International Mathematical Forum, Vol. 6, 2011, no. 66, 3293 - 3304

A Compact Algorithm for Solving Generalized Lyapunov Matrix Equations

Amer Kaabi

Department of Mathematics
Abadan Branch, Islamic Azad University, Abadan, Iran
kaabi.amer@yahoo.com

Abstract

This paper considers the solution of large-scale generalized continuous-time Lyapunov matrix equations of the form $EXA^T + AXE^T - GG^T = 0$. Such equations arise in stability theory, optimal control problems and balanced model reduction. Numerical examples are presented.

Keywords: projected Lyapunov matrix equations, Kronecker product, Arnoldi process, conjugate gradient method

1 Introduction

Consider the generalized continuous-time algebraic Lyapunov equation (GCALE)

$$EXA^T + AXE^T - GG^T = 0 \tag{1}$$

with given matrices $E, A \in R^{n\times n}$,$G \in R^{n\times p}$ and unknown matrix X. Such equations arise in stability theory [5], optimal control problems [12] and balanced model reduction [16]. It has been shown in [19] that Eq. (1) have a unique Hermitian, positive definite solution X for every Hermitian positive definite matrix G if and only if all eigenvalues of the pencil $\lambda E - A$ are finite and lie in the open left half-plane. The classical numerical methods for the standard Lyapunov equations ($E = I$) are the Bartels-Stewart method[1], the Hammarling method [9] and the Hessenberg-Schur method [8]. An extension of these methods for the generalized Lyapunov equations with the nonsingular matrix E is given in [8, 19]. These methods are based on the preliminary reduction of the matrix (matrix pencil) to the (generalized) Schur form [7] or the Hessenberg-Schur form [8], calculation of the solution of a reduced system and back transformation. Since these methods cost $O(n^3)$ operations and

require $O(n^2)$ memory location, they are restricted to the problems of small or medium size. An alternative approach to solve the (generalized) Lyapunov equations is the sign function method[3, 11, 13]. Comparison of the sign function method to the Bartels-Stewart and Harmmarling methods with respect to accuracy and computational cost can be found in [3].The numerical solution of the generalized Lyapunov equations with singular matrix E is more complicated. Such equations may not have solutions even if all finite eigenvalues of the pencil $\lambda E - A$ lie in the open left half-plane. Moreover, even if a solution exists, it is, in general, not unique. In [22], T. Stykel, generalized a Bartels-Stewart method and a Harmmarling method to compute a partial solution of generalized Lyapunov equation with special right hand side. Also he introduced a spectral condition number and perturbation bounds for such an equation. Projected Lyapunov matrix equations

$$EXA^T + AXE^T + P_r^T GG^T P_r = 0, \quad X = P_l^T X P_l, \tag{2}$$

where P_l and P_r are the spectral projectors onto the left and right deflating subspaces corresponding to the finite eigenvalues of the pencil $\lambda E - A$, can be solved by the iterative methods based on the ADI and Smith techniques [17, 19]. These methods are especially efficient for large sparse Lyapunov equations with low rank right-hand side. In [21], T. Stykel have been presented a modified version of matrix sign function method for obtaining approximate solutions of projected Lyapunov equations (2) with large dense matrix coefficients. This method was first proposed in [4, 11, 12] for standard Lyapunov equations and then extended to generalized Lyapunov equations with nonsingular E in [2, 3, 6, 15]. In this paper we suppose that A and E are symmetric positive matrices. Applying classical algorithm CG to the Kronecker product form of Eq.(1) using $O(n^2)$ memory and arithmetic operation. The main goal of this paper is to reduce these requirements to $O(n)$. This procedure requires a compact representation of certain vectors in $\in R^{n^2}$. We show that it is possible to run this classical algorithm using $O(n)$ rather than $O(n^2)$ memory and time.

