Electronic Transactions on Numerical Analysis.
Volume 54, pp. 558–580, 2021.
Copyright © 2021, Kent State University.
ISSN 1068–9613.
DOI: 10.1553/etna_vol54s558

ETNA
Kent State University **and**
Johann Radon Institute (RICAM)

COMPUTING THE MATRIX FRACTIONAL POWER WITH THE DOUBLE EXPONENTIAL FORMULA[*]

FUMINORI TATSUOKA[†], TOMOHIRO SOGABE[†], YUTO MIYATAKE[‡], TOMOYA KEMMOCHI[†], AND SHAO-LIANG ZHANG[†]

Abstract. Two quadrature-based algorithms for computing the matrix fractional power A^α are presented in this paper. These algorithms are based on the double exponential (DE) formula, which is well-known for its effectiveness in computing improper integrals as well as in treating nearly arbitrary endpoint singularities. The DE formula transforms a given integral into another integral that is suited for the trapezoidal rule; in this process, the integral interval is transformed into an infinite interval. Therefore, it is necessary to truncate the infinite interval to an appropriate finite interval. In this paper, a truncation method, which is based on a truncation error analysis specialized to the computation of A^α, is proposed. Then, two algorithms are presented—one where A^α is computed with a fixed number of abscissa points and one with A^α computed adaptively. Subsequently, the convergence rate of the DE formula for Hermitian positive definite matrices is analyzed. The convergence rate analysis shows that the DE formula converges faster than Gaussian quadrature when A is ill-conditioned and α is a non-unit fraction. Numerical results show that our algorithms achieve the required accuracy and are faster than other algorithms in several situations.

Key words. matrix function, matrix fractional power, numerical quadrature, double exponential formula

AMS subject classifications. 65F60, 65D30

1. Introduction.

We consider the computation of the matrix power A^α for a matrix $A \in \mathbb{C}^{n\times n}$ and a real number $\alpha \in \mathbb{R}$. The matrix exponential is defined as

$$\exp(X) = I + X + X^2/2! + X^3/3! + \cdots \qquad (X \in \mathbb{C}^{n\times n}),$$

and the matrix logarithm is defined as an inverse function of the matrix exponential, i.e., any matrix $Y \in \mathbb{C}^{n\times n}$ satisfying $\exp(Y) = A$ is a logarithm of A. If all the eigenvalues of Y lie in the strip $\{z \in \mathbb{C}\colon |\mathrm{Im}(z)| < \pi\}$, then Y is called the principal logarithm and is denoted by $\log(A)$. Then, the principal matrix power A^α is defined as follows:

$$A^\alpha = \exp(\alpha \log(A)).$$

It is known that A^α exists and is unique when all the eigenvalues of A lie in the set $\{z \in \mathbb{C}\colon z \notin (-\infty, 0]\}$; see, e.g., [11]. The matrix fractional power arises in several situations in computational science, e.g., fractional differential equations [2, 5, 16], lattice QCD calculation [4], and the computation of the weighted matrix geometric mean [8].

In this study, we compute A^α using the double exponential (DE) formula [17] for the integral representation:

$$A^\alpha = \frac{\sin(\alpha\pi)}{\alpha\pi} A \int_0^\infty (t^{1/\alpha} I + A)^{-1}\,\mathrm{d}t \qquad (0 < \alpha < 1); \tag{1.1}$$

[*]Received December 2, 2020. Accepted June 2, 2021. Published online on September 6, 2021. Recommended by Dario Bini. This paper is an augmented version of our previous report (F. Tatsuoka, T. Sogabe, Y. Miyatake, and S.-L. Zhang, A note on computing the matrix fractional power using the double exponential formula, Trans. Jpn. Soc. Ind. Appl. Math., 28 (2018), pp. 142–161), which is in Japanese. There are two differences between this paper and the previous report. One is that this paper presents the convergence rate analysis of the double exponential (DE) formula, while the previous report does not. The other is that this work employs a new transformation $t(x) = \exp(\alpha\pi \sinh(x)/2)$, while the previous one used the transformation $t(x) = \exp(\alpha \sinh(x))$. Several numerical results have shown that the transformation employed in this work is better than the previous one in terms of convergence speed.

[†]Department of Applied Physics, Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8603, Japan (`{f-tatsuoka,sogabe,kemmochi,zhang}@na.nuap.nagoya-u.ac.jp`).

[‡]Cybermedia Center, Osaka University, 1-32 Machikaneyama, Toyonaka, Osaka 560-0043, Japan (`miyatake@cas.cmc.osaka-u.ac.jp`).

see, e.g., [11, Eq. (1.4)]. Without loss of generality, we assume $0 < \alpha < 1$ in this work because for any $\alpha \in \mathbb{R}$, we can compute $A^\alpha = A^{\lfloor \alpha \rfloor} A^{\lfloor \alpha \rfloor - \alpha}$, where $\lfloor \alpha \rfloor \in \mathbb{N}$ and $\lfloor \alpha \rfloor - \alpha \in [0, 1)$. The DE formula transforms the integral (1.1) into another integral suited for the trapezoidal rule. In particular, the transformed integrand decays double exponentially, and the transformed interval is the infinite interval $(-\infty, \infty)$; see, Section 2. Hence, it is necessary to truncate the infinite interval to a finite interval appropriately to utilize the DE formula.

In this paper, we first present an algorithm for computing A^α using the m-point DE formula. To develop this algorithm, we analyze the truncation error of the DE formula for (1.1) and propose a method for truncating the infinite interval to a finite interval within a given tolerance. Then, we present an adaptive quadrature algorithm based on the first algorithm to achieve the required accuracy. This adaptive quadrature algorithm employs an a posteriori error estimation technique discussed in [6]. Subsequently, we analyze the convergence rate of the DE formula for Hermitian positive definite (HPD) matrices and compare it with that of Gaussian quadrature. The convergence rate analysis gives us an a priori estimate of the discretization error similarly as that for Gaussian quadrature in [8]. Hence, for HPD matrices, the number of required abscissa points can be estimated on the basis of this error estimation.

Several computational methods for A^α have been proposed. Examples are the Schur-Padé algorithm [11, 12], the Schur logarithmic algorithm [13], and quadrature-based algorithms [2, 6, 8, 9]. Among them, quadrature-based algorithms have two advantages. One advantage is that these algorithms can compute $A^\alpha \boldsymbol{b}$ ($\boldsymbol{b} \in \mathbb{R}^n$) without computing A^α itself when A is large and either sparse or structured. The other is that these algorithms can be parallelized easily in the sense that the integrand can be computed independently at each abscissa point. See [18, Sect. 18] for more details.

Quadrature-based algorithms were developed based on two integral representations of A^α. One representation utilizes the Cauchy integral [11, Eq. (1.1)], which has been the basis for several algorithms [9]. These algorithms are specialized for the case where all the eigenvalues of A lie on or near the positive real axis. Their convergence rate is better than that of other quadrature-based algorithms, but they require complex arithmetic even if A is a real matrix. To the best of our knowledge, for a general matrix A, no algorithms based on the Cauchy integral are available because of the difficulty in selecting the integral path. The other integral representation is (1.1). Unlike the Cauchy integral, the integral representation (1.1) allows us to avoid selecting an integral path and complex arithmetic for real matrices. In [2, 6, 8], the integral (1.1) was transformed into other integrals on $[-1, 1]$, and the transformed integrals were computed with Gaussian quadrature. For example, the transformations $t(u) = (1+u)/(1-u)$ and $t(v) = (1-v)^\alpha/(1+v)^\alpha$ were considered.

There are two motivations for considering the DE formula. First, the DE formula may converge faster than Gaussian quadrature. The DE formula is known for its effectiveness in dealing with integrals on (half) infinite intervals as well as integrals with endpoint singularities; see, e.g., [14, 18]. Hence, the DE formula is often used when the convergence of Gaussian quadrature is slow. Indeed, the convergence of Gaussian quadrature for (1.1) will be slow when the condition number of A is large and α is a non-unit fraction. This slow convergence is indicated by our numerical results and the analysis for HPD matrices in [8]. Second, the DE formula can be used as an efficient adaptive quadrature algorithm. One of the problems of Gaussian quadrature is that it incurs high computational costs for an a posteriori error estimate. For HPD matrices, one can use an a priori estimate technique that computes λ^α for λ being the extreme eigenvalues of A. This estimation incurs lower costs than the computation of the integrand in (1.1); see, [8]. For general matrices, one can use an a posteriori estimate technique that compares two approximations of A^α with different numbers of abscissa points [6]. However, this estimation incurs high computational costs for Gaussian quadrature, because the integrand must be computed at all the abscissa points for a new approximation with a different number of points. In contrast to Gaussian quadrature, the DE formula employs the trapezoidal rule. The trapezoidal rule can reuse an approximation at a mesh of size h for computing a new approximation at a mesh of size $h/2$. Hence, the DE formula reduces the computational cost.

The rest of this paper is organized as follows. In the last of this section, a brief review of the DE formula is given. In Section 2, the truncation error of the DE formula for A^α is analyzed, and the truncation method and the algorithms are presented. In Section 3, the convergence rate of the DE formula is analyzed and compared with that of Gaussian quadrature. Numerical results are shown in Section 4, and conclusions are given in Section 5.

Notation: Unless otherwise stated, throughout this paper, $\|\cdot\|$ denotes a consistent matrix norm, e.g., the p-norm or the Frobenius norm, while $\|\cdot\|_2$ represents the 2-norm. The symbol $\kappa(A)$ indicates the condition number, i.e., $\kappa(A) = \|A\|_2\|A^{-1}\|_2$, and $\rho(A)$ is the spectral radius of A, i.e., $|\lambda_{\max}|$ for $\lambda_{\max}$ the largest absolute eigenvalue of A. The inverse of $\sinh$ is denoted by asinh. The strip of width $2d$ about the real axis is denoted by $\mathcal{D}_d := \{z \in \mathbb{C} \colon |\mathrm{Im}(z)| < d\}$.

1.1. The DE formula. The DE formula exploits the fact that the trapezoidal rule for the integrals of analytic functions on the real line converges exponentially; see, e.g., [14, 17]. The procedure based on the DE formula for a given integral $\int_a^b f(t)\,\mathrm{d}t$, where f is a scalar function, is as follows:

1. Apply a change of variables $t = t(x)$ to the integral, where the transformed interval is the infinite interval $(-\infty, \infty)$ and the transformed integrand $t'(x)f(t(x))$ decays double exponentially as $x \to \pm\infty$.[1]
2. Truncate the infinite interval to a finite interval $[l, r]$.
3. Compute $\int_l^r t'(x)f(t(x))\,\mathrm{d}x$ by the trapezoidal rule.

If f is analytic on (a, b) and the change of variables is appropriate, then the transformed integrand $t'(x)f(x)$ is analytic on $(-\infty, \infty)$ even if $f(t)$ is not analytic at $t = a, b$. Thus, the DE formula can treat nearly arbitrary endpoint singularities. In addition, the transformation with double exponential decay of the transformed integrand has a certain optimality property; see, e.g., [14, 15]. Therefore, the DE formula can be used for computing integrals on the (half) infinite interval.

For the integral (1.1), one can apply the double exponential formula. The integrand in (1.1) is a matrix function of A in the sense of [10, Def. 1.2], and therefore we have the explicit expressions of each element at hand. Considering the Jordan decomposition, each element of the integrand is 0 or of the form $\frac{\partial^k}{\partial t^k}(t^\alpha + \lambda)^{-1}$ for $k \geq 0$ and λ being the eigenvalues of A. Thus, with transformations such as $t(x) = \exp(\alpha\pi\sinh(x)/2)$, each element of the integrand in (1.1) decays double exponentially as $x \to \infty$, and one can compute integrals simultaneously.

