Numer. Math. (2016) 134:327–342
DOI 10.1007/s00211-015-0777-7

Numerische Mathematik

Efficient low-rank solution of generalized Lyapunov equations

Stephen D. Shank[1,2] · Valeria Simoncini[3,4] ·
Daniel B. Szyld[1]

Received: 12 November 2014 / Revised: 8 October 2015 / Published online: 12 November 2015

Abstract An iterative method for the low-rank approximate solution of a class of generalized Lyapunov equations is studied. At each iteration, a standard Lyapunov equation is solved using Galerkin projection with an extended Krylov subspace method. This Lyapunov equation is solved inexactly, thus producing a nonstationary iteration. Several theoretical and computational issues are discussed so as to make the iteration efficient. Numerical experiments indicate that this method is competitive vis-à-vis the current state-of-the-art methods, both in terms of computational times and storage needs.

Mathematics Subject Classification 65F10 · 65N22 · 15A06

This version dated 7 October 2015. This research is supported in part by the U.S. National Science Foundation under Grants DMS-1115520 and DMS-1418882, and by the FARB12SIMO Grant of the Università di Bologna.

✉ Daniel B. Szyld
szyld@temple.edu

Stephen D. Shank
sshank@temple.edu

Valeria Simoncini
valeria.simoncini@unibo.it

1 Department of Mathematics, Temple University (038-16), 1805 N. Broad Street, Philadelphia, PA 19122-6094, USA

2 Present Address: Center for Computational Genetics and Genomics, Temple University, Bio-Life Building (015-00) 1900 N. 12th Street, Philadelphia, PA 19122-6078, USA

3 Dipartimento di Matematica, Università di Bologna, Piazza di Porta San Donato, 5, 40127 Bologna, Italy

4 IMATI-CNR, Pavia, Italy

1 Introduction

We are interested in the solution of generalized Lyapunov equations of the form

$$AX + XA^T + \sum_{j=1}^{m} N_j X N_j^T + BB^T = 0, \tag{1}$$

where the matrices A and N_j are $n \times n$, and A nonsingular, while B is $n \times \ell$, with $\ell < n$, and often much smaller than n. In general, one assumes that the matrices N_j are much sparser than A and/or of much smaller rank, and that A is stable, i.e., with its spectrum contained in the left-half plane. This type of equation, and with these characteristics, naturally arises in the context of model order reduction of bilinear control systems and linear parameter-varying systems as well as for the stability analysis of linear stochastic differential equations; see, e.g., [1,3,4], and references therein.

We are particularly interested in the large-scale case, when n is large and thus, this makes it impossible to simply rewrite (1) as a linear system with n^2 unknowns, and solve it with traditional linear solvers. Furthermore, one important concern here is storage. We seek solution methods where the storage (and computations) are $O(n)$ and not $O(n^2)$.

We seek a symmetric positive definite solution to (1), and this solution is the one that is meaningful in certain applications; see, e.g., [3] for an explanation. We consider low-rank approximations to X of the form ZZ^T, with Z $n \times r$, and r much smaller than n. Such low-rank approximations are known to exist in the case of the standard Lyapunov equation, i.e., (1) with $m = 0$, when for instance the singular values of the solution decay rapidly to zero; see, e.g., our comments later in Sect. 4.2. For the generalized equation (1), an argument is given in [1] indicating that this might also be the case.

We were inspired to consider this problem by the paper [1], where several low-rank solution methods are explored. Krylov subspace methods, such as preconditioned Conjugate Gradients (PCG) or BiCGstab, preconditioned with a bilinear ADI preconditioner, appeared to be competitive in terms of number of iterations and of the rank of the approximate solution. These preconditioned subspace methods attack the problem in its Kronecker formulation, and are implemented using mostly calculations on the low-rank factors, and these are appropriately truncated, as proposed in [13]; see also a description of this truncation later in Sect. 4.1.

In this paper we present an efficient low-rank iterative method to approximate the positive definite solutions of generalized Lyapunov equations of the form (1). The numerical experiments in Sect. (5) show that our approach uses less storage and less computational time than the best approach in [1]. We present several computational tools, and a new theorem which gives the foundation for some of those tools in the next few sections. The full algorithm collecting these tools is presented in Sect. 4.3. We mention that some of the material in this paper can also be found in the recent thesis [19].

In the following, $\|x\|$ denotes the Euclidean norm of the vector x; $\|M\|$ denotes any operator or matrix norm induced by any vector norm, while $\|M\|_F$ denotes the Frobenius norm.

2 Stationary iterative methods

We rewrite (1) as

$$\mathcal{M}(X) - \mathcal{N}(X) + BB^T = 0, \tag{2}$$

where $\mathcal{M}(X) = AX + XA^T$ is the Lyapunov operator, and

$$-\mathcal{N}(X) = \sum_{i=1}^{m} N_j X N_j^T. \tag{3}$$

We note that if C is a positive semidefinite matrix, then $\mathcal{N}(C)$ is a negative semidefinite matrix. Moreover, if A is stable, the solution matrix of a Lyapunov equation $\mathcal{M}(X) + C = 0$ with C positive semidefinite, is always positive semidefinite; see, e.g., [14, p. 443].

Let us further denote by $\Lambda(\mathcal{L})$ the spectrum of an operator $\mathcal{L}$, and by $\rho(\mathcal{L})$ its spectral radius. As done in [3], we shall assume that the pair (A, B) is controllable, and that there exists a positive-definite solution X to (1). Under these hypotheses, one can infer that $\sigma(A) \subset \mathbb{C}^-$ and $\rho(\mathcal{M}^{-1}\mathcal{N}) < 1$; see, e.g., [3, Theorem 4.1].

