Electronic Transactions on Numerical Analysis.
Volume 43, pp. 60-69, 2014.

ISSN 1068-9613.

ETNA
Kent State University
http://etna.math.kent.edu

THE DAVISON-MAN METHOD REVISITED*

MILOUD SADKANE†

Abstract. The Davison-Man method is an iterative technique for solving Lyapunov equations where the approximate solution is updated through a matrix integral and a doubling procedure. In theory, convergence is quadratic, but, in practice, there are examples where the method stagnates and no further improvement is seen. In this work, an implementation that avoids stagnation and improves the computed solution is proposed and justified.

Key words. Davison-Man, Lyapunov equation

AMS subject classifications. 65F30, 65F10

1. Introduction. Lyapunov equations appear in many applications such as control and model reduction; see, e.g., [2, 4]. For small-size matrix equations, the method of Bartels and Stewart [1] is widely used. After transformation to Schur form, this method solves the resulting triangular systems and then recovers the solution by pre and post matrix multiplications. It requires $\mathcal{O}(n^2)$ memory storage and $\mathcal{O}(n^3)$ operations, where n is the size of the matrix equations, and it is implemented in MATLAB's `lyap` function.

In 1968, Davison and Man [3] proposed an iterative method for solving Lyapunov equations. The method updates the approximate solution through matrix integrals and a doubling procedure involving matrix exponentials. In their implementation, the first integral is roughly approximated by the left-rectangle numerical integration formula, and the matrix exponentials are computed by a Crank-Nicolson type method. This results in inaccurate computed solutions.

The purpose of the present note is to revisit this method. We show theoretically and numerically that when the first integral and the matrix exponentials are computed accurately, the method converges quadratically. However, in practice, the iterations may stagnate and no further improvement is seen. We show how to avoid stagnation while continuing to improve the solution. Numerical illustrations carried out in MATLAB are given throughout the note.

The following notation is used: $\|\ \|_2$ and $\|\ \|_F$ denote the 2-norm and Frobenius norm, $A = A^T > 0$ ($A = A^T \geq 0$) means that the matrix A is symmetric and positive definite (semidefinite), I denotes the identity matrix, where the order is clear from the context, $A \otimes B$ denotes the Kronecker product of the matrices A and B, and $\mathrm{vec}(A)$ denotes an ordered stack of the columns of the matrix A from left to right starting with the first column; see, e.g., [5].

2. The Davison-Man algorithm. Consider the Lyapunov equation

$$AX + XA^T + C = 0, \tag{2.1}$$

where $A, C \in \mathbb{R}^{n\times n}$. We assume that A is stable (its eigenvalues have negative real parts) and $C = C^T \geq 0$, which is the case in many applications (see, e.g., [2]), although the case $C \neq C^T$ is also important and will be considered later; see Section 2.2.

Then it is known that the unique solution to (2.1) is given by

$$X = \int_0^\infty e^{tA} C e^{tA^T} dt = X^T \geq 0. \tag{2.2}$$

*Received October 8, 2013. Accepted June 18, 2014, Published online on September 5, 2014. Recommended by L. Reichel.

†Université de Brest, CNRS-UMR 6205, Laboratoire de Mathématiques de Bretagne Atlantique, 6. Av. Le Gorgeu, 29238 Brest Cedex 3, France (`miloud.sadkane@univ-brest.fr`).

ETNA
Kent State University
http://etna.math.kent.edu

We can write

$$X = \lim_{j\to\infty} X_j, \quad X_j = \int_0^{2^j h} e^{tA} C e^{tA^T}\, dt, \tag{2.3}$$

where h is a scaling parameter. From (2.2) and (2.3) we have

$$X = X_j + e^{2^j hA} X e^{2^j hA^T}, \tag{2.4}$$

and since A is stable and $e^{2^j hA} = \left(e^{2^{j-1}hA}\right)^2$, we have $e^{2^j hA} \to 0$ quadratically as $j \to \infty$. The residual associated with X_j is $R_j = AX_j + X_j A^T + C$ and can be written

$$R_j = \int_0^{2^j h} \left(\frac{d}{dt} e^{tA} C e^{tA^T}\right) dt + C = e^{2^j hA} C e^{2^j hA^T}.$$

When $j = 0$, equation (2.4) is the Stein equation

$$X - e^{hA} X e^{hA^T} = X_0. \tag{2.5}$$

Since A is stable, the eigenvalues of e^{hA} lie in the open unit disk and the Smith or squared Smith method [6] can be applied to (2.5) provided that e^{hA} and X_0 are computed accurately.