This paper is organized as follows. In Section 2, a brief description of Arnoldi algorithms is given, and compact form of conjugate gradient method to generalized Lyapunov matrix equations is presented in Section 3. In Section 4 some numerical examples are tested. Finally, Section 5 summarizes the main conclusion of this paper.

2 Arnoldi algorithms

Consider the Krylov subspace

$$K_j(\tilde{A}, \tilde{g}) = span\{\tilde{g}, \tilde{A}\tilde{g}, ..., \tilde{A}^{j-1}\tilde{g}\} \subseteq R^{n^2}$$

where $\tilde{A} = (A \otimes E + E \otimes A)$ and $\tilde{g} = vec(GG^T)$. It is clear that

$$K_j(\tilde{A}, \tilde{g}) \subseteq K_{j+1}(\tilde{A}, \tilde{g}),$$

and there exist $\tilde{m}$ such that

$$K_j(\tilde{A}, \tilde{g}) = K_{\tilde{m}}(\tilde{A}, \tilde{g})$$

for all $j \geq \tilde{m}$. The smallest $\tilde{m}$ such that

$$K_j(\tilde{A}, \tilde{g}) = K_{\tilde{m}}(\tilde{A}, \tilde{g}), \quad j \geq \tilde{m}$$

is called the grade of $\tilde{g}$ with respect to $\tilde{A}$. Arnoldi method can be used for the construction of an orthonormal basis $u_1, \ldots, u_m$ for $K_m(\tilde{A}, \tilde{g})$ where $\tilde{A} \in \mathbb{R}^{n^2 \times n^2}$ and $\tilde{g} \in \mathbb{R}^{n^2}$. The modified version of Arnoldi method can be described as follows [20]:

Algorithm 1. Arnoldi modified Gram-Schmidt

1. Choose integer number m and vector $\tilde{g} \in \mathbb{R}^{n^2}$ and compute $u_1 = \tilde{g}/\|\tilde{g}\|_2$.
2. For $j = 1, 2, ..., m$, Do
3. Compute $w_j = \tilde{A}u_j$
4. For $i = 1, ..., j$, Do
5. $h_{ij} = u_i^T w_j$
6. $w_j = w_j - h_{ij}u_i$
7. EndDo
8. $h_{j+1,j} = \| w_j \|_2$. If $h_{j+1,j} = 0$ Stop
9. $u_{j+1} = u_j/h_{j+1,j}$
10. EndDo

The nondefined h_{ij} are assumed to be zero. With $n^2 \times m$ matrix $U_m = [u_1, u_2, \ldots, u_m]$, we know $H_m = U_m^T \tilde{A} U_m$ is an upper $m \times m$ Hessenberg matrix whose entries are the scalers h_{ij}. From Algorithm 1 we deduce the following relation

$$\tilde{A}U_m = U_{m+1}\hat{H}_m, \tag{3}$$

where the $(m + 1) \times m$ matrix $\hat{H}_m$ is the same as H_m except for an additional row whose only nonzero element is $h_{m+1,m}$ in the $(m + 1, m)$ position [20]. As is well-known in the particular case where $\tilde{A}$ is symmetric, Arnoldi algorithm simplifies to the Lanczos process in which case H_m becomes symmetric tridiagonal. In the other hand, the block Arnoldi algorithm constructs an orthonormal basis $V_1, V_2, ..., V_m$, of the block Krylov subspace $\mathcal{K}_m(A, V_1)$. The algorithm is described as follows:

Algorithm 2. The block Arnoldi algorithm

1. Choose a unitary $n \times p$ matrix V_1.
2. For $j = 0, 1, ..., m$, Do
3. Compute $H_{i,j} = V_i^T A V_j, \qquad i = 1, 2, ..., j$
4. Compute $W_j = AV_j - \sum_{i=1}^{j} V_i H_{i,j}$
5. Compute the QR decomposition:$W_j = V_{j+1} H_{j+1,j}$ $(Q_j = V_{j+1}; R_j = H_{j+1,j})$
6. EndDo

The blocks $V_1, V_2, ..., V_m$ constructed by the algorithm have their columns mutually orthogonal provided that the upper triangular matrices $H_{j+1,j}$ are of maximum rank and this will be assumed in this paper. If $H_{j+1,j} = 0_{p\times p}$ then $\mathcal{K}_j(A, V_1)$ is invariant under A.

