since A is Noetherian (see [**1**, Exercise 7, p. 126]). Thus the height of M is exactly 2, as required by (i). The proof of the theorem is complete. ■

ACKNOWLEDGMENTS. I wish to express my warm gratitude to Professor Ram Murty for an interesting discussion that led to the idea of writing this paper.

REFERENCE

1. M. F. Atiyah and I. G. Macdonald, *Introduction to Commutative Algebra*, Addison-Wesley, Reading, MA, 1969.

Fabrizio Zanello, Department of Mathematics and Statistics, Queen's University, Kingston, Ontario, K7L 3N6 Canada
fabrizio@mast.queensu.ca

Solving Inequalities and Proving Farkas's Lemma Made Easy

David Avis and Bohdan Kaluzny

1. INTRODUCTION. Every college student has learned how to solve a system of linear equations, but how many would know how to solve $Ax \leq b$ for $x \geq 0$ or show that there is no solution? Solving a system of linear inequalities has traditionally been taught only in higher level courses and is given an incomplete treatment in introductory linear algebra courses. For example, the text of Strang [**4**] presents linear programming and states Farkas's lemma. It does not, however, include any proof of the finiteness of the simplex method or a proof of the lemma. Recent developments have changed the situation dramatically. Refinements of the simplex method by Bland [**1**] in the 1970s led to simpler proofs of its finiteness, and Bland's original proof was simplified further by several authors. In this paper we use a variant of Bland's pivot rule to solve a system of inequalities directly, without any need for introducing linear programming. We give a simple proof of the finiteness of the method, based on ideas contained in the paper of Fukuda and Terlaky [**3**] on the related criss-cross method. Finally, if the system is infeasible, we show how the termination condition of the algorithm gives a certificate of infeasibility, thus proving the Farkas lemma. Terminology and notation used here follows that of Chvátal's linear programming book [**2**].

We consider the following problem: given a matrix $A = [a_{ij}]$ in $R^{m\times n}$ and a column vector b in R^m, find $x = (x_1, x_2, \dots, x_n)^T$ that satisfies the following linear system, or prove that no such vector x exists:

$$Ax \leq b, \qquad x \geq 0. \tag{1}$$

We illustrate a simple method for doing this with an example:

$$\begin{aligned} -x_1 - 2x_2 + x_3 &\leq -1 \\ x_1 - 3x_2 - x_3 &\leq 2 \\ -x_1 - 2x_2 + 2x_3 &\leq -2 \end{aligned} \tag{2}$$

with $x_i \geq 0$ ($i = 1, 2, 3$). We first convert this system of inequalities into a system of equations by introducing a new nonnegative *slack* variable for each inequality. This slack variable represents the difference between the right- and left-hand sides of the inequality. In our example, we need three new variables, which we label x_4, x_5, and x_6. Putting these variables on the left-hand side, and the others on the right-hand side we have the following system:

$$\begin{aligned} x_4 &= -1 + x_1 + 2x_2 - x_3 \\ x_5 &= 2 - x_1 + 3x_2 + x_3 \\ x_6 &= -2 + x_1 + 2x_2 - 2x_3 \end{aligned} \tag{3}$$

It is easy to see that any nonnegative solution of (2) then extends to a nonnegative solution of (3) by assigning the slack variables values via their respective equations. Conversely, a nonnegative solution of (3) when restricted to x_1, x_2, and x_3 gives a solution to (2). We call a system of equations such as (3) a *dictionary*. The variables on the left-hand side are called *basic*, and the variables on the right-hand side are called *cobasic*. We get a *basic solution* to the equations in (3) by setting all the cobasic variables to zero, which gives $x_4 = -1$, $x_5 = 2$, $x_6 = -2$. Unfortunately this is not a nonnegative solution. The algorithm proceeds as follows: it finds the smallest-indexed basic variable that is set to a negative value. In this case it is x_4. In the equation for x_4 it identifies the cobasic variable with the smallest index that has a positive coefficient (in this case it is x_1), solves this equation for x_1, and substitutes the result for x_1 in the other equations. This yields a new dictionary:

