SPARSITY-PRESERVING SOR ALGORITHMS FOR SEPARABLE QUADRATIC AND LINEAR PROGRAMMING

by

O. L. Mangasarian

Computer Sciences Technical Report #438

July 1981

Errata for TR #438

Page 5: Add to equation (5):

if $M_{jj} > 0$; if $M_{jj} \leq 0$ replace M_{jj}^{-1} by 1.

Page 7: Add to first sentence of page:

and no row of A is identically zero.

Page 11: Add after first sentence of page:

To avoid trivial cases, we shall assume that each row and column of A, as well as the vector $\binom{c}{b}$ is nonzero.

Page 12, Line 7: Section 3 should read Section 2.

Page 15, Equation (32): Add left paranthesis to the left of u_j^i.

Revised March 1982

Sparsity-Preserving SOR Algorithms for Separable Quadratic and Linear Programming

O. L. Mangasarian

ABSTRACT

The main purpose of this work is to give explicit sparsity-preserving SOR (successive overrelaxation) algorithms for the solution of separable quadratic and linear programming problems. The principal and computationally-distinguishing feature of the present SOR algorithms is that they preserve the sparsity structure of the problem and do not require the computation of the product of the constraint matrix by its transpose as is the case in earlier SOR algorithms for linear and quadratic programming.

AMS (MOS) Subject Classifications: 90C05, 90C20, 65F10

Key Words: Linear programming, quadratic programming, SOR methods, relaxation methods

---

Sponsored by the United States Army under Contract No. DAAG29-80-C-0041. This material is based upon work supported by the National Science Foundation under Grant No. MCS-790166.

1. Introduction

Recently iterative SOR methods have received widespread attention in the solution of the symmetric and nonsymmetric linear complementarity problem [3,11,15,16,1], quadratic and linear programming problems [4,6,12,13]. In the case of the latter two problems which are our principal concerns here, the recently proposed SOR algorithms do not preserve any sparsity that the original problems may have had. This is due to the fact that algorithms as presented in [12] require the product of the constraint matrix by its transpose, which can cause loss of both sparsity and accuracy. In this work we shall present some explicit realizations of the algorithms of [12,13] which will not require the multiplication of the constraint matrix by its transpose. These computationally improved realizations which follow from the algorithms of [12] have not been given explicitly before. The absence of such sparsity-preserving algorithms has been a critical factor in preventing the application of SOR methods to many large important but highly structured problems such as economic equilibrium problems, transportation and network flow problems. In addition some of the present realizations of the SOR algorithms (e.g. (14) and (32) below) require only simple operations on the rows of the constraint matrix, and hence very large problems can be tackled by such SOR realizations, because only linear row arrays are needed in the computations. These advantages become even more pronounced if these linear row arrays are sparse and hence can be stored in packed form.

The paper is organized as follows. In Section 2 we give an SOR algorithm for the symmetric linear complementarity problem or equivalently

for the quadratic programming problem with nonnegativity constraints only. This is a special case of the general algorithm presented in [11] but given here, in a simple explicit form in terms of the rows of the matrix defining the problem, principally to make it preserve problem sparsity. In Section 3 we consider a separable quadratic programming problem and give a version of the SOR algorithm of [12] which does not require multiplication of the constraint matrix by its transpose. Hence this present form of the algorithm is now ideally suited for large sparse problems. In Section 4 two sparsity-preserving SOR algorithms for linear programming are given. One is based on finding the "smallest" optimal primal-dual solution (LPSOR1) [13] and the other is based on perturbing a linear program to a separable quadratic program and then solve the latter by the method of Section 3 (LPSOR2) [12]. Computational experience with a version of LPSOR1 [9] on problems of size up to 800 constraints and 1000 variables and the non-sparsity-preserving version of LPSOR2 [12] have been very encouraging. It is hoped that further refinements will make SOR methods simple, robust and commercially viable methods for solving very large separable quadratic and linear programs.