Let $\mathcal{H}_k$ denote the block upper Hessenberg matrix whose nonzero blocks $H_{i,j}$ are obtained by block Arnoldi algorithm and $\mathcal{V}_k = [V_1, V_2, ..., V_k]$. Also, $\hat{\mathcal{H}}_k$, is the same as $\mathcal{H}_k$ except for an additional row block whose only nonzero element is $H_{k+1,k}$ in the $(k+1, k)$ position. Thus, from the block Arnoldi algorithm we deduce the following relations

$$A\mathcal{V}_k = \mathcal{V}_k \mathcal{H}_k + V_{k+1} H_{k+1,k} E_k^T, \quad k < m,$$

$$\mathcal{H}_m = \mathcal{V}_m^T A \mathcal{V}_m, \qquad \mathcal{V}_m^T \mathcal{V}_m = I_{mp}, \qquad k = m,$$

where E_k consists of the last p_k columns of the n_k by n_k identity matrix I_{n_k}. Also,

$$A\mathcal{V}_k = V_{k+1} \hat{\mathcal{H}}_k.$$

The parameters p_k and n_k computed by algorithm in order to keep track of the partitioning of the matrices $\mathcal{V}_k$ and $\mathcal{H}_k$. Specifically, $\mathcal{V}_k$ is an $n \times n_k$ matrix and H_{ij} is p_i by p_j matrix. If we attempt to run $k > m$ steps of Algorithm 2, then it terminates normally after $k = m$ steps. We will use $\mathcal{H}_m$ and $\mathcal{V}_m$ to define a set of matrices $\{\mathcal{V}_j\}_{j=1}^{\infty}$ and $\{\bar{\mathcal{H}}_j\}_{j=1}^{\infty}$ as follows. Let for $j < m$, $\mathcal{V}_j = \mathcal{V}_m(:, 1 : n_j)$ and $\bar{\mathcal{H}}_j = \mathcal{H}_j(1 :, n_{j+1}, 1 : n_j)$. Also, for $j \geq m$, $\mathcal{V}_j = \mathcal{V}_m$ and $\bar{\mathcal{H}}_j = \mathcal{H}_m$. Then,

$$A\mathcal{V}_j = \mathcal{V}_j \bar{\mathcal{H}}_j, \quad j = 1, 2, ...$$

This type of factorization allows us to do induction on j without concerning ourselves with whether $j < m$ or $j \geq m$. This simplifies the analysis of algorithms such as conjugate gradient, where we might be interested in executing more than m iterations.

Theorem 1. Let A, E, G, n_j and $\mathcal{V}_j$ be as in above. Then there exist n_j by n_j matrices Π_j, such that the vectors u_j computed by applying Algorithm 1 to the matrix $\tilde{A}$, and the vector $\tilde{g}$ satisfy

$$u_j = vec(\mathcal{V}_j \Pi_j \mathcal{V}_j^T).$$

Proof : Let $\beta = \| \tilde{g} \|_2$. Initially, the block Arnoldi algorithm does a rank revealing QR decomposition of G,

$$GP = [\hat{Q}_1 \quad \hat{Q}_2] \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix},$$

where P is a permutation matrix, from which it follows

$$GG^T = \hat{Q}_1(R_{11}R_{11}^T + R_{12}R_{12}^T)\hat{Q}_1^T.$$

since $\mathcal{V}_1 = \hat{Q}_1$ by definition, we see that

$$u_1 = \tilde{g}/\beta = vec(GG^T)/\beta = vec(\mathcal{V}_1\Pi_1\mathcal{V}_1^T)$$

provided we choose

$$\Pi_1 = (R_{11}R_{11}^T + R_{12}R_{12}^T)/\beta.$$

It is clear that Π_1 is a symmetric n_1 by n_1 matrix. Thus the theorem is true for $i = 1$. Now, suppose that the statement is true for $i = 1, 2, ...j$. By the Arnoldi algorithms and properties of the Kronecker product

