Computing the matrix exponential and the Cholesky factor of a related finite horizon Gramian

Tony Stillfjord and Filip Tronarp

Centre for Mathematical Sciences, Lund University
October 21, 2023

Abstract

In this article, an efficient numerical method for computing finite-horizon controllability Gramians in Cholesky-factored form is proposed. The method is applicable to general dense matrices of moderate size and produces a Cholesky factor of the Gramian without computing the full product. In contrast to other methods applicable to this task, the proposed method is a generalization of the scaling-and-squaring approach for approximating the matrix exponential. It exploits a similar doubling formula for the Gramian, and thereby keeps the required computational effort modest. Most importantly, a rigorous backward error analysis is provided, which guarantees that the approximation is accurate to the round-off error level in double precision. This accuracy is illustrated in practice on a large number of standard test examples.

The method has been implemented in the Julia package FiniteHorizonGramians.jl, which is available online under the MIT license. Code for reproducing the experimental results is included in this package, as well as code for determining the optimal method parameters. The analysis can thus easily be adapted to a different finite-precision arithmetic.

1 Introduction

Consider a pair of matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. This article is concerned with the numerical approximation of two matrix functions $\Phi(A)$ and $G(A, B)$, defined by

$$\Phi(A) = e^A, \tag{1a}$$

$$G(A, B) = \int_0^1 e^{A\tau} BB^* e^{A^*\tau} \,\mathrm{d}\tau, \tag{1b}$$

where * denotes the Hermitian conjugate. The first function, Φ, is just the matrix exponential with specialized notation, which shall simplify the subsequent discussion. The second function, G, is the controllability Gramian of the pair (A, B) over the unit interval, and may equivalently be characterized as the solution at the end-point of the following Lyapunov differential equation (Abou-Kandil et al., 2012)

$$\dot{Q}(t) = AQ(t) + Q(t)A^* + BB^*, \quad Q(0) = 0, \quad t \in [0, 1], \tag{2}$$

that is, $G(A, B) = Q(1)$. It is immediately clear that the controllability Gramian over the interval $[0, t]$ may be obtained by $G(tA, \sqrt{t}B)$. The controllability Gramian is always positive semi-definite. If additionally the pair (A, B) is controllable, then it is positive definite. Therefore, G has a Cholesky factorization $G(A, B) = U^*(A, B)U(A, B)$ for some upper triangular matrix function $U(A, B)$[1].

[1] However, when the Gramian fails to be positive definite, then the Cholesky factor is not unique.

1.1 Contribution

The aim is to develop a numerical algorithm for the computation of both matrix exponential, $\Phi(A)$, and a upper triangular Cholesky factor, $U(A, B)$, without forming $G(A, B)$ as an intermediate step. This has applications in numerically robust implementations of linear filters and smoothers, via the so called square-root or array algorithms (Anderson and Moore, 2012, Kailath et al., 2000), and possibly in robust state-space balancing and truncation algorithms (Antoulas, 2005). The approach proposed here is based on a certain doubling recursion for both Φ and G (Anderson and Moore, 2012), which extends the scaling and squaring method of the matrix exponential (Al-Mohy and Higham, 2010, Higham, 2005, 2008). This is similar to the doubling recursion for computing both the matrix exponential and its Fréchet derivative (Higham, 2008, Section 10.6), which was turned into a serviceable algorithm by computing the Fréchet derivative of the initial Padé approximation of the matrix exponential by Al-Mohy and Higham (2009). However, this provides no information on how to construct the initial approximations which are to be doubled. That is accomplished here by drawing on a connection between the diagonal Padé approximants and a certain Petrov–Galerkin approximation of $\Phi(At)$ on the interval $[0, 1]$ (Moore, 2011). This allows for the development of an algorithm that almost computes Φ in the conventional way while also providing an initial approximation to the Cholesky factor U, both which are then propagated through the doubling recursion.

The error analysis for Φ is the same as for the conventional method (Higham, 2005), while the error analysis for G is more intricate. By lifting the problem to the computation of the Gramian functional

$$\mathcal{G}(\tilde{A}, \tilde{B}) = \int_0^1 e^{\tilde{A}(t)} \tilde{B}(t) \tilde{B}^*(t) e^{\tilde{A}^*(t)} \, \mathrm{d}t, \tag{3}$$

such that $G(A, B) = \mathcal{G}(At, B)$, it is demonstrated that the proposed method is backward stable, in the sense that there are perturbations $\Delta A(t)$ and $\Delta B(t)$ such that the approximated Gramian is given by

$$\mathcal{G}(At + \Delta A, B + \Delta B). \tag{4}$$

Norm bounds on $2^{-s}A$ are obtained which guarantee that the relative errors in the perturbed data to $\mathcal{G}$ are bounded by unit roundoff in the supremum norm. The norm of B turns out to be irrelevant for the backward errors. As the algorithm closely resembles the classical scaling and squaring algorithm with Padé approximants, it is expected to be appropriate to use under the same circumstances. This is not only corroborated by the error analysis but also through numerical experiments.

The proposed algorithm and the analysis informing its design has been implemented in the Julia programming language (Bezanson et al., 2017). The resulting software package, FiniteHorizonGramians.jl[2], is released under the MIT license.

The rest of the article is organized as follows. This section concludes with a discussion on related work. The main ideas behind the algorithm construction are established in Section 2. In Section 3, an error analysis is established and in Section 4 the rank properties of the approximated Gramian in relation to the true Gramian are discussed. The final algorithm design is settled in Section 5, where numerical experiments are also carried out, and concluding remarks are given in Section 6.

[2] https://github.com/filtron/FiniteHorizonGramians.jl.

1.2 Related work

The development of methods for computing matrix exponentials have a long history (Moler and Van Loan, 1978). A prominent approach is based on the scaling and squaring method using various base approximations (Moler and Van Loan, 2003), where the Padé approximations have become preferred (Al-Mohy and Higham, 2010, Güttel and Nakatsukasa, 2016, Higham, 2005, 2008). Computing the matrix exponential and some associated quantity has been done by Al-Mohy and Higham (2009), in the case of the Fréchet derivative. However, to the authors' knowledge no such development has been done for computing full Cholesky factors of finite horizon Gramians. In the following, the literature closest to the present contribution is reviewed.

Differential Lyapunov equations. Many numerical methods have been suggested for solving large-scale differential Lyapunov equations (2), e.g. BDF and Rosenbrock methods (Benner and Mena, 2018, Mena, 2007), projection methods (Behr et al., 2019, Kirsten and Simoncini, 2020, Koskela and Mena, 2020), exponential integrators (Li et al., 2021), splitting schemes (Ostermann et al., 2019, Stillfjord, 2015, 2018), and numerical quadrature (Stillfjord, 2015, 2018). Most of these methods are designed for differential Riccati equations, but they reduce to methods for Lyapunov equations by setting the nonlinear term to zero. The focus in all these works has been on low-rank solutions $G(A, B) = U^*U$, where $U^* \in \mathbb{R}^{n\times r}$ with $r \ll n$ and large n where even storing the solution $G(A, B)$ might be problematic. With the exception of the Krylov methods, these methods lack rigorous error analyses. In practice, they typically produce approximations with errors in the range $[10^{-3}, 10^{-12}]$. This is in contrast to the present interest, which is to obtain approximations to full-rank upper triangular Cholesky factors $U \in \mathbb{R}^{n\times n}$, for n moderately small, which are fully accurate in double precision arithmetic.

Algebraic Lyapunov equations. When A is Hurwitz, the controllability Gramian over the interval $[0, t]$ tends to the solution of a the algebraic Lyapunov equation as $t \to \infty$, namely

$$AG + GA^* = -BB^*.$$

For such equations, there is a similarly wide spread of numerical methods. For an overview of the currently popular large-scale situation, see the surveys by Benner and Saak (2013), Simoncini (2016). Most notable is, perhaps, the commonly used LRCF-ADI method (Benner et al., 2008, Li and White, 2002). However, also here the focus is on low-rank factorizations with $U^* \in \mathbb{R}^{n\times r}$, and while the error analyses are more mature than for the differential case they are generally not sharp.

