6 \\ x_4 &= 7 - 1.5x_1 - 2.5x_3 + 0.5x_5 + 0.5x_6 \\ \hline w &= -2 + 0.25x_1 + 1.25x_3 - 0.25x_5 - 0.75x_6. \end{aligned}$$

After the second iteration, with x_3 entering and x_0 leaving:

$$\begin{aligned} x_3 &= 1.6 - 0.2x_1 + 0.2x_5 + 0.6x_6 - 0.8x_0 \\ x_2 &= 2.2 + 0.6x_1 + 0.4x_5 + 0.2x_6 - 0.6x_0 \\ x_4 &= 3 - x_1 - x_6 + 2x_0 \\ \hline w &= -x_0. \end{aligned} \tag{3.12}$$

The last dictionary (3.12) is optimal. Since the optimal value of the auxiliary problem is zero, dictionary (3.12) points out a feasible solution of the original problem: $x_1 = 0$, $x_2 = 2.2$, $x_3 = 1.6$. Furthermore, (3.12) can be easily converted into the desired feasible dictionary of the original problem. To obtain the first three rows of the desired dictionary, we simply copy down the first three rows of (3.12), omitting all the terms involving x_0:

$$\begin{aligned} x_3 &= 1.6 - 0.2x_1 + 0.2x_5 + 0.6x_6 \\ x_2 &= 2.2 + 0.6x_1 + 0.4x_5 + 0.2x_6 \\ x_4 &= 3 - x_1 - x_6. \end{aligned} \tag{3.13}$$

To obtain the last row, we have to express the original objective function

$$z = x_1 - x_2 + x_3 \tag{3.14}$$

in terms of the nonbasic variables x_1, x_5, x_6. For this purpose, we simply substitute from (3.13) into (3.14), obtaining

$$\begin{aligned} z &= x_1 - (2.2 + 0.6x_1 + 0.4x_5 + 0.2x_6) + (1.6 - 0.2x_1 + 0.2x_5 + 0.6x_6) \\ &= -0.6 + 0.2x_1 - 0.2x_5 + 0.4x_6. \end{aligned}$$

In short, the desired dictionary reads

$$\begin{aligned} x_3 &= 1.6 - 0.2x_1 + 0.2x_5 + 0.6x_6 \\ x_2 &= 2.2 + 0.6x_1 + 0.4x_5 + 0.2x_6 \\ x_4 &= 3 - x_1 - x_6 \\ \hline z &= -0.6 + 0.2x_1 - 0.2x_5 + 0.4x_6. \end{aligned}$$

Clearly, the same procedure will transform an optimal dictionary of the auxiliary problem into a feasible dictionary of the original problem whenever x_0 is nonbasic in the former.

Now, let us review the general situation. We have learned how to construct the auxiliary problem and its first feasible dictionary. In the process of solving the auxiliary problem, we may encounter a dictionary where x_0 competes with other variables for leaving the basis. If and when that happens, it is only natural to choose x_0 as the actual leaving variable; immediately after pivoting, we obtain a dictionary where

x_0 is nonbasic, and so the value of w is zero. (3.15)

Clearly, a feasible dictionary with this property is optimal. However, we may also reach the optimum of the auxiliary problem while x_0 is still basic. Thus, we may obtain an optimal dictionary where

x_0 is basic and the value of w is nonzero (3.16)

or, conceivably, an optimal dictionary where

x_0 is basic and the value of w is zero. (3.17)

Let us examine case (3.17). Since the next-to-last dictionary was not yet optimal, the value of $w = -x_0$ must have changed from some negative level to zero in the last iteration. To put it differently, the value of the basic variable x_0 must have dropped from some positive level to zero in the last iteration. But then x_0 was a candidate for leaving the basis; yet, contrary to our policy, we did not pick it. This contradiction shows that (3.17) cannot occur. Hence the optimal dictionary of the auxiliary problem has either property (3.15) or property (3.16). In the former case, we construct a feasible dictionary of the original problem as illustrated previously and proceed to solve the original problem by the simplex method; in the latter case, we simply conclude that the original problem is infeasible.

This strategy is known as the *two-phase simplex method*. In the *first phase*, we set up and solve the auxiliary problem; if the optimal dictionary turns out to have property (3.15) then we proceed to the *second phase*, solving the original problem itself. We shall return to the two-phase simplex method in Chapter 8.

□

THE FUNDAMENTAL THEOREM OF LINEAR PROGRAMMING

This name is given to the following result.

THEOREM 3.4. Every LP problem in the standard form has the following three properties:

(i) If it has no optimal solution, then it is either infeasible or unbounded.

(ii) If it has a feasible solution, then it has a basic feasible solution.

(iii) If it has an optimal solution, then it has a basic optimal solution.