$w_j = \tilde{A}u_j = (A \otimes E + E \otimes A)u_j =$
$(A \otimes E)u_j + (E \otimes A)u_j = (A \otimes E)vec(\mathcal{V}_j\Pi_j\mathcal{V}_j^T) + (E \otimes A)vec(\mathcal{V}_j\Pi_j\mathcal{V}_j^T) =$
$vec(A\mathcal{V}_j\Pi_j\mathcal{V}_j^TE^T + E\mathcal{V}_j\Pi_jV_j^TA^T) = vec(\mathcal{V}_{j+1}\hat{\mathcal{H}}_j\Pi_j\mathcal{V}_j^TE^T + E\mathcal{V}_j\Pi_j\hat{\mathcal{H}}_j^T\mathcal{V}_{j+1}^T)$.
We claim that there exists a n_{j+1} by n_{j+1} symmetric matrix Γ_j such that

$$w_j = vec(\mathcal{V}_{j+1}\Gamma_j\mathcal{V}_{j+1}^T). \tag{4}$$

Since $\mathcal{V}_{j+1}\mathcal{V}_{j+1}^T$ is the orthogonal projection onto the column space of $\mathcal{V}_{j+1}$, which includes the column space $\mathcal{V}_j$, thus

$$\mathcal{V}_j = \mathcal{V}_{j+1}\mathcal{V}_{j+1}^T\mathcal{V}_j,$$

and we may write

$$\mathcal{V}_{j+1}\hat{\mathcal{H}}_j\Pi_j\mathcal{V}_j^TE^T + E\mathcal{V}_j\Pi_j\hat{\mathcal{H}}_j^T\mathcal{V}_{j+1}^T = \mathcal{V}_{j+1}(\hat{\mathcal{H}}_j\Pi_j\mathcal{V}_j^T\mathcal{V}_{j+1}E^T + E\mathcal{V}_{j+1}^T\mathcal{V}_j\Pi_j\hat{\mathcal{H}}_j^T)\mathcal{V}_{j+1}^T.$$

We see that

$$\Gamma_j = \hat{\mathcal{H}}_j\Pi_j\mathcal{V}_j^T\mathcal{V}_{j+1}E^T + E\mathcal{V}_{j+1}^T\mathcal{V}_j\Pi_j\hat{\mathcal{H}}_j^T, \tag{5}$$

is the choice which satisfies equation (4). Clearly, if Π_j symmetric, then Γ_j is symmetric. The inner lop of the Arnoldi consists of the linear updates

$$w_j = w_j - u_i h_{ij},$$

where $h_{ij} = u_i^T w_j$, which correspond to the linear updates

$$\Gamma_j = \Gamma_j - \mathcal{V}_{j+1}^T \mathcal{V}_i \Pi_i \mathcal{V}_i^T \mathcal{V}_{j+1} h_{ij}. \tag{6}$$

This observation shows that the compact representation of w_j can be maintained and if Γ_j is symmetric before the inner loop, then the final value of Γ_j will be symmetric as well. After the inner loop has been computed, $h_{j+1,j} = \| w_j \|_2$ is computed and compared to zero. If $h_{j+1,j} = 0$, then the algorithm terminates, otherwise

$$u_{j+1} = w_j / h_{j+1,j},$$

which corresponds to

$$\Pi_{j+1} = \Gamma_j / h_{j+1,j},$$

and we have

$$u_{j+1} = vec(\mathcal{V}_{j+1} \Pi_{j+1} \mathcal{V}_{j+1}^T).$$

This completes the proof. □

3 Compact form of CG method to Eq.(1)

As we know, the generalized Lyapunov matrix equations (1) can be written as a big linear system of equations