2. Algorithms based on the DE formula.

This section presents two algorithms for computing A^α utilizing the DE formula. Here, the change of variables $t(x) = \exp(\alpha\pi\sinh(x)/2)$ is considered for the DE formula. Thus, the algorithms are based on the integral representation as follows:

$$A^\alpha = \int_{-\infty}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x,$$

[1]More precisely, the expression "decays double exponentially" means that $t'(x)f(t(x))$ is a function in the Hardy space $\mathrm{H}^\infty(\mathcal{D}_d, \omega)$, where ω is a function satisfying the conditions in [15, Thm. 3.2] and $\mathrm{H}^\infty(\mathcal{D}_d, \omega)$ is a function space defined in [15, Sect. 2].

where

$$F_{\mathrm{DE}}(x) = t'(x)F(t(x)), \qquad F(t) = \frac{\sin(\alpha\pi)}{\alpha\pi} A(t^{1/\alpha} I + A)^{-1}.$$

The first algorithm computes A^α based on the m-point DE formula. To control the truncation error, the algorithm selects a finite interval $[l, r]$ satisfying

$$\left\| \int_{-\infty}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x - \int_{l}^{r} F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\|_2 \leq \frac{\epsilon}{2} \tag{2.1}$$

for a given tolerance ϵ. Then, $\int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x$ is computed by using the m-point trapezoidal rule. The second algorithm adaptively computes $\int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x$ until the discretization error is smaller than or equal to $\epsilon/2$. Hence, the total error of the second algorithm is smaller than or equal to ϵ.

The analysis of the truncation error for a given finite interval is provided in Section 2.1. In Section 2.2, we propose a truncation method for (2.1). We explain the algorithms in Section 2.3.

2.1. Truncation error analysis. The bound for the truncation error reads as follows:

LEMMA 2.1. *Let A have no eigenvalues on the closed negative real axis, and $\alpha \in (0, 1)$. For a given interval $[l, r]$, let $a = \exp(\alpha\pi \sinh(l)/2)$ and $b = \exp(\alpha\pi \sinh(r)/2)$. For $a \leq (2\|A^{-1}\|)^{-\alpha}$ and $b \geq (2\|A\|)^\alpha$, we have*

$$\begin{aligned} &\left\| \int_{-\infty}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x - \int_{l}^{r} F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\| \\ &\qquad \leq \frac{\sin(\alpha\pi)[\alpha + (1+\alpha)\|I\|]}{\alpha\pi(1+\alpha)} a + \frac{\sin(\alpha\pi)(3-2\alpha)\|A\|}{\pi(1-\alpha)(2-\alpha)} b^{1-1/\alpha}. \end{aligned} \tag{2.2}$$

Proof. From the triangle inequality, the truncation error is bounded by

$$\left\| \int_{-\infty}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x - \int_{l}^{r} F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\| \leq \left\| \int_{-\infty}^{l} F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\| + \left\| \int_{r}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\|. \tag{2.3}$$

We prove Lemma 2.1 by showing that

$$\left\| \int_{-\infty}^{l} F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\| = \left\| \int_{0}^{a} F(t)\,\mathrm{d}t \right\| \leq \frac{\sin(\alpha\pi)[\alpha + (1+\alpha)\|I\|]}{\alpha\pi(1+\alpha)} a \tag{2.4}$$

and

$$\left\| \int_{r}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\| = \left\| \int_{b}^{\infty} F(t)\,\mathrm{d}t \right\| \leq \frac{\sin(\alpha\pi)(3-2\alpha)\|A\|}{\pi(1-\alpha)(2-\alpha)} b^{1-1/\alpha}. \tag{2.5}$$

First, we show (2.4). For all $t \in [0, a]$ with $a \leq (2\|A^{-1}\|)^{-\alpha}$, we have $t \leq (2\|A^{-1}\|)^{-\alpha}$, and therefore, $\|t^{1/\alpha} A^{-1}\| \leq 1/2$ (< 1). By applying the Neumann series expansion to $(t^{1/\alpha} I + A)^{-1}$, we get

$$(t^{1/\alpha} I + A)^{-1} = A^{-1}[I - (-t^{1/\alpha} A^{-1})]^{-1} = A^{-1} \sum_{k=0}^{\infty} (-1)^k A^{-k} t^{k/\alpha}.$$

Therefore, the integral can be rewritten as follows:

$$\begin{aligned}
A\int_0^a (t^{1/\alpha}I+A)^{-1}\,\mathrm{d}t &= A\int_0^a A^{-1}\left[\sum_{k=0}^{\infty}(-1)^k A^{-k}t^{k/\alpha}\right]\mathrm{d}t\\
&= \sum_{k=0}^{\infty}(-1)^k A^{-k}\left[\int_0^a t^{k/\alpha}\,\mathrm{d}t\right] = \sum_{k=0}^{\infty}(-1)^k A^{-k}\left(\frac{1}{k/\alpha+1}a^{k/\alpha+1}\right)\\
&= aI+\sum_{k=1}^{\infty}(-1)^k A^{-k}\left(\frac{1}{k/\alpha+1}a^{k/\alpha+1}\right).
\end{aligned}$$

Thus, we get

$$\begin{aligned}
\left\|\int_0^a F(t)\,\mathrm{d}t\right\| &= \left\|\frac{\sin(\alpha\pi)}{\alpha\pi}A\int_0^a (t^{1/\alpha}I+A)^{-1}\,\mathrm{d}t\right\|\\
&= \frac{\sin(\alpha\pi)}{\alpha\pi}\left\|aI+\sum_{k=1}^{\infty}(-1)^k A^{-k}\left(\frac{1}{k/\alpha+1}a^{k/\alpha+1}\right)\right\|\\
&\le \frac{\sin(\alpha\pi)}{\alpha\pi}\left[a\,\|I\|+\sum_{k=1}^{\infty}\|A^{-1}\|^k\left(\frac{1}{k/\alpha+1}a^{k/\alpha+1}\right)\right].
\end{aligned} \tag{2.6}$$

Furthermore, from $a \le (2\|A^{-1}\|)^{-\alpha}$, we obtain

$$\begin{aligned}
\sum_{k=1}^{\infty}\|A^{-1}\|^k\left(\frac{1}{k/\alpha+1}a^{k/\alpha+1}\right) &= \sum_{k=1}^{\infty}\frac{\alpha a}{k+\alpha}\left\|a^{1/\alpha}A^{-1}\right\|^k\\
&\le \sum_{k=1}^{\infty}\frac{\alpha a}{1+\alpha}\left(\frac{1}{2}\right)^k = \frac{\alpha}{1+\alpha}a.
\end{aligned} \tag{2.7}$$

We arrive at (2.4) by substituting (2.7) in (2.6).

Next, we show (2.5). The outline of this proof is almost the same as that of (2.4). For all $t \in [b,\infty)$ with $b \ge (2\|A\|)^{\alpha}$, we have $\|t^{-1/\alpha}A\| \le 1/2\ (<1)$. By applying the Neumann series expansion to $(t^{1/\alpha}I+A)^{-1}$, we get

$$(t^{1/\alpha}I+A)^{-1} = t^{-1/\alpha}[I-(-t^{-1/\alpha}A)]^{-1} = t^{-1/\alpha}\sum_{k=0}^{\infty}(-1)^k t^{-k/\alpha}A^k.$$

Therefore, the integral can be rewritten as

$$\begin{aligned}
\int_b^{\infty}(t^{1/\alpha}I+A)^{-1}\,\mathrm{d}t &= \int_b^{\infty}\left[t^{-1/\alpha}\sum_{k=0}^{\infty}(-1)^k t^{-k/\alpha}A^k\right]\mathrm{d}t\\
&= \sum_{k=0}^{\infty}\left[(-1)^k A^k\int_b^{\infty}t^{-(k+1)/\alpha}\,\mathrm{d}t\right] = \sum_{k=0}^{\infty}(-1)^k A^k\frac{\alpha}{k+1-\alpha}b^{1-(k+1)/\alpha}\\
&= \frac{\alpha}{1-\alpha}b^{1-1/\alpha}I+\sum_{k=1}^{\infty}(-1)^k A^k\frac{\alpha}{k+1-\alpha}b^{1-(k+1)/\alpha}.
\end{aligned} \tag{2.8}$$

Thus,

$$\begin{aligned}(2.9)\quad \left\|\int_b^\infty F(t)\,\mathrm{d}t\right\| &= \left\|\frac{\sin(\alpha\pi)}{\alpha\pi}A\int_b^\infty (t^{1/\alpha}I+A)^{-1}\,\mathrm{d}t\right\| \\ &= \left\|\frac{\sin(\alpha\pi)}{\alpha\pi}\left[\frac{\alpha}{1-\alpha}b^{1-1/\alpha}A + A\sum_{k=1}^\infty(-1)^kA^k\frac{\alpha}{k+1-\alpha}b^{1-(k+1)/\alpha}\right]\right\| \\ &\le \frac{\sin(\alpha\pi)}{\pi}\|A\|\left[\frac{1}{1-\alpha}b^{1-1/\alpha}+\sum_{k=1}^\infty\|A\|^k\frac{1}{k+1-\alpha}b^{1-(k+1)/\alpha}\right].\end{aligned}$$

Moreover, from $b \ge (2\|A\|)^\alpha$, we find

$$\begin{aligned}(2.10)\quad \sum_{k=1}^\infty\|A\|^k\frac{1}{k+1-\alpha}b^{1-(k+1)/\alpha} &= b^{1-1/\alpha}\sum_{k=1}^\infty\frac{1}{k+1-\alpha}\|b^{-1/\alpha}A\|^k \\ &\le b^{1-1/\alpha}\sum_{k=1}^\infty\frac{1}{2-\alpha}\left(\frac{1}{2}\right)^k = \frac{1}{2-\alpha}b^{1-1/\alpha}.\end{aligned}$$

We obtain (2.5) by substituting (2.10) into (2.9). In conclusion, by combining (2.3), (2.4), and (2.5), we arrive at (2.2). □

REMARK 2.2. The technique of applying the Neumann series expansion to the integrand was considered in [6, Thm. 2.1], and the following upper bound was derived:

$$\left\|\int_b^\infty (t^{1/\alpha}I+A)^{-1}\,\mathrm{d}t\right\| \le \frac{2b^{1-1/\alpha}}{1/\alpha-1}.$$

Hence, the relation

$$(2.11)\quad \left\|\int_b^\infty F(t)\,\mathrm{d}t\right\| \le \frac{2\sin(\alpha\pi)\|A\|}{\pi(1-\alpha)}b^{1-1/\alpha}$$

holds. Note that the upper bound (2.5) is smaller than (2.11) because

$$\begin{aligned}&\frac{2\sin(\alpha\pi)\|A\|}{\pi(1-\alpha)}b^{1-1/\alpha} - \frac{\sin(\alpha\pi)(3-2\alpha)\|A\|}{\pi(1-\alpha)(2-\alpha)}b^{1-1/\alpha} \\ &\quad = \frac{\sin(\alpha\pi)\|A\|}{\pi(1-\alpha)}b^{1-1/\alpha}\left(2-\frac{3-2\alpha}{2-\alpha}\right) = \frac{\sin(\alpha\pi)\|A\|}{\pi(1-\alpha)}b^{1-1/\alpha}\frac{1}{2-\alpha} > 0.\end{aligned}$$

This difference originates from the separation of the sum into two terms in (2.8).

