On real logarithms of nearby matrices and structured matrix interpolation

Luca Dieci,

School of Mathematics, Georgia Institute of Technology, Atlanta, Georgia 30332, U.S.A. E-mail: **dieci@math.gatech.edu**

Benedetta Morini, Alessandra Papini and Aldo Pasquali

Dep. Energetica S. Stecco, Univ. of Florence, via C. Lombroso 6-17, 50134 Florence, Italy E-mail: **papini@de.unifi.it**

Abstract

Theoretical and algorithmic results are given for the numerical computation of real logarithms of nearby matrices. As an application, and an original motivation for this study, interpolation for sequences of invertible matrices is considered particularly for matrices with a given structure (for example, orthogonal, symplectic, or positive definite), so that the resulting interpolants share the structural properties of the data. Error analysis, implementation details and examples are provided.

1 Introduction

A frequent computational problem is to integrate matrix differential equations. Quite often, solutions of these matrix equations are matrices of a well defined type, for example, orthogonal, symplectic, or positive definite matrices. (In this work, a positive definite matrix is tacitly assumed symmetric.) Recall, $A \in \mathbb{R}^{2n\times 2n}$ is *Hamiltonian* if $A^TJ + JA = 0$, where $J = \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix}$, and is *symplectic* if $A^TJA = J$. In practice, good choices exist for discretizing these matrix equations, in particular guaranteeing that, at grid points, the approximations share the structural properties of the exact solution, that is, orthogonality, symplecticity, or positive definiteness. Thus, we consider a data

⋆ Supported in part under NSF Grant DMS-9625813, NATO Grant CRG 950865, and MURST and CNR Grants (Italy).

set $\{t_j, A_j\}$, $j = 0, 1, \ldots$, where $A_j \in \mathbb{R}^{n\times n}$ are invertible. Typically, $h_j := t_{j+1} - t_j$ is small. We can (and do) think of A_j as $A(t_j)$ for some smooth A.

An issue which has not received much attention is *structured matrix interpolation*. For the above data set, we need a representation of the data valid for all $t \in [t_0, t_N]$, and such that it reduces to A_j at t_j. Naturally, the resulting interpolant, $\tilde{A}$, should be a smooth real function, and $\|A(t) - \tilde{A}(t)\|$ should be nicely bounded (unless otherwise stated, $\|\cdot\|$ refers to the 2–norm). This is no different than interpolation for the entries of the matrices A_j, and in fact polynomial interpolation of matrices is common in engineering applications [1,10]. However, our interest is in interpolating so that $\tilde{A}(t)$ *retains the structure* of the A_j, that is that $\tilde{A}(t)$ is orthogonal, symplectic, or positive definite, if the A_j's are such. As far as we know, this problem has not been addressed in this generality. One possibility would be to simply interpolate, polynomially, the given matrices to give $P_A(t)$. Now, usually $P_A(t)$ does not share the structure of the data, so we may need to modify it appropriately. In the Frobenius norm, conceptually it is easy to find the closest positive semi-definite matrix to a symmetric one, as well as the closest orthogonal matrix to a given invertible matrix, so we could obtain (at each given t) a structurally "correct" answer in these cases. However, this does not usually give a smooth interpolant. Our idea for structured matrix interpolation is quite straightforward: (i) associate to the matrices A_k their real logarithms (which will be clarified below), and call these L_k, (ii) perform polynomial interpolation on the L_k, and call $\tilde{L}(t)$ the resulting interpolant, and then (iii) let $\tilde{A}(t) = e^{\tilde{L}(t)}$. Clearly, there are non-trivial issues to resolve: lack of uniqueness, computational expense, and potential numerical instabilities. This simple idea will be modified and extended. Note that, in step (i) we need to compute logarithms of matrices which are presumably close to one another. This motivates our first problem: how to compute logarithms of nearby matrices.

Logarithms. Because of our applications, we consider only real logarithms of real matrices; the derivation would simplify considerably if we could consider complex logarithms. By a real logarithm of A, we mean any matrix $X \in \mathbb{R}^{n\times n}$ such that $e^X = A$. Now, suppose we have two matrices, A_1 and A_2, with $\|A_2 - A_1\|$ small, and we have computed L_1, the logarithm of A_1. Intuitively, one would expect L_2, the logarithm of A_2, to be "close" to L_1. (It turns out that L_2 is not necessarily close to L_1.) Can one take advantage of knowledge of L_1 when computing L_2?

As far as we know, all previous work on computing the logarithm of a matrix A has focussed on the case of computing a logarithm for a given matrix [4,7,8]. Also, the focus has been on computing a logarithm which is a *primary matrix function* of A; i.e., a primary logarithm. Recall, a primary matrix function is a function on the spectrum, $\lambda(A)$, of the matrix A (see [5]). Moreover, computing a primary logarithm has always been restricted to the principal logarithm

of A with eigenvalues on the main branch; i.e., they have imaginary parts within $(-\pi, \pi)$. Sensitivity analysis for logarithms of matrices has only been completed for the principal logarithm. Recall, a principal logarithm admits the integral representation

$$L \;=\; \log(A) \;=\; \int_0^1 (A-I)[(A-I)t+I]^{-1}dt\,. \tag{1.1}$$

Finding a logarithm of a matrix presents intrinsic difficulties not present, for example, for the exponential of a matrix. Chief is the choice of the branch, which is not clear, except for positive definite matrices, where the principal logarithm is a natural choice. The principal branch might not be appropriate, for example, if we envision a continuation process. Intimately connected with this issue is whether or not we need a logarithm which is a primary matrix function. It is important, therefore, to understand what distinguishes a non-primary logarithm from a primary one. Since $\log(A)$ is any matrix X such that $e^X = A$, the following possibilities are typical. First a non-primary logarithm occurs when the eigenvalues of A are real and negative, but there is a real L such that $e^L = A$; e.g.,

$$A = -\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad L = \begin{pmatrix} 0 & \pi \\ -\pi & 0 \end{pmatrix}.$$

(Note, the eigenvalues of L are on the border of the main branch.) Another possibility occurs when, corresponding to a repeated eigenvalue of A, we take different values for the logarithm (that is, on different branches); necessarily, the end result is a non-primary logarithm.

Example 1.1 Consider the matrices

$$A(t) = e^{\alpha(t)} \begin{pmatrix} \cos(\beta(t)) & \sin(\beta(t)) \\ -\sin(\beta(t)) & \cos(\beta(t)) \end{pmatrix}, \quad L(t) = \begin{pmatrix} \alpha(t) & \beta(t) \\ -\beta(t) & \alpha(t) \end{pmatrix} \tag{1.2}$$

where $\beta(t) = 2\pi \sin t$, and α is a smooth function. If $\alpha(t) = 0$, $A(t)$ is orthogonal and symplectic. It is simple to show that $e^{L(t)} = A(t)$ for all t, thus $L(t)$ is a logarithm of $A(t)$. The eigenvalues of $A(t)$ and $L(t)$ are, respectively,

$$e^{\alpha(t)}[\cos(\beta(t)) \pm i \sin(\beta(t))] \qquad \text{and} \qquad \alpha(t) \pm i\beta(t)\,.$$

Clearly, $A(\pi/2) = e^{\alpha(\pi/2)}\, I$, while

$$L(\pi/2) = \begin{pmatrix} \alpha(\pi/2) & 2\pi \\ -2\pi & \alpha(\pi/2) \end{pmatrix},$$

so $L(\pi/2)$ cannot be a primary matrix function. Likewise, $L(\pi/6)$ cannot be a primary matrix function since $A(\pi/6)$ has both eigenvalues on the negative real axis. As t crosses $\pi/6$, the eigenvalues of $L(t)$ exit from the principal branch $(-\pi, \pi)$, and re-enter it at $t = 5\pi/6$. Naturally, passing from one branch to another implies passing from a primary to a non-primary matrix function by stepping on the boundary of the open connected set where the function $\log(z)$ is analytic. Moreover, remaining on the main branch would require introducing a discontinuity. For example, the principal logarithm associated with $A(t)$ is

$$
\begin{aligned}
&L_p(t) = L(t)\,, \quad 0 \le t < \pi/6\,,\\
&L_p(t) = L(t) + \begin{pmatrix} 0 & -2\pi \\ 2\pi & 0 \end{pmatrix}, \quad \pi/6 < t < 5\pi/6\,, \quad \text{etc.}
\end{aligned}
$$

L_p is clearly not defined at $t = \pi/6$, etc., whereas $L(t)$ in (1.2) is smooth for all t. Note, $\|A(t+\epsilon) - A(t)\| = O(\epsilon)$ when $t = \pi/6 - \epsilon/2$, but $\|L_p(t+\epsilon) - L_p(t)\| = O(1)$. Summarizing, care must be taken to obtain a smooth logarithm; to this end, we cannot consider just principal logarithms.

