4-1997

Least-Squares Approximate Solution of Overdetermined Sylvester Equations

A. Scottedward Hodel
Auburn University Main Campus

Pradeep Misra
Wright State University - Main Campus, pradeep.misra@wright.edu

SIAM J. MATRIX ANAL. APPL.
Vol. 18, No. 2, pp. 279–290, April 1997

002

LEAST-SQUARES APPROXIMATE SOLUTION OF OVERDETERMINED SYLVESTER EQUATIONS*

A. SCOTTEDWARD HODEL† AND PRADEEP MISRA‡

Abstract. We address the problem of computing a low-rank estimate Y of the solution X of the Lyapunov equation $AX + XA' + Q = 0$ *without* computing the matrix X itself. This problem has applications in both the reduced-order modeling and the control of large dimensional systems as well as in a hybrid algorithm for the rapid numerical solution of the Lyapunov equation via the alternating direction implicit method. While no known methods for low-rank approximate solution provide the two-norm optimal rank k estimate X_k of the exact solution X of the Lyapunov equation, our iterative algorithms provide an effective method for estimating the matrix X_k by minimizing the error $\|AY + YA' + Q\|_F$.

Key words. Sylvester equation, least squares, iterative, conjugate gradient

AMS subject classifications. 15A06, A5A24, 65F05

PII. S0895479893252337

1. Introduction.

The Lyapunov equation

$$AX + XA' + Q = 0, \tag{1.1}$$

$A, Q \in \mathbb{R}^{n\times n}, Q = Q'$ plays a significant role in numerous problems in control, communication systems theory, and power systems. Recent applications of the Lyapunov equation include the design of reduced-order state estimators and controllers [2], [20], [28], [29] and the solution of robust decentralized control problems [30], [33]. The Lyapunov equation also has applications in stability analysis [21], [25]. Standard methods for the numerical solution of the Lyapunov equation [1], [12] make use of the real Schur decomposition $A = USU'$, where U is an orthogonal matrix and S is quasi-upper triangular. The matrix U is used to transform the Lyapunov equation (1.1) into a form that is readily solved through forward substitution. More recently, Lu [26] and Wachspress [34] proposed the use of the alternating direction implicit (ADI) method for the iterative solution of Lyapunov equations for which all eigenvalues of the matrix A (or of $-A$) are in the right half of the complex plane.

Recently proposed numerical techniques for the numerical solution of the Lyapunov equation have involved iterative solution techniques [23], [19], [31] or low-rank approximate solution techniques [17], [18], [22]. Each of these methods requires the numerical solution of either a reduced-order Lyapunov equation

$$(V'AV)\Sigma_V + \Sigma_V(V'A'V') + V'QV = 0 \tag{1.2}$$

or a least-squares problem

$$\Sigma_V = \arg\min \|AV\Sigma_V V' + V\Sigma_V V'A' + Q\|_F . \tag{1.3}$$

*Received by the editors July 19, 1993; accepted for publication (in revised form) by S. J. Hammarling April 1, 1996. This work was supported in part by NSF grant ECS-9110083.
http://www.siam.org/journals/simax/18-2/25233.html

†Department of Electrical Engineering, 200 Broun Hall, Auburn University, Auburn, AL 36849 (scotte@eng.auburn.edu).

‡RC-311, Department of Electrical Engineering, Wright State University, Dayton, OH 45435 (pmisra@valhalla.cs.wright.edu).

The numerical solution of generalized Lyapunov equations

$$AXB' + BXA' + C = 0$$

may be achieved through the use of a QZ decomposition [27] of the matrix pencil (A, B) [7], [8]; low-rank approximate solution techniques may be applied to these problems in a fashion analogous to the standard case (1.1).

In this paper, we address the least-squares solution of minimizations of the form

$$\min_X \|AXB' + CXD' + F\|_F, \tag{1.4}$$

where (for simplicity in exposition) A, B, C, and $D \in \mathbb{R}^{n\times k}$, $X \in \mathbb{R}^{k\times k}$, $F \in \mathbb{R}^{n\times n}$, and $k \ll n$. Note that (1.3) then simply reduces to a special case of (1.4). The minimization (1.4) can be transformed to a minimization of the form

$$\min \left\|\bar{A}\bar{x} + \bar{b}\right\|_2 \tag{1.5}$$

through a Kronecker product expansion; see [24]. Techniques for the solution of large, sparse least-squares problems (1.5) have been addressed in several iterative algorithms, e.g., [9], [13], [32]. It should be noted that, unlike the Kronecker product expansion of the Lyapunov equation (1.1), the Kronecker product expansion (1.5) of the least-squares minimization (1.4) yields a *dense* matrix $\bar{A}$ in general, since no sparsity structure can be assumed for the matrices A, B, C, D, and F in applications that do not involve Krylov subspaces [15], [16], [18], [19].

It should be noted that a difficulty associated with flexible structures (second-order PDEs) that does not usually occur in heat flow problems is that the discretizations $\dot{x} = Ax + Bu$ do not automatically satisfy the constraint $A + A' < 0$ discussed in [15] and [16]. Hence, a least-squares approach as proposed in this paper becomes preferable to a reduced-order Lyapunov equation (the approach studied at length in [15] and [31]).

In the case $k = n$, (1.4) becomes a generalized Sylvester equation

$$AXB' + CXD' + F = 0, \tag{1.6}$$

which can be solved by reduction of the matrix pencils (A, C) and (D, B) to Schur-triangular form and Hessenberg-triangular form [27], respectively, and then by applying a modified version of the Golub–Nash–Van Loan algorithm [10]. Unfortunately, this approach is not directly applicable to the minimization (1.4); in particular, if $\operatorname{rank}\left(\begin{bmatrix} A & C \end{bmatrix}\right) = 2k$ then all of the generalized eigenvalues of the pencil $(A - \lambda C)$ are zero, and no useful decomposition of the problem can be obtained. However, the solution of (1.6) plays a key role in our algorithm for the solution of (1.4).