2.2. A truncation method based on the error analysis. On the basis of Lemma 2.1, we propose a method to truncate the infinite interval to a finite interval within the given tolerance below.

PROPOSITION 2.3. *For given $\epsilon > 0$, let*

$$l = \operatorname{asinh}\left(\frac{2\log(a)}{\alpha\pi}\right), \qquad r = \operatorname{asinh}\left(\frac{2\log(b)}{\alpha\pi}\right),$$

where

$$a = \min\left\{\frac{\epsilon}{4}\frac{\pi\alpha(1+\alpha)}{\sin(\alpha\pi)(1+2\alpha)}, (2\|A^{-1}\|_2)^{-\alpha}\right\},$$

$$b = \max\left\{\left[\frac{\epsilon}{4}\frac{\pi(1-\alpha)(2-\alpha)}{\sin(\alpha\pi)(3-2\alpha)\,\|A\|_2}\right]^{\alpha/(\alpha-1)}, (2\|A\|_2)^\alpha\right\}.$$

Then, the following holds:

$$\left\| \int_{-\infty}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x - \int_{l}^{r} F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\|_2 \leq \frac{\epsilon}{2}. \tag{2.12}$$

Proof. From the definitions of a and b, the inequalities $a \leq (2\|A^{-1}\|_2)^{-\alpha}$ and $b \geq (2\|A\|_2)^{\alpha}$ are valid. Therefore, the interval $[l, r]$ satisfies the assumptions for Lemma 2.1. Hence, it follows that

$$\begin{aligned} &\left\| \int_{-\infty}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x - \int_{l}^{r} F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\|_2 \\ &\qquad \leq \frac{\sin(\alpha\pi)(1+2\alpha)}{\alpha\pi(1+\alpha)} a + \frac{\sin(\alpha\pi)(3-2\alpha)\|A\|_2}{\pi(1-\alpha)(2-\alpha)} b^{1-1/\alpha}. \end{aligned} \tag{2.13}$$

From the definition of a and b, we have

$$\frac{\sin(\alpha\pi)(1+2\alpha)}{\alpha\pi(1+\alpha)} a \leq \frac{\epsilon}{4}, \qquad \frac{\sin(\alpha\pi)(3-2\alpha)\|A\|_2}{\pi(1-\alpha)(2-\alpha)} b^{1-1/\alpha} \leq \frac{\epsilon}{4}. \tag{2.14}$$

We obtain (2.12) by substituting (2.14) into (2.13). ∎

2.3. Algorithms. We first explain the algorithm for computing A^{α} based on the m-point DE formula. This algorithm computes $[l, r]$ that satisfies (2.12) by Proposition 2.3 and computes $\int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x$ by the m-point trapezoidal rule. The details of the algorithm are given in Algorithm 1.

Algorithm 1 m-point DE formula for computing A^{α}.

```
    Input $A$, $\alpha \in (0,1)$, $\epsilon > 0$, $m$.
1:  $l, r = \texttt{GetInterval}(A, \alpha, \epsilon)$
2:  Set $\tilde{F}_{\mathrm{DE}}(x) = \exp(\alpha\pi\sinh(x)/2)\cosh(x)\left[\exp(\pi\sinh(x)/2)I + A\right]^{-1}$
3:  $h = (r-l)/(m-1)$
4:  $T = h[\tilde{F}_{\mathrm{DE}}(l) + \tilde{F}_{\mathrm{DE}}(r)]/2 + h\sum_{k=1}^{m-2}\tilde{F}_{\mathrm{DE}}(l+kh)$
    Output $\sin(\alpha\pi)AT/2 \approx A^{\alpha}$

5:  function $\texttt{GetInterval}(A, \alpha, \epsilon)$
6:      Compute $\|A\|_2$, $\|A^{-1}\|_2$
7:      $a_1 = [\alpha\pi(1+\alpha)\epsilon]/[4\sin(\alpha\pi)(1+2\alpha)]$,    $a_2 = (2\|A^{-1}\|_2)^{-\alpha}$
8:      $a = \min\{a_1, a_2\}$
9:      $b_1 = [\pi(1-\alpha)(2-\alpha)\epsilon]^{\alpha/(\alpha-1)}/[4\sin(\alpha\pi)(3-2\alpha)\,\|A\|_2]^{\alpha/(\alpha-1)}$,    $b_2 = (2\|A\|_2)^{\alpha}$
10:     $b = \max\{b_1, b_2\}$
11:     $l = \operatorname{asinh}(2\log(a)/\alpha\pi)$,    $r = \operatorname{asinh}(2\log(b)/\alpha\pi)$
12:     return $l, r$
13: end function
```

When the tolerance ϵ is sufficiently small, the accurate computation of $\|A\|_2$, $\|A^{-1}\|_2$ in step 6 of Algorithm 1 is not necessary because the errors originating from these values have little effect on the truncation error. Assume that ϵ is sufficiently small and $[a, b] = [a_1, b_1]$. Let $\Delta \in \mathbb{R}$ be such that the computed 2-norm of A is equal to $\|A\|_2(1+\Delta)$. Then, the computed value of b without rounding errors is

$$b = \left[\frac{\pi(1-\alpha)(2-\alpha)}{4\sin(\alpha\pi)(3-2\alpha)\,\|A\|_2}\epsilon_b\right]^{\alpha/(\alpha-1)},$$

where $\epsilon_b = \epsilon/(1+\Delta)$. Hence, the upper bound of the truncation error is

$$\left\| \int_{-\infty}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x - \int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x \right\| \le \frac{1}{2}(\epsilon+\epsilon_b) \le \frac{1}{2}\epsilon\left(1+\frac{1}{1-|\Delta|}\right).$$

For example, when $\Delta = 10^{-2}$, the upper bound of the truncation error is at most 1.005ϵ. Therefore, the effect of these errors will be negligible. In practice, the effect of the error can be canceled by setting ϵ of Algorithm 1 to $\epsilon = \tilde{\epsilon}/(1+1/(1-|\tilde{\Delta}|))$, where $\tilde{\epsilon}$ is the tolerance for the truncation error of the DE formula and $\tilde{\Delta}$ is the tolerance for the relative error used in the computation of $\|A\|_2$.

It may be desirable to apply bounds for the relative truncation error rather than for the absolute truncation error; in other words, one may wish to select $[l, r]$ that satisfies

$$\frac{\left\|A^\alpha - \int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x\right\|_2}{\|A^\alpha\|_2} \le \frac{\tilde{\epsilon}}{2}$$

for a given tolerance $\tilde{\epsilon}$. In this case, one can find such an interval by setting $\epsilon = \rho(A)^\alpha\tilde{\epsilon}$ as input for Algorithm 1. This is because $\rho(A)^\alpha = \rho(A^\alpha)$ bounds $\|A^\alpha\|_2$ from below. The relation $\rho(A)^\alpha = \rho(A^\alpha)$ can be verified from the fact that the eigenvalues of A^α are λ_i^α, where λ_i are the eigenvalues of A; see, e.g., [10, Thm. 1.13].

Before explaining the adaptive quadrature algorithm, we consider an a posteriori estimation of the discretization error. Let m be the number of abscissa points, $h = (r-l)/(m-1)$ be the mesh size, and $T(h)$ be the trapezoidal rule with the mesh size h:

$$T(h) := \frac{h}{2}\left(F_{\mathrm{DE}}(l) + F_{\mathrm{DE}}(r)\right) + h\sum_{k=1}^{m-2} F_{\mathrm{DE}}(l+kh).$$

Then, $T(h/2)$ can be computed as

$$T\left(\frac{h}{2}\right) = \frac{1}{2}T(h) + \frac{h}{2}\sum_{k=1}^{m-1} F_{\mathrm{DE}}\left(l + (2k-1)\frac{h}{2}\right).$$

The upper bound for the discretization error of $T(h/2)$ can be obtained by applying the discussion in [6, p. 424] to the trapezoidal rule. Assume that h is small enough to satisfy $\|T(h) - \int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x\| \le \mu$ and $\|T(h/2) - \int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x\| \le c\mu$, with $0 < c \le 1/2$. If $\|T(h/2) - T(h)\| \le \tilde{\mu}$, then

$$\left\|T(h) - \int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x\right\| \le \left\|T(h) - T\left(\frac{h}{2}\right)\right\| + \left\|T\left(\frac{h}{2}\right) - \int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x\right\|,$$

that is, $\mu \le \tilde{\mu} + c\mu$, or equivalently, $\mu \le \tilde{\mu}/(1-c)$. Hence,

$$\left\|\int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x - T\left(\frac{h}{2}\right)\right\| \le \left\|T(h) - T\left(\frac{h}{2}\right)\right\|. \tag{2.15}$$

Now we state the adaptive quadrature algorithm. First, the algorithm computes $[l, r]$, as described in Algorithm 1. Then, the algorithm repeats the computation of $\int_l^r F_{\mathrm{DE}}(x)\,\mathrm{d}x$ by increasing the number of abscissa points until the discretization error is smaller than $\epsilon/2$. The details of the algorithm are given in Algorithm 2.

Algorithm 2 Adaptive quadrature algorithm for A^α based on the DE formula.

```
    Input $A$, $\alpha \in (0,1)$, $\epsilon > 0$, $m_0 \geq 2$.
 1: $l, r = \texttt{GetInterval}(A, \alpha, \epsilon)$        ▷ GetInterval is defined in Algorithm 1.
 2: Set $\tilde{F}_{\mathrm{DE}}(x) = \exp(\alpha\pi\sinh(x)/2)\cosh(x)\,[\exp(\pi\sinh(x)/2)I + A]^{-1}$
 3: $s = -1$
 4: $h_0 = (r-l)/(m_0-1)$
 5: $T_0 = h_0[\tilde{F}_{\mathrm{DE}}(l) + \tilde{F}_{\mathrm{DE}}(r)]/2 + h_0\sum_{k=1}^{m-2}\tilde{F}_{\mathrm{DE}}(l+kh)$
 6: repeat
 7:     $s = s+1$
 8:     $h_{s+1} = h_s/2$
 9:     $T_{s+1} = T_s/2 + h_{s+1}\sum_{k=1}^{m_s-1}\tilde{F}_{\mathrm{DE}}(l+(2k-1)h_{s+1})$
10:     $m_{s+1} = 2m_s - 1$
11: while $\sin(\alpha\pi)\|AT_{s+1} - AT_s\|/2 > \epsilon/2$
12: $T = T_{s+1}$
    Output $\sin(\alpha\pi)AT/2 \approx A^\alpha$
```

REMARK 2.4. The computational cost of Algorithm 1 for a dense matrix A is about $(8m/3+2)n^3$ if the computational cost for computing the norms $\|A\|_2$ and $\|A^{-1}\|_2$ is of the order $\mathcal{O}(n^2)$. Similarly, the cost of Algorithm 2 is about $(8m_{s+1}/3+2(s+1))n^3$. Further, the cost incurred in computing $A^\alpha\boldsymbol{b}$ with Algorithm 1 is $mc_{\mathrm{abscissa}} + c_{\mathrm{mul}} + c_{\mathrm{norm}}$, where c_{abscissa} is the computational cost of computing $\tilde{F}_{\mathrm{DE}}(x)\boldsymbol{b}$, c_{mul} is that of a matrix-vector multiplication, and c_{norm} is that of computing the norms $\|A\|_2$ and $\|A^{-1}\|_2$. The computational cost of Algorithm 2 is $m_{s+1}c_{\mathrm{abscissa}} + (s+1)c_{\mathrm{mul}} + c_{\mathrm{norm}}$. Because one can use low-accuracy norms, the total computational cost will be almost proportional to the total number of abscissa points.

