A class of preconditioners for weighted least squares problems*

Venansius Baryamureeba[†] Trond Steihaug[‡] Yin Zhang[§]

April 30, 1999

Abstract

We consider solving a sequence of weighted linear least squares problems where the changes from one problem to the next are the weights and the right hand side (or data). This is the case for primal-dual interior-point methods. We derive a class of preconditioners based on a low rank correction to a Cholesky factorization of a weighted normal equation coefficient matrix with the previous weight.

Key Words. Weighted linear least squares, Preconditioners, Preconditioned conjugate gradient for least squares, Linear programming, Primal-dual infeasible-interior-point algorithms.

1 Introduction

In this paper, we present a class of preconditioners based on low rank corrections to the Cholesky factorization of a weighted normal equation coefficient matrix. This class of preconditioners leads to good performance for interior-point methods for linear programming. Particularly, we have implemented primal-dual Newton method to test this class of preconditioners. The numerical results on large scale problems based on this method indicate that its performance is at least as good as that of a primal-dual Newton method using only Cholesky factorization.

---

*This is Technical Report N0. 170 at the Department of Informatics, Uinversity of Bergen, N-5020 Bergen, Norway.

[†]Department of Informatics, University of Bergen, N-5020 Bergen, Norway.

[‡]Department of Informatics, University of Bergen, N-5020 Bergen, Norway.

[§]Department of Computational and Applied Mathematics, Rice University, Houston, Texas 77005, USA.

The outline of the paper is as follows: In Section 2, we show that a primal-dual interior-point algorithm generates a sequence of weighted least squares problems. In Section 3, we propose the class of preconditioners. We give theoretical results to support the choice of the index set $\mathcal{Q}$ that defines the low rank corrections. We present different ways of computing and using the preconditioner. In Section 4, we report numerical results.

Throughout this paper we use the following notation. The symbol $\min_i$ or $\max_i$ is for all i for which the argument is defined. For any matrix A, A_{ij} is the element in the i-th row and j-th column, A_j is the j-th column, $A_{j\bullet}$ is the j-th row, and $\mathrm{nnz}(A)$ is the number of nonzero elements in A. The vector $e = (1, \ldots, 1)^T$ is an n-vector of all ones. The vector norm $\|x\|^2 = x^T x$ is the Euclidean norm. The notation $x > 0$ ($x \geq 0$) means that all components of the vector x are positive (nonnegative). The symbol 0 will be used to denote the number zero, the zero vector, and the zero matrix. The symbol I is used to denote the (square) identity matrix; its size will always be apparent from the context. For any square matrix X with real eigenvalues, $\lambda_i(X)$ are the eigenvalues of X arranged in nondecreasing order. $\lambda_{\min}(X)$ and $\lambda_{\max}(X)$ denote the smallest and largest eigenvalues of X respectively; i.e.,

$$\lambda_{\min}(X) = \lambda_1(X) \leq \lambda_2(X) \leq \cdots \leq \lambda_m(X) = \lambda_{\max}(X).$$

If X is symmetric and positive definite, then the above arrangement gives

$$\lambda_i(X^{-1}) = \frac{1}{\lambda_{m-i+1}(X)}. \tag{1}$$

The letters L and R represent lower triangular factors or lower Cholesky factors, and the type of factors will always be apparent from the context.

2 The Problem

Let G be an n-by-n symmetric positive definite weight matrix. For an n-vector z, we define its G-norm by $\|z\|_G^2 = z^T G z$. The weighted least squares problem is

$$\min_y \|A^T y - h\|_G, \tag{2}$$

where $A \in \Re^{m \times n}$, $y \in \Re^m$, $h \in \Re^n$, and $m \leq n$. The matrix A will throughout this paper be assumed to have full rank m. In the applications of weighted least squares problems in this paper, the weight matrix is positive definite and diagonal. We consider a sequence of weighted least squares problems where G and h vary from system to system but where A is constant. The solution of the weighted problem (2) is given by the normal equations

$$AGA^T y = AGh, \tag{3}$$

or as component y of the solution to the augmented system

$$\begin{bmatrix} -G^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} h \\ 0 \end{pmatrix}. \tag{4}$$

In the rest of this section a primal-dual interior-point method will be discussed. Consider a primal linear programming problem in standard form:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to:} & Ax = b, \;\; x \geq 0 \end{array} \tag{5}$$

whose dual is

$$\begin{array}{ll} \text{maximize} & b^T y \\ \text{subject to:} & A^T y + z = c, \;\; z \geq 0. \end{array}$$

For a positive barrier parameter μ, the barrier problem associated with the primal problem (5) is

$$\begin{array}{ll} \text{minimize} & c^T x - \mu \sum_{j=1}^{n} \ln x^j \\ \text{subject to:} & Ax = b, \; x > 0. \end{array} \tag{6}$$

The first order optimality conditions for the barrier problem (6) are equivalent to

$$F(x, y, z) = \begin{pmatrix} Ax - b \\ A^T y + z - c \\ XZe - \mu e \end{pmatrix} = 0, \quad (x, z) \geq 0, \tag{7}$$

where $X = \text{diag}(x)$ and $Z = \text{diag}(z)$. The equation $XZe - \mu e$ is nonlinear, and depends on the barrier parameter μ. The system (7) is called the perturbed KKT conditions [7] for the linear programming problem. Now consider applying Newton's method to the system of nonlinear equations (7). For a fixed value of μ, the linear equation that defines a Newton step is

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ Z & 0 & X \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = - \begin{pmatrix} Ax - b \\ A^T y + z - c \\ XZe - \mu e \end{pmatrix}. \tag{8}$$

Let $G = XZ^{-1}$, and $\tilde{h} = c - A^T y - \mu X^{-1} e$. We can simplify (8) to

$$\begin{bmatrix} -G^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} \tilde{h} \\ b - Ax \end{pmatrix}, \tag{9}$$

and $\Delta z = X^{-1}(\mu e - Z\Delta x) - z$.

If x is strictly feasible (i.e., $x > 0$ and satisfies $Ax = b$), then a comparison of (4) and (9) shows that the augmented system (9) can be associated with a weighted least squares problem

$$\min_{\Delta y} \|A^T \Delta y - \tilde{h}\|_G^2. \tag{10}$$

However, $x > 0$ is not necessarily feasible in interior-point methods. In this case, let $\tilde{x}$ be any point such that $A\tilde{x} = b$. Then, the normal equations for (9) take on the form

$$\begin{aligned} AGA^T \Delta y &= AG\tilde{h} + b - Ax \\ &= AG\tilde{h} + A(\tilde{x} - x) \\ &= AG(\tilde{h} + G^{-1}(\tilde{x} - x)) \\ &= AGh \end{aligned} \tag{11}$$

which is equivalent to the weighted least squares problem

$$\min_{\Delta y} \|A^T \Delta y - h\|_G^2,$$

where $h = \tilde{h} + G^{-1}(\tilde{x} - x)$. It can be shown that after solving for Δy in (11), Δx and Δz can be easily found either from

$$\begin{aligned} \Delta x &= -G(c - A^T(y + \Delta y) - \mu X^{-1} e), \\ \Delta z &= X^{-1}(\mu e - Z\Delta x) - z, \end{aligned}$$

or from

$$\begin{aligned} \Delta z &= c - A^T(y + \Delta y) - z, \\ \Delta x &= Z^{-1}(\mu e - X\Delta z) - x. \end{aligned}$$

At every interior-point iteration (11) is solved. We have thus shown that a sequence of Newton step for (8) at x_k, y_k, z_k, and μ_k is a sequence of weighted least squares problems.