Note that from (2.3), the matrix X_j can be decomposed as

$$X_j = X_{j-1} + e^{2^{j-1}hA} X_{j-1} e^{2^{j-1}hA^T}. \tag{2.6}$$

Writing (2.4) at iteration $j-1$ and subtracting it from (2.6) gives

$$X - X_j = e^{2^{j-1}hA}(X - X_{j-1}) e^{2^{j-1}hA^T}.$$

As a consequence

$$X - X_j = e^{(2^j-1)hA}(X - X_0) e^{(2^j-1)hA^T},$$

which shows that the convergence is essentially quadratic.

The computation of X_j is summarized in the following algorithm.

Algorithm 1 (Davison-Man).

Input: matrices A and C and a parameter h.
Output: matrix X_j.

$X_0 = \int_0^h e^{tA} C e^{tA^T}\, dt$ and $E_0 = e^{hA}$
for $j = 1, 2, \ldots$ **do**
  $X_j = X_{j-1} + E_{j-1} X_{j-1} E_{j-1}^T$
  $E_j = E_{j-1}^2$
end for

It is clear that X_0 and E_0 must be computed accurately. The algorithm requires $\mathcal{O}(n^2)$ memory storage and $\mathcal{O}(n^3)$ operations. In [3] (see also [4]), the parameter h is chosen as $h = \frac{1}{20|\lambda_{\max}(A)|}$, where $\lambda_{\max}(A)$ is the eigenvalue of A with maximal real part or magnitude, and the matrices X_0 and E_0 are approximated, respectively, by the left-rectangle method and a modification of the Crank-Nicolson method. With these approximations there is no guarantee that the computed solution will be accurate.

ETNA
Kent State University
http://etna.math.kent.edu

The problem that may affect the convergence of Algorithm 1 is related to the stability of A. In theory, as mentioned above, the fact that the eigenvalues lie in the left half of the complex plane guarantees convergence. In practice, it is the behavior of the exponentials e^{tA} for $t > 0$ that governs the convergence: an excessive growth of $\|e^{tA}\|_2$ leads to divergence of the sequence X_j, and a rapid decay of this function leads to stagnation.

A sufficient but not necessary condition to avoid the growth of $\|e^{tA}\|_2$ is

$$\lambda_{\max}\left((A + A^T)/2\right) \le \mu < 0, \tag{2.7}$$

where $\lambda_{\max}$ denotes the largest eigenvalue and μ is a negative scalar, ideally far from 0.

This condition implies that (see, e.g., [5])

$$\|e^{tA}\|_2 \le e^{t\mu}.$$

It also ensures that the solution X and hence the sequence X_j remain bounded since

$$\|X_j\|_2 \le \|C\|_2 \int_0^\infty \|e^{tA}\|_2^2\, dt \le \frac{\|C\|_2}{2|\mu|}.$$

In practice, the growth of the exponentials is reflected by a growth in the sequence X_j. This situation is easy to handle: when such a growth is observed, Algorithm 1 must be stopped.

The opposite situation is the rapid decay of $\|e^{tA}\|_2$. This is reflected by a stagnation of the sequence X_j at early iterations as the for-loop of Algorithm 1 clearly shows. This is the point we would like to discuss in this note. An illustration is given below.

Example 1. In this and the following examples, the parameter h is chosen to be of the form $h = a/\|A\|$, where $0 < a < 1$ and $\|A\|$ is some norm of A, e.g., $h = 0.5/\|A\|_1$. The matrix X_0 is approximated by the adaptive Simpson quadrature rule and E_0 is computed by the scaling and squaring method (MATLAB's quadv and expm functions).

We consider equation (2.1) with $n = 500$. The matrix A is Toeplitz bidiagonal with -2 on its diagonal and 1 on its upper diagonal, and $C = ee^T$, where e is the vector of all ones. Figure 2.1 (left) shows that the quadratic convergence is followed by stagnation.

