Calculating a function of a matrix with a real spectrum

P. Kubelík¹ · V. G. Kurbatov² · I. V. Kurbatova³

Received: 17 June 2021 / Accepted: 1 October 2021 / Published online: 9 October 2021
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract
Let $T$ be a square matrix with a real spectrum, and let $f$ be an analytic function. The problem of the approximate calculation of $f(T)$ is discussed. Applying the Schur triangular decomposition and the reordering, one can assume that $T$ is triangular and its diagonal entries $t_{ii}$ are arranged in increasing order. To avoid calculations using the differences $t_{ii} - t_{jj}$ with close (including equal) $t_{ii}$ and $t_{jj}$, it is proposed to represent $T$ in a block form and calculate the two main block diagonals using interpolating polynomials. The rest of the $f(T)$ entries can be calculated using the Parlett recurrence algorithm. It is also proposed to perform some scalar operations (such as the building of interpolating polynomials) with an enlarged number of significant decimal digits.

Keywords Matrix function · Polynomial interpolation · Divided differences · Reordering · Multiprecision arithmetic · Schur–Parlett algorithm

Mathematics Subject Classification (2010) MSC 65F60 · MSC 97N50

V. G. Kurbatov
kv51@inbox.ru

P. Kubelík
petr.kubelik@jh-inst.cas.cz

I. V. Kurbatova
irakurbatova@gmail.com

¹ J. Heyrovský Institute of Physical Chemistry, Academy of Sciences of the Czech Republic, Dolejškova 3, 18223 Prague 8, Czech Republic
² Department of System Analysis and Control, Voronezh State University 1, Universitetskaya Square, Voronezh 394018, Russia
³ Department of Software Development and Information Systems Administration, Voronezh State University 1, Universitetskaya Square, Voronezh 394018, Russia
1 Introduction

The problem [17, 22] of calculating analytic functions $f$ of a matrix $R$ is of constant interest since it has at least one extensive application: the solution of a linear differential equation $\dot{x} = Rx$ with a matrix coefficient $R$ can be expressed in terms of the function $e^{RT}$. We only mention some other applications: the cosine function [19, 20, 51], the power function [8, 23], the sign function [6, 22, 31], Green’s function [17, 36, 41], the gamma and beta functions [10, 29, 52], the Lambert function [28]. Usually, a function of a matrix can be calculated only approximately. The main objectives are (i) consideration of matrices of large size, (ii) obtaining the final result with high accuracy, and (iii) fast computations. In this paper, we propose a new idea which expands the possibilities of accurate calculations.

We present a numerical algorithm for the approximate calculation of an analytic function $f$ of a square matrix $R$ with a real spectrum. It is natural to consider this algorithm as a modification of the block Schur–Parlett algorithm [11, 18, 22]. A trivial example of a matrix with a real spectrum is a Hermitian matrix. This example is not very interesting from the point of view of this paper because it can be handle by more simple tools. A more substantive example of a matrix with a real spectrum is [56, section 12.69] a matrix of the form $F^{-1}G$, where $F$ and $G$ are Hermitian and $F$ is positive definite. The matrix $F^{-1}G$ arises when one transforms the linear pencil $\lambda \mapsto \lambda F - G$ to the form $\lambda \mapsto \lambda 1 - F^{-1}G$, where $1$ is the identity matrix. In its turn, linear pencils $\lambda \mapsto \lambda F - G$ with positive definite matrices $F$ and $G$ describe [40, 53, 55] RC-circuits and some other passive systems.

First, we construct the Schur decomposition, i.e., the representation $R = QTQ^H$, where $Q$ is unitary and $T$ is triangular. We recall that the Schur decomposition is computed with high backward stability by the $QR$-algorithm [18, 25, 56, 57]. We also note that the Schur decomposition is used as a beginning step in many numerical spectral algorithms. Clearly, one has

$$f(R) = Qf(T)Q^H.$$  

Thus, the problem of calculating $f(R)$ is reduced to calculating $f(T)$ for a triangular matrix $T$. (We recall that the diagonal entries of $T$ are the eigenvalues of both $T$ and $R$.) Then we apply an additional unitary similarity transformation that arranges the diagonal entries of $T$ in the increasing order (see Section 2).

The most known and widely used algorithm for calculating an analytic function $f$ of an upper triangular matrix $T$ is due to Parlett [46, 47], see also its expositions in [18, section 9.1.4] and [22, p. 85]. If the matrix $T$ has no multiple eigenvalues, the calculation of the entries $f_{ij}$ of the matrix $F = f(T)$ (where $T$ is upper triangular) can be carried out by the formulae

$$f_{ii} = f(t_{ii}),$$

$$f_{ij} = t_{ij} f_{jj} - f_{ii} + \sum_{k=i+1}^{j-1} t_{ik} f_{kj} - f_{ik} t_{kj}, \quad i < j. \quad (1)$$

It is seen from (1) that the diagonals of the matrix $F$ can be calculated one after another. The usage of the Schur decomposition and then application of formulae (1) is usually called [11, 18, 22] the Schur–Parlett algorithm.
Unfortunately, formulae (1) do not work if $T$ has multiple eigenvalues, i.e. $t_{ii} = t_{jj}$ for some $i \neq j$. A solution to this problem was also proposed by Parlett [46], see also [18, 22, 42]: the diagonal entries should be reordered so that the matrix can be represented in a block form with equal eigenvalues $t_{ii}$ included in a common blocks; after that, a block analogue [18, 22] of formulae (1) can be applied.

A similar problem arises when $T$ has very close (though unequal) diagonal entries $t_{ii}$ and $t_{jj}$. In this case, the calculation of the denominators $t_{jj} - t_{ii}$ in (1) causes a severe loss of accuracy. For a detailed discussion of this phenomenon, see [43, 44]. A partial solution to this problem was proposed in [11]. Close eigenvalues should be united into clusters. After that the diagonal entries of the triangular matrix should be reordered in such a way that the eigenvalues from the same cluster follow one after another. Thus, the matrix is divided into blocks so that the eigenvalues from the same cluster correspond the same block. Finally, the function $f$ of diagonal blocks (having small spectrum) can be calculated using the Taylor (or interpolating) polynomials; then the remaining blocks of $F$ can be calculated using the block version of (1); note that the absence of $t_{jj} - t_{ii}$ with close $t_{jj}$ and $t_{ii}$ in the neighboring blocks is ensured by the fact that the clusters are separated from each other. A modification of this algorithm that does not use a reordering was discussed in [35]. A version of the algorithm from [11] was proposed in [24]. This version is convenient when the Taylor polynomials cannot be used (e.g., because the derivatives of $f$ cannot be accurately computed); we especially emphasize that in [24], as in the present paper, higher precision arithmetic is used for delicate calculations.

