Computing the Wave-Kernel Matrix Functions

Nadukandi, Prashanth and Higham, Nicholas J.

2018

MIMS EPrint: **2018.4**

Manchester Institute for Mathematical Sciences
School of Mathematics

The University of Manchester

Reports available from: http://eprints.maths.manchester.ac.uk/
And by contacting: The MIMS Secretary
School of Mathematics
The University of Manchester
Manchester, M13 9PL, UK

ISSN 1749-9097

COMPUTING THE WAVE-KERNEL MATRIX FUNCTIONS*

PRASHANTH NADUKANDI† AND NICHOLAS J. HIGHAM†

Abstract. We derive an algorithm for computing the wave-kernel functions $\cosh\sqrt{A}$ and $\operatorname{sinhc}\sqrt{A}$ for an arbitrary square matrix A, where $\operatorname{sinhc} z = \sinh(z)/z$. The algorithm is based on Padé approximation and the use of double angle formulas. We show that the backward error of any approximation to $\cosh\sqrt{A}$ can be explicitly expressed in terms of a hypergeometric function. To bound the backward error we derive and exploit a new bound for $\|A^k\|^{1/k}$ that is sharper than one previously obtained by Al-Mohy and Higham (*SIAM J. Matrix Anal. Appl.*, 31(3):970–989, 2009). The amount of scaling and the degree of the Padé approximant are chosen to minimize the computational cost subject to achieving backward stability for $\cosh\sqrt{A}$ in exact arithmetic. Numerical experiments show that the algorithm behaves in a forward stable manner in floating-point arithmetic and is superior in this respect to the general purpose Schur–Parlett algorithm applied to these functions.

Key words. wave kernel, matrix function, Padé approximation, backward stability, hypergeometric function, matrix norm estimation

AMS subject classifications. 15A60, 65F30, 65F60

1. Introduction.

The general solution of the scalar wave equation

$$\frac{\partial^2}{\partial t^2}u(x,t) - \Delta u(x,t) = b(x,t), \tag{1.1a}$$

$$u(x,0) = f(x), \quad \frac{\partial}{\partial t}u(x,0) = g(x), \tag{1.1b}$$

where Δ is the Laplacian operator in x, has the formal expression [13, p. 119], [14], [32, p. x],

$$u(x,t) = \cos(t\sqrt{-\Delta})f + t\operatorname{sinc}(t\sqrt{-\Delta})g + \int_0^t (t-s)\operatorname{sinc}\big((t-s)\sqrt{-\Delta}\big)b(\cdot,s)\,\mathrm{d}s. \tag{1.2}$$

Here, $\cos(t\sqrt{-\Delta}) = \cosh t\sqrt{\Delta}$, $\operatorname{sinc}(t\sqrt{-\Delta}) = \operatorname{sinhc} t\sqrt{\Delta}$, and $\operatorname{sinhc} z = \sinh(z)/z$, with $\operatorname{sinhc} 0 \coloneqq 1$.

The two fundamental solutions to (1.1) are obtained by applying the operators $\cosh t\sqrt{\Delta}$ and $t\operatorname{sinhc} t\sqrt{\Delta}$ to the Dirac delta function. These solutions are the kernels of the linear (integral) transformation that maps the external input $b(x,t)$ and the initial data $f(x)$ and $g(x)$ to the general solution of (1.1). Greiner et al. [14] have explicitly computed the wave kernels for several subelliptic second-order operators.

We will focus on the algebraic second-order Cauchy problem where f, g, b, and u are vectors in $\mathbb{C}^n$ independent of x and $A \in \mathbb{C}^{n\times n}$ is an arbitrary square matrix:

$$u''(t) - Au(t) = b(t), \qquad u(0) = u_0, \quad u'(0) = u_0'. \tag{1.3}$$

Such linear second-order ODE systems are obtained (for instance) from a semidiscretization of (1.1) by the finite element method. For this algebraic system the wave kernels are the matrix functions $\cosh t\sqrt{A}$ and $t\operatorname{sinhc} t\sqrt{A}$.

*Version of August 1, 2018.

† School of Mathematics, The University of Manchester Manchester, M13 9PL, United Kingdom. (prashanth.nadukandi@manchester.ac.uk, nick.higham@manchester.ac.uk).

Funding: Nadukandi was supported by an individual fellowship from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 702138. Higham was supported by Engineering and Physical Sciences Research Council grant EP/P020720/1 and the Royal Society.

In view of these connections we will call the functions $\cosh\sqrt{z}$ and $\operatorname{sinhc}\sqrt{z}$ the wave-kernel functions. In this paper we derive a new algorithm for computing the wave-kernel matrix functions $\cosh\sqrt{A}$ and $\operatorname{sinhc}\sqrt{A}$ for an arbitrary square matrix A. We emphasize that A in (1.3) is the given matrix. Typically (for example, in [6], [19]), the matrix in (1.3) is assumed to be given in the form A^2. Treating general A presents new challenges for the backward error analysis, as we will see.

The wave-kernel functions have the power series representations

$$\cosh\sqrt{z} = \sum_{n=0}^{\infty} \frac{z^n}{(2n)!}, \quad \operatorname{sinhc}\sqrt{z} = \sum_{n=0}^{\infty} \frac{z^n}{(2n+1)!}. \tag{1.4}$$

Both series have an infinite radius of convergence and hence both are entire functions (analytic in the whole complex plane). Since cosh and sinhc are even functions there are no square root terms in (1.4) and so in the matrix case there are no questions about the existence of the matrix square root or of which square root to take.

In many applications A is symmetric, but nonsymmetric A arise in stability and position feedback control of circulatory systems [33, chap. 5], constrained external damping in rotatory shafts [23, p. 43], frictional contact stability and control of robot grasping arrangements [29, chap. 4], [30], and semi-Lagrangian formulation of flows [25].

The stability analysis of second-order ODE systems is done in the frequency domain assuming a time periodic external input $b(t)$. In applications where $b(t)$ is a non-periodic function of time we have to work in the time domain. The time integration of stiff ODE systems is a challenging task. The wave-kernel matrix functions are useful for deriving accurate time integrators suitable for stiff second-order ODE systems.

When A is a large, possibly sparse matrix there are various approaches to computing the action of a matrix function $f(A)$ on a vector b [18, chap. 13]. One is to generate approximations to $f(A)b$ from a Krylov subspace $\mathcal{K}(A,b)$ [16]. Krylov subspace methods reduce the approximation of $f(A)b$ to the computation of $f(H)e_1$ for a much smaller upper Hessenberg matrix H, where e_1 is the first unit vector. Another approach is to apply a series approximation along with a suitable scaling strategy [4].

In this work we develop algorithms for computing the wave-kernel matrix functions based on Padé approximation. The algorithms scale the matrix ($A \leftarrow 4^{-s}A$), evaluate a Padé approximant, then undo the effect of the scaling via recurrences. The amount of scaling and the Padé degree are based on the backward error of the Padé approximant to $\cosh\sqrt{4^{-s}A}$. We obtain an explicit expression for the backward error, valid for any rational approximation, involving a hypergeometric function. For Padé approximants we expand this expression in a power series and bound it in terms of quantities $\|A^k\|^{1/k}$. Our technique for exploiting these quantities is a refinement of that introduced by Al-Mohy and Higham [3] and yields bounds never larger and possibly much smaller. The resulting algorithm is backward stable for computing $\cosh\sqrt{A}$ and mixed forward–backward stable for computing $\operatorname{sinhc}\sqrt{A}$, where stability is with respect to truncation error in exact arithmetic.

Prior work on computing the wave-kernel matrix functions and their action on vectors has mainly been restricted to the case where the matrix A is symmetric positive definite [11], [12], [15], [31]. An exception is Al-Mohy's recent work [2], wherein algorithms to compute the action of trigonometric and hyperbolic matrix functions are derived for any square matrix A. The wave-kernel matrix functions are included as a special case. The approach taken therein is to bound the absolute forward

error of approximations based on truncated Taylor series of the matrix functions evaluated at a scaled value of the matrix A. Extending Al-Mohy's analysis to bound the relative forward error is desirable but appears difficult, because it would require a tight lower bound on the norm of the matrix function. Our approach of bounding the (relative) backward error provides a scale-independent measure and it avoids any need for consideration of condition numbers when assessing bounds.

To obtain the backward error result needed to derive our algorithm we need to understand the behavior of the inverse of the function $\cosh\sqrt{z}$. The necessary results are given in section 2.

In section 3 we derive a new bound for the norm of a general matrix power series in terms of bounds for the quantities $\max_{k\geq 2m}\|A^k\|^{1/k}$. The backward error analysis, and its application to Padé approximants, is given in section 4. Our algorithm for computing the wave-kernel matrix functions is presented in section 5, where careful attention is given to the choice of the parameter s (the amount of scaling) and m (the Padé degree).

The Schur–Parlett algorithm [10], [18, chap. 9], designed for general matrix functions, can also be used to compute the wave-kernel matrix functions. This algorithm requires the ability to compute the derivatives at scalar arguments of the wave-kernel functions, which are given by

$$\frac{\mathrm{d}^k}{\mathrm{d}z^k}\cosh\sqrt{z} = \sum_{n\geq 0}\frac{(n+1)_k}{(2k+2n)!}z^n = \frac{1}{(k+1)_k}{}_0F_1\left(;k+\frac{1}{2};\frac{z}{4}\right), \tag{1.5a}$$

$$\frac{\mathrm{d}^k}{\mathrm{d}z^k}\operatorname{sinhc}\sqrt{z} = \sum_{n\geq 0}\frac{(n+1)_k}{(2k+2n+1)!}z^n = \frac{1}{(k+1)_{k+1}}{}_0F_1\left(;k+\frac{3}{2};\frac{z}{4}\right), \tag{1.5b}$$

where $(a)_n$ is the Pochhammer symbol and ${}_0F_1(;a;z)$ is a hypergeometric function, both of which are defined in appendix A.2. Numerical experiments are given in section 6 to test the practical behavior of our algorithm and to compare it with the Schur–Parlett algorithm.

2. Fundamental regions and principal inverse of $\cosh\sqrt{z}$. We begin by developing understanding of the inverse of $\cosh\sqrt{z}$ that will be needed for the backward error analysis.

A region that is mapped by a function in a one-to-one manner onto the whole complex plane, except for one or more cuts, is called a fundamental region of that function [1, p. 98].

LEMMA 2.1 (Fundamental regions of $\cosh\sqrt{\ }$). *As n runs over positive integers, the parametric curves*

$$\Gamma_n(t) \coloneqq t^2 - n^2\pi^2 + \mathrm{i}\,(2n\pi t)$$

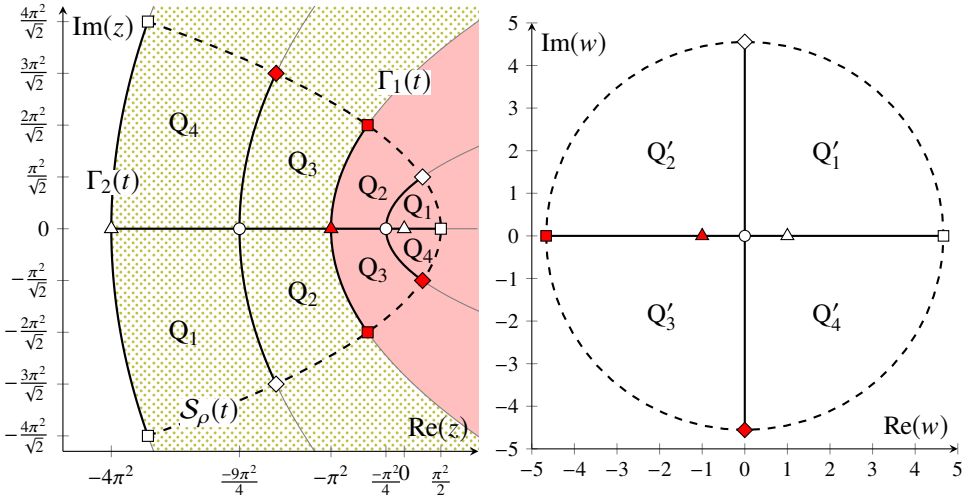
divide the complex plane into fundamental regions of $\cosh\sqrt{z}$.

