Efficiency of Simplex Method

- Worst case vs Average Case
- (Dantzig) $m < 50$, $m+n < 200$

$\#(\text{iterations}) \leq 3m/2$, (usually) rarely $\approx 3m$

[C] p.46 [C] p.50

TABLE 4.1 Average Number of Iterations Required by the Largest-Coefficient Rule

m \ n	10	20	30	40	50
10	9.40	14.2	17.4	19.4	20.2
20		25.2	30.7	38.0	41.5
30			44.4	52.7	62.9
40				67.6	78.7
50					95.2

Source: D. Avis and V. Chvátal (1978).

Table 4.2 Average Numbers of Iterations Required by the Largest-Increase Rule

m \ n	10	20	30	40	50
10	7.02	9.17	10.8	12.1	12.6
20		16.2	20.2	24.2	27.3
30			28.7	34.5	39.4
40				43.3	39.9
50					58.9

Source: D. Avis and V. Chvátal (1978).

Worst Case Example: Klee-Minty (1970)

[V] p. 46

The example is quite simple to state:

$$
\begin{aligned}
&\text{maximize} && \sum_{j=1}^{n} 2^{n-j} x_j \\
(4.1) \qquad &\text{subject to} && 2\sum_{j=1}^{i-1} 2^{i-j} x_j + x_i \le 100^{i-1} && i = 1, 2, \ldots, n \\
& && x_j \ge 0 && j = 1, 2, \ldots, n.
\end{aligned}
$$

It is instructive to look closely at the constraints. The first three constraints are

$$
\begin{cases}
x_1 \le 1 \\
4x_1 + x_2 \le 100 \\
8x_1 + 4x_2 + x_3 \le 10{,}000.
\end{cases}
$$

$\max \zeta = 4x_1 + 2x_2 + x_3$

Worst Case Example : Klee-Minty (1970)

[V] p.47

$$
\begin{aligned}
&\text{maximize} && \sum_{j=1}^{n} 2^{n-j} x_j - \frac{1}{2} \sum_{j=1}^{n} 2^{n-j} \beta_j \\
(4.2) \qquad &\text{subject to} && 2 \sum_{j=1}^{i-1} 2^{i-j} x_j + x_i \le \sum_{j=1}^{i-1} 2^{i-j} \beta_j + \beta_i \qquad i = 1, 2, \ldots, n \\
& && x_j \ge 0 \qquad\qquad\qquad\qquad\quad j = 1, 2, \ldots, n.
\end{aligned}
$$

$1 \ll \beta_1 \ll \beta_2 \ll \cdots \ll \beta_n$

Largest Coefficient Rule : require $2^n - 1$ steps

Smallest Coefficient Rule : just 1 steps.

Worst Case Example: Klee-Minty (1970)

[C] p.48

Using the largest-coefficient rule, we construct the following sequence of dictionaries. The initial dictionary:

$$
\begin{aligned}
x_4 &= 1 - x_1 \\
x_5 &= 100 - 20x_1 - x_2 \\
x_6 &= 10{,}000 - 200x_1 - 20x_2 - x_3 \\
\hline
z &= 100x_1 + 10x_2 + x_3.
\end{aligned}
$$

$2^3 - 1$ steps

In the first iteration, we were led to an unfortunate choice of the entering variable: had we made x_3 rather than x_1 enter the basis, we would have pivoted directly to the final dictionary. In view of this blunder, it is natural to question the expediency of the largest-coefficient rule: perhaps the simplex method would *always* go through only a small number of iterations if it were directed by some other rule. In fact, the largest-coefficient rule is not quite natural. More specifically, it ranks the potential candidates for entering the basis according to their coefficients in the last row of the dictionary: variables with larger coefficients appear to be more promising. But appearances are misleading and the ranking order is easily upset by changes in the scale on which each candidate is measured. For instance, the substitution

Worst Case Example: Klee-Minty (1970)

[CJ p. 48]

Using the largest-coefficient rule, we construct the following sequence of dictionaries. The initial dictionary:

$$\begin{aligned} x_4 &= 1 - x_1 \\ x_5 &= 100 - 20x_1 - x_2 \\ x_6 &= 10{,}000 - 200x_1 - 20x_2 - x_3 \\ \hline z &= 100x_1 + 10x_2 + x_3. \end{aligned}$$