3 A New Class of Preconditioners

In a primal-dual interior-point algorithm, we can use a mixed strategy for solving the linear system (3). We use a direct method in every even iteration and an iterative method in every odd iteration. At an odd iteration, we construct a preconditioner based on low rank corrections to the Cholesky factor computed at the previous iteration.

3.1 The Preconditioner

Let $G, H \in \Re^{n\times n}$ be positive definite and diagonal matrices. We consider the case where we already have the Cholesky factorization of AHA^T, and we want to solve a linear system with coefficient matrix AGA^T. We employ a preconditioned conjugate gradient method with a preconditioner AKA^T, where K is an $n \times n$ positive definite and diagonal matrix constructed from H and G so that the difference between AKA^T and AHA^T is a low rank matrix. In this case, the Cholesky factorization of AHA^T can be effectively used to solve linear systems with the preconditioner AKA^T as the coefficient matrix. We now give our construction of the matrix K.

For a given index set

$$\mathcal{Q} \subseteq \{j :\ 1 \leq j \leq n \text{ and } G_{jj} \neq H_{jj}\},$$

let the $n \times n$ diagonal matrices $\bar{H}, \bar{G}$ and K be given by

$$\bar{H}_{jj} = \begin{cases} H_{jj} & \text{if } j \in \mathcal{Q} \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

$$\bar{G}_{jj} = \begin{cases} G_{jj} & \text{if } j \in \mathcal{Q} \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

and

$$K = H + \bar{G} - \bar{H}. \tag{14}$$

Note that $K_{jj} = G_{jj}$ if $j \in \mathcal{Q}$, otherwise $K_{jj} = H_{jj}$. Clearly, AKA^T is a symmetric positive definite matrix.

Let $\bar{A} \in \Re^{m\times q}$, where $q = |\mathcal{Q}|$, comprise of all columns A_j such that $j \in \mathcal{Q}$ and let $\bar{D} \in \Re^{q\times q}$ be the diagonal matrix corresponding to the nonzero diagonal elements of $\bar{G} - \bar{H}$. In this notation,

$$AKA^T = A(H + \bar{G} - \bar{H})A^T = AHA^T + \bar{A}\bar{D}\bar{A}^T,$$

namely, AKA^T is a rank q-update of AHA^T. In particular, if

$$\mathcal{Q} = \{j : \ 1 \leq j \leq n \text{ and } G_{jj} \neq H_{jj}\},$$

then $AKA^T = AGA^T$. However, as will become clear later, we will always keep q fairly small in our algorithms.

Goldfarb and Liu [10] solve a sequence of weighted least squares problems, where they consider the major changes in the diagonal elements of a symmetric positive definite matrix G between two consecutive computation steps. At computation step k they update a Cholesky factor at $k-1$ by taking into consideration all the major changes in the ratio of diagonal elements of G_{k-1} and G_k. A QR factorization of A^T is computed once, but the factors must be stored. This approach of storing the factors may not be suitable for large scale problems.

3.2 Properties of the Preconditioner

We will derive bounds for the condition number of the preconditioned matrix $(AKA^T)^{-1}AGA^T$ that will provide guidance for our selection of the index set $\mathcal{Q}$.

Lemma 3.1 *Let $G, H \in \Re^{n\times n}$ be symmetric. If H is positive definite, then*

$$\lambda_{\min}(H^{-1}G) \leq \lambda_i((AHA^T)^{-1}AGA^T) \leq \lambda_{\max}(H^{-1}G). \tag{15}$$

Moreover, (15) also holds if G is positive definite, and H and AHA^T are nonsingular.

Proof: First we note that the eigenvalues of $(AHA^T)^{-1}AGA^T$ are all real because it is similar to the symmetric matrix

$$(AHA^T)^{-1/2}(AGA^T)(AHA^T)^{-1/2}.$$

Let $\tilde{A} = AH^{1/2}$ and $\hat{G} = H^{-1/2}GH^{-1/2}$, then

$$(AHA^T)^{-1}AGA^T = (\tilde{A}\tilde{A}^T)^{-1}\tilde{A}\hat{G}\tilde{A}^T. \tag{16}$$

Let

$$\tilde{A} = U\Sigma V^T = U\Sigma_1 V_1^T \tag{17}$$

be the singular value decomposition of $\tilde{A}$, where $U \in \Re^{m\times m}$ and $V \in \Re^{n\times n}$ are orthogonal matrices and $\Sigma \in \Re^{m\times n}$ is a positive definite diagonal matrix.

Let $\Sigma_1 \in \Re^{m\times m}$ and $V_1 \in \Re^{n\times m}$ comprise of the first m columns of Σ and V respectively. Substituting (17) into the right hand side of (16), and noting that $V_1^T V_1 = I$ and $U^T U = I$ gives

$$(AHA^T)^{-1}AGA^T = (U\Sigma_1^{-1})V_1^T \hat{G} V_1 (\Sigma_1 U^T).$$

Therefore by similarity, $\lambda_i((AHA^T)^{-1}AGA^T) = \lambda_i(V_1^T \hat{G} V_1)$. Thus, there exists a unit eigenvector $y_i \in \Re^m$ of $V_1^T \hat{G} V_1$ such that

$$\lambda_i((AHA^T)^{-1}AGA^T) = \lambda_i(V_1^T \hat{G} V_1) = y_i^T V_1^T \hat{G} V_1 y_i \geq \lambda_{\min}(\hat{G}),$$

where the last inequality follows from the fact that $V_1 y_i$ is a unit vector in $\Re^n$. Similarly, we can prove that $\lambda_i((AHA^T)^{-1}AGA^T) \leq \lambda_{\max}(\hat{G})$. Finally we note that $\hat{G}$ is similar to $H^{-1}G$.

Considering the matrix $(AGA^T)^{1/2}(AHA^T)^{-1}(AGA^T)^{1/2}$, the proof for the second assertion follows from a similar argument. ∎

Theorem 3.1 *Let $G, H \in \Re^{n\times n}$ be positive definite and diagonal. For a given $\mathcal{Q}$ let K be defined as in (14). The eigenvalues of $(AKA^T)^{-1}AGA^T$ satisfy*

$$\min\left\{1, \min_{j\notin\mathcal{Q}} \frac{G_{jj}}{H_{jj}}\right\} \leq \lambda_i((AKA^T)^{-1}AGA^T) \leq \max\left\{1, \max_{j\notin\mathcal{Q}} \frac{G_{jj}}{H_{jj}}\right\}.$$

Proof: Since K defined by equation (14) is positive definite and diagonal, Lemma 3.1 implies that

$$\min_j \{G_{jj}/K_{jj}\} \leq \lambda_i((AKA^T)^{-1}AGA^T) \leq \max_j \{G_{jj}/K_{jj}\}. \tag{18}$$

For $j \in \mathcal{Q}$ then $G_{jj}/K_{jj} = 1$, and if $j \notin \mathcal{Q}$ then $G_{jj}/K_{jj} = G_{jj}/H_{jj}$. We thus have

$$\max_j \left\{\frac{G_{jj}}{K_{jj}}\right\} = \max\left\{1, \max_{j\notin\mathcal{Q}} \{G_{jj}/H_{jj}\}\right\}$$

and

$$\min_j \left\{\frac{G_{jj}}{K_{jj}}\right\} = \min\left\{1, \min_{j\notin\mathcal{Q}} \{G_{jj}/H_{jj}\}\right\}. \blacksquare$$

The following observations are now in order. If $\lambda_{\max}(H^{-1}G) > 1$, the index set $\mathcal{Q}$ should include indices corresponding to the large G_{jj}/H_{jj} in order to reduce on the upper bound. Similarly, to increase the lower bound when $\lambda_{\min}(H^{-1}G) < 1$, indices corresponding to the small G_{jj}/H_{jj} should be included in $\mathcal{Q}$. The following corollary gives bounds on the eigenvalues of the preconditioned matrix $(AKA^T)^{-1}AGA^T)$ if $\mathcal{Q}$ is chosen by such a strategy.

