JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 72, No. 1, JANUARY 1992

New Iterative Methods for Linear Inequalities[1]

K. YANG[2] AND K. G. MURTY[3]

Communicated by O. L. Mangasarian

Abstract. New iterative methods for solving systems of linear inequalities are presented. Each step in these methods consists of finding the orthogonal projection of the current point onto a hyperplane corresponding to a surrogate constraint which is constructed through a positive combination of a group of violated constraints. Both sequential and parallel implementations are discussed.

Key Words. Linear inequalities, surrogate constraints, iterative methods, sequential implementation, parallel implementation.

1. Introduction

We consider the problem of finding a feasible solution to

$$Ax \leq b, \tag{1}$$

where $A=(a_{ij}) \in \mathbb{Z}^{m \times n}$ and $b=(b_i) \in \mathbb{Z}^m$. Large-scale versions of this problem appear in image reconstruction from projections (Ref. 1), which is becoming important in many scientific fields. In medical science, computerized tomography reconstructs the images of cross sections of the human body by processing data obtained from measuring the attenuation of X-rays passing

[1]The authors are grateful to a referee for pointing out the result in Lemma 5.1 and its importance in the proof of Theorem 5.1. This work was supported partially by NSF Grant No. ECS-85-21183.

[2]Formerly, Graduate Student, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan. Presently, Assistant Professor, Department of Industrial and Manufacturing Engineering, Wayne State University, Detroit, Michigan.

[3]Professor, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan.

0022-3239/92/0100-0163$06.50/0

through it. Other image reconstruction problems arise in remote sensing (Ref. 2), seismic tomography (Ref. 3), and industrial nondestructive testing.

One approach transforms (1) into a linear program and solves it by methods such as Karmarkar's method (Refs. 4 and 5) or the simplex method. These methods require matrix operations which are often impractical for the large-scale systems that arise in applications such as image reconstruction.

The second approach involves using iterative methods, the basic computation step in which is extremely simple and easy to program. Because of these advantages, the linear inequality solvers employed in image reconstruction are most often iterative methods. One class of iterative methods is derived from the relaxation method for linear inequalities (Refs. 6 and 7) and Kaczmarz's method (Ref. 8) for linear equations. The name refers to the fact that they consider one constraint at a time, so in each step, all but one constraint are relaxed. In each iteration, a violated constraint is identified and an orthogonal projection is made onto it from the current point. So they are also called successive orthogonal projection methods. Bregman (Refs. 9 and 10), Eremin (Ref. 11), and Gubin *et al.* (Ref. 12) extended this idea to finding a point in a convex set defined by a system of inequalities involving convex functions. An orthogonal projection onto a single linear constraint is computationally inexpensive, but considering only one constraint at a time leads to slow convergence. Instead, it is better to process a group of constraints at a time. But making an orthogonal projection onto the affine space corresponding to a group of constraints is computationally expensive; the amount of work for it grows as the cube of the number of constraints in the group.

Another class of iterative methods is derived from Cimmino's algorithm (Ref. 13) for linear equations. Censor and Elfving (Ref. 14) and De Pierro and Iusem (Ref. 15) developed a Cimmino-type algorithm for linear inequalities. This method makes orthogonal projections simultaneously onto each of the violated constraints from the current point and takes the new point to be a convex combination of those projection points. Cimmino's method is amenable to parallel implementation, but making projections onto every violated constraint is again computationally expensive, and the method tends to have slow convergence.

In this paper, we propose surrogate constraint methods which are able to process a group of violated constraints at a time but retain the same computational simplicity of the relaxation method, and at the same time are highly amenable to massively parallel implementation. In each iteration, a surrogate constraint is derived from a group of violated constraints. The current point is orthogonally projected onto this surrogate constraint treated as an equation, and the process is repeated until a feasible solution is found.

2. Notation and Assumptions

A_i = ith row vector of A, assumed $\neq 0$, and integer for all i;
K = set of feasible solutions of (1), assumed $\neq \varnothing$;
$I \subset \{1, \ldots, m\}$ denotes an index set, which identifies a subset of the constraints;
$K_i = \{x \mid A_i x \leq b_i\}$ is the half space corresponding to the ith constraint;
$H_i = \{x \mid A_i x = b_i\}$ is the hyperplane corresponding to the boundary of the ith constraint;
$K_I = \{\bigcap K_i\}_{i \in I}$, $K_I \neq \varnothing$, since $K_I \supset K$;
$|S|$ = cardinality of the set S;
$A_I = |I| \times n$ matrix with rows A_i, $i \in I$;
$b_I = (b_i, i \in I)$, column vector;
$\|x\| = +\sqrt{\sum x_j^2}$, Euclidean norm of the vector x;
$d(x, H_i)$ = minimum Euclidean distance from x to H_i;
$d(x, K_i)$ = minimum Euclidean distance from x to K_i;
$d = 0$, if $x \in K_i$, and $d = d(x, H_i)$, otherwise;
$\phi(x) = \sup\{d(x, K_i): i = 1 \text{ to } m\}$;
$d(x, K)$ = minimum Euclidean distance from x to K;
L = length of the binary encoding of all data in (1);
$I(x) = \{i: A_i x - b_i > 0\}$, for any $x \in \mathbb{R}^n$.

$I(x)$ is the index set of violated constraints at x. The point $x \in K$ iff $I(x) = \varnothing$. Finding $I(x)$ is highly amenable to massive parallel implementation. We can use up to m simple processors operating in parallel, each one dedicated to checking a separate constraint or a small group of constraints in (1). The following lemma (Refs. 16 and 17) will be used in the convergence proofs.

Lemma 2.1. If the system (1) is feasible, then there is a feasible solution $\hat{x}$, with $|\hat{x}_j| \leq 2^L/2n$, $j = 1, \ldots, n$.

