Mathematical Programming Study 8 (1978) 24–34.
North-Holland Publishing Company

NOTES ON BLAND'S PIVOTING RULE*

D. AVIS

McGill University, Montréal, Québec, Canada

V. CHVÁTAL

Université de Montréal, Montréal, Québec, Canada

Received 24 January 1977
Revised manuscript received 8 May 1978

Dedicated to the memory of D. Ray Fulkerson

Recently R.G. Bland proposed two new rules for pivot selection in the simplex method. These elegant rules arise from Bland's work on oriented matroids; their virtue is that they never lead to cycling. We investigate the efficiency of the first of them. On randomly generated problems with 50 nonnegative variables and 50 additional inequalities, Bland's rule requires about 400 iterations on the average; the corresponding figure for the popular "largest coefficient" rule is only about 100. Comparable behaviour seems to persist even on highly degenerate problems. On the theoretical side, we analyse the performance of Bland's rule on the classical Klee–Minty examples: for problems with n nonnegative variables and n additional inequalities, the number of iterations is bounded from below by the n-th Fibonacci number.

Key words: Simplex Method, Pivoting, Number of Iterations, Degeneracy and Cycling, Monte Carlo Experiments, Klee–Minty Examples.

1. Introduction

The commonly used pivoting rules for the simplex method can lead to cycling: examples of that phenomenon have been constructed by Hoffman [5], Beale [1] and others. Although cycling is virtually unknown in practice, its potential threat is unpleasant from the theoretical point of view. Fortunately, it can be avoided by precautions such as the perturbation technique [3] or, equivalently, the lexicographic technique [4]. Quite recently, R.G. Bland [2] dealt cycling an impressive *coup de grâce* by presenting an elegant proof that a certain simple and natural pivoting rule never leads to cycling. Bland's rule is extremely simple indeed:

(i) among all the candidates to enter the basis, select the variable with the smallest subscript,

(ii) if two or more variables compete for leaving the basis, select again the variable with the smallest subscript.

The importance of this "smallest subscript" rule is primarily theoretical: it was Bland's work on oriented matroids that led him to the discovery of the rule. (In

* This research was supported by NRC grant A9211.

fact, [2] contains *two* different pivoting rules that never lead to cycling: the second of them is slightly more complicated and we shall not discuss it here.) It seems likely that Bland's pivoting rules may find applications in other theoretical studies involving the simplex method.

Even though the smallest subscript rule was not meant to compete with other pivoting rules in actual computations, it makes sense to ask how its performance compares with those of the commonly used pivoting rules. (On the intuitive level, it may seem that the odds are stacked against the smallest subscript rule since that rule ignores much of the valuable information contained in a simplex tableau: taking into account only the *signs* of the coefficients in the objective row, it disregards their *magnitudes*. In our opinion, such feeling does not carry much weight unless confirmed experimentally: one might as well argue that the smallest subscript rule should have the edge because of its highly systematic nature. After all, doesn't it have the edge where cycling is concerned?) A satisfactory approximation to the precise answer can be provided by Monte Carlo experiments. In the early Sixties, Kuhn and Quandt [8] carried out such experiments to compare the performances of nine different pivoting rules. Following their example, we have compared the smallest subscript rule with two of the most popular pivoting rules:

(α) among all the candidates to enter the basis, select that variable whose "relative cost" has the largest absolute value,

(β) among all the candidates to enter the basis, select that variable whose entrance will bring about the largest increase of the objective function. The results are reported in the next section: the smallest subscript rule (γ) did not fare too well. For problems with fifty nonnegative variables and fifty additional inequality constraints, it required close to four hundred iterations on the average; the average number of iterations required by (α) to solve the same problems was less than one hundred. Experimenting with some highly degenerate problems, we were surprised to find that (α) was still better than (γ). In fact, the ratios of the corresponding numbers of iterations were quite close to those for nondegenerate problems of comparable sizes.