Corollary 3.1 *Let $G, H \in \Re^{n\times n}$ be positive definite and diagonal. Let γ_j be the sorted elements of G_{jj}/H_{jj} in nondecreasing order:*

$$\gamma_1 = \min_j \{G_{jj}/H_{jj}\} \le \gamma_2 \le \cdots \le \gamma_n = \max_j \{G_{jj}/H_{jj}\}.$$

Let $q = q_1 + q_2$ and $\mathcal{Q}$ comprise of the indices corresponding to the q_1 largest and q_2 smallest diagonal elements of $H^{-1}G$. Then

$$\min\{\gamma_{q_2+1}, 1\} \le \lambda_i((AKA^T)^{-1}AGA^T) \le \max\{\gamma_{n-q_1}, 1\}.$$

Proof: Observe that

$$\max_{j\notin\mathcal{Q}} \{G_{jj}/K_{jj}\} = \max_{j\notin\mathcal{Q}} \{G_{jj}/H_{jj}\} = \gamma_{n-q_1}$$

and

$$\min_{j\notin\mathcal{Q}} \{G_{jj}/K_{jj}\} = \min_{j\notin\mathcal{Q}} \{G_{jj}/H_{jj}\} = \gamma_{q_2+1},$$

and the result follows from Theorem 3.1.■

The following lemma will be used in the proof of Theorem 3.2.

Lemma 3.2 *Let $G, H, K \in \Re^{n\times n}$ be symmetric positive definite matrices, and $E = (AKA^T)^{-1} - (AHA^T)^{-1}$. The eigenvalues of $EAGA^T$ are bounded by*

$$\lambda_{\min}((K^{-1} - H^{-1})G) \le \lambda_i(EAGA^T) \le \lambda_{\max}((K^{-1} - H^{-1})G).$$

Proof: Let $\tilde{A} = AG^{1/2}, \hat{H} = G^{-1/2}HG^{-1/2}$, and $\hat{K} = G^{-1/2}KG^{-1/2}$. Then $AGA^T = \tilde{A}\tilde{A}^T, AHA^T = \tilde{A}\hat{H}\tilde{A}^T$, and $AKA^T = \tilde{A}\hat{K}\tilde{A}^T$. Let $\tilde{A} = U\Sigma_1 V_1^T$ be the thin singular value decomposition, where $U \in \Re^{m\times m}$ and $V_1 \in \Re^{n\times m}$ are orthogonal matrices, and $\Sigma_1 \in \Re^{m\times m}$ is a positive definite diagonal matrix. Noting that $V_1^TV_1 = I$ and $U^TU = I$ gives

$$\begin{aligned}(AKA^T)^{-1}AGA^T &= U\Sigma_1^{-1}V_1^T\hat{K}^{-1}V_1\Sigma_1U^T \\ &= (\Sigma_1U^T)^{-1}V_1^T\hat{K}^{-1}V_1(\Sigma_1U^T)\end{aligned}$$

and

$$\begin{aligned}(AHA^T)^{-1}AGA^T &= U\Sigma_1^{-1}V_1^T\hat{H}^{-1}V_1\Sigma_1U^T \\ &= (\Sigma_1U^T)^{-1}V_1^T\hat{H}^{-1}V_1(\Sigma_1U^T).\end{aligned}$$

Thus

$$\begin{aligned}EAGA^T &= ((AKA^T)^{-1} - (AHA^T)^{-1})AGA^T \\ &= (\Sigma_1U^T)^{-1}V_1^T[\hat{K}^{-1} - \hat{H}^{-1}]V_1(\Sigma_1U^T).\end{aligned}$$

By similarity, $\lambda_i(EAGA^T) = \lambda_i(V_1^T[\hat{K}^{-1} - \hat{H}^{-1}]V_1)$. Thus, there exists a unit eigenvector $y_i \in \Re^m$ of $V_1^T[\hat{K}^{-1} - \hat{H}^{-1}]V_1$ such that

$$\begin{aligned}\lambda_i(EAGA^T) &= \lambda_i(V_1^T[\hat{K}^{-1} - \hat{H}^{-1}]V_1) \\ &= y_i^T V_1^T[\hat{K}^{-1} - \hat{H}^{-1}]V_1 y_i \\ &\leq \lambda_{\max}(\hat{K}^{-1} - \hat{H}^{-1}),\end{aligned}$$

where the last inequality follows from the fact that $V_1 y_i$ is a unit vector in $\Re^n$. Similarly, we can prove that

$$\lambda_i(EAGA^T) \geq \lambda_{\min}(\hat{K}^{-1} - \hat{H}^{-1}).$$

Finally, we note that $\hat{K}^{-1} - \hat{H}^{-1} = G^{1/2}(K^{-1} - H^{-1})G^{1/2}$ is similar to $(K^{-1} - H^{-1})G$. ■

Theorem 3.2 below gives a bound on the eigenvalues of $(AKA^T)^{-1}AGA^T$ that is independent of the choice of $\mathcal{Q}$. The result states that whatever way $\mathcal{Q}$ is chosen, the largest eigenvalue of the preconditioned matrix is not greater than the maximum eigenvalue of $(AHA^T)^{-1}AGA^T$ plus 1.

Theorem 3.2 *Let $G, H \in \Re^{n\times n}$ be positive definite and diagonal, K be given in (14), and $M = (AHA^T)^{-1}AGA^T$. The eigenvalues of the matrix $(AKA^T)^{-1}AGA^T$ are bounded by*

$$\frac{\lambda_{\min}(M)}{\lambda_{\min}(M)+1} \leq \lambda_i((AKA^T)^{-1}AGA^T) \leq \lambda_{\max}(M) + 1.$$

Proof: Let $E = (AKA^T)^{-1} - (AHA^T)^{-1}$. Then the system

$$(AKA^T)^{-1}AGA^T = (AHA^T)^{-1}AGA^T + EAGA^T,$$

is similar to the symmetric system

$$\begin{aligned}& (AGA^T)^{1/2}(AKA^T)^{-1}(AGA^T)^{1/2} \\ = \;& (AGA^T)^{1/2}(AHA^T)^{-1}(AGA^T)^{1/2} \\ + \;& (AGA^T)^{1/2}E(AGA^T)^{1/2}.\end{aligned}$$

Considering the above symmetric system, it follows from Wilkinson ([15], pp.101-102) that

$$\begin{aligned}& \lambda_i((AGA^T)^{1/2}(AKA^T)^{-1}(AGA^T)^{1/2}) \\ \leq \;& \lambda_i((AGA^T)^{1/2}(AHA^T)^{-1}(AGA^T)^{1/2}) \\ + \;& \lambda_{\max}((AGA^T)^{1/2}E(AGA^T)^{1/2}),\end{aligned}$$

and therefore by similarity

$$\lambda_i((AKA^T)^{-1}AGA^T) \leq \lambda_i((AHA^T)^{-1}AGA^T) + \lambda_{\max}(EAGA^T).$$