The method of choice for small-scale algebraic Lyapunov equations still seems to be the method developed by Hammarling (1982). It is a modification of the Bartels-Stewart method (Bartels and Stewart, 1972) based on Schur factorization, and directly computes the Cholesky factors of the solution. However, to the authors knowledge, there is no extension to the case of differential Lyapunov equations.

2 Sketch of algorithm

In this section, the main ideas of the proposed algorithm are established. In particular, the doubling formula for both Φ and G is established in Section 2.1. The doubling recursion requires initial approximations of Φ and G, and such an approximation based on a Petrov–Galerkin method in a shifted Legendre basis is reviewed in Section 2.2.

2.1 Doubling formulae

It is well known that the matrix exponential satisfies the doubling formula $\Phi(A) = \Phi^2(A/2)$, which is a crucial component of the standard algorithm for numerically computing matrix exponentials (Higham, 2005). It is perhaps less known that G also satisfies a doubling formula, but such ideas have been around for a long time (Anderson and Moore, 2012, Section 6.7). The result is as follows.

Lemma 1. *The function $G(A, B)$ satisfies the following doubling formula*

$$G(A, B) = G(A/2, B/\sqrt{2}) + e^{A/2} G(A/2, B/\sqrt{2}) e^{A^*/2}. \tag{5}$$

Proof. Split the integral defining $G(A, B)$ in half:

$$\begin{aligned} G(A, B) &= \int_0^{1/2} e^{A\tau} B B^* e^{A^*\tau} \,\mathrm{d}\tau + e^{A/2} \int_{1/2}^1 e^{A(\tau-1/2)} B B^* e^{A^*(\tau-1/2)} \,\mathrm{d}\tau \, e^{A^*/2} \\ &= G(A/2, B/\sqrt{2}) + e^{A/2} G(A/2, B/\sqrt{2}) e^{A^*/2}. \end{aligned}$$

Here, the last step is a change of variables $\tau \mapsto \tau - 1/2$ in the second integral. □

Now let s be an integer and define the following series of functions for $k = 0, 1, \ldots, s$:

$$\begin{aligned} \Phi_k(A) &= \Phi(A/2^{s-k}), \\ G_k(A, B) &= G(A/2^{s-k}, B/\sqrt{2^{s-k}}). \end{aligned}$$

From the doubling formulae, a recursion for Φ_k and G_k is readily obtained:

$$\Phi_{k+1}(A) = \Phi_k^2(A), \tag{7a}$$

$$G_{k+1}(A, B) = \Phi_k(A) G_k(A, B) \Phi_k^*(A) + G_k(A, B). \tag{7b}$$

This suggests that an efficient algorithm for approximating Φ and G may be obtained by simply adding some extra computations to the scaling and squaring algorithm for the matrix exponential. However, the goal is to compute a Cholesky factor of the Gramian, which can be achieved by factoring $G_k(A, B) = U_k(A, B)^* U_k(A, B)$. By (7b), the factor $U_k(A, B)$ satisfies the recursion

$$U_{k+1}^*(A, B) U_{k+1}(A, B) = \begin{bmatrix} U_k(A, B) \Phi_k^*(A) \\ U_k(A, B) \end{bmatrix}^* \begin{bmatrix} U_k(A, B) \Phi_k^*(A) \\ U_k(A, B) \end{bmatrix}, \tag{8}$$

and $U_{k+1}(A, B)$ may then be obtained by taking the upper triangular factor of the QR decomposition of the last factor on the right-hand side. For a complete algorithm, it remains to obtain the initial approximations of $\Phi_0(A)$ and $U_0(A, B)$. An approach for this is developed in the following.

2.2 The initial approximations

In view of the doubling formula in (7), it remains to obtain initial approximations $\widehat{\Phi}_0$ and $\widehat{G}_0$ of the matrix exponential Φ_0 and the Gramian G_0, respectively. For this purpose, define $A_s = A/2^s$ and $B_s = B/\sqrt{2^s}$. Then approximations are sought for

$$\begin{aligned} \Phi_0(A) &= \Phi(A_s), \\ G_0(A, B) &= G(A_s, B_s). \end{aligned}$$

Consider an order q expansion of $t \mapsto \Phi(A_s t)$ in terms of Legendre polynomials P_k on the unit interval:

$$\widehat{E}_q(t) = \sum_{k=0}^{q} C_k P_k(t) \approx \Phi(A_s t). \tag{10}$$

Then, since $P_k(1) = 1$ and $\|P_k\|^2 = \frac{1}{2k+1}$, $\widehat{\Phi}_0(A)$ and $\widehat{G}_0(A, B)$ can be formed as

$$\widehat{\Phi}_0(A) = \widehat{E}_q(1) = \sum_{k=0}^{q} C_k, \tag{11a}$$

$$\widehat{G}_0(A, B) = \sum_{k=0}^{q} \frac{1}{2k+1} C_k B_s B_s^* C_k^*. \tag{11b}$$

Furthermore, the Gramian may be written as $\widehat{G}_0(A, B) = \tilde{U}_0^* \tilde{U}_0$, where

$$\tilde{U}_0^* = \begin{bmatrix} C_0 B_s & C_1 B_s/\sqrt{3} & \cdots & C_k B_s/\sqrt{2k+1} & \cdots & C_q B_s/\sqrt{2q+1}, \end{bmatrix} \tag{12}$$

and an upper triangular (not necessarily square) square-root factor of $\widehat{G}_0(A, B)$ may be obtained by the QR factorization. There are various ways of constructing the expansion coefficients in (10), such as projections in $\mathscr{L}_2([0, 1])$ or various Petrov–Galerkin type methods. However, a very particular Petrov–Galerkin method gives $\widehat{\Phi}_0(A)$ as a diagonal Padé approximation (Moore, 2011). Namely, if the coefficients satisfy

$$\begin{bmatrix} \mathrm{I} & -3\mathrm{I} & 5\mathrm{I} & \dots & \dots & \dots \\ -A_s & 6\mathrm{I} & A_s & 0 & \dots & \vdots \\ 0 & -A_s & 10\mathrm{I} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & A_s & \vdots \\ \vdots & \ddots & \ddots & \ddots & (4q-2)\mathrm{I} & 0 \\ 0 & \dots & \dots & 0 & -A_s & (4q+2)\mathrm{I} \end{bmatrix} \begin{bmatrix} \tilde{C}_0 \\ \tilde{C}_1 \\ \vdots \\ \vdots \\ \tilde{C}_{q-1} \\ \tilde{C}_q \end{bmatrix} = \begin{bmatrix} \mathrm{I} \\ 0 \\ \vdots \\ \vdots \\ 0 \\ 0 \end{bmatrix},$$

where $\tilde{C}_k = C_k/(2k+1)$, then they are rational functions in A_s, say,

$$C_k(A_s) = D_q^{-1}(A_s) L_k(A_s). \tag{13}$$

Their sum evaluates to

$$\sum_{k=0}^{q} C_k(A_s) = D_q^{-1}(A_s) N_q(A_s) = r_q(A_s), \tag{14}$$

where N_q and D_q are the numerator and denominator, respectively, in the diagonal Padé approximation r_q of e^z, see Moore (2011) and references therein[3]. Tables of coefficients for computing C_q, D_q and N_q have been generated symbolically in Julia using the Symbolics.jl package (Gowda et al., 2022). They are provided in Appendix B for $q = 3, 5, 7, 9, 13$.

[3] Unless it is necessary for the clarity of exposition, the explicit dependence on A_s in $C_k(A_s)$, $D_q(A_s)$ and $N_q(A_s)$ will henceforth be suppressed.