$$(A \otimes E + E \otimes A) vec(X) = vec(GG^T) \tag{7}$$

where $\otimes$ denote the Kronecker product, and for each matrix $L = [l_{ij}] \in R^{m \times n}$, the vector $vec(L)$ is define as following

$$vec(L) = (l_{11}, ..., l_{m1}, l_{12}, ..., l_{m2}, ..., l_{1n}, ..., l_{mn})^T.$$

In this section we show that the conjugate gradient algorithm for linear systems can be adapted to the generalized Lyapunov matrix equations $EXA^T + AXE^T - GG^T = 0$, written in Kronecker product form

$$\tilde{A}\tilde{x} = \tilde{g}, \tag{8}$$

where

$$\tilde{A} = (A \otimes E + E \otimes A), \quad \tilde{x} = vec(X), \quad \tilde{g} = vec(GG^T).$$

It is clear that if A and E are symmetric positive definite, then $\tilde{A}$ is symmetric positive definite [10]. However, while it is possible to apply the classical algorithm directly to the equivalent linear system it is not practical to do so, because of the time and memory requirements which are both $O(n^2)$. We state the classical algorithm as Algorithm 3 for sole purpose of proving Theorem 1. The tilde superscript is used to emphasize that we are computing in R^{n^2} rather than R^n. The conjugate gradient algorithm for obtaining approximate solution of Eq. (4) as following:

Algorithm 3. Clasical conjugate gradient algorithm for solving Eq. (4)

1. Compute $\tilde{r}_0 = \tilde{g} - \tilde{A}\tilde{x}_0, \tilde{p}_0 = \tilde{r}_0$
2. For $j = 0, 1, ...,$until convergence, Do
3. $\alpha_j = (\tilde{r}_j, \tilde{r}_j)/(\tilde{A}\tilde{p}_j, \tilde{p}_j)$
4. $\tilde{x}_{j+1} = \tilde{x}_j + \alpha_j \tilde{p}_j$
5. $\tilde{r}_{j+1} = \tilde{r}_j - \alpha_j \tilde{A}\tilde{p}_j$
6. $\beta_j = (\tilde{r}_{j+1}, \tilde{r}_{j+1})/(\tilde{r}_j, \tilde{r}_j)$
7. $\tilde{p}_{j+1} = \tilde{r}_{j+1} + \beta_j \tilde{p}_j$
8. EndDo

Theorem 2. Let A, E, G, $\mathcal{V}_j$ be as in section 2. Then there exist n_j by n_j symmetric matrices R_j, P_j, and X_j, such that the vectors computed by Algorithm 3 are given by

$$\tilde{r}_j = vec(\mathcal{V}_j R_j \mathcal{V}_j^T), \quad \tilde{p}_j = vec(\mathcal{V}_j P_j \mathcal{V}_j^T)$$

and

$$\tilde{x}_j = vec(\mathcal{V}_j X_j \mathcal{V}_j^T).$$

Proof : The proof is by induction on j and follows Algorithm 3 step by step. It is clear that the statements of the theorem are true for $j = 1$. From the QR factorization of G (see the proof of theorem 1)we have

$$GG^T = \hat{Q}_1(R_{11}R_{11}^T + R_{12}R_{12}^T)\hat{Q}_1^T,$$

and we can write

$$R_1 = P_1 = R_{11}R_{11}^T + R_{12}R_{12}^T,$$

and

$$X_1 = zeros(n_1, n_1).$$

Now assume that the statements of the theorem are true for $j \geq 1$. Then,

$$\alpha_j = (\tilde{r}_j, \tilde{r}_j)/(\tilde{A}\tilde{p}_j, \tilde{p}_j) = \| R_j \|_F / trace(\Gamma(P_j)^T \mathcal{V}_{j+1}^T \mathcal{V}_j P_j \mathcal{V}_j^T \mathcal{V}_{j+1})$$

where we have defined

$$\Gamma(P_j) = \hat{\mathcal{H}}_j P_j \mathcal{V}_j^T E^T \mathcal{V}_{j+1} + \mathcal{V}_{j+1}^T E \mathcal{V}_j P_j \hat{\mathcal{H}}_j^T.$$