Following [3], we thus consider a stationary iteration of the form

$$\mathcal{M}(X_k) = \mathcal{N}(X_{k-1}) - BB^T, \quad k = 1, 2, \ldots. \tag{4}$$

Similar approaches have been pursued in the recent literature on more general linear matrix equations, see, e.g., [16]. As opposed to the approach taken in [3] for small problems, here we focus on low-rank approximations of the form $X_k = Z_k Z_k^T$, and begin with $X_0 = 0$; the basic recurrence is recast as Algorithm 2.1. The fact that $\rho(\mathcal{M}^{-1}\mathcal{N}) < 1$ ensures that the classical iteration is in fact a convergent fixed-point iteration for our problem for any initial approximation X_0; see, e.g., [2].

Algorithm 2.1 Stationary Iterations for generalized Lyapunov equations

```
Input: Problem data $A$, $N_j$, and $B$, inner and outer tolerances
Output: $Z$ so that $X = ZZ^T$ is an approximation to the solution of (1)
1: Approximately Solve $AX + XA^T + BB^T = 0$ for $X_1 = Z_1 Z_1^T$
2: for $k = 2, 3, \ldots$ do
3:     Set $B_k = [N_1 Z_{k-1}, \cdots, N_m Z_{k-1}, B]$
4:     Approximately Solve $AX + XA^T + B_k B_k^T = 0$ for $X_k = Z_k Z_k^T$
5:     If sufficiently accurate then stop
6: end for
```

The recurrence in Algorithm 2.1 requires a careful implementation to become effective, both in terms of memory usage and computational costs. One important consideration is that the solution of the Lyapunov equation in Step 4 of Algorithm 2.1 can be performed very efficiently by using recently developed low-rank solvers; see [20]. Here we will use the extended Krylov subspace Galerkin projection, as proposed in [21] and analyzed in [12], which provided satisfactory results in our experiments.

Moreover, the Lyapunov solves at each iteration in (4) can be solved at different accuracy as the iteration in k progresses: in particular, the stopping tolerance can be refined as the iterations proceed.

Stationary iterative methods such as (4) have a linear convergence rate, and in general one would not expect them to compete favorably with Krylov subspace methods such as preconditioned Conjugate Gradients or BiCGStab applied to the Kronecker form of the problem, as the latter methods often exhibit superlinear convergence. In this case, though, the experiments reported in [1] (as well as those we report in Sect. 5) show that for generalized Lyapunov equations of the form (2) these methods converge linearly as well. Other considerations for improvements that provide better convergence time and storage needs are presented in subsequent sections.

3 Inexact iterations

Classical stationary methods for solving $\mathbf{Ax} = \mathbf{b}$ with $\mathbf{A} = \mathbf{M} - \mathbf{N}$ and $\mathbf{M}$ nonsingular, proceed starting with an initial vector $\mathbf{x}_0$, and computing $\mathbf{x}_k$ from the iteration

$$\mathbf{Mx}_k = \mathbf{Nx}_{k-1} + \mathbf{b}, \quad k = 1, 2, \ldots \tag{5}$$

As already mentioned, such iteration converges for any $\mathbf{x}_0$ if and only if $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$; see, e.g., [22]. We are interested in the *inexact* solution of the linear system with $\mathbf{M}$ at each iteration, i.e., for $k = 1, 2, \ldots,$

$$\text{approximately solve } \mathbf{Mx}_k = \mathbf{Nx}_{k-1} + \mathbf{b}, \tag{6}$$

such that the norm of the (inner) residual $\mathbf{s}_k := \mathbf{Nx}_{k-1} + \mathbf{b} - \mathbf{Mx}_k$ is below some tolerance, called inner tolerance. Iterations such as (6) are often called *nonstationary iterations* since they can be interpreted as an iteration of the form (5), where the matrix $\mathbf{M}$ changes from one step to the next. We next show that if we ask that the inner residual is reduced in a way proportional to the outer residual $\mathbf{r}_{k-1} = \mathbf{b} - \mathbf{Ax}_{k-1}$, that is,

$$\|\mathbf{s}_k\| \leq \eta \|\mathbf{r}_{k-1}\| \tag{7}$$

for an appropriate value of η, then the convergence delay associated with the inexact solves remains under control. To this end, we need to assume a stronger condition than $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$, namely that $\|\mathbf{M}^{-1}\mathbf{N}\| < 1$, for some operator norm induced by a vector norm.

Theorem 1 *Let* $\mathbf{A} = \mathbf{M} - \mathbf{N}$, $\mathbf{M}$ *nonsingular such that* $\|\mathbf{M}^{-1}\mathbf{N}\| < 1$. *Let* $\mathbf{x}_\star$ *be the solution of* $\mathbf{Ax} = \mathbf{b}$, *and consider a nonstationary iteration of the form* (6). *Then*

(i) *If we impose* (7) *for some fixed* $\eta > 0$ *so that*

$$\gamma := \|\mathbf{M}^{-1}\mathbf{N}\| + \eta\|\mathbf{M}^{-1}\|\|\mathbf{A}\| < 1,$$

then the (*linear*) *iteration converges with factor no greater than* γ;

(ii) *If we impose that* $\|\mathbf{M}^{-1}\mathbf{s}_k\| \leq \eta\|\mathbf{M}^{-1}\mathbf{r}_{k-1}\|$, *for fixed* $\eta > 0$, *so that*

$$\gamma_1 = \|\mathbf{M}^{-1}\mathbf{N}\| + \eta\|\mathbf{M}^{-1}\mathbf{A}\| = \|\mathbf{M}^{-1}\mathbf{N}\| + \eta\|\mathbf{I} - \mathbf{M}^{-1}\mathbf{N}\| < 1, \tag{8}$$

then the convergence factor is no greater than γ_1.

