Research Article

Fuminori Tatsuoka*, Tomohiro Sogabe, Tomoya Kemmochi, and Shao-Liang Zhang

Computing the matrix exponential with the double exponential formula

https://doi.org/10.1515/spma-2024-0013
received January 17, 2024; accepted May 27, 2024

Abstract: This article considers the computation of the matrix exponential e^A with numerical quadrature. Although several quadrature-based algorithms have been proposed, they focus on (near) Hermitian matrices. In order to deal with non-Hermitian matrices, we use another integral representation including an oscillatory term and consider applying the double exponential (DE) formula specialized to Fourier integrals. The DE formula transforms the given integral into another integral whose interval is infinite, and therefore, it is necessary to truncate the infinite interval. In this article, to utilize the DE formula, we analyze the truncation error and propose two algorithms. The first one approximates e^A with the fixed mesh size, which is a parameter in the DE formula affecting the accuracy. The second one computes e^A based on the first one with automatic selection of the mesh size depending on the given error tolerance.

Keywords: matrix function, matrix exponential, numerical quadrature, double exponential formula

MSC 2020: 65F60, 65D30

1 Introduction

The exponential of a matrix $A \in \mathbb{C}^{n\times n}$ is defined as follows:

$$e^A \coloneqq I + A + \frac{1}{2!}A^2 + \frac{1}{3!}A^3 + \cdots,$$

and it arises in several situations in scientific computing. One of the applications is the exponential integrator, a class of numerical solvers for stiff ordinary differential equations [13]. In recent years, e^A also arises in the analysis of directed graphs [4]. Thus, several classes of computational methods of e^A and $e^A\boldsymbol{b}$ ($\boldsymbol{b} \in \mathbb{C}^n$) have been proposed. For example, methods based on Padé approximation of e^z at $z = 0$ [1,2,7], methods based on projections onto Krylov-like subspaces [8,18], methods based on the best approximation of e^z on $(-\infty, 0]$ [19], and methods based on numerical quadrature [5,24]. For a more detailed review of the computational methods of the matrix exponential, see, e.g. [14,15].

In general, quadrature-based methods have two advantages: these algorithms can compute $e^A\boldsymbol{b}$ without computing e^A itself when A is large and either sparse or structured, and it is possible to easily make these algorithms parallel in the sense that the integrand can be computed independently on each abscissa [23, Sect. 18].

*** Corresponding author: Fuminori Tatsuoka,** Department of Applied Physics, Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8603, Japan, e-mail: f-tatsuoka@na.nuap.nagoya-u.ac.jp
Tomohiro Sogabe: Department of Applied Physics, Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8603, Japan, e-mail: sogabe@na.nuap.nagoya-u.ac.jp
Tomoya Kemmochi: Department of Applied Physics, Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8603, Japan, e-mail: kemmochi@na.nuap.nagoya-u.ac.jp
Shao-Liang Zhang: Department of Applied Physics, Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8603, Japan, e-mail: zhang@na.nuap.nagoya-u.ac.jp

For similar reasons, quadrature-based algorithms for other matrix functions have been developed in recent years [21,22]. Thus, we consider quadrature-based algorithms in this article.

Quadrature-based algorithms in [5,24] focus on (nearly) Hermitian matrices. In detail, they are proposed for the efficient computation of Bromwich integrals whose integrand has singularities on or near the negative real axis. For the computation of e^A, (all) the eigenvalues of A are the singularities of the integrand. Hence, when A has eigenvalues with large imaginary parts, the singularities are also located far from the negative real axis, and these algorithms would be inaccurate or may not converge.

The motivation of this study is to construct quadrature-based algorithms that can be used for non-Hermitian matrices. Here, we consider another integral representation

$$\mathrm{e}^A = \frac{2}{\pi}\int_0^\infty x\sin(x)(x^2I + A^2)^{-1}\mathrm{d}x. \tag{1}$$

The derivation of (1) is presented in A. The integral representation (1) holds when all the eigenvalues of A lie in the open left half plane $\{z \in \mathbb{C} : \mathrm{Re}(z) < 0\}$, but this condition can be assumed without loss of generality because $\mathrm{e}^{A+sI} = \mathrm{e}^s\mathrm{e}^A$ $(s \in \mathbb{C})$. Unlike the Cauchy integral used in [5,24], the integral representation (1) allows us to avoid considering its integral path even when A is a non-Hermitian matrix.

Since the integrand in (1) includes an oscillatory term $\sin(x)$ and the interval is half infinite, it is difficult to compute (1) by using typical quadrature formula such as Gaussian quadrature. To compute (1), we consider applying the double exponential (DE) formula specialized to Fourier integrals [17].

In this article, we propose algorithms using the DE formula to compute $\mathrm{e}^A\boldsymbol{b}$ (or e^A) for non-Hermitian matrices with numerical quadrature. The DE formula transforms a given integral into another integral suited for the trapezoidal rule. Because the transformed integral interval is infinite, we need to truncate the infinite sum of the discretized integral appropriately into a finite one. Thus, we propose a truncation method of the infinite sum based on error analysis. In addition, we show an automatic quadrature algorithm that selects the mesh size of the trapezoidal rule depending on the given tolerance.

The organization of this article is as follows: The DE formula used in this article is introduced in Section 2. In Section 3, we analyze the truncation error and propose algorithms. Numerical results are presented in Section 4, and we conclude this article in Section 5.

2 The DE formula for Fourier integrals

The DE formula exploits the fact that the trapezoidal rule for the integrals of analytic functions on the real line converges exponentially [20]. Several types of change of variable are proposed to deal with different types of integrals. In [16], integral forms of

$$I = \int_0^\infty g(x)\sin(x)\mathrm{d}x \tag{2}$$

are considered, where g is a scalar function decaying polynomially as $x \to \infty$.

The DE formula for (2) is as follows. We first select the mesh size $h > 0$ to be used for the trapezoidal rule. Next, we apply a change of variable $x = x_h(t)$ such that $x_h'(t)$ decays double exponentially as $t \to -\infty$ and $x_h(t) - \pi t/h$ decays double exponentially as $t \to \infty$. Then, we compute the transformed integral with the trapezoidal rule:

$$I = \int_{-\infty}^\infty x_h'(t)g(x_h(t))\sin(x_h(t))\mathrm{d}t \approx h\sum_{k=-\infty}^\infty x_h'(kh)g(x_h(kh))\sin(x_h(kh)).$$

The summand decays double exponentially as $|k| \to \infty$, and therefore, the infinite sum can be approximated by a sum of not too many terms. Examples of such change of variable are as follows:

$$x_h(t) = \frac{\pi}{h}\frac{t}{1 - \exp(-\alpha \sinh t)}, \quad (\alpha = 6),$$

which is proposed in [16], or

$$x_h(t) = \frac{\pi}{h}\frac{t}{1 - \exp(-2t - \alpha(1 - \mathrm{e}^{-t}) - \beta(\mathrm{e}^{t} - 1))} \quad \left(\beta = \frac{1}{4}, \quad \alpha = \frac{\beta}{\sqrt{1 + \log(1 + \pi/h)/4h}}\right), \tag{3}$$

which is proposed in [17]. The implementation notes for (3) is presented in Appendix B.