Now,

$$\tilde{x}_{j+1} = \tilde{x}_j + \alpha_j \tilde{p}_j = vec(\mathcal{V}_j X_j \mathcal{V}_j^T) + \alpha_j vec(\mathcal{V}_j P_j \mathcal{V}_j^T) = vec(\mathcal{V}_j(X_j + \alpha_j P_j)\mathcal{V}_j^T),$$

and we have

$$X_{j+1} = \mathcal{V}_{j+1}^T \mathcal{V}_j (X_j + \alpha_j P_j) \mathcal{V}_j^T \mathcal{V}_{j+1}.$$

Similarly,

$$\tilde{r}_{j+1} = \tilde{r}_j - \alpha_j \tilde{A}\tilde{p}_j = \tilde{r}_j - \alpha_j (A \otimes E + E \otimes A)\tilde{p}_j = vec(\mathcal{V}_j R_j \mathcal{V}_j^T) -$$
$$\alpha_j vec(A\mathcal{V}_j P_j \mathcal{V}_j^T E^T + E\mathcal{V}_j P_j \mathcal{V}_j^T A^T) = vec(\mathcal{V}_{j+1}(\mathcal{V}_{j+1}^T \mathcal{V}_j R_j \mathcal{V}_j^T \mathcal{V}_{j+1} - \alpha_j \Gamma(P_j)\mathcal{V}_{j+1}^T),$$

and we have

$$R_{j+1} = \mathcal{V}_{j+1}^T \mathcal{V}_j R_j \mathcal{V}_j^T \mathcal{V}_{j+1} - \alpha_j \Gamma(P_j).$$

Finally, we have

$$\beta_j = (\tilde{r}_{j+1}, \tilde{r}_{j+1})/(\tilde{r}_j, \tilde{r}_j) = \| R_{j+1} \|_F \, / \, \| R_j \|_F,$$

and

$$\tilde{p}_{j+1} = \tilde{r}_{j+1} + \beta_j \tilde{p}_j = vec(\mathcal{V}_{j+1} R_{j+1} \mathcal{V}_{j+1}^T) + \beta_j vec(\mathcal{V}_j P_j \mathcal{V}_j^T) =$$
$$vec(\mathcal{V}_{j+1}(R_{j+1} + \beta_j \mathcal{V}_{j+1}^T \mathcal{V}_j P_j \mathcal{V}_j^T \mathcal{V}_{j+1})),$$

which yields

$$P_{j+1} = R_{j+1} + \beta_j \mathcal{V}_{j+1}^T \mathcal{V}_j P_j \mathcal{V}_j^T \mathcal{V}_{j+1}.$$

We note, that if P_j is symmetric, then $\Gamma(P_j)$ is symmetric and the symmetry of R_{j+1}, P_{j+1}, and X_{j+1} follows from the symmetry of R_j, P_j, and X_j. We have shown that the statements of the theorem are true for $j+1$, which completes the proof. □

From the proof of above theorem it is clear that once the matrices $\hat{\mathcal{H}}_j$ has been computed by the block Arnoldi algorithm, then the rest of computation can be done with small and dense matrices. We formalized this statement in Algorithm 4.

