On two numerical methods for the solution of large-scale algebraic Riccati equations

Valeria Simoncini, Daniel B. Szyld
and Marlliny Monsalve

Report 12-11-26
November 2012. Revised February 2013[1]

This report is available in the World Wide Web at
`http://www.math.temple.edu/~szyld`

[1] Original title: On the numerical solution of large-scale Riccati equations

ON TWO NUMERICAL METHODS FOR THE SOLUTION OF LARGE-SCALE ALGEBRAIC RICCATI EQUATIONS[†]

VALERIA SIMONCINI[‡], DANIEL B. SZYLD[§], AND MARLLINY MONSALVE[¶]

Abstract. The inexact Newton-Kleinman method is an iterative scheme for numerically solving large scale algebraic Riccati equations. At each iteration, the approximate solution of a Lyapunov linear equation is required. Specifically designed projection of the Riccati equation onto an iteratively generated approximation space provides a possible alternative. Our numerical experiments with enriched approximation spaces seem to indicate that this latter approach is superior to Newton-type strategies on realistic problems, thus giving experimental ground for recent developments in this direction. As part of an explanation of why this is so, we derive several matrix relations between the iterates produced by the same projection approach applied to both the (quadratic) Riccati equation and its linear counterpart, the Lyapunov equation.

Key words. Riccati, Lyapunov, Kleinman-Newton, Krylov projection, residual norm computation

AMS subject classifications. 47J20, 65F30, 49M99, 49N35, 93B52

1. Introduction. We study the numerical solution of the following large-scale continuous-time algebraic Riccati equation[1],

$$AX + XA^T - XBB^TX + C^TC = 0, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times p}$, $C \in \mathbb{R}^{s\times n}$, with $p, s \ll n$, and A^T is the transpose of A. This type of equation appears in many areas of science and engineering, and in particular in control problems; see, e.g., [7], [9], [12], [13], [23], [33] and references therein. Throughout this paper we shall assume that A is passive. A matrix is said to be *stable* (resp., *passive*) if its spectrum (resp., field of values[2]) lies in the left-half plane; thus a passive matrix is always stable. A symmetric solution X to the Riccati equation (1.1) is called *stabilizing* if $A^T - BB^TX$ is stable.

Under the above assumptions, the sought after stabilizing solution is known to be symmetric and positive semidefinite. When n is large, one usually looks for a symmetric positive semidefinite approximation to the solution of low rank, in the form $ZZ^T \approx X$, so that only the tall matrix Z needs to be stored.

Several methods have been proposed for the numerical solution of (1.1); we refer, for instance, to the thorough presentation in the recent monograph [9]. In this paper we focus on methods for large matrices, for which storing the n^2 entries of the full solution is unrealistic. In this situation, the roster of practical methods is essentially reduced to two approaches[3].

[†]This version dated 5 February 2013

[‡]Dipartimento di Matematica, Università di Bologna, Piazza di Porta S. Donato, 5, I-40127 Bologna, Italy; and also CIRSA, Ravenna, Italy (valeria.simoncini@unibo.it).

[§]Department of Mathematics, Temple University (038-16), 1805 N. Broad Street, Philadelphia, Pennsylvania 19122-6094, USA (szyld@temple.edu). This research is supported in part by the U.S. National Science Foundation under grant DMS-1115520 and by the U.S. Department of Energy under grant DE-FG02-05ER25672.

[¶]Escuela de Computación, Apartado postal 47002, Facultad de Ciencias Caracas 1041-A, Universidad Central de Venezuela, Caracas, Venezuela, (marlliny.monsalve@.ciens.ucv.ve).

[1]The standard form of the algebraic equation is $A^TX + XA - XBB^TX + C^TC = 0$. Here we use A in place of A^T for ease of presentation when working with Krylov subspace methods.

[2]The field of values of an $n \times n$ matrix A is the set $W(A) = \{x^*Ax : x \in \mathbb{C}^n, x^*x = 1\}$, where x^* is the conjugate transpose of x.

[3]A third approach, that of using a doubling algorithm in a large setting [11] is still unproven.

One such approach is to use variants of Newton's method. As is well known, Newton's method exhibits locally quadratic convergence and each iteration requires the solution of an inner problem involving the Jacobian of the original function. In the case of the Riccati equation (1.1), this inner problem is a Lyapunov equation, i.e., an equation of the form

$$\mathcal{A}\mathcal{X} + \mathcal{X}\mathcal{A}^T + \mathcal{C}^T\mathcal{C} = 0, \tag{1.2}$$

for some appropriate $\mathcal{A} \in \mathbb{R}^{n\times n}$, $\mathcal{C} \in \mathbb{R}^{s\times n}$. These coefficient matrices change from one Newton step to the next; see, e.g., [7], [8], [21], [30], and section 4 for more details. When these linear equations are approximately solved, the overall approach takes the name of *inexact* Newton method. Known subtle ingredients of the inexact Newton method include the choice of the starting approximation, the selection of the inner solver for (1.2), and the derivation of a stopping criterion involving easily computable quantities. The Alternating Direction Implicit method (ADI) is a typical inner solution strategy, with an a-priori selection of the employed parameters [7], [8]. Instead, we experiment with recently developed and competitive projection-type methods as Lyapunov equation inner solvers [46], which do not require parameter tuning, and allow one to derive an extremely cheap expression for the true residual norm $\|\mathfrak{R}(ZZ^T)\|$, where

$$\mathfrak{R}(X) := AX + XA^T - XBB^TX + C^TC,$$

without explicitly computing the full residual matrix $\mathfrak{R}(ZZ^T)$; see section 4.1. We observed that a few steps of a projection-type solver for the linear companion Lyapunov equation $AX+XA^T+C^TC = 0$ may yield a particularly good starting approximation for the Newton iteration. Some matrix relations are derived to help justify this choice.

By mimicking the linear case, the other method we consider is the projection of the original Riccati equation (1.1) onto an appropriate subspace, the solution of the projected problem by imposing a Galerkin condition, and the expansion of the solution back to the whole space. We shall refer to this approach as the *Galerkin Projection* (GP) method. This method has been experimentally explored with the projection space being the extended Krylov subspace [27], [46]. Further details are given in section 2.

To the best of our knowledge, there is no work comparing these two practical approaches. In our numerical experiments, we have found that the projection method is superior when solving (1.1). Depending on the chosen approximation space, it converges faster (and often much faster) than the inexact Newton approach, and produces approximations of comparable rank; see section 5. While we do not have a full understanding of why this is so, we derived several equalities and bounds comparing the residuals corresponding to the Riccati and to the related Lyapunov approximate solutions indicating that in many cases, GP on the Riccati equation will converge faster than the corresponding Galerkin projection strategy applied to the Lyapunov equation (with the same coefficient matrices); see section 3.

A shortcoming of projection procedures such as GP is that it is not known in advance whether a *stabilizing* approximate solution can be obtained. In our experience projection methods indeed seem to produce stabilizing solutions, even though we do not have a way to guarantee this a priori. Nonetheless, we believe that the projection approach should in many cases be preferred in terms of computational performance and memory requirements over the inexact Newton method.

We end this introduction with additional definitions and notation.