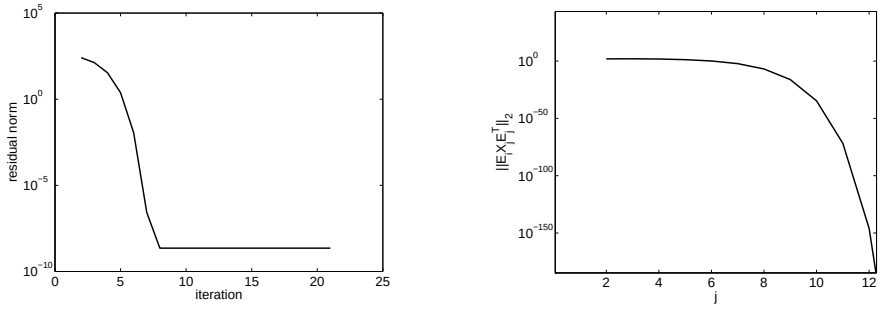

FIG. 2.1. *Left: Convergence of Algorithm 1. Right: Behavior of* $\|E_j X_j E_j^T\|_2$ *(Example 1).*

It is clear that the source of the stagnation is the rapid decay of the term $E_{j-1}X_{j-1}E_{j-1}^T$; see Figure 2.1(right). Since $\|E_{j-1}X_{j-1}E_{j-1}^T\|_2 \le \|X_{j-1}\|\|e^{2^{j-1}hA}\|_2^2$, a few iterations (8 in the present example) suffice to make the entries of $E_{j-1}X_{j-1}E_{j-1}^T$ tiny so that $X_j = X_{j-1} + E_{j-1}X_{j-1}E_{j-1}^T \approx X_{j-1}$. One way to avoid stagnation is to stop the iterations when the norm of $E_{j-1}X_{j-1}E_{j-1}^T$ becomes negligible. This gives rise to Algorithm 2.

ETNA
Kent State University
http://etna.math.kent.edu

Algorithm 2

Input: matrices A and C and parameters h and tol.
Output: approximate solution X_k of (2.1).
$X_0 = \int_0^h e^{tA} C e^{tA^T} dt$, $E_0 = e^{hA}$, $k = 0$
while $\|E_k X_k E_k^T\|_2 > \text{tol}$ **do**
$k := k + 1$
$X_k = X_{k-1} + E_{k-1} X_{k-1} E_{k-1}^T$
$E_k = E_{k-1}^2$
end while

This algorithm avoids stagnation but does not necessarily lead to a better solution. Below we discuss two ways to improve the computed solution.

2.1. Postprocessing. A first way is to refine the approximate solution through a "Galerkin process" which uses the space spanned by the columns of the solution computed by Algorithm 2. This simple strategy will be referred to as postprocessing. It is summarized in the following algorithm.

Algorithm 3 Postprocessing step

Input: approximate solution X_k computed by Algorithm 2.
Output: refined solution $\hat{X}$.
Compute $V_k = \text{orth}(X_k)$, $A_k = V_k^T A V_k$, $C_k = V_k^T C V_k$
Solve for Y: $A_k Y + Y A_k^T + C_k = 0$
Set $\hat{X} = V_k Y V_k^T$

In Algorithm 3, $\text{orth}(X_k)$ denotes any column orthonormalization procedure applied to X_k. The Lyapunov equation for Y may be solved by the lyap function or by simply reusing Algorithm 2. Note that when condition (2.7) is satisfied, $A_k + A_k^T = V_k^T (A + A^T) V_k < 0$. In particular, A_k is stable.

To illustrate the contribution of the postprocessing step, consider again Example 1. Algorithm 2 stops at iteration $k = 8$ with an approximate solution X_k, whose residual norm is $\|R_k\|_2 = 2.25 \cdot 10^{-9}$. The postprocessing step, that is, a new application of Algorithm 2 to the equation $A_k Y + Y A_k^T + C_k = 0$, leads to $\hat{X}$ with $\|A\hat{X} + \hat{X}A^T + C\|_2 = 6.00 \cdot 10^{-12}$.

To understand the effect of the postprocessing step, we assume that condition (2.7) is satisfied, denote by Y_k an approximation of Y, and by

$$\Delta_k = A_k Y_k + Y_k A_k^T + C_k \tag{2.8}$$

the corresponding residual. Also, let $\hat{X}_k = V_k Y_k V_k^T$ be the refined solution associated with Y_k and

$$\hat{R}_k = A\hat{X}_k + \hat{X}_k A^T + C \tag{2.9}$$

the corresponding residual.