The DE formula can be applied to (1) because the integrand is a rational function of A in the sense of [12, Definition 1.2]. See [21, Section 1.1] for more details of the discussion.

3 Computing e^A with the DE formula

In this section, we propose algorithms for e^A based on the DE formula:

$$\mathrm{e}^A = \int_{-\infty}^{\infty} F_h(t, A)\mathrm{d}t \approx \sum_{k=l}^{r} hF_h(kh, A), \tag{4}$$

where

$$F_h(t, A) \coloneqq x_h'(t)\sin(x_h(t))G(x_h(t), A), \quad G(x, A) \coloneqq \frac{2}{\pi}x(x^2 I + A^2)^{-1}.$$

We may not write the second argument of F_h and G when it is obvious from the context.

The error of the DE formula can be divided into the discretization error and the truncation error:

$$\int_{-\infty}^{\infty} F_h(t, A)\mathrm{d}t - \sum_{k=l}^{r} hF_h(kh, A) = \left(\int_{-\infty}^{\infty} F_h(t)\mathrm{d}t - \sum_{k=-\infty}^{\infty} hF_h(kh)\right) + \left(\sum_{k=-\infty}^{\infty} hF_h(kh) - \sum_{k=l}^{r} hF_h(kh)\right).$$

We employ *a posteriori* error estimation technique to presume the discretization error and provide an upper bound on the truncation error. In Section 3.1, we analyze the truncation error for the given interval, and in Section 3.2, we present the two algorithms; notably, one algorithm uses the technique to achieve the required accuracy.

3.1 Truncation error

We show an upper bound on the truncation error as follows:

Proposition 1. *Suppose that all the eigenvalues of $A \in \mathbb{C}^{n\times n}$ lie in the left half plane. Let $u(t) = 1 - \pi t/hx_h(t)$, where $x_h(t)$ is a DE transformation for Fourier integrals. Given $l, r \in \mathbb{Z}$ such that $l < r$, $x_h(lh) \le \min\{1/\sqrt{2\|A^{-2}\|}, \pi\}$, and $x_h'(kh) \le 2\pi/h$ for $k \ge r + 1$, we have*

$$\begin{aligned}&\left\| \sum_{k=-\infty}^{\infty} hF_h(kh) - \sum_{k=l}^{r} hF_h(kh) \right\| \\ &\quad \le \frac{2h}{\pi}\sum_{k=-\infty}^{l-1} x_h'(kh) + 2\pi \sum_{k=r+1}^{\infty} \frac{ku(kh)}{1 - u(kh)}(\|(A + \mathrm{i}x_h(kh)I)^{-1}\| + \|(A - \mathrm{i}x_h(kh)I)^{-1}\|),\end{aligned} \tag{5}$$

where $\|\cdot\|$ is any subordinate norm.

Proof. From the triangle inequality, it holds that

$$\left\| \sum_{k=-\infty}^{\infty} hF_h(kh) - \sum_{k=l}^{r} hF_h(kh) \right\| \le \left\| \sum_{k=-\infty}^{l-1} hF_h(kh) \right\| + \left\| \sum_{k=r+1}^{\infty} hF_h(kh) \right\|.$$

In this proof, we show (5) that

$$\left\| \sum_{k=-\infty}^{l-1} hF_h(kh) \right\| < \frac{2h}{\pi} \sum_{k=-\infty}^{l-1} x_h'(kh), \tag{6}$$

$$\left\| \sum_{k=r+1}^{\infty} hF_h(kh) \right\| \le 2\pi \sum_{k=r+1}^{\infty} \frac{ku(kh)}{1-u(kh)} (\|(A + \mathrm{i}x_h(kh)I)^{-1}\| + \|(A - \mathrm{i}x_h(kh)I)^{-1}\|). \tag{7}$$

We first show (6). The assumption (0<) $x_h(lh) \le 1/\sqrt{2\|A^{-2}\|}$ gives that for $k \le l, \|x_h(kh)^2 A^{-2}\| \le 1/2$ (<1). Hence, by using the Neumann expansion, we have

$$\begin{aligned} \|x_h(kh)^2(x_h(kh)^2 I + A^2)^{-1}\| &= \|x_h(kh)^2 A^{-2}[-(-x_h(kh)^2 A^{-2}) + I]^{-1}\| \\ &= \left\| \sum_{m=0}^{\infty} [-x_h(kh)^2 A^{-2}]^{m+1} \right\| \\ &\le \sum_{m=1}^{\infty} \|x_h(kh)^2 A^{-2}\|^m \le 1. \end{aligned}$$

In addition, because (0<) $x_h(kh) \le \pi$ for $k < l$, we have

$$\begin{aligned} \left\| \sum_{k=-\infty}^{l-1} hF_h(kh) \right\| &= \frac{2h}{\pi} \left\| \sum_{k=-\infty}^{l-1} \frac{x_h'(kh)\sin(x_h(kh))}{x_h(kh)} x_h(kh)^2 (x_h(kh)^2 I + A^2)^{-1} \right\| \\ &\le \frac{2h}{\pi} \sum_{k=-\infty}^{l-1} x_h'(kh) \frac{\sin(x_h(kh))}{x_h(kh)} \\ &\le \frac{2h}{\pi} \sum_{k=-\infty}^{l-1} x_h'(kh). \end{aligned}$$

Next, we show (7). It is easily seen that

$$\begin{aligned} \left\| \sum_{k=r+1}^{\infty} hF_h(kh) \right\| &\le \frac{h}{\pi} \sum_{k=r+1}^{\infty} x_h'(kh) |\sin(x_h(kh))| \|2x_h(kh)(x_h(kh)^2 I + A^2)^{-1}\| \\ &= \frac{h}{\pi} \sum_{k=r+1}^{\infty} x_h'(kh) |\sin(x_h(kh))| [\|(A + \mathrm{i}x_h(kh)I)^{-1}\| + \|(A - \mathrm{i}x_h(kh)I)^{-1}\|]. \end{aligned}$$

Here, because $x_h'(kh) \le 2\pi/h$ for $k \ge r+1$ and

$$|\sin(x_h(kh))| = \left| \sin\left(\frac{k\pi}{1-u(kh)} \right) \right| = \left| \sin\left(k\pi + \frac{k\pi u(kh)}{1-u(kh)} \right) \right| \le \frac{k\pi u(kh)}{1-u(kh)},$$

we have (7), and (5) is proved. □

We remark that the transformation (3) satisfies $x_h'(t) < 2\pi/h$ for $t \ge t_0$, where t_0 is the solution of $-2t - \alpha(1 - \mathrm{e}^{-t}) - \beta(\mathrm{e}^t - 1) = \log(1 - 1/\sqrt{2})$. For more details, see Appendix B.