We denote by $M \geq 0$ a positive semidefinite matrix M. We say that $M \geq N$ if $M - N \geq 0$. We denote by $\|M\|$ the matrix spectral norm induced by the vector 2-norm, i.e., $\|M\| = \sup_{\|v\|_2=1} \|Mv\|_2$, while $\|M\|_F$ stands for the Frobenius norm, i.e., $\|M\|_F^2 = \Sigma_{i,j}|m_{ij}|^2$, where m_{ij}'s are the elements of M. By I we denote the identity matrix.

2. Galerkin projection methods for Lyapunov and Riccati equations. In this section we review projection methods with a Galerkin condition applied to the residual. We do this first for the (linear) Lyapunov equation (1.2), and then show how this idea was also applied to Riccati equations.

Projection strategies reduce the problem dimension so that the reduced problem can be numerically solved with a method for small matrices. Saad [43] seems to have been the first to propose the idea of projecting the Lyapunov equation to a smaller space $\mathcal{S}_m$ of dimension $\mathcal{O}(m)$, solving the projected equation there, and expanding the solution to the larger space. Let V_m be a matrix whose orthonormal columns span $\mathcal{S}_m$, and consider approximate solutions of (1.2) of the form

$$\mathcal{X}_m = V_m \mathcal{Y}_m V_m^T. \tag{2.1}$$

We determine an approximation $\mathcal{Y}_m$ by requiring that the residual

$$\mathfrak{L}(\mathcal{X}) := \mathcal{A}\mathcal{X} + \mathcal{X}\mathcal{A}^T + \mathcal{C}^T\mathcal{C}$$

satisfy (Galerkin condition)

$$V_m^T \mathfrak{L}(\mathcal{X}_m) V_m = 0.$$

Such condition could be formally imposed, for instance, by using a Kronecker formulation of the problem [43]. Together with (2.1) one obtains the much smaller (projected) Lyapunov equation

$$\mathcal{T}_m \mathcal{Y} + \mathcal{Y}\mathcal{T}_m^T + \mathcal{C}_m^T \mathcal{C}_m = 0, \tag{2.2}$$

where the reduced matrices are

$$\mathcal{T}_m = V_m^T \mathcal{A} V_m \quad \text{and} \quad \mathcal{C}_m = \mathcal{C} V_m.$$

In [43], Saad used the standard block Krylov subspace

$$\mathcal{K}_m = \mathcal{K}_m(A, C^T) = \text{range}([C^T, AC^T, A^2C^T, \ldots, A^{m-1}C^T]), \tag{2.3}$$

and in this case V_m is $n \times ms$. Since then, other spaces have been used, including the extended Krylov subspace

$$\mathbb{K}_m = \mathbb{K}_m(A, C^T) = \mathcal{K}_m(A, C^T) + \mathcal{K}_m(A^{-1}, A^{-1}C^T), \tag{2.4}$$

which was first introduced in [15]. It was suggested for the solution of Lyapunov equations in [46], and its convergence analyzed in [16], [32]. In this case, V_m is $n \times 2ms$, assuming the space has maximum possible dimension.

A more general approximation space can be determined by allowing the solution of more general systems. For given parameters $\sigma_2, \ldots, \sigma_m \in \mathbb{C}$, this leads to the rational Krylov subspace[4]

$$\mathbf{K}_m = \text{range}([C^T, (A - \sigma_2 I)^{-1}C^T, \ldots, (A - \sigma_m I)^{-1}C^T]), \tag{2.5}$$

[4]Classical definitions replace the first instance of C^T with $(A - \sigma_1 I)^{-1}C^T$ for some given $\sigma_1 \in \mathbb{C}$ [42]. In the matrix equation context, including C^T appears to be computationally crucial; see, e.g., [16], [32].

whose dimension is not greater than ms. This class of subspaces was first introduced by Ruhe for solving eigenvalue problems [42], and it has been largely explored in the context of model order reduction, matrix function approximations and in the solution of matrix equations; see, e.g., [1], [17], [16], [24], [25], [26], [31], [40], and references therein. Certain solution strategies, such as the ADI method [36] (see, further [7, §4], [8] and references therein), also implicitly build rational Krylov subspaces [7], [34]. The selection of the parameters σ's is crucial for the quality of the approximation space. For the Lyapunov equation, values of σ's on the mirrored spectral region of A represent an effective selection; we refer to, e.g., [3], [45], for a more detailed analysis. The parameters may be either estimated a-priori [41], or computed adaptively as the generation of the space proceeds [18]. In our numerical experiments we shall employ this latter approach. We mention in passing that for the solution of large scale Lyapunov equations, in addition to the ones already mentioned, alternative approaches have been recently proposed, which address different important issues such as rank or data-sparsity optimization; see, e.g., [2], [22], [28], [47] and references therein.

Using a similar philosophy to that for Lyapunov, in [27] it was proposed to apply the same projection procedure with a Galerkin condition, directly to the Riccati equation (1.1) using the extended Krylov subspace as projection space. Obviously, the idea is more general, and may be applied to other approximation spaces, such as the rational Krylov subspace. We will include this choice in our experiments.

All Krylov subspaces discussed here contain the columns of C^T; see (2.3)–(2.5), and we assume for the rest of the paper that all columns of C^T are in $\mathcal{S}_m$. Thus, GP for Riccati equations can be described as follows. Let the orthonormal columns of the matrix V_m span a desired approximation space $\mathcal{S}_m$ of dimension $O(m)$, with $m \ll n$, so that $C^T = V_m C_m^T$ with $C_m^T = V_m^T C^T$, and let us build the reduced matrices $T_m = V_m^T A V_m$, $B_m = V_m^T B$. Then for an approximation $X_m = V_m Y_m V_m^T$, the Galerkin condition $V_m^T \mathfrak{R}(X_m) V_m = 0$ yields the projected (much smaller) Riccati equation

$$T_m Y + Y T_m^T - Y B_m B_m^T Y + C_m^T C_m = 0. \tag{2.6}$$

Since A is passive, T_m is passive and thus stable therefore, for $m > s$ and $m > p$, the projected Riccati equation (2.6) has a unique stabilizing solution. Let $Y_m^{(R)}$ be such a solution obtained by a Schur-decomposition based algorithm [9]; then, the approximation to the solution of (1.1) is determined as $X_m^{(R)} = V_m Y_m^{(R)} V_m^T$.

3. Matrix relations for the iterates of the Lyapunov and Riccati equations. In this section we show theoretical results comparing the projected approximate solution to the (quadratic) Riccati equation (1.1) with that of the (linear) Lyapunov equation obtained by setting $B = 0$ in (1.1). Our aim is to try to shed some light on the reasons why the two iterations often behave very similarly, as if the quadratic term did not have any influence. More precisely, in some cases the residual convergence history seems to be the same if one solves either (1.1) or the Lyapunov equation

$$AX + XA^T + C^T C = 0. \tag{3.1}$$

In this setting, the projected Lyapunov equation in the subspace spanned by the columns of V_m is

$$T_m Y + Y T_m^T + C_m^T C_m = 0, \tag{3.2}$$

with solution $Y_m^{(L)}$, which is expanded as $X_m^{(L)} = V_m Y_m^{(L)} V_m^T$.