$2^3 - 1$ steps

$$\bar{x}_1 = x_1, \quad \bar{x}_2 = 0.01x_2, \quad \bar{x}_3 = 0.0001x_3$$

converts the Klee–Minty problem (4.3) into the form

$$\begin{aligned} \text{maximize} \quad & 100\bar{x}_1 + 1{,}000\bar{x}_2 + 10{,}000\bar{x}_3 \\ \text{subject to} \quad & \bar{x}_1 \le 1 \\ & 20\bar{x}_1 + 100\bar{x}_2 \le 100 \\ & 200\bar{x}_1 + 2{,}000\bar{x}_2 + 10{,}000\bar{x}_3 \le 10{,}000 \\ & \bar{x}_1, \bar{x}_2, \bar{x}_3 \ge 0. \end{aligned}$$

In the first dictionary associated with this new version of (4.3), the nonbasic variable $\bar{x}_3$ appears most attractive, and so the simplex method reaches the optimal solution in only one iteration.

Other Pivoting Rules (independent of scale)

[V] p.49

ALTERNATIVE PIVOTING RULES

Thus we are led to ranking the candidates x_j for entering the basis according to criteria that are independent of changes of scale. One criterion of this kind is the increase in the objective function obtained when x_j actually enters the basis. The resulting rule (always choose that candidate whose entrance into the basis brings about the largest increase in the objective function) is referred to as the *largest-increase* rule. On the Klee–Minty examples (4.2), the largest-increase rule leads the simplex method to the optimal solution in only one iteration, as opposed to the $2^n - 1$ iterations required by the previously used largest-coefficient rule. However, the new rule does not always lead to a small number of iterations: R. G. Jeroslow (1973) constructed LP problems that are to the largest-increase rule what the Klee–Minty problems are to the largest-coefficient rule. (More precisely, the number of iterations required by the largest-increase rule grows exponentially with m and n.) Again, these examples exploit the myopia inherent in the simplex method. It is conceivable that every easily implemented rule for choosing the entering variable can be tricked in a similar way into requiring very large numbers of iterations.

Other Pivoting Rules (independent of scale)

Which of the two rules is better? On problems arising from applications, the number of iterations required by the largest increase is *usually* smaller than the number of iterations required by the largest coefficient. Simulation experiments lead to a similar outcome (see Table 4.2).

TABLE 4.1 Average Number of Iterations Required by the Largest-Coefficient Rule

m \ n	10	20	30	40	50
10	9.40	14.2	17.4	19.4	20.2
20		25.2	30.7	38.0	41.5
30			44.4	52.7	62.9
40				67.6	78.7
50					95.2

Source: D. Avis and V. Chvátal (1978).

Table 4.2 Average Numbers of Iterations Required by the Largest-Increase Rule

m \ n	10	20	30	40	50
10	7.02	9.17	10.8	12.1	12.6
20		16.2	20.2	24.2	27.3
30			28.7	34.5	39.4
40				43.3	39.9
50					58.9

Source: D. Avis and V. Chvátal (1978).

Other Pivoting Rules (independent of scale)

Which of the two rules is better? On problems arising from applications, the number of iterations required by the largest increase is *usually* smaller than the number of iterations required by the largest coefficient. Simulation experiments lead to a similar outcome (see Table 4.2).

Nevertheless, as the largest-coefficient rule takes less time to execute than the largest increase, it is the former that usually wins in terms of total computing time. More generally, the number of iterations is a poor criterion for assessing the efficiency of a rule for choosing the entering variable. It is the total computing time that counts, and rules that tend to reduce the number of iterations often take too much time to execute. In this light, even the largest-coefficient rule is found too time-consuming and therefore rarely, if ever, used in practice. The choice of entering variables in efficient implementations of the simplex method is influenced by the logistics of handling large problems on a computer; this matter will be studied in Chapter 7.

5.7 Pivot Rules

[MG] p.71

A **pivot rule** is a rule for selecting the entering variable if there are several possibilities, which is usually the case. Sometimes there may also be more than one possibility for choosing the leaving variable, and some pivot rules specify this choice as well, but this part is typically not so important.

The number of pivot steps needed for solving a linear program depends substantially on the pivot rule. (See the example in Section 5.1.) The problem is, of course, that we do not know in advance which choices will be good in the long run.

Here we list some of the common pivot rules. By an "improving variable" we mean any nonbasic variable with a positive coefficient in the z-row of the simplex tableau, in other words, a candidate for the entering variable.