3 Error analysis

In this section, error analysis of the Gramian approximation is performed. From the analysis of (Moore, 2011), it follows that

$$V_t(A_s) = -A_s C_q \int_0^t e^{-A_s\tau} P_q(\tau)\,\mathrm{d}\tau, \tag{15a}$$

$$\widehat{E}_q(t) = e^{A_s t}\Big(\mathrm{I} + V_t(A_s)\Big), \tag{15b}$$

where for each t, $V_t(A_s)$ is a matrix function in A_s. When $\|V_t(A_s)\| < 1$ a backwards error of $\widehat{E}_q$ is given by

$$F_t(A_s) = \log\Big(\mathrm{I} + V_t(A_s)\Big), \tag{16}$$

and since $V_t(A_s)$ is a matrix function in A_s, then so is $F_t(A_s)$. Furthermore, $F_1(A_s)$ is the corresponding backward error for the Padé approximation to the matrix exponential, since $\widehat{E}_q(1) = r_q(A_s)$ by (11a) and (14). Before proceeding, recall the following definitions from Higham (2005).

- θ_q is a number such that $2^s\|F_1(A_s)\| \leq 2^{-53}$ whenever $\|A_s\| \leq \theta_q$.
- ν_q is the maximal radius around the origin for which the Padé denominator, $D_q(z)$, is analytic; $\nu_q = \min\{z\colon D_q(z) = 0\}$.
- $\|D_q^{-1}(A_s)\| \leq \xi_q$ whenever $\|A_s\| \leq \theta_q$.

The following result was obtained by Higham (2005).

Proposition 1. *Let* $s \geq \max\left(0, \log_2 \frac{\|A\|}{\theta_q}\right)$, *then*

$$e^{-A_s} r_q(A_s) = \mathrm{I} + V_1(A_s) = e^{F_1(A_s)}, \tag{17}$$

where $V_1(A_s)$ *and* $F_1(A_s)$ *are well-defined functions of* A_s *in the sense of matrix functions which commute with* A_s. *Furthermore,* $\widehat{\Phi}(A) = e^{A+2^s F_1(A_s)}$ *and*

$$\frac{\|2^s F_1(A_s)\|}{\|A\|} \leq 2^{-53} \approx 1.1 \cdot 10^{-16},$$

so that $\widehat{\Phi}(A) = r_q^{2^s}(A_s)$ *approximates* e^A *to full accuracy in double precision arithmetic, in the backward error sense. Lastly,* $\theta_q \leq \nu_q$ *for* $q \leq 21$.

This result applies to the present approximation to the matrix exponential as it is computed in exactly the same way. However, the error analysis for the Gramian is more involved and shall be pursued in the following.

3.1 Backwards error of Gramian

It appears infeasible to obtain a backwards error for G directly. However, the problem can be *lifted* to the computation of the Gramian functional (3), whose definition is restated:

$$\mathcal{G}(A, B) = \int_0^1 e^{A(t)} B(t) B^*(t) e^{A^*(t)}\,\mathrm{d}t.$$

Note that $\mathcal{G}(At, B) = G(A, B)$ and $\widehat{G}_0(A, B)$ is an approximation to $\mathcal{G}(A_s t, B_s)$. Thus if there are matrix-valued functions $\widehat{A}(t)$ and $\widehat{B}(t)$ such that $\widehat{G}(A, B) = \mathcal{G}(\widehat{A}(t), \widehat{B}(t))$ a

backward result is obtained. It follows from the doubling formula and the discrete variation of constants formula that $\widehat{G}$ is given by

$$\widehat{G} = \sum_{m=0}^{2^s-1} r_q^m(A_s)\widehat{G}_0 r_q^m(A_s^*). \tag{18}$$

The idea is to use this formula to construct $\widehat{A}(t)$ and $\widehat{B}(t)$ such that

$$\widehat{G} = \mathcal{G}(\widehat{A}(t), \widehat{B}(t)).$$

This is done by chopping up the interval $[0,1]$ into 2^s-1 equally large sub-intervals of length 2^{-s} and then matching terms.

$$\begin{aligned}
\mathcal{G}(\widehat{A}(t), \widehat{B}(t)) &= \int_0^1 e^{\widehat{A}(t)}\widehat{B}(t)\widehat{B}^*(t)e^{\widehat{A}^*(t)}\,\mathrm{d}t \\
&= \sum_{k=0}^{2^s-1}\int_{k2^{-s}}^{(k+1)2^{-s}} e^{\widehat{A}(t)}\widehat{B}(t)\widehat{B}^*(t)e^{\widehat{A}^*(t)}\,\mathrm{d}t \\
&= \sum_{k=0}^{2^s-1} 2^{-s}\int_0^1 e^{\widehat{A}(k2^{-s}+t2^{-s})}\widehat{B}(k2^{-s}+t2^{-s})\widehat{B}^*(k2^{-s}+t2^{-s})e^{\widehat{A}^*(k2^{-s}+t2^{-s})}\,\mathrm{d}t
\end{aligned}$$

A match can be obtained by first setting

$$\begin{aligned}
&2^{-s}\int_0^1 e^{\widehat{A}(k2^{-s}+t2^{-s})}\widehat{B}(k2^{-s}+t2^{-s})\widehat{B}^*(k2^{-s}+t2^{-s})e^{\widehat{A}^*(k2^{-s}+t2^{-s})}\,\mathrm{d}t \\
&\quad = r_q^k(A_s)\bigg(\int_0^1 e^{tA_s}(e^{F_t(A_s)}B_s)(e^{F_t(A_s)}B_s)^* e^{tA_s^*}\,\mathrm{d}t\bigg) r_q^k(A_s^*) \\
&\quad = 2^{-s}e^{k2^{-s}A+kF_1(A_s)}\bigg(\int_0^1 e^{t2^{-s}A}(e^{F_t(A_s)}B)(e^{F_t(A_s)}B)^* e^{2^{-s}tA^*}\,\mathrm{d}t\bigg)e^{k2^{-s}A^*+kF_1^*(A_s)},
\end{aligned}$$

for $k = 0, 1, \ldots, 2^s-1$, and then setting

$$e^{\widehat{A}(k2^{-s}+t2^{-s})}\widehat{B}(k2^{-s}+t2^{-s}) = e^{k2^{-s}A+kF_1(A_s)+t2^{-s}A}e^{F_t(A_s)}B,$$

for $t \in [0,1)$, or by change of variables $t \mapsto k2^{-s}+t2^{-s}$

$$e^{\widehat{A}(t)}\widehat{B}(t) = e^{tA+kF_1(A_s)}e^{F_{2^s t-k}(A_s)}B, \quad t \in [k2^{-s}, (k+1)2^{-s}). \tag{19}$$

From this, the following backwards result for the computed Gramian is found.

Proposition 2. *Let $\|V_t(A_s)\| < 1$ on $[0,1]$, then*

$$\widehat{G}(A,B) = \mathcal{G}(At+\Delta A, B+\Delta B),$$

where

$$\Delta A(t) = kF(1, A_s), \tag{20a}$$

$$\Delta B(t) = (e^{F_{2^s t-k}(\cdot, A_s)} - \mathrm{I})B, \tag{20b}$$

for $t \in [k2^{-s}, (k+1)2^{-s})$ and $k = 1, \ldots, 2^s-1$. Furthermore, the relative errors $\Delta A(t)$ and $\Delta B(t)$, respectively, are bounded by

$$\frac{\sup_{t\in[0,1]}\|\Delta A(t)\|}{\sup_{t\in[0,1]}\|tA\|} \le 2^s\frac{\|F_1(A_s)\|}{\|A\|}, \tag{21a}$$

$$\frac{\sup_{t\in[0,1]}\|\Delta B(t)\|}{\sup_{t\in[0,1]}\|B\|} \le \sup_{t\in[0,1]}\|V_t(A_s)\|. \tag{21b}$$