Proof. The parametric curves Γ_n are motivated by the identity

$$\cosh\sqrt{(\rho^2-\lambda^2)+\mathrm{i}2\lambda\rho} = \cosh\rho\cos\lambda + \mathrm{i}\sinh\rho\sin\lambda, \tag{2.1}$$

where $\lambda,\rho\in\mathbb{R}$. The segments of the parametric curve

$$\mathcal{S}_\rho(t) \coloneqq \rho^2 - t^2 + \mathrm{i}\,(2\rho t) \tag{2.2}$$



(a) Domain (z-plane). (b) Range (w-plane).

Fig. 2.1: The fundamental regions Ω_0 (pink solid fill) and Ω_1 (green dotted pattern) are shaded in the z-plane. The function $\cosh\sqrt{z}$ maps regions labelled Q_i in the domain (z-plane) to regions labelled Q_i' in the range (w-plane). The curve $\mathcal{S}_\rho(t)$ defined in (2.2), with $\rho = \pi/\sqrt{2}$, is shown as a dashed line in the z-plane. Points on $\mathcal{S}_\rho(t)$ map to the elliptic curve $(\cosh\rho\cos t) + \mathrm{i}(\sinh\rho\sin t)$ shown as a dashed line in the w-plane. Some salient points in the w-plane are shown by uniquely shaded markers: ○, △, ▲, □, ■, ◇, ◆, and their pre-images are shown in the z-plane. With the aid of these marker points we can associate every curve shown in the z-plane with a corresponding curve in the w-plane.

that lie in the strict interior of the region bounded by $\Gamma_n(t)$ and $\Gamma_{n+1}(t)$ are $\{\,\mathcal{S}_\rho(t) : n\pi < t < (n+1)\pi\,\}$ and $\{\,\mathcal{S}_\rho(t) : -(n+1)\pi < t < -n\pi\,\}$. To fix ideas we show $\Gamma_1(t)$, $\Gamma_2(t)$, and $\mathcal{S}_\rho(t)$ with $\rho = \pi/\sqrt{2}$, in Figure 2.1a.

Using (2.1), we find that when $\rho > 0$, $\cosh\sqrt{\ }$ maps the $\mathcal{S}_\rho$ curve segments to the strict upper and strict lower segments of the elliptic curve $\cosh\rho\cos t + \mathrm{i}\sinh\rho\sin t$ in a bijective manner. As $\rho \to 0$, the $\mathcal{S}_\rho$ curve segments converge to the line segment S_0, and $\cosh\sqrt{\ }$ maps the corresponding $\mathcal{S}_0$ line segment to the line $-1 < w < 1$ in a bijective manner. By varying ρ from 0 to ∞ the $\mathcal{S}_\rho$ curve segments will sweep out the strict interior of the region bounded by $\Gamma_n(t)$ and $\Gamma_{n+1}(t)$. The image of the $\mathcal{S}_\rho$ curve segments sweep out the entire w-plane except for two cuts $(-\infty, -1)$ and $(1, \infty)$ along the real axis. Here $w = -1$ and $w = 1$ are branch points.

The proof that the convex region of $\Gamma_1(t)$ is a fundamental region of $\cosh\sqrt{\ }$ proceeds in a similar fashion by considering the curve segment $\{\,\mathcal{S}_\rho(t) : -\pi < t < \pi\,\}$. The image of this $\mathcal{S}_\rho$ curve segment sweeps out the entire w-plane except for a cut $(-\infty, -1)$. Here only $w = -1$ is a branch point. ∎

We denote by Ω_n the fundamental region bounded by $\Gamma_n(t)$ and $\Gamma_{n+1}(t)$, and by Ω_0 the convex region of $\Gamma_1(t)$. In Figure 2.1a we show Ω_0 (pink shading) and Ω_1 (green dotted shading).

The curve $\Gamma_n(t)$ corresponds to both edges of the positive cut if n is even, and

to the edges of the negative cut if n is odd. To maintain a bijective mapping, we will include the curve segments $\Gamma_n(t<0)$ and $\Gamma_{n+1}(t<0)$ in Ω_n if n is odd. If n is even, then the curve segments $\Gamma_n(t\geq 0)$ and $\Gamma_{n+1}(t\geq 0)$ are included in Ω_n. The curve segment $\Gamma_1(t\geq 0)$ is included in Ω_0.

DEFINITION 2.2 (Principal domain of $\cosh\sqrt{}$). *We call the fundamental region Ω_0 the principal domain. It contains the origin (marked $\triangle$ in Figure 2.1a), whose image in the w-plane is not a branch point.*

The fundamental regions of $\cosh\sqrt{}$ are the branches of its compositional inverse.

DEFINITION 2.3 (Principal inverse of $\cosh\sqrt{}$). *Let w belong to the complex plane with a cut along the real axis from $-\infty$ to -1 and let z belong to the principal domain Ω_0. The principal inverse $(\cosh\sqrt{})^{-1}$ is the bijective mapping $w\to z$ such that $w=\cosh\sqrt{z}$.*

LEMMA 2.4. *The principal inverse $(\cosh\sqrt{})^{-1}$ is analytic at all points other than those on the branch cut along the real axis from $-\infty$ to -1.*

Proof. Since $\cosh\sqrt{}$ is entire and its derivative is nonzero[1] for any interior point $a\in\Omega_0$, we can use the Lagrange inversion theorem (see appendix A.1) to express $(\cosh\sqrt{})^{-1}$ as a power series that converges in some neighbourhood of $\cosh\sqrt{a}$. Thus $(\cosh\sqrt{})^{-1}$ is analytic at $\cosh\sqrt{a}$. Hence $(\cosh\sqrt{})^{-1}$ is analytic at all points other than those on the branch cut. ∎

A consequence of Lemma 2.4 is that the radius of convergence of the power series of $(\cosh\sqrt{})^{-1}$ about $\cosh\sqrt{a}$ is equal to $|1+\cosh\sqrt{a}|$ which is the distance of $\cosh\sqrt{a}$ to the nearest branch point $w=-1$.

The sum of a convergent power series of a multi-valued function might fall in a branch different from the principal branch. Should this be the case, the equality of the function to its power series will not hold. For the equality to hold the disc of convergence should not touch or cross the specified branch cut. We will use the power series of $(\cosh\sqrt{})^{-1}$ about the point $w=1$ and the largest disc centred at this point touches the branch cut at the branch point $w=-1$. Hence, the equality of $(\cosh\sqrt{})^{-1}$ with its power series about $w=1$ holds inside the disc $|w-1|<2$.

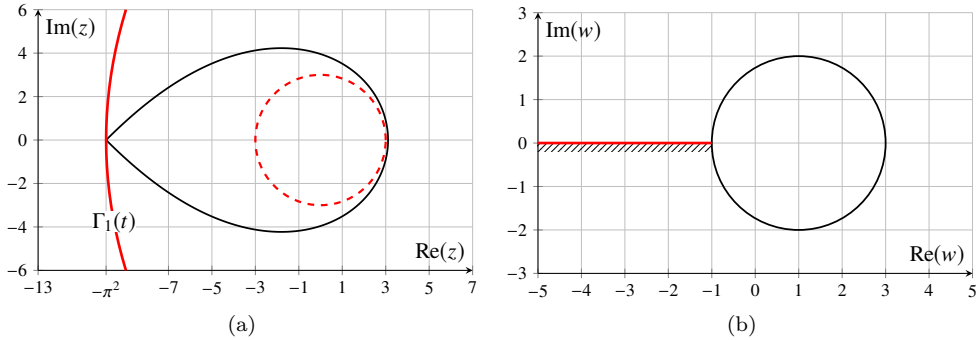
The power series of $(\cosh\sqrt{})^{-1}$ about the point $w=\cosh\sqrt{0}=1$ can be shown, using the Lagrange inversion theorem, to be

$$\begin{aligned}(\cosh\sqrt{})^{-1}w &= \sum_{n=1}^{\infty}\frac{(w-1)^n}{n!}\lim_{x\to 0}\left[\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left(\sum_{m=0}^{\infty}\frac{x^m}{(2m+2)!}\right)^{-n}\right]\\ &= 2(w-1)-\frac{1}{3}(w-1)^2+\frac{4}{45}(w-1)^3-\frac{1}{35}(w-1)^4+\frac{16}{1575}(w-1)^5\\ &\quad -\frac{8}{2079}(w-1)^6+\frac{32}{21021}(w-1)^7-\frac{4}{6435}(w-1)^8+\cdots,\ |w-1|<2. \qquad (2.3)\end{aligned}$$

The preimage of the disc $|w-1|\leq 2$ in the principal domain is shown in Figure 2.2a. Note that it includes the origin and contains the disc $|z|\leq 3$ (dashed line). In the next lemma we show that the power series (2.3) has a succinct hypergeometric representation. This representation is invaluable because for some rational function $h(z)$ we will later want to evaluate partial sums of the power series of $(\cosh\sqrt{})^{-1}h(z)$

[1] The derivative of $(\cosh\sqrt{z})'$ is zero only at points of the form $z=-n^2\pi^2$ for integer $n\geq 1$, which do not belong to the interior of the principal domain Ω_0.

about $z = 0$ and we can delegate the change in expansion point to the computer algebra package Maple, which has knowledge of the hypergeometric function.



(a) (b)

Fig. 2.2: (a) Preimage of the disc $\{\, w : |w-1| \le 2 \,\}$ in the principal domain, along with the disc $|z| \le 3$ (dashed line). (b) The disc $\{\, w : |w-1| \le 2 \,\}$ and the branch cut $(-\infty, -1)$ in the range.

LEMMA 2.5. *The principal inverse* $(\cosh\sqrt{\ })^{-1}$ *has the hypergeometric representation*

$$(\cosh\sqrt{\ })^{-1} w = 2(w-1)\, {}_3F_2\left(1,1,1;\frac{3}{2},2;\frac{1-w}{2}\right), \qquad |w-1| \le 2. \tag{2.4}$$

Proof. A series expansion of $\cosh^{-1} w$ [26, eq. (4.38.4)] about the point $w = 1$ is

$$\cosh^{-1} w = \sqrt{2(w-1)}\left[1 + \sum_{n=1}^{\infty} \frac{1\cdot 3\cdot 5\cdots(2n-1)}{2^{2n} n!(2n+1)}(1-w)^n\right], \ \operatorname{Re} w > 0, \ |w-1| < 2.$$

Using the equations

$$1\cdot 3\cdot 5\cdots(2n-1) = \frac{1}{2}\left(\frac{1}{2}+1\right)\left(\frac{1}{2}+2\right)\cdots\left(\frac{1}{2}+n-1\right)2^n = \left(\frac{1}{2}\right)_n 2^n,$$
$$\left(\frac{1}{2}\right)_{n+1} = \left(\frac{1}{2}\right)_n \frac{2n+1}{2} = \frac{1}{2}\left(\frac{3}{2}\right)_n,$$

we can express

$$\begin{aligned} 1 + \sum_{n=1}^{\infty} \frac{1\cdot 3\cdot 5\cdots(2n-1)}{2^{2n} n!(2n+1)}(1-w)^n &= 1 + \sum_{n=1}^{\infty} \frac{\left(\frac{1}{2}\right)_n \left(\frac{1}{2}\right)_n}{\left(\frac{3}{2}\right)_n n!}\left(\frac{1-w}{2}\right)^n \\ &= {}_2F_1\left(\frac{1}{2},\frac{1}{2};\frac{3}{2};\frac{1-w}{2}\right), \end{aligned}$$

and hence

$$\cosh^{-1} w = \sqrt{2(w-1)}\, {}_2F_1\left(\frac{1}{2},\frac{1}{2};\frac{3}{2};\frac{1-w}{2}\right), \quad \operatorname{Re} w > 0, \ |w-1| < 2. \tag{2.5}$$