Without any loss of generality, we assume that each row of A is normalized so that $\|A_i\| = 1$, for all $i = 1$ to m. This has no effect on K, or K_i, or H_i, but makes it easier to write the projections on H_i. Clearly, a point $x \in K$ iff $\phi(x) = 0$. In practice, we are usually interested in getting an approximate solution for (1) within some tolerance. Given a tolerance $\epsilon > 0$, a point x is said to be feasible to (1) within tolerance ϵ if $\phi(x) \leq \epsilon$, i.e.,

$$x \in K_\epsilon = \{x: A_i x \leq b_i + \epsilon, \text{ for all } i = 1 \text{ to } m\}.$$

3. Basic Surrogate Constraint Method

In this method, when $\bar{x}$ is the current point, if $I(\bar{x}) = I$, a row vector $\pi = (\pi_i: i \in I(\bar{x}))$ of positive weights is selected, and the surrogate constraint

$(\pi A_I)x \leq (\pi b_I)$ generated. The corresponding surrogate hyperplane is

$$H_s = \{x: (\pi A_I)x = (\pi b_I)\}.$$

We assume that the weight vector π is normalized so that

$$\sum (\pi_i: \text{over } i \in I(\bar{x})) = 1.$$

This assumption is made purely for the sake of simplifying some statements. The next point in the method is one on the line segment joining the current point and its reflection on the surrogate hyperplane. The actual point selected in this line segment depends on a parameter λ, which can be set by the user anywhere between 0 and 2 ($\lambda = 1$ corresponds to the orthogonal projection). The algorithm is initiated with some point $x^0 \in \mathbb{R}^n$, which could be 0 or some known near-feasible point. The general step $k+1$, for $k \geq 0$ is given below.

Step $k+1$. Let x^k be the point obtained at the end of the previous step. Identify the index set $I_k = I(x^k)$ of violated constraints. If $I_k = \varnothing$, x^k is feasible to (1); terminate. Otherwise, select a weight vector $\pi(k) = (\pi_i^k: i \in I_k)$, and compute

$$x^{k+1} = x^k - [\lambda(\pi(k)A_{I_k}x^k - \pi(k)b_{I_k})(\pi(k)A_{I_k})^T]/\|\pi(k)A_{I_k}\|^2, \qquad (2)$$

where $0 < \lambda < 2$; go to the next step.

See Fig. 1. The most expensive piece of work in each step, that of finding the index set $I(x^k)$ of violated constraints, is easily implemented in a massive parallel way as discussed above; this is a major advantage of these methods.

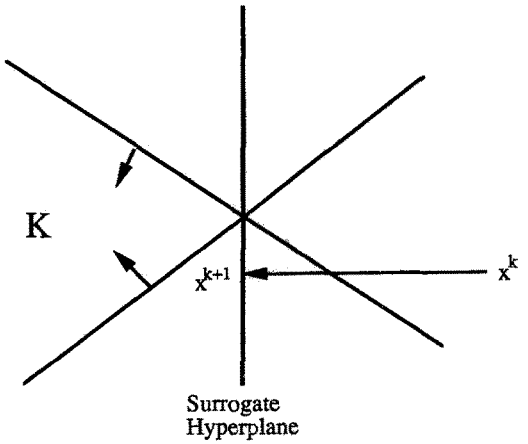

Fig. 1. A step in the surrogate constraint method with $\lambda = 1$. Both constraints are violated at x^k.

A variety of strategies can be used for selecting the weight vector $\pi(k)=(\pi_i^k: i\in I_k)$. One selects all the weights to be equal, i.e.,

$$\pi_i^k=1/|I_k|, \qquad i\in I_k.$$

Another strategy computes the error $r_i=A_ix^k-b_i$, the Euclidean distance from the current point x^k to K_i, for each $i\in I_k$, and takes

$$\pi_i^k=r_i\Big/\Big(\sum (r_i: \text{over } i\in I_k)\Big).$$

A third strategy takes the weight vector to be a convex combination of the above two vectors, etc. For the sake of simplicity, all our convergence proofs will be based on the assumption that the weight vector $\pi=(\pi_i: i\in I(x^k))$ is selected so as to satisfy

$$\pi_i>\gamma, \qquad i\in I(x^k),$$

when x^k is the current point, where γ is some predetermined small positive quantity.

Convergence Results.

Definition 3.1. When $K\neq\varnothing$, a sequence $\{x^k\}_{k=1}^{\infty}$ in $\mathbb{R}^n$ is called strictly Fejer-monotone with respect to the set K if, for every $x\in K$,

$$\|x^{k+1}-x\|<\|x^k-x\|, \qquad \text{for all } k\geq 0. \tag{3}$$

Every Fejer-monotone sequence is bounded if $K\neq\varnothing$, since $\|x^k-x\|$ is always positive and monotonically decreasing with K.

Theorem 3.1. If $K\neq\varnothing$, any sequence $\{x_k\}$ generated by the above algorithm is strictly Fejer-monotone with respect to K.

Proof. Select any point $x\in K$. Define $e^k=x^k-x$, $k=0, 1, \ldots$. For simplicity, denote $\pi(k)$ by π and I_k by I. Then, if $I\neq\varnothing$,

$$\begin{aligned}
e^{k+1}&=e^k-\lambda(\pi A_I x^k-\pi b_I)(\pi A_I)^T/\|\pi A_I\|^2,\\
\|e^{k+1}\|^2&=\|e^k\|^2+\lambda^2(\pi A_I x^k-\pi b_I)^2/\|\pi A_I\|^2\\
&\quad -2\lambda(\pi A_I x^k-\pi b_I)(\pi A_I)(x^k-x)/\|\pi A_I\|^2\\
&=\|e^k\|^2+\lambda^2(\pi A_I x^k-\pi b_I)^2/\|\pi A_I\|^2\\
&\quad -2\lambda(\pi A_I x^k-\pi b_I)(\pi A_I x^k-\pi b_I-\pi A_I x+\pi b_I)/\|\pi A_I\|^2\\
&=\|e^k\|^2+\lambda^2(\pi A_I x^k-\pi b_I)^2/\|\pi A_I\|^2\\
&\quad -2\lambda(\pi A_I x^k-\pi b_I)^2/\|\pi A_I\|^2\\
&\quad +2\lambda(\pi A_I x^k-\pi b_I)(\pi A_I x-\pi b_I)/\|\pi A_I\|^2.
\end{aligned}$$

Since

$$\pi A_I x^k > \pi b_i \quad \text{and} \quad \pi A_I x \leq \pi b_I,$$

we have

$$2\lambda(\pi A_I x^k - \pi b_I)(\pi A_I x - \pi b_I)/\|\pi A_I\|^2 \leq 0.$$

Therefore, it follows that

$$\|e^{k+1}\|^2 \leq \|e^k\|^2 - \lambda(2-\lambda)(\pi A_I x^k - \pi b_I)^2/\|\pi A_I\|^2 < \|e^k\|^2. \tag{4}$$