When working with matrix functions, familiar rules from calculus of one variable may not hold; e.g., for $e^{A+B} = e^A e^B$ we usually need A and B to commute. For logarithms, if $\log(A)\log(B) = \log(B)\log(A)$, then

$$
e^{\log(AB)} = e^{\log(A)+\log(B)}\,. \tag{1.3}
$$

Note, generally (1.3) does not hold if $AB = BA$, as the following shows:

$$
A = I, \;\; B = \begin{pmatrix} 1 & 3 \\ 0 & 2 \end{pmatrix}, \quad \log(A) = \begin{pmatrix} 0 & 2\pi \\ -2\pi & 0 \end{pmatrix}, \quad \log(B) = \begin{pmatrix} 0 & 3\log 2 \\ 0 & \log 2 \end{pmatrix}.
$$

However, (1.3) holds when $AB = BA$ if $\log(A)$ and $\log(B)$ are both primary matrix functions. Then, (1.3) implies that a logarithm of A may be found from the rescaled relation

$$
\log(A) = X + \log(e^{-X}A)\,, \quad \text{if} \quad AX = XA\,. \tag{1.4}
$$

This last result is indicative of the enormous simplifications for commuting matrices. If all matrices $A_0, A_1, \ldots, A_N$ commute, we can re-use much of the computational work (for example, only one Schur decomposition is needed). Also, rescaling as in (1.4) becomes obvious. Unfortunately, commutativity is rare.

Interpolation. The simple idea is: (i) given a suitable invertible function f, transform the data (t_j, A_j) to $(t_j, f(A_j))$, $j = 0, 1, \ldots, N$; (ii) perform standard polynomial interpolation on this transformed data resulting in the interpolant $P(t)$; (iii) finally, use $f^{-1}(P(t))$ as the interpolant for the original data.

Besides computational issues, the key is to determine f so that the interpolant has the desired structure. The following lemmas motivate our choices for f.

Lemma 1.2 *Given an invertible matrix $A \in \mathbb{R}^{n\times n}$ with no eigenvalue on the negative real axis:*

(i) if A is orthogonal, there exists a real skew symmetric logarithm;
(ii) if A is symplectic, there exists a real Hamiltonian logarithm;
(iii) if A is positive definite, there exists a unique real symmetric logarithm.

Moreover, in cases (i) and (ii), the logarithms are uniquely specified if, relative to the eigenvalues λ of A, we specify which the branch of the log *function (e.g., the usual choice would be $-\pi < \mathrm{Im}(\log(\lambda)) < \pi$). Conversely, given a matrix $X \in \mathbb{R}^{n\times n}$ which is skew symmetric, Hamiltonian, or symmetric, then the matrix e^X is orthogonal, symplectic, or positive definite, respectively.*

PROOF. The first part is in [11] for the orthogonal and symplectic cases, whereas the positive definite case is a trivial verification, using a Schur diagonalization of A. The converse is well-known from integration theory for differential equations. □

Remark 1.3 To obtain a real logarithm X, assuming A has no eigenvalues on the negative real axis can be weakened to require that "A has an even number of Jordan blocks of each size for every negative eigenvalue"; however, a real matrix X solution of $e^X = A$ cannot be a primary matrix function if A has any negative eigenvalues [5]. This weaker assumption also guarantees that if A is orthogonal then it has a real skew-symmetric logarithm which can easily be proved as follows: (i) bring A to block-diagonal form, $Q^T A Q = \mathrm{diag}(D_{11}, \ldots, D_{pp})$, where D_{ii} are either $(2,2)$ blocks of the type $\begin{pmatrix} c & s \\ -s & c \end{pmatrix}$, or are 1 or -1; (ii) for the blocks $\begin{pmatrix} c & s \\ -s & c \end{pmatrix}$ take as logarithms the skew-symmetric blocks $\begin{pmatrix} 0 & \beta \\ -\beta & 0 \end{pmatrix}$, with $\beta = \cos^{-1}(c)$; (iii) for the eigenvalue 1 take 0 as logarithm; (iv) for each block $\begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$ take a skew-symmetric logarithm $\begin{pmatrix} 0 & (2k+1)\pi \\ -(2k+1)\pi & 0 \end{pmatrix}$, $k \in \mathbb{Z}$. Sharp conditions under which a symplectic A has a real Hamiltonian logarithm are more involved [3].

Lemma 1.4 *Given an invertible matrix $A \in \mathbb{R}^{n\times n}$ such that $-1 \notin \Lambda(A)$, consider the Möbius transformation*

$$c : z \in \mathbb{C}/\{-1\} \to \gamma \frac{1-z}{1+z}, \quad \gamma \in \mathbb{R}, \gamma \neq 0 . \tag{1.5}$$

If A is orthogonal, symplectic, or positive definite, $c(A)$ is skew-symmetric, Hamiltonian, or symmetric, respectively. Conversely, given $X \in \mathbb{R}^{n\times n}$ such that $-\gamma \notin \Lambda(X)$ and X is skew symmetric or Hamiltonian then $c^{-1}(X)$ is orthogonal or symplectic, respectively. Moreover, if X is symmetric and

$$|\gamma| > \max_{\lambda_i \in \Lambda(X)} |\lambda_i| \tag{1.6}$$

then $c^{-1}(X)$ is positive definite.

PROOF. The orthogonal and symplectic cases are in [2]. If A is positive definite, then $c(A) = c(A)^T$ is obvious. Given X, clearly $c^{-1}(X) = (\gamma I - X)(\gamma I + X)^{-1}$, which is symmetric if X is. Let $Y = c^{-1}(X)$, $\mathbf{y} \in \mathbb{R}^n$ be a nonzero vector, and consider $\mathbf{y}^T Y \mathbf{y}$. Letting $\mathbf{z} = (\gamma I + X)^{-1}\mathbf{y}$,

$$\mathbf{y}^T Y \mathbf{y} = \mathbf{z}^T(\gamma I + X)(\gamma I - X)\mathbf{z} = \|\mathbf{z}\|^2\gamma^2 - \mathbf{z}^T X^2 \mathbf{z} \geq \|\mathbf{z}\|^2(\gamma^2 - \max \lambda_i(X^2)),$$

from which the result follows. □

Remark 1.5 In Lemma 1.4, for orthogonal and symplectic matrices, the value of γ in (1.5) is arbitrary; a typical choice is $\gamma = -1$, in which case $c(A)$ is the *Cayley transform* of A. If X in Lemma 1.4 is skew-symmetric, its eigenvalues are purely imaginary, and so certainly $\pm\gamma \notin \Lambda(X)$. Also, if X is Hamiltonian, and $-\gamma \notin \Lambda(X)$, then certainly also $\gamma \notin \Lambda(X)$ since eigenvalues come in $\pm$ pairs. For $\gamma = -2$ in (1.5), $c(A)$ is just the first diagonal Padé approximation to $\log(A)$, and the inverse transform is precisely the first diagonal Padé approximation of the exponential. Such nice interplay does not persist for higher order diagonal Padé approximations to $\log(A)$ and $\exp(X)$.

2 Logarithms of nearby matrices

A recurring problem in numerical computation is how to take advantage of work done already for slight changes in input. Typically, this problem is encountered for the factorization of a matrix when the change is a low rank modification. We consider the problem of obtaining $\log(A+E)$ once $\log(A)$ is known and E is a matrix of small norm.