In this paper, we consider the case when partitioning into well-separated clusters is impossible: the spectrum of $R$ uniformly (in the statistical sense) fills a segment of the real line. The main idea is as follows. Since we assume that the spectrum of $R$ is real, we can rearrange the diagonal entries of $T$ in increasing order. We again divide the spectrum of $T$ into clusters consisting of close numbers. Then we unite each pair of neighboring clusters and consider the double diagonal blocks that correspond to these pairs (thus each cluster except the two extreme ones is involved into two double diagonal blocks), see Section 3 for details. For example, let the spectrum of $T$ be contained in $[0, 3]$ and we form 3 clusters: eigenvalues that are contained in $[0, 1]$, eigenvalues that are contained in $[1, 2]$, and eigenvalues that are contained in $[2, 3]$; then we consider two (overlapping) double diagonal blocks that correspond to the eigenvalues from $[0, 2]$ and $[1, 3]$. We calculate the function $f$ of the double diagonal blocks (note that the diagonal blocks have relatively small spectra) using a polynomial approximation. Unfortunately, since the double diagonal blocks intersect, their common entries are calculated twice. In fact, we use a modification of the described idea to avoid such duplication. We calculate the rest of the entries of $F$ using (1); since the rest of the entries are located on the second usual (i.e., not double) block upper diagonal and higher, the differences $t_{jj} - t_{ii}$ in (1) with close $t_{jj}$ and $t_{ii}$ do not occur.

To calculate the action of $f$ on blocks with a small (real) spectrum, we use the Newton interpolating polynomials with the Chebyshev nodes taken as the points of interpolation. The construction of the Newton polynomial requires the calculation
of divided differences corresponding to close points of interpolation. This operation results in a loss of accuracy. We propose performing it with an enlarged number of decimal digits. No significant time is spent on such calculation (in comparison with matrix operations). By the way, the experiments show that the longest operation is the calculation according to formula (1) of the entries that are far from the main diagonal.

The paper is organized as follows. In Section 2, we recall the Schur decomposition and the reordering, and introduce some notation. In Section 3, we begin the description of our algorithm, represent the matrix in the block form, and describe the simplified version of the algorithm. In Section 4, we discuss the calculation of blocks located on the main diagonal. In Section 5, we recall the Paterson–Stockmeyer algorithm that accelerates the substitution of a matrix into a polynomial. Section 6 describes an algorithm for calculating the second block diagonal. In Section 7, we list the sequence of execution of individual parts of our algorithm. In Section 8, numerical experiments are presented. In Section 9, we give some concluding remarks.

2 Reordering and other preliminaries

In this section, we describe the preliminary stage of our algorithm.

Let $N$ be a natural number and $R$ be a complex matrix of the size $N \times N$ with a real spectrum, and $f$ be a function analytic in a neighborhood of the spectrum of $R$. In this paper, we present an algorithm of the approximate calculation of the matrix $f(R)$. Our algorithm is interesting in the case when the spectrum $\sigma(R)$ of $R$ fills (in the statistical sense) a segment of the real line $\mathbb{R}$ without essential gaps. The algorithm can be modified to the case when the spectrum $\sigma(R)$ lies on a continuous curve in $\mathbb{C}$ without self-intersections.

First, we compute the Schur decomposition $R = QTQ^H$, where the matrix $Q$ is unitary and $T$ is triangular. We recall that the Schur decomposition is performed with high backward stability by the $QR$-algorithm [18, 25, 56, 57]. We recall that $f(R) = Qf(T)Q^H$. Thus, the problem is reduced to the calculation of $f(T)$.

Our algorithm needs that the diagonal entries of $T$ (the eigenvalues of both $T$ and $R$) be arranged in order. Usually, the $QR$-algorithm arranges the diagonal entries of $T$ in an almost descending order of modules. For our notation, it is convenient to have the increasing order. Therefore, we apply the $QR$-algorithm to the matrix $R_1 = -R + \alpha \mathbf{1}$, where $\mathbf{1}$ is the identity matrix and $\alpha \in \mathbb{R}$ is greater than the largest eigenvalue of $R$. For example, one may take for $\alpha$ a number that is greater than the norm $\|R\|_{1 \rightarrow 1} = \max_j \sum_{i=1}^m |r_{ij}|$ of $R$. We calculate the decomposition $R_1 = QT_1Q^H$, where $Q$ is unitary and $T_1$ is triangular. Then we set $T = T_1 + \alpha \mathbf{1}$ and arrive at the Schur decomposition $R = QTQ^H$ of the matrix $R$, in which the diagonal entries $t_{ii}$ of $T$ are arranged in almost increasing order.

Since the order of the diagonal entries $t_{ii}$ may be not fully increasing, we correct it using the procedure of reordering (see, e.g., [7], [26, p. 49]), i.e., we apply several unitary similarity transformations $T' = UTU^H$, where $U$ is unitary, that permutes two adjacent diagonal entries $t_{i-1,i-1}$ and $t_{i,i}$ but preserves the triangular structure of
For completeness, we describe $U$: the entries of $U$ coincide with the entries of the identity matrix except for the block

$$
\begin{pmatrix}
u_{i-1,i-1} & u_{i-1,i} \\ u_{i,i-1} & u_{i,i}
\end{pmatrix} = \begin{pmatrix}
  e^{-i\alpha} \cos \vartheta - \sin \vartheta \\
  \sin \vartheta & e^{i\alpha} \cos \vartheta
\end{pmatrix},
$$

where the parameters $\alpha$ and $\vartheta$ are real. The inverse (conjugate transpose) of this block has the form

$$
\begin{pmatrix}
e^{-i\alpha} \cos \vartheta - \sin \vartheta \\
  \sin \vartheta & e^{i\alpha} \cos \vartheta
\end{pmatrix}^{-1} = \begin{pmatrix}
  e^{i\alpha} \cos \vartheta & \sin \vartheta \\
  -\sin \vartheta & e^{-i\alpha} \cos \vartheta
\end{pmatrix}.
$$

The first admissible pair of the parameters $\alpha$ and $\vartheta$ are defined from the formulae

$$
\alpha_1 = \arg t_{i-1,i-1} - \arg (t_{i-1,i-1} - t_{i,i}),
\cos \vartheta_1 = \frac{|t_{i-1,i-1}|}{\sqrt{|t_{i-1,i-1} - t_{i,i}|^2 + |t_{i-1,i}|^2}},
\sin \vartheta_1 = \frac{|t_{i-1,i-1} - t_{i,i}|}{\sqrt{|t_{i-1,i-1} - t_{i,i}|^2 + |t_{i-1,i}|^2}}.
$$

The second admissible pair is

$$
\alpha_2 = \alpha_1 + \pi, \quad \vartheta_2 = -\vartheta_1.
$$

The two possible values of $\alpha$ and $\vartheta$ lead only to different signs in the entries of $T' = UTU^H$. Therefore, it does not matter which pair to use.