□

Theorem 3.2. If $K \neq \varnothing$, any infinite sequence $\{x^k\}$ generated by the above algorithm has the property

$$\lim_{k\to\infty} \phi(x^k) = 0.$$

Proof. Select any point $x \in K$, and define $e^k = x^k - x$, for all k, as above. Fejer-monotonicity implies that the sequence $\{\|e^k\|\}_{k=1}^{\infty}$ is monotonically decreasing, hence it converges, which implies that

$$\lim_{k\to\infty} \|e^{k+1}\| = \lim_{k\to\infty} \|e^k\|.$$

It follows from (4) that

$$\lim_{k\to\infty} (\pi(k) A_{I_k} x^k - \pi(k) b_{I_k}) = 0.$$

Since

$$\pi_i(k) > \gamma, \qquad \text{for all } i \in I(x^k),$$

$$\left(\sum \pi_i^k : \text{over } i \in I_k\right) = 1,$$

$$A_i x^k - b_i > 0, \qquad \text{for all } i \in I_k,$$

this implies that either

$$\lim_{k\to\infty} (A_i x^k - b_i) = 0, \qquad \text{for all } i \in I_k,$$

or

$$\text{for some } \bar{k} < \infty, \qquad I(x^{\bar{k}}) = \varnothing.$$

Therefore,

$$\lim_{k\to\infty} (\phi(x^k) = \sup_{i\in\{1,\ldots,m\}} d(x^k, K_i) = 0. \qquad \square$$

Lemma 3.1. Assume that $K \neq \varnothing$ and that the sequence $\{x^k\}_{k=1}^{\infty}$ satisfies the following conditions:

(i) $\{x^k\}_{k=1}^{\infty}$ is Fejer-monotone with respect to K;
(ii) $\lim_{k\to\infty} \phi(x_k) = 0$.

Then, $\{x^k\}$ converges to an $x \in K$.

Proof. This follows from Lemmas 5 and 6 of Ref. 12. $\square$

Theorem 3.3. Let $\epsilon > 0$ be a specified error tolerance. If $K \neq \varnothing$, and we set $I(x^k) = \{i: A_i x^k - b_i > \epsilon\}$ in the above algorithm, it converges to a point $x \in K_\epsilon$ in a finite number of iterations.

Proof. (i) First, we show that, for any $x \in K$, if $x^k \notin K_\epsilon$, then

$$\|x^{k+1} - x\| < \|x^k - x\|, \qquad \text{for all } k \geq 0.$$

This follows directly from the fact that, if $x^k \notin K_\epsilon$, then $I(x^k) \neq \varnothing$, so the algorithm will not terminate in the step when x^k is the current point. From (4), we have

$$\|e^{k+1}\|^2 \leq \|e^k\|^2 - \lambda(2-\lambda)(\pi A_I x^k - \pi b_I)^2 / \|\pi A_I\|^2 < \|e^k\|^2.$$

Hence, the result follows.

(ii) Then, we show that, in this case, any infinite sequence $\{x^k\}_{k=1}^{\infty}$ generated by this algorithm has the property

$$\lim_{k\to\infty} \phi(x^k) \leq \epsilon.$$

This follows from the fact that the sequence $\{\|e^k\|\}_{k=1}^{\infty}$ is positive and monotonically decreasing, hence it converges. So,

$$\lim_{k\to\infty} \|e^{k+1}\| = \lim_{k\to\infty} \|e^k\|.$$

So, it follows from (4) that

$$\lim_{k\to\infty} (\pi(k) A_{I_k} x^k - \pi(k) b_{I_k}) = 0. \tag{5}$$

But since

$$\pi(k)>0, \qquad \left(\sum \pi_i^k: \text{over } i\in I_k\right)=1,$$

$$A_i x^k - b_i > \epsilon, \qquad \text{for all } i\in I_k,$$

this implies that

$$\pi(k)A_{I_k}x^k - \pi(k)b_{I_k} > \epsilon,$$

as long as $I_k \neq \varnothing$, which contradicts (5). So there must exist $\bar{k}<\infty$, such that $I(x^{\bar{k}})=\varnothing$, which implies that, at iteration $\bar{k}$,

$$\phi(x^{\bar{k}}) = \sup_{i\in\{1,\ldots,m\}} d((x^{\bar{k}}, K_i) \leq \epsilon.$$

In other words, $x^{\bar{k}}\in K_\epsilon$.

(iii) Bounds on $\bar{k}$. Select a point $x\in K$, and define $e^k = x^k - x$, for all k. From Lemma 2.1, we have

$$\|e^0\| \leq 2^{L-1}/\sqrt{n}.$$

If at iteration k, $I(x^k)\neq\varnothing$, $A_i x^k - b_i \geq \epsilon$, for all $i\in I(x^k)$, it follows that

$$\pi A_{I_k}x^k - \pi b_{I_k} \geq \left(\sum \pi_i: \text{over } i\in I_k\right)(\min\{A_i x^k - b_i: \text{over } i\in I_k\})$$

$$\geq \left(\sum \pi_i: \text{over } i\in I_k\right)\epsilon \geq \epsilon.$$

and that

$$\|\pi A_{I_k}\| \leq (\|\pi\|)(\|A_{I_k}\|) \leq (\|\pi\|)(\|A_{I_k}\|_F)$$

$$= \left(\sum_{i\in I_k} \pi_i^2\right)^{1/2}\left(\sum_{i\in I_k}\sum_{j=1}^{n} |a_{ij}|^2\right)^{1/2} < m,$$

where

$$\|A_{I_k}\|_2 = \sup_{x\neq 0}\|A_{I_k}x\|/\|x\|,$$

and where

$$\|A_{I_k}\|_F = \left(\sum_{i\in I_k}\sum_{j=1}^{n} |a_{ij}|^2\right)^{1/2}$$

is the Frobenius norm of A_{I_k}. It follows that

$$\|e^{k+1}\|^2 < \|e^k\|^2 - \lambda(2-\lambda)\epsilon^2/m^2.$$

Therefore, this algorithm converges within $\bar{k}$ steps, where

$$\bar{k} \leq m^2 2^{2L-2}/n\lambda(2-\lambda)\epsilon^2. \qquad \square$$

4. Sequential Surrogate Constraint Method

In many applications, the matrix A is often very large (m and n are 10^5 or more) and sparse (less than 0.1% of entries are nonzero), and working on the whole matrix A may be almost impossible. So, it is preferable to work on a small subset of constraints of (1) at a time. Specifically, the matrix A can be partitioned into p submatrices, and the right-hand side vector b can be partitioned compatibly into p subvectors, as follows:

$$A = \begin{bmatrix} A^1 \\ \vdots \\ A^t \\ \vdots \\ A^p \end{bmatrix}, \qquad b = \begin{bmatrix} b^1 \\ \vdots \\ b^t \\ \vdots \\ b^p \end{bmatrix}, \tag{6}$$

where A^t is an $m_t \times n$ matrix, b^t has m_t rows, $t=1$ to p, and

$$\sum_{t=1}^{p} m_t = m.$$

Now, we will show that the surrogate constraint method can be used to solve the system (1) by successively applying on the subsystems,

$$A^t x \leq b^t, \qquad t=1 \text{ to } p,$$

in cyclic order.