Without loss of generality, we may assume that, after a reordering if necessary, $\mathcal{Q} = \{1, 2, \cdots, q\}$. Define

$$G = \begin{bmatrix} G_1 & 0 \\ 0 & G_2 \end{bmatrix}, \quad H = \begin{bmatrix} H_1 & 0 \\ 0 & H_2 \end{bmatrix}, \quad K = \begin{bmatrix} G_1 & 0 \\ 0 & H_2 \end{bmatrix};$$

where $G_1, H_1 \in \Re^{q \times q}$; and $G_2, H_2 \in \Re^{(n-q)\times(n-q)}$ are all diagonal submatrices. Then

$$K^{-1}G = \begin{bmatrix} I & 0 \\ 0 & H_2^{-1}G_2 \end{bmatrix}, \quad H^{-1}G = \begin{bmatrix} H_1^{-1}G_1 & 0 \\ 0 & H_2^{-1}G_2 \end{bmatrix},$$

and

$$(K^{-1} - H^{-1})G = (K^{-1}G - H^{-1}G) = \begin{bmatrix} I - H_1^{-1}G_1 & 0 \\ 0 & 0 \end{bmatrix}. \tag{19}$$

Lemma 3.2 implies that

$$\begin{aligned} \lambda_{\max}(EAGA^T) &\leq \lambda_{\max}((K^{-1} - H^{-1})G) \\ &\leq \max\left\{1 - \min_{j \in \mathcal{Q}} \{G_{jj}/H_{jj}\}, 0\right\} \\ &< 1 \end{aligned}$$

and

$$\lambda_i((AKA^T)^{-1}AGA^T) \leq \lambda_i((AHA^T)^{-1}AGA^T) + 1. \tag{20}$$

To derive the lower-bound on $\lambda_i((AKA^T)^{-1}AGA^T)$, consider

$$AKA^T = AHA^T + A(\bar{G} - \bar{H})A^T$$

and the relationship (1). Then

$$\begin{aligned} & \frac{1}{\lambda_i((AKA^T)^{-1}AGA^T)} \\ = \; & \lambda_{m-i+1}((AGA^T)^{-1}AKA^T) \\ \leq \; & \lambda_{m-i+1}((AGA^T)^{-1}AHA^T) + \lambda_{\max}((AGA^T)^{-1}A(\bar{G} - \bar{H})A^T) \\ \leq \; & \frac{1}{\lambda_i((AHA^T)^{-1}AGA^T)} + \lambda_{\max}(G^{-1}(\bar{G} - \bar{H})) \\ \leq \; & \frac{1}{\lambda_i((AHA^T)^{-1}AGA^T)} + 1, \end{aligned}$$

where we used the fact that

$$\lambda_{\max}(G^{-1}(\bar{G}-\bar{H})) \leq \max\left\{\max_{j\in\mathcal{Q}}\left\{1-\frac{H_{jj}}{G_{jj}}\right\},0\right\} < 1.$$

Thus,

$$\lambda_i((AKA^T)^{-1}AGA^T) \geq \frac{\lambda_i((AHA^T)^{-1}AGA^T)}{1+\lambda_i((AHA^T)^{-1}AGA^T)}. \tag{21}$$

Considering the upper bound for $\lambda_{\max}((AKA^T)^{-1}AGA^T)$ in (20) and the lower bound for $\lambda_{\min}((AKA^T)^{-1}AGA^T)$ in (21) ends the proof. ■

Lemma 3.3 gives bounds on the eigenvalues of a matrix after a series of rank-one corrections. This result will be used in the proof of Theorem 3.3 to give bounds on the eigenvalues of the preconditioned matrix $(AKA^T)^{-1}AGA^T$.

Lemma 3.3 *Suppose $B_q = B_0 + \sum_{j=1}^{q}\tau_j c_j c_j^T$, where $c_j \in \Re^m$, $\tau_j \neq 0$, $q < m$, and $B_0 \in \Re^{m\times m}$ is symmetric. Let $q = q_1 + q_2$, where q_1 is the number of indices such that $\tau_j > 0$. Then*

$$\lambda_{\min}(B_q) \leq \lambda_{q_1+1}(B_0) \quad \textit{and} \quad \lambda_{\max}(B_q) \geq \lambda_{m-q_2}(B_0).$$

Proof: Observe that

$$B_j = B_{j-1} + \tau_j c_j c_j^T,\ 1 \leq j \leq q.$$

The proof follows from Wilkinson ([15], pp. 94-97). ■

Theorem 3.3 gives a lower bound on $\lambda_{\max}((AKA^T)^{-1}AGA^T)$ and an upper bound on $\lambda_{\min}((AKA^T)^{-1}AGA^T)$.

Theorem 3.3 *Let $G, H \in \Re^{n\times n}$ be positive definite and diagonal matrices. Let $|\mathcal{Q}| = q = q_1 + q_2 < m$, where q_1 is the number of indices in $\mathcal{Q}$ such that $G_{jj} > H_{jj}$. Assume that $\mathcal{Q}$ contains no index such that $G_{jj} = H_{jj}$. Then*

$$\lambda_{\min}((AKA^T)^{-1}AGA^T) \leq \lambda_{q_2+1}((AHA^T)^{-1}AGA^T)$$

and

$$\lambda_{\max}((AKA^T)^{-1}AGA^T) \geq \lambda_{m-q_1}((AHA^T)^{-1}AGA^T).$$

Proof: Consider again

$$AKA^T = AHA^T + \bar{A}(\bar{G}-\bar{H})\bar{A}^T = AHA^T + \bar{A}\bar{D}\bar{A}^T.$$

Let $\bar{D}_1$ and $-\bar{D}_2$ be diagonal matrices with the positive and negative diagonal elements of $\bar{D}$ on their diagonals, respectively. Clearly, $\bar{D}_1$ and $\bar{D}_2$ are positive definite matrices of sizes $q_1 \times q_1$ and $q_2 \times q_2$, respectively. Corresponding to the indices of $\bar{D}_1$ and $\bar{D}_2$, we similarly define $\bar{A}_1$ and $\bar{A}_2$. Then

$$AKA^T = AHA^T + \bar{A}_1\bar{D}_1\bar{A}_1^T - \bar{A}_2\bar{D}_2\bar{A}_2^T. \tag{22}$$

Let

$$B_0 = (AGA^T)^{-1/2}AHA^T(AGA^T)^{-1/2}$$

and

$$B_q = (AGA^T)^{-1/2}AKA^T(AGA^T)^{-1/2}.$$

We define

$$c_j = \begin{cases} \left[(AGA^T)^{-1/2}\bar{A}_1\bar{D}_1^{1/2}\right]_j, & j = 1, 2, \cdots, q_1, \\ \left[(AGA^T)^{-1/2}\bar{A}_2\bar{D}_2^{1/2}\right]_{j-q_1}, & j = q_1 + 1, \cdots, q_1 + q_2, \end{cases}$$

and

$$\tau_j = \begin{cases} 1, & j = 1, 2, \cdots, q_1, \\ -1, & j = q_1 + 1, \cdots, q_1 + q_2. \end{cases}$$

It follows from (22) that

$$B_q = B_0 + \sum_{j=1}^{q} \tau_j c_j c_j^T.$$

From Lemma 3.3 and using (1)

$$\begin{aligned} \lambda_{\max}(B_q^{-1}) &= \frac{1}{\lambda_{\min}(B_q)} \\ &\geq \frac{1}{\lambda_{q_1+1}(B_0)} = \lambda_{m-q_1}(B_0^{-1}) \end{aligned}$$

Similarly

$$\begin{aligned} \lambda_{\min}(B_q^{-1}) &= \frac{1}{\lambda_{\max}(B_q)} \\ &\leq \frac{1}{\lambda_{m-q_2}(B_0)} = \lambda_{q_2+1}(B_0^{-1}) \end{aligned}$$

The theorem now follows from noting the similarity of B_q^{-1} to $(AKA^T)^{-1}AGA^T$ and B_0^{-1} to $(AHA^T)^{-1}AGA^T$. ■

The results in Corollary 3.1 suggest choosing $\mathcal{Q}$ to contain indices corresponding to the largest and smallest diagonal elements of $H^{-1}G$. Let $q = |\mathcal{Q}|$ be the number of elements in the index set. If $\lambda_{\min}(H^{-1}G) < 1$ and $\lambda_{\max}(H^{-1}G) > 1$ then choose $\mathcal{Q}$ to comprise of indices corresponding to q_1 largest and q_2 smallest diagonal elements of $H^{-1}G$ where $q = q_1 + q_2$. For $\lambda_{\min}(H^{-1}G) > 1$ the index set $\mathcal{Q}$ consists of the indices corresponding to q largest diagonal elements of $H^{-1}G$ and in the case $\lambda_{\max}(H^{-1}G) < 1$ the index set consist of the indices corresponding to q smallest diagonal elements.