PROOF. The first phase of the two-phase simplex method either discovers that the problem is infeasible or else it delivers a basic feasible solution. The second phase of the two-phase simplex method either discovers that the problem is unbounded or else it delivers a basic optimal solution. ■

Note that the first property is not shared by problems whose constraints may include *strict* linear inequalities $\sum a_j x_j < b$. To take a trivial example, the problem

maximize x subject to $x < 0$

is neither infeasible nor unbounded and yet it has no optimal solution. The remaining two properties (ii) and (iii) tell us that, when looking for feasible or optimal solutions of an LP problem in the standard form, we may confine our search to a *finite* set. These two properties, easy to establish from scratch, are often used to motivate the simplex method. Our exposition has followed the reverse pattern, with an emphasis placed on actually solving the problem—and the fundamental theorem of linear programming obtained as an effortless afterthought. □

PROBLEMS

△ **3.1** Maximize $x_1 + 3x_2 - x_3$

subject to
$$\begin{aligned} 2x_1 + 2x_2 - x_3 &\leq 10 \\ 3x_1 - 2x_2 + x_3 &\leq 10 \\ x_1 - 3x_2 + x_3 &\leq 10 \\ x_1, x_2, x_3 &\geq 0. \end{aligned}$$

3.2 In the tableau format, a natural tie-breaking rule for the choice of the pivot row favors the rows that appear higher up in the tableau. Show that in the following example (constructed by H. W. Kuhn), this tie-breaking rule leads to cycling:

maximize $2x_1 + 3x_2 - x_3 - 12x_4$

subject to
$$\begin{aligned} -2x_1 - 9x_2 + x_3 + 9x_4 &\leq 0 \\ \tfrac{1}{3}x_1 + x_2 - \tfrac{1}{3}x_3 - 2x_4 &\leq 0 \\ x_1, x_2, x_3, x_4 &\geq 0. \end{aligned}$$

3.3 Solve problem 3.2 by the perturbation technique.

3.4 Arrange the following expressions in a sequence from lexicographically smallest to lexicographically largest:

$3 - \varepsilon_1$
3
$2 + 10\varepsilon_1$
$3 - 4\varepsilon_1 + \varepsilon_2$
$\varepsilon_2 + 3\varepsilon_3$
$3 + 4\varepsilon_1 + \varepsilon_3$
$3 - 4\varepsilon_1 + \varepsilon_2 + \varepsilon_3$.

3.5 Prove: If $r = r_0 + r_1\varepsilon_1 + \cdots + r_m\varepsilon_m$ is lexicographically smaller than $s = s_0 + s_1\varepsilon_1 + \cdots + s_m\varepsilon_m$ and if s is lexicographically smaller than $t = t_0 + t_1\varepsilon_1 + \cdots + t_m\varepsilon_m$, then r is lexicographically smaller than t.

3.6 Use the result of problem 3.5 to prove that, in every finite set of distinct expressions, such as r and s in (3.5), there is an expression that is lexicographically smaller than all the others.

3.7 Prove that for every pair of expressions in (3.5) there is a positive number δ such that the following two statements are equivalent: (i) r is lexicographically smaller than s; (ii) for every choice of numbers $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ such that

$$0 < \varepsilon_1 < \delta \quad \text{and} \quad 0 < \varepsilon_i < \delta\varepsilon_{i-1} \qquad \text{for all } i = 2, 3, \ldots, m$$

r is numerically smaller than s.

3.8 Use Theorem 3.2 and the result of problem 3.7 to prove the following. For every LP problem

$$\text{maximize} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \le b_i \qquad (i = 1, 2, \ldots, m)$$

$$x_j \ge 0 \qquad (j = 1, 2, \ldots, n)$$

there is a positive number δ such that the simplex method used to

$$\text{maximize} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \le b_i + \varepsilon^i \qquad (i = 1, 2, \ldots, m)$$

$$x_j \ge 0 \qquad (j = 1, 2, \ldots, n)$$

terminates whenever $0 < \varepsilon < \delta$.

△ **3.9** Solve the following problems by the two-phase simplex method:

a. maximize $3x_1 + x_2$
subject to
$$\begin{aligned} x_1 - x_2 &\le -1 \\ -x_1 - x_2 &\le -3 \\ 2x_1 + x_2 &\le 4 \\ x_1, x_2 &\ge 0 \end{aligned}$$

b. maximize $3x_1 + x_2$
subject to
$$\begin{aligned} x_1 - x_2 &\le -1 \\ -x_1 - x_2 &\le -3 \\ 2x_1 + x_2 &\le 2 \\ x_1, x_2 &\ge 0 \end{aligned}$$

c. maximize $3x_1 + x_2$
subject to
$$\begin{aligned} x_1 - x_2 &\le -1 \\ -x_1 - x_2 &\le -3 \\ 2x_1 - x_2 &\le 2 \\ x_1, x_2 &\ge 0. \end{aligned}$$

3.10 Prove or disprove: A feasible dictionary whose last row reads $z = z^* + \sum \bar{c}_j x_j$ describes an optimal solution if and only if $\bar{c}_j \le 0$ for all j.

4

How Fast Is the Simplex Method?

The subject of this chapter is the number of iterations in the simplex method. We shall also comment on the distinction between theoretically satisfactory and practically satisfactory algorithms, with a particular regard to linear programming.

TYPICAL NUMBER OF ITERATIONS

For *practical* problems of the form

$$\text{maximize} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \le b_i \qquad (i = 1, 2, \ldots, m) \tag{4.1}$$

$$x_j \ge 0 \qquad (j = 1, 2, \ldots, n)$$

with $m < 50$ and $m + n < 200$, Dantzig (1963, p. 160) reported the number of iterations as being usually less than $3m/2$ and only rarely going to $3m$. This observation agrees with empirical findings obtained more recently for much larger problems: the