3 Partitioning

We recall that the diagonal entries $t_{ii}$ of the triangular matrix $T$ of the size $N \times N$ are real and arranged in ascending order.

Let $a$ be the minimum of the diagonal entries $t_{ii}$, and let $b$ be greater than the maximum of the diagonal entries $t_{ii}$. Thus, the spectrum of $T$ is contained in the half-open interval $[a, b)$. We choose the parameter $\rho > 0$. The algorithm depends on this parameter. In our numerical examples in Section 8, $\rho = 2$ or (sometimes) $\rho = 1$. The meaning of the parameter $\rho$ is as follows: we consider the loss of accuracy to be admissible if formula (1) is applied only in the case $|t_{jj} - t_{ii}| \geq \rho$. We postpone discussing the choice of $\rho$ until Section 9.

We split $[a, b)$ into parts of length $\rho$:

$$
[a, b) \subseteq [a, a + \rho) \cup [a + \rho, a + 2\rho) \cup [a + 2\rho, a + 3\rho) \cup \cdots \cup [a + (n-1)\rho, a + n\rho).
$$

Thereby, we assume that $b \in [a + (n - 1)\rho, a + n\rho]$. We divide the set $t_{11}, t_{22}, \ldots, t_{NN}$ of the diagonal entries of $T$ into $n$ clusters:

1The usage of a half-open interval is due only to the uniformity of the notation. In practice, it is more convenient to use the closed interval $[a, b]$. Springer
the first cluster consists of diagonal entries such that
\[ t_{1,1}, t_{2,2}, \ldots, t_{k_1 k_1} \in [a, a + \rho); \]
the second cluster consists of diagonal entries such that
\[ t_{k_1+1, k_1+1}, t_{k_1+2, k_1+2}, \ldots, t_{k_2 k_2} \in [a + \rho, a + 2\rho); \]
...;
the \( n \)th cluster consists of diagonal entries such that
\[ t_{k_{n-1}+1, k_{n-1}+1}, t_{k_{n-1}+2, k_{n-1}+2}, \ldots, t_{k_n k_n} \in [a + (n - 1)\rho, a + n\rho). \]

We divide the matrix \( T \) into blocks according to this clusterization:
\[
T = \begin{pmatrix}
T_{1,1} & T_{1,2} & \cdots & T_{1,n-1} & T_{1,n} \\
0 & T_{2,2} & \cdots & T_{2,n-1} & T_{2,n} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & T_{n-1,n-1} & T_{n-1,n} \\
0 & 0 & \cdots & 0 & T_{n,n}
\end{pmatrix}. \tag{2}
\]

Now we come to the main idea of the paper. To begin with, we describe its simplest variant. We fill the first two diagonals of the block matrix \( F \) by applying the function \( f \) to the double blocks
\[
\begin{pmatrix}
T_{1,1} & T_{1,2} \\
0 & T_{2,2}
\end{pmatrix}, \quad \begin{pmatrix}
T_{2,2} & T_{2,3} \\
0 & T_{3,3}
\end{pmatrix}, \quad \ldots, \quad \begin{pmatrix}
T_{n-1,n-1} & T_{n-1,n} \\
0 & T_{n,n}
\end{pmatrix}. \tag{3}
\]
Since these blocks have relatively narrow spectra, the result of the application of the function \( f \) to the double blocks can be approximately calculated, e.g., by substituting them into the interpolating polynomial of \( f \). (Unfortunately, the diagonal blocks \( F_{2,2}, \ldots, F_{n-1,n-1} \) of \( F = f(T) \) are calculated twice. Of course, this leads to a loss of time. Below we eliminate this duplication.) Thus, we calculate two main diagonals of the block matrix \( f(T) \). After that, the remaining entries of \( F \) are calculated by formula (1). This is the main idea of the paper. In the rest of the paper, we discuss its improvements that work faster and more accurate. Namely, we describe how to calculate the two main block diagonals of \( F \) separately.

The idea of calculating first the entries of two main block diagonals and then the other entries was used earlier in [42], but for a somewhat different purpose, namely, for applying \( f \) to the two-diagonal Opitz matrix [45], which allows one to calculate the divided differences of \( f \) in order to then substitute them into the Newton interpolating polynomial for calculating \( f(R) \). We use an analogous idea to calculate \( f(R) \) directly.

4 Calculation of \( F_{k,k} \)

In this section, we discuss the calculating the function \( f \) of the diagonal blocks of (2). By construction, the spectrum of any block \( T_{k,k} \) is real and contained in a segment of length \( \rho \). This circumstance simplifies the problem.

We recall the following well-known statement.
**Proposition 1** ([22, Theorem 1.13]) Let a block matrix $T$ have form (2) and the matrix $F = f(T)$ be presented in the same block form:

$$
F = \begin{pmatrix}
F_{1,1} & F_{1,2} & \ldots & F_{1,n-1} & F_{1,n} \\
0 & F_{2,2} & \ldots & F_{2,n-1} & F_{2,n} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \ldots & F_{n-1,n-1} & F_{n-1,n} \\
0 & 0 & \ldots & 0 & F_{n,n}
\end{pmatrix}.
$$

(4)

Then

$$
F_{k,k} = f(T_{k,k}), \quad k = 1, 2, \ldots, n.
$$

We fix $k = 1, 2, \ldots, n$. Let $[\alpha_k, \beta_k] \subseteq \mathbb{R}$ be a segment that contains the spectrum of the block $T_{k,k}$ (e.g., $\alpha_k$ is the minimal eigenvalue of $T_{k,k}$ and $\beta_k$ is the maximal one). Clearly, $\beta_k - \alpha_k \leq \rho$. We denote by $\gamma_k$ the center of the segment $[\alpha_k, \beta_k]$, i.e., the point $\frac{\alpha_k + \beta_k}{2}$.

We denote briefly by $A$ the matrix $T_{k,k} - \gamma_k I$, where $I$ is the identity matrix, and we denote by $g$ the shifted function $f$:

$$
g(\lambda) = f(\lambda + \gamma_k).
$$

Clearly, $f(T_{k,k}) = g(A)$ and the spectrum of $A$ is contained in

$$
[\hat{\alpha}, \hat{\beta}] = [-\delta_k, \delta_k],
$$

where $\delta_k = \frac{\beta_k - \alpha_k}{2}$.