We consider a Taylor expansion approach. Let G be the logarithm function and F the exponential function. That is, we have decided upon a stem function log acting on complex numbers z : $\log(z)$, and therefore consider only primary matrix functions $\log(A)$. Now, consider the stem function as defined locally. To

have $\log(A)$ real, we must take values of $\log(z)$ which are complex conjugate relative to complex conjugate eigenvalues of A. Formally,

$$G(A+E) = G(A) + G'(A)E + O(\|E\|^2)\,, \tag{2.1}$$

where $G'(A)$ is the Fréchet derivative of the function G. (We will see that this is well defined, and also justify the term $O(\|E\|^2)$ in (2.1).) The following iterative process is immediate. With $A_0 = A$ and $L_0 = G(A_0)$ given,

$$\begin{aligned} L_{k+1} &= L_k + G'(A_k)E_k\,, \qquad A_k = F(L_k), \\ E_k &= A + E - F(L_k)\,, \qquad k = 0, 1, \ldots\,. \end{aligned} \tag{2.2}$$

We need a concrete representation for $G'(A_k)E_k$. Restricting attention to principal logarithms, an expression for $G'(A)E$ [4] is

$$G'(A)E = \int_0^1 [(A-I)t + I]^{-1} E [(A-I)t + I]^{-1} dt\,. \tag{2.3}$$

Remark 2.1 Under the stated restrictions, (2.2)–(2.3) is a locally quadratically convergent iteration. However, an iterative method based on (2.2)–(2.3) is not convenient; it is expensive, works only for principal logarithms, and is not competitive with the formulations below.

Consider the relation $e^{\log(A)} = A$, or $F(G(A)) = A$. Using the integral formula for the Fréchet derivative of the exponential [12], and letting $X = G(A)$,

$$F'(X):\ Z \to F'(X)Z \ = \ \int_0^1 F((1-t)X) Z F(tX) dt\,. \tag{2.4}$$

Therefore, if the mapping $Z \to F'(G(A))Z$ is invertible, using the chain rule for the composite function,

$$G'(A)\ E \ = \ [F'(G(A))]^{-1}\ E\,. \tag{2.5}$$

This is a general expression for the Fréchet derivative of G when $F'(G(A))$ is invertible. Now unroll (2.4): $Z \to \mathrm{vec}(Z)$, where $\mathrm{vec}(Z) \in I\!R^{n^2}$ is the vector of entries the columns of Z (first to last). Thus, an equivalent requirement is that the $n^2 \times n^2$ matrix

$$D(X) := \int_0^1 e^{tX^T} \otimes e^{(1-t)X} dt$$

be invertible. Using elementary properties of the Kronecker product [5],

$$\nu \in \Lambda(D(X)) \Rightarrow \nu = \int_0^1 e^{\lambda t} e^{\mu(1-t)} dt\,, \quad \lambda,\, \mu \in \Lambda(X)\,;$$

therefore,

$$\nu = e^{\lambda}\,,\ \lambda = \mu\,; \qquad \nu = \frac{e^{\lambda} - e^{\mu}}{\lambda - \mu}\,,\ \ \lambda \neq \mu\,.$$

Now, recall that $\lambda,\, \mu \in \Lambda(\log(A))$, and $\nu = 0$ (hence $D(X)$ is not invertible) if and only if $\lambda = \mu \pm 2ik\pi$, $k \in \mathbb{N}$. This is not possible for primary matrix functions $\log(A)$. Thus, we have the following lemma.

Lemma 2.2 *If* $\log(A)$ *is a primary matrix function, then its Fréchet derivative is given by* (2.5).

Remark 2.3 The expression (2.5) was derived in [8] for the principal logarithm. It suffices that the logarithms be primary matrix functions.

An alternative to the Taylor expansion of $G(A+E)$ is provided by Newton's method for the nonlinear equation

$$F(L) - A = 0\,. \tag{2.6}$$

Given L_0, iterate

$$F'(L_k)\,(L_{k+1} - L_k) \;=\; A - A_k, \quad A_k = F(L_k)\,,\ \ k = 0, 1, \ldots, \tag{2.7}$$

where $F'(L_k)(L_{k+1} - L_k)$ is given by (2.4). Invertibility of $F'(L_k)$ requires that no two eigenvalues λ and μ of L_k may be of the type $\lambda = \mu \pm 2ik\pi$. Recall,

$$F(X+Z) \;=\; F(X) + \int_0^1 F((1-t)X) Z F(t(X+Z)) dt\,. \tag{2.8}$$

Now, from (2.7), with the restriction on the eigenvalues,

$$F'(L_k)[(L_{k+1} - L) + (L - L_k)] = F(L) - F(L_k),$$

so

$$L_{k+1} - L = (F'(L_k))^{-1} \int_0^1 F(L_k(1-t))\ (L - L_k)\ (F(Lt) - F(L_k t)) dt,$$

and the last term under the integral sign is $O(L - L_k)$, so we have local quadratic convergence.

Remark 2.4 With the restriction on the eigenvalues of L_k, iteration (2.7) (using (2.4)) is the same as (2.2)–(2.5). Generally it is not possible to use (2.7) to approximate logarithms L which are not primary matrix functions of A.

The only known technique for approximating, by Newton's method, logarithms which are not primary matrix functions uses (2.7) when L_0 commutes with L [2]. In this case (2.7) can be rewritten

$$\begin{aligned} L_{k+1} &= L_k + e^{-L_k}(A - A_k), \quad \text{or} \\ L_{k+1} &= L_k + e^{-L_k}A - I, \quad k = 0, 1, \ldots \end{aligned} \tag{2.9}$$

Summary. There are three possible formulations of Newton's method: (2.2), (2.7), and (2.9). For the derivative in (2.2), we can use (2.3) if we are on the principal branch, or (2.5) for any other primary logarithm. For the derivative in (2.7), we can use (2.4). For primary logarithms, (2.2) and (2.7) are equivalent. Finally, (2.9) is valid only if $L_0 L = L L_0$; then (2.9) can be used to approximate any logarithm (primary or non-primary).

Known conditioning results are restricted to the principal logarithm [4,8]. In [4,8], a representation of the Fréchet derivative of the logarithm is justified for principal logarithms only. On the other hand, from Lemma 2.2, the Fréchet derivative is well defined for any primary logarithm. Moreover, the argument following (2.8) legitimizes the term $O(\|E\|^2)$ in (2.1). Thus, we can use (2.1) to express the sensitivity of the log function. In particular, if $L \neq 0$ and $L + \Delta L$ are primary logarithms of A and $A + E$, then

$$\frac{\|\Delta L\|}{\|L\|} \leq \|G'(A)\| \, \frac{\|A\|}{\|L\|} \, \frac{\|E\|}{\|A\|} \, + O(\|E\|^2)\,. \tag{2.10}$$

The *condition number* of the logarithm function G at A,

$$\|G'(A)\| \, \frac{\|A\|}{\|L\|}, \tag{2.11}$$

acts as a relative error magnification factor.

Remark 2.5 From (2.11), using (2.5), the closer $\log(A)$ is to a non-primary matrix function, the worse its conditioning.

Next, we turn our attention to principal logarithms.

Theorem 2.6 *Given $A \in \mathbb{R}^{n\times n}$ with principal logarithm $\log(A)$. Let $E \in \mathbb{R}^{n\times n}$ be such that $A+E$ has a principal logarithm, $\log(A+E)$. Then*

$$\log(A+E) = \log(A) + \int_0^1 ((A-I)t+I)^{-1} E\,[(A+E-I)t+I]^{-1}\,dt\,. \quad (2.12)$$

PROOF. From (1.1),

$$\log(A+E) \;=\; \int_0^1 (A+E-I)((A+E-I)t+I)^{-1}dt\,. \quad (2.13)$$

Now,

$$\begin{aligned}(A+E-I)[(A+E-I)t+I]^{-1} &= (A-I)[(A-I)t+I]^{-1} \\ &+ [(A-I)t+I]^{-1}E[(A+E-I)t+I]^{-1}\,. \end{aligned} \quad (2.14)$$

Using (1.1) for $\log(A)$ and (2.13) for $\log(A+E)$, (2.12) follows. □

Remark 2.7 (2.12) is valid since the principal logarithm $\log((A+E-I)t+I)$ exists for all $t \in [0,1]$, not just at $t = 0, 1$. Our path $(A+E-I)t+I$ from I to $A+E$, which exploits knowledge of $\log(A)$, differs greatly from the more direct homotopy path $A+tE$. Exploiting the latter path is profitable when $(A+tE)$ is invertible for $0 \le t \le 1$ and the principal logarithm of $(A+E)A^{-1}$ exists, to compute the logarithm of the rescaled matrix $(A+E)A^{-1}$:

$$\log((A+E)A^{-1}) \;=\; E\,\int_0^1 (A+tE)^{-1}dt\,.$$

(This expression coincides with (2.12) when A and E commute (recall (1.4)).)