Another question of possible interest is theoretical: could it be that the number of iterations required by (γ) is bounded from above by a fixed polynomial in the size of the problem? For the pivoting rule (α), the same question was answered negatively in the pioneering paper [7] by Klee and Minty; later on, a similar result for (β) was established by Jeroslow [6]. Klee and Minty begin their exposition by presenting very simple problems with n nonnegative variables and n additional inequalities for which the simplex method may take $2^n - 1$ iterations if the entering variables are chosen in a consistently ill-advised way. We shall prove that the number of iterations required by (γ) to solve these problems is bounded from below by the n-th Fibonacci number. The same holds for the completely degenerate problems resulting from the original examples when each right-hand side is replaced by a zero.

2. The experiments

There are many different ways of formalizing the intuitive concept of a random linear programming problem. It seems that each of them is open to some criticism; furthermore, it is questionable whether the LP problems solved in practice are "random" in some sense. For the purpose of our Monte Carlo experiments, we adopted the format used by Kuhn and Quandt [8] and we have considered problems of the form

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^{n} x_j, \\ &\text{subject to} && \sum_{j=1}^{n} a_{ij}x_j \leq 10^4 \quad (i = 1, 2, \ldots, m), \\ &&& x_j \geq 0 \qquad (j = 1, 2, \ldots, n) \end{aligned}$$

with m and n coming from $\{10, 20, \ldots, 50\}$ and such that $m \leq n$. For each of the fifteen combinations of m and n, we have generated 100 problems by selecting the coefficients a_{ij} at random from the set $\{1, 2, \ldots, 1000\}$. Each of these problems was solved by the revised simplex method three times over: with (α), with (β) and with Bland's rule (γ). The mean numbers M_I of iterations per problem and their standard deviations σ_I (in parentheses) are shown in Table 1. The mean times M_T per problem and their standard deviations σ_T appear in Table 2. The computations where done on Control Data Cyber 70 Model 74 at Université de Montréal.

These results may be compared with those reported by Kuhn and Quandt [8] when the problem sizes overlap (10 × 10 and 20 × 20). Our values of M_I agree to within 5% of the values given there for methods α and β. Although the values of M_T are not directly comparable, it appears that method β performs worse in our study. The reason for this is that we used the revised simplex method whereas the standard simplex method is used in [8].

Table 1
Mean iterations (M_I); standard deviation (σ_I) in parentheses.

n m	10	20	30	40	50
10	9.40 (2.60)	14.2 (3.8)	17.4 (3.93)	19.4 (4.36)	20.2 (4.82)
20		25.2 (6.12)	30.7 (8.55)	38.0 (8.90)	41.5 (9.62)
30			44.4 (10.5)	52.7 (13.0)	62.9 (14.7)
40				67.6 (14.4)	78.7 (18.2)
50					95.2 (17.0)

Method α

m \ *n*	10	20	30	40	50
10	7.02	9.17	10.8	12.1	12.6
	(1.90)	(2.72)	(2.87)	(3.24)	(3.04)
20		16.2	20.24	24.2	27.3
		(3.55)	(4.03)	(4.65)	(5.97)
30			28.7	34.5	39.4
			(4.80)	(6.17)	(7.83)
40				43.3	39.9
				(7.30)	(8.92)
50					58.9
					(9.35)

Method β

m	10	20	30	40	50
10	15.0	31.1	42.3	51.0	58.7
	(5.51)	(10.4)	(13.3)	(15.8)	(14.8)
20		56.1	85.6	122	154
		(15.4)	(23.7)	(29.1)	(33.6)
30			128	182	238
			(30.3)	(42.9)	(55.7)
40				242	316
				(44.6)	(68.1)
50					391
					(76.3)

Method γ

Table 2
Mean time (M_T); standard deviation (σ_T) in parentheses

m \ *n*	10	20	30	40	50
10	0.035	0.064	0.083	0.112	0.134
	(0.019)	(0.032)	(0.035)	(0.035)	(0.033)
20		0.271	0.363	0.445	0.500
		(0.066)	(0.106)	(0.102)	(0.150)
30			0.871	1.20	1.49
			(0.219)	(0.305)	(0.362)
40				2.29	2.91
				(0.482)	(0.677)
50					5.20
					(0.924)