Algorithm 4. Compact version of conjugate gradient algorithm for Eq.(4)

1. Set $X_1 = zeros(n_1, n_1)$ and compute $R_1 = P_1 = \mathcal{V}_1^T G G^T \mathcal{V}_1$.
2. For $j = 1, 2, ...,$until convergence, Do
3. $\quad \Gamma(P_j) = \hat{\mathcal{H}}_j P_j \mathcal{V}_j^T E^T \mathcal{V}_{j+1} + \mathcal{V}_{j+1}^T E \mathcal{V}_j P_j \hat{\mathcal{H}}_j^T$
4. $\quad \alpha_j = \| R_j \|_F^2 \, / trace(\Gamma(P_j)^T \mathcal{V}_{j+1}^T \mathcal{V}_j P_j \mathcal{V}_j^T \mathcal{V}_{j+1})$
5. $\quad X_{j+1} = \mathcal{V}_{j+1}^T \mathcal{V}_j (X_j + \alpha_j P_j) \mathcal{V}_j^T \mathcal{V}_{j+1}$
6. $\quad R_{j+1} = \mathcal{V}_{j+1}^T \mathcal{V}_j R_j \mathcal{V}_j^T \mathcal{V}_{j+1} - \alpha_j \Gamma(P_j)$
7. $\quad \beta_j = \| R_{j+1} \|_F^2 \, / \, \| R_j \|_F^2$
8. $\quad P_{j+1} = R_{j+1} + \beta_j \mathcal{V}_{j+1}^T \mathcal{V}_j P_j \mathcal{V}_j^T \mathcal{V}_{j+1}$
9. EndDo

Table 1: Number of iterations and CPU times to converge for the first set of examples.

n^2	100	400	900	1600	2500
CG	756(0.06)	2164(5.31)	3984(60.3)	5932(298)	8126(978)
CCG	525(0.05)	1512(4.02)	2785(49.5)	4125(221)	6251(635)

4 Numerical Experiments

In this section, we compared the performance of the classical conjugate gradient (CG) algorithm and compact version of conjugate gradient algorithm (CCG) for solving Eq.(1). All the numerical experiments presented in this section were computed with some MATLAB codes on a personal computer Pentium3-800EB MHs. For all the examples, we used the stopping criterion

$$\frac{\parallel EX_kA^T + AX_kE^T - GG^T \parallel_F}{\parallel GG^T \parallel_F} \leq 10^{-6},$$

and the maximum number of iterations allowed set to 10000. For all the experiments, the initial guess was $X_0 = 0$. The matrix G is defined so that a true solution of equation (1) is the matrix of all ones. For the first set of our examples, we used the matrices

$$A = [(2^{-t}-1)I_n + diag(1,2,...,n) + U_n^T] + [(2^{-t}-1)I_n + diag(1,2,...,n) + U_n^T]^T$$

and

$$E = [I_n + 2^{-t}U_n] + [I_n + 2^{-t}U_n]^T,$$

where U_n is the $n \times n$ matrix with unit entries below the diagonal and all other entries zero. The results were reported in Table 1 with different values of n, i.e., $n = 10, 20, 30, 40$ and 50. Here, we let $t = 1$, and $p = 10$. For the second set of experiments we used some matrices from Harwell-Boing collection for the matrix A. Generic properties of these matrices and results are given in Table 2. Also, the matrix G is defined so that a true solution of equation (1) is the matrix of all ones. The results which have reported in Table1 and Table 2, shown that all generalized Lyapunov equations were solved in a relatively small number of iterations by CCG algorithm.

5 Conclusion

As we know, applying Krylov subspace methods directly to Kronecker form of generalized Lyapunov matrix equations using $O(n^2)$ memory and time. Modi-

Table 2: Generic properties of matrix A, number of iterations and CPU times to converge for the second set of examples.