Let us now write the power series of $(\cosh\sqrt{\ })^{-1}w$ given in (2.3) in the form

$$(\cosh\sqrt{\ })^{-1}w = 2(w-1)\sum_{n=0}^{\infty} c_n(1-w)^n, \quad |w-1|<2. \tag{2.6}$$

Equations (2.5) and (2.6), and Clausen's identity [8]

$${}_2F_1\Big(a,b;a+b+\frac{1}{2};\xi\Big)^2 = {}_3F_2\Big(2a,2b,a+b;2a+2b,a+b+\frac{1}{2};\xi\Big)$$

with $a=b=1/2$ and $\xi=(1-w)/2$, are used in the relation $(\cosh\sqrt{\ })^{-1}w = (\cosh^{-1}w)^2$ to arrive at

$$\sum_{n=0}^{\infty} c_n(1-w)^n = {}_3F_2\Big(1,1,1;\frac{3}{2},2;\frac{1-w}{2}\Big), \quad \operatorname{Re} w>0,\ |w-1|<2. \tag{2.7}$$

As we have only nonnegative integer powers of $1-w$ in (2.7) and ${}_3F_2(1,1,1;3/2,2;1-w/2)$ converges for $|w-1|=2$ (see appendix A.2), the equality holds in the disc $|w-1|\le 2$ without the restriction $\operatorname{Re} w>0$. Thus we obtain (2.4). ∎

3. Bounding a matrix power series. In the design of our algorithm we will need to bound the norm of a matrix power series that represents the error in an approximation. This is a standard requirement in algorithms based on Padé approximants [3], [5], [6], [7], [20]. In this section we derive a new bound for the norm of an arbitrary matrix power series

$$g_\ell(A) = \sum_{i=\ell}^{\infty} c_i A^i.$$

We denote by $\|\cdot\|$ any consistent matrix norm with $\|I\|=1$.

Al-Mohy and Higham [3, Thm. 1.1] note that

$$\|g_\ell(A)\| \le \sum_{i=\ell}^{\infty} |c_i|\|A^i\| = \sum_{i=\ell}^{\infty} |c_i| \left(\|A^i\|^{1/i}\right)^i \le \sum_{i=\ell}^{\infty} |c_i|\beta^i,$$

where $\beta = \max_{i\ge\ell}\|A^i\|^{1/i}$. The motivation for this bound is that $\|A^i\|^{1/i}$ satisfies $\rho(A)\le\|A^i\|^{1/i}\le\|A\|$ and can be much smaller than $\|A\|$ for a nonnormal matrix, so the bound can be much smaller than $\sum_{i=\ell}^{\infty}|c_i|\|A\|^i$.

In seeking a more easily computed quantity than β, Al-Mohy and Higham [3, Lem. 4.1] show that if a,b,i,j are nonnegative integers such that $ai+bj\ge 1$ then

$$\|A^{ai+bj}\|^{1/(ai+bj)} \le \max\big(\|A^a\|^{1/a}, \|A^b\|^{1/b}\big). \tag{3.1}$$

We specialize this result as follows. Denote by $\gcd(a,b)$ the greatest common divisor of a and b.

THEOREM 3.1. *Let a, b, k, and m be positive integers. Then*

$$\alpha_m(A) = \min_{\substack{\gcd(a,b)=1,\\ ab-a-b<2m}} \max\big(\|A^a\|^{1/a}, \|A^b\|^{1/b}\big) \tag{3.2}$$

satisfies

$$\max_{k\ge 2m}\|A^k\|^{1/k} \le \alpha_m(A) \le \|A\|. \tag{3.3}$$

Furthermore, $\alpha_m(A)$ is nonincreasing in m.

Proof. Observe that as i and j run independently over the nonnegative integers the values of $ai + bj$ run over a certain subset of the nonnegative integers. If a and b are co-prime, that is, $\gcd(a, b) = 1$, it is well known that this subset includes all positive integers greater than $ab-a-b$. The number $ab-a-b$ is called[2] the Frobenius number [28] of the set $\{a, b\}$. If $ab - a - b < 2m$ then from (3.1) we get the bound $\max_{k\geq 2m} \|A^k\|^{1/k} \leq \max\left(\|A^a\|^{1/a}, \|A^b\|^{1/b}\right)$. Taking the minimum of these bounds over all co-prime a and b we obtain the lower bound in (3.3). That α_m is nonincreasing in m is because the set of a and b in the minimum defining α_m grows with m. ∎

Al-Mohy and Higham [3, Thm. 4.2] chose $b = a + 1$ in (3.1), for which the co-prime condition is naturally satisfied and the condition $ab - a - b < 2m$ simplifies to $(a - 1)a \leq 2m$. However, a stronger bound is obtained by not limiting the co-primes in Theorem 3.1, as the following example confirms.

Example 3.2. The columns of the matrix

$$\begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 3 & 3 \\ 3 & 5 & 7 & 9 & 11 & 4 & 5 \end{bmatrix}$$

represent all possible co-primes a, b satisfying $ab - a - b < 2m$ for $m = 5$ (we exclude pairs with a or b equal to 1, as this case simply gives $\|A^k\| \leq \|A\|^k$). Using the inequality $\max\left(\|A^a\|^{1/a}, \|A^{a+b}\|^{1/(a+b)}\right) \leq \max\left(\|A^a\|^{1/a}, \|A^b\|^{1/b}\right)$ the set of co-primes needed to compute $\alpha_5(A)$ reduces to

$$\begin{bmatrix} 2 & 3 & 3 \\ 11 & 4 & 5 \end{bmatrix}$$

and so

$$\alpha_5(A) = \min \max \begin{bmatrix} \|A^2\|^{1/2} & \|A^3\|^{1/3} & \|A^3\|^{1/3} \\ \|A^{11}\|^{1/11} & \|A^4\|^{1/4} & \|A^5\|^{1/5} \end{bmatrix},$$

where the max operates along the columns of the matrix and produces a row vector. Choosing A to be any involutory matrix with $\|A\| > 1$ we get

$$\max_{k\geq 10} \|A^k\|^{1/k} \leq \alpha_5(A) = \min \max \begin{bmatrix} 1 & \|A\|^{1/3} & \|A\|^{1/3} \\ \|A\|^{1/11} & 1 & \|A\|^{1/5} \end{bmatrix} = \|A\|^{1/11},$$

which is smaller than the bound obtained by Al-Mohy and Higham [3, Thm. 4.2]

$$\begin{aligned} \min_{(a-1)a\leq 10} \max\left(\|A^a\|^{1/a}, \|A^{a+1}\|^{1/(a+1)}\right) &= \min \max \begin{bmatrix} \|A^2\|^{1/2} & \|A^3\|^{1/3} \\ \|A^3\|^{1/3} & \|A^4\|^{1/4} \end{bmatrix} \\ &= \min \max \begin{bmatrix} 1 & \|A\|^{1/3} \\ \|A\|^{1/3} & 1 \end{bmatrix} = \|A\|^{1/3}, \end{aligned}$$

by a factor of $\|A\|^{8/33}$, which can be arbitrarily large because an involutory matrix can have arbitrarily large norm (for example, the matrix $\left[\begin{smallmatrix} 1-b & b \\ 2-b & b-1 \end{smallmatrix}\right]$) is involutory for any b). This improvement can lead to a saving of several matrix multiplications in our algorithm, and indeed other algorithms with a similar derivation, such as the scaling and squaring algorithm for the matrix exponential [3].

[2]We thank Hung Bui and Sean Prendiville for pointing this out during a discussion on the Euclidean algorithm.

A drawback of the α_m, compared with the quantities used by Al-Mohy and Higham with $b = a + 1$, is that they involve norms of higher powers of A, so in principle are more expensive to compute. Two factors mitigate the expense. First, we will estimate the norms without computing the matrix powers explicitly, making the overall cost $O(n^2)$ flops compared with the $O(n^3)$ flops cost of the whole algorithm when A is dense. Second, we will exploit matrix powers that are explicitly computed within the algorithm in order to reduce the cost further.

4. Error analysis for the wave kernels.

4.1. Approximation error. Let $h(z)$ denote an approximation to $\cosh\sqrt{z}$ for z in a disc centered at the origin such that $|h(z) - \cosh\sqrt{z}| \to 0$ as $z \to 0$. The forward error $e(z)$ of the approximation $h(z)$ to $\cosh\sqrt{z}$ is defined by

$$e(z) = h(z) - \cosh\sqrt{z}. \tag{4.1}$$

For z in the principal domain Ω_0, the backward error $E(z)$ of the approximation $h(z)$ to $\cosh\sqrt{z}$ is defined using the principal inverse as

$$E(z) = (\cosh\sqrt{\ })^{-1}h(z) - z, \tag{4.2}$$

so that

$$\cosh\sqrt{z} \approx h(z) = \cosh\sqrt{z + E(z)} = \cosh\sqrt{z} + e(z). \tag{4.3}$$

As $(\cosh\sqrt{\ })^{-1}$ is analytic everywhere except on its branch cut, E is analytic if h is analytic and does not take values on this branch cut.

For a given tolerance ϵ we wish to identify a disc centered at the origin such that $|E(z)| \le \epsilon|z|$ inside that disc. In order to do this we need a representation to quantify the backward error.

LEMMA 4.1. *For all z in the principal domain of $\cosh\sqrt{\ }$, if $h(z)$ is any approximation to $\cosh\sqrt{z}$ such that $|1 - h(z)| \le 2$ then the backward error $E(z)$ has the hypergeometric representation*

$$E(z) = 2\big(h(z) - 1\big)\,{}_3F_2\left(1, 1, 1; \frac{3}{2}, 2; \frac{1 - h(z)}{2}\right) - z. \tag{4.4}$$

Proof. For any z such that $|1 - h(z)| \le 2$, the hypergeometric series

$${}_3F_2\left(1, 1, 1; \frac{3}{2}, 2; \frac{1 - h(z)}{2}\right)$$

converges. If z belongs to the intersection of this region with the principal domain Ω_0, then the backward error result follows from (2.4) and (4.2). ∎

Our attempts to identify the fundamental regions of $\operatorname{sinhc}\sqrt{\ }$ were not fruitful. Without this knowledge the backward error in the approximations to $\operatorname{sinhc}\sqrt{z}$ cannot be uniquely defined. So we construct approximations to $\operatorname{sinhc}\sqrt{z}$ using $h(z)$ and derive mixed forward–backward error bounds.

LEMMA 4.2. *For all z in the principal domain of $\cosh\sqrt{\ }$, if $h(z)$ is any approximation to $\cosh\sqrt{z}$ such that $|1 - h(z)| \le 2$ then for $E(z)$ given in (4.4) then*

(a) $\operatorname{sinhc}\sqrt{z} \approx 2h'(z) = \big(1 + E'(z)\big)\operatorname{sinhc}\sqrt{z + E(z)}$, and

(b) $E'(z)$ has the hypergeometric representation

$$E'(z) = 2h'(z)\,{}_3F_2\left(1,1,1;\frac{3}{2},2;\frac{1-h(z)}{2}\right) + \frac{1}{3}\big(1-h(z)\big)\,{}_3F_2\left(2,2,2;\frac{5}{2},3;\frac{1-h(z)}{2}\right)h'(z) - 1. \tag{4.5}$$

Proof. Clearly $h(z)$ has no singularities in the region $\{\, z : |1-h(z)| \le 2 \,\}$ and by definition $h(z)$ does not take values on the branch cut $(-\infty,-1)$. So from Lemma 2.4 we see that $E(z)$ is analytic in this region, which leads to the identity

$$2\frac{\mathrm{d}}{\mathrm{d}z}\cosh\sqrt{z+E(z)} = \big(1+E'(z)\big)\operatorname{sinhc}\sqrt{z+E(z)}. \tag{4.6}$$

The mixed error result (a) follows by taking derivatives in (4.3) and using (4.6). The result (b) follows by taking derivatives in (4.4) and using the identity

$$\frac{\mathrm{d}}{\mathrm{d}z}\,{}_3F_2\left(1,1,1;\frac{3}{2},2;z\right) = \frac{1}{3}\,{}_3F_2\left(2,2,2;\frac{5}{2},3;z\right). \qquad \square$$

A matrix function is completely determined by the values of the function and its derivatives on the spectrum of the matrix [18]. Since the functions $\cosh\sqrt{z}$ and $\operatorname{sinhc}\sqrt{z}$ are entire, the matrix functions $\cosh\sqrt{A}$ and $\operatorname{sinhc}\sqrt{A}$ are defined for all A. The approximation $h(A)$ is defined if the set of eigenvalues of A does not contain the singularities of $h(z)$. Let $\rho(A)$ denote the spectral radius of A.