We denote by $m_k$ the degree of a polynomial $p$, which we will use for the approximation of $g$ (the polynomial $p$ has $m = m_k + 1$ coefficients). In our numerical experiments in Section 8, we take $m_k = 15$. We prefer to calculate $g(A)$ by means of an interpolating polynomial. Since the spectrum of $A$ is real, it is reasonable to use as the points of interpolation the zeroes of the Chebyshev polynomial of the first kind $T_m(x) = \cos(m \arccos x)$ scaled to the segment $[\hat{\alpha}, \hat{\beta}]$, which contains the spectrum of $A$, i.e., the points

$$
\lambda_i = \frac{1}{2} (\hat{\alpha} + \hat{\beta}) + \frac{1}{2} (\hat{\beta} - \hat{\alpha}) \cos \left( \frac{2i - 1}{2m} \pi \right), \quad i = 1, 2, \ldots, m.
$$

(5)

| Degree $m$ of $p_m$ | 7 | 13 | 14 | 15 |
|-------------------|---|----|----|----|
| $\max_{\lambda \in [-1,1]} |e^{\lambda} - p_m(\lambda)|$ | $2.07 \times 10^{-7}$ | $1.46 \times 10^{-15}$ | $4.82 \times 10^{-17}$ | $1.51 \times 10^{-18}$ |
| degree $m$ of $p_m$ | 7 | 11 | 12 | 15 |
| $\max_{\lambda \in [-\frac{1}{2}, \frac{1}{2}]} |e^{\lambda} - p_m(\lambda)|$ | $8.02 \times 10^{-10}$ | $2.60 \times 10^{-16}$ | $4.98 \times 10^{-18}$ | $2.30 \times 10^{-23}$ |

Table 1: The value $\max_{\lambda \in [\hat{\alpha}, \hat{\beta}]} |e^{\lambda} - p_m(\lambda)|$ of the approximation of the function $\lambda \mapsto e^{\lambda}$ on $[\hat{\alpha}, \hat{\beta}]$ by the interpolating polynomial $p_m$ of degree $m$ with the scaled Chebyshev nodes taken as the points of interpolation.
We call points (5) the Chebyshev nodes of degree \( m \) on the segment \([\hat{\alpha}, \hat{\beta}]\). See, e.g., [4, 13, 54] for a discussion of why the choice of Chebyshev nodes is good. We mention that the Chebyshev nodes of degree 16 on \([-1, 1]\) are the numbers

\[
\pm \sqrt{2 - \sqrt{2 + \sqrt{2 + \sqrt{2}}}}, \quad \pm \sqrt{2 - \sqrt{2 + \sqrt{2 - \sqrt{2}}}}, \\
\pm \sqrt{2 - \sqrt{2 - \sqrt{2 + \sqrt{2}}}}, \quad \pm \sqrt{2 - \sqrt{2 - \sqrt{2 - \sqrt{2}}}}, \\
\pm \sqrt{2 + \sqrt{2 + \sqrt{2 + \sqrt{2}}}}, \quad \pm \sqrt{2 + \sqrt{2 + \sqrt{2 - \sqrt{2}}}}, \\
\pm \sqrt{2 + \sqrt{2 + \sqrt{2 - \sqrt{2}}}}, \quad \pm \sqrt{2 + \sqrt{2 + \sqrt{2 - \sqrt{2}}}}.
\]

Thus, they can be easily calculated with any desirable accuracy.

We assume that the entries of the matrix \( T \) are known with the standard accuracy (IEEE double-precision binary floating-point format), which is about 16 significant decimal digits. Therefore, there is no need to know other parameters with greater accuracy if calculations involve \( T \). In particular, knowing the interpolating polynomial (i.e., the coefficients of the interpolating polynomial) with higher accuracy makes no sense. Furthermore, the approximation of the function \( g \) by an interpolating polynomial \( p \) with higher accuracy is also senseless. Moreover, the powers \( T^k \) are calculated with a fewer number of significant digits than \( T \) itself. Therefore, the leading coefficients of the interpolating polynomial may be calculated with lower accuracy.

The choice of the parameter \( \rho \) depends on the function \( f \). In this paper, we restrict ourselves to discussing the choice of \( \rho \) for the simplest function \( \lambda \mapsto e^{\lambda} \).

Numerical experiments on the approximation of the function \( \lambda \mapsto e^{\lambda} \) on the segment \([-\rho/2, \rho/2]\) by the interpolating polynomial \( p \) of degree \( m \) generated by the Chebyshev nodes are given in Table 1. One can see from Table 1 that for \( \rho = 2 \), the polynomial of the 14th degree provides about 16 significant decimal digits; and for \( \rho = 1 \), the polynomial of the 12th degree provides about 16 significant decimal digits. At the same time, the Paterson–Stockmeyer algorithm we use gives the best economy for the degree of a polynomial of the form \( l^2 - 1 \); the closest degree of the form \( l^2 - 1 \) to both the 14th and the 12th is the 15th. Therefore, we prefer \( \rho = 2 \).

Both the Lagrange and Newton\(^2\) forms of the interpolating polynomial contain differences \( \lambda_i - \lambda_j \), where \( \lambda_i \) are the points of interpolation. Therefore the approximate calculation of the interpolating polynomial accompanies by a loss of accuracy (for the Chebyshev nodes \( \lambda_i \), the smallest of the numbers \( |\lambda_i - \lambda_j| \) is about \( 0.098 \cdot (\beta_k - \alpha_k) \)). We construct the interpolating polynomial in the Newton form because (according to numerical experiments) the Newton form gives 2 additional significant decimal digits of accuracy in comparison with the Lagrange form.