Proof. That $\widehat{A}(t) = tA + \Delta A(t)$ and $\widehat{B}(t) = B + \Delta B(t)$ satisfy the matching condition (19) is readily verified. Furthermore, a bound on the relative error of $\Delta A(t)$ is given by

$$\frac{\sup_{t\in[0,1]} \|\Delta A(t)\|}{\sup_{t\in[0,1]} \|tA\|} = \frac{\max_k \sup_{t\in[k2^{-s},(k+1)2^{-s}]} \|kF_1(A_s)\|}{\|A\|} = \frac{\max_k \|kF_1(A_s)\|}{\|A\|}$$
$$= (2^s - 1)\frac{\|F_1(A_s)\|}{\|A\|} \leq 2^s \frac{\|F_1(A_s)\|}{\|A\|}.$$

Finally, a bound for the relative error in ΔB is found by

$$\frac{\sup_{t\in[0,1]} \|\Delta B(t)\|}{\sup_{t\in[0,1]} \|B\|} = \max_k \sup_{t\in[k2^{-s},(k+1)2^{-s}]} \|(e^{F_{2^s t-k}(A_s)} - \mathrm{I})B\|$$
$$= \frac{\sup_{t\in[0,1]} \|(e^{F_t(A_s)} - \mathrm{I})B\|}{\|B\|}$$
$$\leq \sup_{t\in[0,1]} \|e^{F_t(A_s)} - \mathrm{I}\| = \sup_{t\in[0,1]} \|V_t(A_s)\|.$$

□

3.2 Controlling the backward error of the computed Gramian

In view of Propositions 1 and 2, it remains to find a maximal value η_q such that $\|A_s\| < \eta_q$ implies

$$\sup_{t\in[0,1]} \|V_t(A_s)\| < 2^{-53}. \tag{22}$$

The scaling parameter s may then be selected as

$$s = \left\lceil \log_2 \frac{\|A\|}{\min(\eta_q, \theta_q)} \right\rceil, \tag{23}$$

to ensure that both the errors in the computed matrix exponential and the Gramian are at most unit roundoff (in double precision arithmetic). Before proceeding, the following result giving a more explicit expression for V_t is required.

Proposition 3. *For the last coefficient C_q in the Legendre expansion of $e^{A_s t}$ it holds that*

$$C_q = \frac{q!}{(2q)!}(-A_s)^q D_q^{-1}(A_s). \tag{24}$$

Furthermore, an explicit expression for V is given by

$$V_t(A_s) = \frac{q!}{(2q)!}(-A_s)^{q+1} D_q^{-1}(A_s) \int_0^t e^{-A_s\tau} P_q(\tau)\,\mathrm{d}\tau. \tag{25}$$

Proof. Since $\widehat{E}_q(1) = r_q(A_s)$ it holds that

$$A_s C_q \int_0^1 e^{A_s(1-\tau)} P_q(\tau)\,\mathrm{d}\tau = e^{A_s} - r_q(A_s) = \frac{(-1)^q}{(2q)!} A_s^{2q+1} D_q^{-1}(A_s) \int_0^1 e^{A_s\tau}(1-\tau)^q \tau^q\,\mathrm{d}\tau,$$

where the last equality is the Padé remainder (Higham, 2008). By Rodrigues' formula the term $P_q(\tau)$ can be replaced by $\frac{1}{q!}\frac{\mathrm{d}^q}{\mathrm{d}\tau^q}(\tau^2 - \tau)^q$. Repeated integration by parts and reversing the integration interval, thus shows that the left-hand side is given by

$$A_s C_q \int_0^1 e^{A_s(1-\tau)} \frac{1}{q!}\frac{\mathrm{d}^q}{\mathrm{d}\tau^q}(\tau^2 - \tau)^q\,\mathrm{d}\tau = A_s C_q \frac{A_s^q}{q!} \int_0^1 e^{A_s(1-\tau)}(\tau^2 - \tau)^q\,\mathrm{d}\tau$$
$$= A_s C_q \frac{A_s^q}{q!} \int_0^1 e^{A_s\tau}\tau^q(1-\tau)^q\,\mathrm{d}\tau.$$

Consequently, the above equality reads

$$A_s C_q \frac{A_s^q}{q!} \int_0^1 e^{A_s\tau}\tau^q(1-\tau)^q \,\mathrm{d}\tau = \frac{(-1)^q}{(2q)!} A_s^{2q+1} D_q^{-1}(A_s) \int_0^1 e^{A_s\tau}(1-\tau)^q\tau^q \,\mathrm{d}\tau,$$

and the statement follows by matching terms. □

Proposition 3 ensures that V_t is analytic in A_s in the neighbourhood $\|A_s\| < \nu_q$. Consequently, it has an absolutely convergent power series expansions, which is more conveniently expressed using the following auxiliary function

$$\psi(t,\eta) = \frac{q!}{(2q)!}(-\eta)^{q+1} D_q^{-1}(\eta) e^{-\eta t}, \tag{26}$$

so that

$$V_t(A_s) = \int_0^t \psi(\tau, A_s) P_q(\tau) \,\mathrm{d}\tau.$$

The function ψ is analytic in the same region as V_t is. Therefore,

$$\psi(t,\eta) = \sum_{n=q+1}^{\infty} \psi_n(t)\eta^n,$$

and a bound on the norm of V_t is obtained by

$$\bar{V}_n(t) = \left| \int_0^t \psi_n(\tau) P_q(\tau) \,\mathrm{d}\tau \right|, \tag{27a}$$

$$\bar{V}(t,\eta) = \sum_{n=q+1}^{\infty} \bar{V}_n(t)\eta^n, \tag{27b}$$

$$\beta(\eta) = \sup_{t\in[0,1]} \bar{V}(t,\eta), \tag{27c}$$

$$\sup_{t\in[0,1]} \|V_t(A_s)\| \le \beta(\|A_s\|). \tag{27d}$$

It is clear that $t \mapsto \bar{V}(t,\|A_s\|)$ has extrema at the zeros $z_1, z_2, \ldots, z_q$ of the Legendre polynomial P_q. However, it is not clear that these are the only extrema, even though numerical experiments certainly suggest this is the case. In any case, define $z_0 = 0$ and $z_{q+1} = 1$ and form grids constructed by uniformly placing p points in the intervals $(z_0, z_1), \ldots, (z_q, z_{q+1})$, totalling $(q+1)(p-1)$ unique points. Let $t_1, \ldots, t_{(q+1)(p-1)}$ be the union of these grids, and approximate β by

$$\beta(\eta) \approx \hat{\beta}(\eta) = \max_i \bar{V}(t_i, \eta). \tag{28}$$

Maximal positive numbers η_q for $q = 1, \ldots, 21$ such that $\hat{\beta}(\eta) \le 2^{-53}$ are computed, for $p = 2, 2^6, 2^7$, in the Julia programming language (Bezanson et al., 2017), by using arbitrary precision arithmetic, truncating the sum (27b) at the 150th order term, and computing the coefficients $\bar{V}_n$ by Taylor mode automatic differentiation (Benet and Sanders, 2019). The results are tabulated in table 3.2 along with the quantities θ_q, ν_q, and ξ_q obtained by (Higham, 2005). As the maximal η_q were the same for all selected p only the result for $p = 2$ is presented. Additionally, the series η_q and θ_q for $q = 1, \ldots, 21$ are drawn in Figure 3.2. It is evident that ensuring that the Gramian is computed to unit roundoff precision implies a more aggressive scaling of A, particularly for $2 \le q \le 11$, while for $q = 1$ and $q \ge 12$ only 2 additional downscalings are required.