3.2 Algorithms

In this subsection, we propose two algorithms. The first one computes e^A by using the DE formula with the given mesh size h and the second one computes e^A with automatic mesh size selection.

In the first algorithm, the input matrix A is shifted so that all the eigenvalues of A lie in the left half plain. Subsequently, the algorithm selects the truncation point of the discretized integral so that the truncation error in 2-norm would be less than the given error tolerance. Then the algorithm calculates the exponential of the shifted matrix. Finally, we obtain e^A by scaling the exponential of the shifted matrix exponential. The details of the algorithm are given in Algorithm 1.

Algorithm 1 Computing e^A with the DE formula with given mesh size

Input $A \in \mathbb{C}^{n\times n}$, mesh size $h > 0$, tolerance for the truncation error $\varepsilon > 0$, shift parameter $\sigma < 0$

1: Compute the right-most eigenvalue of A, λ_{right}

2: $\tilde{A} = A + (\sigma - \lambda_{\text{right}})I$, $\tilde{\varepsilon} = \varepsilon/|e^{\lambda_{\text{right}}-\sigma}|$ ▷ Because of the scaling of A, the tolerance also needs to be scaled.

3: $l, r = \text{GetInterval}(\tilde{\varepsilon}, h, \tilde{A})$

4: $\tilde{X} = h\sum_{k=l}^{r} F_h(kh, \tilde{A})$

Output $X = e^{\lambda_{\text{right}}-\sigma}\tilde{X} \approx e^A$

5: **function** GetInterval (ε, h, A) ▷ Compute an interval whose truncation error is approximately smaller than ε

6: Find the maximum $l \in \mathbb{Z}$ satisfying $2h(\sum_{k=-\infty}^{l-1} x'(kh))/\pi \le \varepsilon/2$

7: Find the minimum $r \in \mathbb{Z}$ satisfying $4\pi\|A^{-1}\|_2\sum_{k=r+1}^{\infty} ku(kh)/(1 - u(kh)) \le \varepsilon/2$

8: **return** l, r

9: **end function**

Algorithm 1 requires the shift parameter σ, and this parameter must be chosen appropriately. Indeed, when σ is positive, the integral representation (1) does not make sense. In addition, when σ is large in the negative direction, it makes inaccurate results because the condition number for e^A becomes large as $\|A\|$ is large, see [9, Section 1]. From the numerical examples in Section 4.2, it may be better to choose σ from $[-5, 0)$.

The computation of the resolvent $(x_h(kh)^2 I + \tilde{A}^2)^{-1}$ is necessary in Step 4 of Algorithm 1, where $\tilde{A}$ is the shifted matrix computed at Step 2. The condition number for the resolvent can be as large as the square of the condition number of $\tilde{A}$, and it may result in low accuracy. In this case, although the computational cost increases, we can evaluate it as follows:

$$(x_h(kh)^2 I + \tilde{A}^2)^{-1} = \frac{\mathrm{i}}{2x_h(kh)}[(\mathrm{i}x_h(kh)I + \tilde{A})^{-1} - (-\mathrm{i}x_h(kh)I + \tilde{A})^{-1}]$$

for accuracy.

To obtain an upper bound on the 2-norm of the truncation error using Proposition 1, it is necessary to compute the norm $\|(\tilde{A} + \mathrm{i}x_h(kh)I)^{-1}\|_2$ for $k \ge r + 1$. As the computation of the norm at each k is challenging, the algorithm assumes that the upper bound on the norm can be approximated by $\|\tilde{A}^{-1}\|_2$. In other words, the upper bound on the truncation error $\|\sum_{k=r+1}^{\infty} hF_h(kh)\|_2$ is approximated by

$$4\pi\|\tilde{A}^{-1}\|_2 \sum_{k=r+1}^{\infty} \frac{ku(kh)}{1 - u(kh)}.$$

Hence, for highly non-normal matrices such that $\|\tilde{A}^{-1}\|_2$ is much larger than $\|(\tilde{A} + \mathrm{i}x_h(kh)I)^{-1}\|_2$, the computational cost of the algorithm may be large because of the overestimation of the truncation error.

The infinite sums in steps 6 and 7 of Algorithm 1 can be approximated without too much computations because the summands $x_h'(kh)$ and $ku(kh)/(1 - u(kh))$ decay double exponentially. For example, Figure 1 illustrates the value of $|x_h'(kh)|$ and $|ku(kh)/(1 - u(kh))|$ for $h = 0.05$, and it shows that these sums can be presumed by computing the first 50 summands.

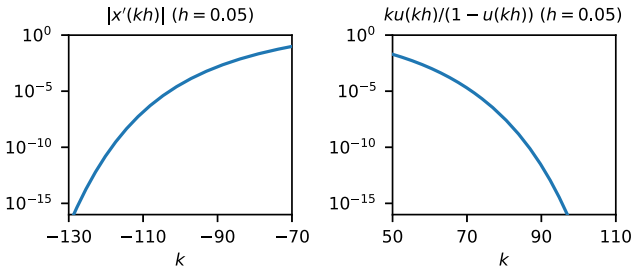

Figure 1: The absolute value of the summands in the upper bound on the truncation error. The figure on the left illustrates $|x'_h(kh)|$ in (5), and the one on the right illustrates $|ku(kh)/(1\ \ u(kh))|$. The transformation (3) is used for this calculation.

We next propose an automatic quadrature algorithm. Ooura and Mori proposed an automatic quadrature algorithm in [17, Section 5] that is applicable when the convergence rate of the DE formula, the constant $\rho > 0$ such that $|\mathcal{I} - h\sum_{k=-\infty}^{\infty} x'_h(kh)g(x_h(kh))\sin(x_h(kh))| = O(\exp(-\rho/h))$, is known.

In the computation of $e^{\tilde{A}}$, the error $\|e^{\tilde{A}} - h\sum_{k=-\infty}^{\infty} F_h(kh, \tilde{A})\|$ can be bounded by $\gamma\exp(-\rho/h)$ with some positive constants γ and ρ because any element of $F_h(t)$ is an analytic function on a strip region $\{z \in \mathbb{C} : |\mathrm{Im}(z)| < d\}$ for some $d > 0$. However, the convergence rate ρ varies with the eigenvalue distribution of $\tilde{A}$ because the singularities of the integrand in (4) are solutions z of $x_h(z)^2 = -\tilde{\lambda}^2$, where $\tilde{\lambda}$ is any eigenvalue of $\tilde{A}$. Thus, we add a presumption of the convergence rate to the algorithm. Once we presume γ and ρ, we can select an appropriate mesh size h by solving $|e^{\lambda_{\mathrm{right}}-\sigma}|\varepsilon = \gamma\exp(-\rho/h)$ for h with a given tolerance ε.