Proof Since $\mathbf{M}\mathbf{x}_\star - \mathbf{N}\mathbf{x}_\star = \mathbf{b}$, we first write $\mathbf{x}_k - \mathbf{x}_\star = \mathbf{M}^{-1}\mathbf{N}(\mathbf{x}_{k-1} - \mathbf{x}_\star) + \mathbf{M}^{-1}\mathbf{s}_k$. Taking norms and using that $\mathbf{b} - A\mathbf{x}_\star = 0$, we have that

$$\begin{aligned}\|\mathbf{x}_k - \mathbf{x}_\star\| &\leq \|\mathbf{M}^{-1}\mathbf{N}\|\,\|\mathbf{x}_{k-1} - \mathbf{x}_\star\| + \|\mathbf{M}^{-1}\|\,\|\mathbf{s}_k\| \\ &\leq \|\mathbf{M}^{-1}\mathbf{N}\|\,\|\mathbf{x}_{k-1} - \mathbf{x}_\star\| + \eta\|\mathbf{M}^{-1}\|\,\|\mathbf{b} - \mathbf{A}\mathbf{x}_{k-1}\| \\ &= \|\mathbf{M}^{-1}\mathbf{N}\|\,\|\mathbf{x}_{k-1} - \mathbf{x}_\star\| + \eta\|\mathbf{M}^{-1}\|\,\|\mathbf{A}(\mathbf{x}_\star - \mathbf{x}_{k-1})\| \\ &\leq (\|\mathbf{M}^{-1}\mathbf{N}\| + \eta\|\mathbf{M}^{-1}\|\,\|\mathbf{A}\|)\|\mathbf{x}_{k-1} - \mathbf{x}_\star\|,\end{aligned}$$

and the first assertion of the theorem follows. The second assertion follows in a similar manner.

We remark that in order to satisfy the hypotheses of Theorem 1, one needs to have an appropriate value of the factor η in (7) so that, for example

$$\eta < \frac{1 - \|\mathbf{M}^{-1}\mathbf{N}\|}{\|\mathbf{M}^{-1}\mathbf{A}\|} = \frac{1 - \|\mathbf{M}^{-1}\mathbf{N}\|}{\|\mathbf{I} - \mathbf{M}^{-1}\mathbf{N}\|}. \tag{9}$$

In fact, for the convergence factor in (8) one has

$$\begin{aligned}\|\mathbf{M}^{-1}\mathbf{N}\| + \eta\|I - \mathbf{M}^{-1}\mathbf{N}\| &\leq \|\mathbf{M}^{-1}\mathbf{N}\| + \eta(1 + \|\mathbf{M}^{-1}\mathbf{N}\|) \\ &= (1 + \eta)\|\mathbf{M}^{-1}\mathbf{N}\| + \eta.\end{aligned} \tag{10}$$

Therefore, if $\|\mathbf{M}^{-1}\mathbf{N}\| \approx 0.3$, an inner tolerance of $\eta = 10^{-3}$ would give in (10) that $\|\mathbf{M}^{-1}\mathbf{N}\| + \eta\|I - \mathbf{M}^{-1}\mathbf{N}\| \approx 0.301$, indicating a very small deterioration of the convergence rate.[1] We should mention also that in the case of (i), we would have to replace $\|\mathbf{M}^{-1}\mathbf{A}\|$ in (9) with the more stringent $\|\mathbf{M}^{-1}\|\,\|\mathbf{A}\|$, but this is more a consequence of the proof than what one needs in practice. In fact, the main message from Theorem 1 is that we can solve (5) inexactly, with an inner residual norm which should decrease proportionally to the outer residual norm. We note that a similar result could be obtained for the error norm by using the technique in [16, Theorem 4.1], with a variable inner tolerance of the type in (7), but for the error.

We would like to put Theorem 1 in context within the literature of inexact stationary iterations. Such iterations were called nested, since another (inner) iterative method was used to approximately solve (5). These nested—also called inner–outer—iterations, or two-stage iterations, were analyzed at least as far back as [17]. In this reference, as well as in [15], and others, convergence is shown, without an estimate

[1] The value 0.3 is an estimate of the norm $\|\mathcal{M}^{-1}\mathcal{N}\|_F$ for the first example HEAT1 in Sect. 5. In general, this norm is not near one for the problems considered here, where the rank of $N_j \ll n$, and these matrices are extremely sparse.

of the convergence rate, nor the dependence of the rate on the inexactness of the solution of the inner iterations. On the other hand, in these papers, only convergence of the outer and the inner methods is assumed whereas here we have to assume norm-convergence; cf. [9] and references therein. We also mention that the condition (7) was used in [10] for inexact Richardson and Chebyshev iterations, and it is often used in inexact Newton for nonlinear systems [5].

Theorem 1 can be modified so as to include the case of an approximate right-hand side $\mathbf{b}_k \approx \mathbf{N}\mathbf{x}_{k-1} + \mathbf{b}$, as detailed in the following result.

Corollary 1 *Let* $\mathbf{A} = \mathbf{M} - \mathbf{N}$, $\mathbf{M}$ *nonsingular such that* $\|\mathbf{M}^{-1}\mathbf{N}\| < 1$.
Let $\mathbf{x}_\star$ *be the solution of* $\mathbf{A}\mathbf{x} = \mathbf{b}$, *and consider the nonstationary iteration*:

$$\begin{aligned} &\textit{determine} \quad && \mathbf{b}_k \approx \mathbf{N}\mathbf{x}_{k-1} + \mathbf{b}, \\ &\textit{approximately solve} \quad && \mathbf{M}\mathbf{x}_k = \mathbf{b}_k. \end{aligned} \tag{11}$$

Let $\tilde{\mathbf{s}}_k = \mathbf{b}_k - \mathbf{M}\mathbf{x}_k$ *be the residual of* (11), *and* $\mathbf{r}_{k-1} = \mathbf{b} - \mathbf{A}\mathbf{x}_{k-1}$. *If the iteration is performed so that* $\max\{\|\tilde{\mathbf{s}}_k\|, \|(\mathbf{N}\mathbf{x}_{k-1} + \mathbf{b}) - \mathbf{b}_k\|\} \le \frac{1}{2}\eta\|\mathbf{r}_{k-1}\|$, *and* η *satisfies either of the hypotheses of Theorem* 1, *then the iteration* (11) *converges linearly to* $\mathbf{x}_\star$ *with the convergence rates given in Theorem* 1.