Table 1: Maximal values θ_q and η_q of $\|2^{-s}A\|$ such that $2^s\|F_1(A_s)\| \leq 2^{-53}$ and $\sup_{t\in[0,1]}\|V_t(A_s)\| \leq 2^{-53}$, respectively, $\nu_q = \min\{|z| : D_q(z) = 0\}$, and upper bound ξ_q for $\|D_q^{-1}(A)\|$.

q	1	2	3	4	5	6	7	8	9	10
θ_q	3.7e-8	5.3e-4	1.5e-2	8.5e-2	2.5e-1	5.4e-1	9.5e-1	1.5e0	2.1e0	2.8e0
η_q	1.8e-8	2.4e-5	6.7e-4	5.3e-3	2.1e-2	6.0e-2	1.3e-1	2.4e-1	4.1e-1	6.2e-1
ν_q	2.0e0	3.5e0	4.6e0	6.0e0	7.3e0	8.7e0	9.9e0	1.1e1	1.3e1	1.4e1
ξ_q	1.0e0	1.0e0	1.0e0	1.0e0	1.1e0	1.3e0	1.6e0	2.1e0	3.0e0	4.3e0
q	11	12	13	14	15	16	17	18	19	20	21
θ_q	3.6e0	4.5e0	5.4e0	6.3e0	7.3e0	8.4e0	9.4e0	1.1e1	1.2e1	1.3e1	1.4e1
η_q	8.9e-1	1.2e0	1.5e0	1.9e0	2.4e0	2.9e0	3.4e0	4.0e0	4.6e0	5.2e0	5.8e0
ν_q	1.5e1	1.7e1	1.8e1	1.9e1	2.1e1	2.2e1	2.3e1	2.5e1	2.6e1	2.7e1	2.8e1
ξ_q	6.6e0	1.0e1	1.7e1	3.0e1	5.3e1	9.8e1	1.9e2	3.8e2	8.3e2	2.0e3	6.2e3

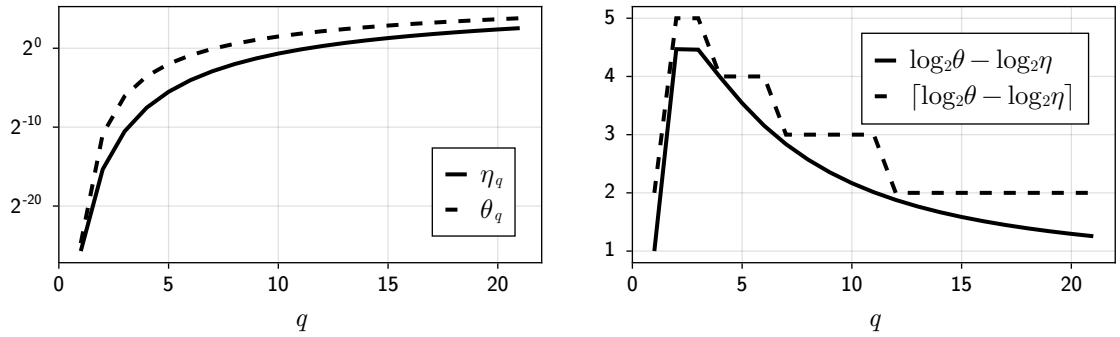

Figure 1: The series θ_q and η_q for $q = 1, \ldots, 21$ (right), and the difference in the scaling parameter s when selected to satisfy $\|A_s\| \leq \theta_q$ or $\|A_s\| \leq \eta_q$, respectively.

4 Rank properties of the approximate Gramian

The discussion has hitherto been centred on controlling the error. However, another important aspect is the rank properties of the Gramian. The pair (A, B) is said to be completely controllable if the controllability matrix

$$\mathcal{C}(A,B) = \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix}, \tag{29}$$

is of full rank, which is equivalent to the Gramian being of full rank (Anderson and Moore, 2007). In fact, $\mathcal{C}(A,B)$ and $G(A,B)$ share nullspace. From the perspective of applications in control and estimation it thus interesting to investigate whether the method produces an approximation, $\widehat{G}$, that mathematically reproduces the controllability properties of (A,B). It follows from the variation of constants representation (18) and (12), that the computed Gramian may be written as $\widehat{G} = \widehat{\mathcal{C}}_{q,s}\widehat{\mathcal{C}}_{q,s}^*$ with

$$\tilde{U}_0^* = D_q^{-1}(A_s)\begin{bmatrix} L_0(A_s)B & \cdots & L_k(A_s)B/\sqrt{2k+1} & \cdots & L_q(A_s)B/\sqrt{2q+1} \end{bmatrix} \tag{30a}$$

$$\widehat{\mathcal{C}}_{q,s} = \begin{bmatrix} r_q(A_s)\tilde{U}_0^* & r_q^2(A_s)\tilde{U}_0^* & \cdots & r_q^{2^s-1}(A_s)\tilde{U}_0^*. \end{bmatrix} \tag{30b}$$

The computed Gramian resembles the outer product of the controllability matrix with itself, except for the fact that the matrix is formed with a basis different from the monomial one. More specifically, for $k = 0, 1, \ldots, q$ and $m = 0, 1, \ldots, 2^s - 1$, define the functions

$$e_{k,m}(z) = \frac{1}{\sqrt{(2k+1)2^s}} r_q^m(z2^{-s}) D_q^{-1}(zs^{-s}) L_k(zs^{-s}). \tag{31}$$

Then $\widehat{\mathcal{C}}_{q,s}$ is a block matrix consisting of the following blocks:

$$e_{k,m}(A)B, \quad k = 0, 1, \ldots, q, \quad m = 0, 1, \ldots, 2^s - 1.$$

The polynomials L_k are of degree at most q, from which it follows that the functions $\tilde{e}_{k,m}(z) = D_q^{2^s}(z2^{-s})e_{k,m}(z)$ are polynomials of at most degree $2^s q$. If there is an s such that they span the space of polynomials of degree at most $n-1$, then there exists an invertible matrix T such that

$$D_q^{2^s}(A_s)\widehat{\mathcal{C}}_{q,s}T = \begin{bmatrix} \mathcal{C}(A,B) & 0 \end{bmatrix}, \tag{32}$$

and consequently, the rank properties in the computed Gramian are the same as in the exact Gramian. In order to find such an s, the following assumption is required.

Assumption 1. *The polynomials $L_0, L_1, \ldots, L_q$ are linearly independent.*

The table of coefficients in Appendix B certainly verifies this assumption for $q = 3, 5, 7, 9, 13$. Furthermore, the following result on the zeros of N_q and D_q shall prove useful.

Lemma 2. *The polynomials N_q and D_q have no zeros in common.*

Proof. Ehle (1969, Theorem 2.1, p. 22) states that all the zeros of N_q are in the open left half plane. Therefore, by the well known relation, $D_q(z) = N_q(-z)$, all zeros of D_q are in the open right half plane, which gives the desired conclusion. □

It remains to study the span of the union of the following sets

$$\Pi^m = \left\{ \tilde{e}_{0,m}(z), \tilde{e}_{1,m}(z), \ldots, \tilde{e}_{q,m}(z) \right\}, \quad m = 0, 1, \ldots, 2^s - 1. \tag{33}$$

Lemma 3. *Let Assumption 1 hold. Then* Π^m *are sets of linearly independent functions for* $m = 0, 1, \ldots, 2^s - 1$.

Proof. Let v_k be some coefficients for the expansion of the zero function in the set Π^m, that is

$$0 = \sum_{k=0}^{q} v_k r_q^m(z) L_k(z) D_q^{2^s-1}(z) \iff 0 = \sum_{k=0}^{q} v_k L_k(z),$$

which by assumption is equivalent to $v_k = 0$ for $k = 0, 1, \ldots, q$. □

Proposition 4. *For the sets* $\Pi^m, \quad m = 0, 1, \ldots, 2^s - 1$, *the following holds:*

$$\dim \operatorname{span} \cup_{m=0}^{2^s-1} \Pi^m = q2^s + 1.$$

Proof. The idea is to show that $\operatorname{span} \Pi^m$ and $\operatorname{span} \Pi^{m+l}$ for $l \geq 1$ can only intersect for $l = 1$ and that the dimension of this intersection is 1. The conclusion is then obtained by use of Lemma 3. Expanding the zero function in the set $\Pi^m \cup \Pi^{m+l}$ gives

$$0 = \sum_{k=0}^{q} v_k r_q^m(z) L_k(z) D_q^{2^s-1}(z) + \sum_{k=0}^{q} w_k r_q^{m+l}(z) L_k(z) D_q^{2^s-1}(z),$$

which is equivalent to

$$0 = D_q^l(z) \sum_{k=0}^{q} v_k L_k(z) + N_q^l(z) \sum_{k=0}^{q} w_k L_k(z). \tag{34}$$

D_q and N_q are polynomials of degree q, which by Lemma 2 have no zeros in common. Therefore (34) is impossible to satisfy for $l \geq 2$ unless $v_k = w_k = 0$ for $k = 0, 1, \ldots, q$. For $l = 1$ the only possibility is that, for some arbitrary constant c,

$$\sum_{k=0}^{q} v_k L_k(z) = \pm c N_q(z), \tag{35a}$$

$$\sum_{k=0}^{q} w_k L_k(z) = \mp c D_q(z). \tag{35b}$$

which concludes the proof. □

The consequence of Proposition 4 is that the number of squarings s, needs to satisfy

$$s \geq \left\lceil \log_2 \frac{n-1}{q} \right\rceil, \tag{36}$$

for the computed Gramian to have the same rank as the exact one.