Let $\mathcal{A} = I \otimes A + A \otimes I$, $\mathcal{V}_k = V_k \otimes V_k$, and $\mathcal{P}_k = \mathcal{A}\mathcal{V}_k \left(\mathcal{V}_k^T \mathcal{A} \mathcal{V}_k\right)^{-1} \mathcal{V}_k^T$ be the projector onto $\text{range}(\mathcal{A}\mathcal{V}_k)$ along the orthogonal complement of $\text{range}(\mathcal{V}_k)$. Since

$$\lambda_{\max}\left(\left(\mathcal{V}_k^T \left(\mathcal{A} + \mathcal{A}^T\right) \mathcal{V}_k\right)/2\right) \le \lambda_{\max}\left(\left(\mathcal{A} + \mathcal{A}^T\right)/2\right) \le 2\lambda_{\max}\left((A + A^T)/2\right),$$

condition (2.7) implies in particular that $\| \left(\mathcal{V}_k^T \mathcal{A} \mathcal{V}_k\right)^{-1} \|_2 = \left\| \int_0^\infty e^{t\mathcal{V}_k^T \mathcal{A} \mathcal{V}_k} dt \right\|_2 \le 1/2|\mu|$ and hence, $\|\mathcal{P}_k\|_2 \le \|\mathcal{A}\|_2/2|\mu| \le \|A\|_2/|\mu|$. Therefore, $\|\mathcal{P}_k\|_2$ remains bounded by a small constant when $\|A\|_2/|\mu|$ is not large.

ETNA
Kent State University
http://etna.math.kent.edu

The link between Δ_k and $\hat{R}_k$ is given by the following simple proposition.

PROPOSITION 2.1. *The residual $\hat{R}_k$ associated with the refined solution $\hat{X}_k$ satisfies*

$$V_k^T \hat{R}_k V_k = \Delta_k, \tag{2.10}$$

$$\mathrm{vec}(\hat{R}_k) = (I - \mathcal{P}_k)\,\mathrm{vec}(C) + \mathcal{P}_k \mathcal{V}_k \mathrm{vec}(\Delta_k). \tag{2.11}$$

Proof. Equality (2.10) comes from (2.8) and (2.9). It is easy to check that

$$\begin{aligned}
\mathrm{vec}(\hat{R}_k) &= \mathcal{A}\mathrm{vec}(\hat{X}_k) + \mathrm{vec}(C),\\
\mathrm{vec}(\hat{X}_k) &= \mathcal{V}_k \mathrm{vec}(Y_k),\\
\mathrm{vec}(\Delta_k) &= \left(\mathcal{V}_k^T \mathcal{A} \mathcal{V}_k\right) \mathrm{vec}(Y_k) + \mathcal{V}_k^T \mathrm{vec}(C).
\end{aligned}$$

Inserting the expression of $\mathrm{vec}(Y_k)$ from the last equality into that of $\mathrm{vec}(\hat{X}_k)$ leads to (2.11). ∎

Since $\mathcal{P}_k$ is a projector onto $\mathrm{range}(\mathcal{A}\mathcal{V}_k)$, we have for all matrices S (of appropriate sizes)

$$\mathrm{vec}(\hat{R}_k) = (I - \mathcal{P}_k)\,(\mathrm{vec}(C) + \mathcal{A}\mathcal{V}_k \mathrm{vec}(S)) + \mathcal{P}_k \mathcal{V}_k \mathrm{vec}(\Delta_k).$$

Hence,

$$\|\hat{R}_k\|_F \le \|\,(I - \mathcal{P}_k)\,\|_2 \min_S \|C + A(V_k S V_k^T) + (V_k S V_k^T) A^T\|_F + \|\mathcal{P}_k\|_2 \|\Delta_k\|_F.$$

Since a factorization of X_k of the form $V_k S V_k^T$ can obviously be found, the above inequality implies that

$$\|\hat{R}_k\|_F \le \|I - \mathcal{P}_k\|_2 \|R_k\|_F + \|\mathcal{P}_k\|_2 \|\Delta_k\|_F. \tag{2.12}$$

Note that this inequality brings together the three residuals involved in the postprocessing step.

Recall that the norm of X_k remains bounded by a small constant due to condition (2.7). However, X_k can be ill-conditioned (a part of its eigenvalues can be zero or close to zero). It is interesting to ask which columns of V_k contribute to the approximation. In order to answer this question, consider the spectral decomposition of X_k

$$X_k = U D U^T = U_1 D_1 U_1^T + U_2 D_2 U_2^T, \quad \lambda_{\max}(D_2) < \varepsilon \le \lambda_{\min}(D_1), \tag{2.13}$$

where $U = [U_1, U_2]$ and $D = \mathrm{blockdiag}(D_1, D_2)$ denote the matrices of orthonormal eigenvectors and eigenvalues of X_k and ε is a small tolerance that separates the largest and smallest eigenvalues. A natural choice is $V_k = U_1$. It leads to