Proof The result follows by writing $\mathbf{s}_k = \mathbf{N}\mathbf{x}_{k-1} + \mathbf{b} - \mathbf{M}\mathbf{x}_k = \tilde{\mathbf{s}}_k + \mathbf{N}\mathbf{x}_{k-1} + \mathbf{b} - \mathbf{b}_k$ and using the triangle inequality, so that the hypotheses of Theorem 1 hold.

We end this section with another result for the nonstationary iterations in Corollary 1, which will be relevant in low-rank computations.

Proposition 1 *Let* $\mathbf{A} = \mathbf{M} - \mathbf{N}$, $\mathbf{M}$ *nonsingular, and consider the nonstationary iteration and the hypotheses in Corollary* 1. *Then the residual* $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$ *satisfies*

$$\|\mathbf{r}_k\| \le \eta\|\mathbf{r}_{k-1}\| + \|\mathbf{N}(\mathbf{x}_k - \mathbf{x}_{k-1})\|. \tag{12}$$

Proof We write

$$\begin{aligned} \mathbf{r}_k &= \mathbf{b} - \mathbf{A}\mathbf{x}_k = \mathbf{b} - \mathbf{M}\mathbf{x}_k + \mathbf{N}\mathbf{x}_k \\ &= \mathbf{b} - \mathbf{M}\mathbf{x}_k + \mathbf{N}\mathbf{x}_{k-1} - \mathbf{N}\mathbf{x}_{k-1} + \mathbf{b}_k - \mathbf{b}_k + \mathbf{N}\mathbf{x}_k \\ &= [(\mathbf{b} + \mathbf{N}\mathbf{x}_{k-1}) - \mathbf{b}_k] + \tilde{\mathbf{s}}_k + \mathbf{N}(\mathbf{x}_k - \mathbf{x}_{k-1}) \end{aligned}$$

and the result follows from the triangle inequality and taking into account the hypotheses of Corollary 1.

4 An efficient implementation of the classical iteration

All the general results discussed in the previous section can be used in our context to improve the efficiency of Algorithm 2.1. However, to make the iteration really competitive, both in terms of storage and execution time, strategies need to be employed that take into account the matrix low-rank peculiarity of the problem. In the following we describe a few such tools, which will drastically improve the performance of our final implementation.

4.1 Matrix truncation

A tool widely used in low-rank approximation methods is truncation, whereby a low-rank product $X = GDG^T$, with G being $n \times k$ is approximated by $\mathcal{T}(X) = \widetilde{G}\widetilde{D}\widetilde{G}^T$, where $\widetilde{G}$ is $n \times p$, with $p \ll k$, and such that

$$\|X - \mathcal{T}(X)\|_F \leq \tau \, \|X\|_F \tag{13}$$

for some prescribed truncation tolerance τ; see, e.g., [13,16]. The procedure is implemented by computing a reduced QR decomposition of $G = Q_G R_G$ and an eigenvalue decomposition of the small symmetric matrix $R_G D R_G^T = USU^T$, $S = \mathrm{diag}(s_1, \ldots, s_p)$.

Truncation in the right-hand side is performed by retaining only the p eigenpairs in USU^T such that the eigenvalues satisfy $\sqrt{s_{p+1}^2 + \cdots + s_k^2} \leq \tau\sqrt{s_1^2 + \cdots + s_k^2}$, where the eigenvalues s_i, $i = 1, \ldots, p$, are sorted in non-increasing absolute value.

Truncation thus returns $\widetilde{G} = G\widetilde{U}$ and $\widetilde{D} = \widetilde{S}$. In the sequel, we shall also use the short-hand notation $\widetilde{Z} = \mathcal{T}(Z)$ instead of $\widetilde{Z}\widetilde{Z}^T = \mathcal{T}(ZZ^T)$.

The original right-hand side of our iterative step (4) is

$$\begin{aligned}
\mathcal{N}(X_{k-1}) &- BB^T \\
&= -(N_1 Z_{k-1} Z_{k-1}^T N_1^T + \cdots + N_m Z_{k-1} Z_{k-1}^T N_m^T + BB^T) \\
&= -[N_1 Z_{k-1}, \ \ldots \ , N_m Z_{k-1}, \ B][N_1 Z_{k-1}, \ \ldots \ , N_m Z_{k-1}, \ B]^T,
\end{aligned} \tag{14}$$

which is replaced by its truncation $\mathcal{T}(\mathcal{N}(X_{k-1}) - BB^T)$, for an appropriate value of the truncation parameter. In our context, Corollary 1 provides the allowable value for the truncation parameter, i.e., how inexact the right-hand side can be.

Truncation is also used after forming the next iterate X_k, so as to keep the approximation of low-rank. In this case, truncation is based on a singular value dropping of the iterate factor; see Sect. 4.2 for more details.

4.2 Separate right-hand sides

In Step 4. of Algorithm 2.1 we solve the Lyapunov equation

$$AX + XA^T + FF^T = 0, \quad F = [f_1, \ldots, f_\ell], \tag{15}$$

using a Galerkin method based on the *extended* Krylov subspace, where this space is defined as

$$\mathbb{EK}_j(A, F) = \mathbb{K}_j(A, F) + \mathbb{K}_j(A^{-1}, A^{-1}F),$$

where $\mathbb{K}_j(A, F)$ is the (block) Krylov subspace range$([F, AF, \ldots, A^{j-1}F])$.

In our case, the right-hand side is given by $-FF^T = \mathcal{T}(\mathcal{N}(X_{k-1}) - BB^T) \equiv -B_k B_k^T$. In spite of the truncation, however, the rank of B_k could be so high that the computational cost of building an orthonormal basis of $\mathbb{EK}_j(A, B_k)$ could dominate the overall cost. In addition, unless dramatic loss of rank occurs, storing such a basis requires $\mathcal{O}(2jp)$ vectors, where p is the rank of B_k. In our setting, these memory requirements quickly become prohibitive as the linear iteration progresses, as p usually increases.