Method α

n m	10	20	30	40	50
10	0.056	0.108	0.165	0.222	0.270
	(0.023)	(0.040)	(0.052)	(0.061)	(0.070)
20		0.498	0.828	1.07	1.37
		(0.112)	(0.188)	(0.227)	(0.327)
30			1.99	3.33	4.48
			(0.351)	(0.600)	(0.947)
40				6.39	9.18
				(1.14)	(1.73)
50					16.70
					(2.79)

Method β

n m	10	20	30	40	50
10	0.048	0.097	0.141	0.180	0.215
	(0.026)	(0.036)	(0.048)	(0.060)	(0.054)
20		0.458	0.719	0.948	1.19
		(0.122)	(0.203)	(0.215)	(0.273)
30			1.88	2.94	3.80
			(0.445)	(0.684)	(0.898)
40				6.10	8.16
				(1.15)	(1.75)
50					15.66
					(3.03)

Method γ

In many practical LP problems, the coefficient matrices are sparse. In order to provide at least a crude approximation of this situation, we also considered random LP problems in which the coefficients were set to zero with probability 0.75. (Needless to point out, the resulting problems lacked the special structure found in sparse problems occurring in practice.) We ran 100 such problems of sizes 10 × 10 and 10 × 50; the smallest subscript rule performed a little better than in the original tests, although it was still inferior to (α).

In addition, we considered LP problems of the form

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

To solve these problems, we used a modified version of the smallest subscript rule in which the variables were first renumbered to satisfy the inequalities

$c_1 \geq c_2 \geq \cdots \geq c_n$. On 100 test problems of sizes $m = n = 10$ and $m = 10$, $n = 50$, the smallest subscript rule actually had slightly lower values of M_T than either of (α) and (β). (The value of M_T for method (γ) included the time required to renumber the variables.) Similar "hybrid" pivoting rules, combining the practical virtues of (α) with the theoretical virtues of (γ), might be worth exploring.

Since cycling cannot occur in nondegenerate problems, there is hardly any reason to use (γ) in the absence of degeneracy. Even for degenerate problems cycling is extremely rare: Wolfe [9] seems to be the only source reporting a *practical* problem that cycled. However, degeneracy often indicates the advent of another unpleasantness: for highly degenerate problems, the simplex method may stay at the same solution for many iterations, changing only the bases. This phenomenon may be called *stalling*; since the smallest subscript rule provides an effective cure for cycling, one might hope that it will also reduce stalling. Led by this sentiment, we experimented with some highly degenerate problems as described below. To our surprise, (α) still turned out to be superior to (γ).

Of all the highly degenerate problems that come up in practice, the assignment problem is perhaps the most notorious. We solved 100 assignment problems of size 5×5 using the standard simplex code (the constraint matrix has size 10×25). The cost coefficients were drawn at random from the set $\{0, 1, 2, \ldots, 100\}$; the average number of iterations required by (γ) was 28.8 as opposed to only 8.58 required by (α). For 10 assignment problems of size 10×10, the corresponding numbers were 98.7 and 19.3.

Finally, we considered problems of the form

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

In solving these problems, the simplex method remains stalled at each iteration except possibly the final one. Indeed, (1) has feasible origin; if it has any other solution then the entire ray defined by this solution is feasible. The practical significance of (1) stems from solving the system of inequalities

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

since (1) is, in a sense, dual to (2). We generated 100 problems (1) of size $m = n = 10$ with coefficients a_{ij}, b_i selected at random and independently of each other from the set $\{-500, -499, \ldots, 500\}$. On average, ($\gamma$) required 15.48 iterations whereas (α) required only 7.88 iterations. For 10 problems of size $m = n = 20$, the numbers were 63.0 and 32.3.