The integral in (2.12) is the harder to evaluate the closer we are to the boundary of the main branch for the eigenvalues of the logarithm(s) of A and $A+E$, and these are ill-conditioned principal logarithms (see Remark 2.5). Consider replacing

$$\int_0^1 ((A-I)t+I)^{-1}\;E\;((A+E-I)t+I)^{-1}\;dt := \int_0^1 M(t)dt\,,$$

by a quadrature rule

$$Q := \sum_{k=1}^{N} g_k M(t_k)\,.$$

Let

$$V(t) := (A-I)[(A-I)t+I]^{-1},$$
$$W(t) := (A+E-I)[(A+E-I)t+I]^{-1},$$

so

$$M(t) = W(t) - V(t) = (A-I)^{-1}V(t)EW(t)(A+E-I)^{-1}.$$

For standard quadrature rules based on polynomial interpolation, an estimate of the error in replacing $\int_0^1 M(t)dt$ by Q is of the form $ng(N)\max_{0\le t\le 1}\|M^{(k)}(t)\|$ for an appropriate value k. We can obtain an error estimate using the rule on the entries of M; the factor n arises because of norm arguments, while $g(N)$ is a term which depends on the number of points, and decreases with increasing N. A lengthy but simple computation shows that

$$V^{(j)}(t) = (-1)^j j!\, V^{j+1}(t), \qquad W^{(j)}(t) = (-1)^j j!\, W^{j+1}(t)\,,$$

and thus

$$M^{(k)}(t) = (-1)^k k!\,(A-I)^{-1}V(t)\left[\sum_{j=0}^{k} V^j(t)EW^{k-j}(t)\right]W(t)(A+E-I)^{-1}. \tag{2.15}$$

So, we need good bounds for $\|[(A-I)t+I]^{-1}\|$ and $\|[(A+E-I)t+I]^{-1}\|$ to obtain reasonable error bounds. For example, for Gauss rules, the error estimate (see also Corollary 4.4 in [4])

$$\|\int_0^1 M(t)dt - Q]\| \le n\frac{(N!)^4}{(2N+1)((2N)!)^3}\max_{0\le t\le 1}\|M^{(2N)}(t)\|.$$

holds. If $\|A-I\| \le \omega < 1$ and $\|A+E-I\| \le \omega$, we can sharpen this to

$$\|\int_0^1 M(t)dt - Q\| \le n\frac{(N!)^4}{(2N)!\,(2N)!}\frac{1}{(1-\omega)^2}\left(\frac{\omega}{1-\omega}\right)^{2N}\|E\|\,.$$

Theorem 2.8 *Consider* (2.12) *and suppose that Gauss quadrature rules are used to approximate the integral. Assume that A and $A+E$ are both orthogonal or symplectic, respectively. Also, assume that* $\log(A)$ *is either exact, or replaced by a skew-symmetric or Hamiltonian approximation, respectively. Then, the resulting approximation to* $\log(A+E)$*, if defined, is skew-symmetric or Hamiltonian, respectively.*

PROOF. Recall the integral formula (2.13) for $\log(A+E)$. From [4, Theorem 4.3], Gauss rules at N points on (2.13) are equivalent to (N,N) diagonal Padé approximants to $\log(A+E)$. Moreover, from [2, Theorem 2.2], diagonal Padé approximants are skew-symmetric or Hamiltonian matrices if $A+E$ is orthogonal or symplectic, respectively. Finally, because of (2.14) and the linearity of Gauss rules, using a Gauss rule on (2.13) is the same as using a Gauss rule on

$$\int_0^1 (A-I)[(A-I)t+I]^{-1}dt, \quad \int_0^1 [(A-I)t+I]^{-1}E[(A+E-I)t+I]^{-1}dt$$

separately. The first integral represents $\log(A)$, thus Gauss rules on the second integral deliver the stated structure. □

We briefly discuss positive definite matrices. Two facts set these matrices apart: (i) we are interested only in the unique real symmetric (principal) logarithm; (ii) the series expansion ("series 2" of [4])

$$\log(A) \;=\; 2\sum_{k=0}^{\infty} \frac{1}{2k+1}[(A-I)(A+I)^{-1}]^{2k+1} \tag{2.16}$$

converges. In contrast, the more familiar series ("series 1" of [4])

$$\log(A) \;=\; -\sum_{k=1}^{\infty} \frac{(I-A)^k}{k}\,, \tag{2.17}$$

requires $\rho(I-A)<1$ for convergence. To rescale a positive definite matrix A, we use (1.5) restricted to diagonal scalings:

$$\log(A) \;=\; -I\log d + \log(dA)\,, \quad d>0\,. \tag{2.18}$$

Lemma 2.9 *Let A be positive definite, and $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n > 0$ be its eigenvalues. Then, the value $d>0$ for which $\|I-dA\|<1$ is minimized is*

$$d \;=\; \frac{2}{\lambda_1+\lambda_n}\,. \tag{2.19}$$

The value $d>0$ for which $\|(I-dA)(I+dA)^{-1}\|$ is minimized is

$$d \;=\; \frac{1}{\sqrt{\lambda_1\lambda_n}}\,. \tag{2.20}$$

PROOF. A simple verification suffices. □

Remark 2.10 Note, scaling with d in (2.19) gives $\|I-dA\| = (\mathrm{cond}(A) -$

$1)/(\mathrm{cond}(A)+1)$, and with d in (2.20) gives $\|(I-dA)(I+dA)^{-1}\| = (\mathrm{cond}(A^{1/2})-1)/(\mathrm{cond}(A^{1/2})+1)$. Thus, unless $\mathrm{cond}(A)$ is reasonably small, or we are content with low accuracy, the series (2.16) and (2.17) (even after rescaling) are not competitive with a Schur decomposition approach computationally. To obtain inexpensive estimates of the optimal d, we estimate $\lambda_1,\ \lambda_n$ from Gerschgorin segments on $A,\ D^{-1}AD,\ D^{-1/2}AD^{1/2}$, with $D = \mathrm{diag}(A)$, [6], by calculating these segments both by row and column; estimates of 0 for λ_n are replaced by the machine precision `EPS`. This is sufficient for rescaling.

3 Matrix interpolation

We consider interpolating a sequence of matrices of orthogonal, symplectic, or positive definite type, so that the result is orthogonal, symplectic, or positive definite, respectively. The basic results are Lemmas 1.2 and 1.4. Consider the choice in Lemma 1.2.

Logarithm and exponential. We have the data set $(t_i,\, A_i)_{i=0}^N$, with $t_i \neq t_j$, $i \neq j$, where $A_i \in I\!R^{n\times n}$, $A_i = A(t_i)$, with A a smooth function. Assume that $A(t) = e^{L(t)} \in I\!R^{n\times n}$ for some smooth $L(t)$. Suppose also that $L_i = \log(A_i) = L(t_i)$, $i = 0, \ldots, N$.

Theorem 3.1 *Let $\tilde{L}(t)$ be the unique interpolatory polynomial to $(t_i,\, L_i)_{i=0}^N$, and let*

$$\tilde{A}(t) \;=\; e^{\tilde{L}(t)}\,. \tag{3.1}$$

If $L \in \mathcal{C}^{N+1}$,

$$\begin{aligned}
&\|A(t) - \tilde{A}(t)\| \leq \\
&\quad \frac{|\prod_{i=0}^N (t-t_i)|}{(N+1)!}\, \max_\eta \|L^{(N+1)}(\eta)\|\, \max_{0\leq s\leq 1} (\|e^{\tilde{L}(t)(1-s)}\|\, \|e^{L(t)s}\|)\,, \\
&\|I - \tilde{A}(t)A^{-1}(t)\| \leq \\
&\quad \frac{|\prod_{i=0}^N (t-t_i)|}{(N+1)!}\, \max_\eta \|L^{(N+1)}(\eta)\|\, \max_{0\leq s\leq 1} (\|e^{\tilde{L}(t)(1-s)}\|\, \|e^{L(t)(s-1)}\|)\,,
\end{aligned} \tag{3.2}$$

where $\eta = \eta(t)$ is a multi-index, that is, $L^{(N+1)}(\eta)$ is the matrix of entries $L_{ij}^{(N+1)}(\eta_{ij})$ with $t_0 \leq \eta_{ij} \leq t_N$. Moreover, for all t, $\tilde{A}(t)$ is orthogonal, symplectic, or positive definite, if $L(t)$ is skew-symmetric, Hamiltonian, or symmetric (so $A(t)$ is orthogonal, symplectic, or positive definite) respectively.