5 Design of algorithm and numerical experiments

The goal of this section is to arrive at a final design of the algoritm, with the discussion in sections 3 and 4 in mind. Other than achieving a backward error of at most unit round-off and a computed Gramian that preserves the controllability properties of (A, B), other design criteria involve minimizing the number of squarings to avoid the over-scaling phenomena, while also staying as close as possible to the conventional algorithm for the matrix exponential. Therefore, only the orders $q = 3, 5, 7, 9, 13$ are considered.

Pre-processing. For the input matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$, it is assumed that $m \leq n$. This is not an unreasonable assumption, as otherwise B would be overparametrized. More specifically, if $m > n$, then the QR decomposition of B^* is given by

$$B^* = Q_{B^*}\widetilde{B}^*, \tag{37}$$

where $Q_{B^*} \in \mathbb{R}^{m\times n}$ and $\widetilde{B}^* \in \mathbb{R}^{n\times n}$. It is evident from the definition (1b) that

$$G(A, B) = G(A, \widetilde{B}).$$

It is therefore reasonable to assume that $m \leq n$, at least after initial pre-processing of the matrix B. Furthermore, the Gramian is invariant under similarity transforms. That is, given an invertible matrix T,

$$G(TAT^{-1}, TB) = TG(A, B)T^{-1}. \tag{38}$$

It is common in implementations of the matrix exponential to select T as a so-called balancing transform. However, the discussion of (Al-Mohy and Higham, 2011, p. 496) recommends that balancing should not be used by default, and is therefore not included as a pre-processing step of the algorithm proposed here.

Order adaptation. In view of the conclusions of the error analysis of Section 3, modifications of the order adaption in the conventional algorithm (Higham, 2005) are required. Namely, as $\eta_q < \theta_q$ for $1 \leq q \leq 21$, the scaling parameter needs to be selected as

$$s \geq \lceil \log_2(\|A\|/\eta_q) \rceil. \tag{39}$$

Furthermore, in view of the discussion in Section 4, the scaling parameter also needs to satisfy the bound (36). Therefore, in order to avoid the over-scaling phenomena, it appears numerically advantageous to simply select the smallest q that satisfies

$$\|A\| \leq \eta_q, \tag{40a}$$
$$n \leq q + 1, \tag{40b}$$

for $q = 3, 5, 7, 9$. If no such q is found then the order 13 method is used and the scaling parameter is selected as

$$s = \left\lceil \log_2 \max\left(\frac{\|A\|}{\eta_q}, \frac{n-1}{q}\right)\right\rceil. \tag{41}$$

Implementing the initial values. From the table of coefficients, the polynomials L_k are even for even k and odd for odd k for $q = 3, 5, 7, 9, 13$. Consequently, the same evaluation strategy as used by Higham (2005) may be adopted, and the initial Cholesky factor of the Gramian may be accumulated from B and A^2B for $q = 3, 5, 7, 9$. Similarly, for $q = 13$, the initial Cholesky factor is accumulated from B, A^2B, A^4B, A^6B.

5.1 Numerical experiments

The numerical performance of the proposed method is examined on a series of test pairs (A, B). The backward error analysis of Proposition 2 suggests that B is of little importance and may thus be chosen arbitrarily. Suitable collections of test matrices for A shall later be defined in various ways. Unlesss otherwise stated, a ground-truth is obtained by computing the Gramian via the matrix fraction decompositon (Axelsson and Gustafsson, 2014) in

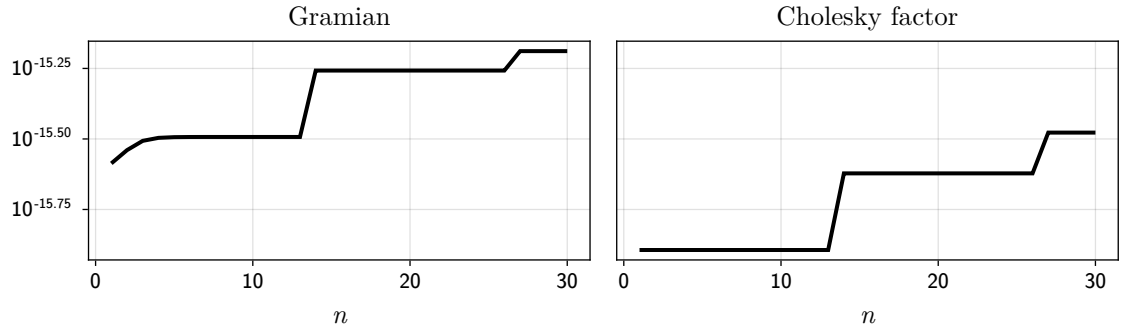

Figure 2: The results of experiment 0. The relative error in the computed Gramians (left) and Cholesky factors (right).

arbitrary precision (Fousse et al., 2007), and projecting the result on the set of Hermitian matrices. The matrix exponential for the ground-truth is computed using the software package ExponentialUtilities.jl [4] (Rackauckas and Nie, 2017) with the generic method using a Padé approximation of order 13. All relative errors are computed in the 1-norm.

Experiment 0. The matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times 1}$ are defined as

$$A_{i,j} = \begin{cases} 1, & \text{if } j+1 = i, \\ 0, & \text{otherwise} \end{cases} \qquad \text{and} \qquad B_i = \begin{cases} 1, & \text{if } i = 1, \\ 0, & \text{otherwise} \end{cases} .$$

This experiment serves as a "unit test" in the sense that A is nilpotent of index n and the base approximations of order q are exact for nilpotent matrices of order $q+1$. Furthermore, as $e^{At}B = \left[1, \quad t, \quad t^2/2, \quad \cdots, \quad t^{n-1}/(n-1)!\right]^*$, the Gramian and its Cholesky factor can be computed in closed form by switching to the Legendre basis. The dimension n ranges from 2 to 30.

The relative errors are shown in Figure 2. It can be seen that the proposed algorithm performs to an acceptable precision, just as predicted by the error analysis. Furthermore, the error in the computed Cholesky factor is almost half an order of magnitude smaller than that of the Gramian.

Experiment 1. A is selected from a collection of 10×10 matrices provided by a subset of the "builtin" matrices of the software package MatrixDepot.jl (Zhang and Higham, 2016). The matrices that were excluded were either sparse matrices, not conforming to the general API, or matrices with positive eigenvalues and very large norms, for which the excessive amount of doublings lead to numerical problems. The full list of excluded matrices is given in Appendix A. The matrix B is selected as a $10 \times m$ matrix with $m = 1, 5, 10$, and the elements drawn independently from the standard Normal distribution. The experiment is conducted 50 times for each selection of m so that the effect of randomization can be assessed. Scatter plots of the relative errors over all simulations are shown in Figure 3. It is again evident that the proposed algorithm performs in accordance with expectation. One exception is the matrix numbered 22. This matrix is known as, `"invol"`, it has positive eigenvalues equal to one and a 1-norm resulting in 25 doublings, which is problematic but

[4] The default implementation of the matrix exponential in LinearAlgebra.jl precludes the use of arbitrary precision floats.