For any $x \in \mathbb{R}^n$, define

$$I^t(x) = \{i: i\text{th constraint in } t\text{th subsystem is violated by } x\}.$$

We denote by $\pi^t = (\pi_1^t, \ldots, \pi_{m_t}^t)$ the weight vector for the tth subsystem, $t=1$ to p. When the current point is x^k, and we have to operate on the tth subsystem next, we will set

$$\pi_i^t > 0, \qquad \text{if } i \in I^t(x^k),$$

$$\pi_i^t = 0, \qquad \text{otherwise.}$$

Then, the corresponding surrogate constraint for the tth subsystem is

$$\pi^t A^t x \leq \pi^t b^t,$$

and the surrogate hyperplane of the tth subsystem is

$$H_s^t = \{x: \pi^t A^t x = \pi^t b^t\}.$$

The algorithm goes through major cycles. In every major cycle, each of the p subsystems is operated on once, in serial order $t=1$ to p. Initialization is the same as in the previous algorithm. Consider a major cycle. In this major cycle, operate on subsystems in the order $t=1$ to p.

Let x^k be the current point, and let the tth subsystem be the one to be operated next. Find $I^t(x^k)$.

If $I^t(x^k)=\varnothing$, define $x^{k+1}=x^k$, and go to the next subsystem with x^{k+1}, if $t<p$. If $t=p$, this completes the major cycle. If there is no change in the current point throughout this major cycle, then the current point is feasible to (1); terminate. Otherwise, go to the next major cycle with the current point.

If $I^t(x^k)\neq\varnothing$, select a weight vector π^t and define

$$x^{k+1}=x^k-\lambda d^k, \tag{7a}$$

where

$$d^k=(\pi^t A^t x^k-\pi^t b^t)(\pi^t A^t)^T/\|\pi^t A^t\|^2 \tag{7b}$$

and $0<\lambda<2$. With x^{k+1}, go to the next subsystem if $t<p$, or to the next major cycle if $t=p$.

Convergence Results.

Lemma 4.1. Let $c^0\in\mathbb{R}^n$, and let C^T be a row vector in $\mathbb{R}^n$. Let Γ be the half space $\{x: C^T x\leq C^t c^0\}$. If $z\notin\Gamma$ is such that its orthogonal projection on Γ is c^0, then

$$\|y-z_\alpha\|<\|y-z\|, \text{ for all } y\in\Gamma \text{ and } z_\alpha=z-\alpha(z-c^0) \text{ for } 0<\alpha<2.$$

See Fig. 2.

Proof. See Refs. 6 and 11. □

Lemma 4.2. Suppose that the point $x^{k+1}=x^k-\alpha d^k$ is obtained by operating on the tth subsystem with x^k as the current point, where d^k is given by (7b). Let Γ^t be the half-space corresponding to the surrogate constraint in

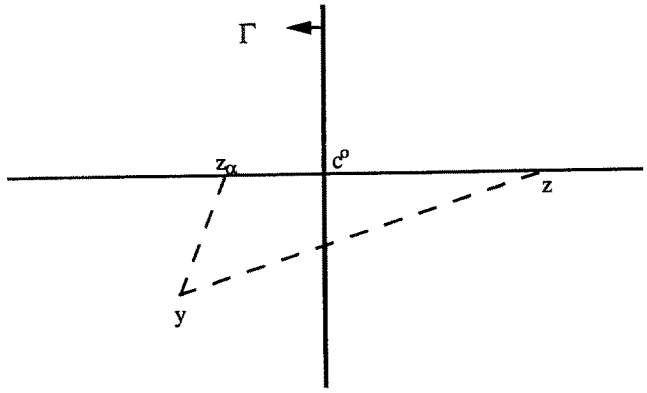

Fig. 2. Illustration of Lemma 4.1.

this step,

$$\pi^t A^t x \leq \pi^t b^t.$$

Then $K \subset \Gamma^t$ and

$$\|y-(x^k-\alpha d^k)\| < \|y-x^k\|, \quad \text{for all } y \in \Gamma^t \text{ and } 0<\alpha<2. \tag{8}$$

Proof. Any point feasible to the subsystem alone must satisfy the surrogate constraint, i.e., must be in Γ^t. So, $K \subset \Gamma^t$. The surrogate hyperplane in this iteration is

$$H_s^t = \{x: \pi^t A^t x = \pi^t b^t\},$$

and $x^k - d^k$ is the orthogonal projection of x^k onto it. So,

$$\pi^t b^t = \pi^t A^t (x^k - d^k).$$

Relation (8) now follows by using the result in Lemma 4.1. □

Theorem 4.1. In this algorithm, if $x^{k+1} \neq x^k$, then

$$\|x-x^{k+1}\| < \|x-x^k\|, \qquad \text{for all } x \in K.$$

Proof. It follows directly from Lemma 4.2. □

Theorem 4.2. If $K \neq \varnothing$, any sequence $\{x^k\}_{k=1}^{\infty}$ generated by this algorithm has the property that

$$\lim_{k\to\infty} \phi(x^k) = 0.$$

Proof. For any $x \in K$, Theorem 4.1 implies that the sequence $\{\|x^k - x\|\}_{k=1}^{\infty}$ is monotonically decreasing, thus it converges. So, if $t(k)$ denotes the subsystem operated upon when x^k is the current point, and if $\pi^{t(k)}$ is the weight vector used in that step, then

$$\lim_{k\to\infty} (\pi^{t(k)} A^{t(k)} x^k - \pi^{t(k)} b^{t(k)}) = 0.$$

Since

$$\pi_i^{t(k)} > 0, \qquad \text{for all } i \in I^{t(k)}(x^k),$$

and sums to one over these i, and since

$$A_i^{t(k)} x^k - b_i^{t(k)} > 0, \qquad \text{for all } i \in I^{t(k)}(x^k),$$

this implies that either

$$\lim_{k\to\infty} (A_i^{t(k)} x^k - b_i^{t(k)}) = 0, \qquad \text{for all } i \in I^{t(k)}(x^k),$$

or there exist an $\bar{r} < \infty$ such that, at major cycle $\bar{r}$,

$$I^t(x^k) = \varnothing, \qquad \text{for all } t = 1 \text{ to } p.$$

Therefore,

$$\lim_{k\to\infty} \phi(x^k) = 0. \qquad \square$$

Theorem 4.3. Let $\epsilon > 0$ be a specified error tolerance. If $K \neq \varnothing$, and if we operate this algorithm by setting

$$I^t(x^k) = \{i: A_i^t x^k - b_i^t > \epsilon\}, \qquad \text{for all } k,$$

it converges to a point $x \in K_\epsilon$ in a finite number of major cycles.