By combining Theorem 3.3 and Corollary 3.1 we have that if $\gamma_{n-q_1} > 1$, $\gamma_{q_2+1} < 1$ and $q = q_1 + q_2 < m$, the condition number of the preconditioned matrix $(AKA^T)^{-1}AGA^T$ satisfies

$$\frac{\lambda_{m-q_1}((AHA^T)^{-1}AGA^T)}{\lambda_{q_2+1}((AHA^T)^{-1}AGA^T)} \leq \kappa((AKA^T)^{-1}AGA^T) \leq \frac{\gamma_{n-q_1}}{\gamma_{q_2+1}}. \tag{23}$$

3.3 Computing the Preconditioner

We shall consider two approaches of computing $(AKA^T)^{-1}d$: using the Sherman Morrison-Woodbury formula or updating the Cholesky factor (or triangular factors).

3.3.1 The Sherman Morrison-Woodbury Formula

To simplify the notation in the rest of this section, let $M = (AKA^T)^{-1}$. For $AKA^T = (AHA^T + \bar{A}\bar{D}\bar{A}^T)$ the Sherman Morrison-Woodbury formula (see [11] pp. 50, for example) gives

$$M = (AHA^T)^{-1}[I - \bar{A}(\bar{D}^{-1} + \bar{A}^T(AHA^T)^{-1}\bar{A})^{-1}\bar{A}^T(AHA^T)^{-1}].$$

Assume that we have a Cholesky factorization of AHA^T. Let $LL^T = AHA^T$ and compute $V = L^{-1}\bar{A}$. Thus

$$\begin{aligned} M &= (LL^T)^{-1} - L^{-T}V(\bar{D}^{-1} + V^TV)^{-1}V^TL^{-1} \\ &= L^{-T}(I - VF^{-1}V^T)L^{-1}, \end{aligned} \tag{24}$$

where $F = \bar{D}^{-1} + V^TV$. For an m-vector d we have

$$Md = L^{-T}[r - (V(F^{-1}(V^Tr)))],$$

where $r = L^{-1}d$. In computing Md we need to store L, V, and the factors of F. Note that F is symmetric but may be indefinite. The cost of computing F is approximately $2q + 2q$ nnz(V) floating point operations.

By a solve we mean either a backward or forward substitution. If the right-hand side is sparse, then we can reduce the number of floating point operations.

The cost in computing and using M is dominated by the number of solves carried out. The initial cost of M comprises of q solves, and the cost of computing and factorizing the q-by-q matrix F. The computation cost of Md consists of 2 solves plus approximately $4\text{nnz}(V) + 2q^2 + m$ floating point operations where $\text{nnz}(V)$ is the number of nonzero elements in V.

Numerical testing on large Netlib test problems shows that V is usually a sparse matrix. One approach to compute $V = L^{-1}\bar{A}$ is to carry out q solves with q columns of $\bar{A}$. This will be solves with very sparse right hand sides.

An alternative way even exploits more of the sparsity structure of $\bar{A}$. Assume that the number of nonzero rows in $\bar{A}$ is p and p is small relative to q, i.e., $p \ll q$. Let e_i be the i-th unit vector in $\Re^m$ corresponding to the index i of a nonzero row of $\bar{A}$. Let E be the $m \times p$ matrix comprising of the columns e_i for all i corresponding to the nonzero rows of $\bar{A}$. Then $E^T\bar{A}$ is a $p \times q$ sub-matrix of $\bar{A}$ obtained from deleting all zero rows of $\bar{A}$. Then $L^{-1}\bar{A} = (L^{-1}E)(E^T\bar{A})$. Thus we only need to carry out p solves with E and then multiply the result by $E^T\bar{A}$. Again note that the sparsity in the right hand sides in E should be exploited.

3.3.2 Updating the triangular factors

The sparsity structure of AKA^T is the same as for AHA^T. When we update the triangular factors of AHA^T the sparsity structure of these factors is preserved. In this case it may be more efficient to update the triangular factors than using the technique in the previous subsection. Thus, we want to compute R and T so that

$$M \equiv (RTR^T)^{-1} = (LDL^T + \bar{A}\bar{D}\bar{A}^T)^{-1}, \tag{25}$$

where R and L are lower triangular and T and D are positive diagonal, and L and D are given.

Bennett [3] presents an algorithm for updating triangular factors of a general matrix of the form $B = A + XCY^T$, when the factors of A are given. For A and C symmetric and $X = Y$, the algorithm simplifies and the number of operations is halved. This algorithm is stable [9] when applied to AKA^T and AHA^T since both are symmetric and positive definite. We rewrite the algorithm below.

Algorithm 3.1 *Triangular factor update for dense matrices*

Set $V = \bar{A}$ *and* $C = \bar{D}$.
for $i = 1, \ldots, m$ **do**
  $p = (V_{i\bullet} C)^T$
  $T_{ii} = D_{ii} + V_{i\bullet}\, p$
  $u = (1/T_{ii})\, p$
  $C \leftarrow C - u\, p^T$
  for $j = i+1, \ldots, m$ **do**
    $V_{j\bullet} \leftarrow V_{j\bullet} - L_{ji} V_{i\bullet}$
    $R_{ji} = L_{ji} + V_{j\bullet}\, u$
  end for
end for

In computing R and T, L and D can be used to store the elements of R and T. In the case of rank-one correction ($q = 1$), Algorithm 3.1 is identical to a special case of the algorithm by Gill et.al [9]. Algorithm 3.1 is an extention from rank-one to rank-q correction for $q \geq 1$.

Algorithm 3.1 is generalized to the sparse case in Baryamureeba and Steihaug [2]. The sparse algorithm is developed for the special class of updates where $\bar{A}$ is a sub-matrix of A. This algorithm is given below:

Algorithm 3.2 *Triangular factor update for sparse matrices*

Set $V = \bar{A}$, $C = \bar{D}$ *and* $R = 0$.
for $i = 1, \ldots, m$ **do**
  if $V_{i\bullet} \neq 0$ **then**
    $p = (V_{i\bullet} C)^T$
    $T_{ii} = D_{ii} + V_{i\bullet}\, p$
    $u = (1/T_{ii})\, p$
    $C \leftarrow C - u\, p^T$
    for $j = i+1, \ldots, m$ **do**
      if $L_{ji} \neq 0$ **then**
        $V_{j\bullet} \leftarrow V_{j\bullet} - L_{ji} V_{i\bullet}$
        $R_{ji} = L_{ji} + V_{j\bullet}\, u$
      end if
    end for
  end if
end for

The final V in Algorithm 3.2 can be shown to satisfy $V = L^{-1}\bar{A}$.
Algorithm 3.2 is very effective when there are either many zero rows in $\bar{A}$ or

few nonzero elements in the unit lower triangular factor L. It is important to note that in Algorithm 3.2, the sparsity structure of the triangular factors is preserved for the special case where $\bar{A}$ consists of columns from A and $\bar{D}$ is a diagonal matrix. Since the matrices K and H are diagonal, AKA^T and AHA^T have the same sparsity structure. Thus if R and L are Cholesky factors of AKA^T and AHA^T respectively, then they have the same sparsity structure. This is achieved in Algorithm 3.2 by setting $R_{ji} = 0$ whenever $L_{ji} = 0$.