\(^2\)We recall the definition of the Newton form of the interpolating polynomial below in formula (12).
Table 2  The effect of the number of significant decimal digits (in the values of points of interpolation) on the error in the coefficients of the interpolating polynomial (of degree 15 with the Chebyshev nodes taken as the points of interpolation) of the function $\lambda \mapsto e^\lambda$

|          | [-1, 1] |          |          |
|----------|---------|----------|----------|
|          | significant decimal digits | absolute error in the coefficients | relative error in the coefficients |
| on [-1, 1] | 23      | $9.82 \times 10^{-17}$ | $3.47 \times 10^{-7}$ |
|          | 30      | $9.82 \times 10^{-24}$ | $3.47 \times 10^{-14}$ |
|          | 33      | $9.82 \times 10^{-27}$ | $3.47 \times 10^{-17}$ |
| on [-1/2, 1/2] | 26      | $6.08 \times 10^{-17}$ | $4.80 \times 10^{-6}$ |
|          | 36      | $6.08 \times 10^{-27}$ | $4.80 \times 10^{-16}$ |
|          | 37      | $6.08 \times 10^{-28}$ | $4.80 \times 10^{-17}$ |

To achieve the accuracy in interpolating polynomials in 16 significant decimal digits, we propose calculating them with an enlarged number of decimal digits. Since the calculation of the interpolating polynomial is a scalar (not matrix) operation, the usage of the enlarged number of decimal digits does not lead to an essential loss of time. Numerical experiments presented in Table 2 (for the approximation of the exponential function by the interpolating polynomial with 16 Chebyshev nodes) show that for $\beta_k - \alpha_k = 2$, in order to provide a relative error about $10^{-16}$ in all coefficients of the interpolating polynomial, it is necessary to use 33 significant decimal digits in the values of points of interpolation $\lambda_i$ (if $\beta_k - \alpha_k = 1$, to provide a relative error about $10^{-16}$ in coefficients, one must use 37 significant decimal digits). Nevertheless, the largest coefficient (of the interpolating polynomial for the exponential function) is about 1 and the smallest one is about $7.76 \times 10^{-13}$ for 16 Chebyshev nodes and $\rho = 2$ (is about $7.67 \times 10^{-13}$ for $\rho = 1$). Table 2 shows that if one wants the absolute error in coefficients less than $10^{-16}$, it is enough to use 23 decimal digits in the case $\rho = 2$ (and 26 decimal digits in the case $\rho = 1$). In our numerical experiments (Section 8), we use 30 decimal digits in $\lambda_k$ for $\rho = 2$ (and 36 for $\rho = 1$); such accuracy is needed for the calculation of the second block diagonal, see Section 6.

Some estimates of $\|p_k(A) - g(A)\|$ are given in [38].

5 The Paterson–Stockmeyer algorithm

Numerical experiments show that the accuracy provided by substituting a matrix directly into the Newton interpolating polynomial is insignificantly higher than the accuracy of substituting it into the same polynomial after expanding in powers of $\lambda$. Therefore, we substitute matrices into interpolating polynomials represented in the form

$$p(\lambda) = c_0 + c_1\lambda + c_2\lambda^2 + c_3\lambda^3 + c_4\lambda^4 + c_5\lambda^5 + \ldots + c_m\lambda^m$$

3The “worse” accuracy for the smaller segment is explained by the fact that on a smaller segment the coefficients of the interpolating polynomial are closer to zero.
because when a polynomial is expanded in powers of $\lambda$, the Paterson–Stockmeyer algorithm [48], [18, section 9.2.4] can be used. This algorithm allows one to reduce the number of matrix multiplications and thus speed up calculations. We describe here only one specific case of this algorithm that we use in our numerical experiments in Section 8. See also modifications of the Paterson–Stockmeyer algorithm in [5, 50].

Let $p$ be a polynomial of degree 15 expanded in powers of $\lambda$:

$$p(\lambda) = c_0 + c_1 \lambda + c_2 \lambda^2 + c_3 \lambda^3 + c_4 \lambda^4 + c_5 \lambda^5 + \ldots + c_{15} \lambda^{15}.$$  

It is required to substitute into $p$ a given square matrix $A$, i.e., to calculate

$$p(A) = c_0 1 + c_1 A + c_2 A^2 + c_3 A^3 + c_4 A^4 + c_5 A^5 + \ldots + c_{15} A^{15}.$$  

In this paper, by the Paterson–Stockmeyer algorithm we mean the following procedure. First, we calculate the powers $A^2, A^3, A^4$. This stage requires 3 matrix multiplications.

Next, we calculate the new coefficients (no matrix multiplications are required)

$$c_0^{(1)} = c_0 1 + c_1 A + c_2 A^2 + c_3 A^3,$$

$$c_1^{(1)} = c_4 1 + c_5 A + c_6 A^2 + c_7 A^3,$$

$$c_2^{(1)} = c_8 1 + c_9 A + c_{10} A^2 + c_{11} A^3,$$

$$c_3^{(1)} = c_{12} 1 + c_{13} A + c_{14} A^2 + c_{15} A^3.$$  

Now $p(A)$ can be represented in powers of $A^4$:

$$p(A) = c_0^{(1)} + c_1^{(1)} A^4 + c_2^{(1)} (A^4)^2 + c_3^{(1)} (A^4)^3.$$  

For the final calculation of $p(A)$, we apply the Horner method:

$$p(A) = c_0^{(1)} + \left( c_1^{(1)} + (c_2^{(1)} + (c_3^{(1)} A^4) A^4) \right) A^4.$$  

The last calculation requires 3 additional matrix multiplications. In total, we need 6 matrix multiplications for calculating a polynomial of degree 15.

### 6 Calculation of $F_{k,k+1}$

Let $U \subset \mathbb{C}$ be an open set, and let $f : U \to \mathbb{C}$ be an analytic function. The divided difference of $f$ is [14, 30] the function $f^{[1]} : U \times U \to \mathbb{C}$ defined by the formula

$$f^{[1]}(\lambda, \mu) = \begin{cases} 
\frac{f(\lambda) - f(\mu)}{\lambda - \mu}, & \text{if } \lambda \neq \mu, \\
f'(\lambda), & \text{if } \lambda = \mu.
\end{cases}$$
The simplest examples of divided differences are

\[ v^{[1]}_1(\lambda, \mu) = 1 \quad \text{for } v_1(\lambda) = \lambda, \]
\[ v^{[1]}_n(\lambda, \mu) = \lambda^{n-1} + \lambda^{n-2} \mu + \cdots + \mu^{n-1} \quad \text{for } v_n(\lambda) = \lambda^n, \]
\[ v^{[1]}_{1/2}(\lambda, \mu) = \frac{1}{\sqrt{\lambda + \mu}}, \quad \text{where } v_{1/2}(\lambda) = \sqrt{\lambda}, \]
\[ r^{[1]}_1(\lambda, \mu) = \frac{1}{(\lambda - \lambda_0)(\mu - \mu_0)} \quad \text{for } r_1(\lambda) = \frac{1}{\lambda_0 - \lambda}. \]