Proof. We proceed as in the proof of Theorem 3.3. From Theorem 4.1, we know that, if $x^k \notin K_\epsilon$, then

$$\|x - x^{k+1}\| < \|x - x^k\|, \qquad \text{for all } x \in K.$$

Define $e^k = x^k - x$. Since $\{\|e^k\|\}$ is monotonically decreasing and bounded below, it converges, and this implies that

$$\lim_{k\to\infty} \|e^{k+1}\| = \lim_{k\to\infty} \|e^k\|.$$

But this happens only if there exists an $\bar{r} < \infty$ such that, at major cycle $\bar{r}$, the current point is x^k and $I^t(x^k) = \varnothing$, for all $t = 1$ to p. Otherwise for all k, there

exists a subsystem t such that $I^t(x^k) \neq \varnothing$, then $\pi^t A^t x^k - \pi^t b^t > \epsilon$, hence

$$\lim_{k\to\infty} \|e^k\| \neq \lim_{k\to\infty} \|e^{k+1}\|,$$

a contradiction. So, operated as stated, this algorithm must converge to a point in K_ϵ in a finite number of major cycles.

Now, we derive a bound on $\bar{r}$. From Lemma 2.1, it follows that $\|e^0\| \leq 2^{L-1}/\sqrt{n}$. If at the beginning of a major cycle r, the current point $x^k \notin K_\epsilon$, there exists at least one subsystem t, and at least one constraint in it, say, the ith, such that $A_i^t x^k - b_i^t > \epsilon$. So, a change of point must occur in some step in this major cycle. Let t be the subsystem operated on in that step; let x^g, x^{g+1} be the points at the beginning and end of this step; and let π^t be the weight vector used. So,

$$e^{g+1} = e^g - \lambda(\pi^t A^t x^g - \pi^t b^t)(\pi^t A^t)^T / \|\pi^t A^t\|^2,$$

$$\|e^{g+1}\|^2 \leq \|e^g\|^2 - \lambda(2-\lambda)(\pi^t A^t x^g - \pi^t b^t)^2 / \|\pi^t A^t\|^2.$$

Now, as in the proof of Theorem 3.3, we get

$$\|e^{g+1}\|^2 < \|e^g\|^2 - \lambda(2-\lambda)\epsilon^2/m^2.$$

From this, it follows that this algorithm terminates within $\bar{r}$ major cycles, where

$$\bar{r} \leq m^2 2^{2L-2}/n\lambda(2-\lambda)\epsilon^2. \qquad \square$$

5. Parallel Surrogate Constraint Method

The algorithm of the previous section can be modified so as to work on all the subsystems $A^t x \leq b^t$, $t = 1$ to p, simultaneously in parallel. We will call this version the parallel surrogate constraint method. This method is also initiated with some point $x^0 \in \mathbb{R}^n$. The general step $k+1$, for $k \geq 0$, is given below.

Step $k+1$. Let x^k be the point obtained at the end of the previous step. For each subsystem $t = 1$ to p, find $I^t(x^k)$ as defined in the previous section. If $I^t(x^k) = \varnothing$, define $P_t(x^k) = x^k$. If $I^t(x^k) \neq \varnothing$, select the weight vector π^t as in the previous section, and define $P_t(x^k) = x^k - d^k$, where

$$d^k = (\pi^t A^t x^k - \pi^t b^t)(\pi^t A^t)^T / \|\pi^t A^t\|^2.$$

If $I^t(x^k)=\varnothing$, for all $t=1$ to p, then x^k is feasible to (1); terminate. Otherwise, define

$$P(x^k)=\sum_{t=1}^{p}\tau_t P_t(x^k),$$

where τ_t are nonnegative numbers summing to 1 with $\tau_t>0$ for all t such that $I^t(x^k)\neq\varnothing$. Define

$$x^{k+1}=x^k+\lambda\,(P(x^k)-x^k), \tag{9}$$

where $0<\lambda<2$, and go to the next step.

Convergence Results.

Lemma 5.1. Let p be a positive integer, let $V_t\in\mathbb{R}^n$, for $t=1$ to p, and let

$$V=\sum_{t=1}^{p}\tau_t V_t, \qquad \sum_{t=1}^{p}\tau_t=1, \qquad 0\leq\tau_t\leq 1, \qquad \text{for all } t=1 \text{ to } p.$$

Then,

$$\|V\|^2\leq\sum_{t=1}^{p}\tau_t\|V_t\|^2.$$

Proof. We proceed by induction on p. For $p=2$,

$$V=\tau_1V_1+\tau_2V_2,$$

where

$$\tau_1+\tau_2=1, \qquad \tau_1\geq 0, \qquad \tau_2\geq 0.$$

So,

$$\|V\|^2=\|\tau_1V_1+\tau_2V_2\|^2=\tau_1^2\|V_1\|^2+\tau_2^2\|V_2\|^2+2\tau_1\tau_2(V_1)^TV_2.$$

Therefore,

$$\begin{aligned}
&\tau_1\|V_1\|^2+\tau_2\|V_2\|^2-\|V\|^2\\
&=\tau_1\|V_1\|^2+\tau_2\|V_2\|^2-\tau_1^2\|V_1\|^2-\tau_2^2\|V_2\|^2-2\tau_1\tau_2(V_1)^TV_2\\
&=\tau_1(1-\tau_1)\|V_1\|^2+\tau_2(1-\tau_2)\|V_2\|^2-2\tau_1\tau_2(V_1)^TV_2\\
&=\tau_1\tau_2\|V_1\|^2+\tau_2\tau_1\|V_2\|^2-2\tau_1\tau_2(V_1)^TV_2\\
&=\tau_1\tau_2\|V_1-V_2\|^2\geq 0.
\end{aligned}$$

It follows that

$$\|V\|^2\leq\tau_1\|V_1\|^2+\tau_2\|V_2\|^2.$$

Hence, the assertion is valid for $p=2$.