3. Convergence of the DE formula for HPD matrices.

When A is an HPD matrix, the analysis of a quadrature formula for A^α can be reduced to that of the quadrature formula for the scalar fractional power. For the DE formula, the following relation holds:

$$\left\| \int_{-\infty}^{\infty} F_{\mathrm{DE}}(x)\,\mathrm{d}x - h\sum_{k=-\infty}^{\infty} F_{\mathrm{DE}}(kh) \right\|_2$$
$$= \max_{\lambda\in\Lambda(A)} \left| \int_{-\infty}^{\infty} f_{\mathrm{DE}}(x,\lambda)\,\mathrm{d}x - h\sum_{k=-\infty}^{\infty} f_{\mathrm{DE}}(kh,\lambda) \right|,$$

where h is the mesh size, $\Lambda(A)$ is the spectrum of A, and f_{DE} is the scalar counterpart of $F_{\mathrm{DE}}(x)$. Further, f_{DE} can be expressed as

$$f_{\mathrm{DE}}(x,\lambda) = \frac{\sin(\alpha\pi)\lambda}{2}\frac{\exp(\alpha\pi\sinh(x)/2)\cosh(x)}{\exp(\pi\sinh(x)/2)+\lambda}. \tag{3.1}$$

Section 3.1 presents our analysis of the discretization error of the trapezoidal rule for the integral $\int_{-\infty}^{\infty} f_{\mathrm{DE}}\,\mathrm{d}x$, i.e., the DE formula for λ^α. The error of the DE formula for A^α is discussed in Section 3.2, and the DE formula is compared with Gaussian quadrature in Section 3.3.

3.1. Discretization error of the DE formula for λ^α.

First, we recall the following upper bound for the discretization error of the trapezoidal rule for an integral of an analytic function on the real axis:

THEOREM 3.1 (see, [18, Thm. 5.1]). *Suppose that g is analytic in the strip $\mathcal{D}_d = \{z \in \mathbb{C} \colon |\mathrm{Im}(z)| < d\}$ for some $d > 0$. Further, suppose that $g(z) \to 0$ uniformly as $|z| \to \infty$ in the strip, and for some c, it satisfies*

$$\int_{-\infty}^{\infty} |g(x + \mathrm{i}y)|\,\mathrm{d}x \le c \tag{3.2}$$

for all $y \in (-d, d)$. Then, for any $h > 0$, the sum $h\sum_{k=-\infty}^{\infty} g(kh)$ exists and satisfies

$$\left| h\sum_{k=-\infty}^{\infty} g(kh) - \int_{-\infty}^{\infty} g(x)\,\mathrm{d}x \right| \le \frac{2c}{\exp(2\pi d/h) - 1}.$$

To apply Theorem 3.1 to f_{DE}, we need to identify the strip in which f_{DE} is analytic. We identify such a region using the following proposition:

PROPOSITION 3.2. *For $\lambda > 0$, $f_{\mathrm{DE}}(z, \lambda)$ defined in (3.1) is analytic in the strip $\mathcal{D}_{d_0(\lambda)}$, where*

$$d_0(\lambda) = \arcsin\left(\sqrt{\frac{(\log\lambda)^2 + 5\pi^2/4 - \sqrt{[(\log\lambda)^2 + 5\pi^2/4]^2 - \pi^4}}{\pi^2/2}}\right). \tag{3.3}$$

Proof. We prove Proposition 3.2 by calculating the imaginary part of a pole of f_{DE} closest to the real axis among all the poles of f_{DE}. Because $\exp(\alpha\pi\sinh(z)/2)\cosh(z)$ is analytic in $\mathbb{C}$, we consider only the poles of $1/[\exp(\pi\sinh(z))/2) + \lambda]$.

In fact, the poles are calculated exactly. Consider the following equation:

$$\exp\left(\frac{2}{\pi}\sinh z\right) = -\lambda. \tag{3.4}$$

Let x and y be the real and imaginary parts of z in (3.4), respectively. Then, (3.4) can be rewritten as

$$\exp\left(\frac{\pi}{2}\sinh z\right) = \exp\left(\frac{\pi}{2}\sinh x\cos y + \mathrm{i}\frac{\pi}{2}\cosh x \sin y\right), \tag{3.5}$$

$$-\lambda = \exp(\log\lambda + \mathrm{i}k\pi) \qquad (k = \pm1, \pm3, \pm5\ldots). \tag{3.6}$$

From the right-hand side of (3.5) and that of (3.6), it follows that

$$\pi\sinh(x)\cos(y)/2 = \log\lambda, \tag{3.7}$$

$$\pi\cosh(x)\sin(y)/2 = k\pi. \tag{3.8}$$

To solve (3.7), we consider two cases: either λ is 1 or not. In case $\lambda = 1$, equation (3.7) leads to $\cos y = 0$. Therefore, the minimum absolute solution y is $\pm\pi/2$.

In case $\lambda \ne 1$, $\cos y \ne 0$ because $\log\lambda \ne 0$. After squaring the left and right of both equations of (3.7), we obtain

$$\left(\frac{\pi^2}{4} + \frac{(\log\lambda)^2}{1 - \sin^2 y}\right)\sin^2 y = k^2\pi^2 \tag{3.9}$$

by substituting the relations $\cosh^2 x = 1 + \sinh^2 x$ and $\cos^2 y = 1 - \sin^2 y (\ne 0)$. Equation (3.9) can be solved easily because (3.9) is a quadratic equation in $\sin^2 y$. Thus, the imaginary parts of the solutions are

$$y_k = \arcsin\left(\sqrt{\frac{(\log\lambda)^2 + \pi^2/4 + k^2\pi^2 - \sqrt{[(\log\lambda)^2 + \pi^2/4 + k^2\pi^2]^2 - k^2\pi^4}}{\pi^2/2}}\right).$$

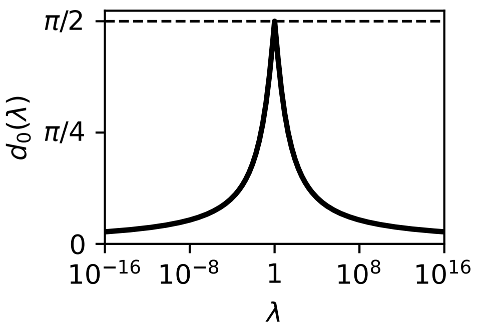

FIG. 3.1. *Distance $d_0(\lambda)$ (in (3.3)) of a strip where $f_{\mathrm{DE}}(z,\lambda)$ is analytic. Note that the horizontal axis is logarithmic.*

We can verify that $|y_1| = |y_{-1}| \le |y_k|$ for all $|k| > 1$. Therefore, the imaginary parts of the poles closest to the real axis are $\pm y_1$.

Because $y_1 = \pi/2$ when $\lambda = 1$, we can combine the two cases for λ. In conclusion, $f_{\mathrm{DE}}(z,\lambda)$ is analytic in $\mathcal{D}_{d_0(\lambda)}$, where $d_0(\lambda) = y_1$. $\square$

To apply Theorem 3.1 to f_{DE}, we make sure that f_{DE} satisfies the condition (3.2). Let δ be a sufficiently small positive number. Then, f_{DE} is continuous and bounded in $\mathcal{D}_{d_0(\lambda)-\delta}$. In addition, by using the results in Appendix A, we can show that there exist constants $c_l, c_r > 0$ and $l, r \in \mathbb{R}$ such that for all $|y| \le d_0(\lambda) - \delta$ $(< \pi/2)$,

$$|f_{\mathrm{DE}}(x+\mathrm{i}y)| < c_l \exp\left(-\frac{\alpha\pi\cos(d_0(\lambda)-\delta)}{2}\sinh(|x|)\right)\cosh x \qquad (x < l),$$
$$|f_{\mathrm{DE}}(x+\mathrm{i}y)| < c_r \exp\left(-\frac{(1-\alpha)\pi\cos(d_0(\lambda)-\delta)}{2}\sinh x\right)\cosh x \qquad (x > r).$$

Therefore, f_{DE} satisfies condition (3.2).

To conclude, the discretization error of the DE formula for λ^α is estimated as

$$\left|\lambda^\alpha - h\sum_{k=-\infty}^{\infty} f_{\mathrm{DE}}(kh,\lambda)\right| \le \mathcal{O}\left(\exp\left(-\frac{2\pi d_0(\lambda)}{h}\right)\right). \tag{3.10}$$

The estimate (3.10) has a limitation in the sense that it only yields the convergence rate. The error of the DE formula goes to 0 as $\lambda \to 0$ since $f_{\mathrm{DE}}(x,\lambda) \to 0$. However, (3.10) does not indicate this because $d_0(\lambda) \to 0$ as $\lambda \to 0$.

3.2. Convergence of the DE formula for A^α. First, let us consider the eigenvalues of A that determine the convergence rate of the DE formula for A^α. Since the error of the DE formula for λ^α is $\mathcal{O}(\exp(-2\pi d_0(\lambda)/h))$, its convergence is slow when $|\log(\lambda)|$ is large, i.e., λ is far away from 1. This is because $d_0(\lambda)$ decreases monotonically when $\lambda > 1$ and increases monotonically when $0 < \lambda < 1$. In addition, the convergence rate of the DE formula for λ^α is approximately equal to that of the DE formula for $(1/\lambda)^\alpha$ because $d_0(\lambda) = d_0(1/\lambda)$. This relation $d_0(\lambda) = d_0(1/\lambda)$ is true because λ appears only in the form of $(\log\lambda)^2$ in $d_0(\lambda)$. These properties of $d_0(\lambda)$ can be verified in Figure 3.1, which displays $d_0(\lambda)$ for $\lambda \in [10^{-16}, 10^{16}]$. Therefore, for sufficiently small h, the discretization error depends on the maximum and the minimum eigenvalues of A, say $\lambda_{\max}, \lambda_{\min}$, and the error can be written as

$$\left\| A^\alpha - h \sum_{k=-\infty}^{\infty} F_{\mathrm{DE}}(kh) \right\|_2 = \max_{\lambda \in \{\lambda_{\max}, \lambda_{\min}\}} \left| \lambda^\alpha - h \sum_{k=-\infty}^{\infty} f_{\mathrm{DE}}(kh, \lambda) \right| .$$

For h that is not sufficiently small, the discretization error may depend on eigenvalues other than the extreme eigenvalues. This is because the error of the DE formula depends on the coefficient multiplying the exponential factor in the discretization error as well as the exponential factor.

Now, we assume that $\lambda_{\max}\lambda_{\min} = 1$ because this assumption minimizes the value

$$\max\{|\log(\lambda_{\max})|, |\log(\lambda_{\min})|\},$$

which determines the convergence rate of the DE formula. Note that the previous condition $\lambda_{\max}\lambda_{\min} = 1$ can be satisfied without loss of generality because we can compute

$$A^\alpha = (\lambda_{\max}\lambda_{\min})^{\alpha/2}(A/\sqrt{\lambda_{\max}\lambda_{\min}})^\alpha,$$

where the product of the maximum and the minimum eigenvalues of $A/\sqrt{\lambda_{\max}\lambda_{\min}}$ is 1. Under this assumption, we have

$$\begin{aligned} \left\| A^\alpha - \sum_{k=-\infty}^{\infty} hF_{\mathrm{DE}}(kh) \right\|_2 &\le \mathcal{O}\left(\exp\left(-\frac{2\pi d_0(\lambda_{\max})}{h} \right) \right) \\ &= \mathcal{O}\left(\exp\left(-\frac{2\pi d_0(\sqrt{\kappa(A)})}{h} \right) \right) \end{aligned} \tag{3.11}$$

because $d_0(\lambda_{\max}) = d_0(1/\lambda_{\max}) = d_0(\lambda_{\min})$.