We briefly describe now the notation used. All matrices and vectors are real. For the $m\times n$ matrix A we write $A \in R^{m\times n}$ and denote row i by $A_{i\cdot}$, column j by $A_{\cdot j}$ and the element in row i and column j by A_{ij}. For x in the real n-dimensional Euclidean space R^n, element i is denoted by x_i and x_+ will denote the vector with components $(x_+)_i = \max\{x_i,0\}$, $i=1,\ldots,n$. All vectors are column vectors unless transposed by the superscript T. $\|x\|$ will denote the 2-norm,

$(x^Tx)^{\frac{1}{2}} = (\sum_{j=1}^{n} x_j^2)^{\frac{1}{2}}$. A matrix C in $R^{n\times n}$ is positive semidefinite if $x^TCx \geq 0$ for all x in R^n and positive definite if $x^TCx > 0$ for all nonzero x in R^n. For brevity we shall sometimes omit mentioning the dimensionality of a vector or matrix, it being obvious from the context. The vector e will be a vector ones in a Euclidean space of appropriate dimension. For a twice differentiable function $\phi: R^m \times R^n \to R$, $\nabla_u\phi(u,v)$ will denote the $m\times 1$ gradient vector with elements $\frac{\partial\phi(u,v)}{\partial u_i}$, $i=1,\ldots,m$, $\nabla_v\phi(u,v)$ will denote the $n\times 1$ gradient vector with elements $\frac{\partial\phi(u,v)}{\partial v_i}$, $i=1,\ldots,n$, $\nabla\phi(u,v) = \begin{bmatrix}\nabla_u\phi(u,v)\\ \nabla_v\phi(u,v)\end{bmatrix}$, and $\nabla^2\phi(u,v)$ will denote the Hessian in $R^{(n+m)\times(n+m)}$ with submatrix components denoted as follows

$$\nabla^2\phi(u,v) = \begin{bmatrix}\nabla_{uu}\phi(u,v) & \nabla_{uv}\phi(u,v)\\ \nabla_{vu}\phi(u,v) & \nabla_{vv}\phi(u,v)\end{bmatrix}$$

2. SOR Algorithm for the Symmetric Linear Complementarity Problem

We consider here the problem of finding z in R^k such that

$$Mz + q \geq 0,\ z \geq 0,\ z^T(Mz+q) = 0 \tag{1}$$

where M is a symmetric matrix in $R^{k\times k}$ and $q \in R^k$. Conditions (1) are [10] the necessary optimality conditions for the quadratic programming problem

$$\underset{z\in R^k}{\text{minimize}}\ \tfrac{1}{2}z^TMz + q^Tz \quad \text{subject to} \quad z \geq 0 \tag{2}$$

Conditions (1) are sufficient for z to solve (2) whenever M is positive semidefinite [10].

In [4,6,11] iterative SOR methods have been proposed for solving (1), but without paying any special attention to possible sparsity that the problem may have. We give below a sparsity-preserving SOR algorithm based on that of [11]. Our proofs here depend intimately on the results of this reference. If we define

$$\theta(z) := \tfrac{1}{2}z^TMz + q^Tz \tag{3}$$

then the SOR algorithm for solving (2) can be represented as a gradient projection algorithm of the following type

$$z_j^{i+1} = \left(z_j^i - \omega(\nabla^2\theta(z^i))_{jj}^{-1}\nabla_{z_j}\theta(z_1^{i+1},\ldots,z_{j-1}^{i+1},z_j^i,\ldots,z_k^i)\right)_+ \qquad j=1,\ldots,k \tag{4}$$

where ω is the relaxation factor or stepsize that must be in the open interval $(0,2)$ and i represents the ith iteration. More specifically we have the following.