We propose the numerical solution of the minimization (1.4) through a preconditioned conjugate gradient (CG) algorithm [6]; the development of our algorithm is as follows. First, in section 2 we present an overview of Krylov subspace techniques as related to the numerical solution of the Lyapunov equation. In section 3 we give an overview of the minimization of (1.4) and present algorithms for its numerical solution in section 4. Following this, we present numerical examples in section 5. In section 6 we make some concluding remarks.

2. Krylov subspaces and iterative techniques. Krylov subspace techniques have gained increasing popularity in the solution of large, sparse systems of linear equations

$$Ax = b. \tag{2.1}$$

A Krylov subspace $\mathcal{K}(A, v, k)$ is defined as

$$\mathcal{K}(A, v, k) = \text{span}(\begin{bmatrix} v & Av & \cdots & A^{k-1}v \end{bmatrix}),$$

where $A \in \mathbb{R}^{n\times n}$, $v \in \mathbb{R}^n$, and k is an integer. One typically uses the Arnoldi algorithm [11] or a variation thereof to compute an orthogonal matrix $V_k \in \mathbb{R}^{n\times k}$, or simply V, such that $\text{span}(V) = \mathcal{K}(A, v, k)$. The Arnoldi algorithm generates a sequence of orthogonal matrices V_k such that $AV_k = V_{k+1}H_{k+1}$, where $H_{k+1} \in \mathbb{R}^{k+1\times k}$ is an upper Hessenberg matrix; i.e., $i > j + 1 \Rightarrow H_{ij} = 0$.

The GMRES algorithm [32] uses Krylov subspace bases V_k obtained by the Arnoldi algorithm to "project" the underlying problem (2.1) into a low-rank minimization

$$y^* = \arg\min_y \left\| H_{k+1}y - {V_{k+1}}'b \right\|_2$$

and approximates the solution x of (2.1) as $x \approx V_k y^*$. If the corresponding residual is too large, then the algorithm may either (1) increase the dimension k of the Krylov subspace or (2) use iterative refinement on the residual with a rank k Krylov subspace (GMRES(k)). Barring algorithm stagnation due to the identification of an A-invariant subspace (a consequence of catastrophic breakdown in the Arnoldi method) [4], the iterative application of GMRES(k) guarantees monotone decreasing residuals corresponding to each iteration.

Hu and Reichel [19] propose an iterative algorithm, based on GMRES [32], for the solution of large, sparse Sylvester equations

$$AX + XB + C = 0. \tag{2.2}$$

Each iteration of the Hu–Reichel algorithm uses Krylov subspaces of G and H to construct a minimization (1.4) with $\text{rank}\left(\begin{bmatrix} A & C \end{bmatrix}\right) = \text{rank}\left(\begin{bmatrix} B & D \end{bmatrix}\right) = k + 1$ whose solution X is obtained by a CG algorithm. A related approach is proposed by Jaimoukha and Kasenally [23].

Hodel and Poolla [17] and Hodel, Tenison, and Poolla [18] iteratively compute estimates of the dominant invariant subspace of the solution X of the Lyapunov equation (1.1). Similarly, Hodel [16] proposes gradient-based schemes that attempt to identify a low-rank subspace basis V that minimizes the associated residual of the Lyapunov equation. Since each of these algorithms identifies a subspace basis $V \in \mathbb{R}^{n\times k}$ and not a low-rank approximate solution $\hat{X} \in \mathbb{R}^{n\times n}$ of the Lyapunov equation (1.1), either of these algorithms may be used in tandem with the minimization of (1.4) to obtain a low-rank estimate $\hat{X} = V\Sigma V'$, where Σ is computed from (1.4). This approach does not necessarily yield estimates $\hat{X}$ that lie in a Krylov subspace.

Saad [31] obtains a low-rank approximate solution of the Lyapunov equation (1.1) by applying Krylov subspaces to the identity

$$X = \int_0^\infty e^{A't} Q e^{At} dt, \tag{2.3}$$

where A is stable (all eigenvalues lie in the left half plane); the evaluation of this integral is clearly undesirable when A is not stable. This algorithm computes an estimate $\hat{X} = V\Sigma V'$ of X by solving a reduced-order Lyapunov equation (1.2). This approach is applied in [22] to construct low-order models/controllers for very large, sparse linear dynamic systems. While error bounds are available for this approach, care must be taken in its application, especially when the matrix $(A + A')$ is not negative definite; see [17]. Further issues in the use of the integral (2.3) are discussed in [15].

3. Reduction of problem dimension. The minimization (1.4) can be rewritten as a standard least-squares problem (1.5) through a Kronecker product expansion. More precisely, in (1.5) we let $\bar{A} = (B \otimes A + D \otimes C)$, $\bar{x} = \mathrm{vec}(X)$, and $\bar{b} = \mathrm{vec}(F)$, where $Y \otimes Z = \left[\ y_{ij}Z\ \right]$ is the Kronecker product of two arbitrary matrices Y and Z, and $\mathrm{vec}(A)$ is the vector stack of the matrix A; e.g., if $Z \in \mathbb{R}^{n\times m}$, then

$$\mathrm{vec}(Z) = \left[\ Z_{\cdot 1}{}' \quad \cdots \quad Z_{\cdot m}{}'\ \right]',$$

where $Z_{\cdot j}$ is the jth column of the matrix Z. (Observe that $\mathrm{vec}(ABC) = (C' \otimes A)\,\mathrm{vec}(B)$ [3].) The Kronecker product expansion of (1.4) yields an overdetermined sparse system of n^2 equations in k^2 unknowns so that a naive application of a QR algorithm would require $O(n^2k^4 + k^6)$ flops to obtain the optimal solution X. If $k < n/3$, then the dimension of the minimization may be reduced, as shown in the following lemma.

