INCOMPLETE FACTORIZATION CONSTRAINT PRECONDITIONERS FOR SADDLE-POINT MATRICES

H. S. DOLLAR* AND A. J. WATHEN*

Abstract. We consider the application of the conjugate gradient method to the solution of large symmetric, indefinite linear systems. Special emphasis is put on the use of constraint preconditioners and a new factorization that can reduce the number of flops required by the preconditioning step. Results concerning the eigenvalues of the preconditioned matrix and its minimum polynomial are given. Numerical experiments validate these conclusions.

1. Introduction. There are many areas where we wish to solve matrix systems of the form

$$\underbrace{\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix}}_{\mathcal{A}} \begin{bmatrix} x \\ y \end{bmatrix} = \underbrace{\begin{bmatrix} c \\ d \end{bmatrix}}_{b}, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric and $B \in \mathbb{R}^{m\times n}$. We shall assume that $m \leq n$, B is of full rank, and A is positive definite in the nullspace of B. The symmetric matrix $\mathcal{A}$ is generally indefinite (i.e. it has both positive and negative eigenvalues.)

Example 1.1 (convex quadratic programming problem). Consider the convex quadratic programming problem

$$\min_x f(x) = \tfrac{1}{2}x^T Ax + s^T x \text{ subject to } Bx = t, \tag{1.2}$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric, positive definite, $B \in \mathbb{R}^{m\times n}$ is the full row rank matrix of linear constraints and vectors x, s and t have appropriate dimensions. Any finite solution of (1.2) is a stationary point of the Lagrangian function

$$L(x, \lambda) = \frac{1}{2}x^T Ax + s^T x - \mu^T(Bx - t),$$

where the entries μ_i of the vector μ are referred to as *Lagrangian multipliers*. Differentiating L with respect to x and μ reveals that the solution of (1.2) satisfies $n + m$ linear equations of the form (1.1), with $y = -\mu$, $c = -s$, and $d = t$. These are known as the Karush-Kuhn-Tucker (KKT) conditions, [10].

Example 1.2 (saddle-point problems). Mixed finite element approximations of variational problems are expressible in the following common form where $\alpha(\cdot,\cdot)$ and $\beta(\cdot,\cdot)$ are bilinear forms defining the specific problem and $\langle\cdot,\cdot\rangle$ is an inner product.

Find u and p in relevant spaces such that

$$\begin{array}{rcl} \alpha(u,v) + \beta(v,p) & = & \langle f, v\rangle, \\ \beta(u,q) & = & \langle g, q\rangle \end{array} \tag{1.3}$$

for appropriate v and q. Hence (1.3) leads to systems of the form (1.1), where x are the coefficients of the approximation u and y are the coefficients of the approximation of p with respect to chosen bases. See, for example, Quarteroni and Valli [7, Chapters 7,9].

*Oxford University Computing Laboratory, Numerical Analysis Group, Wolfson Building, Parks Road, Oxford, OX1 3QD, U.K. (`hsd@comlab.ox.ac.uk, wathen@comlab.ox.ac.uk`).

Solution of systems of equations of the form (1.1) can be achieved by a number of methods. For large, sparse or structured matrices iterative methods are an attractive option. In particular, Krylov subspace methods apply techniques that involve orthogonal projections onto subspaces of the form

$$\mathcal{K}(\mathcal{A}, b) \equiv \operatorname{span}\{b, \mathcal{A}b, \mathcal{A}^2 b, \ldots, \mathcal{A}^{n-1}b, \ldots\}.$$

The conjugate gradient method (CG), minimum residual method (MINRES) and generalized minimal residual method (GMRES) are all common iterative Krylov subspace methods. The CG method is used for symmetric positive definite matrices, MINRES for symmetric and possibly indefinite matrices, and GMRES for unsymmetric matrices, [9].

MINRES and GMRES will find the solution of the linear system (1.1) within $n+m$ iterations in exact arithmetic, but CG may fail because of the system being indefinite. GMRES is just an expensive way to implement MINRES when $\mathcal{A}$ is symmetric as here. For very large systems this upper bound on the number of iterations is not helpful and such methods would generally not be employed unless many fewer iterations were required in practice. It is often advantageous to use a preconditioner, $\mathcal{P}$, with such iterative methods. The preconditioner should reduce the number of iterations required for convergence but not significantly increase the amount of computation required at each iteration, [9, Chapter 13].

Keller, Gould and Wathen [6] investigated the use of a preconditioner of the form

$$\mathcal{P} = \begin{bmatrix} G & B^T \\ B & 0 \end{bmatrix}, \tag{1.4}$$

where G approximates but is not the same as A. $\mathcal{P}$ is called a *constraint preconditioner*. The proof of the following theorem can be found in [6].