Induction Hypothesis. Assume that the assertion is valid for $p-1$. We will prove that the assertion is also valid for p under the induction hypothesis. Let

$$\bar{V}=\sum_{t=1}^{p-1}\tau_t V_t/(1-\tau_p),$$

$$V=\sum_{t=1}^{p}\tau_t V_t=\sum_{t=1}^{p-1}\tau_t V_t+\tau_p V_p=(1-\tau_p)\bar{V}+\tau_p V_p.$$

From the above argument, we have

$$\begin{aligned}\|V\|^2 &\le \tau_p\|V_p\|^2+(1-\tau_p)\|\bar{V}\|^2\\ &\le \tau_p\|V_p\|^2+(1-\tau_p)\sum_{t=1}^{p-1}\tau_t\|V_t\|^2/(1-\tau_p)=\sum_{t=1}^{p}\tau_t\|V_t\|^2,\end{aligned}$$

by the induction hypothesis. So, the assertion is valid for all finite positive integers p. □

Theorem 5.1. In this algorithm, if $x^{k+1}\neq x^k$, then

$$\|x-x^{k+1}\|<\|x-x^k\|, \qquad \text{for all } x\in K.$$

Proof. We have

$$x^{k+1}=x^k-\lambda\sum_{t=1}^{p}\tau_t(\pi^t A^t x^k-\pi^t b^t)(\pi^t A^t)^T/\|\pi^t A^t\|^2.$$

Let $e^k=x^k-x$,

$$V_t=(\pi^t A^t x^k-\pi^t b^t)(\pi^t A^t)^T/\|\pi^t A^t\|^2,$$

$$V=\sum_{t=1}^{p}\tau_t(\pi^t A^t x^k-\pi^t b^t)(\pi^t A^t)^T/\|\pi^t A^t\|^2=\sum_{t=1}^{p}\tau_t V_t,$$

$$e^{k+1}=e^k-\lambda\sum_{t=1}^{p}\tau_t(\pi^t A^t x^k-\pi^t b^t)(\pi^t A^t)^T/\|\pi^t A^t\|^2=e^k-\lambda V,$$

$$\begin{aligned}\|e^{k+1}\|^2 &= \|e^k\|^2+\lambda^2 V^T V-2\lambda V^T e^k\\ &= \|e^k\|^2+\lambda^2\|V\|^2-2\lambda\sum_{t=1}^{p}\tau_t(\pi^t A^t x^k-\pi^t b^t)\\ &\qquad\times[\pi^t A^t(x^k-x)]/\|\pi^t A^t\|^2\\ &= \|e^k\|^2+\lambda^2\|V\|^2-2\lambda\sum_{t=1}^{p}\tau_t(\pi^t A^t x^k-\pi^t b^t)\\ &\qquad\times[\pi^t(A^t x^k-b^t)]/\|\pi^t A^t\|^2\end{aligned}$$

$$+2\lambda \sum_{t=1}^{p} \tau_t(\pi^t A^t x^k - \pi^t b^t)[\pi^t(A^t x - b^t)]/\|\pi^t A^t\|^2$$

$$\leq \|e^k\|^2 + \lambda^2 \|V\|^2 - 2\lambda \sum_{t=1}^{p} \tau_t(\pi^t A^t x^k - \pi^t b^t)^2/\|\pi^t A^t\|^2$$

$$= \|e^k\|^2 + \lambda^2 \|V\|^2 - 2\lambda \sum_{t=1}^{p} \tau_t \|V_t\|^2.$$

From Lemma 5.1, we have

$$\lambda^2 \|V\|^2 \leq \lambda^2 \sum_{t=1}^{p} \tau_t \|V_t\|^2,$$

and so,

$$\|e^{k+1}\|^2 \leq \|e^k\|^2 - \lambda(2-\lambda) \sum_{t=1}^{p} \tau_t \|V_t\|^2 < \|e^k\|^2. \tag{10}$$

□

Theorem 5.2. If $K \neq \varnothing$, any sequence $\{x^k\}_{k=1}^{\infty}$ generated by this algorithm has the property

$$\lim_{k\to\infty} \phi(x^k) = 0.$$

Proof. For any $x \in K$, Theorem 5.1 implies that the sequence $\{\|x^k - x\|\}_{k=1}^{\infty}$ is monotonically decreasing, hence it converges. It follows from (10) that

$$\lim_{k\to\infty} \sum_{t=1}^{p} \tau_t(\pi^t A^t x^k - \pi^t b^t)^2/\|\pi^t A^t\|^2 = 0. \tag{11}$$

Since $\tau_t > 0$ for all t such that $I^t(x^k) \neq \varnothing$, and since the τ_t sum to 1, relation (11) implies that

$$\lim_{k\to\infty} (\pi^t A^t x^k - \pi^t b^t) = 0, \qquad \text{for all such } t.$$

Since for each subsystem, π_j^t are strictly positive for all $j \in I^t(x^k)$, and sum to 1, this implies that either

$$\lim_{k\to\infty} (A_j^t x^k - b_j^t) = 0, \qquad \text{for all } j \in I^t(x^k),$$

and for all subsystems, $t = 1$ to p, or there exists a $\bar{k} < \infty$ such that

$$I^t(x^{\bar{k}}) = \varnothing, \qquad \text{for all } t = 1 \text{ to } p.$$

Therefore,

$$\lim_{k\to\infty} \phi(x^k)=0. \qquad \square$$

For the sake of simplicity, the following finite convergence proofs will be based on the assumption that τ_t are all equal for all $t=1$ to p, that is,

$$\tau_t=1/p, \qquad \text{for all } t.$$

In practice, a different choice may yield better computational performance. This has to be determined in a computational experiment.

Theorem 5.3. Let $\epsilon>0$ be a specified error tolerance, if $K\neq\varnothing$, and we define

$$I^t(x^k)=\{i: A_i^t x^k - b_i^t > \epsilon\}$$

in this algorithm. Then, it converges to a point $x\in K_\epsilon$ in a finite number of steps.

Proof. We proceed as in the proof of Theorem 3.3. From Theorem 5.1, we know that, if $x^k\notin K_\epsilon$, then

$$\|x-x^{k+1}\|<\|x-x^k\|, \qquad \text{for all } x\in K.$$

Define $e^k=x^k-x$. Since $\{\|e^k\|\}$ is monotonically decreasing and bounded below, it converges. This implies that

$$\lim_{k\to\infty}\|e^{k+1}\|=\lim_{k\to\infty}\|e^k\|.$$

But this happens only if there exist a $\bar{k}<\infty$ such that, at the $\bar{k}$th iteration,

$$I^t(x^{\bar{k}})=\varnothing, \qquad \text{for all } t=1 \text{ to } p.$$

Otherwise for all k, there exists at least one subsystem t such that $I^t(x^k)\neq\varnothing$, then $\pi^t A^t x^k-\pi^t b^t>\epsilon$. Since

$$\|e^{k+1}\|^2\leq\|e^k\|^2+(\lambda^2-2\lambda)\sum_{t=1}^{p}\tau_t(\pi^t A^t x^k-\pi^t b^t)^2/\|\pi^t A^t\|^2,$$

this implies that

$$\lim_{k\to\infty}\|e^k\|\neq\lim_{k\to\infty}\|e^{k+1}\|,$$

a contradiction. So, this version of the algorithm must converge to a point in K_ϵ in a finite number $\bar{k}$ of steps.