PROOF. Writing

$$\tilde{L}(t) = \sum_{i=0}^{N} l_i(t) L_i \,, \quad l_i(t) = \prod_{k=0,\, k\neq i}^{N} \frac{(t-t_k)}{(t_i - t_k)} \,,$$

since skew-symmetric, Hamiltonian, and symmetric matrices are closed under sum, so is $\tilde{L}(t)$, and the last part follows at once. The error estimates follow from taking norms in the relation

$$e^{L(t)} - e^{\tilde{L}(t)} = \int_0^1 e^{\tilde{L}(t)(1-s)} (L(t) - \tilde{L}(t)) e^{L(t)s} ds \,,$$

and from the interpolation error formula

$$\|L(t) - \tilde{L}(t)\| = \frac{1}{(N+1)!} |\prod_{i=0}^{N} (t - t_i)| \, \|L^{(N+1)}(\eta)\| \,. \qquad \square$$

Remark 3.2 The first estimate in (3.2) gives the absolute error, while the second is essentially the relative error.

Remark 3.3 If $A(t)$ is orthogonal, and hence $L(t)$ and $\tilde{L}(t)$ are skew-symmetric and $\tilde{A}(t)$ is orthogonal, then $\|e^{L(t)s}\| = \|e^{\tilde{L}(t)s}\| = 1$, and the only error is from interpolating the L_i.

The strongest assumptions we have made are that $A(t) = e^{L(t)}$ for some real $L \in \mathcal{C}^{N+1}$, and that we can obtain $L(t_i)$ from the $A(t_i)$. The second assumption has nontrivial practical implications. The assumption $A(t) = e^{L(t)}$ with smooth L may not be restrictive, for example, when L can be taken as a principal logarithm, as for positive definite A. This follows at once from the integral representation (1.1), and suggests that the overall interpolation problem is simpler if $A(t)$ is not far from the identity. To ensure that we need to deal only with principal logarithms, a variety of transformations are available.

Possible rescalings

Example 3.4 Suppose $A(t)$ has been measured only at (t_1, A_1) and (t_2, A_2), that $A \in \mathcal{C}^2$ (at least), and that $\|I - A_{1,2}\| > 1$ but A_1 and A_2 are close. Let $h := t_2 - t_1$, and $E := A_1 - A_2$. The previous construction approximates $A(t)$ on $[t_1, t_2]$ by $\tilde{A}(t) = e^{\tilde{L}(t)}$, where

$$\tilde{L}(t) = \frac{t_2 - t}{h} \log(A_1) + \frac{t - t_1}{h} \log(A_2).$$

Now, suppose E is small enough so that $\|E A_1^{-1}\| < 1$. Consider

$$\hat{A}(t) \;=\; e^{\hat{L}(t)} \; A_1 \,, \quad \hat{L}(t) = T_1 \, \frac{t - t_1}{h} \,, \quad T_1 = \log(A_2 A_1^{-1}) \,. \qquad (3.3)$$

Then, $\hat{A}(t)$ is a second order approximation to $A(t)$ which is generally different from $\tilde{A}(t)$. The advantage of (3.3) is that$A_2A_1^{-1}$ is now close to the identity, since $A_2 = (I - EA_1^{-1})A_1$, from which it follows that $\|I - A_2A_1^{-1}\| = \|EA_1^{-1}\| < 1$. Thus, it should be simpler to find T_1 (as principal logarithm) than to find L_1 and L_2; moreover, there is one less logarithm to compute. Symplecticity and orthogonality are clearly preserved by (3.3). If $A(t)$ is positive definite, to be sure that we only take logarithms of positive definite matrices, we may rescale as follows. Suppose E is such that $\max|\lambda(E(A_1+A_2)^{-1})| < \max(\|(I - A_1)(I + A_1)^{-1}\|, \|(I - A_2)(I + A_2)^{-1}\|)$, and let $A_1 = CC^T$ be the Cholesky factorization. Consider

$$\hat{A}(t) \;=\; C\; e^{\hat{L}(t)}\; C^T\,, \quad \hat{L}(t) = T_1\;\frac{t - t_1}{h}\,, \quad T_1 = \log(C^{-1}A_2C^{-T})\,. \tag{3.4}$$

This is also a second order approximation to $A(t)$, but now the matrix $C^{-1}A_2C^{-T}$ is better scaled (recall (2.16)). In fact, $\|(I - C^{-1}A_2C^{-T})(I + C^{-1}A_2C^{-T})^{-1}\| = \max|\lambda(E(A_1 + A_2)^{-1})|$.

Here, we outline an extension of (3.3) which gives better computational results, and on which we will report in Section 5. For this rescaling strategy to be successful, we need the matrices to be inverted to be well conditioned, as otherwise a loss of precision may take place. We rescale the data set $(t_i, A_i)_{i=0}^N$ to $(t_i, A_iB)_{i=0}^N$, where B is chosen so that the A_iB are closer to the identity than are the A_i. Then, the interpolant (3.1) is used on this rescaled data set. In summary,

$$\bar{A}(t) \;=\; e^{\bar{L}(t)}B^{-1}\,, \quad \bar{L}(t) = \sum_{i=0}^{N} l_i(t)\bar{L}_i\,, \quad \bar{L}_i = \log(A_iB)\,. \tag{3.5}$$

Error estimates such as those in (3.2) are naturally modified for (3.5). Clearly, if the matrices A_i's are symplectic and orthogonal, so is $\bar{A}(t)$ in (3.5). In the numerical experiments of Section 5, we use (3.5) for symplectic and orthogonal matrices by taking $B^{-1} = A_{\lfloor N/2\rfloor}$.

Remark 3.5 For positive definite matrices, we may replace (3.5) by

$$\bar{A}(t) \;=\; Ce^{\bar{L}(t)}C^T\,, \quad \bar{L}(t) = \sum_{i=0}^{N} l_i(t)\bar{L}_i\,, \quad \bar{L}_i = \log(C^{-1}A_iC^{-T})\,, \tag{3.6}$$

where in practice we use $CC^T = A_{\lfloor N/2\rfloor}$. Note, (3.6) and (3.5) are theoretically identical, if we use $B = C^{-T}C^{-1}$, since then $A_iB = C(C^{-1}A_iC^{-T})C^{-1}$. However, with (3.6), we compute logarithms only of positive definite matrices.

Cayley transform. A more efficient alternative to the logarithm and expo-

nential is to represent the data (t_i, A_i) using the rational function of Lemma 1.4, fixing $\gamma = -1$ in (1.5). Assuming that $-1 \notin \Lambda(A_i)$, $i = 0, \ldots, N$, and that $1 \notin \Lambda(\tilde{C}(t))$,

$$\begin{aligned} A_i \to C_i &= -(I - A_i)(I + A_i)^{-1}\,, \\ \tilde{C}(t) = \sum_{i=0}^{N} C_i l_i(t)\,, &\quad \text{and} \quad \tilde{A}(t) \;=\; (I + \tilde{C}(t))\;(I - \tilde{C}(t))^{-1}\,. \end{aligned} \tag{3.7}$$

Theorem 3.6 *With the above notation, let $A(t) \in \mathbb{R}^{n\times n}$ be invertible, assume that $-1 \notin \Lambda(A(t))$, $t \in [t_0, t_N]$, and that $1 \notin \Lambda(\tilde{C}(t))$. Let $C(t) = -(I - A(t))(I + A(t))^{-1}$, and assume that $A \in \mathcal{C}^{N+1}$. Then,*

$$\begin{aligned} &\|A(t) - \tilde{A}(t)\| \le \\ &\quad 2\,\frac{|\prod_{i=0}^{N}(t - t_i)|}{(N+1)!}\,\max_{\eta}\|C^{(N+1)}(\eta)\|\,\|(I - C(t))^{-1}\|\,\|(I - \tilde{C}(t))^{-1}\|\,, \\ &\|I - A^{-1}(t)\tilde{A}(t)\| \le \\ &\quad 2\,\frac{|\prod_{i=0}^{N}(t - t_i)|}{(N+1)!}\,\max_{\eta}\|C^{(N+1)}(\eta)\|\,\|(I + C(t))^{-1}\|\,\|(I - \tilde{C}(t))^{-1}\|\,, \end{aligned} \tag{3.8}$$

where $\eta = \eta(t)$ is a multi-index and $C^{(N+1)}(\eta)$ is the matrix of entries $C_{ij}^{(N+1)}(\eta_{ij})$ with $t_0 \le \eta_{ij} \le t_N$. Moreover, for all t, $\tilde{A}(t)$ is orthogonal or symplectic, if $A(t)$ is orthogonal or symplectic, respectively. If $A(t)$ is positive definite, then $\tilde{A}(t)$ is also positive definite, if $\max_{\lambda\in\Lambda(\tilde{C}(t))}|\lambda| < 1$.