LARGEST COEFFICIENT. Choose an improving variable with the largest coefficient in the row of the objective function z. This is the original rule, suggested by Dantzig, that maximizes the improvement of z *per unit increase* of the entering variable.

LARGEST INCREASE. Choose an improving variable that leads to the largest *absolute* improvement in z. This rule is computationally more expensive than the LARGEST COEFFICIENT rule, but it locally maximizes the progress.

STEEPEST EDGE. Choose an improving variable whose entering into the basis moves the current basic feasible solution in a direction closest to the direction of the vector $\mathbf{c}$. Written by a formula, the ratio

$$\frac{\mathbf{c}^T(\mathbf{x}_{\text{new}} - \mathbf{x}_{\text{old}})}{\|\mathbf{x}_{\text{new}} - \mathbf{x}_{\text{old}}\|}$$

should be maximized, where $\mathbf{x}_{\text{old}}$ is the basic feasible solution for the current simplex tableau and $\mathbf{x}_{\text{new}}$ is the basic feasible solution for the tableau that would be obtained by entering the considered improving variable into the basis. (We recall that $\|\mathbf{v}\| = (v_1^2 + v_2^2 + \cdots + v_n^2)^{1/2} = \sqrt{\mathbf{v}^T\mathbf{v}}$ denotes the Euclidean length of the vector $\mathbf{v}$, and the expression $\mathbf{u}^T\mathbf{v}/(\|\mathbf{u}\| \cdot \|\mathbf{v}\|)$ is the cosine of the angle of the vectors $\mathbf{u}$ and $\mathbf{v}$.)

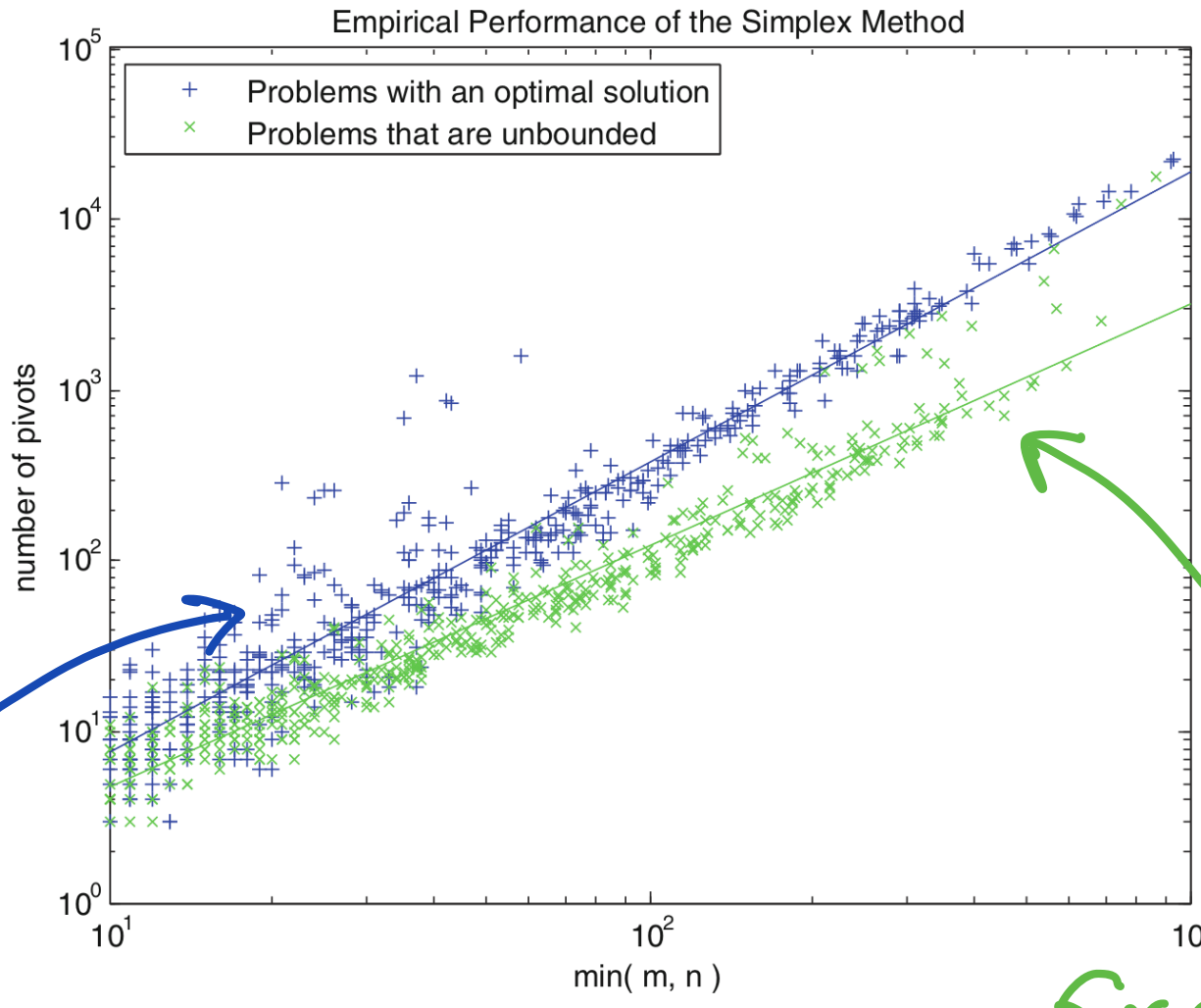
The STEEPEST EDGE rule is a champion among pivot rules in practice. According to extensive computational studies it is usually faster than all other pivot rules described here and many others. An efficient approximate implementation of this rule is discussed in the glossary under the heading "Devex."