LCPSOR Algorithm

Choose $z^0 \in R^n_+$, $\omega \in (0,2)$. Having z^i compute z^{i+1} as follows:

$$z_j^{i+1} = (z_j^i - \omega M_{jj}^{-1}(\underset{\text{for } j>1}{\sum_{\ell=1}^{j-1}} M_{j\ell} z_\ell^{i+1} + \underset{j=1,\ldots,k}{\sum_{\ell=j}^{k}} M_{j\ell} z_\ell^i + q_j))_+ \quad \text{if } M_{jj} > 0 \qquad (5)$$

If $M_{jj} \leq 0$, set M_{jj}^{-1} to 1 in (5).

The following convergence theorem follows directly from [11].

Theorem 1: LCPSOR Convergence

(i) Let M be symmetric. Each accumulation point of (5) solves (1). If in addition M is positive semidefinite then each accumulation point of (5) solves (2) as well.

(ii) Let M be symmetric and positive semidefinite and such that

$$Mz + q > 0 \quad \text{for some } z \in R^n \qquad (6)$$

Then the sequence $\{z^i\}$ of the LCPSOR algorithm (5) is bounded and has an accumulation point that solves both (1) and (2).

(iii) Let M be symmetric and positive semidefinite and such that problem (1) (or equivalently problem (2)) has a nonempty bounded solution set. Then the sequence $\{z^i\}$ of the LPSOR algorithm (5) is bounded and has an accumulation point that solves both (1) and (2).

(iv) Let M be symmetric and positive definite. Then the sequence $\{z^i\}$ of the LCPSOR algorithm (5) converges to the unique solution $\bar{z}$ of (1) and (2).

Proof

Parts (i), (ii) and (iv) follow from Theorem 2.1, Theorem 2.2 and Corollary 2.2 of [11] respectively. To establish (iii) we note that from Lemma 2.3(b) of [11] that if the sequence $\{z^i\}$ of (5) is unbounded then there exists a $\bar{y} \in R^k$ such that

$$0 \neq \bar{y} \geqq 0, \; M\bar{y} = 0, \; q^T\bar{y} \leqq 0$$

This contradicts the boundedness assumption on the solution set of (1) since if $\bar{z}$ solves (1) then $\bar{z} + \lambda\bar{y}$ also solves (1) for all $\lambda \geqq 0$ because $\bar{z} + \lambda\bar{y} \geqq 0$, $M(\bar{z}+\lambda\bar{y}) + q \geqq 0$ and

$$0 \leqq (\bar{z}+\lambda\bar{y})^T(M(\bar{z}+\lambda\bar{y})+q) = \lambda q^T\bar{y} \leq 0. \qquad \square$$

3. SOR Algorithm for Separable Quadratic Programming

We consider here the separable quadratic program

$$\underset{x\in R^n}{\text{minimize}} \ \frac{1}{2}x^TDx + c^Tx \quad \text{subject to} \quad Ax \geq b, \ x \geq 0 \tag{7}$$

where D is a positive diagonal matrix in $R^{n\times n}$, $A \in R^{m\times n}$, $c \in R^n$, $b \in R^m$ and no row of A is identically zero. For more general quadratic programs see [12]. Associated with this quadratic program is the dual quadratic program [5,17,10]

$$\underset{(x,u,v)\in R^{n+m+n}}{\text{maximize}} \ -\frac{1}{2}x^TDx + b^Tu \quad \text{subject to} \quad Dx - A^Tu - v + c = 0, \ (u,v) \geq 0 \tag{8}$$

which upon elimination of x by using the constraint relation

$$x = D^{-1}(A^Tu+v-c) \tag{9}$$

gives

$$\underset{(u,v)\in R^{m+n}}{\text{minimize}} \ \frac{1}{2}(A^Tu+v-c)^TD^{-1}(A^Tu+v-c) - b^Tu \quad \text{subject to} \quad (u,v) \geq 0 \tag{10}$$