PROOF. Since $C(t) = -(I - A(t))(I + A(t))^{-1}$ and $A \in \mathcal{C}^{N+1}$, then $C \in \mathcal{C}^{N+1}$. Moreover,

$$A(t) - \tilde{A}(t) \;=\; 2(I - C(t))^{-1}(C(t) - \tilde{C}(t))(I - \tilde{C}(t))^{-1}.$$

Taking norms and using the polynomial interpolation error formula, (3.8) follows. The last part is a consequence of Lemma 1.4. □

Remark 3.7 We restrict to the case $\gamma = -1$ for simplicity, since every value of $\gamma \neq 0$ gives the same $\tilde{A}(t)$.

Remark 3.8 If $A(t)$ is orthogonal, and hence C, $\tilde{C}$ are skew-symmetric and $\tilde{A}$ is orthogonal, it is easy to obtain the bounds

$$\begin{aligned} &2\|(I - C(t))^{-1}\|\,\|(I - \tilde{C}(t))^{-1}\| \le 2\,, \\ &2\|(I + C(t))^{-1}\|\,\|(I - \tilde{C}(t))^{-1}\| \le 2\,. \end{aligned} \tag{3.9}$$

Thus, at worst, the interpolation error is magnified by the factor 2. (Compare with Remark 3.2 [2]).

Of course, the rescaling strategy for the logarithm and exponential pair (see (3.5)–(3.6)) may be used to make the C_i's closer to the zero matrix.

There is a major difference between Theorems 3.1 and 3.5. In both cases, we need assumptions on the data (i.e., on $A(t)$), however in Theorem 3.5 we also need a restriction on the spectrum of the interpolant: $1 \notin \Lambda(\tilde{C}(t))$; using standard polynomial interpolation on the C_i, there is no guarantee of this. It is easy to construct positive definite matrices A_i violating this condition even when all C_i's have eigenvalues in $(0, 1)$.

4 Implementation issues

Here, we discuss implementations and a (leading order) flops estimate. [1]

Logarithms of nearby matrices. For principal logarithms, we have experimented with (2.12) restricted to Gauss-Legendre rules, both for accuracy and to recover skew-symmetric/Hamiltonian logarithms (see Theorem 2.8). The cost is that associated with evaluating the integrand, $8n^3/3$. Unless $\|A - I\|$ and $\|A + E - I\|$ are small (say, $< 1/2$), the large number of quadrature points needed for a good accuracy makes the procedure very expensive.

For Newton's method in the formulation (2.7), each iteration requires computing the exponential of a matrix and the solution of $F'(L_k)\delta_k = E_k$, where $\delta_k = L_{k+1} - L_k$ and $E_k = A - A_k$. For principal logarithms, we may use (2.3) and approximate δ_k by a quadrature rule, but this is usually very expensive. Instead, we may use (2.4) to define δ_k implicitly. Then, Newton's method can be used to compute non-principal primary logarithms. Thus, we need to find δ_k from

$$F'(L_k)\delta_k := \int_0^1 e^{(1-t)L_k}\delta_k e^{tL_k}dt = E_k\,. \tag{4.1}$$

Consider the operator

$$\delta \;\to\; F'(L)\delta \;:=\; \int_0^1 F((1-t)L)\delta F(tL)dt$$

[1] Here, a flop is the combined expense of one floating point multiplication and one floating point addition.

and approximate this operator using the composite trapezoidal rule,

$$F'_J(L)\Delta_J := \frac{1}{2^{J+1}}[e^L\Delta_J + 2\sum_{j=1}^{2^J-1} e^{jL/2^J}\Delta_J e^{(2^J-j)L/2^J} + \Delta_J e^L]\,. \qquad (4.2)$$

The following equivalent formulation is much easier to evaluate (only requiring the matrices $e^{L/2^{J-j}}$ which are already available if e^L is computed by a standard scaling and squaring method [8]). Let

$$\begin{aligned} Z_J &:= (e^{L/2^J}\Delta_J + \Delta_J e^{L/2^J})/2^{J+1}\,, \\ Z_{j-1} &:= e^{L/2^j}Z_j + Z_j e^{L/2^j}\,, \quad j = J,\, J-1,\ldots,\,1\,, \end{aligned}$$

then

$$F'_J(L)\Delta_J = Z_0\,.$$

In [8], Kenney and Laub proved that the operator $F'_J(L)$ is invertible for $J > 0$ and for the principal logarithm L. In fact, it is invertible for any primary logarithm. It is enough to note that $[F'_J(L)]^{-1}$ can be found by inverting the above procedure; i.e., if $Z_0 := E = F'_J(L)\Delta_J$, solving sequentially for $Z_1,\, Z_2, \ldots, Z_J$ and Δ_J from

$$\begin{aligned} e^{L/2^j}Z_j + Z_j e^{L/2^j} &= Z_{j-1}\,, \\ e^{L/2^J}\Delta_J + \Delta_J e^{L/2^J} &= 2^{J+1}Z_J\,, \end{aligned}$$

then $F'_J(L)$ is invertible when these (Sylvester) equations are uniquely solvable. This necessitates that $u + v \neq 0$ for $u,\, v \in \Lambda(e^{L/2^j})$, $j \geq 1$, which is true for any primary logarithm L, since $\lambda \neq \mu \pm 2ki\pi$ when $\lambda,\, \mu \in \Lambda(L)$. In short, we have shown the following lemma.

Lemma 4.1 *If $L = \log(A)$ is a primary matrix function, then its Fréchet derivative can be approximated by $[F'_J(L)]^{-1}$.*

Thus, the solution δ_k of $F'(L_k)\delta_k = E_k$ can be approximated by the solution $\Delta_{k,J}$ of $F'_J(L_k)\Delta_{k,J} = E_k$. To implement this, we use the standard procedure of scaling and squaring followed by using a $(6,6)$ diagonal Padé approximant to evaluate $e^{L_k/2^J}$ (in essence, the function `expm` in `Matlab` [9]); J was chosen so that $\|L_k/2^J\| < 1/4$, which gave sufficient accuracy for $\Delta_{k,J}$, and represented a good compromise between quadrature error, speed of convergence, and cost. To solve the Sylvester equations $BX + XB = C$, we used the function `schur` of `Matlab` to bring B to quasi-triangular form, then solved the block-triangular linear system $[I \otimes B + B^T \otimes I]\mathrm{vec}(X) = \mathrm{vec}(C)$. The Schur reduction is of L_k, so that for each iteration all computations are performed on quasi-triangular matrices, and requires about $8n^3$ flops [8]. At the last stage, we must work

with full matrices (E_k and Δ_k) which requires $4n^3$ extra flops. To compute the cost of one iteration, we must add the cost of computing $A_k = e^{L_k}$, i.e., $(16/3 + J)n^3/6$ flops, and the cost of solving $J + 1$ Sylvester equations, i.e., $(J + 1)n^3$ flops. This is expensive, but permits recovery of a primary, non-principal, logarithm close to a given one.

Now we consider the convergence properties of the process arising when we replace the integral (4.1) by a quadrature rule as in (4.2). The approximate Newton iteration is

$$\begin{aligned} F_J'(Y_k)\Delta_{k,J} &= A - B_k, \quad B_k = e^{Y_k}, \\ Y_{k+1} &= Y_k + \Delta_{k,J}, \quad k = 0, 1, \ldots, \end{aligned} \tag{4.3}$$

with $Y_0 = L_0$ given, where $F_J'(Y_k)\Delta_{k,J}$ is given as in (4.2).