LEMMA 3.1. *Let $A, B, C, D \in \mathbb{R}^{n\times k}, F \in \mathbb{R}^{n\times n}$, and let $A_1, A_2, B_1, C_1, D_1, D_2 \in \mathbb{R}^{k\times k}$ satisfy the QR factorizations*

$$\left[\ Q_1^{(1)} \quad Q_2^{(1)} \quad Q_3^{(1)}\ \right] \begin{bmatrix} C_1 & A_1 \\ 0 & A_2 \\ 0 & 0 \end{bmatrix} = \left[\ C \quad A\ \right],$$

$$\left[\ Q_1^{(2)} \quad Q_2^{(2)} \quad Q_3^{(2)}\ \right] \begin{bmatrix} B_1 & D_1 \\ 0 & D_2 \\ 0 & 0 \end{bmatrix} = \left[\ B \quad D\ \right],$$

where $Q_1^{(j)}, Q_2^{(j)} \in \mathbb{R}^{n\times k}$ and $Q_3^{(j)} \in \mathbb{R}^{n\times n-2k}$, and $[Q_1^{(j)} \quad Q_2^{(j)} \quad Q_3^{(j)}]$ is an orthogonal basis of $\mathbb{R}^n$, $j = 1, 2$. Then $X \in \mathbb{R}^{k\times k}$ minimizes $\|AXB' + CXD' + F\|_F$ if and only if X minimizes

$$\left\| \begin{bmatrix} A_1XB_1{}' + C_1XD_1{}' \\ A_2XB_1{}' \\ C_1XD_2{}' \end{bmatrix} + \begin{bmatrix} \hat{F}_{11} \\ \hat{F}_{21} \\ \hat{F}_{12} \end{bmatrix} \right\|_F, \tag{3.1}$$

where $\hat{F}_{ij} = Q_i^{(1)\prime} F Q_j^{(2)}$.

Proof. Let $Q_j = [Q_1^{(j)} \quad Q_2^{(j)} \quad Q_3^{(j)}]$, $j = 1, 2$, so that

$$\left[\ C \quad A\ \right] = Q_1 \begin{bmatrix} C_1 & A_1 \\ 0 & A_2 \\ 0 & 0 \end{bmatrix}, \qquad \left[\ B \quad D\ \right] = Q_2 \begin{bmatrix} B_1 & D_1 \\ 0 & D_2 \\ 0 & 0 \end{bmatrix}$$

and define

$$\hat{F} = Q_1{}' F Q_2 = \begin{bmatrix} \hat{F}_{11} & \hat{F}_{12} & \hat{F}_{13} \\ \hat{F}_{21} & \hat{F}_{22} & \hat{F}_{23} \\ \hat{F}_{31} & \hat{F}_{32} & \hat{F}_{33} \end{bmatrix}.$$

Then

$$\begin{aligned} &\min \|AXB' + CXD' + F\|_F \\ &\quad = \min \left\| Q_1{}' \left(AXB' + CXD' + F\right) Q_2 \right\|_F \end{aligned}$$

$$= \min \left\| \begin{bmatrix} A_1 \\ A_2 \\ 0 \end{bmatrix} X \begin{bmatrix} B_1{}' & 0 & 0 \end{bmatrix} + \begin{bmatrix} C_1 \\ 0 \\ 0 \end{bmatrix} X \begin{bmatrix} D_1{}' & D_2{}' & 0 \end{bmatrix} + \hat{F} \right\|_F$$

$$= \min \left\| \begin{bmatrix} A_1 X B_1{}' + C_1 X D 1' + \hat{F}_{11} & C_1 X D_2{}' + \hat{F}_{12} & \hat{F}_{13} \\ A_2 X B_1{}' + \hat{F}_{21} & \hat{F}_{22} & \hat{F}_{23} \\ \hat{F}_{31} & \hat{F}_{32} & \hat{F}_{33} \end{bmatrix} \right\|_F .$$

Since $\hat{F}_{13}, \hat{F}_{22}, \hat{F}_{23}, \hat{F}_{31}, \hat{F}_{32}$, and $\hat{F}_{33}$ are constant for all values of X, the above minimization is unaffected by these terms, and the lemma follows. ∎

Remark 3.1. The practical reduction of (1.4) to (3.1) for the general case (F does not possess a sparse or other exploitable structure) can be accomplished in $O(n^2k)$ flops as follows.

1. Compute and store the Householder vectors $h_i^{(j)}$, $i = 1, 2k$, $j = 1, 2$, obtained in the QR factorizations of $\begin{bmatrix} C & A \end{bmatrix}$ and $\begin{bmatrix} D & B \end{bmatrix}$ in $O(nk^2)$ flops.
2. Accumulate Householder reflections $H_i^{(j)} = (I - (2/h_i^{(j)\prime} h_i^{(j)}) h_i^{(j)} h_i^{(j)\prime})$ to obtain
$$\bar{Q}_j = \begin{bmatrix} Q_1^{(j)} & Q_2^{(j)} \end{bmatrix},$$
$i = 1, \ldots, 2k$, $j = 1, 2$, in $O(nk^2)$ flops.
3. Compute $Z = F\bar{Q}_2 \in \mathbb{R}^{n \times 2k}$ in $O(n^2k)$ flops.
4. Compute
$$\begin{bmatrix} \hat{F}_{11} \\ \hat{F}_{21} \end{bmatrix} = \bar{Q}_1' Z \text{ and } \hat{F}_{12} = Q_1^{(1)\prime} Z \begin{bmatrix} 0_{k \times k} \\ I_k \\ 0 \end{bmatrix}$$
(i.e., multiply by the last k columns of Z) in $O(nk^2)$ flops.