In the algorithm, we first select three mesh sizes $h_1 > h_2 > h_3 > 0$. Let $\tilde{X}_i (i = 1, 2, 3)$ be the computational result of the DE formula for $e^{\tilde{A}}$ with the mesh size h_i. If $\tilde{X}_3$ is much more accurate than $\tilde{X}_1$ and $\tilde{X}_2$, the errors of $\tilde{X}_1$ and $\tilde{X}_2$ can be approximated as follows:

$$\|\tilde{X}_i - e^{\tilde{A}}\| \approx \tilde{\varepsilon}_i \coloneqq \|\tilde{X}_i - \tilde{X}_3\| \quad (i = 1, 2).$$

By solving $\tilde{\varepsilon}_i = \gamma\exp(-\rho/h_i)(i = 1, 2)$ for ρ and γ, we obtain

$$\rho = \frac{h_1 h_2}{h_1 - h_2}\log\left(\frac{\tilde{\varepsilon}_1}{\tilde{\varepsilon}_2}\right), \quad \gamma = \tilde{\varepsilon}_1\exp\left(\frac{\rho}{h_1}\right),$$

and we can approximate $\|\tilde{X}_3 - e^{\tilde{A}}\| \approx \tilde{\varepsilon}_3 \coloneqq \gamma\exp(-\rho/h_3)$. Because $\tilde{\varepsilon}_3$ is just an approximation, we select a safety parameter $\eta > 0$ and stop the algorithm if $\tilde{\varepsilon}_3 \le \eta\tilde{\varepsilon}$ to obtain $\tilde{X}_3 (\approx e^{\tilde{A}})$. If $\tilde{\varepsilon}_3 > \eta\tilde{\varepsilon}$, we set $h_4 = \rho/\log(\gamma/\eta\tilde{\varepsilon})$ and evaluate the DE formula (4) once more. When $\tilde{\varepsilon}_1 \lesssim \tilde{\varepsilon}_2$, it means that the initial mesh size is too large to assume the exponentially convergence of the DE formula (4). For this case, we set $h_i \leftarrow h_{i+1}$ $(i = 1, 2)$ and repeat the procedure. The detail of the algorithm is given in Algorithm 2.

Algorithm 2 Automatic quadrature algorithm for e^A based on the DE formula

Input $A \in \mathbb{C}^{n\times n}$, tolerance for the error $\varepsilon > 0$, shift parameter $\sigma < 0$, initial mesh size $h_1 > 0$, safety parameter $\eta > 0$, the minimum mesh size $h_{\min} > 0$

Output $X \approx e^A$

1 : Compute the rightmost eigenvalue of A, λ_{right}

2 : $\tilde{A} = A + (\sigma - \lambda_{\mathrm{right}})I$, $\quad \tilde{\varepsilon} = \varepsilon/|e^{\lambda_{\mathrm{right}}-\sigma}|$

3 : $h_2 = h_1/2$, $\quad h_3 = h_1/4$

4: $l_i, r_i = \mathrm{GetInterval}(\tilde{\varepsilon}/2, h_i, \tilde{A})$ $(i = 1, 2, 3)$ ▷ Set $\tilde{\varepsilon}/2$ for the input of `GetInterval` to bound the truncation error by $\tilde{\varepsilon}/2$

5: $\tilde{X}_i = h_i\sum_{k=l_i}^{r_i} F_{h_i}(kh_i, \tilde{A})$ $(i = 1, 2, 3)$

6: $\tilde{\varepsilon}_i = \|\tilde{X}_i - \tilde{X}_3\|_2 \quad (i = 1, 2)$
7: $\rho = h_1 h_2 \log(\tilde{\varepsilon}_1/\tilde{\varepsilon}_2)/(h_1 - h_2), \quad \gamma = \tilde{\varepsilon}_1 \exp(\rho/h_1)$
8: $\tilde{\varepsilon}_3 = \gamma \exp(-\rho/h_3)$
9: **if** $\tilde{\varepsilon}_3 < \tilde{\varepsilon}/\eta$ **then**
10: **return** $X = \mathrm{e}^{\lambda_{\mathrm{right}} - \sigma}\tilde{X}_3$
11: **else**
12: $h_4 = \rho/\log(\gamma\eta/\tilde{\varepsilon})$
13: **if** $\tilde{\varepsilon}_1 \lesssim \tilde{\varepsilon}_2$ or $h_4 < h_{\min}$ **then**
14: Set $h_i \leftarrow h_{i+1} (i = 1, 2, 3)$ and $\tilde{X}_i \leftarrow \tilde{X}_{i+1} \quad (i = 1, 2)$
15: $l_3, r_3 = \mathrm{GetInterval}(\tilde{\varepsilon}/2, h_3, \tilde{A})$
16: $\tilde{X}_3 = h_3 \sum_{k=l_3}^{r_3} F_{h_3}(kh_3, \tilde{A})$
17: go to Step 6.
18: **else**
19: $l_4, r_4 = \mathrm{GetInterval}(\tilde{\varepsilon}/2, h_4, \tilde{A})$
20: $\tilde{X}_4 = h_4 \sum_{k=l_4}^{r_4} F_{h_4}(kh_4, \tilde{A})$
21: **return** $X = \mathrm{e}^{\lambda_{\mathrm{right}} - \sigma}\tilde{X}_4$
22: **end if**
23: **end if**

4 Numerical examples

The computation is carried out with Julia 1.10.0 on an Intel Core i5-9600K CPU with 32 GB RAM. The IEEE double-precision arithmetic is used unless otherwise stated. The reference solutions in Section 4.5 are computed by [13/13]-type Padé approximation with the `BigFloat` data type of Julia, which gives roughly 77 significant decimal digits. The Padé approximation is performed after scaling so that the 1 norm of the matrix is less than 5.37. The programs for these experiments are available on GitHub[1]. For the DE formula, we use the change of variable in (3), and the infinite sums in `GetInterval` are truncated to 50 terms.

4.1 Computing the scalar exponential with the DE formula

By using the result in [3], the upper bound on the error of the DE formula can be obtained by $\|\mathrm{e}^A - h\sum_{k=l}^{r} F_h(kh, A)\|_2 \le (1 + \sqrt{2}) \max_{z \in \mathcal{W}(A)} |\mathrm{e}^z - h\sum_{k=l}^{r} F_h(kh, z)|$, where $\mathcal{W}(A) \coloneqq \{\boldsymbol{x}^{\mathsf{H}} A \boldsymbol{x} : \boldsymbol{x} \in \mathbb{C}^n, \|\boldsymbol{x}\|_2 = 1\}$ is the field-of-value of A. Hence, it is worth observing the error of the DE formula for the scalar exponential $|\mathrm{e}^z - h\sum_{k=l}^{r} F_h(kh, z)|$. Figure 2 visualizes the error, and we can notice the followings:

- When z is on or near the negative real axis, the error of the DE formula with $h = 0.1$ is about 10^{-16}.
- When $-\mathrm{Re}(z)$ is large, the error of the DE formula with $h = 0.2$ is about 10^{-16}.
- When z is in a sector region $\{z \in \mathbb{C} : |\arg(-z)| < \pi/4\}$, the error of the DE formula with $h = 0.05$ is about 10^{-16}.