**Proposition 2** (see, e.g., [39, Proposition 44]) *The function* \( f^{[1]} \) *is analytic in* \( U \times U \).

The Taylor series for the divided difference of an analytic function \( f \) at a point \((\lambda_0, \mu_0)\) has the form

\[
f^{[1]}(\lambda, \mu) = \sum_{n=0}^{\infty} \frac{f^{(n+1)}(\lambda_0)}{(n+1)!} v^{[1]}_{n+1}(\lambda - \lambda_0, \mu - \mu_0),
\]

where \( v_n(\lambda) = \lambda^n \). In particular, for \( \exp_t(\lambda) = e^{\lambda t} \) at the point \((0, 0)\) the expansion is

\[
\exp^{[1]}_t(\lambda, \mu) = \sum_{n=0}^{\infty} \frac{t^n}{(n+1)!} v^{[1]}_{n+1}(\lambda, \mu) = \sum_{n=0}^{\infty} \frac{t^n}{(n+1)!} \sum_{i=0}^{n} \lambda^{n-i} \mu^i.
\]

Let \( A, B, \) and \( H \) be matrices of the sizes \( \nu \times \nu, \kappa \times \kappa, \) and \( \nu \times \kappa \) respectively. We denote by \( \sigma(A) \) and \( \sigma(B) \) the spectra of \( A \) and \( B \). Let \( f \) be an analytic function defined on an open set \( U \subset \mathbb{C} \) such that \( \sigma(A) \cup \sigma(B) \subset U \). We call the divided difference of \( f \) applied to \( A \) and \( B \) at \( H \) the matrix

\[
f^{[1]}(A, B) \diamond H = \frac{1}{(2\pi i)^2} \int_{\Gamma_2} \int_{\Gamma_1} f^{[1]}(\lambda, \mu)(\lambda I - A)^{-1} H(\mu I - B)^{-1} d\lambda d\mu,
\]

where \( \Gamma_1 \) surrounds \( \sigma(A) \) and \( \Gamma_2 \) surrounds \( \sigma(B) \) in the counterclockwise direction, and \( f \) is analytic inside both \( \Gamma_1 \) and \( \Gamma_2 \). For more about general functions of two matrices, see [15, 16, 33, 34, 39].

We mention the following statement (which we do not use explicitly).

**Proposition 3** ([32], [39, Theorem 45], [37, Theorem 17]) *Under the above assumptions*

\[
f^{[1]}(A, B) \diamond H = \frac{1}{2\pi i} \int_{\Gamma} f(\lambda)(\lambda I - A)^{-1} H(\mu I - B)^{-1} d\lambda,
\]

where \( \Gamma \) surrounds \( \sigma(A) \cup \sigma(B) \) in the counterclockwise direction.

Our calculation of \( F_{k,k+1} \) is based on the following proposition.

**Proposition 4** [9, 12, 37] *The blocks* \( F_{k,k+1} \) *of the second diagonal of block matrix* \( (4) \) *can be represented in the form*

\[
F_{k,k+1} = f^{[1]}(T_{k,k}, T_{k+1,k+1}) \diamond T_{k,k+1}.
\]
In this section, we describe an algorithm for approximate calculation of (7).
We fix \( k = 1, 2, \ldots, n - 1 \). We denote by \([\alpha_1, \beta_1]\) and \([\alpha_2, \beta_2]\) segments that contain the spectra \( \sigma(T_{k,k}) \) and \( \sigma(T_{k+1,k+1}) \) respectively. We introduce the notation \( \gamma_1 = \frac{\alpha_1 + \beta_1}{2} \) and \( \gamma_2 = \frac{\alpha_2 + \beta_2}{2} \). We denote the Chebyshev nodes referred to the segments
\[
[\hat{\alpha}_1, \hat{\beta}_1] = [\alpha_1 - \gamma_1, \beta_1 - \gamma_1] \text{ and } [\hat{\alpha}_2, \hat{\beta}_2] = [\alpha_2 - \gamma_2, \beta_2 - \gamma_2]
\]
by \( \lambda_1, \lambda_2, \ldots, \lambda_{m_1} \) and \( \mu_1, \mu_2, \ldots, \mu_{m_2} \) respectively.

We consider the shifted function
\[
h(\lambda, \mu) = f^{[1]}(\lambda + \gamma_1, \mu + \gamma_2).
\]
We set
\[
A = T_{k,k} - \gamma_1 1, \quad B = T_{k+1,k+1} - \gamma_2 1, \quad H = T_{k,k+1}.
\]
By (7) and the definition,
\[
F_{k,k+1} = f^{[1]}(T_{k,k}, T_{k+1,k+1}) \circ T_{k,k+1}
\]
\[
= \frac{1}{(2\pi i)^2} \int_{\Gamma_2} \int_{\Gamma_1} f^{[1]}(\lambda, \mu)(\lambda 1 - T_{k,k})^{-1} T_{k,k+1}(\mu 1 - T_{k+1,k+1})^{-1} d\lambda d\mu
\]
\[
= \frac{1}{(2\pi i)^2} \int_{\Gamma_2} \int_{\Gamma_1} f^{[1]}(\lambda, \mu)(\lambda 1 - T_{k,k})^{-1} H(\mu 1 - T_{k+1,k+1})^{-1} d\lambda d\mu.
\]
We make the change of variable \( \lambda + \gamma_1 \to \lambda \) in the internal integral:
\[
\int_{\Gamma_1} f^{[1]}(\lambda, \mu)(\lambda 1 - T_{k,k})^{-1} d\lambda = \int_{\Delta_1} f^{[1]}(\lambda + \gamma_1, \mu)(\lambda 1 + \gamma_1 1 - T_{k,k})^{-1} d\lambda
\]
\[
= \int_{\Delta_1} f^{[1]}(\lambda + \gamma_1, \mu)(\lambda 1 - A)^{-1} d\lambda,
\]
where \( \Delta_1 = \Gamma_1 - \gamma_1 \) surrounds \( \sigma(A) \). We make the similar change of variable \( \mu + \gamma_2 \to \mu \) in the external integral:
\[
\int_{\Delta_2} \int_{\Delta_1} f^{[1]}(\lambda + \gamma_1, \mu + \gamma_2)(\lambda 1 - A)^{-1} H(\mu 1 + \gamma_2 1 - T_{k+1,k+1})^{-1} d\lambda d\mu
\]
\[
= \int_{\Delta_2} \int_{\Delta_1} f^{[1]}(\lambda + \gamma_1, \mu + \gamma_2)(\lambda 1 - A)^{-1} H(\mu 1 - B)^{-1} d\lambda d\mu
\]
\[
= \int_{\Delta_2} \int_{\Delta_1} h(\lambda, \mu)(\lambda 1 - A)^{-1} H(\mu 1 - B)^{-1} d\lambda d\mu,
\]
where \( \Delta_2 = \Gamma_2 - \gamma_2 \) surrounds \( \sigma(B) \). Thus,
\[
F_{k,k+1} = h(A, B) \circ H.
\]
(8)
We construct the interpolating polynomial
\[ p(\lambda, \mu) = \sum_{i=0}^{m_1-1} \sum_{j=0}^{m_2-1} c_{ij} \lambda^i \mu^j \]  
(9)

that agrees with \( h \) at the points \((\lambda_i, \mu_j)\), i.e.,
\[ p(\lambda_i, \mu_j) = h(\lambda_i, \mu_j), \quad i = 1, 2, \ldots, m_1; \quad j = 1, 2, \ldots, m_2. \]  
(10)

But first, we recall some notation and terminology connected with the Newton interpolating polynomial.