THEOREM 1.1. *Let $\mathcal{A} \in \mathbb{R}^{(n+m)\times(n+m)}$ be a symmetric and indefinite matrix of the form*

$$\mathcal{A} = \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix},$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric and $B \in \mathbb{R}^{m\times n}$ is of full rank. Assume Z is an $n \times (n-m)$ basis for the nullspace of B. Preconditioning $\mathcal{A}$ by a matrix of the form

$$\mathcal{P} = \begin{bmatrix} G & B^T \\ B & 0 \end{bmatrix},$$

where $G \in \mathbb{R}^{n\times n}$ is symmetric, $G \neq A$, and $B \in \mathbb{R}^{m\times n}$ is as above, implies that

- *the matrix $\mathcal{P}^{-1}\mathcal{A}$ has*
  1. *an eigenvalue at 1 with multiplicity $2m$, and*
  2. *$n-m$ eigenvalues λ which are defined by the generalized eigenvalue problem $Z^T A Z x_z = \lambda Z^T G Z x_z$.,*
- *the dimension of the Krylov subspace $\mathcal{K}(\mathcal{P}^{-1}\mathcal{A}, b)$ is at most $n-m+2$.*

The system (1.1) is symmetric and indefinite, implying that we are unable to apply the CG method directly in a robust way since it might fail, [1]. Gould, Hribar and Nocedal [5] show how the CG method can still be used when solving quadratic programming problems. We extend their method to the general problem of solving systems of the form (1.1) where a constraint preconditioner of the form (1.4) is also used in Sections 2 and 3. In Section 4 we introduce a new factorization for the preconditioner $\mathcal{P}$. This, previously unpublished, factorization was recently developed by Wil Schilders (Tech. Univ. Eindhoven/Philips), [8]. We shall refer to this as the Schilders Factorization. In Section 5 we give numerical examples to demonstrate the effectiveness the use of this method.

2. CG method for the reduced system. We wish to solve the system (1.1) to find $[x^{*T} y^{*T}]^T$. Let Z be an $n \times (n-m)$ matrix spanning the nullspace of B, then $BZ = 0$. The columns of B^T together with the columns of Z span $\mathbb{R}^n$ and any solution x^* of linear equations $Bx = d$ can be written as

$$x^* = B^T x^*_{_B} + Z x^*_{_Z}. \tag{2.1}$$

Substituting this into (1.1) gives

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} B^T x^*_{_B} + Z x^*_{_Z} \\ y^* \end{bmatrix} = \begin{bmatrix} c \\ d \end{bmatrix}. \tag{2.2}$$

Let us split the matrix $\mathcal{A}$ into a block 3×3 structure where each corner block is of dimension m by m. We can also expand out the vector $[x^{*T} y^{*T}]^T$ into a matrix-vector product, $\mathcal{H}\rho^*$. Let $Z = [Z_1^T Z_2^T]^T$. Expression (2.2) then becomes

$$\begin{bmatrix} A_{1,1} & A_{1,2} & B_1^T \\ A_{2,1} & A_{2,2} & B_2^T \\ B_1 & B_2 & 0 \end{bmatrix} \underbrace{\begin{bmatrix} B_1^T & Z_1 & 0 \\ B_2^T & Z_2 & 0 \\ 0 & 0 & I \end{bmatrix}}_{\mathcal{H}} \underbrace{\begin{bmatrix} x^*_{_B} \\ x^*_{_Z} \\ y^* \end{bmatrix}}_{\rho^*} = \begin{bmatrix} c_1 \\ c_2 \\ d \end{bmatrix}. \tag{2.3}$$

To maintain symmetry of our system we premultiply (2.3) by $\mathcal{H}^T$. Multiplying out the matrix expression $\mathcal{H}^T \mathcal{A} \mathcal{H}$ and simplifying, we obtain the linear system

$$\begin{bmatrix} BAB^T & BAZ & BB^T \\ Z^T AB^T & Z^T AZ & 0 \\ BB^T & 0 & 0 \end{bmatrix} \begin{bmatrix} x^*_{_B} \\ x^*_{_Z} \\ y^* \end{bmatrix} = \begin{bmatrix} Bc \\ Z^T c \\ d \end{bmatrix}. \tag{2.4}$$