Observe that the dominant computational cost of $O(n^2k)$ flops occurs in step 3. This cost can be greatly reduced if matrix-vector products Fv can be computed in much less than n^2 flops, e.g., if F is sparse or low-rank. The latter will be the case in controller/model reduction applications such as [28]. Since the number of inputs/outputs is greatly exceeded by the number of states in a typical dynamic system, the matrix F is given by $\hat{B}\hat{B}'$, $B \in \mathbb{R}^{n \times m}$, $m \ll n$. Then step 3 above can be computed in $O(mnk)$ time, rendering the overall complexity to $O(nmk)$ or $O(nk^2)$ (whichever is smaller).

Remark 3.2. If the least-squares minimization (1.4) is obtained via Krylov subspaces as in [19], then the corresponding minimization has $A, B, C, D \in \mathbb{R}^{(k+1) \times k}$, which obviates the need for the above reduction.

4. Iterative solution by CG methods. We shall henceforth assume that the minimization (1.4) has been posed in the form of Lemma 3.1; i.e., we wish to solve the least-squares problem

$$\min \left\| \begin{bmatrix} A_1 X B_1{}' + C_1 X D_1{}' \\ A_2 X B_1{}' \\ C_1 X D_2{}' \end{bmatrix} + \begin{bmatrix} \hat{F}_{11} \\ \hat{F}_{21} \\ \hat{F}_{12} \end{bmatrix} \right\|_F . \tag{4.1}$$

As in the case of (1.4), (4.1) can be solved by a Kronecker product expansion (1.5) with

$$\bar{A} = \begin{bmatrix} L_1 \\ L_2 \\ L_3 \end{bmatrix} \triangleq \begin{bmatrix} B_1 \otimes A_1 + D_1 \otimes C_1 \\ B_1 \otimes A_2 \\ D_2 \otimes C_1 \end{bmatrix}, \tag{4.2}$$

$$\bar{x} = \mathrm{vec}(X), \text{ and } \bar{b} = \begin{bmatrix} \bar{b}_1 \\ \bar{b}_2 \\ \bar{b}_3 \end{bmatrix} \triangleq \begin{bmatrix} \mathrm{vec}(\hat{F}_{11}) \\ \mathrm{vec}(\hat{F}_{21}) \\ \mathrm{vec}(\hat{F}_{12}) \end{bmatrix}.$$

The QR method allows the computation of a matrix X that minimizes (4.1) in $O(k^6)$ flops.

We may also minimize (4.1) in $O(k^5)$ flops by applying the CG algorithm [14] to the normal equations $\bar{A}'\bar{A}\bar{x} = -\bar{A}'\bar{b}$ as follows.

ALGORITHM 1. Solution of (4.1) by CGs.

Inputs $A_1, A_2, B_1, C_1, D_1, D_2, \hat{F}_{11}, \hat{F}_{21}, \hat{F}_{12} \in \mathbb{R}^{k\times k}$.

Outputs $X \in \mathbb{R}^{k\times k}$ satisfying the minimization (4.1).

1. $X_0 = 0$, $j = 0$, $R_{11}^{(0)} = \hat{F}_{11}$, $R_{21}^{(0)} = \hat{F}_{21}$, $R_{12}^{(0)} = \hat{F}_{12}$,

$$R_0 = A_1'\hat{F}_{11}B_1 + C_1'\hat{F}_{11}D_1 + A_2'\hat{F}_{21}B_1 + C_1'\hat{F}_{12}D_2$$

2. while $R_j \neq 0$
   (a) $j = j + 1$
   (b) if $j = 1$
   $P_j = R_0$
   (c) else $\beta_j = \dfrac{\mathrm{vec}(R_{j-1})'\mathrm{vec}(R_{j-1})}{\mathrm{vec}(R_{j-2})'\mathrm{vec}(R_{j-2})}$, $P_j = R_{j-1} + \beta_j P_{j-1}$
   (d) end if
   (e) Compute $W_{11}^{(j)} = A_1P_jB_1' + C_1P_jD_1'$, $W_{21}^{(j)} = A_2P_jB_1'$, and $W_{12}^{(j)} = C_1P_jD_2'$, and let $W_j = [\ \mathrm{vec}(W_{11}^{(j)})'\ \ \mathrm{vec}(W_{21}^{(j)})'\ \ \mathrm{vec}(W_{12}^{(j)})'\ ]'$.
   (f) $\alpha_j = \dfrac{\mathrm{vec}(R_{j-1})'\mathrm{vec}(R_{j-1})}{\mathrm{vec}(P_j)'\bar{A}p_j}$
   (g) $X_j = X_{j-1} + \alpha_j P_j$
   (h) Compute residuals

$$R_{11}^{(j)} = A_1X_{j-1}B_1' + C_1X_{j-1}D_1' + \hat{F}_{11} \qquad R_{21}^{(j)} = A_2X_{j-1}B_1' + \hat{F}_{21}$$
$$R_{12}^{(j)} = C_1X_{j-1}D_2' + \hat{F}_{12}$$
$$R_j = A_1'R_{11}^{(j)}B_1 + C_1'R_{11}^{(j)}D_1 + A_2'R_{21}^{(j)}B_1 + C_1'R_{12}^{(j)}D_2$$

3. end while
4. $X = X_j$

In exact arithmetic, the CG algorithm will converge in at most k^2 iterations; if $\bar{A}'\bar{A}$ is a rank l modification to the identity matrix, then the CG algorithm will converge in at most l iterations; see [11] and [14] for details. The chief disadvantage of the CG method is the loss of $(\bar{A}'\bar{A})$ orthogonality between the vectors p_j as j increases; that is, computed vectors p_j do not satisfy the relation $p_j'\bar{A}'\bar{A}\begin{bmatrix} p_1 & \cdots & p_{j-1} \end{bmatrix} = 0$. Because of this numerical behavior, the CG algorithm has come to be regarded as a purely iterative method for large, sparse linear systems of equations. However, if rank $\left(L_2'L_2 + L_3'L_3\right)$ is not too large, as in [19], then convergence can be accelerated by using a preconditioned conjugate gradient (PCG) algorithm [6]. The PCG algorithm is based on the use of a splitting $(\bar{A}'\bar{A}) = M + N$, where M is symmetric, positive definite, and easy to invert and "near" $\bar{A}$. The PCG algorithm is as follows.