THEOREM 4.3. *If A has eigenvalues in the principal domain of $\cosh\sqrt{\ }$ and $h(z)$ is any approximation to $\cosh\sqrt{z}$ such that $\rho(I-h(A)) \le 2$ then*

(a) $\cosh\sqrt{A} \approx h(A) = \cosh\sqrt{A+E(A)}$, where E is given by (4.4), and

(b) $\operatorname{sinhc}\sqrt{A} \approx 2h'(A) = (I+E'(A))\operatorname{sinhc}\sqrt{A+E(A)}$, where E' is given by (4.5).

Proof. (a) and (b) follow by applying Lemmas 4.1 and 4.2 on the spectrum of A. $\square$

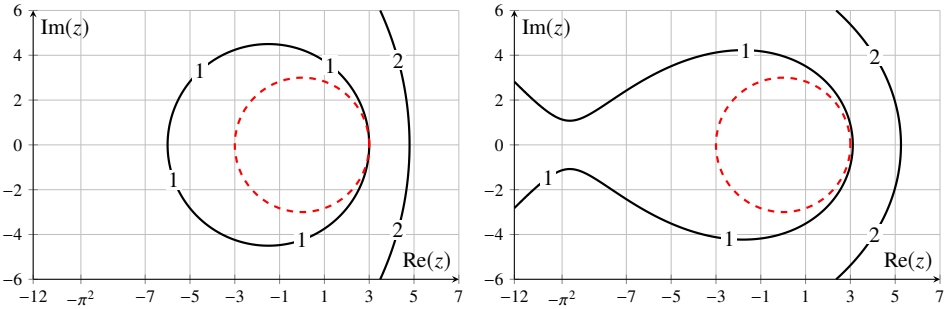
4.2. Padé approximants. Let $r_m(z) = p_m(z)/q_m(z)$ be the $[m/m]$ (diagonal) Padé approximant to $\cosh\sqrt{z}$. Thus p_m and q_m are polynomials of degree at most m, $q_m(0) = 1$, and $r_m(z) - \cosh\sqrt{z} = O(z^{2m+1})$. We are not aware of a proof of the existence of r_m for all m. Nevertheless, r_m exists for a particular m if the $m \times m$ Toeplitz matrix with (i,j) entry $1/\big(2(i-j+m)\big)!$ is nonsingular [24, p. 362]. Using Maple we have verified the existence of the first 100 diagonal Padé approximations.

The contours of $|1-r_m(z)|/2$ are shown in Figure 4.1 for $m \le 4$. Observe that in all the sub-figures the disc $|z| \le 3$ (dashed line) is contained inside the contour $|1-r_m(z)|/2 = 1$; we will prove that this is the case for all $m \le 20$.

LEMMA 4.4. *Let $p(z)$ and $q(z)$ be polynomials such that $q(0) = 1$ and the coefficients of both $p(z)$ and $q(-z)$ are positive real numbers. If R is a positive real number such that $q(-R) < 2$ then for $|z| \le R$,*

$$2 - q(-R) \le |q(z)| \le q(-R), \tag{4.7}$$

$$\frac{|p(z)|}{|q(z)|} \le \frac{p(R)}{2-q(-R)}. \tag{4.8}$$



(a) $m = 1$ (b) $m = 2$

(c) $m = 3$ (d) $m = 4$

Fig. 4.1: The contours of $|w-1|/2$, where $w = r_m(z)$ is the $[m/m]$ Padé approximant to $\cosh\sqrt{z}$ of degree $m \in \{1, 2, 3, 4\}$, along with the circle $|z| = 3$ (dashed line).

Proof. Given that $q(0) = 1$ and $q(-z)$ has real positive coefficients, the term $q(-|z|) - 1$ is positive and the inequality $|q(z) - 1| \le q(-|z|) - 1 \le q(-R) - 1$ holds in the region $|z| \le R$. In other words, $q(z)$ is contained in a circle with center $(1, 0)$ and radius $q(-R) - 1$. It follows that $\max\{0, 2 - q(-R)\} \le |q(z)| \le q(-R)$. If $q(-R) < 2$ and $p(z)$ has real positive coefficients, then the inequality in (4.8) is obtained by taking the ratio of the upper bound of $|p(z)|$ with the lower bound of $|q(z)|$. ∎

LEMMA 4.5. *For the* $[m/m]$ *Padé approximant* $r_m(z)$ *to* $\cosh\sqrt{z}$ *the condition* $|1 - r_m(z)| \le 2$ *is satisfied inside the disc* $|z| \le 3$ *for all* $m \le 20$. *For a matrix argument* A,

$$\rho(A) \le 3 \Rightarrow \rho\left(I - r_m(A)\right) < 2 \quad \textit{for } m \le 20. \tag{4.9}$$

Proof. We will first prove that $r_m(z)$ is analytic in the disc $|z| \le 3$ for $m \le 20$. Let $p_m(z)$ and $q_m(z)$ denote the numerator and denominator polynomials of r_m. Using Maple we have obtained symbolically the coefficients of $p_m(z)$ and $q_m(z)$ for $m \in \{1, 2, \ldots, 20\}$ and found that $p_m(z)$, $q_m(-z)$, and $p_m(z) - q_m(z)$ have positive real coefficients. Choosing $R = 3$, we find that the first element of the sequence $\{2 - q_m(-R)\}$ is $3/4$ and the next 19 elements are, to 4 significant digits,

$$\{.8613, .9079, .9313, .9453, .9546, .9612, .9661, .9699, .9730,$$
$$.9754, .9775, .9793, .9807, .9820, .9832, .9842, .9851, .9858, .9866\}.$$

Hence $2 - q_m(-R) \geq 3/4$ for $m \leq 20$, and it follows from the lower bound in (4.7) that $q_m(z)$ has no zeros in the disc $|z| \leq 3$ for $m \leq 20$. Therefore $r_m(z)$ is analytic in the disc $|z| \leq 3$ for $m \leq 20$.

Likewise, the first element of the sequence $\{[p_m(3) - q_m(3)]/[2 - q_m(-3)]\}$ is 2 and the next 19 elements are, to 5 significant digits,

$$\{.97443, .96533, .96182, .96017, .95928, .95874, .95840, .95816, .95799,$$
$$.95787, .95778, .95770, .95764, .95760, .95756, .95753, .95750, .95748, .95746\} \times 2.$$

Hence the first 20 elements of the sequence are less than or equal to 2. Substituting $p(z)$ with $p_m(z) - q_m(z)$ in Lemma 4.4, it follows from (4.8) that

$$|z| \leq 3 \Rightarrow |1 - r_m(z)| = \frac{|p_m(z) - q_m(z)|}{|q_m(z)|} \leq \frac{p_m(3) - q_m(3)}{2 - q_m(-3)} \leq 2 \quad \text{for } m \leq 20 \tag{4.10}$$

The result (4.9) follows by applying (4.10) to the spectrum of A. ∎

For $m \leq 20$ we can therefore replace the condition $|1 - r_m(z)| \leq 2$ with the condition $|z| \leq 3$ in Lemmas 4.1 and 4.2. Likewise, we can replace the condition $\rho(I - r_m(A)) \leq 2$ in Theorem 4.3 with the more readily verifiable condition $\rho(A) \leq 3$ for $m \leq 20$.

We make the following conjecture based on similar observations for $m > 20$.

CONJECTURE 4.6. *For all m, the $[m/m]$ Padé approximant $r_m(z)$ to $\cosh\sqrt{z}$ satisfies $|1 - r_m(z)| \leq 2$ in the disc $|z| \leq 3$.*

4.3. Error bounds for Padé approximants. The forward error $e_m(z)$ and backward error $E_m(z)$ of the $[m/m]$ Padé approximant $r_m(z)$ to $\cosh\sqrt{z}$ are defined, as in (4.1) and (4.2), by

$$e_m(z) = r_m(z) - \cosh\sqrt{z}, \quad E_m(z) = (\cosh\sqrt{\ })^{-1} r_m(z) - z. \tag{4.11}$$

Recall that $|\cosh\sqrt{z} - 1| < 2$ for $|z| \leq 3$ (Figure 2.2) and $|r_m(z) - 1| \leq 2$ for $|z| \leq 3$ and $m \leq 20$ (Lemma 4.5). Therefore from Lemma 4.1 we find that $E_m(z)$ is analytic for $|z| \leq 3$ and $m \leq 20$. From (4.11) and the fact that $\cosh\sqrt{\ }$ is entire, we obtain

$$\begin{aligned} e_m(z) &= \cosh\sqrt{z + E_m(z)} - \cosh\sqrt{z} \\ &= E_m(z)(\cosh\sqrt{\ })'z + \frac{1}{2!}E_m(z)^2(\cosh\sqrt{\ })''z + \cdots. \end{aligned}$$

Since $e_m(z)$ is $O(z^{2m+1})$, by the definition of r_m, it follows that $E_m(z)$ is $O(z^{2m+1})$. Thus

$$E_m(z) = z\sum_{k\geq 0} c_{m,k} z^{2m+k} = z\widehat{E}_m(z) \quad \text{for } |z| \leq 3,\ m \leq 20, \tag{4.12}$$

for some coefficients $c_{m,k}$, where $\widehat{E}_m(z)$ denotes the relative backward error. For a matrix argument A it follows from (4.12) that

$$E_m(A) = A\sum_{k\geq 0} c_{m,k} A^{2m+k} = A\widehat{E}_m(A), \quad \text{for } \rho(A) \leq 3,\ m \leq 20.$$

Using Theorem 3.1 we have

$$\|\widehat{E}_m(A)\| \leq \sum_{k\geq 0} |c_{m,k}| \alpha_m(A)^{2m+k}, \tag{4.13}$$

where α_m is defined in (3.2).

Taking derivatives in (4.12) and replacing z by A it follows that

$$E'_m(A) = \sum_{k\geq 0}(2m+k+1)c_{m,k}A^{2m+k}, \quad \text{for } \rho(A)\leq 3,\ m\leq 20,$$

and then Theorem 3.1 gives

$$\|E'_m(A)\| \leq \sum_{k\geq 0}(2m+1+k)|c_{m,k}|\alpha_m(A)^{2m+k}. \tag{4.14}$$

(Recall that $E'_m(z)$ occurs in the error expansion for $\operatorname{sinhc}\sqrt{A}$ in Theorem 4.3 (b).)