Let be given distinct \(4\) points of interpolation \(z_1, z_2, \ldots, z_m\) (complex numbers) and the family \(f_1, f_2, \ldots, f_m\) of complex numbers. The \textit{divided differences} \([14, 30]\) of the family \(f_1, f_2, \ldots, f_m\) (with respect to the points \(z_1, z_2, \ldots, z_m\)) are the family \(f_i^{j+1}\) defined by the recurrence relations
\[
\begin{align*}
  f_i^1 &= f_i, \\
  f_i^{j+1} &= \frac{f_i^{j+1} - f_i^j}{z_{i+1} - z_i}, \\
  & \quad \ldots \ldots \ldots \ldots \\
  f_i^{j+1} &= \frac{f_i^{j+1} - f_i^{j-1}}{z_{i+j} - z_i}.
\end{align*}
\]

It is convenient to arrange the divided differences in the table

\[
\begin{array}{cccccc}
  f_1^1 & f_2^2 & \cdots & f_i^i & \cdots & f_m^{m-1} & f_m^m \\
  f_1^2 & f_2^3 & \cdots & f_i^{i+1} & \cdots & f_m^{m-1} & f_m^m \\
  f_1^3 & f_2^4 & \cdots & f_i^{i+2} & \cdots & f_m^{m-1} & f_m^m \\
  \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
  f_1^{m-1} & f_2^m & \cdots & f_i^{m-1} & \cdots & f_m^{m-1} & f_m^m \\
  f_1^m & f_2^m & \cdots & f_i^m & \cdots & f_m^m & \cdots
\end{array}
\]  
(11)

We note that the construction of the Newton interpolating polynomial (12) requires only the first column of this table. But to calculate the first column, all the others are needed.

We recall \([14, 30]\) that for any distinct points of interpolation \(z_1, z_2, \ldots, z_m\) and values \(f_1, f_2, \ldots, f_m\), the interpolating polynomial\(^5\) can be represented in the Newton form (also called the Newton interpolating polynomial)
\[
\begin{align*}
  p(z) &= f_1^1 + f_1^2 (z - z_1) + f_1^3 (z - z_1)(z - z_2) \\
  & \quad + f_1^4 (z - z_1)(z - z_2)(z - z_3) + \ldots \\
  & \quad + f_1^{m-1} (z - z_1)(z - z_2) \ldots (z - z_{m-2}) \\
  & \quad + f_1^m (z - z_1)(z - z_2) \ldots (z - z_{m-1}).
\end{align*}
\]  
(12)

Now we begin the construction of (9).

\(^4\)In this paper, we do not use interpolating polynomials with multiple points of interpolation.

\(^5\)An \textit{interpolation polynomial} is the polynomial \(p\) of degree \(m = m - 1\) satisfying the conditions \(p(z_1) = f_1, p(z_2) = f_2, \ldots, p(z_m) = f_m\).
For each $j = 1, 2, \ldots, m_2$, we consider the function

$$g_j(\lambda) = h(\lambda, \mu_j).$$

Then we construct the interpolating polynomial $b_j$ of degree $m = m_1 - 1$ that agrees with the function $g_j$ at the points $\lambda_1, \lambda_2, \ldots, \lambda_{m_1}$, i.e.,

$$b_j(\lambda_i) = g_j(\lambda_i) = h(\lambda_i, \mu_j), \quad i = 1, 2, \ldots, m_1. \quad (13)$$

We represent the interpolating polynomials $b_j$ in the Newton form

$$b_j(\lambda) = d_{0,j} + d_{1,j}(\lambda - \lambda_1) + d_{2,j}(\lambda - \lambda_1)(\lambda - \lambda_2) + \ldots + d_{m_1-1,j}(\lambda - \lambda_1)\ldots(\lambda - \lambda_{m_1-1}), \quad (14)$$

where $d_{0,j}, d_{1,j}, \ldots, d_{m_1-1,j}$ are the divided differences of the family $h(\lambda_1, \mu_j), h(\lambda_2, \mu_j), \ldots, h(\lambda_{m_1-1}, \mu_j)$.

Then for each $i = 0, 1, \ldots, m_1 - 1$, we consider the family

$$(d_i)^1 = d_{i,1}, \quad (d_i)^2 = d_{i,2}, \quad \ldots, \quad (d_i)^{m_2-1} = d_{i,m_2-1}, \quad (d_i)^{m_2} = d_{i,m_2}$$

and calculate the divided differences with respect to the points of interpolation $\mu_1, \mu_2, \ldots, \mu_{m_2}$:

$$(d_i)^1 \quad (d_i)^2 \quad \ldots \quad (d_i)^k \quad \ldots \quad (d_i)^{m_2-2} \quad (d_i)^{m_2-1} \quad (d_i)^{m_2}$$

$$(d_i)^3 \quad (d_i)^4 \quad \ldots \quad (d_i)^{k+1} \quad \ldots \quad (d_i)^{m_2-1}$$

$$(d_i)^{m_2-2} \quad (d_i)^{m_2-1} \quad \ldots \quad (d_i)^{m_2}$$

After that we construct the Newton interpolating polynomial

$$r_i(\mu) = (d_i)^1 + (d_i)^2(\mu - \mu_1) + \ldots + (d_i)^{m_2}(\mu - \mu_1)\ldots(\mu - \mu_{m_2-1}). \quad (15)$$

Since $r_i$ is the interpolating polynomial, one has

$$r_i(\mu_j) = d_{i,j}. \quad (16)$$

Finally, we consider the polynomial in two variables

$$p(\lambda, \mu) = r_0(\mu) + r_1(\mu) (\lambda - \lambda_1) + r_2(\mu) (\lambda - \lambda_1)(\lambda - \lambda_2) + \ldots + r_{m_1-1}(\mu) (\lambda - \lambda_1)\ldots(\lambda - \lambda_{m_1-1}). \quad (17)$$

From (16) one has

$$p(\lambda, \mu_j) = d_{0,j} + d_{1,j}(\lambda - \lambda_1) + d_{2,j}(\lambda - \lambda_1)(\lambda - \lambda_2) + \ldots + d_{m_1-1,j}(\lambda - \lambda_1)\ldots(\lambda - \lambda_{m_1-1}) = b_j(\lambda).$$

In particular, from (13) it follows that conditions (10) are fulfilled:

$$p(\lambda_i, \mu_j) = b_j(\lambda_i) = h(\lambda_i, \mu_j), \quad i = 1, 2, \ldots, m_1; \quad j = 1, 2, \ldots, m_2.$$

Thus, the interpolating polynomial is constructed. Expanding the parentheses in (17) and (15) we arrive at representation (9). For more about interpolation of functions of several variables, see [49, ch. 5].