$$V_k^T R_k V_k = V_k^T (A X_k + X_k A^T + C) V_k = A_k D_1 + D_1 A_k^T + C_k.$$

Therefore,

$$\|A_k D_1 + D_1 A_k^T + C_k\|_2 \le \|R_k\|_2.$$

Similar bounds can be obtained by increasing or decreasing the number of columns in U, irrespective of the choice of ε. Moreover, with the choice $V_k = U_1$, we have

$$\|A(V_k D_1 V_k^T - X_k) + (V_k D_1 V_k^T - X_k) A^T + R_k\|_2 \le 2\varepsilon \|A\|_2 + \|R_k\|_2,$$

ETNA
Kent State University
http://etna.math.kent.edu

and so

$$\min_S \|AV_kSV_k^T + V_kSV_k^TA^T + C\|_2 \le 2\varepsilon\|A\|_2 + \|R_k\|_2.$$

This bound can be improved by increasing the size of D_1 since this will lead to a small ε. The choice $D_1 = D$, that is, $V_k = U$, eliminates the term $2\varepsilon\|A\|_2$.

This discussion suggests that choices in favor of a larger size of D_1 may be efficient even if they involve extra costs. The following proposition results from the above discussion and the residual formula (2.11).

PROPOSITION 2.2. *If condition (2.7) is satisfied and $\|A\|_2/|\mu|$ is not large, then the residual $\hat{R}_k$ associated with the refined solution $\hat{X}_k$ satisfies*

$$\|\hat{R}_k\|_2 = \mathcal{O}\left(\|\left(I - \mathcal{P}_k\right)\operatorname{vec}(C)\|_2 + \|\Delta_k\|_2\right)$$

and the constant in the $\mathcal{O}$-notation is small.

To illustrate this proposition, consider the matrices A and C of Example 1 with $n = 50$. For this case, Algorithm 2 stops at iteration $k = 8$ with a solution whose residual norm is $\|R_k\|_2 = 5.64 \cdot 10^{-10}$. Table 2.1 shows the effect of the postprocessing when $V_k = U_1$ is chosen to satisfy (2.13) with different values of ε. For each ε, the table displays the number of columns of V_k denoted by $\operatorname{col}(V_k)$, the norms $\|\Delta_k\|_2$ and $\|\hat{R}_k\|_2$, and analogous norms $\|\Delta_k^{\mathsf{lyap}}\|_2$ and $\|\hat{R}_k^{\mathsf{lyap}}\|_2$ obtained when the lyap function instead of Algorithm 2 is used to compute Y_k. The norms of $\|I - \mathcal{P}_k\|_2$ and $\|\left(I - \mathcal{P}_k\right)\operatorname{vec}(C)\|_2$ are given in Table 2.2.

TABLE 2.1
Results of the postprocessing step with different values of ε (Example 1, $n = 50$).

ε	$\operatorname{col}(V_k)$	$\|\Delta_k\|_2$	$\|\hat{R}_k\|_2$	$\|\Delta_k^{\mathsf{lyap}}\|_2$	$\|\hat{R}_k^{\mathsf{lyap}}\|_2$
10^{-6}	5	$6.20 \cdot 10^{-10}$	$3.41 \cdot 10^{-6}$	$4.21 \cdot 10^{-14}$	$3.41 \cdot 10^{-6}$
10^{-8}	7	$5.53 \cdot 10^{-10}$	$1.75 \cdot 10^{-8}$	$1.79 \cdot 10^{-13}$	$1.75 \cdot 10^{-8}$
10^{-10}	9	$5.51 \cdot 10^{-10}$	$5.32 \cdot 10^{-10}$	$9.07 \cdot 10^{-14}$	$9.02 \cdot 10^{-11}$
10^{-12}	11	$5.30 \cdot 10^{-10}$	$5.30 \cdot 10^{-10}$	$1.15 \cdot 10^{-13}$	$1.12 \cdot 10^{-12}$
10^{-14}	12	$5.30 \cdot 10^{-10}$	$5.30 \cdot 10^{-10}$	$3.07 \cdot 10^{-13}$	$5.54 \cdot 10^{-13}$
10^{-16}	50	$2.47 \cdot 10^{-12}$	$2.48 \cdot 10^{-12}$	$9.97 \cdot 10^{-14}$	$9.37 \cdot 10^{-14}$

TABLE 2.2
Results of the postprocessing step with different values of ε (Example 1, $n = 50$).