Using Maple we have obtained symbolically the first 600 terms in the power series of $E_m(z)$ for $m\leq 20$ and found that except for $m=2$ the coefficients of the series have alternating signs. For $m=2$, the coefficients have a structured pattern of alternating signs

$$\{-1,1,\underbrace{-1,1,\ldots,-1,1}_{28\text{ terms}},\underbrace{1,-1,\ldots,1,-1}_{28\text{ terms}},\underbrace{-1,1,\ldots,-1,1}_{28\text{ terms}},\ldots\}.$$

Note that the first 30 coefficients have alternating signs. The first term is the coefficient of z^5 and it is of the order of 10^{-7}. Its product with 3^5 is of the order of 10^{-4}. The 30th term is the coefficient of z^{34} and it is of the order of 10^{-37}. Its product with 3^{34} is of the order of 10^{-21}. Effectively, then, in the context of double-precision arithmetic with $|z|\leq 3$, we can regard $E_m(z)$, and also $\widehat{E}_m(z)$, as having power series with alternating coefficients. Then $\sum_{k\geq 0}|c_{m,k}|\alpha_m(A)^{2m+k} = |\widehat{E}_m(-\alpha_m(A))|$ and the bound for $\|\widehat{E}_m(A)\|$ in (4.13) simplifies to

$$\|\widehat{E}_m(A)\| \leq |\widehat{E}_m(-\alpha_m(A))| \leq |\widehat{E}_m(-3)| \quad \text{for } m\leq 20,\ \alpha_m(A)\leq 3$$

Additionally, $\sum_{k\geq 0}(2m+1+k)|c_{m,k}|\alpha_m(A)^{2m+k} = |E'_m(-\alpha_m(A))|$ and the bound for $\|E'_m(A)\|$ in (4.13) simplifies to

$$\|E'_m(A)\| \leq |E'_m(-\alpha_m(A))| \leq |E'_m(-3)| \quad \text{for } m\leq 20,\ \alpha_m(A)\leq 3.$$

The IEEE double precision unit roundoff u is 2^{-53}. Table 4.1 contains the values of $|\widehat{E}_m(-3)|$ and the radius

$$\theta_m = \max\{\, x : |\widehat{E}_m(-x)| = u \,\}. \tag{4.15}$$

Table 4.2 contains the values of $|E'_m(-3)|$ and the radius

$$\theta'_m = \max\{\, x : |E'_m(-x)| = u \,\}. \tag{4.16}$$

In these tables the values of θ_m, θ'_m, $|\widehat{E}_m(-3)|$ and $|E'_m(-3)|$ are computed using variable precision arithmetic with 100 significant digits.

Observe that for $m\geq 6$ we have both $|\widehat{E}_m(-3)| < u$ and $|E'_m(-3)| < u$. Additionally, for $m<6$ the values of θ'_m in Table 4.2 are smaller than the corresponding values in Table 4.1. So choosing θ'_m from Table 4.2 we get

$$\|\widehat{E}_m(A)\| \leq u, \quad \|E'_m(A)\| \leq u \qquad \text{for } m\leq 20,\ \alpha_m(A)\leq \min\left(3,\theta'_m\right).$$

Table 4.1: Relative backward error bound $|\widehat{E}_m(-3)|$ and values of θ_m in (4.15) for the $[m/m]$ Padé approximants to $\cosh\sqrt{z}$ for IEEE double precision arithmetic.

m	$\lvert\widehat{E}_m(-3)\rvert$	θ_m	m	$\lvert\widehat{E}_m(-3)\rvert$	θ_m
1	4.68×10^{-2}	1.63×10^{-7}	6	2.24×10^{-21}	>3
2	8.85×10^{-5}	3.46×10^{-3}	7	2.29×10^{-26}	>3
3	2.94×10^{-8}	1.26×10^{-1}	8	1.37×10^{-31}	>3
4	2.93×10^{-12}	8.75×10^{-1}	9	5.08×10^{-37}	>3
5	1.17×10^{-16}	2.98	10	1.23×10^{-42}	>3

Table 4.2: Mixed error bound $|E'_m(-3)|$ and values of θ'_m in (4.16) for approximations to $\operatorname{sinhc}\sqrt{z}$ for IEEE double precision arithmetic.

m	$\lvert E'_m(-3)\rvert$	θ'_m	m	$\lvert E'_m(-3)\rvert$	θ'_m
1	1.58×10^{-1}	9.42×10^{-8}	6	3.04×10^{-20}	>3
2	4.80×10^{-4}	2.31×10^{-3}	7	3.57×10^{-25}	>3
3	2.20×10^{-7}	9.14×10^{-2}	8	2.40×10^{-30}	>3
4	2.79×10^{-11}	6.66×10^{-1}	9	9.95×10^{-36}	>3
5	1.36×10^{-15}	2.36	10	2.66×10^{-41}	>3

The double-angle formulas

$$\cosh 2\sqrt{A} = 2\bigl(\cosh\sqrt{A}\bigr)^2 - I, \quad \operatorname{sinhc} 2\sqrt{A} = \operatorname{sinhc}\sqrt{A}\cosh\sqrt{A}, \tag{4.17}$$

hold for all A. When $\alpha_m(A) > \min(3,\theta'_m)$, we scale down A by a factor 4^s such that $\alpha_m(4^{-s}A) \le \min\bigl(3,\theta'_m\bigr)$, compute approximations to $\cosh\sqrt{4^{-s}A}$ and $\operatorname{sinhc}\sqrt{4^{-s}A}$ and then scale up using the double-angle recurrence

$$\begin{aligned} &C_0(A) = r_m(4^{-s}A), && S_0(A) = 2r'_m(4^{-s}A),\\ &C_{i+1}(A) = 2C_i(A)^2 - I, && S_{i+1}(A) = S_i(A)C_i(A), \quad i = 0,\dots,s-1,\\ &\cosh\sqrt{A} \approx C_s(A), && \operatorname{sinhc}\sqrt{A} \approx S_s(A). \end{aligned} \tag{4.18}$$

If the scaling up phase is done in exact arithmetic, then from Theorem 4.3 we find

$$C_s(A) = \cosh\sqrt{A\bigl(I + \widehat{E}_m(4^{-s}A)\bigr)}, \tag{4.19a}$$

$$S_s(A) = \bigl(I + E'_m(4^{-s}A)\bigr)\operatorname{sinhc}\sqrt{A\bigl(I + \widehat{E}_m(4^{-s}A)\bigr)}, \tag{4.19b}$$

with

$$\|\widehat{E}_m(4^{-s}A)\| \le u, \quad \|E'_m(4^{-s}A)\| \le u.$$

Thus in exact arithmetic the approximation $C_s(A)$ to $\cosh\sqrt{A}$ is backward stable and the approximation $S_s(A)$ to $\operatorname{sinhc}\sqrt{A}$ is mixed forward–backward stable.

5. Algorithm for computing the wave kernels. The matrices $r_m(A)$ and $r'_m(A)$ are obtained by solving $q_m(A)r_m(A) = p_m(A)$ and $q_m^2(A)r'_m(A) = w_m(A)$, where

$$w_m(A) \coloneqq p'_m(A)q_m(A) - p_m(A)q'_m(A).$$

Table 5.1: Parameter σ that minimizes the number of matrix multiplications μ_t for each degree m in the Paterson–Stockmeyer algorithm to evaluate $p_m(A)$, $q_m(A)$ and $w_m(A)$, along with $\mu_* = \mu_t - (\sigma - 1)$.

m	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
σ	1	2	3	4	5	6	4	4	5	5	6	6	7	7	8	8	9	9	10	10
μ_t	0	1	3	4	5	6	7	8	9	9	10	10	11	11	12	12	13	13	14	14
μ_*	0	0	1	1	1	1	4	5	5	5	5	5	5	5	5	5	5	5	5	5

Using Maple we have obtained symbolically the coefficients of the polynomials p_m and q_m, evaluated them using variable precision arithmetic, and stored them as IEEE double precision floating point numbers. To reduce cost and to avoid bringing any finite precision cancellation errors to prominence, we also evaluated symbolically and stored numerically the coefficients of the degree $2m-2$ polynomial $w_m(A)$. All these are off-line calculations, done in advance.

We will use the Paterson–Stockmeyer (PS) algorithm [18, p. 73], [27] to compute the polynomials $p_m(A)$, $q_m(A)$, and $w_m(A)$. Let $\sigma \leq m$ be a positive integer and suppose we compute and store $A^2, A^3, \ldots, A^\sigma$, which requires $\mu_\sigma = \sigma - 1$ matrix multiplications. The total number of matrix multiplications in the PS algorithm is then

$$\mu_t = (\sigma - 1) + 2\left\lfloor \frac{m}{\sigma} \right\rfloor - 2(\sigma \mid m) + \left\lfloor \frac{2m-2}{\sigma} \right\rfloor - \big(\sigma \mid (2m-2)\big),$$

where $\lfloor m/\sigma \rfloor$ is the largest integer less than or equal to m/σ and $\sigma \mid m$ is either 1 (if σ divides m) or 0 (otherwise).

For each m, the σ that mininimizes the number of matrix multiplications in the PS algorithm to compute $p_m(A)$, $q_m(A)$ and $w_m(A)$ is shown in Table 5.1. For $m \leq 20$, this cost jumps between degree m and $m+1$ only for $m \in \{1\text{–}8, 10, 12, 14, 16, 18, 20\}$. Hence we will consider only these m in our algorithm. The matrices whose columns are the co-primes required to compute $\alpha_m(A)$, for these m are shown in Table 5.2.

The definition of $\alpha_m(A)$ involves norms of various powers of A defined in (3.2) We will compute only those powers needed for the evaluation of the polynomials and will use those powers to estimate the norms of the others. We use the 1-norm and estimate norms using the block algorithm of Higham and Tisseur [22], which estimates $\|B\|_1$ using a few matrix–vector products with B and B^T. We denote a call to the estimator by `normest1`$(A^{n_1}, A^{n_2}, \ldots, A^{n_k})$, which means that the algorithm estimates $\|A^{n_1+n_2+\cdots+n_k}\|_1$ by forming matrix–vector products $A^{n_1+n_2+\cdots+n_k}x$ as $A^{n_1}(A^{n_2}(\ldots(A^{n_k}x)))$ (and similarly for the transpose).

In using `normest1` we want to do as few matrix–vector products as possible. We will use the powers of A stored in the PS algorithm to this end. For instance, consider $m = 1$, for which only A is stored. To compute $\alpha_1(A)$ we estimate $\|A^2\|_1^{1/2}$ and $\|A^3\|_1^{1/3}$ (see Table 5.2) by calling `normest1`(A, A) and `normest1`(A, A, A), respectively. If we proceed to $m = 2$, we compute and store A^2 (see Table 5.1). To compute $\alpha_2(A)$ we compute $\|A^2\|_1^{1/2}$ directly and estimate $\|A^5\|_1^{1/5}$ with the call `normest1`(A, A^2, A^2). Note that it makes no difference to the quality of the estimate how the matrix–vector products are factored; our aim is purely to minimize the cost.