We observe that an $m \times m$ system determines $x^*_{_B}$:

$$BB^T x^*_{_B} = d. \tag{2.5}$$

Since B is of full rank, BB^T is symmetric, positive definite. We can therefore solve this system using the Cholesky factorization of BB^T, [3, p. 143]. From (2.4), having found $x^*_{_B}$ we can find $x^*_{_Z}$ by solving

$$A_{_{ZZ}} x^*_{_Z} = -c_{_Z}, \tag{2.6}$$

where

$$A_{_{ZZ}} = Z^T AZ, \qquad c_{_Z} = Z^T(AB^T x^*_{_B} - c).$$

The matrix A is symmetric and positive definite in the nullspace of B, hence $Z^T AZ$ is symmetric, positive definite. Anticipating our technique, we can apply the CG method to compute an approximate solution to the system (2.6). Substituting this into (2.1) will give us an approximate solution for x^*. Using (2.4) with (2.1) we obtain a system that can be solved to give an approximate solution for y^*:

$$BB^T y^* = B(c - Ax^*) \tag{2.7}$$

This system can be solved using the Cholesky factorization that we calculate when finding ${x_B}^*$.

Let us consider the practical application of the CG method to the system (2.6). As we noted in Section 1, the use of preconditioning can improve the rate of convergence of the CG iteration. Let us assume that a preconditioner W_{ZZ} is given, where W_{ZZ} is a symmetric, positive definite matrix of dimension $n - m$. Let us consider the class of preconditioners of the form $W_{ZZ} = Z^T G Z$, where G is a symmetric matrix such that $Z^T G Z$ is positive definite. The preconditioned CG method applied to the $(n-m)$-dimensional reduced system $A_{ZZ} {x_Z}^* = -c_Z$ is as follows [3, p. 532].

ALGORITHM 2.1 (preconditioned CG for reduced systems). *Choose an initial point* x_Z*, compute* $r_Z = Z^T A Z x_Z + c_Z$*,* $g_Z = (Z^T G Z)^{-1} r_Z$ *and* $p_Z = -g_Z$*. Repeat the following steps, until a termination step is satisfied:*

$$\alpha = {r_Z}^T g_Z / {p_Z}^T Z^T A Z p_Z, \tag{2.8}$$

$$x_Z \leftarrow x_Z + \alpha p_Z, \tag{2.9}$$

$${r_Z}^+ = r_Z + \alpha Z^T A Z p_Z, \tag{2.10}$$

$${g_Z}^+ = (Z^T G Z)^{-1} {r_Z}^+, \tag{2.11}$$

$$\beta = ({r_Z}^+)^T {g_Z}^+ / {r_Z}^T g_Z, \tag{2.12}$$

$$p_Z \leftarrow -{g_Z}^+ + \beta p_Z, \tag{2.13}$$

$$g_Z \leftarrow {g_Z}^+ \quad \textit{and} \quad r_Z \leftarrow {r_Z}^+. \tag{2.14}$$

Gould, Hribar and Nocedal [5] suggest terminating this iteration when ${r_Z}^T (Z^T G Z)^{-1} r_Z$ is sufficiently small. In the next section, we modify this algorithm to avoid operating with the null space basis Z.

3. CG method for the full system. If we were to use Algorithm 2.1 to compute an approximate solution, it must be multiplied by Z and substituted into (2.1). Alternatively, Algorithm 2.1 may be rewritten so that the multiplication by Z and the addition of the term $B^T {x_B}^*$ is computed within the CG iteration, [5]. In the following algorithm, the n-vectors x, r, g, p satisfy $x = Z x_Z + B^T {x_B}^*$, $Z^T r = r_Z$, $g = Z g_Z$, and $p = Z g_Z$. We also define the scaled projection matrix

$$P = Z (Z^T G Z)^{-1} Z^T. \tag{3.1}$$

We will later see that P is independent of the choice of null space basis Z.

ALGORITHM 3.1 (preconditioned CG in expanded form). *Choose an initial point* x *satisfying* $Bx = d$*, compute* $r = Ax - c$*,* $g = Pr$*, and* $p = -g$*. Repeat the following steps, until a convergence test is satisfied:*

$$\alpha = r^T g / p^T A p, \tag{3.2}$$

$$x \leftarrow x + \alpha p, \tag{3.3}$$

$$r^+ = r + \alpha A p, \tag{3.4}$$

$$g^+ = P r^+, \tag{3.5}$$

$$\beta = (r^+)^T g^+ / r^T g, \tag{3.6}$$

$$p \leftarrow -g^+ + \beta p, \tag{3.7}$$

$$g \leftarrow g^+ \quad \textit{and} \quad r \leftarrow r^+. \tag{3.8}$$

Note the definition of g^+ via the projection step (3.5). Following the terminology of [5], the vector g^+ will be called the *preconditioned residual*. It is defined to be in the null space of B. We now wish to be able to apply the projection operator $Z(Z^TGZ)^{-1}Z^T$ without a representation of the null space basis Z. If G is nonsingular, then P can be expressed as

$$P = G^{-1}(I - B^T(BG^{-1}B^T)^{-1}BG^{-1}). \tag{3.9}$$

We can find g^+ by solving the system

$$\begin{bmatrix} G & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} g^+ \\ v^+ \end{bmatrix} = \begin{bmatrix} r^+ \\ 0 \end{bmatrix} \tag{3.10}$$

whenever $z^TGz \neq 0$ for all nonzero z for which $Bz = 0$, [2, section 5.4.1]. The idea in Algorithm 3.2 below is to replace (3.5) with the solution of (3.10) to define the same g^+: this is why a constraint preconditioner of the form (1.4) is needed.