Theorem 4.2 *For the iteration* (4.3),

$$\begin{aligned} \Delta_{k+1,J} = -F_J'(Y_{k+1})^{-1}\Big[&\int_0^1 e^{(1-t)Y_k}\Delta_{k,J}(e^{tY_{k+1}} - e^{tY_k})dt \\ &+ (F'(Y_k) - F_J'(Y_k))\Delta_{k,J}\Big]. \end{aligned} \tag{4.4}$$

PROOF. A simple verification suffices. □

Remark 4.3 The expression (4.4) highlights that the decrease in magnitude of the updates in (4.3) is affected by two factors. First is the usual factor of a Newton iteration which is quadratic in the previous update (since the last term under the integral sign in (4.4) is $O(\Delta_{k,J})$); the second factor in (4.4) arises from the inexact evaluation of the integral in (4.1), and depends linearly on the previous update. If the composite trapezoidal rule (4.2) is used to obtain F_J', then the contraction factor for the linearly decreasing term is proportional to h^2, where $h = 1/2^J$. Thus, there is a clear trade-off between speed of convergence and cost. It is inefficient to use an extremely accurate approximation to the integral in (4.1), and the strategy we adopted of selecting $J, \|L_k\|/2^J < 1/4$, gives an excellent compromise between speed of convergence and overall cost.

Interpolation. The basic technique is $\tilde{A}(t) = f^{-1}(P(t))$, where

$$P(t) = \sum_{i=0}^{N} l_i(t) f(A_i)$$

and $l_i(t)$ are the cardinal functions of Lagrange interpolation. The cost is

$N+1$ evaluations of f, plus evaluating f^{-1} for each value $\tilde{A}(t)$ that we must compute.

The *log-exp method* arises from choosing $f(A_i) = \log(A_i)$ with f^{-1} the exponential function. The cost of evaluating $\log(A_i)$ is $O(n^3)$, and the size of the constant in the order term usually depends on $\|I - A\|$. There is potential for ill-conditioning when A_i has eigenvalues near the negative real axis. In general, to obtain a smooth interpolant, non-principal logarithms must be computed, and possibly also non-primary ones. The *Cayley method* results from taking $f(A_i)$ to be the Cayley transform of A_i and f^{-1} its inverse transform. Computing f and its inverse requires $4n^3/3$ flops. Clearly, this is less expensive than the log-exp method, but it can be unstable because of the inversion of $I - \tilde{C}(t)$ in (3.7). Moreover, it is not defined if $-1 \in \Lambda(A_i)$.

To alleviate these difficulties, for both methods we experimented with the rescaling in (3.5) and (3.6). The rescaling cost is $(\frac{1}{3} + N)n^3$ flops, and, with respect to the un-rescaled method, we need one less f-evaluation and one more matrix multiplication to compute $\bar{A}(t)$.

5 Examples

All our computations used `Matlab` (`EPS` $\simeq 2.2 * 10^{-16}$). We have solved many problems, some of them arising from systems of differential equations. Here, we report only on those tests which highlight the typical behavior for computing logarithms of nearby matrices, computing non-principal logarithms, and structured interpolation of sequences of matrices.

Example 5.1 This is Example 1.1 with $\alpha(t) = 0$. Recall that $A(t)$ is orthogonal and symplectic, $L(t)$ is skew-symmetric and Hamiltonian, and $-1 \in \Lambda(A(t))$, for $t = \pm\pi/6 + 2k\pi$; so, in the neighborhood of $\pm\pi/6 + 2k\pi$, we cannot use the Cayley method for interpolation purposes, and we need non-principal (and possibly non-primary) logarithms for the log-exp method.

Example 5.2 $A(t) = Q_1(t)Q_2(t)$ where

$$Q_1(t) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos a_1 t & \sin a_1 t & 0 \\ 0 & -\sin a_1 t & \cos a_1 t & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$Q_2(t) = \begin{pmatrix} \cos a_2 t & \sin a_2 t & 0 & 0 \\ -\sin a_2 t & \cos a_2 t & 0 & 0 \\ 0 & 0 & \cos a_2 t & \sin a_2 t \\ 0 & 0 & -\sin a_2 t & \cos a_2 t \end{pmatrix} , \ t \in [0,1].$$

$A(t)$ is orthogonal, but we do not know $\log(A(t))$; the matrices $A(t)$ and $A(s)$ do not commute ($t \neq s$).

(a) $a_1 = 1,\ a_2 = 4$. With $A(0) = I$, $\log(A(0)) = 0$, $\log(A(t))$ is inside the principal branch, but is very close to the boundary for $t \in (0.75, 0.85)$.
(b) $a_1 = -4,\ a_2 = 6$. With $A(0) = I$, $\log(A(0)) = 0$, $\log(A(t))$ exits from the principal branch for some $t \in (0.7, 0.8)$. We do not know if $A(t) = e^{L(t)}$ for a smooth $L(t)$.

Example 5.3 $A(t) = S_1(t)S_2(t)$ where:

$$S_1(t) = \begin{pmatrix} I & Z(t) \\ 0 & I \end{pmatrix} , \quad S_2(t) = \begin{pmatrix} I & 0 \\ Y(t) & I \end{pmatrix} ,$$

$$Z(t) = \sin(a_1 t) \begin{pmatrix} 1 & 1/3 \\ 1/3 & 9/4 \end{pmatrix} , \quad Y(t) = \sin(a_2 t) \begin{pmatrix} 7/2 & 50/3 \\ 50/3 & 4 \end{pmatrix} , \ t \in [0,1] ,$$

$a_1 = 0.45$, $a_2 = 0.45\sqrt{2}$. $A(t)$ is symplectic, but we do not know $\log(A(t))$, and $A(t)$ and $A(s)$ do not commute ($t \neq s$). With $A(0) = I$, $\log(A(0)) = 0$, $\log(A(t))$ is inside the principal branch; $\|I - S(t)\|$ grows as t nears 1 ($\|I - S(1)\| \simeq 13.2$).

Example 5.4 The positive definite matrix

$$A(t) = \exp\left(\frac{D(t)Q(t) + Q^T(t)D(t)}{2}\right),$$

where

$$D(t) = \begin{pmatrix} -1 + \frac{t}{2} & 0 & 0 & 0 \\ 0 & 1 - \frac{t}{2} & 0 & 0 \\ 0 & 0 & -\frac{1+t}{2} & 0 \\ 0 & 0 & 0 & \frac{1+t}{2} \end{pmatrix} , \ t \in [0,1] ,$$

and $Q(t)$ is $A(t)$ of Example 5.2(b).

Logarithms of nearby matrices We used Gauss N-point quadrature for the integral formula (2.12). In Table 1, we show results obtained on Example 5.1, with logarithms of $A + E$ computed to limiting precision. Let L_c be the computed logarithms. Then, we show $\mathbf{err} = \|e^{L_c} - (A+E)\|/\|A+E\|$; $\mathbf{err}_m = \|\log(A+E) - L_c\|/\|\log(A+E)\|$, where $\log(A+E)$ is the exact logarithm of $A + E$; in later tables, it is the principal logarithm returned by the

`Matlab` function `logm`. The number of points needed to obtain the required accuracy decreases with $\|E\|$. In Table 2, we highlight, for the same example, the difficulties occuring near the boundary of the principal branch. Eventually no accuracy is obtained with 16 Gauss points, but the relevant matrix structure is always preserved to machine precision.

Table 1. Formula (2.12) - Example 5.1: $A = A(0.1)$, $A + E = A(0.1 + \delta t)$

err	$\mathbf{err}_m$	$\|E\|/\|A\|$	N	$\|I - (A + E)\|$	δt
5.33e-16	3.61e-16	6.11e-01	16	1.17e+00	1.00e-01
1.18e-15	9.88e-16	6.25e-02	10	6.76e-01	1.00e-02
1.20e-15	1.05e-15	6.25e-03	9	6.23e-01	1.00e-03
2.57e-15	4.07e-15	6.25e-04	8	6.18e-01	1.00e-04

Table 2. Formula (2.12) - Example 5.1: $A = A(t - \pi/48)$, $A + E = A(t)$

err	$\mathbf{err}_m$	$\|E\|/\|A\|$	N	$\|I - (A + E)\|$	t
2.22e-16	2.87e-16	4.08e-01	8	4.08e-01	$\pi/48$
2.62e-15	1.84e-15	4.03e-01	15	1.15e+00	$3\pi/48$
1.52e-02	5.48e-03	3.72e-01	16	1.97e+00	$7\pi/48$
2.00e+00	9.95e-01	3.61e-01	16	2.00e+00	$8\pi/48$

Tables 3-5 present results obtained with Newton's method (2.4)-(2.7). In the first run of Newton's method (first line in each table), L_0 is the principal logarithm of A; for successive executions, we used the previously computed logarithm as initial guess. In these tables, k is the number of iterations; $\Delta L_k = \|L_k - L_{k-1}\|/\|L_k\|$; J is the number of Sylvester equations solved per iteration. Stopping criteria were: $\mathbf{err} < \texttt{tollf}$, or $\|L_k - L_{k-1}\| < \texttt{tolla} + \texttt{tollr}\|L_k\|$, with $\texttt{tolla} = \texttt{tollr} = 10^{-3}\sqrt{\texttt{EPS}}$, $\texttt{tollf} = 10^{-15}$.