ALGORITHM 2. Generalized CGs.

Inputs $M, N \in \mathbb{R}^{n\times n}$, both symmetric, M positive definite and (by assumption) $\bar{A} = M + N$ positive definite, and $b \in \mathbb{R}^n$.

Outputs $x \in \mathbb{R}^n$ satisfying $\bar{A}x + b = 0$.

1. $x_{-1} = x_0 = 0$, $j = 0$, $r_0 = b$
2. while $r_j \neq 0$
   (a) $j = j + 1$; compute $z_j := -M^{-1} r_{j-1}$.
   (b) $\gamma_j = \dfrac{{z_j}' M z_j}{{z_j}' \bar{A} z_j}$
   (c) $\omega_j = 1$ if $j = 1$, else $\omega_j = \left(1 - \dfrac{\gamma_j ({z_j}' M z_j)}{\omega_{j-1}\gamma_{j-1}({z_{j-1}}' M z_{j-1})}\right)^{-1}$
   (d) $x_j = x_{j-2} + \omega_j \left(\gamma_j z_j + x_{j-1} - x_{j-2}\right)$.
   (e) $r_j = \bar{A}x_j + \bar{b}$.
3. end while
4. $x = x_j$

We apply the PCG algorithm to the minimization (4.1) as follows. The normal equations corresponding to (4.1) are

$$(L'L)\, x = -L'\bar{f}.$$

Observe that $L'L = {L_1}'L_1 + {L_2}'L_2 + {L_3}'L_3$, and so this problem decomposes naturally to the splitting $M = {L_1}'L_1$, $N = \left({L_2}'L_2 + {L_3}'L_3\right)$. In order to solve $Mz_j = -r_j$, observe that if L_1 is nonsingular then z_j satisfies

$$\begin{aligned} 0 &= {L_1}'L_1 z_j + {L_1}' r_1^{(j-1)} + {L_2}' r_2^{(j-1)} + {L_3}' r_3^{(j-1)} \\ &= L_1 z_j + r_1^{(j-1)} + {L_1}^{-T}\left({L_2}' r_2^{(j-1)} + {L_3}' r_3^{(j-1)}\right), \end{aligned}$$

which may be solved in matrix form in $O(k^3)$ flops as

$$\begin{aligned} 0 &= {A_1}' T_j B_1 + {C_1}' T_j D_1 - \left({A_2}' R_{21}^{(j-1)} B_1 + {C_1}' R_{12}^{(j-1)} D_2\right), \\ 0 &= A_1 Z_j {B_1}' + C_1 Z_j {D_1}' + (R_{11}^{(j-1)} + T_j). \end{aligned}$$

The resulting matrix-valued PCG algorithm is shown below.

ALGORITHM 3. Solution of (4.1) by PCGs.

Inputs $A_1, A_2, B_1, C_1, D_1, D_2, \hat{F}_{11}, \hat{F}_{21}, \hat{F}_{12} \in \mathbb{R}^{k\times k}$.

Outputs $X \in \mathbb{R}^{k\times k}$ satisfying the minimization (4.1).

1. $X_{-1} = X_0 = 0$, $j = 0$, $R_{11}^{(0)} = \hat{F}_{11}$, $R_{21}^{(0)} = \hat{F}_{21}$, $R_{12}^{(0)} = \hat{F}_{12}$,

$$R_0 = {A_1}' \hat{F}_{11} B_1 + {C_1}' \hat{F}_{11} D_1 + {A_2}' \hat{F}_{21} B_1 + {C_1}' \hat{F}_{12} D_2$$

2. while $R_j \neq 0$
   (a) $j = j + 1$; Solve for T_j and Z_j:

$$\begin{aligned} 0 &= {A_1}' T_j B_1 + {C_1}' T_j D_1 - \left({A_2}' R_{21}^{(j-1)} B_1 + {C_1}' R_{12}^{(j-1)} D_2\right) \\ 0 &= A_1 Z_j {B_1}' + C_1 Z_j {D_1}' + (R_{11}^{(j-1)} + T_j) \end{aligned}$$

   (b) Compute $W_{11}^{(j)} = A_1 Z_j {B_1}' + C_1 Z_j {D_1}'$, $W_{21}^{(j)} = A_2 Z_j {B_1}'$, and $W_{12}^{(j)} = C_1 Z_j {D_2}'$, and let $W_j = \left[\ \mathrm{vec}(W_{11}^{(j)})'\ \ \mathrm{vec}(W_{21}^{(j)})'\ \ \mathrm{vec}(W_{12}^{(j)})'\ \right]'$.
   (c) $\gamma_j = \dfrac{-\mathrm{vec}(Z_j)'\mathrm{vec}(R_{j-1})}{\mathrm{vec}(W_j)'\mathrm{vec}(W_j)}$

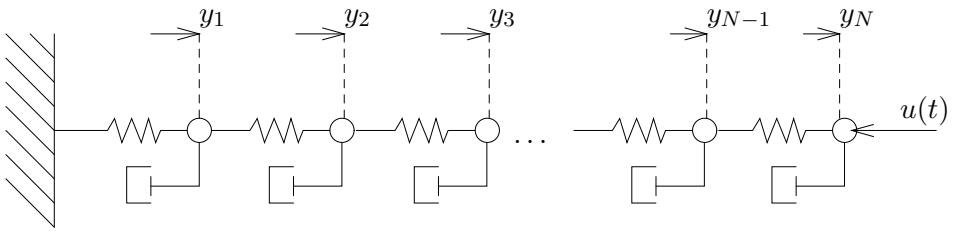

FIG. 5.1. *Mass-spring-dashpot system.*

(d) $\omega_j = 1$ if $j = 1$, else $\omega_j = \left(1 - \dfrac{\gamma_j (z_j{}' r_{j-1})}{\omega_{j-1}\gamma_{j-1}(z_{j-1}{}' r_{j-2})}\right)^{-1}$

(e) $X_j = X_{j-2} + \omega_j\,(\gamma_j Z_j + X_{j-1} - X_{j-2})$.