From these findings, when A is a near Hermitian matrix, the error in 2-norm of the DE formula for e^A with $h = 0.1$ will be about 10^{-16}.

1 https://github.com/f-ttok/article-expmde/

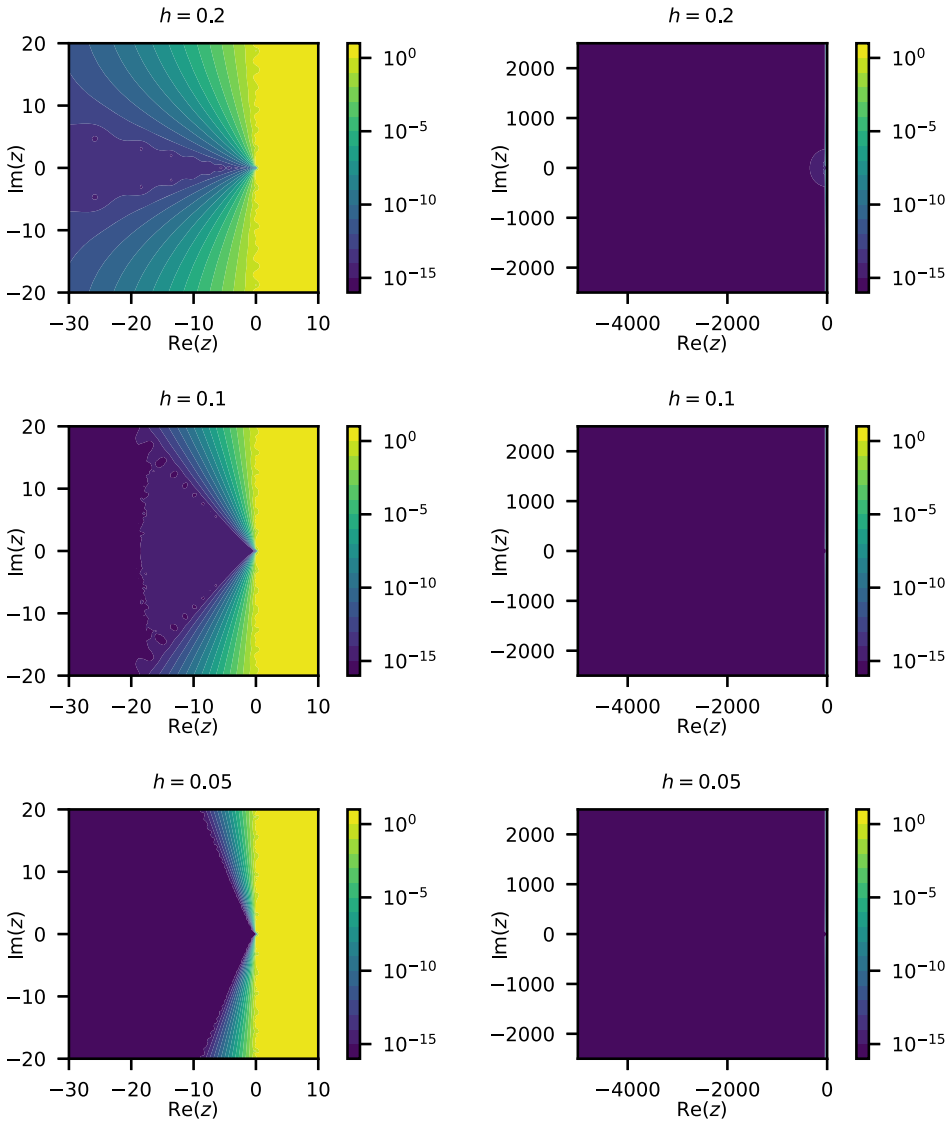

Figure 2: The error of the double exponential formula for the scalar exponential e^z. The mesh size is set to $h = 0.2, 0.1, 0.05$. The left column shows the error for the area $\{z \in \mathbb{C} : -30 \le \mathrm{Re}(z) \le 10, -20 \le \mathrm{Im}(z) \le 20\}$ and the right one shows the error for $\{z \in \mathbb{C} : -5{,}000 \le \mathrm{Re}(z) \le 0, -2{,}500 \le \mathrm{Im}(z) \le 2{,}500\}$.

4.2 Accuracy dependence on the shift parameter σ

Algorithm 1 requires the parameter σ for the input. For reference, we see the accuracy dependence of the DE formula on σ. We consider two test matrices $A_k = ZD_kZ^{-1}$ $(k = 1, 2)$, where $Z \in \mathbb{C}^{50\times 50}$ is generated by using a function `randsvd` in MatrixDepot.jl [25] so that $\kappa(Z) = 10^2$, and $D_k \in \mathbb{C}^{50\times 50}$ is a diagonal matrix whose diagonals are $d_i = 1 - 10^{2k(i-1)/49} + \mathrm{i}\nu_i/20$ $(i = 1, \ldots, 50, \nu_i \sim \mathcal{N}(1, 0))$.

Figure 3 shows the error of the DE formula for $\sigma \in [-10, 5]$. When $\sigma \ge 0$, the error of the DE formula is large for both matrices because A does not satisfy the condition for (1). Thus, σ must be smaller than 0. On the other hand, as σ becomes small, the computational result becomes inaccurate. Hence, it would be better to set $\sigma \in [-5, 0)$.

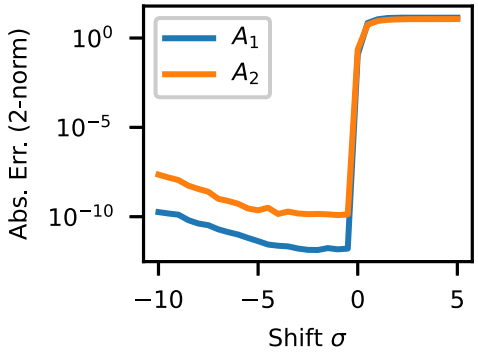

Figure 3: Accuracy dependence of the double exponential formula on the shift parameter σ.

4.3 Accuracy control of automatic quadrature

Here, we see the accuracy control of Algorithm 2. Figure 4 shows the error of Algorithm 2, where the test matrices are the same as in Section 4.2, and the safe parameter is set to $\eta = 2$. We note that these test matrices have field-of-value that crosses the imaginary axis: $\max_{z\in W(A_1)}\mathrm{Re}(z) \approx 612$, and $\max_{z\in W(A_2)}\mathrm{Re}(z) \approx 61{,}403$.

The left figure shows the accuracy dependence on the input tolerance ε, and it shows that the accuracy of Algorithm 2 could be controlled by ε. The right figure shows the error and its presumed value of the computational results for the selected mesh size for the case, where $A = A_1$ and $\varepsilon = 10^{-10}$. From the figure, it is found that the algorithm could presume the error.