From Lemma 2.1, it follows that

$$\|e^0\| \leq 2^{L-1}/\sqrt{n}.$$

If $I^t(x^k) \neq \varnothing$ for at least one subsystem, say, the tth subsystem, then from (10) we get

$$\|e^{k+1}\|^2 \leq \|e^k\|^2 - \lambda(2-\lambda) \sum_{t=1}^{p} \tau_t(\pi^t A^t x^k - \pi^t b^t)^2 / \|\pi^t A^t\|^2$$

$$\leq \|e^k\|^2 - \lambda(2-\lambda)\tau_t(\pi^t A^t x^k - \pi^t b^t)^2 / \|\pi^t A^t\|^2.$$

Since $\tau_t = 1/p < 1/m$, we have

$$\pi^t A^t x^k - \pi^t b^t \geq \left(\sum_i \pi_i\right) \cdot \epsilon = \epsilon,$$

$$\|\pi^t A^t\| \leq \|\pi^t\|_2 \|A^t\|_2 \leq \|\pi^t\|_2 \|A^t\|_F < m.$$

It follows that

$$\|e^{k+1}\|^2 < \|e^k\|^2 - \lambda(2-\lambda)\epsilon^2/m^3.$$

Therefore, this version of the algorithm converges to a point in K_ϵ within $\bar{k}$ steps, where

$$\bar{k} \leq m^3 2^{2L-2}/(n\lambda(2-\lambda)\epsilon^2). \qquad \square$$

In the parallel method discussed above, we obtained for the tth subsystem a surrogate constraint

$$\pi^t A^t x \leq \pi^t b^t \tag{12}$$

and a point $P_t(x^k)$ by projecting x^k onto this surrogate hyperplane, for each $t = 1$ to p such that $I^t(x^k) \neq \varnothing$. The new point x^{k+1} is derived from a weighted average of these $P_t(x^k)$. So, this method can be viewed as a Cimmino-type method using groups of constraints, instead of individual constraints, and surrogation within each group. Another parallel method would just obtain the surrogaté constraint (12) for each subsystem t such that $I^t(x^k) \neq \varnothing$. Then, it would take a positive combination of all such surrogate constraints generated, leading to a surrogate constraint for the entire original system (1). If the weight assigned to t is $\delta_t > 0$, this constraint will be

$$\sum (\delta_t(\pi^t A^t x\text{: over } t \text{ such that } I^t(x^k) \neq \varnothing)$$

$$\leq \sum \delta_t(\pi^t b^t\text{: over } t \text{ such that } I^t(x^k) \neq \varnothing). \tag{13}$$

The point $P(x^k)$ is then defined to be the orthogonal projection of x^k onto (13) treated as an equation, and the next point x^k is obtained as in (9) using

this $P(x^k)$. This method is essentially the algorithm of Section 3 using a parallel implementation for identifying all the violated constraints, with a different processor examining the constraints in each subsystem.

6. Comparisons with Earlier Methods

Let us compare the surrogate constraint methods with the relaxation method for solving linear inequalities as well as Cimmino's method. In the relaxation method, at each iteration an orthogonal projection is made from the current point x^k onto an individual K_i for some i. However, K_i only contains the information in one constraint. Sometimes, the projection on K_i offers little improvement in reducing the distance from the current point x^k to the set K. On the other hand, the surrogate hyperplane contains information from more than one violated constraint, so it is expected to generate a better new point than the relaxation method. See Fig. 3.

Cimmino's method for linear inequalities identifies all violated constraints at each iteration. Orthogonal projections are made simultaneously onto all violated constraints from the current point, and the new point is a convex combination of those projection points. See Fig. 3.

Computational experiments have been carried out to compare the sequential surrogate constraint method with the version of the relaxation method that processes the inequalities in cyclical order. We give below our

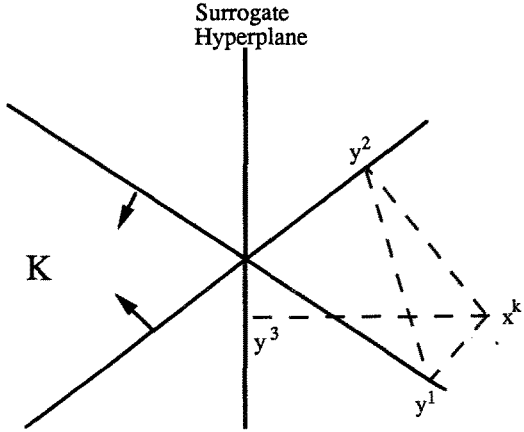

Fig. 3. Comparison of the surrogate constraint method (SCM) with other methods. For $\lambda = 1$, the relaxation method yields either the point y^1 or y^2; Cimmino's method yields a point on the line segment joining y^1 and y^2; SCM yields the point y^3.

preliminary computational results on randomly generated large sparse problems carried out on the IBM 3090-400/VM mainframe computer at the University of Michigan. The problems are generated in such a way that the system would have an interior feasible solution. The sequential surrogate constraint method is implemented using the following weights for generating the surrogate constraint. When operating on subsystem t with the current point x^k, if $I^t(x^k) \neq \emptyset$, we take

$$\pi^t = (\pi_i^t : i \in I^t(x^k)),$$

where

$$\pi_i^t = 0.2 r_i^t \Big/ \left(\sum r_i^t : \text{over } i \in I^t(x^k) \right) + 0.8/|I^t(x^k)|,$$

$$r_i^t = A_i^t x^k - b_i^t, \qquad \text{for all } i \in I^t(x^k).$$

The value of λ for both methods was taken to be 1.7.

The results are listed in Table 1. Five test problems were generated in each dimension and solved to the accuracy 10^{-9}. The speedup of the surrogate constraint method over the relaxation method ranged from 30 to 60. The speedup increases as the problem size increases.

In the relaxation method, in each sweep, all the constraints are examined once from top to bottom. The average number of sweeps before termination varied from 5 to 10 among the problem sizes. Since the current point changes after each projection, it is not possible to implement a sweep in this method in a parallel fashion.