4 Numerical Results

In this section, we report some numerical results. We have implemented a primal-dual Newton algorithm (PDN), where Cholesky factorization is used at all iterations, and some variants of the mixed primal-dual Newton algorithm, where Cholesky factorization and preconditioned conjugate gradient method are used alternatively at the even and odd iterations. All the numerical experiments are performed on a Sun Ultra 10 Workstation.

We choose a starting point (x_0, y_0, z_0) that is identical to the one given by Mehrotra [13]. We terminate the interior-point iterations when the relative error ϵ, as defined in (26) below, is less than or equal to the prescribed tolerance $\epsilon^* = 10^{-5}$, or when we exceed the maximum number of interior-point iterations which is set to 300 in our experiments. The relative error is defined as

$$\epsilon = \max\left\{ \frac{\|Ax-b\|}{\max\{1, \|b\|\}}, \frac{\|A^Ty+z-c\|}{\max\{1, \|c\|\}}, \frac{|c^Tx-b^Ty|}{\max\{1, |c^Tx|\}} \right\}. \tag{26}$$

At each iteration, to compute the step $(\Delta x, \Delta y, \Delta z)$ through (8) we set the barrier parameter $\mu = \sigma(x^Tz)/n$ for $\sigma = 0.1$.

The step length is chosen as a parameter $\tau \in (0,1)$ times the step length to the boundary of the nonnegative orthant, where we set the parameter $\tau = 0.99995$.

Our implementation is done entirely in Matlab. We use the Matlab function `symmmd` to compute the symmetric multiple minimum degree ordering for matrices of the form AGA^T where G is diagonal. The direct solver used is the Matlab sparse Cholesky solver `chol`. The iterative solver is a Matlab implementation of a preconditioned conjugate gradient and least squares (PCGLS) method [4]. The termination criteria in PCGLS routine are when either the number of PCGLS iterations has reached a prescribed integer t or when $\|v\|_2 \le 10^{-5}$ where $v = AGA^T\Delta y - AGh$.

The test problems used in our experiments are all from the Netlib set of linear programs [8].

In Figures 1 and 2 below, we give some examples to demonstrate the effectiveness of the preconditioner for the weighted least squares problems. The results are generated from one of the Netlib test problem **blend** at a particular interior-point iteration.

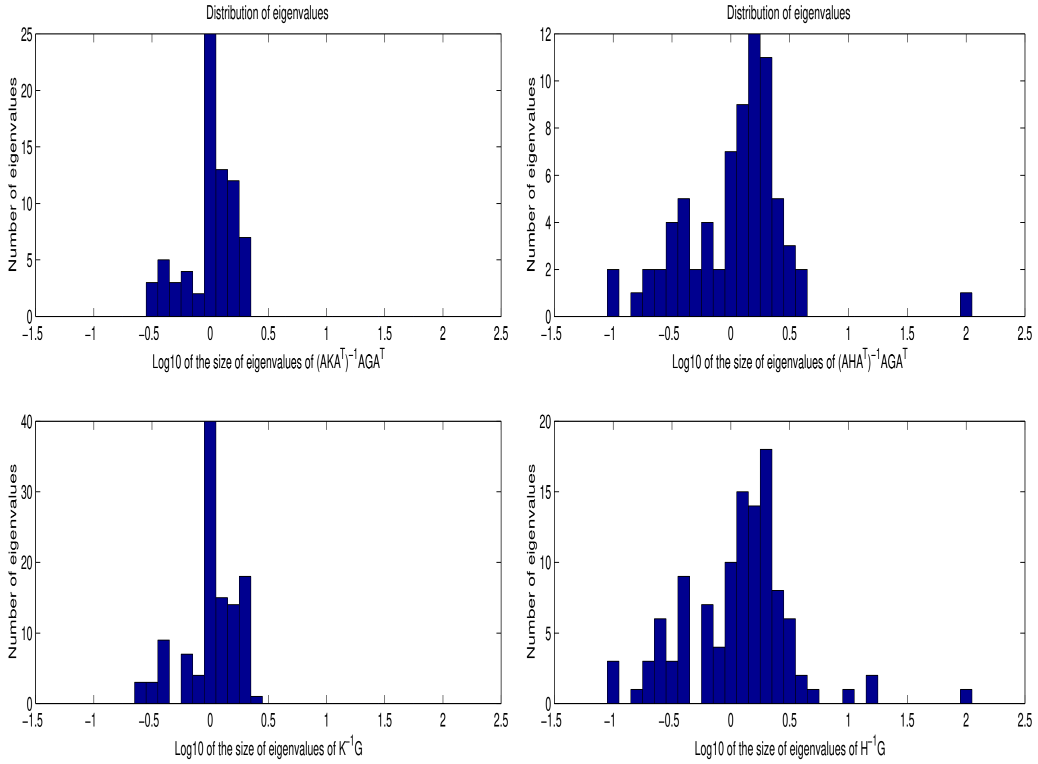

Figure 1: The matrices are extracted at an interior-point iteration in the middle stage for test problem `blend`. We set $q = 30$ and choose $\mathcal{Q}$ to comprise of indices corresponding to the 20 largest and 10 smallest diagonal elements of $H^{-1}G$.

The results in Figures 1 and 2 verify Lemma 3.1 and Corollary 3.1. We observe that the matrix $(AKA^T)^{-1}AGA^T$ has more eigenvalues near 1 than $(AHA^T)^{-1}AGA^T$. The spectrum of $(AKA^T)^{-1}AGA^T$ is contained in that of $K^{-1}G$ and is considerably narrower than that of $(AHA^T)^{-1}AGA^T$ which is contained in that of $H^{-1}G$. Clearly, $(AKA^T)^{-1}AGA^T$ has a smaller condition number than $(AHA^T)^{-1}AGA^T$. The bound on this condition