4.4 Accuracy of Algorithm 2

In this example, we compute the exponential of 51 test matrices from MatrixDepot.jl [25]. The size of these matrices is 10×10, and they are shifted as $A' = A - \lambda_{\mathrm{right}} I$ before the computation. We computed $\mathrm{e}^{A'}$ by using Algorithm 2 with $\varepsilon = 10^{-8}, 10^{-16}$. For reference, we also computed $\mathrm{e}^{A'}$ with exp function in Julia, which uses the scaling and squaring algorithm [11].

Figure 5 shows the error of this example. It is observed that the error of Algorithm 2 is less than 10^{-8} for most matrices when $\varepsilon = 10^{-8}$. This indicates that Algorithm 2 could control the error. In addition, when $\varepsilon = 10^{-16}$, the accuracy of Algorithm 2 is comparable to that of exp function.

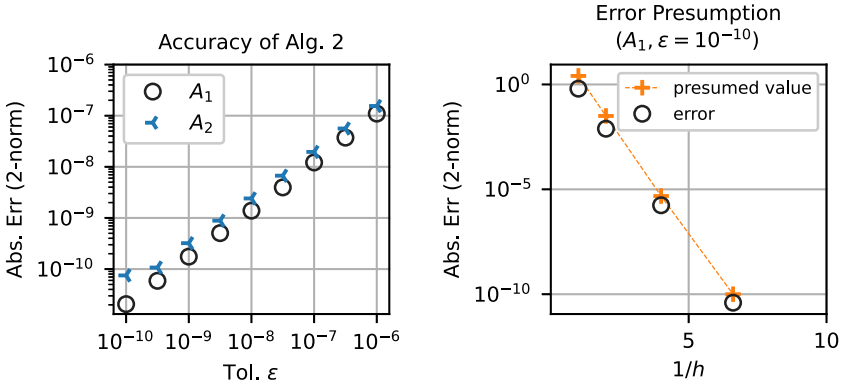

Figure 4: The accuracy of Algorithm 2. The left figure shows the accuracy dependence on the input tolerance ε and the error of the computational results. The right figure shows the error and its presumed value of the case when $A = A_1$ and $\varepsilon = 10^{-10}$. The horizontal axis is the inverse of the selected mesh size, and the vertical axis is the error.

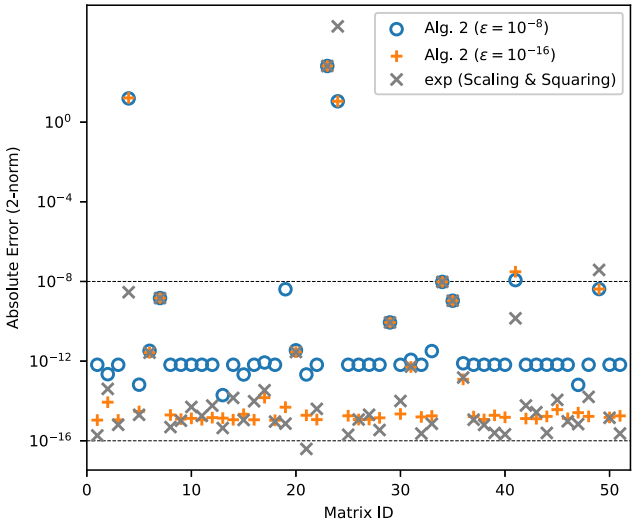

Figure 5: The error $||X - \mathrm{e}^{A'}||_2$ of Algorithm 2 and `exp` function in Julia, which uses the scaling and squaring algorithm in [11].

4.5 A comparison of the DE formula with the Cauchy integral for non-Hermitian matrices

In this example, we compute e^{A} for non-Hermitian matrices by using Algorithm 1 (denoted by `DE`) and an algorithm based on the Cauchy integral.

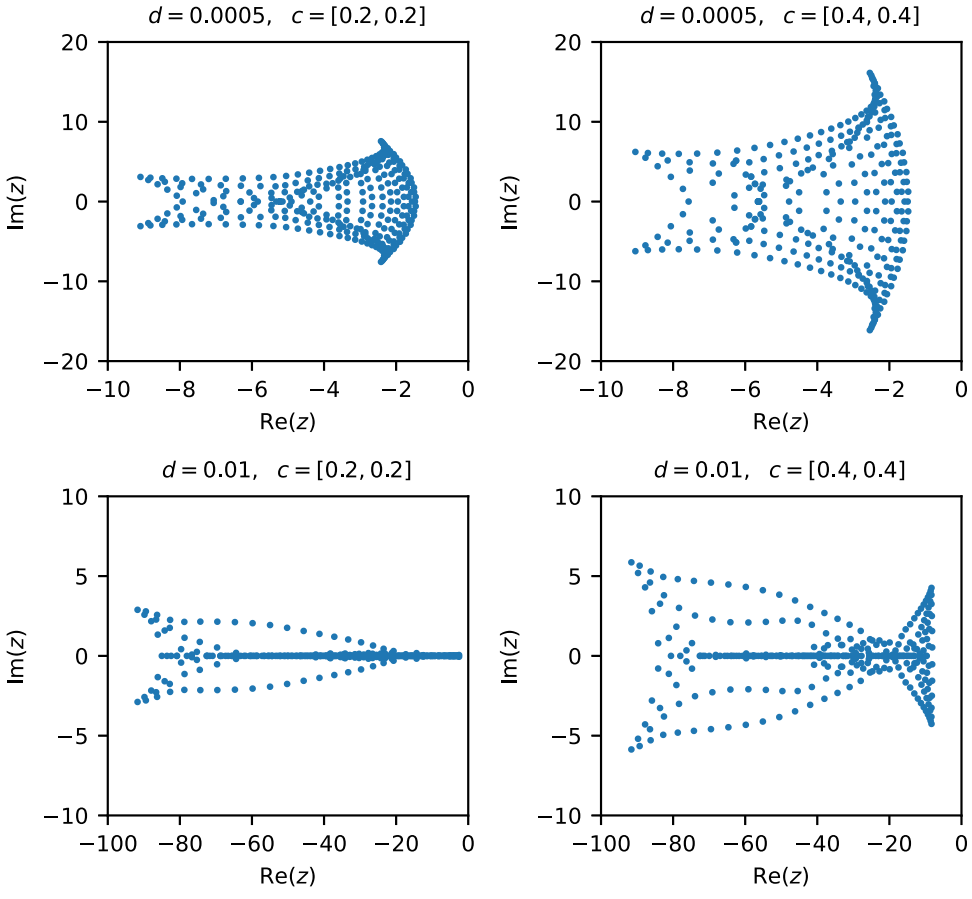

Figure 6: Eigenvalues of the test matrix $M^{-1}K$ in Section 4.5.