ε	$\|I - \mathcal{P}_k\|_2$	$\|\left(I - \mathcal{P}_k\right)\operatorname{vec}(C)\|_2$
10^{-6}	1.08	$3.41 \cdot 10^{-6}$
10^{-8}	1.08	$1.75 \cdot 10^{-8}$
10^{-10}	1.08	$9.03 \cdot 10^{-11}$
10^{-12}	1.08	$1.40 \cdot 10^{-12}$
10^{-14}	1.08	$5.24 \cdot 10^{-13}$
10^{-16}	$4.08 \cdot 10^{-15}$	$9.14 \cdot 10^{-14}$

These tables show clearly that the convergence depends essentially on the maximum of $\|\left(I - \mathcal{P}_k\right)\operatorname{vec}(C)\|_2$ and $\|\Delta_k\|_2$. For example, when the value $\varepsilon = 10^{-6}$ is taken, we have $\|\hat{R}_k\| \approx \|\left(I - \mathcal{P}_k\right)\operatorname{vec}(C)\|_2$, and when $\varepsilon = 10^{-14}$, $\|\hat{R}_k\| \approx \|\Delta_k\|_2$, and these properties

ETNA
Kent State University
http://etna.math.kent.edu

hold with $\|\hat{R}_k^{\mathsf{lyap}}\|_2$ and $\|\Delta_k^{\mathsf{lyap}}\|_2$. The tables also show that ε must be carefully selected so that the columns of X_k and V_k span (numerically) the same space. A look at Figure 2.2 reveals that the numerical rank of X_k is around 11, and Table 2.1 shows that this is obtained with $\varepsilon = 10^{-12}$ and the results improve when $\varepsilon \leq 10^{-12}$. For these values of ε, Table 2.2 shows that $\mathrm{vec}(C)$ is almost in the range of $\mathcal{P}_k$. Note that inequality (2.12) is clearly not satisfied with $\varepsilon = 10^{-6}$ and $\varepsilon = 10^{-8}$.

All these arguments demonstrate that $V_k = U$ is an appropriate choice: it dispenses with the need for selecting ε, it yields $\mathcal{P}_k \approx I$ (and so the condition on $\|A\|_2/|\mu|$ is no longer needed), $\hat{R}_k \approx V_k \Delta_k V_k^T$, and hence, $\|\hat{R}_k\|_2 \approx \|\Delta_k\|_2$.

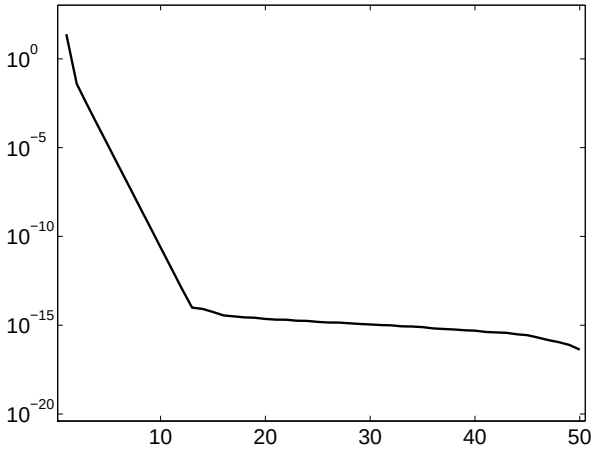

FIG. 2.2. *Singular values of X (Example 1, $n = 50$).*

2.2. Restart. A second way to get out of stagnation is to restart Algorithm 2 according to the principle of "iterative refinement". Suppose that Algorithm 2 has computed an approximate solution X_k which satisfies $\|R_k\|_2 \leq \mathsf{tol}_1 \|C\|_2$, where tol_1 is some tolerance, but that X_k is still considered unsatisfactory. Then we can compute another correction Y by solving the Lyapunov equation $AY + YA^T + R_k = 0$. In other words, we restart Algorithm 2 where C is now replaced by the residual R_k. If Algorithm 2 provides an approximation Y_k such that

$$\|AY_k + Y_k A^T + R_k\|_2 \leq \mathsf{tol}_2 \|R_k\|_2,$$

then

$$\|A(X_k + Y_k) + (X_k + Y_k)A^T + C\|_2 \leq \mathsf{tol}_1 \mathsf{tol}_2 \|C\|_2.$$

Therefore $X_k + Y_k$ is an improvement over X_k. This process can be continued until the solution meets a prescribed tolerance or the number of restarts becomes too large. One possible implementation is sketched in Algorithm 4.