We will provide some norm bounds on the difference between the residual of PG and that of the same projection method applied to the Lyapunov equation. Moreover, we will show that $\|\mathfrak{R}(X_m^{(R)})\|_F \leq \|\mathfrak{L}(X_m^{(R)})\|_F$, where $\mathfrak{L}$ is computed using $\mathcal{A} = A$ and $\mathcal{C} = C$, i.e., that the approximate solution $X_m^{(R)}$ of the Riccati equation gives a lower Riccati residual norm than Lyapunov residual norm, thus highlighting the role of the second order term; a more precise relation will actually be given.

We first focus our attention on the difference in residuals $\mathfrak{R}(X_m^{(R)}) - \mathfrak{L}(X_m^{(L)})$. We assume that the following relation holds

$$AV_m = V_m T_m + \tilde{v}_{m+1} t_m^T, \tag{3.3}$$

where $[V_m, \tilde{v}_{m+1}]$ has orthonormal columns and t_m is a matrix with the same number of columns as $\tilde{v}_{m+1}$. We remark that standard, extended and rational Krylov subspaces all satisfy (3.3) for different choices of $\tilde{v}_{m+1}$. For instance, for the standard block Krylov subspace, (3.3) holds for $t_m^T = t_{m+1,m} e_m^T$, where e_m is matrix containing the last s columns of the $ms \times ms$ identity matrix, and $t_{m+1,m}$ contains the orthogonalization coefficients to obtain the next block of vectors, $v_{m+1} = \tilde{v}_{m+1}$, from the block Arnoldi procedure [44]. We refer to [35] for the case of the Rational Krylov subspace.

With this notation, it is known that (see, e.g., [27], [46])

$$\|\mathfrak{R}(X_m^{(R)})\| = \|t_m^T Y_m^{(R)}\|, \quad \|\mathfrak{L}(X_m^{(L)})\| = \|t_m^T Y_m^{(L)}\|.$$

With the same technical tools as those used to derive the two equalities above, we can prove the following result for the norm of the difference $\mathfrak{R}(X_m^{(R)}) - \mathfrak{L}(X_m^{(L)})$.

PROPOSITION 3.1. *Let the orthonormal columns of V_m span $\mathcal{S}_m$, and be such that the Arnoldi relation* (3.3) *holds. Let $Y_m^{(R)}$ and $Y_m^{(L)}$ be the solutions of the projected Riccati equation* (2.6) *and projected Lyapunov equation* (3.2)*, respectively. Let $X_m^{(R)} = V_m Y_m^{(R)} V_m^T$ and $X_m^{(L)} = V_m Y_m^{(L)} V_m^T$. Then*

$$\|\mathfrak{R}(X_m^{(R)}) - \mathfrak{L}(X_m^{(L)})\|_* = c\|t_m^T (Y_m^{(R)} - Y_m^{(L)})\|_*, \tag{3.4}$$

where $c = 1$ for $\|\ \|_* = \|\ \|$ and $c = \sqrt{2}$ for $\|\ \|_* = \|\ \|_F$.

Proof. Using the fact that $C = V_m^T C_m$, the relation (3.3), and the fact that $Y_m^{(R)}$ solves (2.6) we have

$$\begin{aligned}
\mathfrak{R}(X_m^{(R)}) &= AV_m Y_m^{(R)} V_m^T + V_m Y_m^{(R)} V_m^T A^T - V_m Y_m^{(R)} V_m^T B B^T V_m Y_m^{(R)} V_m^T + C^T C \\
&= V_m T_m Y_m^{(R)} V_m^T + \tilde{v}_{m+1} t_m^T Y_m^{(R)} V_m^T + V_m Y_m^{(R)} T_m^T V_m^T \\
&\quad + V_m Y_m^{(R)} t_m \tilde{v}_{m+1}^T - V_m Y_m^{(R)} B_m B_m^T Y_m^{(R)} V_m^T + V_m C_m^T C_m V_m^T \\
&= \tilde{v}_{m+1} t_m^T Y_m^{(R)} V_m^T + V_m Y_m^{(R)} t_m \tilde{v}_{m+1}^T \\
&= \tilde{V}_{m+1} \begin{bmatrix} 0 & Y_m^{(R)} t_m \\ t_m^T Y_m^{(R)} & 0 \end{bmatrix} \tilde{V}_{m+1}^T
\end{aligned}$$

where $\tilde{V}_{m+1} = [V_m, \tilde{v}_{m+1}]$. Similarly, using that $Y_m^{(L)}$ solves (3.2), we have

$$\mathfrak{L}(X_m^{(L)}) = \tilde{V}_{m+1} \begin{bmatrix} 0 & Y_m^{(L)} t_m \\ t_m^T Y_m^{(L)} & 0 \end{bmatrix} \tilde{V}_{m+1}^T$$

Therefore, the result follows from writing

$$\mathfrak{R}(X_m^{(R)}) - \mathfrak{L}(X_m^{(L)}) = \tilde{V}_{m+1} \begin{bmatrix} 0 & (Y_m^{(R)} - Y_m^{(L)})t_m \\ t_m^T(Y_m^{(R)} - Y_m^{(L)}) & 0 \end{bmatrix} \tilde{V}_{m+1}^T. \quad \square$$

We note that the spectral norm is most used for theoretical and convergence analysis, but it is often hard to compute. On the other hand, the Frobenius norm is easier to compute, and it is used for the algorithms' stopping criteria; see further Remark 4.1 below.

We proceed now to compare the solutions to the two equations.

PROPOSITION 3.2. *Assume that T_m is stable and that the Riccati equation* (2.6) *has a Hermitian solution $Y_m^{(R)}$. Then the solution $Y_m^{(L)}$ to the Lyapunov equation* (3.2) *satisfies $Y_m^{(L)} \geq Y_m^{(R)}$. Moreover,*

$$\|Y_m^{(L)} - Y_m^{(R)}\| \leq \frac{1}{2\alpha}\|B_m^T Y_m^{(R)}\|^2,$$

where $-\alpha$ is the largest eigenvalue of $(T_m + T_m^T)/2$, i.e., the rightmost point of the field of values of T_m.

Proof. Since $Y_m^{(L)}$ is the solution of the Lyapunov equation (3.2), we can write it formally as (see, e.g., [33])

$$Y_m^{(L)} = \int_0^\infty e^{tT_m} C_m^T C_m e^{tT_m^T} dt \ .$$

Similarly, we can write $Y_m^{(R)}$ as the solution of the "Lyapunov" equation (2.6), that is

$$Y_m^{(R)} = \int_0^\infty e^{tT_m}(-Y_m^{(R)} B_m B_m^T Y_m^{(R)} + C_m^T C_m) e^{tT_m^T} dt \ .$$

Therefore

$$M := Y_m^{(L)} - Y_m^{(R)} = \int_0^\infty e^{tT_m} Y_m^{(R)} B_m B_m^T Y_m^{(R)} e^{tT_m^T} dt \ \geq \ 0. \tag{3.5}$$

This proves the first assertion. In addition, we have

$$\begin{aligned} \|M\| &\leq \int_0^\infty \|e^{tT_m}\|^2 \|Y_m^{(R)} B_m B_m^T Y_m^{(R)}\| dt \\ &\leq \int_0^\infty e^{-2\alpha t} dt \|B_m^T Y_m^{(R)}\|^2 = \frac{1}{2\alpha}\|B_m^T Y_m^{(R)}\|^2 \ , \end{aligned}$$

where the last inequality follows, e.g., from [12, Lemma 3.2.1]. $\square$

From Proposition 3.2 it also follows that $X_m^{(L)} \geq X_m^{(R)}$. We also note that these results can also be obtained by reworking [33, Corollary 9.1.6] to our notation and assumptions. Observe also that (3.5) provides an explicit formula for the difference of the two solutions, which we can use to derive a bound on the norm of this difference, as well as on the norm of the difference on the residuals.