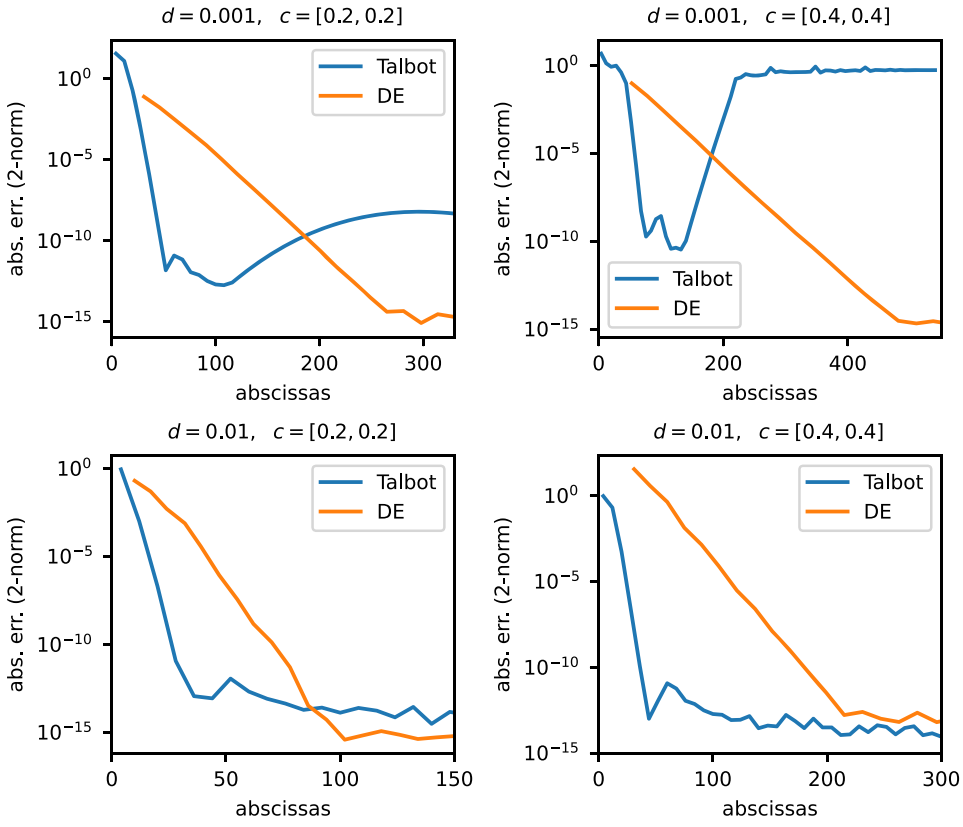

Figure 7: Convergence histories of `DE` and `Talbot` for the problem $\exp(M^{-1}K)$ in Section 4.5.

The matrices are generated from the convection-diffusion problems

$$\begin{aligned} \frac{\partial u}{\partial t} &= d\Delta u - \boldsymbol{c}^\top \nabla u \quad \text{in } \Omega = (0,1)^2, \\ u &= 0 \quad \text{on } \partial\Omega, \\ u(0, \boldsymbol{x}) &= u_0(\boldsymbol{x}), \end{aligned}$$

where $d = 0.001, 0.01$, and $\boldsymbol{c} = [0.2, 0.2], [0.4, 0.4]$. By using the finite element method, we have a linear evolution equation $M\boldsymbol{u}'(t) = K\boldsymbol{u}(t)$ whose solution is $\exp(tM^{-1}K)\boldsymbol{u}(0)$. We use FreeFEM++ [10] for the discretization, and the eigenvalues of $M^{-1}K$ is illustrated in Figure 6.

For `DE`, we set $\sigma = -2.5$ and $\varepsilon \approx 2.2 \times 10^{-16}$. For the Cauchy integral:

$$\mathrm{e}^A = \frac{1}{2\pi \mathrm{i}} \int_\Gamma \mathrm{e}^z (zI + A)^{-1} \mathrm{d}z,$$

where Γ is a contour enclosing the eigenvalue of A, we used the algorithm (denoted by `Talbot`) proposed in [5]. `Talbot` employs the midpoint rule on the Talbot contour

$$\Gamma = \left\{ m(-\tilde{\sigma} + \tilde{\mu}\theta \cot(\tilde{\alpha}\theta) + \tilde{\nu}\mathrm{i}\theta) : -\frac{\pi}{2} \le \theta \le -\frac{\pi}{2} \right\},$$

where m is the number of abscissas, and $\tilde{\sigma}, \tilde{\mu}, \tilde{\alpha}, \tilde{\nu}$ is selected to minimize the error of the midpoint rule. Although `Talbot` is not designed for non-Hermitian matrices, it is not entirely inapplicable to them. Because of the optimality of the integral path of `Talbot`, its convergence is faster than that of `DE`, provided that the eigenvalues do not cross the contour. Here, we intend to show an example illustrating the limitations of `Talbot` and its performance compared to `DE`.

Figure 7 illustrates the convergence histories of these algorithms for $t = 1$. For $d = 0.01$, both `DE` and `Talbot` give accurate results, and `Talbot` converges faster than `DE`. For $d = 0.001$, `DE` still gives accurate results while `Talbot` does not converge. Hence, our algorithm can be a choice of quadrature-based algorithms for non-Hermitian matrices having non-trivial eigenvalue distribution.

5 Conclusion

The DE formula was considered for the computation of e^A. To utilize the DE formula, we analyzed the truncation error and proposed algorithms. Numerical results showed the validity of the algorithm.

Future work includes improvement of the change of variable of the DE formula for e^A and comprehensive performance evaluations of the proposed algorithms on parallel computers for practical problems.

Acknowledgements: The authors are grateful to the two anonymous referees for the careful reading and the comments that substantially enhanced the quality of the manuscript.

Funding information: This work was supported by JSPS KAKENHI Grant Numbers 20H00581 and 20K20397. The authors would like to thank Enago (www.enago.jp) for English language editing.

Author contributions: Fuminori Tatsuoka: conceptualization, experimentation, writing – original draft; Tomohiro Sogabe: writing – review and editing, supervision; Tomoya Kemmochi: writing – review and editing, supervision; Shao-Liang Zhang: writing – review and editing, supervision.

Conflict of interest: The authors declare no conflicts of interest.

Data availability statement: All data that support this study are available at GitHub https://github.com/f-ttok/article-expmde.

References

[1] 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), no. 3, 970–989.

[2] 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), no. 2, 488–511.

[3] M. Crouzeix and C. Palencia, *The numerical range is a* $(1 + \sqrt{2})$*-spectral set*, SIAM J. Matrix Anal. Appl. **38** (2017), no. 2, 649–655.

[4] O. De la Cruz Cabrera, M. Matar, and L. Reichel, *Analysis of directed networks via the matrix exponential*, J. Comput. Appl. Math. **355** (2019), 182–192.

[5] B. Dingfelder and J. A. C. Weideman, *An improved Talbot method for numerical Laplace transform inversion*, Numer Algor, **68** (2015), no. 1, 167–183.

[6] V. Druskin and L. Knizhnerman, *Extended Krylov subspaces: Approximation of the matrix square root and related functions*, SIAM J. Matrix Anal. Appl. **19** (1998), no. 3, 755–771.