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

The step we just performed is called a *pivot* operation, and it is the basic step of the algorithm. In fact it is the only step: we simply repeat this operation. In (4), we first set the cobasic (i.e., right-hand) variables to zero and get the basic solution $x_1 = 1$, $x_5 = 1$, $x_6 = -1$. Again, we find the basic variable with the smallest index and negative value, namely, x_6. In the equation for x_6 we find the smallest-indexed cobasic variable with a positive coefficient, here x_4. We pivot by solving this equation for x_4 and substituting for x_4 in the other equations, obtaining the new dictionary:

$$\begin{aligned} x_1 &= 2 - 2x_2 + 2x_3 + x_6 \\ x_4 &= 1 + x_3 + x_6 \\ x_5 &= 0 + 5x_2 - x_3 - x_6 \end{aligned} \tag{5}$$

We are now in luck. The basic solution is nonnegative, and its restriction to our original three variables gives a feasible solution to (2): $x_1 = 2$, $x_2 = 0$, $x_3 = 0$. So far so good. An immediate question raises itself: What happens if there is no solution to the original problem? Consider the following problem:

$$\begin{aligned} -x_1 + 2x_2 + x_3 &\leq 3 \\ 3x_1 - 2x_2 + x_3 &\leq -17 \\ -x_1 - 6x_2 - 23x_3 &\leq 19 \end{aligned} \tag{6}$$

We get an initial dictionary by introducing three slack variables and letting them be the basic variables:

$$\begin{aligned} x_4 &= 3 + x_1 - 2x_2 - x_3 \\ x_5 &= -17 - 3x_1 + 2x_2 - x_3 \\ x_6 &= 19 + x_1 + 6x_2 + 23x_3 \end{aligned} \tag{7}$$

The algorithm proceeds as before by choosing the equation for x_5 and solving for x_2:

$$\begin{aligned} x_2 &= 17/2 + (3/2)x_1 + (1/2)x_3 + (1/2)x_5 \\ x_4 &= -14 - 2x_1 - 2x_3 - x_5 \\ x_6 &= 70 + 10x_1 + 26x_3 + 3x_5 \end{aligned} \tag{8}$$

Here we encounter something new. We select the equation for x_4, as we should, but find that there is no cobasic variable with a positive coefficient. We rewrite this equation with all variables on the left-hand side, including those with zero coefficients, getting

$$2x_1 + 0x_2 + 2x_3 + \mathbf{1}x_4 + \mathbf{1}x_5 + \mathbf{0}x_6 = -14. \tag{9}$$

This is an example of an *inconsistent equation*. Note that the coefficients of all variables are nonnegative, but the right-hand side is negative. Therefore this equation cannot be satisfied by choosing any combination of nonnegative values for the variables. This equation was derived from the original system by standard operations that do not change the solution set for the equations. Therefore (7), hence (6), has no nonnegative solution. In fact, (9) provides a simple proof of this encoded in the boldface coefficients of the slack variables. We multiply each inequality in (6) by the coefficient of its corresponding slack variable

$$\begin{aligned} \mathbf{1} * (-x_1 + 2x_2 + x_3 \le 3) \\ +\mathbf{1} * (3x_1 - 2x_2 + x_3 \le -17) \\ +\mathbf{0} * (-x_1 - 6x_2 - 23x_3 \le 19) \end{aligned} \tag{10}$$

and add the inequalities in (10) to get

$$2x_1 + 2x_3 \le -14. \tag{11}$$

The final inequality (11) is called an *inconsistent inequality*: all the variables have nonnegative coefficients, yet the right-hand side is negative. The multipliers given by the coefficients of the slack variables are said to furnish a *certificate of infeasibility* for the original system.

We now have a complete description of the algorithm that we christen the "b-rule" for solving problems of form (1):

Step 1: Introduce m slack variables $x_{n+1}, \dots, x_{n+m}$ and use these as the basis (left-hand side) of an initial dictionary:

$$x_{n+i} = b_i - \sum_{j=1}^{n} a_{ij}x_j \qquad (i = 1, \dots, m). \tag{12}$$

Step 2: Set the cobasic (right-hand side) variables to zero. Find the smallest index of a basic (left-hand side) variables with a negative value. If there is none, terminate with a feasible solution.

Step 3: Find the cobasic variable in the equation chosen in step 2 that has the smallest index and a positive coefficient. If there is none, terminate, for the problem is infeasible, and the coefficients of the slack variables represent a certificate of infeasibility. Otherwise, solve this equation for the indicated cobasic variable, and substitute the result in all of the other equations. Go to step 2.