In Table 3, the number of iterations needed for convergence increases close to the boundary of the principal branch, whereas in Table 4 more computation (see J) is needed when $(A + E)$ is far from the identity. From Table 5, on comparing **err** and $\mathbf{err}_m$, the iteration converged to primary but non-principal logarithms of $A(t)$.

Table 3. Newton's method - Example 5.2(a): $A = A(t - 0.05)$, $A + E = A(t)$

err	ΔL_k	k	$\mathbf{err}_m$	$\|E\|/\|A\|$	J	$\|I - (A + E)\|$	t
2.52e-14	4.64e-12	8	4.45e-14	2.26e-01	5	2.00e+00	0.70
2.37e-15	2.24e-13	10	1.92e-14	2.26e-01	5	2.00e+00	0.75
5.45e-15	7.48e-13	15	5.11e-14	2.26e-01	5	2.00e+00	0.80
4.09e-15	2.43e-13	10	4.61e-15	2.26e-01	5	2.00e+00	0.85
5.17e-14	8.74e-12	8	7.24e-14	2.26e-01	5	1.99e+00	0.90

Table 4. Newton's method - Example 5.3: $A = A(t_1)$, $A + E = A(t_2)$							
err	ΔL_k	k	$\mathbf{err}_m$	$\|E\|/\|A\|$	J	$\|I - (A+E)\|$	$t_1 \div t_2$
1.80e-15	1.17e-13	6	2.17e-15	7.02e-01	5	2.64e+00	$0.1 \div 0.2$
1.74e-15	2.39e-13	6	2.51e-15	3.38e-01	5	5.48e+00	$0.3 \div 0.4$
2.44e-14	2.09e-13	6	1.64e-15	1.65e-01	8	1.05e+01	$0.8 \div 0.9$
4.31e-14	4.17e-12	7	2.76e-14	1.49e-01	10	1.65e+01	$0.9 \div 1.0$

Table 5. Newton's method - Example 5.2(b): $A = A(t_1)$, $A + E = A(t_2)$							
err	ΔL_k	k	$\mathbf{err}_m$	$\|E\|/\|A\|$	J	$\|I - (A+E)\|$	$t_1 \div t_2$
3.43e-15	1.89e-13	8	2.02e+00	8.05e-01	5	2.00e+00	$0.70 \div 0.80$
4.85e-15	1.70e-13	7	2.12e+00	4.13e-01	5	1.99e+00	$0.80 \div 0.85$

Interpolation. Tables 6-8 contain results for interpolating the data set $(t_i,\, A_i)_{i=0}^{N}$ by the log-exp/Cayley methods and their rescaled versions. For these, and all other cases we examined, the following trends are observed. Rescaling the data (see Section 3) is beneficial for symplectic and orthogonal sequences (Tables 6, 8), but not for positive definite sequences (Table 7). The log-exp method is consistently more accurate than the Cayley method. If data points are available to arbitrarily raise the order of the interpolant, the Cayley method is less expensive than the log–exp method for a given accuracy requirement. In Tables 6-8, the spacing $h = (t_N - t_0)/N$, **Err** is an estimate of the maximum relative error at the midpoint of the interval: $\mathbf{Err} = \max_j \|A(s_j) - \tilde{A}(s_j)\|/\|A(s_j)\|$, where $s_j = t_m + j(t_N - t_0)/200,\ j = -10, -9, \ldots, 10,\ t_m = (t_N + t_0)/2$, and the error for the rescaled versions is $\mathbf{Err}_s$. The relevant matrix structure (orthogonality, simplecticity, positive definiteness) was always maintained by both the log-exp and Cayley methods. From Table 6, note that the errors for the unrescaled Cayley method may be explained by a large norm of derivatives of $C(t)$ (see Theorem 3.5). Finally, in Table 8, the rescaled methods are compared in a singular case.

Table 6. log-exp/Cayley methods - Example 5.2(a): $[t_0,\, t_N] = [0.5,\, 1]$					
$N+1$	h	log-exp **Err**	log-exp $\mathbf{Err}_s$	Cayley **Err**	Cayley $\mathbf{Err}_s$
4	1.67e-01	5.88e-01	6.79e-04	2.00e+00	4.26e-03
8	7.14e-02	3.90e-02	8.57e-08	1.83e+00	8.55e-07
12	4.55e-02	2.10e-03	1.52e-11	7.11e-01	2.02e-10
16	3.33e-02	6.84e-05	3.11e-15	1.85e+00	4.79e-14
20	2.63e-02	2.26e-05	1.03e-15	2.00e+00	4.92e-16

Table 7. log-exp/Cayley methods - Example 5.4: $[t_0,\ t_N] = [0.25,\ 0.5]$					
$N+1$	h	log-exp $\mathbf{Err}$	log-exp $\mathbf{Err}_s$	Cayley $\mathbf{Err}$	Cayley $\mathbf{Err}_s$
4	8.33e-02	5.77e-03	1.05e-02	6.79e-03	1.42e-02
12	2.27e-02	7.23e-13	4.22e-09	9.32e-10	1.40e-08
16	1.67e-02	5.73e-15	3.39e-12	4.07e-13	1.43e-11

Table 8. Scaled log-exp/Cayley methods - Example 5.1: $[t_0,\ t_N] = [0.3,\ 0.8]$			
$N+1$	h	**log-exp $\mathbf{Err}_s$**	**Cayley $\mathbf{Err}_s$**
4	1.67e-01	5.93e-05	1.41e-03
8	7.14e-02	2.37e-12	4.68e-06
12	4.55e-02	8.52e-16	5.66e-09
16	3.33e-02	-	6.26e-12
24	2.17e-02	-	8.06e-16

References

[1] P. Antsaklis and Z. Gao, Polynomial and rational matrix interpolation: theory and control applications, *Int. J. Control* 58 (1993) 349–404.

[2] L. Dieci, Considerations on computing real logarithms of matrices, Hamiltonian logarithms, and skew-symmetric logarithms, *Linear Algebra Appl.* 244 (1996) 35-54.

[3] L. Dieci, Real Hamiltonian logarithm of a symplectic matrix, preprint (1997).

[4] L. Dieci, B. Morini and A. Papini, Computational techniques for real logarithms of matrices, *SIAM J. Matrix Anal. Appl.* 17 (1996) 570-593.

[5] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, (Cambridge University Press, Cambridge, 1991).

[6] K. R. Jackson and W. L. Seward, Adaptive linear equation solvers in codes for large stiff systems of ODEs, *SIAM. J. Sci. Stat. Comput.* 14 (1993) 800-823.

[7] C. Kenney and A. J. Laub, Padé error estimates for the logarithm of a matrix, *Int. J. Control*, 50 (1989) 707-730.

[8] C. Kenney and A. J. Laub, Condition estimates for matrix functions, *SIAM J. Matrix Anal. Appl.* 10 (1989) 191-209.

[9] *Matlab Reference Guide*, (The MathWorks, Inc., Natick, MA, 1992).

[10] E. Newman, Generation of wide-band data of moments by interpolating the impedance matrix, *IEEE Trans. Antennas & Propagation* 36 (1988) 1820-1824.

[11] Y. Sibuya, Note on real matrices and linear dynamical systems with periodic coefficients, *J. Math. Anal. Appl.* 1 (1960) 363-372.

[12] C. Van Loan, The sensitivity of the matrix exponential, *SIAM J. Numer. Anal.* 14 (1977) 971-981.