In an "ideal" implementation, Algorithm 2 or a variant of this algorithm should be used at each restart of Algorithm 4. This would lead to a dynamic choice of k. However, with this approach, the matrices E_j would be recomputed at each restart. On the other hand, the numerical tests show that the values of k obtained during restarts generally differ by at most 1 or 2 iterations. Therefore, to avoid this drawback, the step $r = 1$ in Algorithm 4 is performed separately so that the matrices E_j are computed only once.

ETNA
Kent State University
http://etna.math.kent.edu

Algorithm 4 [restarted Davison-Man]

Input: matrices A and C, parameter h, tolerance tol,
maximum number of restarts $r_{\max}$.
Output: approximate solution $Y^{(r)}$ of (2.1).

```
Set r = 1, C^(0) = C
Compute (E_j)_{0≤j≤k} and X_k^(1) via Algorithm 2 starting with
X_0^(1) = ∫_0^h e^{tA} C^(0) e^{tA^T} dt
Compute C^(1) = A X_k^(1) + X_k^(1) A^T + C and set Y^(1) = X_k^(1)
while (‖C^(r)‖_2 > tol ‖C^(r-1)‖_2 & r < r_max) do
   r := r + 1
   X_0^(r) = ∫_0^h e^{tA} C^(r-1) e^{tA^T} dt
   for j = 1, ..., k do
      X_j^(r) = X_{j-1}^(r) + E_{j-1} X_{j-1}^(r) E_{j-1}^T
   end for
   Compute C^(r) = A X_k^(r) + X_k^(r) A^T + C^(r-1) and Y^(r) = Y^(r-1) + X_k^(r)
end while
```

The following proposition shows that the quadratic convergence occurs at every restart of Algorithm 4.

PROPOSITION 2.3. *With the notation of Algorithm 4, the approximate solution $Y^{(r)}$ obtained at restart r satisfies*

$$X - Y^{(r)} = \left(e^{2^k hA}\right)^r X \left(e^{2^k hA^T}\right)^r.$$

Proof. We have $Y^{(r)} = \sum_{p=1}^{r} X_k^{(p)}$ where

$$X_k^{(p)} = \int_0^{2^k h} e^{tA} C^{(p-1)} e^{tA^T} dt$$

and

$$C^{(p)} = A X_k^{(p)} + X_k^{(p)} A^T + C^{(p-1)} = e^{2^k hA} C^{(p-1)} e^{2^k hA^T}.$$

Hence, $X_k^{(p)} = \int_{2^k(p-1)h}^{2^k pk} e^{tA} C e^{tA^T} dt$. On the other hand, from (2.2) the solution X can be decomposed as

$$\begin{aligned} X &= \sum_{p=1}^{r} \int_{2^k(p-1)h}^{2^k pk} e^{tA} C e^{tA^T} dt + \int_{2^k ph}^{\infty} e^{tA} C e^{tA^T} dt \\ &= \sum_{p=1}^{r} X_k^{(p)} + \left(e^{2^k hA}\right)^r X \left(e^{2^k hA^T}\right)^r \\ &= Y^{(r)} + \left(e^{2^k hA}\right)^r X \left(e^{2^k hA^T}\right)^r. \end{aligned}$$

□

The importance of a restart is demonstrated in Figure 2.3, which compares Algorithms 1 and 4. The figure shows that while a stagnation of Algorithm 1 occurs at iteration 8 with a residual norm of order $2.25 \cdot 10^{-9}$, the residual norm produced by Algorithm 4 with $\mathrm{tol} = 10^{-12}$ decreases until iteration 14 (corresponding to 2 restarts) to a value of order $7.42 \cdot 10^{-15}$.