This problem (10) is now precisely of the form (2) and the LCPSOR algorithm (5) can be applied to it easily. Because our principal interest here is sparsity preserving we shall spell out the algorithm for solving (10) explicitly. Define the objective function of (10) as

$$\phi(u,v) := \frac{1}{2}(A^Tu+v-c)^TD^{-1}(A^Tu+v-c) - b^Tu \tag{11}$$

then

$$\nabla\phi(u,v) = \begin{bmatrix} AD^{-1}(A^Tu+v-c) - b \\ \\ D^{-1}(A^Tu+v-c) \end{bmatrix} \tag{12}$$

and

$$\nabla^2\phi(u,v) = \begin{bmatrix} AD^{-1}A^T & AD^{-1} \\ D^{-1}A^T & D^{-1} \end{bmatrix} \tag{13}$$

Now the SOR algorithm for solving (10) can be stated as

$$u_j^{i+1} = (u_j^i - \frac{\omega}{(\nabla_{uu}\phi(u^i,v^i))_{jj}} \nabla_{u_j}\phi(u_1^{i+1},\ldots,u_{j-1}^{i+1},u_j^i,\ldots,u_m^i,v^i))_+$$
$$j=1,\ldots,m$$

$$v_j^{i+1} = (v_j^i - \frac{\omega}{(\nabla_{vv}\phi(u^i,v^i))_{jj}} \nabla_{v_j}\phi(u^{i+1},v_1^{i+1},\ldots,v_{j-1}^{i+1},v_j^i,\ldots,v_n^i))_+$$
$$j=1,\ldots,n$$

where ω is a relaxation factor in $(0,2)$. More specifically we have the following.

QPSOR Algorithm

Choose $(u^0,v^0) \in R_+^{m+n}$, $\omega \in (0,2)$. Having (u^i,v^i) compute (u^{i+1},v^{i+1}) as follows:

$$u_j^{i+1} = (u_j^i - \frac{\omega}{\|A_jD^{-\frac{1}{2}}\|^2}(A_jD^{-1}(\sum_{\substack{\ell=1 \\ \text{for } j>1}}^{j-1}(A^T)_{\cdot\ell}u_\ell^{i+1} + \sum_{\ell=j}^{m}(A^T)_{\cdot\ell}u_\ell^i + v^i - c) - b_j))_+$$
$$j=1,\ldots,m \tag{14}$$

$$v^{i+1} = (v^i - \omega(A^Tu^{i+1} + v^i - c))_+$$

Note that any sparsity or structural properties that the matrix A may have are not destroyed in the QPSOR algorithm as would be the case in [12], and in fact may be taken advantage of in the present algorithm.

Remark 2

The iteration (14) is very well suited for matrices A which have a pronounced row structure, for example if A is sparse and the nonzero elements of each row can be easily located without search. On the other hand if the matrix A has a pronounced column structure, then the following alternate but equivalent iteration to (14) may be preferrable:

$$u_j^{i+1} = (u_j^i - \frac{\omega}{\|A_j D^{-\frac{1}{2}}\|^2}((u_1^{i+1}....u_{j-1}^{i+1},u_j^i....u_m^i)A+v^{iT}-c^T)D^{-1}(A^T)_{\cdot j}-b_j))_+$$

$$j=1,....,m \qquad (14')$$

$$v_j^{i+1} = (v_j^i-\omega(u^{i+1^T}A_{\cdot j}+v_j^i-c_j))_+ \quad , \; j=1,....,n$$

Theorem 2: QPSOR Convergence

(i) Each accumulation point (u,v) of the sequence $\{(u^i,v^i)\}$ generated by the QPSOR algorithm (14) solves (10), and the corresponding x determined by (9) solves the quadratic program (7).

(ii) Let the feasible region of the quadratic program (7) satisfy the Slater constraint qualification

$$\{x|Ax>b,\; x>0\} \neq \emptyset \qquad (15)$$

Then the sequence $\{(u^i,v^i)\}$ of the QPSOR algorithm (14) is bounded and has an accumulation point (u,v) and the corresponding x determined by (9) solves (7).