Finally, to compare the DE formula with Gaussian quadrature, let us rewrite the error (3.11) in terms of the number of abscissa points m. Assume that we have a finite interval $[l, r]$ for which the truncation error is smaller than the discretization error. Then, the total error can be rewritten as

$$\begin{aligned} &\left\| A^\alpha - \frac{h}{2}\left(F_{\mathrm{DE}}(l) + F_{\mathrm{DE}}(r)\right) - h \sum_{k=1}^{m-2} F_{\mathrm{DE}}(l + kh) \right\|_2 \\ &\qquad \le \mathcal{O}\left(\exp\left(-\frac{2\pi d_0\left(\sqrt{\kappa(A)}\right)}{r - l} m \right) \right), \end{aligned} \tag{3.12}$$

where $h = (r - l)/(m - 1)$. The error (3.12) can be estimated by computing $\lambda_{\max}^\alpha$ using the m-point DE formula.

3.3. Comparison of the convergence speed. In this subsection, we compare the convergence speeds of three quadrature formulas. The first one, denoted by `DE`, is the m-point DE formula (Algorithm 1). The second one, denoted by `GJ1`, represents the application of the Gauss-Jacobi (GJ) quadrature to

$$A^\alpha = \frac{2\sin(\alpha\pi)}{\alpha\pi} A \int_{-1}^{1} (1-u)^{1/\alpha - 2} \left[(1+u)^{1/\alpha} I + (1-u)^{1/\alpha} A \right]^{-1} \mathrm{d}u, \tag{3.13}$$

which is obtained by applying $t(u) = (1+u)/(1-u)$ to (1.1). Integral representations similar to (3.13) are considered in [6, 8]. The third one, denoted by `GJ2`, represents the application of the GJ quadrature to

$$A^\alpha = \frac{2\sin(\alpha\pi)}{\pi} A \int_{-1}^{1} (1-v)^{\alpha-1}(1+v)^{-\alpha} \left[(1-v)I + (1+v)A\right]^{-1} \mathrm{d}v, \tag{3.14}$$

which is obtained by applying $t(v) = (1-v)^\alpha/(1+v)^\alpha$ to (1.1). Integral representations similar to (3.14) are considered in [2, 8].

In [8], under the assumption that $\lambda_{\max}\lambda_{\min} = 1$, the error of the `GJ2` method is estimated[2] to be

$$\mathcal{O}\left(\exp\left(-2\log\left(\frac{1+[\kappa(A)]^{1/4}}{|1-[\kappa(A)]^{1/4}|}\right)m\right)\right). \tag{3.15}$$

The error of `GJ1` when α is a unit fraction is estimated [8] as

$$\mathcal{O}\left(\exp\left(-2\log\left(\frac{1+[\kappa(A)]^{\alpha/2}+\sqrt{2[\kappa(A)]^{\alpha/2}(1-\cos(\alpha\pi))}}{\sqrt{1+[\kappa(A)]^{\alpha}+2[\kappa(A)]^{\alpha/2}\cos(\alpha\pi)}}\right)m\right)\right). \tag{3.16}$$

When α is a non-unit fraction, the integrand in (3.13) is not analytic at $u = \pm 1$, and the GJ quadrature for (3.13) may not converge exponentially.

Based on the above discussion, we can represent the error of the three quadrature formulas in the form $\mathcal{O}(\exp(-\phi m))$, where ϕ is a constant depending on the quadrature formulas, $\kappa(A)$, and α. Figure 3.2 displays the values of ϕ for $\alpha = 0.1,\ 0.2, \dots, 0.9$ and $\kappa(A) \in [1, 10^{16}]$. For `DE`, a finite interval $[l, r]$ is obtained by using the subroutine `GetInterval` in Algorithm 1. The parameter ϵ is set to $\epsilon = 2^{-53}\kappa(A)^{\alpha/2}$ so that the relative error is smaller than $2^{-53} \approx 1.1 \times 10^{-16}$. For `GJ1`, the speed ϕ is only considered when α is a unit fraction because of the analyticity of the integrand. Note that the vertical axes in Figure 3.2 show the values ϕ. Hence, the convergence is fast for large ϕ.

Figure 3.2 indicates that the convergence speed of `DE` is higher than that of `GJ2` when $\kappa(A)$ is larger than about 10^4. Conversely, the convergence speed of `DE` is lower than that of `GJ2` for small $\kappa(A)$. When α is a unit fraction and $\kappa(A)$ is large, the convergence speed of `GJ1` is higher than that of each of the other algorithms except for $\alpha = 0.5$. From Figure 3.2, we can select the fastest-converging quadrature formula. For example, when we compute A^α where $\alpha = 0.5$ and $\kappa(A) = 10^{10}$, `DE` is the fastest to converge among the three.

4. Numerical experiments.

The numerical experiments were carried out by using Julia 1.5.1 on a Core-i7 (3.6 GHz) CPU with 16 GB RAM. The IEEE double-precision arithmetic is used unless otherwise stated. For arbitrary precision arithmetic, the programs use the `BigFloat` data type of Julia, which gives roughly 77 significant decimal digits. Abscissas and weights in the GJ quadrature are computed with `FastGaussQuadrature.jl`.[3] The test matrices are listed in Table 4.1. The programs for these experiments are available on Github.[4]

To avoid computing extremely large or extremely small values, test matrices are scaled before the computation of A^α. Specifically, $A^\alpha = c^{-\alpha}(cA)^\alpha$ is computed with the value

[2] Generally, the error of the m-point Gaussian quadrature is given by $\mathcal{O}(\tau^{-2m})$ for some constant τ. In this paper, we use the representation $\mathcal{O}(\exp(-2\log(\tau)m))$ because we want to compare the convergence of Gaussian quadrature with that of the DE formula.

[3] https://github.com/JuliaApproximation/FastGaussQuadrature.jl

[4] https://github.com/f-ttok/article-powmde/

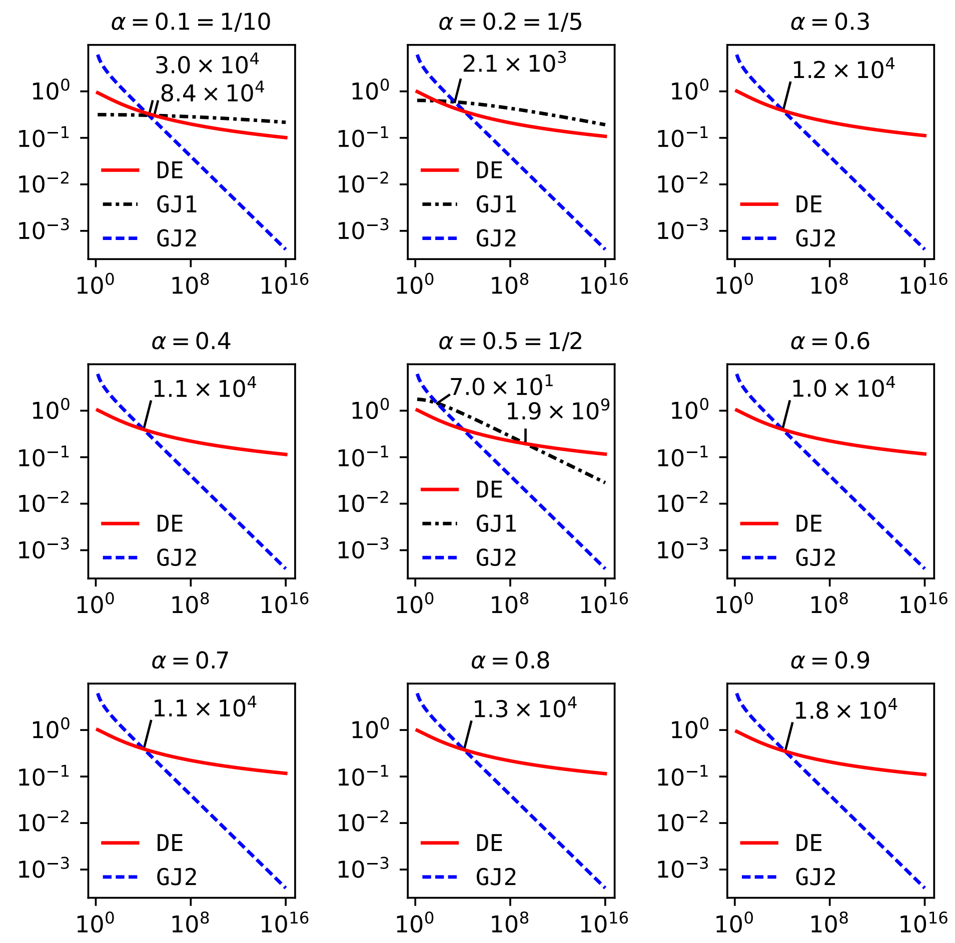

FIG. 3.2. *Convergence speed of the following quadrature formulas: the DE formula (`DE`), the GJ quadrature for (3.13) (`GJ1`), and the GJ quadrature for (3.14) (`GJ2`). The upper bound of the error for `DE`, `GJ1`, and `GJ2` are shown in (3.12), (3.16), and (3.15), respectively. The horizontal axes are the condition number $\kappa(A)$. The vertical axes are the convergence speed, i.e., the constant ϕ in the error of the m-point quadrature formula $\exp(-\phi m)$. Therefore, the convergence of a quadrature formula is fast if the values are large. The condition numbers are indicated for the points where the fastest quadrature formula is changed.*

$c = 1/\sqrt{\sigma_{\max}\sigma_{\min}}$, where $\sigma_{\max}$ and $\sigma_{\min}$ are the maximum and the minimum singular values of A, respectively. In this section, we denote the scaled matrix cA by $\tilde{A}$ for convenience. When A is an HPD matrix, the scaling parameter c can be chosen specialized to `GJ2` so that the error of `GJ2` is small [3]; see, Appendix B. We denote this algorithm `GJ2pre` because we can regard selecting c appropriately as a preconditioning to make good use of `GJ2`.