In what follows we prove:

- the algorithm that we have described halts after a finite number of steps;
- if it halts in step 2, then the basic solution is feasible for (1);
- if it halts in step 3, then the system (1) is infeasible and the slack coefficients "certify" this.

2. PROOF OF CORRECTNESS.

Theorem 1. *The b-rule is finite.*

Proof. Given an input system (1), we construct the initial dictionary (12) and run the b-rule algorithm. Since there are at most $\binom{n+m}{m}$ possible choices of a basis, if the algorithm is not finite (in the sense of halting after finitely many steps), then some bases must be repeated, a process called *cycling*. Assume that this can happen, and choose a system of equations that cycles.

Suppose first that x_{n+m} ($n+m$ being the largest index) enters and leaves the basis during the cycle. When x_{n+m} is chosen to enter the basis we must have an equation of the following form, where B and N denote the set of basic and cobasic indices, respectively:

$$x_k = -b'_k - \sum_{j \in N \setminus \{n+m\}} a'_{kj} x_j + a'_{k,n+m} x_{n+m} \qquad (k \in B). \tag{13}$$

The choice of x_{n+m} as entering variable in this equation implies that $-b'_k < 0$, $a'_{k,n+m} > 0$, and $a'_{kj} \geq 0$ for j in $N \setminus \{n+m\}$. This shows that every solution to the full system of equations with $x_1, \ldots, x_{n+m-1} \geq 0$ must have $x_{n+m} > 0$.

Now consider the stage at which x_{n+m} is chosen to leave the basis. The dictionary has the form:

$$\begin{aligned} x_i &= b'_i + \sum_{j \in N} a'_{ij} x_j \qquad (i \in B \setminus \{n+m\}) \\ x_{n+m} &= -b'_{n+m} + \sum_{j \in N} a'_{n+m,j} x_j \end{aligned} \tag{14}$$

The choice of x_{n+m} ensures that $-b'_{n+m} < 0$ and $b'_i \geq 0$ for i in $B \setminus \{n+m\}$. By setting the cobasic variables to zero, dictionary (14) shows that there exists a solution to the system of equations with $x_1, \ldots, x_{n+m-1} \geq 0$ and $x_{n+m} < 0$. Clearly not both (13) and (14) can hold, so there cannot exist a cycle during which the largest-indexed variable enters and leaves the basis.

Now suppose that there exists a cycle in which x_{n+m} always stays in the basis. Then we can remove x_{n+m} and its corresponding equation without changing the pivot decisions made during the cycle. Similarly, if there exists a cycle where x_{n+m} always stays in the cobasis, then we can remove x_{n+m} from all of the equations without influencing the cycle. Either way we can reduce the original example that cycles to an equivalent example with a cycle during which the largest-indexed variable both enters and leaves the basis. This leads to the two conflicting situations that we met earlier, so a cycle cannot exist: the algorithm is finite. ∎

Since the algorithm is finite, it must halt in either step 2 or step 3. If it terminates in step 2, we have a nonnegative solution to the original system. This follows from the fact that the only operations we performed on the initial dictionary were standard operations for manipulating a system of equations. If the algorithm stops in step 3, we have a certificate of infeasibility that, when stated in general terms, is a variant of the Farkas lemma.

Theorem 2. *Either there exists x in R^n with $x \geq 0$ such that $Ax \leq b$ or there exists y in R^m with $y \geq 0$ such that $y^T A \geq 0$ and $y^T b < 0$.*

Proof. We begin by noting that there cannot exist both a vector x and a vector y satisfying the conditions of the theorem. For otherwise, $0 > y^T b \geq y^T Ax \geq 0$. If such a vector x does not exist, the finiteness of the b-rule implies that the algorithm must halt in step 3. The algorithm returns an inconsistent equation:

$$\sum_{\substack{j=1,\\ j\neq k}}^{n+m} a'_{kj} x_j + x_k = -b'_k, \tag{15}$$

where $b'_k > 0$ and all of the coefficients $a'_{kj} \geq 0$. Set $y_i = a'_{k,n+i} \geq 0$ for $i = 1, \ldots, m$. We observe that equation (15) is obtained from the initial dictionary (12) by multiplying the equation for x_{n+i} by y_i and summing. This is because variable x_{n+i} appears only once in the entire dictionary, as the left-hand side of its defining equation. This shows that $y^T b = -b'_k < 0$ and that