Table 1. Comparison of the relaxation method and the sequential surrogate constraint method.

Problem size		Sparsity	Relaxation method			Sequential surrogate constraint method			
Rows	Columns	%	NP	NS	T	NSS	NR	T	NC
5,000	2,500	2.0	10,500	4.9	17.04	2	2500	0.511	3.4
5,000	5,000	1.0	10,675	5.2	23.49	2	2500	0.544	3.2
10,000	2,500	1.0	22,375	6.3	45.11	5	2000	0.806	2.7
10,000	5,000	0.4	20,985	5.9	54.06	5	2000	1.146	3.7
10,000	10,000	0.4	23,125	7.1	84.22	5	2000	1.371	2.8
18,000	5,000	0.5	41,125	8.4	213.00	9	2000	3.332	3.4
18,000	9,000	0.2	44,750	9.3	255.13	9	2000	4.002	3.8

NP = average number of projections; NSS = number of subsystems;
NS = number of sweeps; NR = number of rows in each subsystem;
T = average CPU time (sec); NC = number of major cycles.

In the sequential surrogate constraint method, in each major cycle, the number of projections made is at most equal to the number of subsystems. In each major cycle, each constraint is examined once, but as explained earlier, this work can easily be parallelized. Also, the number of major cycles needed in the surrogate constraint method is much less than the number of sweeps needed in the relaxation method to achieve the same accuracy.

These computational results are very encouraging. More extensive experimentation is necessary to determine the strategies to implement the surrogate constraint methods for obtaining the best performance, things such as the best choice for the weight vector in each step, etc.

7. Extensions to Linear Equations

It is easy to modify the surrogate constraint methods to solve a system of linear equations,

$$Ax = b, \tag{14}$$

by applying these methods on the following equivalent systems of linear inequalities

$$\begin{aligned} &Ax \leq b, \\ &-Ax \leq -b. \end{aligned} \tag{15}$$

Many of the classical iterative methods (such as the successive approximation method, the Gauss–Seidel method, the SOR method, and the steepest descent method) may not always converge for an arbitrary coefficient matrix A. Some methods require A to be positive definite or diagonally dominant, otherwise those methods would have to be applied to the system

$$A^T Ax = A^T b.$$

In the case of successive approximations, convergence requires that the spectral radius of an approximation matrix be less than one.

The surrogate constraint methods only require that the system (14) be feasible. This is one advantage of the surrogate constraint methods over the classical iterative methods.

For each i, the system (15) has both the constraints $A_i x \leq b_i$ and $A_i x \geq b_i$. When x^k is the current point, if $A_i x^k = b_i$, both these constraints are satisfied. Otherwise, $A_i x^k \neq b_i$, and exactly one of the constraints in the above is violated, while the other one is satisfied. Thus, when x^k is the current point, the set of violated constraints in (15) includes at most one of the constraints from the pair $A_i x \leq b_i$ and $A_i x \geq b_i$. Using this, simplifications can be made in executing the surrogate constraint methods on the system (15).

References

1. CENSOR, Y., and HERMAN, G. T., *On Some Optimization Techniques in Image Reconstruction from Projections*, Applied Numerical Mathematics, Vol. 3, pp. 365–391, 1987.
2. FLEMING, H. E., *Satellite Remote Sensing by the Technique of Computerized Tomography*, Journal of Applied Meteorology, Vol. 21, pp. 1538–1549, 1982.
3. ANDERSON, D. L., and DZIEWONSKI, A. M., *Seismic Tomography*, Scientific American, Vol. 251, pp. 58–66, 1984.
4. KARMARKAR, N., *A New Polynomial Algorithm for Linear Programming*, Combinatorica, Vol. 4, pp. 373–395, 1984.
5. MURTY, K. G., *Linear Complementarity, Linear and Nonlinear Programming*, Heldermann-Verlag, Berlin, Germany, 1988.
6. AGMON, S., *The Relaxation Method for Linear Inequalities*, Canadian Journal of Mathematics, Vol. 6, pp. 382–392, 1954.
7. MOTZKIN, T. S., and SCHOENBERG, I. T., *The Relaxation Method for Linear Inequalities*, Canadian Journal of Mathematics, Vol. 6, pp. 393–404, 1954.
8. KACZMARZ, S., *Angenherte Auflosung von Systemn Linearer Gleichungen*, Bulletin de l'Academie Polonaise des Sciences et des Lettres, Classe des Sciences Mathematiques et Naturelles, Serie A, Sciences Mathematiques, Vol. 35, pp. 355–357, 1937.
9. BREGMAN, L. M., *The Method of Successive Projection for Finding a Common Point of Convex Sets*, Soviet Mathematics Doklady, Vol. 6, pp. 688–692, 1965.
10. BREGMAN, L. M., *The Relaxation Method of Finding the Common Point of Convex Sets and Its Application to the Solution of Problems in Convex Programming*, USSR Computational Mathematics and Mathematical Physics, Vol. 3, pp. 200–217, 1967.
11. EREMIN, I. I., *The Relaxation Method of Solving Systems of Inequalities with Convex Functions on the Left Side*, Soviet Mathematics Doklady, Vol. 6, pp. 219–222, 1965.
12. GUBIN, L. G., POLYAK, B. T., and RAIK, E. V., *The Method of Projections for Finding the Common Point of Convex Sets*, USSR Computational Mathematics and Mathematical Physics, Vol. 6, pp. 1–24, 1967.
13. CIMMINO, G., *Calcolo Approssimato per le Soluzioni dei Sistemi di Equazioni Lineari*, Ricerca Scientifica, Vol. 1, pp. 326–333, 1938.
14. CENSOR, Y., and ELFVING, T., *New Method for Linear Inequalities*, Linear Algebra and Its Applications, Vol. 42, pp. 199–211, 1982.
15. DE PIERRO, A. R., and IUSEM, A. N., *A Simultaneous Projections Method for Linear Inequalities*, Linear Algebra and Its Applications, Vol. 64, pp. 243–253, 1985.
16. GACS, P., and LOVASZ, L., *Khachiyan's Algorithm for Linear Programming*, Mathematical Programming Study, Vol. 14, pp. 61–68, 1981.
17. MURTY, K. G., *Linear Programming*, John Wiley and Sons, New York, New York, 1983.

18. CENSOR, Y., *Row-Action Methods for Huge and Sparse Systems and Their Applications*, SIAM Review, Vol. 23, No. 4, pp. 444–466, 1981.
19. TELGEN, J., *On Relaxation Methods for System of Linear Inequalities*, European Journal of Operational Research, Vol. 9, pp. 184–189, 1982.