Discrepancy in the magnitudes of g^+ and r^+ can cause numerical difficulties for which Gould, Hribar and Nocedal [5] suggest using a residual update strategy that redefines r^+ so that its norm is closer to that of g^+. This dramatically reduces the roundoff errors in the projection operation in practice.

ALGORITHM 3.2 (preconditioned CG with residual update (PCG)). *Apply Algorithm 3.1 with the following changes:*

replace $g = Pr$ by solve

$$\begin{bmatrix} G & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} g \\ v \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix},$$

set $y = v$ and $r \leftarrow r - B^Ty$ at the end of the initialization stage,
replace (3.5) by solve (3.10),
replace (3.8) by $g \leftarrow g^+$ and $r \leftarrow r^+ - B^Tv^+$.

We would therefore like to be able to solve the systems of the form (3.10) efficiently. If we were to use Gaussian Elimination to solve this system then, in general, the number of flops required would be $\mathcal{O}((n+m)^3)$, [3, p. 98]. Though this may be reduced depending on the sparsity, it could be impractical for large systems. In the following section we introduce a different factorization for matrices of the form $\mathcal{P}$ in (1.4). If $2m \leq n$, then we can use this factorization to solve (3.10) in $\mathcal{O}((n-m)^3)$ flops. If $2m > n$, then the number of flops required is $\mathcal{O}(m^3)$.

4. Schilders' factorization for preconditioning step. The preconditioning step of solving (3.10) in Algorithm 3.2 involves a constraint preconditioner of the same form of $\mathcal{P}$ given in (1.4). Let us split $\mathcal{P}$ into a block 3×3 structure as we did for the matrix $\mathcal{A}$ in (2.3). Suppose we choose matrices $L_1 \in \mathbb{R}^{m \times m}$ and $L_2 \in \mathbb{R}^{(n-m)\times(n-m)}$ such that L_2 is nonsingular, and assume that B_1 is nonsingular, then we can factorize $\mathcal{P}$ in the following manner:

$$\mathcal{P} = \underbrace{\begin{bmatrix} {B_1}^T & 0 & L_1 \\ {B_2}^T & L_2 & E \\ 0 & 0 & I \end{bmatrix}}_{\mathcal{P}_1} \underbrace{\begin{bmatrix} D_1 & 0 & I \\ 0 & D_2 & 0 \\ I & 0 & 0 \end{bmatrix}}_{\mathcal{P}_2} \underbrace{\begin{bmatrix} B_1 & B_2 & 0 \\ 0 & {L_2}^T & 0 \\ {L_1}^T & E^T & I \end{bmatrix}}_{\mathcal{P}_3}, \tag{4.1}$$

where

$$D_1 = {B_1}^{-T}G_{1,1}{B_1}^{-1} - {L_1}^T{B_1}^{-1} - {B_1}^{-T}L_1, \tag{4.2}$$

$$D_2 = {L_2}^{-1}(G_{2,2} - {B_2}^TD_1B_2 - EB_2 - {B_2}^TE^T){L_2}^{-T}, \tag{4.3}$$

$$E = G_{2,1}{B_1}^{-1} - {B_2}^TD_1 - {B_2}^T{L_1}^T{B_1}^{-1}, \tag{4.4}$$

Equations (4.2), (4.3) and (4.4) are such that we can choose the matrix G and use this to define D_1, D_2 and E, or we could choose matrices D_1, D_2 and E that then define the matrix G. If we choose to use the second option, then we need to make sure that G satisfies several criteria set out in Sections 2 and 3. These are

- G is non-singular, and
- Z^TGZ is symmetric, positive definite.

Since $\mathcal{P}_1(=\mathcal{P}_3^T)$ is nonsingular, G will certainly satisfy these criteria if $\mathcal{P}_2$ is symmetric, positive definite. This is equivalent to the simple condition that D_2 is symmetric and positive definite.