Matrix	Order	Nnz	Condition number	CG	CCG
Bcsstk01	48	224	1.6e+06	†	435(685)
Bcsstm02	66	66	8	466(191)	165(112)
Bcsstk02	66	2211	1.3e+03	†	673(936)

† =no solution has been obtained after 10000 iterations.

fications that reduce the memory requirements of Krylov subspace methods for solving generalized Lyapunov matrix equations have been investigated. While the using of the Krylov subspace methods for the Lyapunov matrix equation is already well understood (a more detail analysis will appear in [18]), we have only sketch a possible approach for the generalized Lyapunov matrix equations. In this paper, we have shown that it is possible to reduce this requirements to $O(n)$ for conjugate gradient method.

Acknowledgment

This research was supported by research grant at Abadan Azad University, Iran.

References

[1] R. H. Bartels and G. W. Stewart, Solution of matrix equation AX+XB=C. Comm. ACM 15:820-826, 1972.

[2] P. Benner. Contributions to the Numerical Solution of Algebraic Riccati Equations and Related Eigenvalue Problems. Logos Verlag, Berlin, 1997.

[3] P. Benner and E.S Quintana-Orti. Solving stable generalized Lyapunov equations with the matrix sign function. Numerical Algorithms, 20(1):75-100, 1999.

[4] R. Byers. Solving the algebraic Riccati equation with matrix sign function. Linear Algebra Appl., 85:267-279,1987.

[5] F. R. Gantmacher, Theory of Matrices, Chelsea, New York, 1959.

[6] J. D. Gardiner and A.J. Laub. A generalization of the matrix sign function solution for algebraic Riccati equations. Internet. J. Control, 44(3):823-832,1986.

[7] G. H. Golub and G. F. Van Loan, Matrix computaions, third ed., The Johns Hopkins U.P.,Baltimore, London, 1996.

[8] G. H. Golub, S. Nash and G. F. Van Loan, A Hessenberg-Schur method for the problem $AX - XB = C$, IEEE Transe. Automat. Control AC-24:909-913, 1979.

[9] S. Hammarling, Numerical solution of the stable, non-negetive definite Lyapunov equation, IMA J. Numer. Anal., 2 pp. 303-323, 1982.

[10] R. A. Horn and C. R. Johnson, Topics in matrix analysis, Cambridge University Press 1991.

[11] C. S. Kenny, A. J. Laub, The matrix sign function, IEEE Trans. Automat. Control 40(8),pp 1330-1348, 1995.

[12] P. Lancaster, L. Rodman, The Algebraic Riccati Equation, Oxford University Press, Oxford, 1995.

[13] V. B. Larin, Construction of a solution of the generalized Lyapunov equation, Russian Acad. Sci.,Dokl., Math. 47(1), pp 17-20, 1993.

[14] V. B. Larin and F. A. Aliev. Construction of square root factor for solution of the Lyapunov matrix equation. Systems Control Lett.,20(2):109-112, 1993.

[15] V. B. Larin and F. A. Aliev. Generalized Lyapunov equation and factorization of matrix polynomials. Systems Control Lett., 21(6):485-491, 1993.

[16] A. J. Laub, M. T. Heath, C. C. Paige, and R. C. Ward, Computation of sysrem balancing transformations and other applications of simultaneous diagonalization algorithms. IEEE Trans. Automat. Control., vol AC-32, pp 115-122, 1987.

[17] J.-R. Li and J. White. Low-Rank solution of Lyapunov equations. SIAM J. Matrix Anal. Appl., 24(1):260-280, 2002.

[18] C. K. Mikkelsen, Numerical Method for Large Lyapunov Equations, PhD Thesis, Purdue University, 2009.

[19] T. Penzl, A cyclic low-rank Smith method for large sparse Lyaponuv matrix equations, SIAM J, Sci. Comput., 21 (2000),pp. 1401-1418.

[20] Y. Saad, Iterative Methods for Sparse linear Systems(PWS Press, New York, 1995)

[21] T. Stykle, A modified matrix sign function method for projected Lyapunov equations. Technical report 336-2006.

[22] T. Stykle, Numerical solution and perturbation theory for generalized Lyapunov equations. Linear Algebra and applications 349, pp 155-185, 2002.

Received: July, 2011