We note that Higham and Smith [21, p. 20] analysed the stability with respect to

Table 5.2: Matrices whose columns are the co-primes required to compute $\alpha_m(A)$.

m	Co-primes	m	Co-primes
1	$\begin{bmatrix}2\\3\end{bmatrix}$	8	$\begin{bmatrix}2&3&3&4\\17&7&8&5\end{bmatrix}$
2	$\begin{bmatrix}2\\5\end{bmatrix}$	10	$\begin{bmatrix}2&3&3&4&4&5\\21&10&11&5&7&6\end{bmatrix}$
3	$\begin{bmatrix}2&3\\7&4\end{bmatrix}$	12	$\begin{bmatrix}2&3&3&4&4&5&5\\25&11&13&7&9&6&7\end{bmatrix}$
4	$\begin{bmatrix}2&3&3\\9&4&5\end{bmatrix}$	14	$\begin{bmatrix}2&3&3&4&4&5&5&5\\29&13&14&7&9&6&7&8\end{bmatrix}$
5	$\begin{bmatrix}2&3&3\\11&4&5\end{bmatrix}$	16	$\begin{bmatrix}2&3&3&4&4&5&5&5&5&6\\33&16&17&9&11&6&7&8&9&7\end{bmatrix}$
6	$\begin{bmatrix}2&3&3&4\\13&5&7&5\end{bmatrix}$	18	$\begin{bmatrix}2&3&3&4&4&5&5&5&5&6\\37&17&19&11&13&6&7&8&9&7\end{bmatrix}$
7	$\begin{bmatrix}2&3&3&4\\15&7&8&5\end{bmatrix}$	20	$\begin{bmatrix}2&3&3&4&4&5&5&5&5&6\\41&19&20&11&13&7&8&9&11&7\end{bmatrix}$

Table 5.3: Matrices whose columns are the co-primes required to compute $\alpha_m(A)$ sequentially, that is, for each m we update $\alpha_m(A) \leftarrow \min(\alpha_*(A), \alpha_m(A))$ where $\alpha_*(A)$ was computed in the previous step.

m	Co-primes	m	Co-primes	m	Co-primes
1	$\begin{bmatrix}2\\3\end{bmatrix}$	6	$\begin{bmatrix}2&3&4\\13&7&5\end{bmatrix}$	14	$\begin{bmatrix}2&3&5\\29&14&8\end{bmatrix}$
2	$\begin{bmatrix}2\\5\end{bmatrix}$	7	$\begin{bmatrix}2&3\\15&8\end{bmatrix}$	16	$\begin{bmatrix}2&3&3&4&5&6\\33&16&17&11&9&7\end{bmatrix}$
3	$\begin{bmatrix}2&3\\7&4\end{bmatrix}$	8	$\begin{bmatrix}2\\17\end{bmatrix}$	18	$\begin{bmatrix}2&3&4\\37&19&13\end{bmatrix}$
4	$\begin{bmatrix}2&3\\9&5\end{bmatrix}$	10	$\begin{bmatrix}2&3&3&4&5\\21&10&11&7&6\end{bmatrix}$	20	$\begin{bmatrix}2&3&5\\41&20&11\end{bmatrix}$
5	$\begin{bmatrix}2\\11\end{bmatrix}$	12	$\begin{bmatrix}2&3&4&5\\25&13&9&7\end{bmatrix}$		

rounding errors of the double angle recurrence (4.18) for their algorithm to compute the matrix cosine. They found that the relative forward error bound is a sum of terms comprising two factors. The first factor is a power up to the sth of an $O(1)$ scalar independent of A. These factors are innocuous if s is small and are likely to be pessimistic, otherwise. The second factor is a product of terms that depend on the norms of the intermediate $C_i(A)$, and is difficult to bound a priori. The number of such terms grows with s. So to mitigate the potential deterioration of accuracy in

the recurrence, our priority is to minimize s in the scaling stage. The sharper error bounds given in (4.13) and (4.14) and choosing the pair with the larger m and smaller s when there is a choice contribute to this objective.

Recall that $\{A : \rho(A) \le 3\}$ is the set of admissible A for which the error terms $\|\widehat{E}_m(A)\|$ and $\|E'_m(A)\|$ are well-defined. As $\rho(A) \le \alpha_m(A)$, we note that $\mathcal{A}_{m,u} := \{A : \alpha_m(A) \le \min\{\theta'_m, 3\}\}$ is a subset of the admissible set and in this subset the error terms $\|\widehat{E}_m(A)\|$ and $\|E'_m(A)\|$ are bounded by the unit roundoff. Further, as $\alpha_m(A)$ is nonincreasing with m by Theorem 3.1, the subset $\mathcal{A}_{m,u}$ will grow with m if θ'_m does. Observe in Tables 4.1 and 4.2 that for $m \le 5$, θ'_m increases with m and $\theta'_m < 3$. Hence for all $m \le 5$ the subset $\mathcal{A}_{m,u}$ will certainly grow larger with m. Therefore, to avoid scaling in our algorithm we will compute $\alpha_m(A)$ sequentially and check if $A \in \mathcal{A}_{m,u}$.

We note that the co-primes listed in Table 5.2 are appropriate to compute each $\alpha_m(A)$ independently. Suppose we have computed and stored $\alpha_4(A)$ and $\|A^k\|_1^{1/k}$ for $k \in \{2, 3, 4, 5, 9\}$. Observe in Table 5.2 that for $m = 5$ we can reuse the known $\|A^k\|_1^{1/k}$ and we only need to estimate $\|A^{11}\|_1^{1/11}$. By definition $\alpha_5(A) \le \alpha_4(A)$ and observe that both $\max(\|A^3\|_1^{1/3}, \|A^4\|_1^{1/4})$ and $\max(\|A^3\|_1^{1/3}, \|A^5\|_1^{1/5})$ were included while computing $\alpha_4(A)$. So to compute $\alpha_5(A)$ we first assign $\alpha_5(A) \leftarrow \max(\|A^2\|_1^{1/2}, \|A^{11}\|_1^{1/11})$ and then update $\alpha_5(A) \leftarrow \min(\alpha_4(A), \alpha_5(A))$. Following this line of reasoning, the matrices whose columns are the co-primes required to compute $\alpha_m(A)$ sequentially are shown in Table 5.3.

Taking all these aspects into account we now present our algorithm to compute the scaling s and the order m.

ALGORITHM 5.1 (Parameter selection). *Given $A \in \mathbb{C}^{n\times n}$ this algorithm computes the order m and the scaling s such that the relative errors $\|\widehat{E}_m(4^{-s}A)\|_1$ and $\|E'_m(4^{-s}A)\|_1$ in (4.19) are bounded by the IEEE double precision unit roundoff. It uses the quantities θ'_m tabulated in Table 4.2 and the co-primes listed in Table 5.3.*

1 Compute and store $\beta_2 = \texttt{normest1}(A, A)^{1/2}$, $\beta_3 = \texttt{normest1}(A, A, A)^{1/3}$ and $\alpha_1(A) = \max(\beta_2, \beta_3)$.
2 if $\alpha_1(A) \le \theta'_1$, then $m = 1$, $s = 0$, quit, end.
3 Compute and store A^2, $\beta_2 = \|A^2\|_1^{1/2}$, $\beta_5 = \texttt{normest1}(A, A^2, A^2)^{1/5}$ and $\alpha_2(A) = \max(\beta_2, \beta_5)$.
4 if $\alpha_2(A) \le \theta'_2$, then $m = 2$, $s = 0$, quit, end.
5 Compute and store A^3, $\beta_3 = \|A^3\|_1^{1/3}$, $\beta_4 = \texttt{normest1}(A, A^3)^{1/4}$, $\beta_7 = \texttt{normest1}(A, A^3, A^3)^{1/7}$ and $\alpha_3(A) = \min\max\left[\begin{smallmatrix}\beta_2 & \beta_3\\ \beta_7 & \beta_4\end{smallmatrix}\right]$.
6 if $\alpha_3(A) \le \theta'_3$, then $m = 3$, $s = 0$, quit, end.
7 Compute and store A^4, $\beta_4 = \|A^4\|_1^{1/4}$, $\beta_9 = \texttt{normest1}(A, A^4, A^4)^{1/9}$ and $\alpha_4(A) = \min\max\left[\begin{smallmatrix}\beta_2 & \beta_3\\ \beta_9 & \beta_5\end{smallmatrix}\right]$.
8 Update $\alpha_4(A) \leftarrow \min(\alpha_3(A), \alpha_4(A))$.
9 if $\alpha_4(A) \le \theta'_4$, then $m = 4$, $s = 0$, quit, end.
10 Compute and store $\beta_{11} = \texttt{normest1}(A^3, A^4, A^4)^{1/11}$ and $\alpha_5(A) = \max(\beta_2, \beta_{11})$.
11 Update $\alpha_5(A) \leftarrow \min(\alpha_4(A), \alpha_5(A))$.
12 if $\alpha_5(A) \le \theta'_5$, then $m = 5$, $s = 0$, compute and store A^5, quit, end.
13 Compute and store $\beta_{13} = \texttt{normest1}(A, A^4, A^4, A^4)^{1/13}$ and $\alpha_6(A) = \min\max\left[\begin{smallmatrix}\beta_2 & \beta_3 & \beta_4\\ \beta_{13} & \beta_7 & \beta_5\end{smallmatrix}\right]$.

14 Update $\alpha_6(A) \leftarrow \min(\alpha_5(A), \alpha_6(A))$.
15 if $\alpha_6(A) \leq 3$, $m = 6$, then $s = 0$, compute and store A^5, A^6, quit, end.
16 Compute and store $\beta_8 = \texttt{normest1}(A^4, A^4)^{1/8}$,
$\beta_{15} = \texttt{normest1}(A^3, A^4, A^4, A^4)^{1/15}$ and $\alpha_7(A) = \min \max \left[\begin{smallmatrix} \beta_2 & \beta_3 \\ \beta_{15} & \beta_8 \end{smallmatrix} \right]$.
17 Update $\alpha_7(A) \leftarrow \min(\alpha_6(A), \alpha_7(A))$.
18 if $\alpha_7(A) \leq 3$, then $m = 7$, $s = 0$, quit, end.
19 Compute and store $\beta_{17} = \texttt{normest1}(A, A^4, A^4, A^4, A^4)^{1/17}$
and $\alpha_8(A) = \max(\beta_2, \beta_{17})$.
20 Update $\alpha_8(A) \leftarrow \min(\alpha_7(A), \alpha_8(A))$.
21 if $\alpha_8(A) \leq 3$, $m = 8$, then $s = 0$, quit, end.
22 Compute and store A^5, $\beta_5 = \|A^5\|_1^{1/5}$, $\beta_{10} = \texttt{normest1}(A^5, A^5)^{1/10}$
$\beta_{21} = \texttt{normest1}(A, A^5, A^5, A^5, A^5)^{1/21}$ and
$\alpha_{10}(A) = \min \max \left[\begin{smallmatrix} \beta_2 & \beta_3 & \beta_3 & \beta_4 & \beta_5 \\ \beta_{21} & \beta_{10} & \beta_{11} & \beta_7 & \beta_6 \end{smallmatrix} \right]$.
23 Update $\alpha_{10}(A) \leftarrow \min(\alpha_8(A), \alpha_{10}(A))$.
24 if $\alpha_{10}(A) \leq 3$, then $m = 10$, $s = 0$, quit, end.
25 Compute and store A^6, $\beta_6 = \|A^6\|_1^{1/6}$, $\beta_{25} = \texttt{normest1}(A, A^6, A^6, A^6, A^6)^{1/25}$
and $\alpha_{12}(A) = \min \max \left[\begin{smallmatrix} \beta_2 & \beta_3 & \beta_4 & \beta_5 \\ \beta_{25} & \beta_{13} & \beta_9 & \beta_7 \end{smallmatrix} \right]$.
26 Update $\alpha_{12}(A) \leftarrow \min(\alpha_{10}(A), \alpha_{12}(A))$.
27 if $\alpha_{12}(A) \leq 3$, $m = 12$, then $s = 0$, quit, end.
28 Compute and store A^7, $\beta_7 = \|A^7\|_1^{1/7}$, $\beta_{14} = \texttt{normest1}(A^7, A^7)^{1/14}$
$\beta_{29} = \texttt{normest1}(A, A^7, A^7, A^7, A^7)^{1/29}$ and $\alpha_{14}(A) = \min \max \left[\begin{smallmatrix} \beta_2 & \beta_3 & \beta_5 \\ \beta_{29} & \beta_{14} & \beta_8 \end{smallmatrix} \right]$.
29 Update $\alpha_{14}(A) \leftarrow \min(\alpha_{12}(A), \alpha_{14}(A))$.
30 if $\alpha_{14}(A) \leq 3$, then $m = 14$, $s = 0$, quit, end.
31 Compute and store A^8, $\beta_8 = \|A^8\|_1^{1/8}$, $\beta_{16} = \texttt{normest1}(A^8, A^8)^{1/16}$
$\beta_{33} = \texttt{normest1}(A, A^8, A^8, A^8, A^8)^{1/33}$ and
$\alpha_{16}(A) = \min \max \left[\begin{smallmatrix} \beta_2 & \beta_3 & \beta_3 & \beta_4 & \beta_5 & \beta_6 \\ \beta_{33} & \beta_{16} & \beta_{17} & \beta_{11} & \beta_9 & \beta_7 \end{smallmatrix} \right]$.
32 Update $\alpha_{16}(A) \leftarrow \min(\alpha_{14}(A), \alpha_{16}(A))$.
33 if $\alpha_{16}(A) \leq 3$, then $m = 16$, $s = 0$, quit, end.
34 Compute and store A^9, $\beta_9 = \|A^9\|_1^{1/9}$, $\beta_{19} = \texttt{normest1}(A, A^9, A^9)^{1/19}$,
$\beta_{37} = \texttt{normest1}(A, A^9, A^9, A^9, A^9)^{1/37}$ and $\alpha_{18}(A) = \min \max \left[\begin{smallmatrix} \beta_2 & \beta_3 & \beta_4 \\ \beta_{37} & \beta_{19} & \beta_{13} \end{smallmatrix} \right]$.
35 Update $\alpha_{18}(A) \leftarrow \min(\alpha_{16}(A), \alpha_{18}(A))$.
36 if $\alpha_{18}(A) \leq 3$, then $m = 18$, $s = 0$, quit, end.
37 Compute and store A^{10}, $\beta_{10} = \|A^{10}\|_1^{1/10}$, $\beta_{20} = \texttt{normest1}(A^{10}, A^{10})^{1/20}$,
$\beta_{41} = \texttt{normest1}(A, A^{10}, A^{10}, A^{10}, A^{10})^{1/41}$ and
$\alpha_{20}(A) = \min \max \left[\begin{smallmatrix} \beta_2 & \beta_3 & \beta_5 \\ \beta_{41} & \beta_{20} & \beta_{11} \end{smallmatrix} \right]$.
38 Update $\alpha_{20}(A) \leftarrow \min(\alpha_{18}(A), \alpha_{20}(A))$.
39 if $\alpha_{20}(A) \leq 3$, then $m = 20$, $s = 0$, quit, end.
40 Compute $s_k = \texttt{ceil}\,(\log_4[\alpha_k(A)/3])$ for $k = [6, 7, 20]$.
41 $s = s_{20}$
42 $m =$ smallest $k \in [6, 7, 20]$ such that $s_k = s_{20}$.