In using the factorization (4.1) to solve (3.10), we can calculate the `lu` factorizations of $B_1 \in \mathbb{R}^{m\times m}$, $D_2 \in \mathbb{R}^{(n-m)\times(n-m)}$ and $L_2 \in \mathbb{R}^{(n-m)\times(n-m)}$ (which we take to be I) once and then use the factored forms in each iteration. If i iterations are carried out in Algorithm 3.2, then there are $i+1$ solves carried out using the factorization (4.1). For general B, D_1, D_2, E, L_1 and L_2, the total number of flops required by PCG is given by

$$\textit{no.}\quad \textit{flops.} \sim \frac{2}{3}(2(n-m)^3+m^3)+2(i+1)(13m^2+3n^2).$$

However, if D_1 and L_2 are diagonal, then the number of flops required reduces to

$$\textit{no.}\quad \textit{flops.} \sim \frac{2}{3}m^3+4m(i+1)(5m+3n).$$

We would like to balance this reduction in the number of flops with the possible increase in the number of iterations carried out.

Let us consider what happens if $G_{1,1}=A_{1,1}$, $G_{1,2}={G_{2,1}}^T=A_{1,2}$, and we choose some matrix D_2 which is symmetric, positive definite. Equation (4.3) can be rearranged to give an explicit expression for $G_{2,2}$: this is given in Theorem 4.1. We assume that $G_{2,2}\neq A_{2,2}$. By choosing D_2 and using the factorization (4.1) to solve (3.10) we will never have to explicitly form the matrix $\mathcal{P}$.

Theorem 1.1 reveals that the preconditioned system $\mathcal{P}^{-1}\mathcal{A}$ has an eigenvalue at 1 with multiplicity $2m$, and $n-m$ eigenvalues which are defined by the generalized eigenvalue problem $Z^TAZx_z=\lambda Z^TGZx_z$. Noting that B_1 is non-singular, we observe that $Z=[-{B_2}^T{B_1}^{-T}\;\; I]^T{L_2}^{-T}$ is an $n\times(n-m)$ basis for the nullspace of B. Using the above values of $G_{1,1}$, $G_{1,2}$, $G_{2,1}$ and $G_{2,2}$, it is straightforward to show that the $n-m$ eigenvalues can be defined by

$$\widehat{D}_2x_z=\lambda D_2x_z, \tag{4.5}$$

where

$$\widehat{D}_2={L_2}^{-1}(A_{2,2}-{B_2}^TD_1B_2-EB_2-{B_2}^TE^T){L_2}^{-T}. \tag{4.6}$$

Note: both D_2 and $\widehat{D}_2$ are symmetric, positive definite, so the eigenvalues are all real. Using the preconditioning matrix $\mathcal{P}$ with $G=[A_{1,1}\;\; A_{1,2};\;\; A_{2,1}\;\; G_{2,2}]$, we can improve on the Krylov subspace dimension given in Theorem (1.1):

THEOREM 4.1. *Let $\mathcal{A}\in\mathbb{R}^{(n+m)\times(n+m)}$ be a symmetric and indefinite matrix of the form*

$$\mathcal{A}=\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix},$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric and $B \in \mathbb{R}^{m\times n}$ is of full rank with the first m columns linearly dependent of each other. Let $A = [A_{1,1}\ \ A_{1,2};\ A_{2,1}\ \ A_{2,2}]$ and $B = [B_1\ \ B_2]$, where $A_{1,1}, B_1 \in \mathbb{R}^{m\times m}$, $A_{1,2} \in \mathbb{R}^{m\times(n-m)}$, $A_{2,1}, B_2 \in \mathbb{R}^{(n-m)\times m}$ and $A_{2,2} \in \mathbb{R}^{(n-m)\times(n-m)}$.

Assume that $m < n$ and $\mathcal{A}$ is nonsingular. Let us choose any matrices $L_1 \in \mathbb{R}^{m\times m}$, $L_2, D_2 \in \mathbb{R}^{(n-m)\times(n-m)}$ such that L_2 is nonsingular and D_2 is symmetric, positive definite. Assume $\mathcal{A}$ is preconditioned by a matrix of the form

$$\mathcal{P} = \begin{bmatrix} A_{1,1} & A_{1,2} & B_1^T \\ A_{2,1} & G_{2,2} & B_2^T \\ B_1 & B_2 & 0 \end{bmatrix},$$

where

$$G_{2,2} = L_2 D_2 {L_2}^T - {B_2}^T {B_1}^{-T} A_{1,1} {B_1}^{-1} B_2 + A_{2,1} {B_1}^{-1} B_2 + {B_2}^T {B_1}^{-T} A_{1,2},$$

then the dimension of the Krylov subspace $\mathcal{K}(\mathcal{P}^{-1}\mathcal{A}, b)$ is at most $n - m + 1$.