This is a well known shortcoming of block Krylov subspace-based methods, and tangential approaches have been proposed for Lyapunov equations; see, e.g., [7] for additional considerations. Here we propose a different approach, which led to very significant execution time and memory savings, and which was also discussed in [11,18]. More precisely, writing $B_k = [b_1, \dots, b_r]$, then $B_k B_k^T = b_1 b_1^T + \dots + b_r b_r^T$, and exploiting the linearity of the problem

$$X = \mathcal{M}^{-1}(B_k B_k^T) = \sum_i \mathcal{M}^{-1}(b_i b_i^T) \approx \sum_i X_k^{(i)} = X_k,$$

where each $X_k^{(i)}$ is a low-rank approximation to the solution of the corresponding matrix equation $\mathcal{M}(X) = -b_i b_i^T$. For $i = 1, \dots, r$, $X_k^{(i)}$ is obtained with EKSM starting with the single vector b_i. Note that in general, we expect each $X_k^{(i)}$ to be of low rank, as an approximation to a numerically low-rank exact solution; see, e.g., [20, §5] and references therein for a discussion on the singular value decay properties of the solution to Lyapunov equations.

We point out that X_k is never explicitly formed, but the low-rank factorized form of the $X_k^{(i)}$'s is collected and then truncated to give the next low-rank (truncated) iterate X_k, stored in factored form. In fact, since the $X_k^{(i)}$'s are computed sequentially, the truncated factored form is updated "on the fly" as soon as the next $X_k^{(i)}$ is available. A loose truncation (10^{-14}) is performed during the collection of the approximate solutions, while a more severe truncation (10^{-10}) is performed at termination of the inner step. This resulted in very significant memory savings, while being able to reach the sought final accuracy in the residual.

Finally, we note that the obtained approximate solution X_k is not the same as the one that would be obtained by using the full block space with B_k. Nonetheless, to ensure a final (estimated) accuracy of `tol` for X_k, we solve for each single term $X_k^{(i)}$ by requiring an accuracy of `tol`$/r$. More details on stopping criteria are given in the next section.

4.3 The complete implementation

To finalize the algorithm for the efficient solution of (1) we still need to describe our stopping criterion. For our iteration (4) and its nonstationary version one needs to compute the norm of the residual

$$\mathcal{R}(X_k) = BB^T - \mathcal{M}(Z_k Z_k^T) + \mathcal{N}(Z_k Z_k^T) \tag{16}$$

to implement a residual-based stopping criterion. Its explicit low-rank factorization can be written as (here for $m = 1$):

$$\mathcal{R}(ZZ^T) = [Z, AZ, NZ, B] \begin{bmatrix} 0_p & I_p & 0_p & 0_{p\times r} \\ I_p & 0_p & 0_p & 0_{p\times r} \\ 0_p & 0_p & I_p & 0_{p\times r} \\ 0_{r\times p} & 0_{r\times p} & 0_{r\times p} & -I_r \end{bmatrix} \begin{bmatrix} Z^T \\ Z^T A^T \\ Z^T N^T \\ B^T \end{bmatrix},$$

so that, if Z has rank p, the low-rank factor of this residual has $3p + r$ columns. In our implementation, instead, we adapted the bound provided by Proposition 1, which in our setting only needs to estimate $\mathcal{N}(X_k - X_{k-1})$, leading to significantly lower storage and computational requirements. Again for $m = 1$, we have

$$\mathcal{N}(X_{k+1} - X_k) = [NZ_{k+1}, NZ_k] \begin{bmatrix} I_{p_{k+1}} & 0 \\ 0 & -I_{p_k} \end{bmatrix} \begin{bmatrix} Z_{k+1}^T N^T \\ Z_k^T N^T \end{bmatrix} = WDW^T,$$

where the low-rank factor has only $p_{k+1} + p_k$ columns, where p_k is the rank of Z_k. Thus, if we assume (for simplicity and argumentation) that $p = p_k = p_{k+1}$, then additional storage drops from $3p + r$ to $2p$. Therefore, defining $\tau_0^{bound} = 1$, the subsequent estimates of $\|\mathcal{R}(Z_k Z_k^T)\|$ are computed as

$$\tau_k^{bound} = \eta \|B_k B_k^T\|_F \tau_{k-1}^{bound} + \|R_k D R_k^T\|_F / \|B^T B\|_F, \tag{17}$$

where R_k is the square upper triangular matrix of the QR factorization of $W = [NZ_{k+1}, NZ_k]$. We note that whenever needed, the computation of the Frobenius norm is performed by also using invariance properties such as $\|FF^T\|_F = \|F^T F\|_F$.

The final algorithm with all the described ingredients is sketched in Algorithm 4.1, where for simplicity we used $m = 1$, and the expression $\mathcal{R}(X_k)$ refers to the residual defined in (16).

Observe that the variable inner tolerance τ_k^{inner} for the solution of the Lyapunov equation at the kth iteration, decreases by a factor η times the outer residual as prescribed in Theorem 1. This tolerance η is used also as the tolerance for the inexact truncated right-hand side τ_k^{trunc}; to satisfy the hypotheses of Corollary 1 one should prescribe a lower value of η. In other words, Theorem 1 provides a theoretical justification for the choices of the different truncation parameters, and the inexactness in the approximation to the solution of the Lyapunov equations in steps 2 and 9 of the algorithm.

5 Numerical experiments

In this section we report on our numerical experience with the proposed algorithm, and on a comparison with state-of-the-art approaches.