Note that if we arrive at line 40 of Algorithm 5.1, then we have already incurred the cost of computing and storing $A^2, A^3, \ldots, A^{10}$. At this stage of the algorithm scaling is necessary. To minimize the scaling we choose $s = s_{20}$. It might be the case that the scaled matrix will belong to several $\mathcal{A}_{m,u}$. We choose the m that minimizes the multiplication cost μ_*. Observe in Table 5.1 that for $m \leq 20$ the matrix

multiplication count μ_* jumps between degree m and $m+1$ only for $m \in \{2, 6, 7, 20\}$. Using the θ'_m in Table 4.2 we find that $\texttt{ceil}\,(\log_4(\theta'_6/\theta'_2)) = 6$, which means that once we scale down A to enter the set $\mathcal{A}_{6,u}$ we need to scale down further by a factor 4^6 to enter the set $\mathcal{A}_{2,u}$. Hence we exclude the choice $m = 2$ in the line 40 of Algorithm 5.1.

We now present our complete algorithm for computing the wave-kernel matrix functions.

ALGORITHM 5.2 (wave-kernel matrix functions). *Given $A \in \mathbb{C}^{n\times n}$ this algorithm computes the wave-kernel functions $C = \cosh\sqrt{A}$ and $S = \operatorname{sinhc}\sqrt{A}$.*

1 Obtain m and s from Algorithm 5.1 applied to A.
2 Choose σ for this m from Table 5.1.
3 $A \leftarrow 4^{-s}A$ and $A^k \leftarrow 4^{-sk}A^k$ for $k = 1, 2, \dots, \sigma$.
4 Compute the matrix polynomials $p_m(A)$, $q_m(A)$ and $w_m(A)$ using the Paterson–Stockmeyer algorithm and the matrix powers computed on the previous steps.
5 Compute an LU factorization with partial pivoting $LU = q_m(A)$.
6 Compute $C = U^{-1}L^{-1}p_m(A)$ and $S = 2U^{-1}L^{-1}U^{-1}L^{-1}w_m(A)$ by substitution using the LU factors.
7 for $m = 1\colon s$
8 $\quad S \leftarrow SC$
9 $\quad C \leftarrow 2C^2 - I$
10 end

Cost. The highest order term of the total cost of Algorithm 5.2 is

$$\left(\frac{20}{3} + 4s + 2\min\left(m, 4 + \left\lceil\frac{m}{2}\right\rceil\right) + 4(s \neq 0)(m \neq 20)\left\lceil\frac{m}{3}\right\rceil\right) n^3 \text{ flops},$$

for $m \geq 3$. The first term is the cost of the LU decomposition and the substitutions. The second term is the cost of undoing the effect of scaling via recurrences. The third term is the cost of parameter selection and computation of the Padé approximants using the Paterson–Stockmeyer algorithm in the absence of scaling. The fourth term is the additional cost for having computed $A^2, A^3, \dots, A^{10}$ and if in line 42 of Algorithm 5.1 we obtain either $m = 6$ or $m = 7$.

MATLAB functions to compute and test the wave-kernel matrix functions (`wkm.m` and `test_wkm.m`, respectively) are available in the GitHub repository https://github.com/nadukandi/wkm. The raw data used to generate Figure 6.1 is also available in this repository.

6. Numerical examples. All our experiments are performed in MATLAB R2017b, for which the unit roundoff is $u = 2^{-53} \approx 1.11 \times 10^{-16}$. In the first example we consider a matrix whose wave kernels have an explicit representation. For the rest of the test matrices we use Davies's approximate diagonalization method [9] to compute accurate values of $\cosh\sqrt{A}$ and $\operatorname{sinhc}\sqrt{A}$, employing the VPA arithmetic of the Symbolic Math Toolbox at 250 digit precision. In this method we add a random perturbation of norm 10^{-125} to A, diagonalize the result, then apply the wave-kernel functions to the eigenvalues; the perturbation ensures the eigenvalues are distinct so that the diagonalization is always possible.

Recall that Algorithm 5.2 is backward stable in exact arithmetic and we expect the relative (forward) error to be bounded by a modest multiple of the condition number $\operatorname{cond}(f, A)$ times the unit roundoff, where f is the function in question. This condition number is given in [18, chap. 3] and we estimate it using the code `funm_condest1` from

the Matrix Function Toolbox [17].

The test suite consists of 87 mainly 15×15 test matrices adapted from the Matrix Computation Toolbox [17], the MATLAB `gallery` function, and the matrix function literature. The relative forward errors and the error estimates are shown in Figure 6.1, ordered by decreasing $\mathrm{cond}(f, A)$ for the test matrices. Observe that the relative errors are bounded by the estimated condition number times the unit roundoff. Thus our algorithm behaves in a forward stable manner in floating point arithmetic. We compare our algorithm with the MATLAB function `funm`, which uses the Schur–Parlett algorithm of Davies and Higham [10], [18, chap. 9]. Here we supply `funm` with derivatives computed from (1.5). We see that `funm` is generally forward stable but behaves in an unstable manner on several matrices. Algorithm 5.2 clearly has superior stability to `funm`.

In the next experiment we multiply each matrix in the test suite by a factor 60 and compute the wave kernels. The rationale of this experiment is to ensure that some scaling will occur in Algorithm 5.2 and to study the algorithm's robustness to changes in the approximation order and scaling. The relative forward errors and the error estimates are shown in Figure 6.1c and 6.1d. Additionally, the change in the forward errors for Algorithm 5.2 due to the scaling $A \leftarrow 60A$ are shown shown as vertical bars. The results are plotted in the same order: the condition number times the unit roundoff line is no longer monotonic and in a few cases overflow was encountered (these errors are not plotted). The general trend is that the errors increase but this increase is not uniform. Additionally, for some test matrices the errors decrease which is why we chose to illustrate these nonintutive error changes using vertical bars. Nevertheless, we observe that the relative errors for Algorithm 5.2 are again bounded by a modest multiple of the condition number times the unit roundoff.

7. Conclusions. We have developed the first algorithm for computing the wave kernel $\cosh\sqrt{A}$ that is backward stable in exact arithmetic and is suitable for any square matrix A. The algorithm also computes the wave kernel $\mathrm{sinhc}\sqrt{A}$, for which it is mixed forward–backward stable in exact arithmetic. Numerical experiments show that the algorithm behaves in a forward stable manner in floating-point arithmetic, whereas the Schur-Parlett algorithm applied to these functions displays some instability. Several trigonometric matrix functions can be computed from this algorithm by an appropriate change of variables: for instance, $\cos A = \cosh\sqrt{-A^2}$, $\cos\sqrt{A} = \cosh\sqrt{-A}$, and $\mathrm{sinc}\, A = \mathrm{sinhc}\sqrt{-A^2}$.

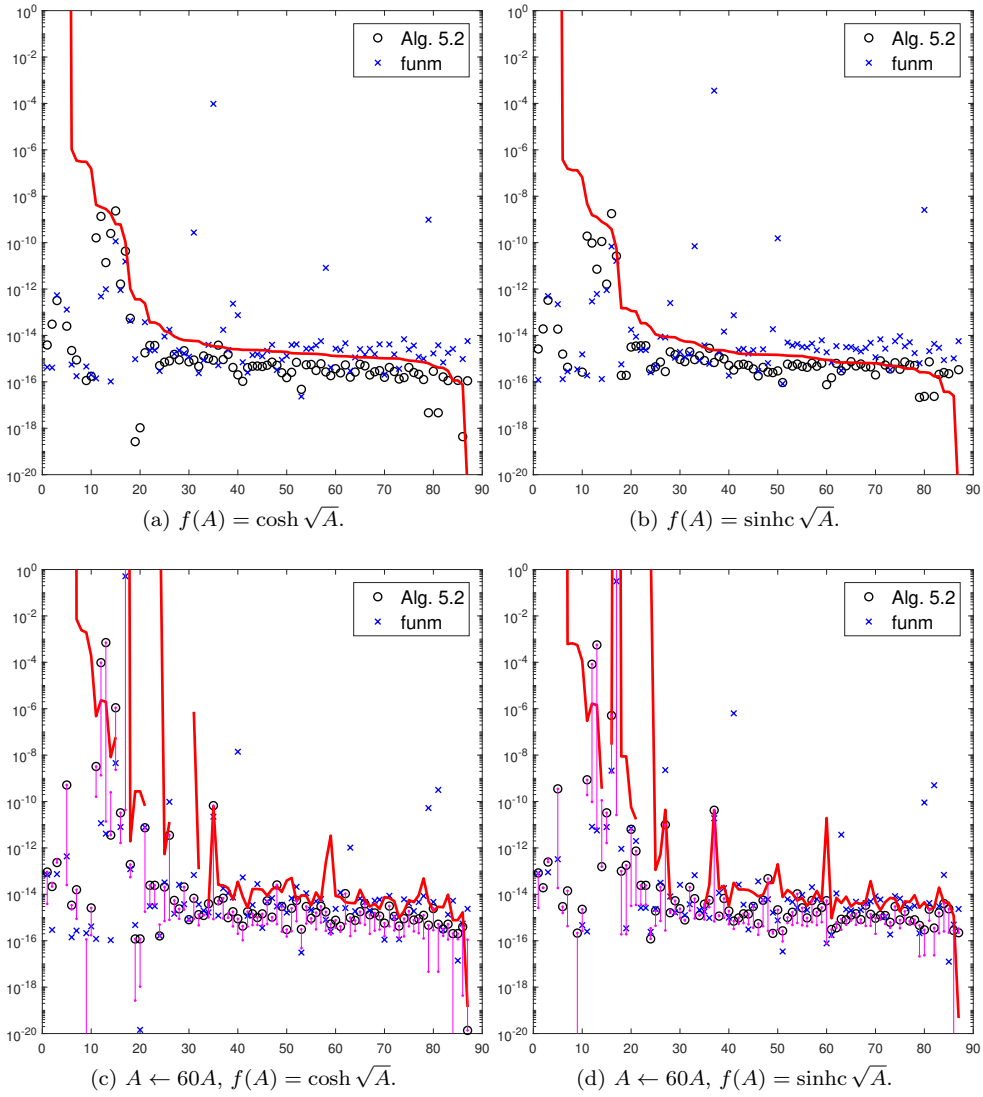
The improved bound for $\|A^k\|$ in section 3 merits investigation for use in existing matrix function algorithms based on Padé approximation, such as those in [3], [5], [6], [7], [20], though it will require reworking of the underlying logic for the choice of the amount of scaling and the Padé degree.