$$\sum_{i=1}^{m} y_i a_{ij} = a'_{kj} \geq 0 \qquad (j = 1, \ldots, n), \tag{16}$$

again by the halting property of the algorithm. Hence $y^T A \geq 0$. ∎

3. CONCLUSION. The b-rule is a finite algorithm that finds a nonnegative solution to a system of linear inequalities. Readers familiar with linear programming will recognize that the b-rule is the dual form of Bland's rule [**1**] (see also [**2**]), with a zero objective (cost) row. Simple variants of the b-rule exist for finding a solution to $Ax \leq b$ or to $Ax = b$ and $x \geq 0$, etc. We can also use the b-rule to obtain algorithmic proofs of the Farkas lemma and the Fundamental Theorem of Linear Inequalities. In practice, the b-rule can be used to find a starting primal-feasible basis for a linear program without having to introduce the traditional "phase-one" artificial variable. However, as with all pivot rules known, in the worst case it may require an exponential number of steps.

REFERENCES

1. R. G. Bland, New finite pivot rules for the simplex method, *Math. Oper. Res.* **2** (1977) 103–107.
2. V. Chvátal, *Linear Programming*, Freeman, New York, 1983.

3. K. Fukuda and T. Terlaky, Criss-cross methods: A fresh view on pivot algorithms, *Math. Program.* **79** (1997) 369–395.
4. G. Strang, *Linear Algebra and Its Applications*, Academic Press, London, 1980.

School of Computer Science, McGill University, 3480 University Street, Montreal, Quebec, H3A 2A7, Canada
{avis,beezer}@cs.mcgill.ca

Cauchy's Interlace Theorem for Eigenvalues of Hermitian Matrices

Suk-Geun Hwang

Hermitian matrices have real eigenvalues. The Cauchy interlace theorem states that the eigenvalues of a Hermitian matrix A of order n are interlaced with those of any principal submatrix of order $n-1$.

Theorem 1 (Cauchy Interlace Theorem). *Let A be a Hermitian matrix of order n, and let B be a principal submatrix of A of order $n-1$. If $\lambda_n \le \lambda_{n-1} \le \cdots \le \lambda_2 \le \lambda_1$ lists the eigenvalues of A and $\mu_n \le \mu_{n-1} \le \cdots \le \mu_3 \le \mu_2$ the eigenvalues of B, then $\lambda_n \le \mu_n \le \lambda_{n-1} \le \mu_{n-1} \le \cdots \le \lambda_2 \le \mu_2 \le \lambda_1$.*

Proofs of this theorem have been based on Sylvester's law of inertia [**3**, p. 186] and the Courant-Fischer minimax theorem [**1**, p. 411], [**2**, p. 185]. In this note, we give a simple, elementary proof of the theorem by using the intermediate value theorem.

Proof. Simultaneously permuting rows and columns, if necessary, we may assume that the submatrix B occupies rows $2, 3, \ldots, n$ and columns $2, 3, \ldots, n$, so that A has the form

$$A = \begin{bmatrix} a & \mathbf{y}^* \\ \mathbf{y} & B \end{bmatrix},$$

where $*$ signifies the conjugate transpose of a matrix. Let $D = \operatorname{diag}(\mu_2, \mu_3, \ldots, \mu_n)$. Then, since B is also Hermitian, there exists a unitary matrix U of order $n-1$ such that $U^*BU = D$. Let $U^*\mathbf{y} = \mathbf{z} = (z_2, z_3, \ldots, z_n)^T$.

We first prove the theorem for the special case where $\mu_n < \mu_{n-1} < \cdots < \mu_3 < \mu_2$ and $z_i \ne 0$ for $i = 2, 3, \ldots, n$. Let

$$V = \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & U \end{bmatrix},$$

in which $\mathbf{0}$ denotes the zero vector. Then V is a unitary matrix and

$$V^*AV = \begin{bmatrix} a & \mathbf{z}^* \\ \mathbf{z} & D \end{bmatrix}.$$

Let $f(x) = \det(xI - A) = \det(xI - V^*AV)$, where I denotes the identity matrix. Expanding $\det(xI - V^*AV)$ along the first row, we get