We turn now to some bounds on the norm of the difference between the residuals.

PROPOSITION 3.3. *Let the hypotheses and notation of Propositions 3.1 and 3.2 hold. Then*

$$\frac{\|B_m^T Y_m^{(R)} t_m\|^2}{2\|T_m^T t_m\|} \leq \|\mathfrak{R}(X_m^{(R)}) - \mathfrak{L}(X_m^{(L)})\| \leq \frac{1}{2\alpha}\|B_m^T Y_m^{(R)}\|^2 \|t_m\|.$$

Proof. The upper bound follows directly from (3.4) and Proposition 3.2. From (2.6) and (3.2) we have that $M = Y_m^{(L)} - Y_m^{(R)}$ satisfies

$$T_m M + M T_m^T + Y_m^{(R)} B_m B_m^T Y_m^{(R)} = 0.$$

Let u be any real vector of unit norm with number of components matching the number of columns of t_m, and let $w = t_m u$. Then

$$w^T (T_m M + M T_m^T + Y_m^{(R)} B_m B_m^T Y_m^{(R)}) w = 0,$$

and thus

$$2 w^T T_m M w + w^T Y_m^{(R)} B_m B_m^T Y_m^{(R)} w = 0. \tag{3.6}$$

Since $w^T T_m M w \geq -\|Mw\| \|T_m^T w\| \geq -\|M t_m\| \|T_m^T t_m\|$, in (3.6) we obtain

$$-2\|M t_m\| \|T_m^T t_m\| + \|B_m^T Y_m^{(R)} t_m u\|^2 \leq 0,$$

for all u of unit norm of appropriate dimension. This inequality holds in particular for the vector u for which $\|B_m^T Y_m^{(R)} t_m u\| = \|B_m^T Y_m^{(R)} t_m\|$, and thus

$$\|M t_m\| \geq \frac{\|B_m^T Y_m^{(R)} t_m\|^2}{2\|T_m^T t_m\|} \ .$$

The result follows from Proposition 3.1. □

As is to be expected, the bounds above depend on the action of B (which defines the quadratic term); cf. further comments below. Our next result relates the norm of the residual of the Riccati equation with that of the Lyapunov equation.

PROPOSITION 3.4. *Let V_m be the matrix whose columns span a given approximation space $\mathcal{S}_m$. Let $X_m^{(R)} = V_m Y_m^{(R)} V_m^T$, where $Y_m^{(R)}$ solves the projected Riccati equation* (2.6) *when using a Galerkin condition on $\mathcal{S}_m$. Then*

$$\|\mathfrak{R}(X_m^{(R)})\|_F^2 = \|\mathfrak{L}(X_m^{(R)})\|_F^2 - \|X_m^{(R)} B B^T X_m^{(R)}\|_F^2. \tag{3.7}$$

Proof. For simplicity, we drop the subscript m. Consider the residual matrix $\mathfrak{R}(X^{(R)}) = A X^{(R)} + X^{(R)} A^T + C^T C - X^{(R)} B B^T X^{(R)}$, that is,

$$\mathfrak{L}(X^{(R)}) = \mathfrak{R}(X^{(R)}) + X^{(R)} B B^T X^{(R)}.$$

From the Galerkin condition, we have that the residual $\mathfrak{R}(X^{(R)})$ is orthogonal (in the Frobenius inner product) to the space range(V_m), so that (3.7) follows. □

We note that $\|X_m^{(R)} B B^T X_m^{(R)}\|_F \leq \|\mathfrak{L}(X_m^{(R)})\|_F$ and $\|\mathfrak{R}(X_m^{(R)})\|_F \leq \|\mathfrak{L}(X_m^{(R)})\|_F$, so that, in particular, $X_m^{(R)}$ cannot be close to the solution of the Lyapunov equation, unless $\|X_m^{(R)} B B^T X_m^{(R)}\|_F$ is small. Of course, $\|B^T X^{(R)}\|_F$ is small if either $\|B\|_F$ or $\|X^{(R)}\|_F$ is small, or if the space range(V_m) is (almost) orthogonal to the columns of B. This implies that it may be worth balancing the magnitude of B, relative to A and C, as was done in [5].

4. Newton's method. The version of Newton's method commonly used for the solution of the continuous-time algebraic Riccati equation (CARE) (1.1) was introduced by Kleinman [30], and it is described in Algorithm 1.

Algorithm 1 Newton's method for CARE: Kleinman iteration

Given $X_0 \in \mathbb{R}^{n\times n}$ such that $X_0 = X_0^T$, $A^T - BB^T X_0$ is stable.
For $k = 0, 1, \ldots,$ until convergence
  Set $\mathcal{A}_k^T = A^T - BB^T X_k$
  Set $\mathcal{C}_k^T = [X_k B \;\; C^T]$
  Solve $\mathcal{A}_k X_{k+1} + X_{k+1}\mathcal{A}_k^T + \mathcal{C}_k^T \mathcal{C}_k = 0$

As a fixed point iteration, the approximation at the kth step of Newton's method, can be obtained as the solution of the linear system given by the Lyapunov equation (1.2) with $\mathcal{A}_k^T = A^T - BB^T X_k$ and $\mathcal{C}_k = [B^T X_k,\; C]$.

A traditional approach has been to solve each of these Lyapunov equations with the ADI method with a sufficiently high and fixed accuracy so as to guarantee that the Newton iteration converges to the stabilizing solution; see, e.g., [7, §4], [8] and references therein.

An alternative is to use an inexact Newton method [14]. This consists of approximating the solution of the Lyapunov equation in the Newton step to a certain tolerance η_k, which is decreasing with k. More precisely, one needs to compute a new approximation $X_{k+1} = \mathcal{X}_{k+1}^{(L)}$ to the solution of (1.2) such that

$$\|\mathfrak{L}(X_{k+1})\| \leq \eta_k \|\mathfrak{R}(X_k)\|. \tag{4.1}$$

A study of different choices for this sequence of tolerances can be found in [20]. In [14], [20], it is shown that with such a sequence, the convergence rate of the inexact Newton method is similar to that of the exact Newton method. We mention that it was shown already in [30] that the sequence $X_k = X_k^{(R)}$ produced by Algorithm 1 is monotone in the sense that $X_k^{(R)} \geq X_{k+1}^{(R)}$. In [21] the use of inexact Newton for Riccati equations is analyzed, and it is shown that this monotonicity property is maintained in the inexact case.