Algorithm 4.1 Low-rank classical iterations (for $m = 1$)

```
Input: Problem data $A$, $N$ and $B$, tolerances $\eta$, $\tau^{\text{outer}}$.
1: Set $\beta = \left\| BB^T \right\|_F$
2: Solve approximately $AX + XA^T + BB^T = 0$ for $X_1 = Z_1 Z_1^T$ so that $\|\mathcal{R}(X_1)\|_F < \eta$
3: Calculate $\tau_1^{\text{bound}} = \eta + \|\mathcal{N}(X_1)\|_F / \beta$
4: for $k = 2, 3, \ldots$ do
5:     Set tolerances $\tau_k^{\text{inner}} = \tau_k^{\text{trunc}} = \eta \tau_{k-1}^{\text{bound}}$
6:     Calculate the low-rank factor $B_k = \mathcal{T}_{\tau_k^{\text{trunc}}}([N Z_{k-1}, B]) = [b_1^{(k)}, \ldots, b_{\nu_k}^{(k)}]$
7:     Set $Z^{(0)} = \emptyset$
8:     for $i = 1, \ldots, \nu_k$ do
9:         Solve approximately $AX + XA^T + b_i^{(k)} (b_i^{(k)})^T = 0$ for $X^{(i)} = Z^{(i)} (Z^{(i)})^T$
                   so that $\|\mathcal{R}(X^{(i)})\|_F < \tau_k^{\text{inner}} / \nu_k$
10:        $\widetilde{Z}^{(i)} = \mathcal{T}_{\tau_k^{\text{trunc}}}([\widetilde{Z}^{(i-1)}, Z^{(i)}])$
11:    end for
12:    Set $Z_k = \widetilde{Z}^{(\nu_k)}$ (s.t. $X_k = Z_k Z_k^T$)
13:    Set $\tau_k^{\text{bound}} = \eta \| B_k B_k^T \|_F \tau_{k-1}^{\text{bound}} + \|\mathcal{N}(X_k - X_{k-1})\|_F / \beta$
14:    If $\tau_k^{\text{bound}} < \tau^{\text{outer}}$, then stop
15: end for
```

In our experiments we considered data stemming from the deterministic bilinear dynamical system

$$E\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \sum_{j=1}^{m} N_j \mathbf{x}(t) u_j(t) + B\mathbf{u}(t) \tag{18}$$

and the matrix solution X may be interpreted as a controllability Gramian of the system [3]. As in [1], in all our examples we used finite difference discretization in space, so that $E = I$. The matrices N_j stem from the discretization of Robin conditions on parts of the boundary, therefore m corresponds to the number of boundary sides on which these conditions are enforced; we refer to [3] for further details and for an example of the construction of the matrices. We note that the case of $E \neq I$, E symmetric positive definite, can be reduced to the case considered here by a Cholesky factorization of E, and the appropriate (implicit) change of the matrices A, N_j, B. The sparsity of E is usually similar to that of A, so that one expects a similar amount of fill in the factors. We also note that, alternatively, one can use the E-inner product throughout, as done, e.g., in a different context in [8], so that one can exploit solution methods for systems with E without requiring an explicit factorization.

All experiments were ran on a 2013 MacBook Pro with a 2.3 GHz Intel Core i7 Processor and 16 GB of memory running OS X 10.9.5 using MATLAB version R2014a.

Our experiments will be based on the following three examples.

Example 1 We consider the heat equation on the unit square, so that A is the discretization of the Laplace operator $L(\mathbf{x}) = \Delta\mathbf{x}$ in $\Omega = (0, 1)^2$. In addition, Robin conditions $\mathbf{n} \cdot \nabla(\mathbf{x}) = \frac{1}{2} u(\mathbf{x} - 1)$ are used on the left boundary of the domain, while

Dirichlet conditions $x = 0$ are used on the rest of the boundary, yielding a single matrix N_1 ($m = 1$); this is a small variant of [3, Example 6.1.2]; see also [1]. We call the resulting problem HEAT1.

By changing the problem so that Robin conditions are imposed on both left and right boundaries, we obtain two terms N_1 and N_2 ($m = 2$); we call the resulting dataset HEAT2.

Example 2 We consider (18) where A is the centered finite difference discretization of the following convection-diffusion operator, $L(\mathbf{x}) := -\Delta\mathbf{x} + \mathbf{x}_y$. The same Robin boundary conditions as in HEAT2 were used.

Example 3 This example is taken from [1, Section 6.2], and it is about a nonlinear RC circuit; it will be called CIRCUIT in later reference.

We start by evaluating the contribution of all enhancements adopted in the proposed implementation on two representative problems. Table 1 shows execution time (in seconds), memory requirements (in terms of real allocations), and solution rank for all variants, depending on whether the following choices were made:
In. = Inexact solves of the Lyapunov equation are performed with variable accuracy (see (7));
Rhs = truncation of the right-hand-side is performed before the inner matrix equation is solved (see Sect. 4.1);
Sep = m distinct Lyapunov equations are solved for B of rank m (see Sect. 4.2);
Res = The true residual norm is estimated with (17).

The final relative residual accuracy for all four methods was set to 10^{-6}. The numbers show that the impact of the various enhancements is similar in the symmetric and nonsymmetric cases, HEAT2 and CIRCUIT, respectively. Here and in the other experiments, we used the inexact relative tolerance $\eta = 10^{-2}$.

Variable accuracy in the inner solves reduces timings as expected (by about 60 %), without increasing the overall memory requirements. The major execution time improvements are obtained by separately solving the inner systems; times are further lowered by the truncation strategy and cheap residual norm estimation.

In the following we compare Algorithm 4.1 (GLEK in what follows, which stands for Generalized Lyapunov Extended Krylov method) with the following algorithms available in the literature. In the symmetric case, we used (i) a version of Algorithm 4.1 where steps 2 and 9 are solved using a rational Krylov subspace method as suggested in [6] (in the following RKSM); (ii) the tensorized Preconditioned CG in [1, Algorithm1] with a bilinear ADI preconditioner (in the following CGB(k), where k is the number of shifts used in the bilinear ADI preconditioner); (iii) the tensorized Preconditioned CG in [1, Algorithm 1], with an ADI preconditioner (in the following CGA(k)) k is the number of shifts used in the standard ADI preconditioner; (iv) Bilinear ADI as a solver, as described in [1, Section 5.1] (hereafter BADI(k), with k as above).