Proof

(i) Follows from Theorem 1(i) and the duality theory of quadratic programming [10, Theorem 8.2.5].

(ii) Because of (15) there exist a $\delta > 0$ such that the perturbed positive definite quadratic program

$$\underset{x\in R^n}{\text{minimize}} \ \tfrac{1}{2}x^TDx + c^Tx \quad \text{subject to} \quad Ax \geq b + e\delta,\ x \geq e\delta$$

has a solution $\tilde{x} \in R^n$ with corresponding multipliers $(\tilde{u},\tilde{v}) \in R^{m+n}$ that satisfy the Karush-Kuhn-Tucker conditions

$$D\tilde{x} + c - A^T\tilde{u} - \tilde{v} = 0,\ A\tilde{x} \geq b + e\delta,\ \tilde{x} \geq e\delta,\ \tilde{u} \geq 0,\ \tilde{v} \geq 0$$

$$\tilde{u}^T(A\tilde{x}-b-e\delta) = 0,\ \tilde{v}^T(\tilde{x}-e\delta) = 0$$

Hence

$$\begin{aligned} \tilde{x} &= D^{-1}(A^T\tilde{u}+\tilde{v}-c) \geq e\delta > 0 \\ AD^{-1}(A^T\tilde{u}+\tilde{v}-c)-b &\geq e\delta > 0 \end{aligned} \tag{16}$$

Conditions (16) are equivalent to condition (6) for problem (10). Hence by Theorem 1(ii) the sequence $\{(u^i,v^i)\}$ of the QPSOR algorithm (14) is bounded and has an accumulation point (u,v) which solves (10). Hence the corresponding x determined by (9) solves (7). □

4. SOR Algorithm for Linear Programming

We consider finally the dual linear programs

$$\underset{x\in R^n}{\text{minimize}} \quad c^T x \qquad \text{subject to} \quad Ax \geq b,\ x \geq 0 \tag{17}$$

and

$$\underset{u\in R^m}{\text{maximize}} \quad b^T u \qquad \text{subject to} \quad A^T u \leq c,\ u \geq 0 \tag{18}$$

where $A \in R^{m\times n}$, $c \in R^n$, and $b \in R^m$. To avoid trivial cases, we shall assume that each row and column of A, as well as the vector $\binom{c}{b}$ is nonzero. It is well known [2] that solving either (17) or (18) is equivalent to solving both (17) and (18) which in turn is equivalent to solving the linear complementarity problem

$$Ny + p \geq 0,\ y \geq 0,\ y^T(Ny+p) = y^T p = 0 \tag{19}$$

where

$$N = \begin{pmatrix} 0 & -A^T \\ A & 0 \end{pmatrix}, \quad p = \begin{pmatrix} c \\ -b \end{pmatrix}, \quad y = \begin{pmatrix} x \\ u \end{pmatrix} \in R^k,\ k = n + m \tag{20}$$

Note that N is skew symmetric, that is $N + N^T = 0$ and hence $y^T N y = 0$. As proposed in [13] one way of solving the linear program (17) is to find the closest point to the origin, in the 2-norm, of the solution set of (19). That is we shall solve the quadratic program

$$\underset{y\in R^k}{\text{Minimize}} \quad \frac{1}{2}\|y\|^2 \qquad \text{subject to} \quad Ny + p \geq 0,\ y \geq 0,\ p^T y \leq 0 \tag{21}$$

Note that under the constraints $Ny + p \geq 0$, $y \geq 0$, the constraint $p^T y \leq 0$ is equivalent to $p^T y = 0$ since $0 \geq p^T y = y^T(Ny+p) \geq 0$. The dual to the quadratic problem (21) is [5,17,10]

$$\underset{(y,s,t,\beta)\in R^{3k+1}}{\text{maximize}} \quad -\frac{1}{2}\|y\|^2 - p^T s \quad \text{subject to} \quad y - N^T s - t + \beta p = 0, \ (s,t,\beta) \geq 0 \tag{21}$$