[7] M. Fasi and N. J. Higham, *An arbitrary precision scaling and squaring algorithm for the matrix exponential*, SIAM J. Matrix Anal. Appl. **40** (2019), no. 4, 1233–1256.

[8] T. Göckler and V. Grimm, *Uniform approximation of* ϕ*-functions in exponential integrators by a rational krylov subspace method with simple poles*, SIAM J. Matrix Anal. Appl. **35** (2014), no. 4, 1467–1489.

[9] S. Güttel and Y. Nakatsukasa, *Scaled and squared subdiagonal Padé approximation for the matrix exponential*, SIAM J. Matrix Anal. Appl. **37** (2016), no. 1, 145–170.

[10] F. Hecht, *New development in FreeFem++*, J. Numer. Math. **20** (2012), no. 3–4, 251–265.

[11] N. J. Higham, *The scaling and squaring method for the matrix exponential revisited*, SIAM J. Matrix Anal. Appl. **26** (2005), no. 4, 1179–1193.

[12] N. J. Higham, *Functions of Matrices: Theory and Computation*, SIAM, Philadelphia, PA, 2008.

[13] M. Hochbruck and A. Ostermann, *Exponential integrators*, Acta Numer. **19** (2010), 209–286.
[14] C. Moler and C. Van Loan, *Nineteen dubious ways to compute the exponential of a matrix*, SIAM Review **20** (1978), no. 4, 801–836.
[15] C. Moler and C. Van Loan, *Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later*, SIAM Review **45** (2003), no. 1, 3–49.
[16] T. Ooura and M. Mori, *The double exponential formula for oscillatory functions over the half infinite interval*, J. Comput. Appl. Math. **38** (1991), no. 1, 353–360.
[17] T. Ooura and M. Mori, *A robust double exponential formula for Fourier-type integrals*, J. Comput. Appl. Math. **112** (1999), no. 1–2, 229–241.
[18] Y. Saad, *Analysis of some Krylov subspace approximations to the matrix exponential operator*, SIAM J. Numer. Anal. **29** (1992), no. 1, 209–228.
[19] T. Schmelzer and L. N. Trefethen, *Evaluating matrix functions for exponential integrators via Carathéodory-Fejér approximation and contour integrals*, Electron. Trans. Numer. Anal. **29** (2007), 1–18.
[20] H. Takahasi and M. Mori, *Double exponential formulas for numerical integration*, Publ. Res. Inst. Math. Sci. **9** (1974), 721–741.
[21] F. Tatsuoka, T. Sogabe, Y. Miyatake, T. Kemmochi, and S.-L. Zhang, *Computing the matrix fractional power with the double exponential formula*, Electron. Trans. Numer. Anal. **54** (2021), 558–580.
[22] F. Tatsuoka, T. Sogabe, Y. Miyatake, and S.-L. Zhang, *Algorithms for the computation of the matrix logarithm based on the double exponential formula*, J. Comput. Appl. Math. **373** (2020), 112396.
[23] L. N. Trefethen and J. A. C. Weideman, *The exponentially convergent trapezoidal rule*, SIAM Rev. **56** (2014), no. 3, 385–458.
[24] J. A. C. Weideman and L. N. Trefethen, *Parabolic and hyperbolic contours for computing the Bromwich integral*, Math. Comp. **76** (2007), no. 259, 1341–1356.
[25] W. Zhang and N. J. Higham, *Matrix depot: An extensible test matrix collection for Julia*, PeerJ Comput. Sci. **2** (2016), e58.

Appendix

A Derivation of the integral representation (1)

From [6, p. 758], it holds that

$$\frac{e^{-\theta b^{1/2}} - 1}{b} = \int_{-\infty}^{0} \frac{\sin(\theta(-u)^{1/2})}{\pi u} \frac{1}{b-u} du, \tag{A1}$$

where $b^{1/2}$ is the principal square root. Because (A1) is based on the Cauchy integral for $(e^{-\theta b^{1/2}} - 1)/b$, see the proof of [6, Prop. 1], we have

$$B^{-1}(e^{-\theta B^{1/2}} - I) = \int_{-\infty}^{0} \frac{\sin(\theta(-u)^{1/2})}{\pi u} (B - uI)^{-1} du,$$

where all the eigenvalues of $B^{1/2}$ are in the open right half plane. By differentiating both sides twice with respect to θ and then substituting $\theta = 1$, we have

$$e^{-B^{1/2}} = \frac{1}{\pi} \int_{-\infty}^{0} \sin((-u)^{1/2})(B - uI)^{-1} du = \frac{2}{\pi} \int_{0}^{\infty} x \sin(x)(x^2 I + B)^{-1} dx, \tag{A2}$$

where $u = -x^2$. We have (1) by substituting $A = -B^{1/2}$ into (A2).

B Derivative of the change of variable

Let $v(t) = -2t - \alpha(1 - e^{-t}) - \beta(e^{t} - 1)$. Then, the transformation (3) for the DE formula is $x_h(t) = \pi t/h(1 - \exp(v(t)))$ and its derivative is

$$x_h'(t) = \frac{\pi}{h} \frac{1 - e^{v(t)} + tv'(t)e^{v(t)}}{(e^{v(t)} - 1)^2}.$$

Because $\exp(v(t)) \to 1$ as $t \to 0$, we would use $x_h(0) = \pi/h(\alpha + \beta + 2)$ and

$$x_h'(0) = \frac{\pi}{2h} \frac{\alpha^2 + 2\alpha\beta + 5\alpha + \beta^2 + 3\beta + 4}{\alpha^2 + 2\alpha\beta + 4\alpha + \beta^2 + 4\beta + 4}.$$

When $v(t) \le \log(1 - 1/\sqrt{2})(<0)$, the transformation (3) holds the inequality $x_h'(t) < 2\pi/h$ that is used in Proposition 1. First, we note that $v(t)$ monotonically decreases for $t > 0$ because $v'(t) = -2 - \alpha e^{-t} - \beta e^{t} < 0$, where $\alpha > 0$ and $\beta > 0$. Therefore, for $t \ge t_0$, where $t_0(>0)$ be such that $v(t_0) = \log(1 - 1/\sqrt{2})$, it leads $1/(1 - e^{v(t)})^2 \le 2$. Finally, we have

Table A1: The value of t_0, which is the solution of $v(t) = \log(1–1/\sqrt{2})$, for several h

h	t_0
10^{0}	0.493
10^{-1}	0.514
10^{-2}	0.524
10^{-3}	0.526

$$\frac{h}{\pi}x_h'(t) = \frac{1 - e^{v(t)} + tv'(t)e^{v(t)}}{(1 - e^{v(t)})^2} \leq 2(1 - e^{v(t)} + tv'(t)e^{v(t)}) < 2.$$

Experimentally, the value t_0 will be about 0.5 as in Table A1, and the right truncation point $t = rh$ is larger than 3. Hence, the transformation (3) will satisfy the assumption of Proposition 1 in practical cases.