From the computational point of view, one further comment is in order. Due to round-off errors, the numerical value zero may be stored in the computer as a nonzero number of a very small magnitude. Therefore a coefficient must be considered negative if and only if it is less than some fixed small negative δ. We found that the number of iterations depended on the value of δ and finally settled on $\delta = 10^{-6}$.

3. The example

Denoting by ϵ a real number such that $0 < \epsilon < \frac{1}{2}$, we shall consider the following LP problem.

$$\begin{aligned} &\text{maximize} \quad \sum_{j=1}^{n} \epsilon^{n-j} x_j, \\ &\text{subject to} \quad 2\sum_{j=1}^{i-1} \epsilon^{i-j} x_j + x_i + x_{n+i} = 1 \quad (i = 1, 2, \dots, n), \\ &\qquad\qquad\qquad\qquad\qquad\quad x_j \geq 0 \quad (j = 1, 2, \dots, 2n). \end{aligned} \tag{3}$$

(The reader acquainted with the paper [7] by Klee and Minty will recognize in (3) a disguised version of their example of Section 4.) Our aim is to show that the simplex method with Bland's pivoting rule, initialized by $x_1 = x_2 = \cdots = x_n = 0$, takes a very large number of iterations to arrive at the optimal solution.

It will be convenient to present first the purely combinatorial part of our argument. For the moment, let n be a positive integer and let Q denote the set of all zero-one sequences of length n. For every two different sequences $a_1 a_2 \dots a_n$ and $b_1 b_2 \dots b_n$ in Q, there is the largest subscript k such that $a_k \neq b_k$; if $\sum_{i=k}^{n} a_i$ is even, then we shall write $a_1 a_2 \dots a_n < b_1 b_2 \dots b_n$. It is easy to verify that $<$ is a linear order on Q and that the last sequence in that order is $000 \dots 01$. For every other sequence $a_1 a_2 \dots a_n$ there is the smallest subscript k such that $\sum_{i=k}^{n} a_i$ is even; it is again easy to verify that $a_1 a_2 \dots a_n$ is related to its immediate successor $b_1 b_2 \dots b_n$ by the formula

$$b_i = \begin{cases} a_i & \text{for } i \neq k, \\ 1 - a_i & \text{for } i = k. \end{cases} \tag{4}$$

For every sequence $a_1 a_2 \dots a_n$ other than $000 \dots 01$ we shall define a sequence $f(a_1 a_2 \dots a_n) = b_1 b_2 \dots b_n$ as follows. If there is a subscript k such that $a_k = 0$ and $\sum_{i=k}^{n} a_i$ is even, then take the smallest subscript with that property and define $b_1 b_2 \dots b_n$ by (4); otherwise let $b_1 b_2 \dots b_n$ be the immediate successor of $a_1 a_2 \dots a_n$. Clearly,

$$a_1 a_2 \dots a_n < f(a_1 a_2 \dots a_n) \tag{5}$$

whenever the right-hand side is defined. It follows that there is a positive integer

$t = t(n)$ such that

$$f^t(111\ldots1111) = 000\ldots0001.$$

We are about to observe that

$$f^{t(n-2)}(111\ldots1111) = 000\ldots0111 \tag{6}$$

and that

$$f^{t(n-1)}(111\ldots1101) = 000\ldots0001. \tag{7}$$

Indeed, if $a_{n-1} = a_n = 1$ but $a_1a_2\ldots a_n \neq 000\ldots0111$, then $f(a_1a_2\ldots a_n)$ can be obtained by adding 11 at the end of $f(a_1a_2\ldots a_{n-2})$. Similarly, if $a_{n-1} = 0$, $a_n = 1$ but $a_1a_2\ldots a_n \neq 000\ldots0001$, then $f(a_1a_2\ldots a_n)$ can be obtained by slipping a zero before the last digit of $f(a_1a_2\ldots a_{n-2}a_n)$. Combining (6) and (7) with the easy observations that