We summarize the inexact Newton method for Riccati equations in Algorithm 2. As mentioned in section 2, in this paper we use direct projection with extended or rational Krylov subspaces as inner Lyapunov equation solver, following the experience and analysis in [16], [18], [46].

Algorithm 2 Inexact Newton method for CARE: Kleinman iteration

Given $X_0 \in \mathbb{R}^{n\times n}$, such that $X_0 = X_0^T$, $A^T - BB^T X_0$ is stable,
  and a sequence of decreasing $\eta_k \in [0, 1)$
For $k = 0, 1, \ldots,$ until convergence
  Set $\mathcal{A}_k^T = A^T - BB^T X_k$
  Set $\mathcal{C}_k^T = [X_k B \;\; C^T]$
  Approximately Solve $\mathcal{A}_k X_{k+1} + X_{k+1}\mathcal{A}_k^T + \mathcal{C}_k^T \mathcal{C}_k = 0$ for X_{k+1}
  such that (4.1) holds

Comparing Algorithm 2 with Algorithm 1, and assuming that the same convergence rate translates in approximately the same number of Newton steps, one can see that savings could be obtained since the computational cost of loosely solving the inner Lyapunov equation may be significantly lower than what would be required by a very stringent stopping tolerance; see [21], and the numerical results in [8], [38].

We also recall two key implementation details from the above mentioned references. The first one is that $\mathcal{A}_k$ is not explicitly computed, instead, matrix-vector

products and solves are done efficiently (see further section 5). The second one is that X_{k+1} should not be stored explicitly: a solution strategy should be used in step 5 that directly yields the low column rank factor Z of $X_{k+1} = ZZ^T$, so that only Z is used throughout whenever matrix vector products with X_{k+1} are required.

4.1. Practical Computation of the Residual. Estimates of the residual norm $\|\mathfrak{R}(X)\|$ are needed, for example, to assess if the method has converged to a given tolerance, or in the case of inexact Newton, to choose the tolerance for the approximate solution of the associated Lyapunov equation (step 5 in Algorithm 2). Computing the residual $\mathfrak{R}(X)$ is expensive and the matrix itself impossible to store in the large scale setting as a generally dense matrix. Thus, alternative estimates either for the matrix itself or more directly for its norm are needed. We refer to [7, §4.2], [46, §3] for discussions on several approaches to this issue.

In this section, we present estimates together with computationally cheap exact expressions for $\|\mathfrak{R}(X_k)\|_F$, i.e., for the Frobenius norm of the residual in Algorithm 2.

REMARK 4.1. The usual rate of convergence results for inexact Newton use the spectral norm of the matrix, and this holds in particular for the condition (4.1). In practical computations, the Frobenius norm is used. The two norms are equivalent, as for any square matrix M it holds that $\|M\| \leq \|M\|_F \leq \sqrt{n}\|M\|$. Therefore, when implementing Algorithm 2, we use the stopping criterion $\|\mathfrak{L}(X_{k+1})\|_F \leq (\eta_k/\sqrt{n})\,\|\mathfrak{R}(X_k)\|_F$, which implies (4.1).

We begin with an equality relating the residual $\mathfrak{R}(X_k)$ with $\mathfrak{L}(X_k)$, the residual of the inner Lyapunov equation.[5]

PROPOSITION 4.2. *Let X_k be the kth iterate produced by Algorithm 2. Then*

$$\mathfrak{R}(X_{k+1}) = \mathfrak{L}(X_{k+1}) - (X_kB - X_{k+1}B)(X_kB - X_{k+1}B)^T. \tag{4.2}$$

Proof. We have that

$$\begin{aligned}
\mathfrak{L}(X_{k+1}) &= \mathcal{A}_k X_{k+1} + X_{k+1}\mathcal{A}_k^T + \mathcal{C}_k^T\mathcal{C}_k \\
&= (A - X_kBB^T)X_{k+1} + X_{k+1}(A^T - BB^TX_k) + C^TC + X_kBB^TX_k \\
&= AX_{k+1} - X_kBB^TX_{k+1} + X_{k+1}A^T - X_{k+1}BB^TX_k + C^TC + X_kBB^TX_k \\
&= \mathfrak{R}(X_{k+1}) + X_{k+1}BB^TX_{k+1} - X_kBBX_{k+1} - X_{k+1}BB^TX_k + X_kBB^TX_k \\
&= \mathfrak{R}(X_{k+1}) + (X_k - X_{k+1})BB^T(X_k - X_{k+1})^T,
\end{aligned}$$

from which the result follows. ∎

From (4.2) one can directly obtain the bound

$$\begin{aligned}
\|\mathfrak{R}(X_{k+1})\|_F &\leq \|\mathfrak{L}(X_{k+1})\|_F + \|(X_kB - X_{k+1}B)(X_kB - X_{k+1}B)^T\|_F \\
&\leq \|\mathfrak{L}(X_{k+1})\|_F + \|X_kB - X_{k+1}B\|_F^2.
\end{aligned} \tag{4.3}$$

This bound is general, and it may be employed for any given approximate solution to the inner linear equation.

When using a Galerkin projection-type method for solving the inner Lyapunov equation, this result can be significantly improved, and the *exact* residual norm can

[5] This result was obtained independently by Jens Saak, with a different proof, and reported in presentations at the Workshop on Matrix Equations and Tensor Techniques, Aachen, 21-22 November 2011, and at the SIAM Conference on Applied Linear Algebra, Valencia, 18-22 June 2012.

be obtained almost for free. We first show that the inequality in (4.3) can be replaced with a related equality.

PROPOSITION 4.3. *Let X_k be the kth iterate produced by Algorithm 2, where the Lyapunov equations are solved by Galerkin projection. Then*

$$\|\mathfrak{R}(X_{k+1})\|_F^2 = \|\mathfrak{L}(X_{k+1})\|_F^2 + \|(X_kB - X_{k+1}B)(X_kB - X_{k+1}B)^T\|_F^2. \tag{4.4}$$

Proof. The proof proceeds in the same manner as that of Proposition 3.4, using the fact that X_k and X_{k+1} are both expressed in terms of spaces that satisfy the orthogonal condition with $\mathfrak{R}(X_{k+1})$. ∎

We end this section showing a convenient practical way to compute the norm of the residual $\|\mathfrak{R}(X_{k+1})\|_F^2$ when the inner solver is a projection method containing the columns of $\mathcal{C}_k^T$.