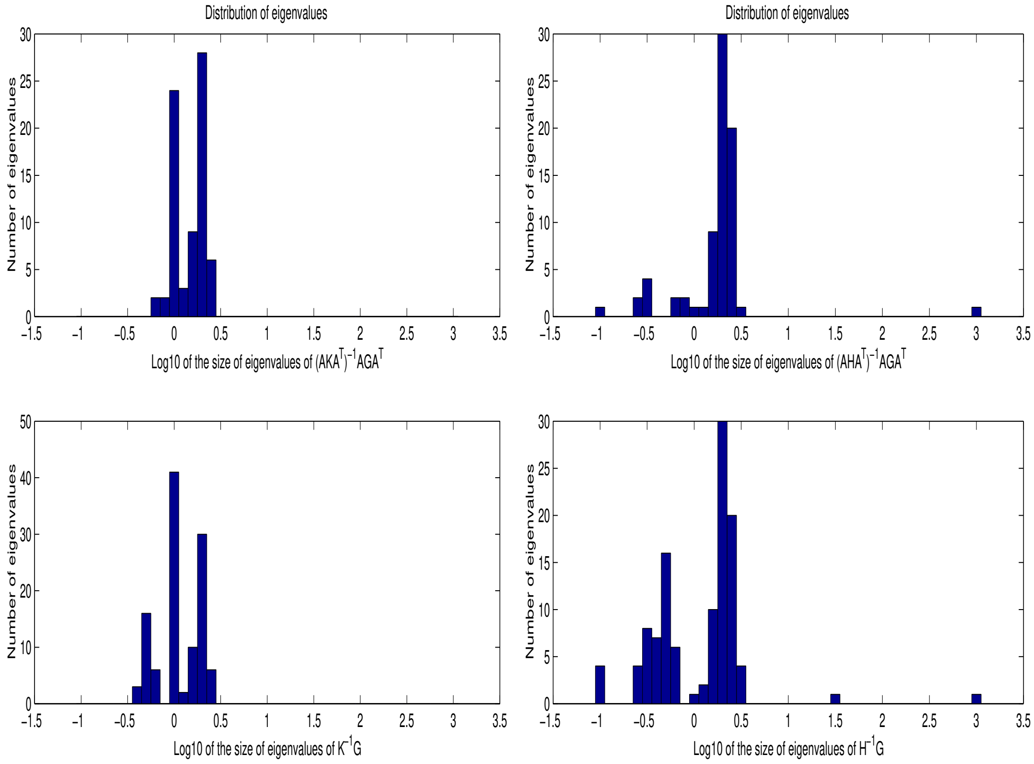

Figure 2: The matrices are extracted at an interior-point iteration in the last stage for test problem `blend`. We set $q = 40$ and choose $\mathcal{Q}$ to comprise of indices corresponding to the 20 largest and 20 smallest diagonal elements of $H^{-1}G$.

number depends on the choice of the index set $\mathcal{Q}$ as seen in (23).

4.1 Guidelines for Choosing q

Our results have suggested that $\mathcal{Q}$ should comprise of the indices corresponding to the largest and smallest diagonal elements of $H^{-1}G$. However, it is not clear yet as to how large the set $\mathcal{Q}$ should be. The choice of the number $q = |\mathcal{Q}|$ will determine the practical efficiency of our approach. It should be evident that the optimal q is problem-dependent and guidelines are needed for the selection of this number. In this subsection, we present some numerical experiments designed for the purpose of providing some guidelines for selecting q.

We first need to introduce some terminology. Given a sparse matrix $A \in \Re^{m\times n}$, let L be the (sparse) Cholesky factor of any matrix of the form AHA^T for some positive diagonal matrix H. By a *dense solve*, we mean the computation of $L^{-1}h$ for a dense vector h, and by a *sparse solve*, we mean the computation of $L^{-1}A_j$ for some column of A. For the given A, the average cost of a sparse solve relative to a dense solve is given by the ratio

$$R_{S/D} = \frac{\text{cost of computing } L^{-1}A}{n \times \text{ cost of a dense solve}}. \tag{27}$$

For given A and G the cost of a sparse computation of AGA^T and a sparse factorization of AGA^T relative to a dense solve are given by the ratios

$$R_{N/D} = \frac{\text{cost of computing } AGA^T}{\text{cost of a dense solve}} \tag{28}$$

and

$$R_{C/D} = \frac{\text{cost of Cholesky factorization of } AGA^T}{\text{cost of a dense solve}} \tag{29}$$

respectively.

Problem	Ratio $R_{S/D}$	Ratio $R_{N/D}$	Ratio $R_{C/D}$	Cost of a dense solve
czprob	0.0409	4.9473	5.8444	15471
d2q06c	0.6170	1.9357	158.3685	479827
d6cube	0.6838	5.2178	128.8655	107682
stocfor2	0.4316	0.5761	80.0852	222527
scsd8	0.1771	4.7726	10.3266	15973
agg3	0.2551	4.0484	30.0756	50612
maros	0.3005	3.6805	29.8506	57885

Table 1: The average cost of a sparse solve relative to a dense solve, the cost of computing AGA^T and the Cholesky factorization relative to a dense solve. The last column indicates the cost of a dense solve in floating point operations.

The results in Table 1 show that a sparse solve costs less than a dense solve does, as is expected. In particular, for test problem `czprob` a dense solve is much more expensive than a sparse solve ($R_{S/D} \ll 1$). This suggests that for problems where a sparse solve is much cheaper to compute relative to a dense solve we should use a larger q and do fewer PCGLS iterations. For most sparse problems the cost of computing AGA^T is comparable to computing a few dense solves. However, as the table indicates, computing the Cholesky factorization is generally much more expensive. Since at every PCGLS iteration we compute 2 dense solves, the results in Table 1 show that we can carry out many PCGLS iterations before we exceed the cost of computing the Cholesky factorization.

One approach to obtain an estimate on the cost ratio of a sparse solve verses a dense solve is to compare the cost of 2 sparse solves with the cost of 1 PCGLS iteration at the beginning of the interior-point algorithm and use this information to decide whether to choose a large q and do few PCGLS iterations or choose a small q and do more PCGLS iterations in the rest of the interior-point iterations.

We will denote the M matrix defined in (24) by M_1, and the M matrix defined in (25) by M_2. Obviously, $M_1 = M_2$; but we use the two distinct symbols because the amounts of computation associated with the two formulas are different. By the cost of computing M_1, we mean the cost of computing the matrices V and F in (24), and the factors of F. On the other hand, by the cost of computing M_2, we mean the cost of computing the matrices R and T in (25).

We define the break-even point for computing M (BEPCM) as the largest integer q for which the cost of computing M is less than or equal to the cost of computing AGA^T and its Cholesky factor, where the cost is measured by floating point operations.

The cost of a PCGLS step consists of the cost of computing M for a given q, plus cost of PCGLS iterations. The cost of a direct step consists of the cost of computing AGA^T, the cost of computing its Cholesky factor and 2 dense solves. When number of iterations per PCGLS step is fixed to one, we define the break-even point for using M (BEPUM) to be the largest integer q for which the cost of a PCGLS step is less than or equal to the cost of a direct step, where the cost is measured by floating point operations unless otherwise indicated.

Table 2 gives the break-even points for both M_1 and M_2 for several Netlib linear programming problems. The figures in the column for BEPCM indicate that in our current implementation M_1 is generally less expensive to compute than M_2 is. The figures in the column for BEPUM give upper bounds for choosing q. For a given problem when the ratio $R_{S/D}$ is very small in Table 1, the difference in the corresponding figures for BEPCM and BEPUM in Table 2 is large, and vice-versa.

For a constant q, we define the break-even point for using M (BEPUM) as the largest number of PCGLS iterations for which the cost of a PCGLS step is less than or equal to the cost of a direct step, where the cost is

Problem Information				BEPCM		BEPUM	
Name	m	n	$nnz(A)/mn$	M_1	M_2	M_1	M_2
czprob	737	3141	0.0041	36	50	18	30
d2q06c	2171	5831	0.0026	155	86	152	85
d6cube	404	6184	0.0151	97	62	94	56
stocfor2	2157	3045	0.0014	109	70	105	69
scsd8	397	2750	0.0079	25	20	17	13
agg3	516	758	0.0122	59	39	54	36
maros	835	1921	0.0063	53	37	49	34

Table 2: Break-even point for computing M (BEPCM) and break-even point for using M (BEPUM) in terms of q, obtained at interior-point iteration number 2. In the BEPUM column, number of iterations per PCGLS step is equal to one. The quantity $nnz(A)$ is the number of nonzero elements of A and $nnz(A)/mn$ represents the sparseness of A.

measured by floating point operations unless unless otherwise indicated.

Table 3 indicates that we need to do a few PCGLS iterations per PCGLS step for test problems `czprob` and `scsd8`. Recall that Table 1 showed that it is much more expensive to compute a dense solve than a sparse solve for these two problems. The small ratio $R_{C/D}$ in Table 1 for test problems `czprob` and `scsd8` supports the small break-even points for these two problems in Table 3.