4.1. Test 1: Selected intervals in our algorithms. In this test, we verify that the subroutine `GetInterval` truncates the infinite interval appropriately. We consider the computation of the square root of test matrices according to the following procedure. First, we compute the reference solution $A^{1/2}$ by using the Denman-Beavers (DB) iteration [10, p. 148] with arbitrary precision. Next, we compute the finite interval in double precision. We set $\epsilon = (c\rho(A))^\alpha 10^{-7}, (c\rho(A))^\alpha 10^{-14}$, where $c = 1/\sqrt{\sigma_{\max}\sigma_{\min}}$, so that the relative error in

TABLE 4.1
Test matrices for the experiments. All matrices are real. The term SPD stands for symmetric positive definite. For `pores_1`*, we computed* $(-A)^\alpha$ *because all the eigenvalues of* `pores_1` *lie in the left half plane. For* `cell1`*,* `TSOPF_RS_b9_c6`*, and* `circuit_3` *we computed* $(A+10^{-8}I)^\alpha$*,* $(A+40.35I)^\alpha$*, and* $(A+3I)^\alpha$*, respectively, because they have real negative eigenvalues.*

Test	Matrix	n	Condition number	Properties
1,2	`ex5` [7]	27	6.6×10^7	SPD
	`pores_1` [7]	30	1.8×10^6	Nonsymmetric
3	`nos4` [7]	100	1.6×10^3	SPD
	`bcsstk04` [7]	132	2.3×10^6	SPD
	`lund_b` [7]	147	3.0×10^4	SPD
4	`SPD_well`	100	1.0×10^2	SPD
	`SPD_ill`	100	1.0×10^7	SPD
	`NS_well`	100	1.0×10^2	Nonsymmetric
	`NS_ill`	100	1.0×10^7	Nonsymmetric
5	`s2rmt3m1` [7]	5489	2.5×10^8	SPD
	`fv3` [7]	9801	2.0×10^3	SPD
	`poisson200`	40000	1.6×10^4	SPD
	`cell1` [7]	7055	8.5×10^8	Nonsymmetric
	`TSOPF_RS_b9_c6` [7]	7224	5.6×10^4	Nonsymmetric
	`circuit_3` [7]	12127	1.1×10^2	Nonsymmetric

TABLE 4.2
Right end values r *of the finite intervals* $[l, r]$ *selected by Algorithm 1.*

Matrix	ϵ	`svdvals`	`Arpack`	`Arpackmod`
`ex5`	10^{-7}	4.0189456993	4.0189454235	4.0501871836
`ex5`	10^{-14}	4.5713980347	4.5713979514	4.5895014861
`pores_1`	10^{-7}	3.9825518994	3.9825518993	4.0149323090
`pores_1`	10^{-14}	4.5506094014	4.5506093993	4.5690896458

the 2-norm is smaller than 10^{-7} or 10^{-14}. The norms $\|\tilde{A}\|_2$ and $\|\tilde{A}^{-1}\|_2$ are computed in three ways. First, the computation is performed using the function `svdvals` in Julia, which computes all the singular values of a matrix. For the second method, the computation is performed to achieve a three-digit accuracy by using `Arpack.jl`,[5] which computes the eigenvalues and singular values by the Arnoldi method. For the third method, the computation is performed using `Arpack.jl` as in the second method, but we set $\epsilon = \tilde{\epsilon}/(1+1/(1-10^{-3}))$ for $\tilde{\epsilon} = 10^{-7}, 10^{-4}$. The third method is denoted as `Arpackmod`. Then, we compute $A^{1/2}$ by using the m-point DE formula with arbitrary precision to avoid the error caused by a matrix inversion. For reference, we also compute $A^{1/2}$ with the wider finite interval $[-6, 6]$.

As a first result, the right end values r of the finite intervals $[l, r]$ computed by `GetInterval` are listed in Table 4.2. The data in Table 4.2 show that there is only a slight difference between the interval with accurate norms (`svdvals`) and that with rough norms (`Arpack`). If we consider the error of the norms (`Arpackmod`), `GetInterval` provides a slightly wider interval than that of `svdvals`. Hence, the error of the norms can be neglected.

[5] `https://github.com/JuliaLinearAlgebra/Arpack.jl`

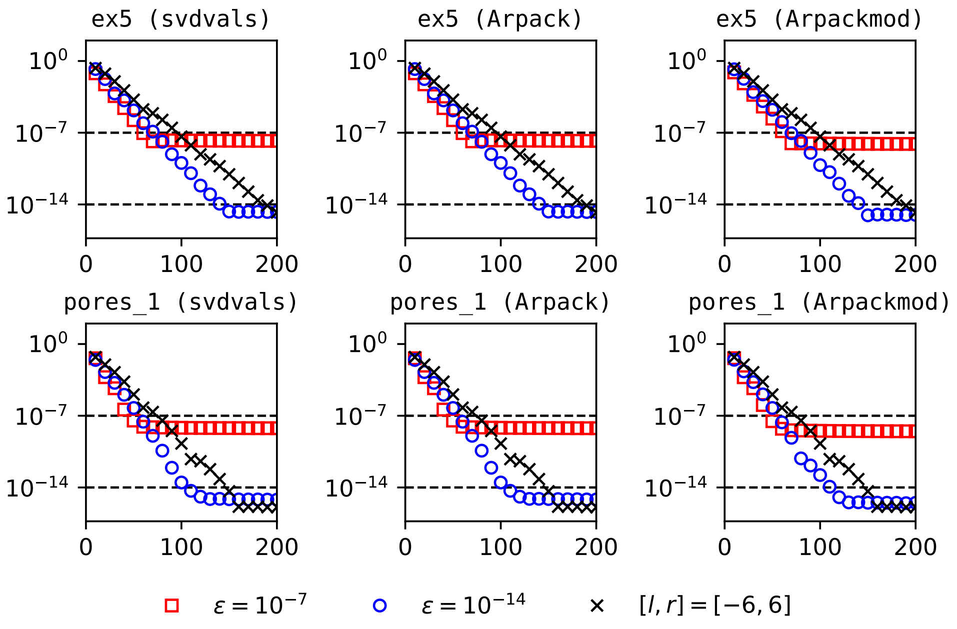

FIG. 4.1. *Convergence histories of the DE formula for $A^{1/2}$ in Test 1. The vertical axes show the relative error in the 2-norm, and the horizontal axis shows the number of abscissa points. The labels in each graph indicate the test matrix and the method for computing the norms $\|\tilde{A}\|_2$, $\|\tilde{A}^{-1}\|_2$. The notation* `svdvals` *indicates that the parameters are computed by using* `svdvals`, `Arpack` *indicates that the norms are computed approximately by using* `Arpack.jl`, *and* `Arpackmod` *indicates that the norms are computed by using* `Arpack.jl`, *but ϵ is set so that the error of norms does not affect the estimate of the truncation error. The values ϵ in the legend represent the tolerance of the relative error, and "$[l, r] = [-6, 6]$" indicates that the approximation is computed with a wide interval* $[-6, 6]$.

Next, in Figure 4.1, we present the convergence histories of the DE formula. Figure 4.1 indicates that the approximations achieved the required accuracy in all cases. Therefore, the truncation error of Algorithm 1 is smaller than the given tolerance. As expected from the results listed in Table 4.2, the computational results are accurate regardless of the accuracy of the parameters. In addition, the DE formula with selected finite intervals converges faster than the DE formula with the wide interval. In conclusion, `GetInterval` determines finite intervals appropriately, and we can use the roughly computed parameters.

4.2. Test 2: Accuracy of Algorithm 2. In this test, we investigate the accuracy of Algorithm 2 by computing A^{α} ($\alpha = 0.2, 0.5, 0.8$). The reference solution A^{α} is computed by using the DB iteration and the inverse Newton method [10, Alg. 7.14] with arbitrary precision. The reference solution $A^{0.8}$ is computed via $A^{0.8} = (A^{1/5})^4$. Then, we compute $\tilde{A}^{\alpha}$ by using Algorithm 2 with arbitrary precision. In Algorithm 2, the parameter ϵ is set so that the relative error in the 2-norm is smaller than 10^{-7} and 10^{-14}.

The relative error and its estimate in Algorithm 2 are given in Figure 4.2. Figure 4.2 indicates that Algorithm 2 achieves the required accuracy for all cases. The behavior of the errors shows that Algorithm 2 controls the truncation error in (2.12), and the behavior of the estimates shows that Algorithm 2 controls the discretization error in (2.15).

4.3. Test 3: Verifying the convergence speed of the DE formula. In this test, we verify the convergence speed of the DE formula for the fractional power of symmetric positive definite (SPD) matrices. We compute $\tilde{A}^{1/2}$ by using the DE formula (Algorithm 1) with

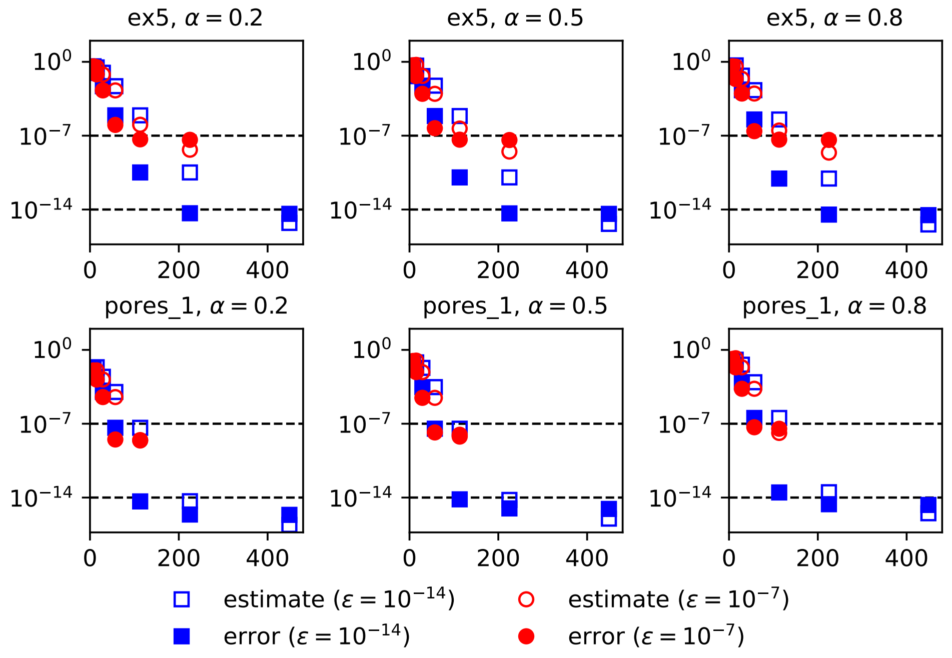

FIG. 4.2. *Relative error and its estimate in Algorithm 1. The vertical axes show the relative error in the 2-norm, and the horizontal axes show the number of abscissa points m_s. The values ϵ represent the tolerance for the relative error.*

$\epsilon = 2^{-53}(c\rho(\tilde{A}))^{\alpha}$ in double-precision arithmetic. Subsequently, we compute $\tilde{\lambda}_{\max}^{1/2}$, i.e., the square root of the maximum eigenvalue of $\tilde{A}$, by using the DE formula with an integral interval identical to that of the DE formula for $\tilde{A}^{1/2}$. The convergence histories of the DE formula are presented in Figure 4.3. In addition, we illustrate the estimates of the convergence rate (3.12) in Figure 4.3.

Figure 4.3 shows that the convergence of the DE formula for $\tilde{A}^{1/2}$ is almost equal to that of the DE formula for $\tilde{\lambda}_{\max}^{1/2}$. Furthermore, the convergence rates of the quadrature formulas are approximately equal to the estimates. These results support our convergence rate analysis, and hence, we can predict the convergence of the DE formula for $\tilde{A}^{\alpha}$ from the behavior of the DE formula for $\tilde{\lambda}_{\max}^{\alpha}$.

4.4. Test 4: Comparison of the DE formula with Gauss-Jacobi quadrature. We compare the convergence of the DE formula with that of GJ quadrature. For this test, we generate four test matrices:

1. We generate two SPD matrices (`SPD_well` and `SPD_ill`).
   1-1. We generate an orthogonal matrix Q by taking the orthogonal vector of the QR decomposition of a random 100×100 matrix.
   1-2. We generate a diagonal matrix $D = \mathrm{diag}(d_1, \dots, d_n)$ whose diagonal elements are derived from the geometric sequence $\{d_i\}_{i=1,\dots,100}$, where $d_1 = \kappa^{-1/2}$ and $d_{100} = \kappa^{1/2}$ for $\kappa = 10^2$.
   1-3. $A_{\text{SPD\_well}} = QDQ^{\top}$
   1-4. We repeat Step 1–2 and 1–3 with the setting $\kappa = 10^7$ and generate $A_{\text{SPD\_ill}}$.