ETNA
Kent State University
http://etna.math.kent.edu

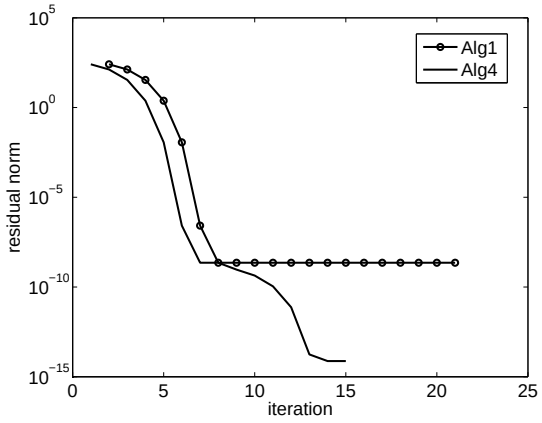

FIG. 2.3. *Convergence of Algorithms 1 and 4 (Example 1).*

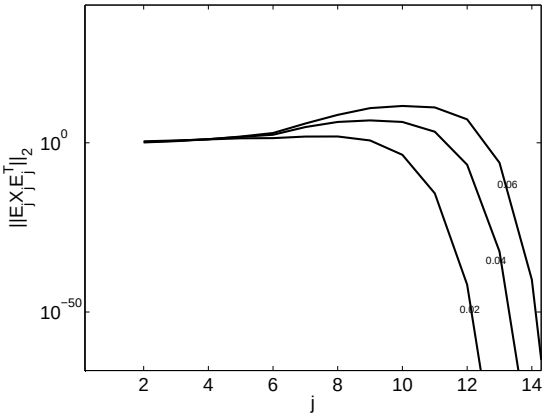

FIG. 2.4. *Behavior of* $\|E_j X_j E_j^T\|_2$ *(Example 2).*

TABLE 2.3
Norms of residuals and solutions computed by Algorithms 2, 3, 4, and lyap *(Example 2).*

α	0.02	0.04	0.06
$\|R_{\mathsf{alg2}}\|_2$	$2.34 \cdot 10^{-12}$	$1.43 \cdot 10^{-8}$	$9.34 \cdot 10^{-4}$
$\|R_{\mathsf{alg3}}\|_2$	$1.72 \cdot 10^{-12}$	$4.93 \cdot 10^{-8}$	$2.06 \cdot 10^{-3}$
$\|R_{\mathsf{alg4}}\|_2$	$3.03 \cdot 10^{-14}$	$6.61 \cdot 10^{-10}$	$1.03 \cdot 10^{-5}$
$\|R_{\mathsf{lyap}}\|_2$	$1.32 \cdot 10^{-13}$	$7.04 \cdot 10^{-9}$	$1.41 \cdot 10^{-4}$
$\|X_{\mathsf{lyap}}\|_2$	94.4	$4.69 \cdot 10^{6}$	$8.10 \cdot 10^{10}$

Example 2. We end this section with a comparison of Algorithms 2, 3, 4, and the function lyap when condition (2.7) is not satisfied and C is arbitrary. We consider the matrix $A = -2I + \alpha T$ of order $n = 500$, where T has all 1s on its strictly upper triangular part and is zero elsewhere, α is a parameter to be varied, and the entries of C are random in $(0, 1)$. For this example, the exponentials decay slowly as α gets larger (compare Figure 2.4 and Figure 2.1 (right)). The tolerance tol in Algorithms 3 and 4 is fixed at 10^{-12} and the maximum number of restarts $r_{\max}$ in Algorithm 4 is fixed at 5. Table 2.3 displays the norms of

ETNA
Kent State University
http://etna.math.kent.edu

the residuals and the norms of the computed solutions. All methods yield the same norms of the solutions. The norms of the relative residuals can be deduced from this table. The table shows that only a minor improvement over Algorithm 2 is obtained, and this improvement is generally due to Algorithm 4.

3. Conclusion. We have proposed modifications of the Davison-Man method for solving Lyapunov equations. We have analyzed the convergence properties of the proposed method. The modifications help prevent stagnation (which was the main limitation from which this method suffers) and improve significantly the computed solution. The extension of this work to Sylvester equations is straightforward.

REFERENCES

[1] R. H. BARTELS AND G. W. STEWART, *Algorithm* 432*: Solution of the matrix equation* $AX + XB = C$, Comm. ACM, 15 (1972), pp. 820–826.
[2] B. N. DATTA, *Numerical Methods in Linear Control Systems. Design and Analysis.*, Elsevier, San Diego, 2004.
[3] E. J. DAVISON AND F. T. MAN, *The numerical solution of* $A'Q + QA = -C$, IEEE Trans. Automatic Contol, AC-13 (1968), pp. 448–449.
[4] Z. GAJIČ AND M. T. J. QURESHI, *Lyapunov Matrix Equation in System Stability and Control*, Dover, Mineola, 2008.
[5] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, 1996.
[6] R. A. SMITH, *Matrix equation* $XA + BX = C$, SIAM J. Appl. Math., 165 (1968), pp. 198–201.