PROPOSITION 4.4. *Let $X_k = V_m\mathcal{Y}_m^{(L)}V_m^T$ be the kth iterate produced by Algorithm 2, obtained by solving the inner Lyapunov equation by means of a projection onto range(V_m) and Galerkin condition on the Lyapunov residual. Let $X_kB = \widehat{V}_1R$ be the skinny QR decomposition of X_kB, and let $G = [R^T\ 0\quad 0\ \cdots\ 0]$ be such that $X_kB = V_mG^T$. Then*

$$\|\mathfrak{R}(X_{k+1})\|_F^2 = \|\mathfrak{L}(X_{k+1})\|_F^2 + \|(G^T - \mathcal{Y}_m^{(L)}V_m^TB)(G^T - \mathcal{Y}_m^{(L)}V_m^TB)^T\|_F^2. \tag{4.5}$$

Proof. Using $X_kB = V_mG^T$ and the representation $X_{k+1} = V_m\mathcal{Y}_m^{(L)}V_m^T$, the term $X_kB - X_{k+1}B$ can be written as

$$X_kB - X_{k+1}B = V_m(G^T - \mathcal{Y}_m^{(L)}V_m^TB). \tag{4.6}$$

Thus, the result follows from using (4.6) and (4.4). ∎

The equality (4.5) shows that it is possible to compute the Frobenius norm of the Riccati residual by using the inner Lyapunov residual norm, together with the Frobenius norm of a small size matrix. The latter norm can also be used directly for the computation of $\|\mathfrak{R}(X_{k+1})\|_F^2 - \|\mathfrak{L}(X_{k+1})\|_F^2$, which we use in the computation of η_{k+1}; see section 5.

REMARK 4.5. The equality (4.5) provides the *exact* value of the Riccati residual norm in exact arithmetic, and not an estimate. In our numerical experiments, the computed values of the right-hand side of (4.5) always provided the value of the residual norm with an accuracy of at least five significant digits.

The identity (4.6) also allows us to replace the inequality (4.3) with the computable bound

$$\|\mathfrak{R}(X_{k+1})\|_F \le \|\mathfrak{L}(X_{k+1})\|_F + \|G^T - \mathcal{Y}_m^{(L)}V_m^TB\|_F^2.$$

5. Numerical Experiments. In this section we report on our numerical experience with the Galerkin projection procedures and with the inexact Newton method. All reported experiments were performed using Matlab 7.13 (R2011b) using the Control System Toolbox for calls to `lyap` and `care` (ver. 9.2) on a machine running Linux with 8Gb memory and a 64-Bit server.

Table 5.1 provides some information on the considered data, including the norm of the involved matrices. The last matrix stems from the finite difference discretization of the 3D Laplace operator on the unit cube, with homogeneous Dirichlet boundary conditions. When not available from the dataset, we used distinct matrices B and C

TABLE 5.1
Relevant information on the data used in our numerical experiments.

Name (Reference)	n	p	s	$\|A\|_F$	$\|B\|_F$	$\|C\|_F$
SP_LAPLACE ([27, Example 4])	6400	4	1	357	159	80
CHIP ([4])	20082	1	5	$2.2\,10^6$	$1.1\,10^5$	2.23
FLOW ([4])	9669	1	5	$4.5\,10^6$	$3.1\,10^5$	2.24
NONSYM ([10]; [39, Section 3.2])	10000	20	20	$4.4\,10^6$	$4.46\,10^2$	$4.47\,10^2$
3DLAPLACE	125000	5	5	$2.3\,10^3$	$7.9\,10^2$	$7.9\,10^2$
3DLAPLACE	125000	20	20	$2.3\,10^3$	$1.6\,10^3$	$1.5\,10^3$

with normally distributed random entries. The relative residual Frobenius norm was considered in our stopping criterion, with the norm of the residual computed efficiently as discussed in the previous sections; 10^{-8} was used as stopping tolerance. As stopping criterion for the inner solver in inexact Newton, we used the sequence of tolerances recommended in [20], namely $\eta_k = |\,||\mathfrak{L}(X_k)||_F - ||\mathfrak{R}(X_k)||_F|/||\mathfrak{R}(X_k)||_F$. When the GP methods were employed, the projected problem was solved every 5 iterations, to limit computations. The same applies to the Lyapunov solvers within the inexact Newton iteration. In all tables, the names GP-EKSM or GP-RKSM were employed to distinguish between the use of the extended or rational Krylov subspaces as approximation space, respectively.

In Newton's method, the inner Lyapunov equation at each iteration is solved by means of the extended Krylov subspace method (EKSM), as described and implemented in [46]. The coefficient matrix in the inner equation is given by $\mathcal{A}_k = A - X_k BB^T$, where X_k is the current approximate solution to the Riccati equation. Systems with this matrix are solved with the Sherman-Morrison-Woodbury formula, so that only systems with A need be solved. To this end, and LU factorization of A is computed at the beginning of the process, so that only back solves are required during all calls to EKSM. On the other hand, were we to use rational Krylov within Newton's method, we would have to solve with $A - X_k BB^T - \sigma_i I = (A - \sigma_i I) - X_k BB^T$. The Sherman-Morrison-Woodbury formula could still be applied, so as to solve only with the sparse matrix $A - \sigma_i I$; see, e.g., [6] for similar considerations. However, σ_i would change at each iteration of the rational Krylov method, therefore requiring a larger amount of computational effort compared with EKSM. This motivated us to discard general rational Krylov subspace methods as inner solver. For the same reasons, we decided not to use other methods, such as ADI-based approaches, that would also need the solution of different shifted systems at each inner iteration. We have also run experiments using as a Lyapunov solver the global Arnoldi method proposed in [29], and have found that this method is slower than the others considered here.

As is well-known the choice of the starting approximation is a bottleneck of Newton-type procedures. We tested the inexact Newton method for two choices of starting approximation X_0: the zero matrix, and the result of applying a fixed number (four) of iterations of GP with EKSM (reported as X_4^{eksm} in the tables). The latter selection was in most cases more effective than the former. We note that while $X_0 = 0$ is stabilizable as an initial matrix, there is no guarantee that $X_0 = X_4^{eksm}$ is, but in practice this caused no problems.

In the context of GP methods, EKSM was implemented as in [46], where only the inner call to the Matlab `care` function substituted that to the `lyap` function; the computation of the residual norm remained unchanged. To solve systems with A, the

TABLE 5.2
Performance of Riccati equation solvers for problem SP_LAPLACE. $(n, p, s) = (6400, 4, 1)$.

	its	inner its	time	space dim	rank(X_f)
Newton $X_0 = 0$	18	5, ..., 5	1.17	50	25
Newton $X_0 = X_4^{eksm}$	10	5, ..., 10	0.77	200	38
GP-EKSM	15		0.17	30	23
GP-RKSM	5		1.14	25	24

sparse matrix was factorized once for all at the beginning. RKSM was implemented with an adaptive selection of the shifts, as done in [18], and multiple columns in C^T were treated by simply using a block form of the algorithm. Once again, the only main change was the substitution of `lyap` with the `care` function for solving the projected problem. Each multiple right-hand side shifted system with $A - \sigma_i I$ was solved using a sparse direct solver (backslash in Matlab [37]).

TABLE 5.3
Performance of Riccati equation solvers for problem CHIP. $(n, p, s) = (20082, 1, 5)$.

	its	inner its	time	space dim	rank(X_f)
Newton $X_0 = 0$	11	5,..., 20	60.66	240	66
Newton $X_0 = X_4^{eksm}$	3	5, 10, 20	33.86	240	66
GP-EKSM	20		22.26	200	66
GP-RKSM	20		28.70	100	67

Tables 5.2 to 5.7 report the CPU time required to achieve convergence, the maximum space dimension employed by the method (for inexact Newton, it is the maximum space dimension generated in the inner iterations), and the rank of the final approximate solution X_f. For completeness, we also report the number of inner iterations and the number of outer iterations, although the latter has a rather different meaning for the inexact Newton method and for the GP methods.