BLAND'S RULE. Choose the improving variable with the smallest index, and if there are several possibilities for the leaving variable, also take the one with the smallest index. BLAND'S RULE is theoretically very significant since it prevents cycling, as we will discuss in Section 5.8.

RANDOM EDGE. Select the entering variable uniformly at random among all improving variables. This is the simplest example of a *randomized pivot rule*, where the choice of the entering variable uses random numbers in some way. Randomized rules are also very important theoretically, since they lead to the current best provable bounds for the number of pivot steps of the simplex method.

Some Empirical Studies

[V]



Blue dots

$\# \approx 0.15 \min(m,n)^{1.70}$

Green dots

$\# \approx 1.80 \min(m,n)^{1.42}$

Some Theoretical Results

[MG] p. 57

Later on, very slow examples of a similar type were discovered for many other pivot rules, among them all the rules mentioned above. Many people have tried to design a pivot rule and prove that the number of pivot steps is always bounded by some polynomial function of m and n, but nobody has succeeded so far. The best known bound has been proved for the following simple randomized pivot rule: Choose a random ordering of the variables at the beginning of the computation (in other words, randomly permute the indices of the variables in the input linear program); then use Bland's rule for choosing the entering variable, and the lexicographic method for choosing the leaving variable. For every linear program with at most n variables and at most n constraints, the expected number of pivot steps is bounded by $e^{C\sqrt{n \ln n}}$, where C is a (not too large) constant. (Here the expectation means the arithmetic average over all possible orderings of the variables.) This bound is considerably better than 2^n, say, but much worse than a polynomial bound.

This algorithm was found independently and almost at the same time by Kalai and by Matoušek, Sharir, and Welzl. For a recent treatment in a somewhat broader context see

B. Gärtner and E. Welzl: Explicit and implicit enforcing—randomized optimization, in *Lectures of the Graduate Program*

Some Theoretical Results

[MG] p.78

In spite of the Klee–Minty cube and similar artificial examples, the simplex method is being used successfully. Remarkable theoretical results indicate that these willful examples are rare indeed. For instance, it is known that if a linear program in equational form is generated in a suitable (precisely defined) way at random, then the number of pivot steps is of order at most m^2 with high probability. More recent results, in the general framework of the so-called *smoothed complexity*, claim that if we take an arbitrary linear program and then we change its coefficients by small random amounts, then the simplex method with a certain pivot rule reaches the optimum of the resulting linear program by polynomially many steps with high probability (a concrete bound on the polynomial depends on a precise specification of the "small random amounts" of change). The first theorem of this kind is due to Spielman and Teng, and for recent progress see

> R. Vershynin: Beyond Hirsch conjecture: Walks on random polytopes and the smoothed complexity of the simplex method, preprint, 2006.

Some Theoretical Results

Polynomial bounded algorithms for Linear Programming does exist!

Interior Method which goes through the interior of the feasible set.

(Simplex Method goes through the boundary of the feasible set.)