(f) Compute residuals

$$R_{11}^{(j)} = A_1 X_{j-1} B_1{}' + C_1 X_{j-1} D_1{}' + \hat{F}_{11} \qquad R_{21}^{(j)} = A_2 X_{j-1} B_1{}' + \hat{F}_{21}$$
$$R_{12}^{(j)} = C_1 X_{j-1} D_2{}' + \hat{F}_{12}$$
$$R_j = A_1{}' R_{11}^{(j)} B_1 + C_1{}' R_{11}^{(j)} D_1 + A_2{}' R_{21}^{(j)} B_1 + C_1{}' R_{12}^{(j)} D_2$$

3. end while
4. $X = X_j$

Remark 4.1. If rank $\left(\begin{bmatrix} L_2 & L_3 \end{bmatrix}\right)$ is small ($\ll k^2$), then in exact arithmetic this algorithm should converge much faster than CG (Algorithm 1); see [6, pp. 319–320]. For example, minimizations (4.1) that arise in [19] can be solved in only $2k$ iterations, or $O(k^4)$ work. However, it should be pointed out that the PCG algorithm is not necessarily numerically superior to the CG algorithm; in particular, the operator M is explicitly inverted in step 2a of Algorithm 1; this is undesirable when L_1 is poorly conditioned.

Remark 4.2. Unfortunately, it is not immediately obvious how the conditioning of L_1 relates to the original matrices A, B, C, D. Hence, an explicit bound on the conditioning of $L_1{}' L_1$ appears impossible to determine. However, one may use Byers's condition estimator [5] to determine when an ill-conditioned system occurs; a variant of this algorithm may be employed with the preconditioner in the present algorithm. The response to an ill-conditioned estimator depends on the scenario in which it occurs; one may simply increase the dimension of V (as in Krylov subspace-based algorithms) or one may dispense with the preconditioner to use either the CG algorithm or (if applicable) the algorithms presented in [19] or [23]. When well conditioned, the PCG algorithm in this paper provides an improvement in algorithm speed.

5. Numerical examples. Algorithms 1 and 3 were tested on a lumped mass-spring-damper model of a vibrating system (see Figure 5.1); such models arise in numerous engineering applications. In Figure 5.1, y_j denotes the displacement of mass j from its rest position; $u(t)$ is an external (controlled) force; and all N masses, springs, and dashpots are assumed to be identical with mass m, stiffness ρ, and damping δ, respectively. The first-order dynamic model of the system is

$$\dot{x} = Ax + Bu,$$

TABLE 5.1
Flop counts vs. k for implementations of Algorithms 1 *and* 3.

k	Algorithm 1	Algorithm 3
5	6635	3.918E+04
10	5.043E+04	3.232E+05
15	1.674E+05	1.172E+06

where $A = \left[\begin{smallmatrix} 0 & I_n \\ A_{21} & -(\delta/m)I_n \end{smallmatrix}\right] \in \mathbb{R}^{2N\times 2N}$,

$$A_{21} = \frac{\rho}{m}\begin{bmatrix} -2 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & 1 & -2 & 1 \\ 0 & \cdots & 0 & 1 & -1 \end{bmatrix}, \text{ and } B = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}.$$

Example systems were run with $N = 100$, 200, 300, and 400. Results presented in this section are for $N = 300$ with parameters (ρ, δ, m) selected as either $(1, 0.1, 1)$ or $(10, 10^{-3}, 10^{-2})$, respectively. The second set of parameters yields a very lightly damped system.

A solution X was sought to the minimization

$$\min_{X\in\mathbb{R}^{k\times k}} \|(AV)XV' + VX(A'V') + BB'\|_F,$$

where V was an orthogonal basis of the Krylov subspace

$$\text{span}\left(\begin{bmatrix} b & Ab & \cdots & A^{k-1}b \end{bmatrix}\right)$$

for $k = 5, 10$, and 15. Numerical implementation of Algorithms 1 and 3 was done using MATLAB version 4.2a on a Sun Sparc-10. Table 5.1 shows returned flop counts per iteration for the two algorithms vs. problem dimension parameter k. Figure 5.2 shows the plots of the residual of the normal equations for the CG and PCG iterations; system parameters were $\delta = 0.1$, $\rho = 1$, and $m = 1$. Observe that the PCG method residual reaches its equilibrium value in roughly k iterations, consistent with its expected convergence behavior. Both algorithms are sensitive to the condition of the underlying system; Figure 5.3 shows the residuals for $\delta = 10^{-3}$, $\rho = 10$, and $m = 0.01$. The deterioration in performance is due to the wide spread in singular values of $L'L$ associated with lightly damped, high-frequency modes of the system (see equation (4.2)).

6. Conclusions. The numerical solution of overdetermined Sylvester equations (1.4) has applications in both the reduced-order modeling and the control of large dimensional systems as well as low-rank approximate solution of Lyapunov equations (1.1) and Sylvester equations (2.2). Our solution procedure involves the reduction of the original problem to a minimization of dimension at most $3k \times k$, followed by either a CG algorithm for the general case, or a PCG algorithm for minimizations (1.4) that are low-rank perturbations of a reduced-order general Sylvester equation (1.6). A CG algorithm requires $O(k^5)$ flops before convergence, while a PCG algorithm may require as few as $O(k^4)$ flops before convergence.