TABLE 5.4
Performance of Riccati equation solvers for problem FLOW. $(n, p, s) = (9669, 1, 5)$.

	its	inner its	time	space dim	rank(X_f)
Newton $X_0 = 0$	11	5, ..., 25	9.18	300	69
Newton $X_0 = X_4^{eksm}$	3	5, 20, 25	4.72	300	70
GP-EKSM	25		2.16	250	69
GP-RKSM	30		4.46	150	69

All tables lead to similar considerations. More precisely, the GP methods display significantly lower CPU times and lower memory requirements (the latter in all cases, except for 3DLAPLACE).

All methods delivered an approximate solution of comparable rank, where the final numerical rank was obtained by a truncated SVD of the approximate solution factor. It is interesting to note, however, that because of the intrinsic nature of the inner-outer Newton process, much more memory is required during the inner step. On the other hand, when a GP method is employed, the solution rank is usually closer to the actual space dimension. We also mention that RKSM, though less efficient than EKSM in most cases, consistently provides better memory allocations, as already

noticed in [18] for the Lyapunov equation.

TABLE 5.5
Performance of Riccati equation solvers for problem NONSYM. $(n, p, s) = (10000, 20, 20)$.

	its	inner its	time	space dim	rank(X_f)
Newton $X_0 = 0$	3	1, 5, 15	28.08	1200	374
Newton $X_0 = X_4^{eksm}$	3	5, 10, 15	28.46	1200	374
GP-EKSM	15		7.21	600	374
GP-RKSM	15		6.11	300	300

To give an idea of the different workload between the Newton and the GP methods, we notice that in Table 5.2, with $s = 1$ and $p = 4$, Newton's method with $X_0 = 0$, requires a total of $18 \cdot 25$ matrix solves with A (each inner EKSM method starts with 5 columns and builds a space of size 50, requiring 25 matrix solves). On the other hand, EKSM applied to the original problem requires only 15 solves with A. Slightly less favorable is the case when $s > 1$. Using the results in Table 5.5, we see that Newton's method with $X_0 = 0$ requires a total of $40 + 40 \cdot 5 + 40 \cdot 15$ solves with A, whereas the direct projection EKSM method requires only $20 \cdot 15$ solves. Clearly, various intermediate figures are obtained as p and s vary.

We notice that using a few iterations of EKSM as initialization for the Newton iteration significantly improves the CPU performance of the approach. On the other hand, such choice may increase the rank of the known term in the first inner iteration. This explains the larger memory requirements in Table 5.2, compared with the choice $X_0 = 0$, for which the first inner solve was very memory efficient, as the process started with the single vector corresponding to $s = 1$.

Finally, we remark that all methods appear to converge to the same solution matrix. An explicit computation for the smallest case, cf. Table 5.2, revealed that the relative difference in the Frobenius norm between the Newton approximate solution with zero initial matrix and the other approximations was always around 10^{-5}, for the considered stopping tolerance.

TABLE 5.6
Performance of Riccati equation solvers for problem 3DLAPLACE. $(n, p, s) = (125000, 5, 5)$.

	its	inner its	time	space dim	rank(X_f)
Newton $X_0 = 0$	15	5, ..., 5	808	100	95
Newton $X_0 = X_4^{eksm}$	10	5, ..., 5	706	100	94
GP-EKSM	20		531	200	105
GP-RKSM	25		524	125	105

TABLE 5.7
Performance of Riccati equation solvers for problem 3DLAPLACE. $(n, p, s) = (125000, 20, 20)$.

	its	inner its	time	space dim	rank(X_f)
Newton $X_0 = 0$	19	5, ..., 5	2332	400	346
Newton $X_0 = X_4^{eksm}$	15	5, ..., 5	2042	400	347
GP-EKSM	15		622	600	364
GP-RKSM	20		720	400	358

6. Conclusions. We have compared experimentally the two classes of methods which have been shown to be effective for the numerical solution of large scale Riccati equations: Inexact Newton-Kleinman, and direct projection onto a Krylov-type subspace, with a Galerkin condition. Our experiments indicate that the Galerkin projection approach using extended or rational Krylov subspaces produces approximations to the solution of the Riccati equations using less computational resources, both in terms of CPU time and storage. We proved some relations between the Riccati and the Lyapunov iterates, which provide some preliminary insights into the understanding of why the direct projection method works better than a Newton approach, where a Lyapunov equation is solved at each Newton step.

Finally, we mention that if memory is a serious concern, then further improvements may be achieved by replacing the rational Krylov subspace with a tangential-type space, as those recently developed in [19].

Acknowledgment. We thank the two anonymous referees for their insightful comments, which helped improved our presentation.

REFERENCES

[1] Athanasios C. Antoulas. *Approximation of large-scale Dynamical Systems*. Advances in Design and Control. SIAM, Philadelphia, 2005.

[2] Ulrike Baur. *Control-Oriented Model Reduction for Parabolic Systems*. Ph.D. thesis, Institut für Mathematik, TU Berlin, 2008.

[3] Bernhard Beckermann and Lothar Reichel. Error estimation and evaluation of matrix functions via the Faber transform. *SIAM Journal on Numerical Analysis*, 47:3849–3883, 2009.

[4] Benchmark Collection. Oberwolfach model reduction benchmark collection, 2003. `http://www.imtek.de/simulation/benchmark`.

[5] Peter Benner. Symplectic balancing of Hamiltonian matrices. *SIAM Journal on Scientific Computing*, 22:1885–1904, 2001.

[6] Peter Benner. Solving large-scale control problems. *IEEE Control Systems Magazine*, 24:44–59, 2004.

[7] Peter Benner, Jing-Rebecca Li, and Thilo Penzl. Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratric optimal control problems. *Numerical Linear Algebra with Applications*, 15:755–777, 2008.

[8] Peter Benner and Jens Saak. A Galerkin-Newton-ADI method for solving large-scale algebraic Riccati equations. Technical Report SPP1253-090, DFG Priority Programme 1253 Optimization with Partial Differential Equations, Erlangen, Germany, January 2010.

[9] Dario A. Bini, Bruno Iannazzo, and Beatrice Meini. *Numerical Solution of Algebraic Riccati Equations*. SIAM, Philadelphia, 2012.

[10] Younes Chahlaoui and Paul Van Dooren. A collection of benchmark examples for model reduction of linear time invariant dynamical systems. Technical Report SLICOT Working Note 2002-2, 2002. Available at `http://www.icm.tu-bs.de/NICONET/benchmodred.html`.

[11] Eric King-wah Chu. Solving large-scale algebraic Riccati equations by doubling. Talk presented at the Seventeenth Conference of the International Linear Algebra Society, Braunschweig, Germany, 22 August 2011.

[12] Martin J. Corless and Arthur D. Frazho. *Linear Systems and Control, An Operator Perspective*, volume 254 of *Monographs and Textbooks in Pure and Applied Mathematics*. Marcel Dekker, New York and Basel, 2003.