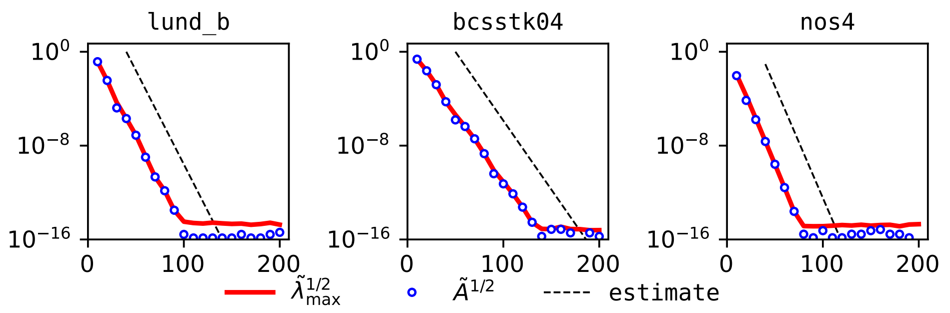

FIG. 4.3. *Convergence histories of the DE formula for the square root of the test matrices and its maximum eigenvalues. The vertical axes show the relative error in the 2-norm, and the horizontal axes show the number of abscissa points.*

2. We generate two nonsymmetric matrices (`NS_well` and `NS_ill`).
   - 2-1. We generate a random 100×100 matrix R.
   - 2-2. We generate $A_{\text{NS\_well}} = \exp(cR)$ with $c = c_{\text{well}} \approx 8.633 \times 10^{-2}$ so that $\kappa(A_{\text{NS\_well}}) \approx 10^2$. The matrix exponential is computed by using the `exp` function in Julia.
   - 2-3. We scale $A_{\text{NS\_well}}$ so that the product of the maximum singular value and the minimum singular value becomes 1.
   - 2-4. We repeat Step 2–2 and 2–3 and generate $A_{\text{NS\_ill}}$ with the setting $c = c_{\text{ill}} \approx 3.029 \times 10^{-1}$ so that $\kappa(A_{\text{NS\_ill}}) \approx 10^7$.

Then, we compute A^α for $\alpha = 0.2, 0.5, 0.8$. The reference solutions are computed as in Test 1. The convergence histories are presented in Figure 4.4. Figure 4.4 shows that the DE formula works well in all cases. In addition, when α is a non-unit fraction ($\alpha = 0.8$) and $\kappa(A)$ is large (`SPD_ill` and `NS_ill`), `DE` is the fastest to converge among the four or three quadrature formulas.

4.5. Test 5: Computational time for $A^\alpha b$. In this test, we measure the computational time of quadrature-based algorithms for computing $A^\alpha \boldsymbol{b}$. The test matrix `poisson200` is generated by $A = L_{200} \otimes I_{200} + I_{200} \otimes L_{200}$, where $L_{200} = \operatorname{tridiag}(-1, 2, -1) \in \mathbb{R}^{200\times 200}$, I_{200} is the identity matrix of size 200, and $\otimes$ is the Kronecker product. The vector $\boldsymbol{b}$ is set to a random vector with norm 1. The test procedure is as follows:

SPD matrices: For SPD matrices, we consider five quadrature-based algorithms: `GJ1`, `GJ2`, `GJ2pre`, `DE`, and the algorithm based on the Cauchy integral presented in [9, Sect. 3], which is denoted by `Cauchy`. The algorithm `Cauchy` requires the computation of the complete elliptic integral of the first kind and the Jacobi elliptic functions. We implement them by using `Elliptic.jl`[6] for real arguments. For complex arguments, we use the relations in [1, Eqs. (16.21.1)–(16.21.4)].
First, we estimate the number of abscissa points m such that the absolute error $\|A^\alpha \boldsymbol{b} - \boldsymbol{x}\|_2$ is smaller than or equal to 10^{-6}. For `GJ1`, `GJ2`, and `DE`, we can predict the convergence of these algorithms by the scalar convergence of the extreme eigenvalue of $\tilde{A}$. Hence, we count the number of abscissa points in the scalar case and use the same number for the matrix case. For `GJ2pre`, we use the upper bound for the error, which can be computed by the extreme eigenvalues of A, derived in [3]. The extreme eigenvalues are computed with three-digit accuracy using `Arpack.jl`.

[6]https://github.com/nolta/Elliptic.jl

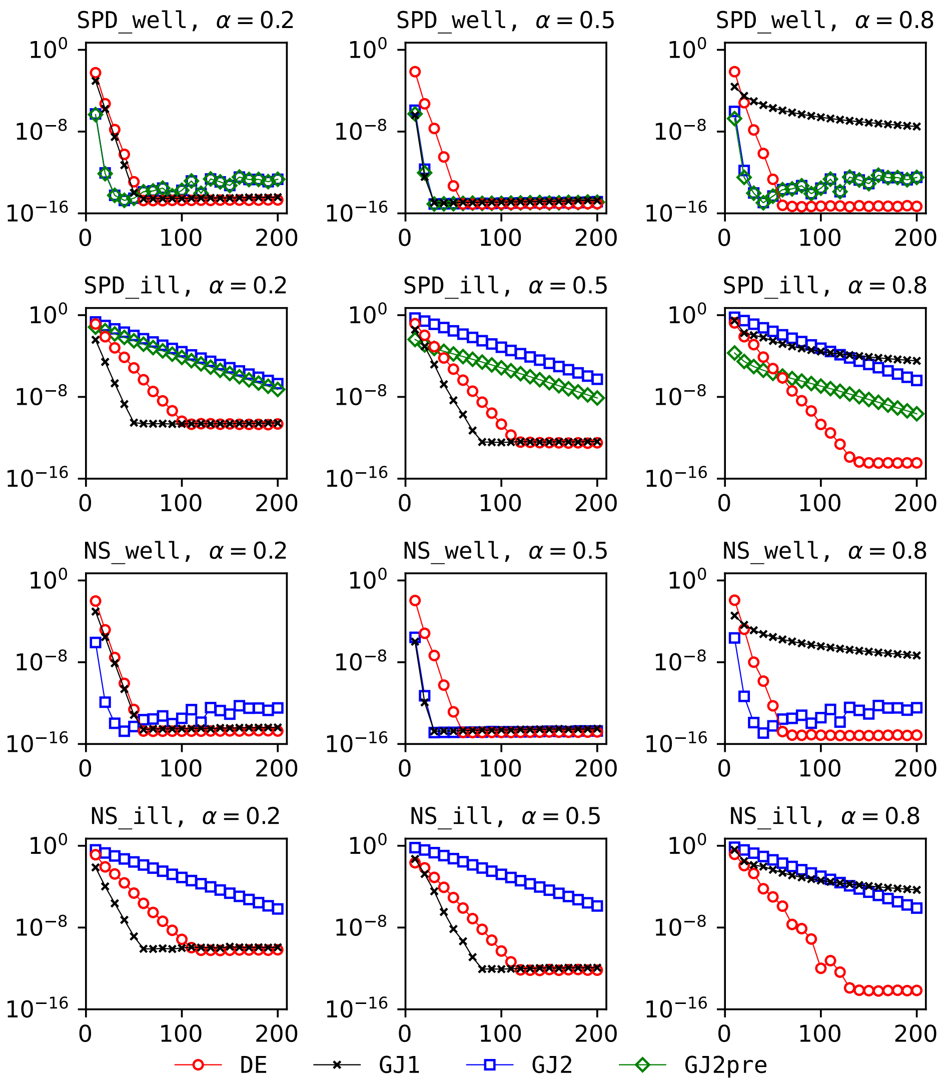

FIG. 4.4. *Convergence histories of the four or three quadrature formulas: The m-point DE formula (*`DE`*), the GJ quadrature for* (3.13) *(*`GJ1`*), the GJ quadrature for* (3.14) *(*`GJ2`*), and preconditioned* `GJ2` *for HPD matrices (*`GJ2pre`*). The vertical axes show the relative error in the Frobenius norm, and the horizontal axes show the number of abscissa points.*

For `Cauchy`, the number of abscissa points is estimated by applying `Cauchy` to a 2×2 matrix whose eigenvalues are the extreme eigenvalues of $\tilde{A}$ as in [8]. When the estimated number of the abscissa points is greater than 1000, we set this number to 1000. Then, we compute $A^{\alpha}\boldsymbol{b}$. The linear systems in the integrand are solved using a sparse direct linear solver in `SuiteSparse.jl`.[7]

[7]`https://github.com/JuliaLinearAlgebra/SuiteSparse.jl`

Nonsymmetric matrices: For general matrices, we consider three adaptive quadrature algorithms, namely, Algorithm 2 (denoted by `DE`) and two algorithms based on the GJ quadrature for (3.13) and (3.14) (denoted by `GJ1` and `GJ2`, respectively). Algorithms `GJ1` and `GJ2` compute approximations by using the $(2^s m_0)$-point GJ quadrature for $s = 0, 1, 2, \ldots$ As in [6, Alg. 5.2], the error of the GJ quadrature is estimated by

$$\|\boldsymbol{x}_{2m} - A^\alpha \boldsymbol{b}\|_2 \leq \|\boldsymbol{x}_{2m} - \boldsymbol{x}_m\|_2 ,$$

where $\boldsymbol{x}_m$ is the approximation of the m-point GJ quadrature. In contrast to the algorithm of [6, Alg. 5.2], our algorithms do not compute the Schur decomposition and the matrix square root because A is sparse.
As for SPD matrices, we set the tolerance so that $\|A^\alpha \boldsymbol{b} - \boldsymbol{x}\|_2 \leq 10^{-6}$. For all algorithms, we set the initial number of abscissa points m_0 to 8. The extreme singular values for scaling the matrices are computed via `Arpack.jl`, and the linear systems are solved using `SuiteSparse.jl`. The algorithms are stopped when the number of the evaluations of the integrand exceeds 1000.

Table 4.3 lists the computational times and the number of evaluations of the integrand of the quadrature-based algorithms for SPD matrices. This computational time includes the time for computing the extreme singular values and the extreme eigenvalues for scaling the test matrices. When α is a non-unit fraction ($\alpha = 0.8$) and A is an ill-conditioned matrix (`s2rmt3m1`), `DE` is the fastest. Otherwise, `GJ1` or `GJ2pre` are the fastest. Although `Cauchy` requires a small number of abscissa points, it is not the fastest in this test because of the use of complex arithmetic. Table 4.4 lists the times of the algorithms for nonsymmetric matrices. For well-conditioned matrix (`circuit_3`), `GJ2` is the fastest, otherwise, `DE` is the fastest. The algorithm `DE` seems to be more robust than `GJ1` and `GJ2` because of the efficiency of the discretization error estimation.

TABLE 4.3
Comparison of quadrature-based algorithms for SPD matrices in terms of CPU time (in seconds) and the number of evaluations of the integrand (in parentheses). If the number of evaluations exceeds 1000, the corresponding results are underlined. The results of the fastest algorithm among the five algorithms are written in bold font.

Matrix	α	`GJ1`		`GJ2`		`GJ2pre`		`DE`		`Cauchy`	
`s2rmt3m1`	0.2	**0.59**	**(39)**	7.20	(511)	6.38	(451)	1.07	(73)	1.30	(21)
`s2rmt3m1`	0.8	<u>14.37</u>	(<u>1000</u>)	10.55	(731)	6.98	(485)	**1.52**	**(95)**	1.90	(31)
`fv3`	0.2	0.30	(23)	0.30	(24)	**0.28**	**(23)**	0.36	(30)	0.41	(11)
`fv3`	0.8	2.10	(193)	0.32	(26)	**0.26**	**(21)**	0.38	(29)	0.38	(10)
`poisson200`	0.2	**1.43**	**(25)**	1.82	(41)	1.69	(38)	1.59	(33)	2.01	(13)
`poisson200`	0.8	14.34	(352)	1.97	(45)	**1.55**	**(34)**	1.65	(33)	1.91	(12)

TABLE 4.4
Comparison of quadrature-based algorithms for nonsymmetric matrices in terms of CPU time (in seconds) and the number of evaluations of the integrand (in parentheses). If the number of evaluations exceeds 1000, the corresponding results are underlined. The results of the fastest algorithm among the three algorithms are written in bold font.