Elimination of y by using the constraint relation

$$\begin{pmatrix} x \\ u \end{pmatrix} = y = N^T s + t - \beta p \tag{22}$$

gives the quadratic program

$$\underset{(s,t,\beta)\in R^{2k+1}}{\text{minimize}} \quad \frac{1}{2}\|N^T s - \beta p + t\|^2 + p^T s \quad \text{subject to} \quad (s,t,\beta) \geq 0 \tag{23}$$

Problem (23) and consequently problem (21) can be solved by the SOR method of Section 2. For that purpose it is convenient to let $\phi(s,t,\beta)$ equal the objective function of (23) that is

$$\phi(s,t,\beta) := \frac{1}{2}\|N^T s - \beta p + t\|^2 + p^T s \tag{24}$$

and consequently

$$\nabla\phi(s,t,\beta) = \begin{bmatrix} N(N^T s - \beta p + t) + p \\ N^T s - \beta p + t \\ -p^T(N^T s - \beta p + t) \end{bmatrix} \tag{25}$$

$$\nabla^2\phi(s,t,\beta) = \begin{bmatrix} NN^T & N & -Np \\ N^T & I & -p \\ -p^T N^T & -p^T & p^T p \end{bmatrix} = \begin{bmatrix} N \\ I \\ -p^T \end{bmatrix} \begin{bmatrix} N^T & I & -p \end{bmatrix} \tag{26}$$

It is obvious from (26) that $\nabla^2\phi(s,t,\beta)$ is positive semidefinite. We can now state an SOR algorithm for solving (23) based on QPSOR.

LPSOR1 Algorithm

Choose $(s^0,t^0,\beta^0) \in R_+^{2k+1}$, $\omega \in (0,2)$. Having (s^i,t^i,β^i) compute $(s^{i+1},t^{i+1},\beta^{i+1})$ as follows:

$$s_j^{i+1}=(s_j^i-\frac{\omega}{\|N_j\|^2}(N_j(\sum_{\ell=1}^{j-1}(N^T)_{\cdot\ell}s_\ell^{i+1}+\sum_{\ell=j}^{k}(N^T)_{\cdot\ell}s_\ell^i-\beta^i p+t^i)+p_j))_+,\ j=1,\ldots,k \quad \text{for } j>1$$

$$t^{i+1} = (t^i-\omega(N^Ts^{i+1}-\beta^i p+t^i))_+ \tag{27}$$

$$\beta^{i+1} = (\beta^i+\frac{\omega}{\|p\|^2}p^T(N^Ts^{i+1}-\beta^i p+t^{i+1}))_+$$

Parts (i) and (ii) of the following convergence theorem follow directly from Theorem 2(i) and Theorem 1(ii) above respectively.

Theorem 3: LPSOR1 Convergence

(i) Each accumulation point (s,t,β) of the sequence $\{(s^i,t^i,\beta^i)\}$ generated by the LPSOR1 algorithm solves the dual program (23) and the corresponding $\binom{x}{u}$ determined by (22) solve the dual linear programs (17)-(18).

(ii) If there exist $(s,t,\beta) \in R^{2k+1}$ satisfying $\nabla\phi(s,t,\beta) > 0$, then the sequence $\{(s^i,t^i,\beta^i)\}$ generated by the LPSOR1 algorithm is bounded and has an accumulation point.

Note that Theorem 2(ii) does not apply here because $Ny + p > 0,\ y > 0$ imply that $p^T y > 0$ and hence we cannot satisfy the Slater constraint qualification that there must exist a y satisfying $Ny + p > 0,\ y > 0$ and $p^T y < 0$. We further note that the condition $\nabla\phi(s,t,\beta) > 0$ is sufficient but not necessary for the boundedness of the sequence $\{(s^i,t^i,\beta^i)\}$. Numerical experiments have revealed no serious problems with unboundedness of the sequence $\{(s^i,t^i,\beta^i)\}$ generated by the LPSOR1 algorithm.