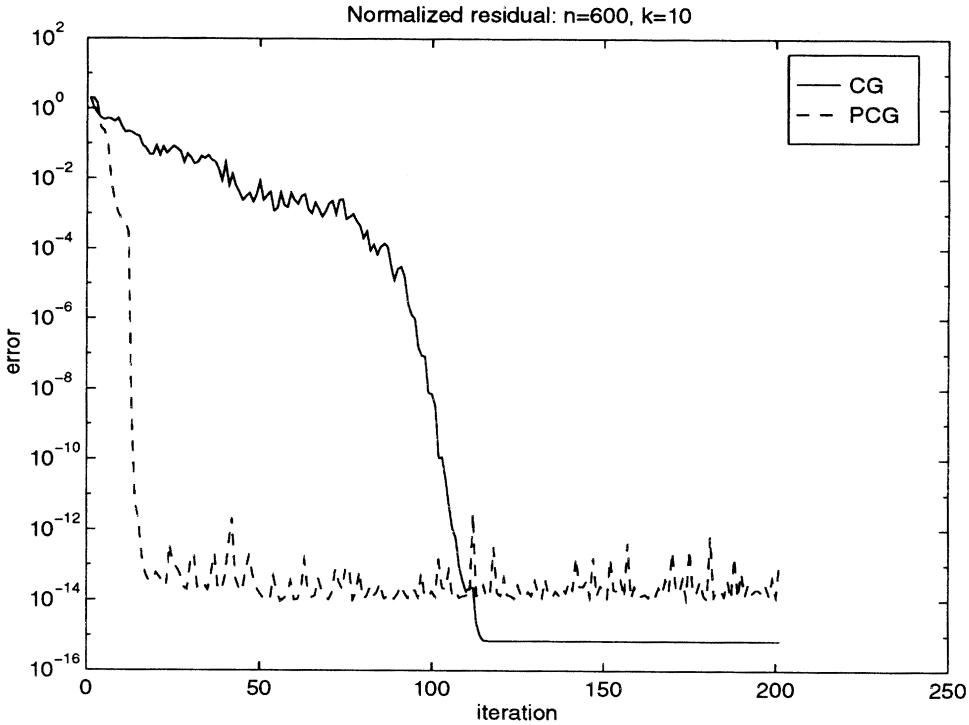

FIG. 5.2. *Residual plot:* $\delta = 0.1$, $\rho = 1$, $m = 1$.

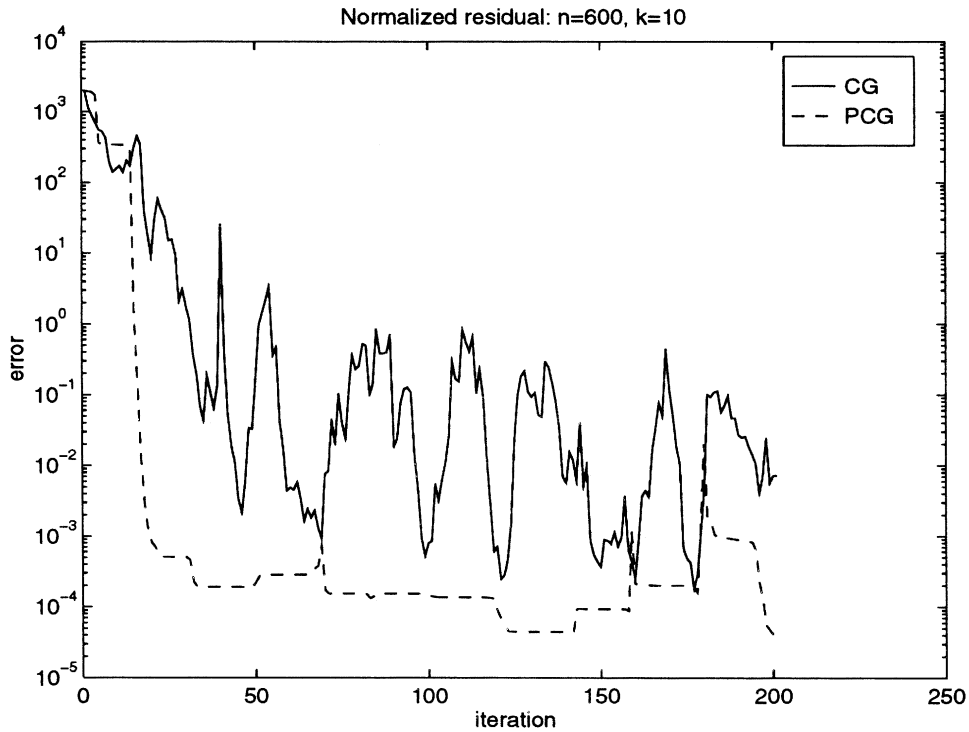

FIG. 5.3. *Residual plot:* $\delta = 10^{-3}$, $\rho = 10$, $m = 0.01$.

Acknowledgment. We wish to thank Gene Golub for suggesting the overdetermined Sylvester equation problem.