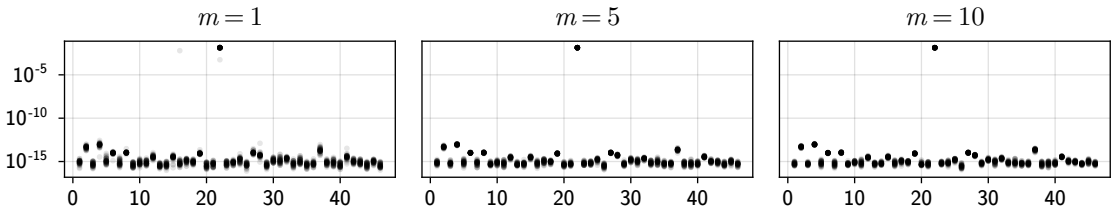

Figure 3: The results of experiment 1. Every value on the horizontal axis corresponds to a specific matrix A from the builtin data set in MatrixDepot.jl. For each A, there are 50 grey dots, each corresponding to the relative error in the computed Gramian for a single randomly generated matrix B.

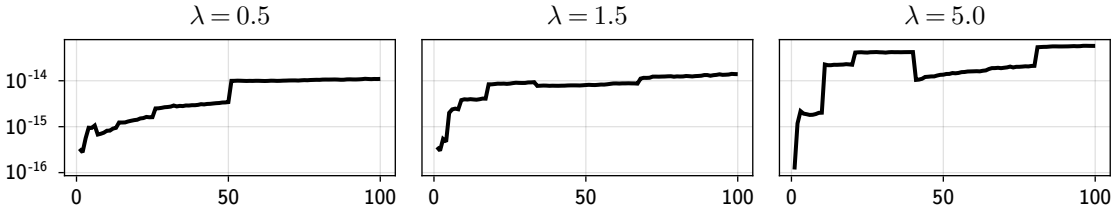

Figure 4: The results of experiment 2. The relative error in the computed Gramians, plotted against the dimension of the Laguerre network (42).

does not result in complete failure. The result also demonstrates that the algorithm is rather insensitive to the selection of B, except for a few outliers in the case $m = 1$.

Experiment 2. The matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times 1}$ are defined by

$$A_{i,j} = \begin{cases} -2\lambda, & i > j, \\ -\lambda, & i = j, \\ 0, & i < j \end{cases}, \tag{42a}$$

$$B = \sqrt{2\lambda}\begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^*. \tag{42b}$$

This is a so called Laguerre network. The parameter λ is a positive number that is selected from $\{1.0, 2.5, 5.0\}$ and n ranges from 1 to 100. A is Hurwitz, and it is readily verified that the identity matrix solves the algebraic Lyapunov equation associated with (A, B),

$$A + A^* = -BB^*.$$

Consequently, the finite horizon Gramian may be computed by the formula (Farrell and Livstone, 1993, Lemma 1)

$$G(A, B) = I - e^A e^{A^*}. \tag{43}$$

This formula is used to compute a reference solution in arbitrary precision and the results are shown in Figure 4. Whereas there is a loss of precision as the dimension grows, the resulting error appears to be acceptable up to dimension at least 100.

6 Conclusions

In this article, a "scaling and squaring" method has been developed for computing the Cholesky factor of the finite horizon Gramian associated with the pair of matrices (A, B).

The method computes the matrix exponential, e^A, in almost the same manner as the conventional algorithm (Higham, 2005). Furthermore, a backward error analysis was made, which ensures that both the matrix exponential and the Gramian are computed to within unit roundoff in double precision arithmetic. As the error analysis and algorithm design piggybacks on the development of the conventional algorithm for computing the matrix exponential, it is expected that the algorithm for the Cholesky factor of the Gramian will have similar numerical performance in practice. This has indeed been demonstrated through a set of experiments. The doubling recursion for both the matrix exponential and the Gramian both require one matrix multiplication each, and the former additionally requires a QR decomposition. Consequently, the doubling phase is expected to be equally problematic for both quantities. Nevertheless, in the numerical experiments the algorithm has peformed to satisfaction.

Acknowledgements

TS and FT were partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

Abou-Kandil, H., Freiling, G., Ionescu, V., and Jank, G. (2012). *Matrix Riccati Equations in Control and Systems Theory*. Birkhäuser.

Al-Mohy, A. H. and Higham, N. J. (2009). Computing the Fréchet derivative of the matrix exponential, with an application to condition number estimation. *SIAM J. Matrix Anal. Appl.*, 30(4):1639–1657.

Al-Mohy, A. H. and Higham, N. J. (2010). A new scaling and squaring algorithm for the matrix exponential. *SIAM J. Matrix Anal. Appl.*, 31(3):970–989.

Al-Mohy, A. H. and Higham, N. J. (2011). Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM journal on Scientific Computing*, 33(2):488–511.

Anderson, B. D. O. and Moore, J. B. (2007). *Optimal Control: Linear Quadratic Methods*. Courier Corporation.

Anderson, B. D. O. and Moore, J. B. (2012). *Optimal Filtering*. Courier Corporation.

Antoulas, A. C. (2005). *Approximation of Large-scale Dynamical Systems*. SIAM.

Axelsson, P. and Gustafsson, F. (2014). Discrete-time solutions to the continuous-time differential Lyapunov equation with applications to Kalman filtering. *IEEE Transactions on Automatic Control*, 60(3):632–643.

Bartels, R. H. and Stewart, G. W. (1972). Algorithm 432: Solution of the matrix equation AX + XB = C. *Commun. ACM*, 15(9):820–826.

Behr, M., Benner, P., and Heiland, J. (2019). Solution formulas for differential Sylvester and Lyapunov equations. *Calcolo*, 56:51.

Benet, L. and Sanders, D. P. (2019). TaylorSeries.jl: Taylor expansions in one and several variables in Julia. *J. Open Source Softw.*, 4(36):1043.

Benner, P., Li, J.-R., and Penzl, T. (2008). Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems. *Numer. Linear Algebra Appl.*, 15(9):755–777.

Benner, P. and Mena, H. (2018). Numerical solution of the infinite-dimensional LQR problem and the associated Riccati differential equations. *J. Numer. Math.*, 26(1):1–20.

Benner, P. and Saak, J. (2013). Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: a state of the art survey. *GAMM-Mitt.*, 36(1):32–52.

Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Rev.*, 59(1):65–98.

Ehle, B. L. (1969). *On Padé approximations to the exponential function and A-stable methods for the numerical solution of initial value problems*. PhD thesis, University of Waterloo Waterloo, Ontario.

Farrell, J. and Livstone, M. (1993). Exact calculations of discrete-time process noise statistics for hybrid continuous/discrete time applications. In *Proceedings of 32nd IEEE Conference on Decision and Control*, pages 857–858. IEEE.

Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., and Zimmermann, P. (2007). MPFR: A multiple-precision binary floating-point library with correct rounding. *ACM Trans. Math. Softw.*, 33(2):13–es.

Gowda, S., Ma, Y., Cheli, A., Gwóźdź, M., Shah, V. B., Edelman, A., and Rackauckas, C. (2022). High-performance symbolic-numerics via multiple dispatch. *ACM Commun. Comput. Algebra*, 55(3):92–96.

Güttel, S. and Nakatsukasa, Y. (2016). Scaled and squared subdiagonal Padé approximation for the matrix exponential. *SIAM J. Matrix Anal. Appl.*, 37(1):145–170.

Hammarling, S. J. (1982). Numerical solution of the stable, non-negative definite Lyapunov equation. *IMA J. Numer. Anal.*, 2(3):303–323.

Higham, N. J. (2005). The scaling and squaring method for the matrix exponential revisited. *SIAM J. Matrix Anal. Appl.*, 26(4):1179–1193.

Higham, N. J. (2008). *Functions of matrices: theory and computation*. SIAM.