Following the lead of Strang and MacNamara [31, p. 527] we plan to pursue the role of wave-kernel matrix functions to consider *waves on graphs* and the application to characterization and classification of directed graphs.

Appendix A. Background.

A.1. Lagrange inversion theorem. If a function $f(z)$ is analytic at a point $z = a$ in its domain and the derivative $f'(a) \neq 0$, then the Lagrange inversion theorem [26, eq. (1.10.13)] allows us to express the inverse function of $f(z)$ as a power series. The theorem states that if $w = f(z)$ then

$$z = f^{-1}(w) = a + \sum_{n=1}^{\infty} \frac{[w - f(a)]^n}{n!} \lim_{x \to a} \left[\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}} \left(\frac{x - a}{f(x) - f(a)} \right)^n \right] \tag{A.1}$$



(a) $f(A) = \cosh \sqrt{A}$. (b) $f(A) = \operatorname{sinhc} \sqrt{A}$.

(c) $A \leftarrow 60A$, $f(A) = \cosh \sqrt{A}$. (d) $A \leftarrow 60A$, $f(A) = \operatorname{sinhc} \sqrt{A}$.

Fig. 6.1: Relative forward errors of Algorithm 5.2 and the MATLAB function `funm` for the computed wave kernels $\cosh \sqrt{A}$ and $\operatorname{sinhc} \sqrt{A}$. The solid red line is the condition number estimate of the matrix functions times the unit roundoff. The results in (a) and (b) are ordered by decreasing $\operatorname{cond}(f, A)$. These orderings are retained in (c) and (d), respectively, in which vertical bars denote the change in the error of Algorithm 5.2 from (a) and (b).

The theorem also guarantees that the series in (A.1) has a nonzero radius of convergence, that is, $f^{-1}(w)$ is an analytic function of w in a neighbourhood of $w = f(a)$.

A.2. Generalized hypergeometric function. The generalized hypergeometric function is defined by the power series

$$ {}_pF_q(a_1,\dots,a_p;b_1,\dots,b_q;z) = \sum_{n\geq 0} \frac{(a_1)_n\,(a_2)_n\cdots(a_p)_n}{(b_1)_n\,(b_2)_n\cdots(b_q)_n}\,\frac{z^n}{n!} $$

where $(a)_n$ is the Pochhammer symbol for the raising factorial:

$$ (a)_0 = 1, \quad (a)_n = a(a+1)(a+2)\cdots(a+n-1). \tag{A.2} $$

The radius of convergence of the power series is ∞ if $p < q+1$, 1 if $p = q+1$, and 0 if $p > q+1$. When $p = q+1$ and $|z| = 1$, the power series converges absolutely if $\mathrm{Re}\big(\sum_i b_i - \sum_j a_j\big) > 0$.

REFERENCES

[1] L. V. Ahlfors, *Complex Analysis: An Introduction to the Theory of Analytic Functions of One Complex Variable*, McGraw-Hill, Inc., New York, USA, third ed., 1979.

[2] A. H. Al-Mohy, *A truncated Taylor series algorithm for computing the action of trigonometric and hyperbolic matrix functions*, SIAM J. Sci. Comput., 40 (2018), pp. A1696–A1713, https://doi.org/10.1137/17M1145227.

[3] A. H. Al-Mohy and N. J. Higham, *A new scaling and squaring algorithm for the matrix exponential*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 970–989, https://doi.org/10.1137/09074721X.

[4] A. H. Al-Mohy and N. J. Higham, *Computing the action of the matrix exponential, with an application to exponential integrators*, SIAM J. Sci. Comput., 33 (2011), pp. 488–511, https://doi.org/10.1137/100788860.

[5] A. H. Al-Mohy and N. J. Higham, *Improved inverse scaling and squaring algorithms for the matrix logarithm*, SIAM J. Sci. Comput., 34 (2012), pp. C153–C169, https://doi.org/10.1137/110852553.

[6] A. H. Al-Mohy, N. J. Higham, and S. D. Relton, *New algorithms for computing the matrix sine and cosine separately or simultaneously*, SIAM J. Sci. Comput., 37 (2015), pp. A456–A487, https://doi.org/10.1137/140973979.

[7] M. Aprahamian and N. J. Higham, *Matrix inverse trigonometric and inverse hyperbolic functions: Theory and algorithms*, SIAM J. Matrix Anal. Appl., 37 (2016), pp. 1453–1477, https://doi.org/10.1137/16M1057577.

[8] T. Clausen, *Ueber die Fälle, wenn die Reihe von der Form* $y = 1+\frac{\alpha}{1}\cdot\frac{\beta}{\gamma}x+\frac{\alpha.\alpha+1}{1.2}\cdot\frac{\beta.\beta+1}{\gamma.\gamma+1}x^2+$ *etc. ein Quadrat von der Form* $z = 1+\frac{\alpha'}{1}\cdot\frac{\beta'}{\gamma'}\cdot\frac{\delta'}{\varepsilon'}x+\frac{\alpha'.\alpha'+1}{1.2}\cdot\frac{\beta'.\beta'+1}{\gamma'.\gamma'+1}\cdot\frac{\delta'.\delta'+1}{\varepsilon'.\varepsilon'+1}x^2+$ *etc. hat.*, Journal für die reine und angewandte Mathematik, 1828 (1828), pp. 89–91, https://doi.org/10.1515/crll.1828.3.89, http://www.degruyter.com/view/j/crll.1828.issue-3/crll.1828.3.89/crll.1828.3.89.xml.

[9] E. B. Davies, *Approximate diagonalization*, SIAM J. Matrix Anal. Appl., 29 (2008), pp. 1051–1064, https://doi.org/10.1137/060659909, http://epubs.siam.org/doi/10.1137/060659909.

[10] P. I. Davies and N. J. Higham, *A Schur–Parlett algorithm for computing matrix functions*, SIAM J. Matrix Anal. Appl., 25 (2003), pp. 464–485, https://doi.org/10.1137/S0895479802410815.

[11] I. P. Gavrilyuk and V. L. Makarov, *Explicit and approximate solutions of second-order evolution differential equations in Hilbert space*, Numerical Methods for Partial Differential Equations, 15 (1999), pp. 111–131, https://doi.org/10.1002/(SICI)1098-2426(199901)15:1⟨111::AID-NUM6⟩3.0.CO;2-L, https://doi.org/10.1002/(SICI)1098-2426(199901)15:1⟨111::AID-NUM6⟩3.0.CO;2-L.

[12] I. P. Gavrilyuk, V. L. Makarov, and V. Vasylyk, *The second-order equations*, in Exponentially Convergent Algorithms for Abstract Differential Equations, Springer Basel, 2011, ch. 4, pp. 127–165, https://doi.org/10.1007/978-3-0348-0119-5_4, http://link.springer.com/10.1007/978-3-0348-0119-5_4.

[13] J. A. Goldstein, *Semigroups of Linear Operators and Applications*, Oxford University Press, 1985.

[14] P. C. Greiner, D. Holcman, and Y. Kannai, *Wave kernels related to second-order operators*, Duke Mathematical Journal, 114 (2002), pp. 329–386, https://doi.org/10.

1215/S0012-7094-02-11426-4, http://projecteuclid.org/Dienst/getRecord?id=euclid.dmj/1087575413/.

[15] V. Grimm and M. Hochbruck, *Rational approximation to trigonometric operators*, BIT Numerical Mathematics, 48 (2008), pp. 215–229, https://doi.org/10.1007/s10543-008-0185-9, http://link.springer.com/10.1007/s10543-008-0185-9.

[16] S. Güttel, *Rational Krylov approximation of matrix functions: Numerical methods and optimal pole selection*, GAMM-Mitteilungen, 36 (2013), pp. 8–31, https://doi.org/10.1002/gamm.201310002, http://doi.wiley.com/10.1002/gamm.201310002.

[17] N. J. Higham, *The Matrix Function Toolbox*. http://www.maths.manchester.ac.uk/~higham/mftoolbox.

[18] N. J. Higham, *Functions of Matrices: Theory and Computation*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008, https://doi.org/10.1137/1.9780898717778.

[19] N. J. Higham and P. Kandolf, *Computing the action of trigonometric and hyperbolic matrix functions*, SIAM J. Sci. Comput., 39 (2017), pp. A613–A627, https://doi.org/10.1137/16M1084225.

[20] N. J. Higham and L. Lin, *An improved Schur–Padé algorithm for fractional powers of a matrix and their Fréchet derivatives*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 1341–1360, https://doi.org/10.1137/130906118.

[21] N. J. Higham and M. I. Smith, *Computing the matrix cosine*, Numer. Algorithms, 34 (2003), pp. 13–26, https://doi.org/10.1023/A:1026152731904.

[22] N. J. Higham and F. Tisseur, *A block algorithm for matrix* 1*-norm estimation, with an application to* 1*-norm pseudospectra*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1185–1201, https://doi.org/10.1137/S0895479899356080.

[23] D. J. Inman, *Vibration with Control*, Wiley, 2006.

[24] A. Magnus and J. Wynn, *On the Padé table of* $\cos z$, Proceedings of the American Mathematical Society, 47 (1975), pp. 361–367, https://doi.org/10.1090/S0002-9939-1975-0367516-3, http://www.ams.org/jourcgi/jour-getitem?pii=S0002-9939-1975-0367516-3.

[25] P. Nadukandi, B. Servan-Camas, P. A. Becker, and J. Garcia-Espinosa, *Seakeeping with the semi-Lagrangian particle finite element method*, Computational Particle Mechanics, 4 (2017), pp. 321–329, https://doi.org/10.1007/s40571-016-0127-2, http://link.springer.com/10.1007/s40571-016-0127-2.

[26] *NIST Handbook of Mathematical Functions*, Cambridge University Press, Cambridge, UK, 2010. http://dlmf.nist.gov.

[27] M. S. Paterson and L. J. Stockmeyer, *On the number of nonscalar multiplications necessary to evaluate polynomials*, SIAM J. Comput., 2 (1973), pp. 60–66, https://doi.org/10.1137/0202007.

[28] J. L. Ramírez Alfonsín, *The Diophantine Frobenius Problem*, Oxford University Press, 2005, https://doi.org/10.1093/acprof:oso/9780198568209.001.0001, http://www.oxfordscholarship.com/view/10.1093/acprof:oso/9780198568209.001.0001/acprof-9780198568209.

[29] A. Shapiro, *Design and Control of an Autonomous Spider-Like Robot for Motion in 2D Tunnels Environments*, PhD thesis, Technion–Israel Institute of Technology, 2003, http://robotics.bgu.ac.il/uploads/7/72/Amir_Shapior_Phd_thesis.pdf.

[30] A. Shapiro, *Stability of second-order asymmetric linear mechanical systems with application to robot grasping*, Journal of Applied Mechanics, 72 (2005), pp. 966–968, https://doi.org/10.1115/1.2042484, http://appliedmechanics.asmedigitalcollection.asme.org/article.aspx?articleid=1415496.

[31] G. Strang and S. MacNamara, *Functions of difference matrices are Toeplitz plus Hankel*, SIAM Review, 56 (2014), pp. 525–546, https://doi.org/10.1137/120897572, http://epubs.siam.org/doi/10.1137/120897572.

[32] M. Taylor, *Noncommutative Harmonic Analysis*, vol. 22 of Mathematical Surveys and Monographs, American Mathematical Society, Providence, Rhode Island, 1986, https://doi.org/10.1090/surv/022, http://www.ams.org/surv/022.

[33] H. Ziegler, *Principles of Structural Stability*, Birkhäuser, Basel, 1977, https://doi.org/10.1007/978-3-0348-5912-7, http://link.springer.com/10.1007/978-3-0348-5912-7.