[13] Biswa N. Datta. *Numerical Methods for Linear Control Systems Design and Analysis*. Elsevier Academic Press, Amsterdam, 2003.

[14] Ron S. Dembo, Stanley C. Eisenstat, and Trond Steihaug. Inexact Newton methods. *SIAM Journal on Numerical Analysis*, 19:400–408, 1982.

[15] Vladimir Druskin and Leonid Knizhnerman. Extended Krylov subspaces: Approximation of the matrix square root and relates functions. *SIAM Journal on Matrix Analysis and Applications*, 19:755–771, 1998.

[16] Vladimir Druskin, Leonid Knizhnerman, and Valeria Simoncini. Analysis of the rational Krylov subspace and ADI methods for solving the Lyapunov equation. *SIAM Journal on Numerical Analysis*, 49:1875–1898, 2011.

[17] Vladimir Druskin, Leonid Knizhnerman, and Mikhail Zaslavsky. Solution of large scale evolutionary problems using rational Krylov subspace optimized shifts. *SIAM Journal on Scientific and Statistical Computing*, 31:3760–3780, 2009.

[18] Vladimir Druskin and Valeria Simoncini. Adaptive rational Krylov subspaces for large-scale dynamical systems. *Systems and Control Letters*, 60:546–560, 2011.

[19] Vladimir Druskin, Valeria Simoncini, and Mikhail Zaslavsky. Adaptive tangential interpolation in rational Krylov subspaces for MIMO model reduction data. Preprint, November 2012.

[20] Stanley C. Eisenstat and Homer F. Walker. Choosing the forcing terms in inexact Newton method. *SIAM Journal on Scientific Computing*, 17:16–32, 1996.

[21] F. Feitzinger, Timo Hylla, and Ekkehard W. Sachs. Inexact Kleiman-Newton method for Riccati equations. *SIAM Journal on Matrix Analysis and Applications*, 31:272–288, 2009.

[22] Matthias Bollhöfer and André K. Eppler. A structure preserving FGMRES method for solving large Lyapunov equations. In M. Günther, A. Bartel, M. Brunk, M. Schöps, and S. Striebel, editors, *Progress in Industrial Mathematics at ECMI 2010, Mathematics in Industry*, vol. 17, pp. 131–136. Springer, Heidelberg, 2012.

[23] Zoran Gajić and Muhammas Tahir Javed Qureshi. *Lyapunov Matrix Equation in System Stabiliy and Control*, volume 195 of *Mathematics in Science and Engineering*. Associated Press, San Diego and London, 1995.

[24] Kyle Gallivan, Eric Grimme, and Paul Van Dooren. A rational Lanczos algorithm for model reduction. *Numerical Algorithms*, 12:33–63, 1996.

[25] Serkan Gugercin, Athanasios C. Antoulas, and Cchristopher Beattie. $\mathcal{H}_2$ model reduction for large-scale linear dynamical systems. *SIAM Journal on Matrix Analysis and Applications*, 30:609–638, 2008.

[26] Stefan Güttel. *Rational Krylov Methods for Operator Functions*. PhD thesis, TU Bergakademie Freiberg, Germany, 2010.

[27] Mohamed Heyouni and Khalid Jbilou. An extended block Arnoldi algorithm for large-scale solutions of the continuous-time algebraic Riccati equation. *Electronic Transactions on Numerical Analysis*, 33:53–62, 2009.

[28] Alan Scottedward Hodel and Kameshwar Poolla. Numerical solution of very large, sparse Lyapunov equations through approximate power iteration. In *Proceedings of the 29th Conference on Decision and Control* (Honolulu, Hawaii), pages 291–296. WA12. IEEE, December 1990.

[29] Khalid Jbilou. Low rank approximate solutions to large Sylvester matrix equations. *Applied Mathematics and Computation*, 177:365–376, 2006.

[30] David L. Kleinman. On an iterative technique for Riccati equation computations. *IEEE Transactions on Automatic Control*, pages 114–115, 1968.

[31] Leonid Knizhnerman, Vladimir Druskin, and Mikhail Zaslavsky. On optimal convergence rate of the rational Krylov subspace reduction for electromagnetic problems in unbounded domains. *SIAM Journal on Numerical Analysis*, 47:953–971, 2009.

[32] Leonid Knizhnerman and Valeria Simoncini. Convergence analysis of the extended Krylov subspace method for the Lyapunov equation. *Numerische Mathematik*, 118:567–586, 2011.

[33] Peter Lancaster and Leiba Rodman. *Algebraic Riccati equations*. Oxford University Press, Oxford, 1995.

[34] Jing-Rebecca Li and Jacob White. Low-Rank solutions of Lyapunov equations. *SIAM Journal on Matrix Analysis and Applications*, 24:260–280, 2002.

[35] Yiding Lin and V. Simoncini. Minimal residual methods for large scale Lyapunov equations. Preprint, August 2012.

[36] An Lu and Eugene L. Wachspress. Solution of Lyapunov equations by Alternating Direction Implicit iteration. *Computers Math. Applic.*, 21:43–58, 1991.

[37] *MATLAB*. The MathWorks, Inc., Natick, MA, USA. `http://www.mathworks.com`.

[38] Marlliny Monsalve and Daniel B. Szyld. Inexact Newton with Krylov projection and recycling for Riccati equations. Poster presented at the Conference on Numerical Linear Algebra: Perturbation, Performance, and Portability, Austin, Texas, 2010.

[39] Kirsten Morris and Carmeliza Navasca. Approximation of low rank solutions for linear quadratic control of partial differential equations. *Computational Optimization and Applications*, 46:93–111, 2010.

[40] K. Henrik A. Olsson and Axel Ruhe. Rational Krylov for eigenvalue computation and model order reduction. *BIT Numerical Mathematics*, 46:99–111, 2006.

[41] Thilo Penzl. Lyapack users guide. Technical Report SFB393/00-33, TU Chemnitz, 09107 Chemnitz, D, 2000. Available from `http://www.tu-chemnitz.de/sfb393/sfb00pr.html`.

[42] Axel Ruhe. Rational Krylov sequence methods for eigenvalue computation. *Linear Algebra and its Applications*, 58:391–405, 1984.

[43] Yousef Saad. Numerical solution of large Lyapunov equations. In M. A. Kaashoek, J. H. van Schuppen, and A. C. Ran, editors, *Signal Processing, Scattering, Operator Theory, and Numerical Methods, Proceedings of the Interntational Symposium MTNS-89, Vol. III*, pages 71–89, Boston, 1990. Birkhauser.

[44] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Second edition, SIAM, Philadelphia, 2003.

[45] John Sabino. *Solution of Large-Scale Lyapunov Equations via the Block Modified Smith Method*. PhD thesis, Rice University, 2006.

[46] Valeria Simoncini. A new iterative method for solving large-scale Lyapunov matrix equations. *SIAM Journal on Scientific Computing*, 29:1268–1288, 2007.

[47] Bart Vandereycken and Stefan Vandewalle. A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations. *SIAM Journal on Matrix Analysis and Applications*, 31:2553–2579, 2010.