We conclude by giving a sparsity-preserving version of the SOR algorithm for solving a linear program that was proposed in [12]. This method is based on the fact [14] that the linear program (17) is solvable if and only if the quadratic program

$$\underset{x\in R^n}{\text{minimize}} \quad \frac{\varepsilon}{2}x^T x + c^T x \qquad \text{subject to} \quad Ax \geq b,\ x \geq 0 \tag{28}$$

is solvable for all $\varepsilon \in (0,\bar{\varepsilon})$ for some $\bar{\varepsilon} > 0$. Furthermore the unique solution of (28) is independent of ε for $\varepsilon \in (0,\bar{\varepsilon})$ and is the closest solution of the linear program (17) to the origin in the 2-norm [14]. Note that $\bar{\varepsilon}$ may be infinite in some special cases. Problem (28) can be solved then by the QPSOR algorithm of Section 3. From (8) the dual to the quadratic program (28) is

$$\underset{(x,u,v)\in R^{n+m+n}}{\text{maximize}} \quad -\frac{\varepsilon}{2}x^T x + b^T u \quad \text{subject to} \quad \varepsilon x - A^T u - v + c = 0,\ (u,v) \geq 0 \tag{29}$$

which upon elimination of x by using the constraint relation

$$x = \frac{1}{\varepsilon}(A^T u + v - c) \tag{30}$$

gives

$$\underset{(u,v)\in R^{m+n}}{\text{minimize}} \ \frac{1}{2}\|A^T u+v-c\|^2 - \varepsilon b^T u \quad \text{subject to} \quad (u,v) \geq 0 \tag{31}$$

Note that (31) is the classical exterior penalty function [7] associated with the dual linear program (18). However the perturbation results of [14] give the stronger result that ε in (31) need not approach zero in order for x defined by (30) to be a solution of (17). In other words if we let $(u(\varepsilon), v(\varepsilon))$ be a solution of (31) for $\varepsilon \in (0,\bar{\varepsilon})$ then $x = \frac{1}{\varepsilon}(A^T u(\varepsilon) + v(\varepsilon) - c)$ is independent of ε and is the closest solution of the linear program (17) to the origin in the 2-norm. Note however $(u(\varepsilon), v(\varepsilon))$ need not be a solution of the dual linear program (18) for $\varepsilon \in (0,\bar{\varepsilon})$, but each accumulation point of $\{(u(\varepsilon_i), v(\varepsilon_i))\}$ will be a solution of (18) if $\{\varepsilon_i\}$ is a decreasing sequence converging to zero. We can now solve (31) by a sparsity-preserving algorithm which follows directly from the QPSOR algorithm of Section 3 by replacing D by εI.

LPSOR2 Algorithm

Choose $(u^0,v^0) \in R_+^{m+n}$, $\omega \in (0,2)$ and $\varepsilon > 0$. Having (u^i,v^i) determine u^{i+1}, v^{i+1} as follows:

$$u_j^{i+1} = \left(u_j^i - \frac{\omega}{\|A_j\|^2}\left(A_j\left(\sum_{\substack{\ell=1 \\ \text{for } j>1}}^{j-1}(A^T)_{\cdot\ell}u_\ell^{i+1} + \sum_{\ell=j}^{m}(A^T)_{\cdot\ell}u_\ell^i + v^i - c\right) - \varepsilon b_j\right)\right)_+ \qquad j=1,\ldots,m$$

$$v^{i+1} = (v^i - \omega(A^T u^{i+1} + v^i - c))_+ \tag{32}$$

Note that this LPSOR2 algorithm, unlike the algorithms proposed in [12], will preserve any sparsity the matrix A may have and there is no need to compute AA^T as was done in [12] and thereby destroying any sparsity that A may have had.