In the nonsymmetric case, BiCGSTAB replaces the CG algorithm in cases (ii) and (iii), as in [1, Algorithm 2] (BiB(k) and BiA(k) in the following).

The codes used for (ii), (iii), and (iv), are those obtained from the authors of [1]. The only modification we implemented is that we use an LU factorizations for $A - \sigma I$ for each shift σ, we store the factors, and reuse them as needed, in the same manner we do this for A in EKSM. This modification reduced the execution time.

Table 1 $n = 10{,}000$: Contribution of the various components to the performance of the final solver

In.	Rhs	Sep	Res	Time	Mem	Rk
HEAT2						
				16.4	250	62
			✓	16.0	214	62
		✓		5.8	254	63
		✓	✓	5.4	227	63
	✓			6.3	230	57
	✓		✓	6.1	191	55
	✓	✓		3.2	230	57
	✓	✓	✓	3.2	212	55
✓				10.6	250	62
✓			✓	10.2	222	62
✓		✓		2.7	254	63
✓		✓	✓	2.4	220	62
✓	✓			4.3	230	57
✓	✓		✓	4.2	185	53
✓	✓	✓		1.8	230	57
✓	✓	✓	✓	1.7	189	53
CIRCUIT						
				74.5	269	67
			✓	73.8	153	67
		✓		13.7	341	85
		✓	✓	12.8	256	85
	✓			45.8	253	63
	✓		✓	44.9	145	63
	✓	✓		11.0	301	75
	✓	✓	✓	10.7	248	75
✓				42.3	269	67
✓			✓	40.6	164	67
✓		✓		6.4	309	77
✓		✓	✓	5.7	214	76
✓	✓			28.4	253	63
✓	✓		✓	28.1	145	63
✓	✓	✓		5.6	289	72
✓	✓	✓	✓	5.2	216	71

In Table 2 we report the results for small dimensional problems ($n = 4900$), which helped us tune the parameters of ADI-based algorithms. For these experiments, as well as for the larger problems later, the final relative residual accuracy for all four methods was set to 10^{-8}. Larger problems were run with the best parameters from this experimental study. This table reports execution time (in seconds), total memory requirements and final rank of the solution. For each example, the smallest value in each column is highlighted in bold. It is interesting that although these problems

Table 2 Results of a small parameter study

Problem	Method	Time	Mem.	Rk.
HEAT1	GLEK	**0.76**	146	**43**
HEAT1	RKSM	9.42	**139**	**43**
HEAT1	CGB(2)	3.91	534	50
HEAT1	CGA(4)	2.01	497	50
HEAT1	BADI(6)	1.31	197	49
CIRCUIT	GLEK	**5.38**	245	105
CIRCUIT	RKSM	62.74	**216**	95
CIRCUIT	BiB(4)	50.61	2581	88
CIRCUIT	BiA(2)	24.43	2565	**83**
CIRCUIT	BADI(8)	34.10	361	90
HEAT2	GLEK	**1.74**	274	**82**
HEAT2	RKSM	19.14	**272**	**82**
HEAT2	CGB(2)	13.08	1016	95
HEAT2	CGA(2)	10.26	1011	95
HEAT2	BADI(4)	3.55	472	94
ADVDIFF	GLEK	**2.33**	**293**	**87**
ADVDIFF	RKSM	26.41	294	88
ADVDIFF	BiB(2)	126.89	3612	94
ADVDIFF	BiA(6)	26.00	2787	101
ADVDIFF	BADI(4)	12.01	507	101

yield matrices only in the order of a few thousands, the overall performance already provides a picture consistent with the one that can be seen for larger problems. The two methods based on Preconditioned CG are not competitive, both with respect to time and to memory requirements. This latter shortcoming will be exacerbated with larger dimensions. GLEK shows the best performance with respect to all three performance measures, except for the Circuit problem, for which memory requirements seem to be slightly higher. Using a rational Krylov method instead of the extended Krylov method, while being competitive in terms of storage, produces considerably slower execution times. This is particular for our setting, where many Lyapunov equations need to be solved.[2] Therefore, we omit results on computations with this method in the experiments that follow.

In Table 3 we report our numerical experiments for two finer discretizations, $n = 22{,}500$ and $n = 102{,}400$. The time and memory requirements are increasingly larger as the matrix dimensions expand, as expected, but the comparative behavior of the methods remains consistent with those for $n = 4900$. We observe in particular the very large number of solves—included in this table for further comparison—for the two Preconditioned CG methods. These operations will be particularly expensive when dealing with three-dimensional problems, but also with two-dimensional more

[2] We mention that similar slow results are obtained if one uses the tangential interpolation version of a rational Krylov method, as proposed in [7].

Table 3 Performance comparison for larger discretizations of the problems

Problem	Method	Time	Mem.	Sol.	Rk.
$n = 22{,}500$					
HEAT1	GLEK	**3.34**	**177**	**410**	**49**
HEAT1	CGA(4)	16.23	608	1496	57
HEAT1	CGB(2)	27.49	621	4108	56
HEAT1	BADI(6)	10.67	225	1852	56
HEAT2	GLEK	**7.74**	**313**	**780**	**93**
HEAT2	CGA(2)	62.02	1209	4076	108
HEAT2	CGB(2)	79.44	1204	10,120	109
HEAT2	BADI(4)	27.00	547	6804	109
ADVDIFF	GLEK	**11.26**	**322**	**802**	**95**
ADVDIFF	BiA(6)	121.60	3301	15,402	111
ADVDIFF	BiB(2)	552.38	4254	49,365	102
ADVDIFF	BADI(4)	29.09	562	7247	111
CIRCUIT	GLEK	**25.12**	**303**	**1784**	117
CIRCUIT	BiA(2)	190.57	3729	21,294	**109**
CIRCUIT	BiB(4)	194.39	2868	35,392	111
CIRCUIT	BADI(8)	267.54	449	11,808	112
$n = 102{,}400$					
HEAT1	GLEK	**18.46**	**213**	**470**	**49**
HEAT1	CGA(4)	131.18	633	1488	60
HEAT1	CGB(2)	258.73	712	5656	60
HEAT1	BADI(6)	117.05	241	2542	60
HEAT2	GLEK	**44.01**	**325**	**903**	**95**
HEAT2	CGA(2)	486.69	1390	5660	117
HEAT2	CGB(2)	645.36	1403	14,050	117
HEAT2	BADI(4)	257.43	587	9332	117
ADVDIFF	GLEK	**80.98**	**333**	**906**	**97**
ADVDIFF	BiA(6)	586.14	3048	14220	121
ADVDIFF	BiB(2)	3316.06	4730	63,800	104
ADVDIFF	BADI(4)	271.46	597	9488	119
CIRCUIT	GLEK	**198.16**	**386**	**2273**	151
CIRCUIT	BiA(2)	2190.25	6546	40,756	**136**
CIRCUIT	BiB(4)	1756.72	4394	56,356	**136**
CIRCUIT	BADI(8)	2358.12	545	18,252	**136**