Matrix	α	`GJ1`		`GJ2`		`DE`	
`cell1`	0.2	4.00	(248)	<u>15.86</u>	(<u>1016</u>)	**3.55**	**(225)**
`cell1`	0.8	<u>15.88</u>	(<u>1016</u>)	<u>15.87</u>	(<u>1016</u>)	**3.60**	**(225)**
`TSOPF_RS_b9_c6`	0.2	0.93	(120)	1.93	(248)	**0.88**	**(113)**
`TSOPF_RS_b9_c6`	0.8	<u>7.82</u>	(<u>1016</u>)	1.92	(248)	**0.87**	**(113)**
`circuit_3`	0.2	2.36	(120)	**0.81**	**(24)**	1.36	(57)
`circuit_3`	0.8	2.21	(120)	**0.91**	**(24)**	1.33	(57)

5. Conclusion. In this work, we considered the DE formula to compute A^α. To utilize the DE formula, we proposed a method of truncating the infinite interval based on the analysis of the truncation error specialized for A^α. Based on the truncation error analysis, we presented two algorithms, including an adaptive quadrature algorithm. We analyzed the convergence rate of the DE formula for HPD matrices and found that when α is a non-unit fraction and $\kappa(A)$ is large, the DE formula converges faster than the GJ quadrature.

We performed five numerical tests. The results of the first two tests show that our algorithms achieve the required accuracy. The result of the third test shows that our estimate of the convergence rate is appropriate. The results of the fourth and fifth tests show that our algorithms work well, especially when the algorithms based on the GJ quadrature converge slowly.

In the future, we plan to consider the practical performance of our algorithm with parallel computing when applied to large practical problems.

Acknowledgments. We would like to thank Prof. Ken'ichiro Tanaka for his valuable comments on the convergence rate analysis of the DE formula. We are grateful to the two anonymous referees for the careful reading and the comments that substantially enhanced the quality of the manuscript. This work was supported by JSPS KAKENHI Grant Numbers 18J22501 and 18H05392. The authors would like to thank Enago (www.enago.jp) for the English language review.

Appendix A. On the behavior of $|f_{\mathrm{DE}}(x+\mathrm{i}y,\lambda)|$ as $x \to \pm\infty$. In this section, we consider the asymptotic behavior of f_{DE} defined in (3.1). We show that there exist constants c_l, c_r, l, and r such that for any y satisfying $|y| < d_0(\lambda) - \delta$ $(< \pi/2)$,

$$|f_{\mathrm{DE}}(x+\mathrm{i}y)| < c_l \exp\left(-\frac{\alpha\pi\cos(d_0(\lambda)-\delta)}{2}\sinh(|x|)\right)\cosh x \qquad (x < l), \tag{A.1}$$

$$|f_{\mathrm{DE}}(x+\mathrm{i}y)| < c_r \exp\left(-\frac{(1-\alpha)\pi\cos(d_0(\lambda)-\delta)}{2}\sinh x\right)\cosh x \quad (x > r). \tag{A.2}$$

From $|\cosh(x+\mathrm{i}y)| = \sqrt{\cosh^2 x + \cos^2 y - 1}$, it follows that

$$|\cosh(x+\mathrm{i}y)| \le \cosh x. \tag{A.3}$$

Besides, for $x < 0$,

$$\left|\exp\left(\frac{\alpha\pi}{2}\sinh(x+\mathrm{i}y)\right)\right| \le \exp\left(-\frac{\alpha\pi}{2}\sinh(|x|)\cos(d_0(\lambda)-\delta)\right). \tag{A.4}$$

Because f_{DE} is bounded in $\mathcal{D}_{d_0(\lambda)-\delta}$, there exists a constant c_1 that satisfies

$$\left|\exp\left(\frac{\pi}{2}\sinh(x+\mathrm{i}y)\right)+\lambda\right| > c_1. \tag{A.5}$$

By combining (A.3), (A.4), (A.5), we get (A.1).

Next, it is true that

$$\left|\exp\left(\frac{\alpha\pi}{2}\sinh(x+\mathrm{i}y)\right)\right| = \exp\left(\frac{\alpha\pi}{2}\sinh(x)\cos(y)\right). \tag{A.6}$$

In addition, there exists a constant c_2 such that for all $x > 0$ satisfying the condition $\exp(\pi\sinh x\cos(d_0(\lambda)-\delta)/2) > \lambda$, it holds that

$$\left|\exp\left(\frac{\pi}{2}\sinh(x+\mathrm{i}y)\right)+\lambda\right| \ge \exp\left(\frac{\pi}{2}\sinh x\cos y\right) - \lambda \ge c_2\exp\left(\frac{\pi}{2}\sinh x\cos y\right).$$

Hence,

$$\frac{|\exp(\alpha\pi\sinh(x+\mathrm{i}y)/2)|}{|\exp(\pi\sinh(x+\mathrm{i}y)/2)+\lambda|} \le \frac{1}{c_2}\exp\left(-\frac{(1-\alpha)\pi}{2}\sinh(x)\cos(y)\right) \le \frac{1}{c_2}\exp\left(-\frac{(1-\alpha)\pi\cos(d_0(\lambda)-\delta)}{2}\sinh(x)\right). \tag{A.7}$$

By combining (A.3), (A.6), and (A.7), we get (A.2).

Appendix B. A preconditioning technique of `GJ2` for HPD matrices. The study [3] considers the computation of fractional powers of a positive self-adjoint operator $\mathcal{L}$ by using the GJ quadrature for the integral

$$\mathcal{L}^{-\alpha} = \frac{2\sin(\alpha\pi)\tau^{1-\alpha}}{\pi}\int_{-1}^{1}(1-t)^{-\alpha}(1+t)^{\alpha-2}\left(\tau\frac{1-t}{1+t}\mathcal{I}+\mathcal{L}\right)\mathrm{d}t, \tag{B.1}$$

where $\alpha \in (0,1)$, $\mathcal{I}$ is the identity operator, and $\tau > 0$ is a parameter [3, Eq. (2)]. When $\mathcal{L}$ is an HPD matrix, we can rewrite (B.1) as (3.14), which is the integral used in `GJ2`, by substituting $\mathcal{L}^{-1} = A$ and $\tau = 1$.

One of the contributions of [3] is a method for selecting τ to reduce the error. The proposed selection is

$$\tau = \begin{cases} \tau^-(m) & (m < \bar{m}), \\ \tau^+(m) & (m \ge \bar{m}), \end{cases}$$

where m is the number of abscissa points,

$$\bar{m} = \frac{\alpha}{2\sqrt{2}}\left(\log(\mathrm{e}^2\kappa)\right)^{1/2}\kappa^{1/4} \qquad \left(\kappa = \frac{\mu_{\max}}{\mu_{\min}}\right),$$

$$\tau^-(m) = \mu_{\min}\left(\frac{\alpha}{2\mathrm{e}m}\right)^2\exp\left(2\mathrm{W}\left(\frac{4\mathrm{e}m^2}{\alpha^2}\right)\right),$$

$$\tau^+(m) = \left(-\frac{\alpha\mu_{\max}^{1/2}\log(\kappa)}{8m} + \sqrt{\left(\frac{\alpha\mu_{\max}^{1/2}\log(\kappa)}{8m}\right)^2 + (\mu_{\max}\mu_{\min})^{1/2}}\right)^2,$$

W is the Lambert W-function, and $\mu_{\max}$ and $\mu_{\min}$ are the maximum and the minimum eigenvalue of $\mathcal{L}$, respectively. One can apply the preconditioning technique to `GJ2`, which computes A^α for $\alpha \in (0,1)$, by substituting $\mu_{\max} = 1/\lambda_{\min}$, $\mu_{\min} = 1/\lambda_{\max}$, where $\lambda_{\max}$ and $\lambda_{\min}$ are the maximum and the minimum eigenvalue of A, and then computing $A^\alpha = \tau^{-\alpha}(\tau A)^\alpha$.

REFERENCES

[1] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, vol. 55 of National Bureau of Standards Applied Mathematics Series, U.S. Government Printing Office, Washington, D.C., 1964.

[2] L. Aceto and P. Novati, *Rational approximation to the fractional Laplacian operator in reaction-diffusion problems*, SIAM J. Sci. Comput., 39 (2017), pp. A214–A228.

[3] ———, *Rational approximations to fractional powers of self-adjoint positive operators*, Numer. Math., 143 (2019), pp. 1–16.

[4] S. AOKI, M. FUKUGITA, S. HASHIMOTO, K. I. ISHIKAWA, N. ISHIZUKA, Y. IWASAKI, K. KANAYA, T. KANEKO, Y. KURAMASHI, M. OKAWA, N. TSUTSUI, A. UKAWA, N. YAMADA, AND T. YOSHIÃĽ, *An exact algorithm for any-flavor lattice QCD with KogutâĂŞSusskind fermion*, Comput. Phys. Commun., 155 (2003), pp. 183–208.

[5] K. BURRAGE, N. HALE, AND D. KAY, *An efficient implicit FEM scheme for fractional-in-space reaction-diffusion equations*, SIAM J. Sci. Comput., 34 (2012), pp. A2145–A2172.

[6] J. R. CARDOSO, *Computation of the matrix pth root and its Fréchet derivative by integrals*, Electron. Trans. Numer. Anal., 39 (2012), pp. 414–436.
http://etna.ricam.oeaw.ac.at/vol.39.2012/pp414-436.dir/pp414-436.pdf

[7] T. A. DAVIS AND Y. HU, *The University of Florida sparse matrix collection*, ACM Trans. Math. Software, 38 (2011), Art. 1, 25 pages.

[8] M. FASI AND B. IANNAZZO, *Computing the weighted geometric mean of two large-scale matrices and its inverse times a vector*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 178–203.

[9] N. HALE, N. J. HIGHAM, AND L. N. TREFETHEN, *Computing $\mathbf{A}^\alpha$, $\log(\mathbf{A})$, and related matrix functions by contour integrals*, SIAM J. Numer. Anal., 46 (2008), pp. 2505–2523.

[10] N. J. HIGHAM, *Functions of Matrices*, SIAM, Philadelphia, 2008.

[11] N. J. HIGHAM AND L. LIN, *A Schur-Padé algorithm for fractional powers of a matrix*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1056–1078.

[12] ———, *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.

[13] B. IANNAZZO AND C. MANASSE, *A Schur logarithmic algorithm for fractional powers of matrices*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 794–813.

[14] M. MORI, *Discovery of the double exponential transformation and its developments*, Publ. Res. Inst. Math. Sci., 41 (2005), pp. 897–935.

[15] M. SUGIHARA, *Optimality of the double exponential formula—functional analysis approach*, Numer. Math., 75 (1997), pp. 379–395.

[16] B. J. SZEKERES AND F. IZSÁK, *Finite difference approximation of space-fractional diffusion problems: the matrix transformation method*, Comput. Math. Appl., 73 (2017), pp. 261–269.

[17] H. TAKAHASI AND M. MORI, *Double exponential formulas for numerical integration*, Publ. Res. Inst. Math. Sci., 9 (1973/74), pp. 721–741.

[18] L. N. TREFETHEN AND J. A. C. WEIDEMAN, *The exponentially convergent trapezoidal rule*, SIAM Rev., 56 (2014), pp. 385–458.