Proof. By assumption, B_1 is nonsingular, so we can use the Schilders' factorization (4.1) to express both $\mathcal{A}$ and $\mathcal{P}$. Using these factorizations we can express $\mathcal{P}^{-1}\mathcal{A}$ as

$$\mathcal{P}^{-1}\mathcal{A} = \begin{bmatrix} I & \Theta & 0 \\ 0 & {L_2}^{-T} {D_2}^{-1} \widehat{D}_2 {L_2}^T & 0 \\ 0 & \Upsilon & I \end{bmatrix}, \tag{4.7}$$

where D_1, $\widehat{D}_2$, and E are given by Equations (4.2), (4.6) and (4.4) respectively, and

$$\Theta = {B_1}^{-1} B_2 (I - {L_2}^{-T} {D_2}^{-1} \widehat{D}_2 {L_2}^T), \tag{4.8}$$

$$\Upsilon = (E^T - {L_1}^T {B_1}^{-1} B_2)(I - {L_2}^{-T} {D_2}^{-1} \widehat{D}_2 {L_2}^T). \tag{4.9}$$

From the eigenvalues of the preconditioned system, it is evident that the characteristic polynomial of the preconditioned system is of the form

$$(\mathcal{P}^{-1}\mathcal{A} - I)^{2m} \prod_{i=1}^{n-m} (\mathcal{P}^{-1}\mathcal{A} - \lambda_i I).$$

To prove the upper bound on the dimension of the Krylov subspace we need to show that the degree of the minimum polynomial is less than or equal to $n - m + 1$. Expanding the polynomial $\prod_{i=1}^{n-m}(\mathcal{P}^{-1}\mathcal{A} - \lambda_i I)$ of degree $n - m$, we obtain a matrix of the form

$$\begin{bmatrix} \prod_{i=1}^{n-m}(1-\lambda_i)I & f_{n-m}(\Theta) & 0 \\ 0 & \prod_{i=1}^{n-m}(S - \lambda_i I) & 0 \\ 0 & f_{n-m}(\Upsilon) & \prod_{i=1}^{n-m}(1-\lambda_i)I \end{bmatrix}, \tag{4.10}$$

where $S = {L_2}^{-T} {D_2}^{-1} \widehat{D}_2 {L_2}^T$ and $f_{m-n}(\cdot)$ is defined by the recursive formula

$$f_{n-m}(\Gamma) = \Gamma \prod_{i=1}^{n-m-1} (1 - \lambda_i) I + (S - \lambda_{n-m} I) f_{n-m}(\Gamma)$$

with base case $f_1(\Gamma) = \Gamma$. Now let us premultiply the matrix (4.10) by $\mathcal{P}^{-1}\mathcal{A} - I$ to give

$$\begin{bmatrix} 0 & \Theta \prod_{i=1}^{n-m}(S - \lambda_i I) & 0 \\ 0 & (S - \lambda I) \prod_{i=1}^{n-m}(S - \lambda_i I) & 0 \\ 0 & \Upsilon \prod_{i=1}^{n-m}(S - \lambda_i I) & 0 \end{bmatrix}. \tag{4.11}$$

We note that the (1,2), (2,2), and (3,2) entries are in fact zero, since the λ_i ($i = 1, \ldots, n-m$) are eigenvalues of S, which is similar to a symmetric matrix and thus diagonalizable. Hence, the degree of the minimum polynomial of $\mathcal{P}^{-1}\mathcal{A}$ is less than or equal to $n - m + 1$. □

We note that if $\lambda_i \neq 1$ for $i = 1, \ldots, j < n-m$ and $\lambda_i = 1$ for $i = j+1, \ldots, n-m$ then $\prod_{i=1}^{j+1}(S - \lambda_i I)$ will be zero. Hence, the polynomial $(\mathcal{P}^{-1}\mathcal{A} - I)^{2m}\prod_{i=1}^{j+1}(\mathcal{P}^{-1}\mathcal{A} - \lambda_i I)$ of degree $j+2$ is the minimum polynomial of $\mathcal{P}^{-1}\mathcal{A}$.

5. Numerical Examples. We apply Algorithm 3.2 to solve a selection of problems from the CUTEr collection [4] where any simple bounds are removed to create a problem of the form

$$\begin{aligned} \min_x f(x) &= \tfrac{1}{2}x^T A x + s^T x \\ \text{subject to } Bx &= t, \end{aligned}$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric and $B \in \mathbb{R}^{m\times n}$ is the full row rank matrix of linear constraints. A is positive definite in the nullspace of B, and vectors x, s and t have appropriate dimensions. If the first m columns of B are linearly dependent, then we carry out a permuted `QR` factorization to find a permutation matrix, M, such that the first m columns of BM are linearly independent, [3, p. 248]. We then set $B \leftarrow BM$, $A \leftarrow M^T AM$, $s \leftarrow M^T s$ and $x \leftarrow M^T x$. The assumption about the nonsingularity of B_1 will now hold.