The following convergence theorem follows directly from the convergence theorem of the QPSOR algorithm, Theorem 2 and the perturbation results of [14].

Theorem 4: LPSOR2 Convergence

(i) Let the linear program (17) have a solution. There exists a real positive number $\bar{\varepsilon}$ such that for each ε in the interval $(0,\bar{\varepsilon})$, each accumulation point (u,v) of the sequence $\{(u^i,v^i)\}$ generated by the LPSOR2 algorithm (32) solves (31) and the corresponding x determined by (30) is independent of ε and is the (unique) solution of the linear program (17) which is closest to the origin in the 2-norm.

(ii) If in addition to the assumptions of part (i) the constraints of the linear program (17) satisfy the Slater constraint qualification (15) then the sequence $\{(u^i,v^i)\}$ of the LPSOR2 algorithm (32) is bounded and has an accumulation point for each $\varepsilon \in (0,\bar{\varepsilon})$.

ACKNOWLEDGEMENTS

I am indebted to Professor K. G. Murty and the referees for constructive remarks that led to improvements of this paper.

References

1. B.-H. Ahn: "Solution of non-symmetric linear complementarity problems by iterative methods", to appear in Journal of Optimization Theory and Applications.

2. R. W. Cottle & G. B. Dantzig: "Complementary pivot theory of mathematical programming", Linear Algebra and Its Applications 1, 1968, 103-125.

3. R. W. Cottle, G. H. Golub & R. S. Sacher: "On the solution of large structured linear complementarity problems, III", Operations Research Report No. 73-8, Stanford University, Stanford, California, 1973.

4. C. W. Cryer: "The solution of a quadratic programming problem using systematic overrelaxation", SIAM Journal on Control 9, 1971, 385-392.

5. W. S. Dorn: "Duality in quadratic programming", Quarterly of Applied Mathematics 18, 1960, 155-162.

6. U. Eckhardt: "Quadratic programming by successive overrelaxation", Kernforschungsanlage Jülich, Technical Report No. Jül-1064-MA, 1974.

7. A. V. Fiacco & G. P. McCormick: "Nonlinear programming: sequential unconstrained minimization techniques", Wiley, New York, 1968.

8. S.-P. Han & O. L. Mangasarian: "A dual differentiable exact penalty function", Computer Sciences Department Technical Report No. 434, University of Wisconsin, Madison, June 1981.

9. J. L. Kreuser: "Computational experience with iterative SOR methods", Computing Activities Department Technical Report, World Bank, Washington, D.C., to appear.

10. O. L. Mangasarian: "Nonlinear programming", McGraw-Hill, New York, 1969.

11. O. L. Mangasarian: "Solution of symmetric linear complementarity problems by iterative methods", Journal of Optimization Theory and Applications 22, 1977, 465-485.

12. O. L. Mangasarian: "Iterative solution of linear programs", SIAM Journal on Numerical Analysis 18(4), August 1981.

13. O. L. Mangasarian: "Least-norm linear programming solution as an unconstrained minimization problem", Mathematics Research Center Technical Report No. 2147, University of Wisconsin, Madison, December 1980, to appear in Journal of Mathematical Analysis and Applications.

14. O. L. Mangasarian & R. R. Meyer: "Nonlinear perturbation of linear programs", SIAM Journal on Control and Optimization 17, 1979, 745-752.

15. J.-S. Pang: "The implicit complementarity problem", in O. L. Mangasarian, R. R. Meyer & S. M. Robinson (editors): "Nonlinear programming 4", Academic Press, New York, 1981, 487-518.

16. J.-S. Pang: "On the convergence of a basic iterative method for the implicit complementarity problem", to appear in Journal of Optimization Theory and Applications.

17. P. Wolfe: "A duality theorem for nonlinear programming", Quarterly of Applied Mathematics 19, 1961, 239-244.