$$f(000\ldots0111) = 000\ldots0101$$

and

$$f^{n-3}(000\ldots0101) = 111\ldots1101$$

We conclude that

$$t(n) = t(n-1) + t(n-2) + n - 2$$

whenever $n \geq 3$. Now let $s = s(n)$ be the positive integer such that

$$f^s(000\ldots0000) = 000\ldots0001.$$

Since

$$f^n(000\ldots0000) = 111\ldots1111,$$

we have $s(n) = n + t(n)$ and so $s(n) = s(n-1) + s(n-2) - 1$ whenever $n \geq 3$. Note that $s(1) = 1$, $s(2) = 3$ and so $s(n)$ is bounded from below by the n-th Fibonacci number. More precisely,

$$s(n) = \frac{2}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \frac{2}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^{n+1} - 1.$$

(We are indebted to R.G. Bland for a substantial simplification of our original proof.)

Let us return to our example (3) in its inequality form,

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^{n} \epsilon^{n-j} x_j, \\ &\text{subject to} && 2\sum_{j=1}^{i-1} \epsilon^{i-j} x_j + x_i \leq 1 \qquad (i = 1, 2, \ldots, n), \\ &&& x_j \geq 0 \qquad (j = 1, 2, \ldots, n). \end{aligned} \tag{8}$$

It will be convenient to denote the objective function in (8) by z_n and to denote the feasibility region in (8) by P_n. If x_n is fixed at zero then (8) reduces to

$$\text{maximize } \epsilon z_{n-1} \text{ over } P_{n-1};$$

if x_{2n}, defined as in (3), is fixed at zero then (8) reduces to

$$\text{maximize } 1-\epsilon z_{n-1} \text{ over } P_{n-1}.$$

We conclude that $z_n < \frac{1}{2}$ whenever $x_n = 0$ and that $z_n > \frac{1}{2}$ whenever $x_{2n} = 0$. An easy extension of this argument yields the following facts whose verifications are left to the reader.

(*) The polytope P_n is combinatorially equivalent to the n-dimensional cube. In fact, there is a natural one-to-one correspondence between the vertices $(x_1, x_2, \dots, x_n)$ of P_n and zero-one sequences $a_1a_2 \dots a_n$ of length n, defined by

$$\begin{aligned} a_j &= 0 \quad \text{if } x_j = 0,\\ &= 1 \quad \text{if } x_{n+j} = 0. \end{aligned}$$

(*) If $a_1a_2 \dots a_n < b_1b_2 \dots b_n$, then the value of z_n at the vertex corresponding to $a_1a_2 \dots a_n$ is strictly smaller than the value of z_n at the vertex corresponding to $b_1b_2 \dots b_n$.

(*) Bland's pivoting rule directs the simplex method from each vertex corresponding to $a_1a_2 \dots a_n$ to the vertex corresponding to $f(a_1a_2 \dots a_n)$.

(*) The simplex method with Bland's pivoting rule, initialized by $x_1 = x_2 = \dots = x_n = 0$, takes $s(n)$ iterations to arrive at the optimal solution. This is the desired conclusion.

Finally, we shall turn to theoretical questions that involve stalling. For each pivoting rule R, we may define $g(R, m, n)$ as the smallest integer such that the simplex method directed by R stalls for at most $g(R, m, n)$ iterations on problems with m equations and n nonnegative variables. Note that every pivoting rule R can be used to solve LP problems