Table 5.1
Values of m, n, nonzeros in A and nonzeros in B for the test problems

Problem	m	n	nz(A)	nz(B)
CVXQP1_M	500	1000	6968	1498
CVXQP3_M	750	1000	6978	2247
DPKLO1	77	133	77	1575
DUAL1	1	85	7031	85
DUAL2	1	96	8920	96
DUAL3	1	111	12105	111
GOULDQP3	349	699	2092	1047
MOSARQP2	600	1500	990	3530

The dimensions of the problems and number of nonzeros in the associated matrices A and B are given in Table 5.1. Tables 5.2, 5.3 and 5.4 show the results of using Algorithm 3.2 with different preconditioners. The results of using the preconditioner $\mathcal{P} = [I\ B^T; B\ 0]$ are shown in Table 5.2. This preconditioner is of the general form considered in [6]. In Tables 5.3 and 5.4 we use preconditioners of the form considered in Theorem 4.1. We compare the number of iterations and the amount of CPU time taken when we use an `lu` factorization of the preconditioner $\mathcal{P}$, and when we use the Schilders' factorization for $\mathcal{P}$. In both cases, we carry out the factorization once and then use the factored forms in each iteration. When producing the Schilders' factorization, we set $L_1 = 0$ and $L_2 = I$. At present, we have not carried out substantial research into how we should choose the values of L_1 and L_2.

The number of iterations carried out is given in the column headed "Its". We also measure the CPU time taken in carrying out the factorization of the preconditioners (FTime) and the CPU time then taken by Algorithm 3.2 when using these factored forms (ITime). The total CPU time used to solve the problem is given by "Total". Each problem is solved ten times and the mean of the CPU times recorded. We also compare the minimum value of $\frac{1}{2}x^T Ax + s^T x$ calculated. The number of digits of

Table 5.2
Solution Statistics: comparisons between iterative (PCG) solvers when using backslash and the Schilders's Factorization in the preconditioning step, $G = I$.

	LU				Schilders'				
Problem	Its	FTime	ITime	Total	Its	FTime	ITime	Total	Digits
CVXQP1_M	237	0.158	10.7	10.9	239	0.533	3.91	4.44	14
CVXQP3_M	73	0.817	4.93	5.74	73	0.70	2.21	2.91	13
DPKLO1	4	0.025	0.025	0.050	4	0.022	0.023	0.045	15
DUAL1	74	0.002	0.076	0.078	74	0.030	0.095	0.125	9
DUAL2	38	0.001	0.050	0.051	38	0.035	0.064	0.099	9
DUAL3	36	0.004	0.057	0.061	36	0.040	0.090	0.130	11
GOULDQP3	18	0.003	0.069	0.072	18	0.769	0.204	0.973	15
MOSARQP2	44	0.065	1.095	1.16	44	1.46	1.12	2.58	16

Table 5.3
Solution Statistics: comparisons between iterative (PCG) solvers when using backslash and the Schilders's Factorization in the preconditioning step, $G_{1,1} = A_{1,1}$, $G_{1,2} = G_{2,1}^T = A_{1,2}$ and $D_2 = diag(L_2^{-1}(A_{2,2} + B_2^T B_1^{-T} A_{1,1} B_1^{-1} B_2 - A_{2,1} B_1^{-1} B_2 - (A_{2,1} B_1^{-1} B_2)^T) L_2^{-T})$.

	LU				Schilders'				
Problem	Its	FTime	ITime	Total	Its	FTime	ITime	Total	Digits
CVXQP1_M	502	8.70	114.8[1]	123.5	502	0.93	5.95[1]	6.88	7
CVXQP3_M	83	16.60	31.10	48.7	78	1.02	2.47	3.49	12
DPKLO1	14	0.046	0.045	0.091	14	0.027	0.05	0.077	6
DUAL1	54	0.039	0.078	0.117	50	0.032	0.056	0.088	6
DUAL2	28	0.045	0.057	0.102	28	0.033	0.036	0.069	6
DUAL3	36	0.055	0.103	0.158	28	0.038	0.048	0.086	16
GOULDQP3	172	2.74	13.16	15.90	126	0.534	0.714	1.25	9
MOSARQP2	20	3.70	2.05	9.75	20	0.108	0.285	0.393	15

Table 5.4
Solution Statistics: comparisons between iterative (PCG) solvers when using backslash and the Schilders's Factorization in the preconditioning step, $G_{1,1} = A_{1,1}$, $G_{1,2} = G_{2,1}^T = A_{1,2}$ and $D_2 = L_2^{-1} A_{2,2} L_2^{-T}$.