REFERENCES

[1] R. H. Bartels and G. W. Stewart, *Solution of the matrix equation $AX + XB = C$*, Comm. of the ACM, 15 (1972), pp. 820–826.
[2] D. S. Bernstein and D. C. Hyland, *The optimal projection equations for reduced-order state estimation*, IEEE Trans. Automat. Control, AC-30 (1985), pp. 583–585.
[3] J. W. Brewer, *Kronecker products and matrix calculus in system theory*, IEEE Trans. Circuits and Systems, CAS-25 (1978), pp. 772–781.
[4] P. N. Brown, *A theoretical comparison of the Arnoldi and GMRES algorithms*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 58–78.
[5] R. Byers, *A LINPACK-style condition estimator for the equation $AX - XB = C$*, IEEE Trans. Automat. Control, AC-29 (1984), pp. 926–928.
[6] P. Concus, G. H. Golub, and D. P. O'Leary, *A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations*, in Sparse Matrix Computations, J. R. Bunch and D. J. Rose, eds., Academic Press, New York, 1976, pp. 309–332.
[7] J. D. Gardiner, A. J. Laub, J. J. Amato, and C. B. Moler, *Solution of the Sylvester matrix equation $AXB^T + CXD^T = E$*, ACM Trans. Math. Software, 18 (1992), pp. 223–231.
[8] J. D. Gardiner, M. R. Wette, A. J. Laub, J. J. Amato, and C. B. Moler, *Algorithm 705: A Fortran-77 software package for solving the Sylvester matrix equation $AXB^T + CXD^T = E$*, ACM Trans. Math. Software, 18 (1992), pp. 232–238.
[9] J. A. George, M. T. Heath, and R. J. Plemmons, *Solution of large-scale sparse least squares problems using auxiliary storage*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 416–429.
[10] G. H. Golub, S. Nash, and C. Van Loan, *A Hessenberg-Schur method for the problem $AX + XB = C$*, IEEE Trans. Automat. Control, AC-24 (1979), pp. 909–913.
[11] G. H. Golub and C. Van Loan, *Matrix Computations*, 2nd ed., Johns Hopkins University Press, Baltimore, MD, 1989.
[12] S. J. Hammarling, *Numerical solution of the stable, non-negative definite Lyapunov equation*, IMA J. Numer. Anal., 2 (1982), pp. 303–323.
[13] M. T. Heath, *Numerical methods for large sparse linear least squares problems*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 497–513.
[14] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. National Bureau of Standards, 49 (1952), pp. 409–436.
[15] A. S. Hodel, *Numerical Methods for the Solution of Large and Very Large, Sparse Lyapunov Equations*, Ph.D. thesis, Department of Electrical Engineering, University of Illinois at Urbana-Champaign, Champaign, IL, 1989.
[16] A. S. Hodel, *Least squares approximate solution of the Lyapunov equation*, in Proceedings of the 30th IEEE Conference on Decision and Control, Brighton, England, 1991, pp. 1619–1624.
[17] A. S. Hodel and K. Poolla, *Heuristic methods to the solution of very large, sparse Lyapunov and algebraic Riccati equations*, in Proceedings of the 27th IEEE Conference Decision and Control, Austin, TX, 1988, pp. 2217–2222.
[18] A. S. Hodel, R. B. Tenison, and K. Poolla, *Numerical solution of large Lyapunov equations by Approximate Power Iteration*, Linear Algebra Appl., 236 (1996), pp. 205–230.
[19] D. Y. Hu and L. Reichel, *Krylov subspace methods for the Sylvester equation*, Linear Algebra Appl., 172 (1992), pp. 283–313.
[20] D. C. Hyland and D. S. Bernstein, *The optimal projection equations for fixed-order dynamic compensation*, IEEE Trans. Automat. Control, AC-29 (1984), pp. 1034–1037.
[21] M. Ilic, *New approaches to voltage monitoring and control*, IEEE Control Systems Magazine, 9 (1989), pp. 3–11.
[22] I. M. Jaimoukha and E. M. Kasenally, *Oblique projection methods for large scale model reduction*, SIAM J. Matrix. Anal. Appl., 16 (1995), pp. 602–627.
[23] I. M. Jaimoukha and E. M. Kasenally, *Krylov subspace methods for solving large Lyapunov equations*, SIAM J. Numer. Anal., 31 (1994), pp. 227–251.
[24] P. Lancaster, *Explicit solutions of linear matrix equations*, SIAM Review, 12 (1970), pp. 544–566.
[25] J. LaSalle and S. Lefschetz, *Stability of Liapunov's Direct Method*, Academic Press, New York, 1961.
[26] A. Lu, *Alternating Direction Implicit Iteration Solution of Lyapunov Equations*, Master's the-

sis, Department of Mathematics, University of Tennessee, Knoxville, TN, 1990.

[27] C. B. Moler and G. W. Stewart, *An algorithm for generalized matrix eigenvalue problems*, SIAM J. Numer. Anal., 10 (1973), pp. 241–256.

[28] B. C. Moore, *Principal component analysis in linear systems: Controllability, observability and model reduction*, IEEE Trans. Automat. Control, AC-26 (1981), pp. 17–32.

[29] S. Richter and E. G. Collins Jr., *A homotopy algorithm for reduced order compensator design using the optimal projection equations*, in Proceedings of the 28th IEEE Conference on Decision and Control, Tampa, FL, 1989, pp. 506–511.

[30] S. Richter and A. S. Hodel, *Homotopy methods for the solution of general modified algebraic Riccati equations*, in Proceedings of the 29th IEEE Conference on Decision and Control, Honolulu, HI, 1990, pp. 971–976.

[31] Y. Saad, *Numerical solution of large Lyapunov equations*, in Signal Processing, Scattering and Operator Theory, and Numerical Methods, Progress in Systems and Control Theory Series, Vol. 5, M. A. Kaashoek, J. H. van Schuppen, and A. C. M. Ran, eds., Birkhauser, Boston, Cambridge, MA, pp. 401–410.

[32] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[33] R. J. Veillette, *Reliable Control of Decentralized Systems: an ARE-based H-infinity Approach*, Ph.D. thesis, Department of Electrical Engineering, University of Illinois at Urbana-Champaign, Champaign, IL, 1989.

[34] E. L. Wachspress, *Iterative solution of the Lyapunov matrix equation*, Appl. Math. Lett., 1 (1989), pp. 87–90.