complex problem geometries. All numbers are in favor of the new algorithm GLEK, except for the Circuit problem, for which once again GLEK yields a solution factor with a slightly larger rank. Both execution times and total memory requirements of GLEK are considerably better than those of available methods, with up to one order of magnitude improvement in some instances.

6 Conclusions

We have proposed a new implementation of the classical iteration for the solution of large-scale generalized Lyapunov equation. Although stationary iterative methods had been used in the past for this problem, several significant improvements had to be designed to make the original strategy competitive in the large-scale setting. One of them is a variable stopping criterion for the inner iteration. Under certain conditions, we prove that the outer iteration will converge with the prescribed level of inexactness of the approximate solutions at each iteration. Our numerical experience - albeit circumscribed to a certain class of problems - seems to show that a careful implementation allows one to solve large problems in a few tens of seconds on a commercially available laptop.

Acknowledgments We are grateful to Peter Benner and Tobias Breiten for providing us with all their MATLAB codes from [1]. We also thank the referees for their questions and comments which helped improve our presentation.

References

1. Benner, P., Breiten, T.: Low-rank methods for a class of generalized Lyapunov equations and related issues. Numerische Mathematik **124**, 441–470 (2013)
2. Berman, A., Plemmons, R.J.: Nonnegative Matrices in the Mathematical Sciences. (3rd edn.) Academic Press, New York (1979). Reprinted by SIAM, Philadelphia (1994)
3. Damm, T.: Direct methods and ADI-preconditioned Krylov subspace methods for generalized Lyapunov equations. Numer. Linear Algebra Appl. **15**, 853–871 (2008)
4. Damm, T., Hinrichsen, D.: Newton's method for a rational matrix equation occurring in stochastic control. Linear Algebra Appl. **332–334**, 81–109 (2001)
5. Dembo, R.S., Eisenstat, S.C., Steihaug, T.: Inexact Newton methods. SIAM J. Numer. Anal. **19**, 400–408 (1982)
6. Druskin, V., Knizhnerman, L., Simoncini, V.: Analysis of the rational Krylov subspace and ADI methods for solving the Lyapunov equation. SIAM J. Numer. Anal. **49**, 1875–1898 (2011)
7. Druskin, V., Simoncini, V., Zaslavsky, M.: Adaptive tangential interpolation in rational Krylov subspaces for MIMO model reduction data. SIAM J. Matrix Anal. Appl. **35**, 476–498 (2014)
8. Fasi, M.: Weighted geometric mean of large-scale matrices: numerical analysis and algorithms. Master's thesis. Department of Computer Science. Università di Bologna, Italy, March 2015
9. Frommer, A., Szyld, D.B.: On necessary conditions for convergence of stationary iterative methods for hermitian definite and semidefinite linear systems. Linear Algebra Appl. **453**, 192–201 (2014)
10. Golub, G.H., Overton, M.L.: The convergence of inexact Chebyshev and Richardson iterative methods for solving linear systems. Numerische Mathematik **53**, 571–593 (1988)
11. Hu, D.Y., Reichel, L.: Krylov subspace methods for the Sylvester equation. Linear Algebra Appl. **172**, 283–313 (1992)
12. Knizhnerman, L., Simoncini, V.: Convergence analysis of the extended Krylov subspace method for the Lyapunov equation. Numerische Mathematik **118**, 567–586 (2011)
13. Kressner, D., Tobler, C.: Low-rank tensor Krylov subspace methods for parametrized linear systems. SIAM J. Matrix Anal. Appl. **32**, 1288–1316 (2011)
14. Lancaster, P., Tismenetsky, M.: The Theory of Matrices, 2nd edn. Academic Press, Orlando (1985)
15. Lanzkron, P.J., Rose, D.J., Szyld, D.B.: Convergence of nested classical iterative methods for linear systems. Numerische Mathematik **58**, 685–702 (1991)
16. Matthies, H.G., Zander, E.: Solving stochastic systems with low-rank tensor compression. Linear Algebra Appl. **436**, 3819–3838 (2012)
17. Nichols, N.K.: On the convergence of two-stage iterative processes for solving linear equations. SIAM J. Numer. Anal. **10**, 460–469 (1973)

18. Penzl, T.: A cyclic low-rank Smith method for large sparse Lyapunov equations. SIAM J. Sci. Comput. **21**, 1401–1418 (1999)
19. Shank, S.D.: Low-rank Solution Methods for Large-scale Linear Matrix Equations. PhD thesis, Department of Mathematics, Temple University, May 2014
20. Simoncini, V.. Computational methods for linear matrix equations. Preprint, Università di Bologna, March 2013, Revised January 2014. To appear in SIAM Review
21. Simoncini, V.: A new iterative method for solving large-scale Lyapunov matrix equations. SIAM J. Sci. Comput. **29**, 1268–1288 (2007)
22. Varga, R.S.: Matrix Iterative Analysis. Prentice-Hall, Englewood Cliffs (1962). Second Edition, revised and expanded. Springer, Berlin (2000)