	LU				Schilders'				
Problem	Its	FTime	ITime	Total	Its	FTime	ITime	Total	Digits
CVXQP1_M	502	8.75	126.6[1]	135.4	502	0.05	6.42[1]	6.47	5
CVXQP3_M	208	15.88	77.82	93.7	138	0.07	3.94	4.01	10
DPKLO1	11	0.037	0.057	0.094	11	0.003	0.046	0.049	12
DUAL1	4	0.039	0.010	0.049	4	0.013	0.008	0.021	13
DUAL2	4	0.044	0.014	0.058	4	0.016	0.015	0.031	14
DUAL3	2	0.054	0.011	0.065	2	0.023	0.009	0.032	15

agreement in these values is given in "Digits". We observe that the differences in rounding errors of the two factorization methods do not greatly affect the calculated minimum values. In all of our test problems we obtain at lest six digits of agreement.

The diagonal form of D_2 used in the examples of Table 5.3 produces a significant drop in the CPU times when we use the Schilders' factorization instead of the `lu` factorization of $\mathcal{P}$. However, this choice of D_2 is not so practical because we have to form the matrix $\widehat{D}_2$, as defined in equation (4.6), and then use the diagonal. If $A_{2,2}$ is positive definite and sparse enough for us to consider factorizing it, then we can set $D_2 = L_2^{-1} A_{2,2} L_2^{-T}$. In this case, the eigenvalue system (4.5) will have at least $n - 3m$ eigenvalues at 1. Hence, the dimension of the Krylov subspace $\mathcal{K}(\mathcal{P}^{-1}\mathcal{A}, b)$ is at most $\min(n - m + 1, 2m + 2)$. Such a preconditioner is used in the examples of Table 5.4. The problems GOULDQP3 and MOSARQP2 have been excluded because of the singularity of their respective $A_{2,2}$ matrices.

All tests were performed on an XP2500+ Athlon with 256MB RAM and running Windows XP Professional. Programs were written in MATLAB® 6.0. We terminate the iteration when $r^T g \leq 10^{-6}$. In these experiments we also terminate if the number of iterations exceeds $n - m + 2$; a superscript ^{1} in Tables 5.2, 5.3 and 5.4 indicates when this limit is reached.

6. Conclusion. In this paper, we have discussed the use of the Preconditioned Conjugate Gradient method for solving indefinite linear systems. We have investigated the use of a new factorization for constraint preconditioners and shown how this can be used in incomplete form to decrease the number of flops required overall by the preconditioned iterative method. We obtained an upper bound on the number of iterations required to solve systems of the form (1.1) by means of appropriate Krylov subspace methods by using a minimum polynomial argument.

We have provided computational evidence that the Preconditioned Conjugate Gradient method works well for nontrivial quadratic programming problems. In some cases, the use of the Schilders' factorization to define the preconditioner used can speed up the solution time by a factor of 20 compared to using the `lu` factorization of the same preconditioner.

Acknowledgements. The authors would like to thank Nick Gould and Wil Schilders for their helpful input during the early stages of this research.

REFERENCES

[1] B. Fischer, *Polynomial based iteration methods for symmetric linear systems*, Wiley-Teubner Series Advances in Numerical Mathematics, John Wiley & Sons Ltd., Chichester, 1996.
[2] P. E. Gill, W. Murray, and M. H. Wright, *Practical optimization*, Academic Press Inc. [Harcourt Brace Jovanovich Publishers], London, 1981.
[3] G. H. Golub and C. F. Van Loan, *Matrix computations*, Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, Baltimore, MD, third ed., 1996.
[4] N. Gould, D. Orban, and P. Toint, *CUTEr (and SifDec), a constrained and unconstrained testing environment, revisited*, Tech. Report TR/PA/01/04, CERFACS, 2001.
[5] N. I. M. Gould, M. E. Hribar, and J. Nocedal, *On the solution of equality constrained quadratic programming problems arising in optimization*, SIAM J. Sci. Comput., 23 (2001), pp. 1376–1395 (electronic).
[6] C. Keller, N. I. M. Gould, and A. J. Wathen, *Constraint preconditioning for indefinite linear systems*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1300–1317 (electronic).
[7] A. Quarteroni and A. Valli, *Numerical approximation of partial differential equations*, vol. 23 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin, 1994.
[8] W. Schilders, *Private communication*, 2002.
[9] H. A. van der Vorst, *Iterative Krylov Methods for Large Linear Systems*, Cambridge Monographs on Applied and Computationl Mathematics, Cambridge University Press, Cambridge, United Kingdom, 1 ed., 2003.
[10] S. J. Wright, *Primal-dual interior-point methods*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.