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

by solving the highly degenerate problems

$$\begin{aligned} &\text{maximize} && t, \\ &\text{subject to} && \sum_{j=1}^{n} a_{ij}x_j - b_it \le 0 \quad (i = 1, 2, \dots, m), \\ &&& -\sum_{i=1}^{m} a_{ij}y_i + c_jt \le 0 \quad (j = 1, 2, \dots, n), \\ &&& -\sum_{j=1}^{n} c_jx_j + \sum_{i=1}^{m} b_iy_i \le 0, \\ &&& x_j \ge 0 \quad (j = 1, 2, \dots, n), \\ &&& y_i \ge 0 \quad (i = 1, 2, \dots, m), \\ &&& t \ge 0 \end{aligned}$$

in at most $g(R, m+n+1, 2m+2n+2)$ iterations. Hence it would be quite exciting to discover a pivoting rule R for which $g(R, m, n)$ is bounded from above by a polynomial in m and n. We are going to show that

$$g((\gamma), n, 2n) \geq s(n).$$

Consider the LP problem formed by replacing the right-hand side of (3) by zeroes.

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^{n} \epsilon^{n-j} x_j, \\ &\text{subject to} && 2\sum_{j=1}^{i-1} \epsilon^{i-j} x_j + x_i + x_{n+i} = 0 \quad (i = 1, 2, \dots, n), \\ &&& x_j \geq 0 \quad (j = 1, 2, \dots, 2n). \end{aligned} \tag{9}$$

This problem is completely degenerate. We will see that the smallest subscript rule generates the same sequence of pivots in (9) as in (3).

It is easy to check that a feasible basis for (3) is also a feasible basis for (9). (The converse is false.) We assume that the simplex method has been initiated on (3) and (9) with the starting basis $\{x_{n+1}, x_{n+2}, \dots, x_{2n}\}$. Inductively, we assume that at the start of iterations $0, 1, \dots, k$ the bases of (3) and (9) coincide. Consider the $(k+1)$-st pivot step. Both problems have the same reduced cost coefficients, so the entering variable, say x_r, is the same in both cases. Let $\bar{a}_{ij}$ $(i = 1, 2, \dots, n;$ $j = 1, 2, \dots, n)$ denote the coefficients in the current simplex tableau. As we saw above, at each iteration x_i is a basic variable if and only if x_{n+i} is a non-basic variable $(i = 1, 2, \dots, n)$. Therefore the coefficients in the tableau satisfy

$$\begin{aligned} &\bar{a}_{ij} = \bar{a}_{i,n+j} = 0 \quad (i = 1, 2, \dots, n, j = 1, 2, \dots, n, i < j); \\ &\bar{a}_{ii} = \bar{a}_{i,n+i} = 1 \quad (i = 1, 2, \dots n). \end{aligned}$$

Thus the first candidate to leave the basis will be the same in both problems. This completes the induction.

References

[1] E.M.L. Beale, "Cycling in the dual simplex algorithm", *Naval Research Logistics Quarterly* 2 (1955) 269–276.

[2] R.G. Bland, "New finite pivoting rules for the simplex method", *Mathematics of Operations Research* 2 (1977) 103–107.

[3] A. Charnes, "Optimality and degeneracy in linear programming", *Econometrica* 20 (1952) 160–170.

[4] G.B. Dantzig, A. Orden and P. Wolfe, "Generalized simplex method for minimizing a linear form under linear inequality constraints", *Pacific Journal of Mathematics* 5 (1955) 183–195.

[5] A.J. Hoffman, "Cycling in the simplex algorithm", National Bureau of Standards Report 2974 (December 1953).

[6] R.G. Jeroslow, "The simplex algorithm with the pivot rule of maximizing improvement criterion", *Discrete Mathematics* 4 (1973) 367–377.

[7] V. Klee and G.J. Minty, "How good is the simplex algorithm?", in: O. Shisha, ed., *Inequalities-III* (Academic Press, New York, 1972) pp. 159–175.

[8] H. Kuhn and R.E. Quandt, "An experimental study of the simplex method", in: N.C. Metropolis et al., eds., *Experimental arithmetic, high-speed computing and mathematics,* Proceedings of Symposia on Applied Mathematics XV (Am. Math. Soe., Providence, RI, 1963) pp. 107–124.

[9] P. Wolfe, "A technique for resolving degeneracy in linear programming", *Journal of SIAM* 11 (1963) 205–211.