Kailath, T., Sayed, A. H., and Hassibi, B. (2000). *Linear estimation*. Prentice Hall.

Kirsten, G. and Simoncini, V. (2020). Order reduction methods for solving large-scale differential matrix Riccati equations. *SIAM J. Sci. Comput.*, 42(4):A2182–A2205.

Koskela, A. and Mena, H. (2020). Analysis of Krylov subspace approximation to large-scale differential Riccati equations. *Electron. Trans. Numer. Anal.*, 52:431–454.

Li, D., Zhang, X., and Liu, R. (2021). Exponential integrators for large-scale stiff Riccati differential equations. *J. Comput. Appl. Math.*, 389:113360.

Li, J.-R. and White, J. (2002). Low rank solution of Lyapunov equations. *SIAM J. Matrix Anal. Appl.*, 24(1):260–280.

Mena, H. (2007). *Numerical Solution of Differential Riccati Equations Arising in Optimal Control of Partial Differential Equations*. PhD thesis, Escuela Politécnica Nacional, Quito, Ecuador. Available as ISBN: 978-9978-383-09-4.

Moler, C. and Van Loan, C. (1978). Nineteen dubious ways to compute the exponential of a matrix. *SIAM Rev.*, 20(4):801–836.

Moler, C. and Van Loan, C. (2003). Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Rev.*, 45(1):3–49.

Moore, G. (2011). Orthogonal polynomial expansions for the matrix exponential. *Linear algebra appl.*, 435(3):537–559.

Ostermann, A., Piazzola, C., and Walach, H. (2019). Convergence of a low-rank Lie-Trotter splitting for stiff matrix differential equations. *SIAM J. Numer. Anal.*, 57(4):1947–1966.

Rackauckas, C. and Nie, Q. (2017). DifferentialEquations.jl – A performant and feature-rich ecosystem for solving differential equations in Julia. *The Journal of Open Research Software*, 5(1).

Simoncini, V. (2016). Computational methods for linear matrix equations. *SIAM Rev.*, 58(3):377–441.

Stillfjord, T. (2015). Low-rank second-order splitting of large-scale differential Riccati equations. *IEEE Trans. Automat. Control*, 60(10):2791–2796.

Stillfjord, T. (2018). Adaptive high-order splitting schemes for large-scale differential Riccati equations. *Numer. Algorithms*, 78(4):1129–1151.

Zhang, W. and Higham, N. J. (2016). Matrix Depot: an extensible test matrix collection for Julia. *PeerJ Computer Science*, 2.

A Additional information on experiments

As pointed out in the main text, some matrices from MatrixDepot.jl were excluded from experiment 1. It was the following matrices:

```
  [
    "blur",
    "hadamard",
    "phillips",
    "rosser",
    "neumann",
    "parallax",
    "poisson",
    "wathen",
    "invhilb",
    "vand",
    "golub",
    "magic",
    "pascal",
]
```

The latter five were problematic in the sense of having eigenvalues with positive real part and very large norms. The former matrices were excluded on the grounds of not conforming with the general API and were usually sparse matrices.

B Coefficient tables for the Legendre expansion of the Matrix exponential

In this section, the necessary quantities to implement the initial approximation of the matrix exponential and the Gramian are listed for $q = 3, 5, 7, 9, 13$. Recall that the initial approximation of the matrix exponential is given by the diagonal Padé approximant

$$r_q(z) = \frac{N_q(z)}{D_q(z)} = \frac{\tilde{N}_q(z)}{\tilde{D}_q(z)},$$

where N_q and D_q are the Padé numerator and denominator, respectively. The numerator $\tilde{N}_q$ and denominator $\tilde{D}_q$ are scaled versions so that all coefficients are integers. The coefficients of $\tilde{N}_q$ are listed as `pade_num`. Furthemore, the coefficients $C_k(z)$ are given by

$$C_k(z) = \frac{\tilde{L}_k(z)}{\tilde{D}_q(z)},$$

where $\tilde{L}_k$ are polynomials whose coefficients are listed as the rows of the matrix referred to as `leg_nums`. They are rescaled versions of L_k, such that $\frac{\tilde{L}_k(z)}{\tilde{D}_q(z)} = \frac{L_k(z)}{D_q(z)}$. Lastly, the square norms of the Legendre polynomials are listed as `sqr_norms`, that is $1, 3, \ldots, 2k + 1, \ldots, 2q + 1$.

B.1 Coefficient tables for $q = 3$

```
pade_num = [120, 60, 12, 1]
leg_nums = [120 0 2 0; 0 60 0 0; 0 0 10 0; 0 0 0 1]
sqr_norms = [1, 3, 5, 7]
```

B.2 Coefficient tables for $q = 5$

```
pade_num = [30240, 15120, 3360, 420, 30, 1]
leg_nums =
[
30240 0 840 0 2 0
0 15120 0 168 0 0
0 0 2520 0 10 0
0 0 0 252 0 0
0 0 0 0 18 0
0 0 0 0 0 1
]
sqr_norms = [1, 3, 5, 7, 9, 11]
```

B.3 Coefficient tables for $q = 7$

```
pade_num = [17297280, 8648640, 1995840, 277200, 25200, 1512, 56, 1]
leg_nums =
[
17297280 0 554400 0 3024 0 2 0
0 8648640 0 133056 0 324 0 0
0 0 1441440 0 11880 0 10 0
0 0 0 144144 0 616 0 0
0 0 0 0 10296 0 18 0
0 0 0 0 0 572 0 0
```

```
0 0 0 0 0 0 26 0
0 0 0 0 0 0 0 1
]
sqr_norms = [1, 3, 5, 7, 9, 11, 13, 15]
```

B.4 Coefficient tables for $q = 9$

```
pade_num =
[
17643225600,
8821612800,
2075673600,
302702400,
30270240,
2162160,
110880,
3960,
90,
1,
]
leg_nums =
[
17643225600 0 605404800 0 4324320 0 7920 0 2 0
0 8821612800 0 155675520 0 617760 0 528 0 0
0 0 1470268800 0 15444000 0 34320 0 10 0
0 0 0 147026880 0 960960 0 1092 0 0
0 0 0 0 10501920 0 42120 0 18 0
0 0 0 0 0 583440 0 1320 0 0
0 0 0 0 0 0 26520 0 26 0
0 0 0 0 0 0 0 1020 0 0
0 0 0 0 0 0 0 0 34 0
0 0 0 0 0 0 0 0 0 1
]
sqr_norms = [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
```

B.5 Coefficient tables for $q = 13$

```
pade_num =
[
64764752532480000,
32382376266240000,
7771770303897600,
1187353796428800,
129060195264000,
10559470521600,
670442572800,
33522128640,
1323241920,
40840800,
960960,
16380,
182,
1,
]
leg_nums =
[
64764752532480000 0 2374707592857600 0 21118941043200 0 67044257280 0 81681600 0 32760 0 2 0
0 32382376266240000 0 647647525324800 0 3620389893120 0 7449361920 0 5569200 0 1080 0 0
0 0 5397062711040000 0 69390806284800 0 260727667200 0 352716000 0 153000 0 10 0
0 0 0 539706271104000 0 4797389076480 0 12443820480 0 10852800 0 2380 0 0
0 0 0 0 38550447936000 0 245321032320 0 439538400 0 232560 0 18 0
0 0 0 0 0 2141691552000 0 9884730240 0 11938080 0 3344 0 0
0 0 0 0 0 0 97349616000 0 324498720 0 248976 0 26 0
0 0 0 0 0 0 0 3744216000 0 8809920 0 3780 0 0
0 0 0 0 0 0 0 0 124807200 0 197064 0 34 0
0 0 0 0 0 0 0 0 0 3670800 0 3496 0 0
0 0 0 0 0 0 0 0 0 0 96600 0 42 0
0 0 0 0 0 0 0 0 0 0 0 2300 0 0
```

```
0 0 0 0 0 0 0 0 0 0 0 0 50 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1
]
sqr_norms = [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27]
```