Hence Tables 1 and 3 together suggest that we should use a large q and do 1 or 2 PCGLS iterations per PCGLS step for `czprob` and `scsd8`. For a given q, the break-even points in Table 3 if they were available during the computation, could be used as upper bounds for the number of iterations per PCGLS step. However, for most problems we do not get close to these numbers since we terminate after fewer iterations (except for `czprob` and `scsd8` problems). The break-even points in Table 3 are non-increasing with respect to q. The higher the value of q is, the smaller the break-even point is.

The results in Tables 1,2 and 3 suggest that for problems where the dense solve is many times more expensive than a sparse solve we choose q large and do a few PCGLS iterations per step, otherwise we can carefully choose a medium sized q and do a reasonable number of PCGLS iterations per step. The idea is to do few PCGLS iterations when the relative error ϵ is large and many PCGLS iterations when the error is less than a predefined constant. This is because we need more accurate steps in the last stage

BEPUM when q is fixed										
q	Problem Name									
	stocfor2		czprob		scsd8		d6cube		d2q06c	
	M_1	M_2	M_1	M_2	M_1	M_2	M_1	M_2	M_1	M_2
1	34	35	1	1	2	2	34	35	66	67
2	33	34	1	1	2	2	34	35	65	66
3	33	34	1	1	2	2	34	34	65	66
4	32	33	1	1	2	2	34	34	65	65
5	32	33	1	1	2	2	33	33	64	64
6	32	32	1	1	2	2	33	32	64	63
7	32	32	1	1	2	2				
8	31	31	1	1	1	1				
10	31	30	1	1	1	1	32	30	62	60
13	30	29	1	1	1	1				
14	30	28	1	1	1	0				
15	29	28	1	1	1					
17	29		1	1	1					
18	29		1	1	0					
19	29		0	1						
20	28	26		1			29	24	58	53
30	26	22		1			25	19	53	45
31	26			0						
40	23	17					22	12	50	40
50	20	12					18	5	46	32

Table 3: The break-even point for using M (BEPUM) in terms of PCGLS iterations, obtained at interior-point iteration number 2.

of the interior-point iterations to be able to attain both primal and dual feasibility.

The numerical experiments in Tables 1,2 and 3 provide guidelines for selecting q based on floating point operations. Similar experiments can be carried out based on CPU time and a similar analysis given.

4.2 Numerical Comparison

In Table 4, we compare the performance of the standard primal-dual Newton (PDN) algorithm with three variants of the mixed primal-dual Newton (Mixed PDN) algorithm: Mixed PDN^i, $i = 1, 2, 3$. When $\epsilon \geq 0.1$ we set $t = 7$, $q = 6$ for Mixed PDN^1; $t = 5$, $q = 20$ for Mixed PDN^2; and $t = 5$, $q = 40$ for Mixed PDN^3; where t is the maximum number of PCGLS iterations allowed. When $\epsilon < 0.1$ we set $t = 40$ for all three with their q values unchanged.

Problem	PDN		Mixed PDN^1		Mixed PDN^2		Mixed PDN^3	
name	iter	time	iter	time	iter	time	iter	time
czprob	56	89.01	59	59.58	58	56.52	58	56.83
d2q06c	54	544.85	57	421.33	61	442.24	55	405.17
d6cube	42	317.58	45	239.16	47	226.72	46	229.93
stocfor2	41	113.26	44	98.05	43	85.32	42	90.74
scsd8	16	21.41	19	17.49	19	17.31	19	17.33

Table 4: Comparison of four algorithms. The preconditioner M used is given by (24).

Recall that the termination criteria in PCGLS routine are when either the number of PCGLS iterations has reached a prescribed integer t or when $\|v\|_2 \leq 10^{-5}$ where $v = AGA^T\Delta y - AGh$.

When $\epsilon \geq 0.1$, in most cases PCGLS iterations are terminated by t other than the error $\|v\|_2$. On the other hand, when $\epsilon < 0.1$ PCGLS iterations are terminated by $\|v\|_2$ other than t. For very large problems PCGLS iterations per step may reach 40 depending on the size of q. We have set the maximum number of PCGLS iterations per step to 40. We are mostly interested in the value of t when $\epsilon \geq 0.1$. In this case numerical experiments show that we do not necessarily need an accurate PCGLS step. So when we increase q we can decrease t and vice versa to reduce the cost. The results in Table 4 show that by carefully choosing q and t we can keep the number of interior-point iterations for mixed PDN algorithms close to that of PDN algorithm, and yet still obtain better CPU time with mixed PDN methods.

5 Final Remarks

At present, the most practically efficient interior-point algorithmic framework for linear programming is the predictor-corrector algorithm of Mehrotra [14], along with Gondzio's multiple corrections [12]. See also [1]. Our comparison is done with the less efficient, but easier to implement, primal-dual Newton algorithm. Nevertheless, we believe that the results in this paper show that the mixed PDN approach is very promising and merits further study.

There are many possibilities for improving the efficiency of the mixed PDN approach. Obviously, the even-odd alternation is not generally optimal. Dynamic schemes of alternation based on easily computable bounds on condition numbers should be investigated. In addition, a mixture of the PCGLS steps with the predictor-corrector steps may also be promising.

References

[1] E.D. Andersen, J. Gondzio, C. Meszaros, X. Xu, *Implementation of interior-point methods for large scale linear programs,* T.Terlaky(ed), Interior Point Methods of Mathematical Programming pp. 189-252. Kluwer Academic Publishers, 1996. Printed in Netherlands.

[2] V. Baryamureeba, and T. Steihaug, *Computational issues for a new class of preconditioners,* To appear in the Proceedings for the 2nd Workshop on Large-Scale Scientific Computations, 1999.

[3] J.M. Bennett, *Triangular factors of Modified matrices*, Numerische Mathematik Vol. 7, pp. 217-221, 1965.

[4] Å. Björck, *Numerical methods for least squares problems*, SIAM, 1996.

[5] E.Y. Bobrovnikova, and Stephen A. Vavasis, *Accurate solution of weighted least squares by iterative methods*, Technical Report, Center for Applied Mathematics, Cornell University, Ithaca, New York 14853, 1997.

[6] J.W. Demmel, M.T. Heath, and H.A. Van der Vorst, *Parallel numerical linear algebra,* Acta Numerica, pp. 111-197, 1993.

[7] A.S. El-Bakry, R.A. Tapia, T. Tsuchiya, and Y. Zhang, *On the formulation of the primal-dual Newton interior-point method for nonlinear*

programming, Technical Report No. TR92-40, Rice University, Houston, Texas 77251, 1992.

[8] D.M. Gay, *Electronic mail distribution of linear programming test problems,* Mathematical Programming Society COAL Newsletter, No. 13, pp.10-12, 1985.

[9] P.E. Gill, G.H. Golub, W. Murray, and M.A. Saunders, *Methods for modifying matrix factorizations,* Mathematics of Computation, Vol. 28, pp. 505-535, 1974.

[10] D. Goldfarb, and S. Liu, *An* $O(n^3L)$ *primal interior-point algorithm for convex quadratic programming,* Mathematical Programming, Vol. 49, pp. 325–340, 1991.

[11] G.H. Golub, and C.H. Van Loan, *Matrix computations,* Third Edition, 1996.

[12] J. Gondzio, *Multiple centrality corrections in a primal-dual method for linear programming,* Computational Optimization and Applications. To appear.

[13] S. Mehrotra, *Implementations of affine scaling methods: Approximate solutions of systems of linear equations using preconditioned conjugate gradient methods,* ORSA J. Comput., 4(1992), pp. 103-118.

[14] S. Mehrotra, *On the implementation of a primal-dual interior point method,* SIAM J. Optimization, Vol. 2, pp. 575-601, 1992.

[15] J.H. Wilkinson, *The Eigenvalue Problem*, 1965.