Springer
Finally, according to (8), we arrive at the approximation
\[ F_{k,k+1} = h(A, B) \diamond H \approx p(A, B) \diamond H = \sum_{i=0}^{m_1-1} \sum_{j=0}^{m_2-1} c_{ij} A^i H B^j. \] (18)

Remark 1 Clearly, the same approximation is obtained if one sets (according to (17))
\[ h(A, B) \diamond H \approx H r_0(B) + (A - \lambda_1 1) H r_1(B) + (A - \lambda_1 1)(A - \lambda_2 1) H r_2(B) \]
\[ + \ldots + (A - \lambda_1 1) \ldots (A - \lambda_{m_1-1} 1) H r_{m_1-1}(B). \]

We do not use this representation because we want to apply the Paterson–Stockmeyer algorithm.

Remark 2 One can interpret equalities (10) as conditions on the coefficients \( c_{ij} \) of polynomial (9). Thus he obtains a system of \( m_1 \cdot m_2 \) linear equations with \( m_1 \cdot m_2 \) unknowns. Above, we have proved that this system has a solution for any free terms \( h(\lambda_i, \mu_j) \). Consequently, this solution (which uniquely determines the interpolating polynomial) is unique.

The usage of the formula
\[ f^{[1]}(\lambda_i, \mu_j) = \frac{f(\lambda_i) - f(\mu_j)}{\lambda_i - \mu_j} \]
can cause a loss of accuracy if \( \lambda_i \) and \( \mu_j \) are close\(^6\). Numerical experiments presented in Table 3 (with 16 Chebyshev nodes and the exponential function) show that 30 significant decimal digits provide the absolute error in \( h(\lambda, \mu) \) about \( 10^{-16} \) for \( (\lambda, \mu) \in \sigma(A) \times \sigma(B) \) in the case \( \rho = 2 \) (but 36 decimal digits provide the absolute error in \( h(\lambda, \mu) \) about \( 10^{-16} \) in the case \( \rho = 1 \)), and 59 significant decimal digits provide the relative error in \( h(\lambda_i, \mu_j) \) about \( 10^{-16} \) for the case \( \rho = 2 \) (but 68 decimal digits provide the relative error in \( h(\lambda_i, \mu_j) \) about \( 10^{-16} \) for the case \( \rho = 1 \)). We prefer to ensure the absolute (as opposed to relative) error for the reasons described in Section 4.

We are now in a position to describe the algorithm of approximate calculation of \( h(A, B) \diamond H \) in full. For definiteness, let \( \rho = 2 \). We calculate the points of interpolation \( \lambda_1, \lambda_2, \ldots, \lambda_{m_1} \) and \( \mu_1, \mu_2, \ldots, \mu_{m_2} \) with 30 significant decimal digits (actually, they should have already been calculated when calculating the diagonal blocks \( f(T_{k,k}) \) and \( f(T_{k+1,k+1}) \)). Then we calculate the numbers
\[ p(\lambda_i, \mu_j) = b_j(\lambda_i) = h(\lambda_i, \mu_j), \quad i = 1, 2, \ldots, m_1; \quad j = 1, 2, \ldots, m_2. \]
After that, for each \( j = 1, 2, \ldots, m_2 \), we find the Newton interpolating polynomials \( b_j \) that agrees with the function \( \lambda \mapsto h(\lambda, \mu_j) \) at the points \( \lambda_1, \lambda_2, \ldots, \lambda_{m_1} \), see formula (14).

---

\(^6\)If one uses 16 Chebyshev nodes as points of interpolation and \( \rho = 2 \), then the minimal value of \( |\lambda_i - \mu_j| \) is about 0.0096 (for \( \rho = 1 \), the minimal value of \( |\lambda_i - \mu_j| \) is about 0.0048).
Table 3  The effect of the number of significant decimal digits (in the values of points of interpolation) on the error in the coefficients of the interpolating polynomial (in two variables of degree $15, 15$ with the Chebyshev nodes taken as the points of interpolation) of the function $e^{1/(\lambda - \mu)}$ on $[-3, -1] \times [-1, 1]$

| significant decimal digits | absolute error in the coefficients | relative error in the coefficients |
|---------------------------|------------------------------------|-----------------------------------|
| 30                        | $7.87 \times 10^{-17}$             | $0.83 \times 10^{-2}$             |
| 44                        | $7.87 \times 10^{-31}$             | $8.33 \times 10^{-17}$            |
| 59                        | $7.87 \times 10^{-46}$             |                                    |

on $[-3/2, -1/2] \times [-1/2, 1/2]$

| significant decimal digits | absolute error in the coefficients | relative error in the coefficients |
|---------------------------|------------------------------------|-----------------------------------|
| 36                        | $1.65 \times 10^{-16}$             | $7.00 \times 10^{-2}$             |
| 53                        | $1.65 \times 10^{-33}$             | $6.00 \times 10^{-17}$            |
| 68                        | $1.65 \times 10^{-48}$             |                                    |

Then, for each $i = 1, 2, \ldots, m_1$, we calculate polynomials (15), substitute them in (17), and expand the parenthesis. As a result, we obtain the approximating polynomial $p$ in the form (9). It remains to substitute $A, B,$ and $H$ into it and obtain approximation (8):

$$h(A, B) \diamond H \approx \sum_{j=0}^{m_2-1} \sum_{i=0}^{m_1-1} c_{ij} A^i H B^j.$$  

To this end, for each $j = 1, 2, \ldots, m_2$, we calculate the sums

$$\sum_{i=0}^{m_1-1} c_{ij} A^i.$$  

To accelerate the process, we apply the Paterson–Stockmeyer algorithm. Then we multiply the results by $H$ and obtain the matrices

$$C_j = \sum_{i=0}^{m_1-1} c_{ij} A^i H, \quad j = 1, 2, \ldots, m_2.$$  

It remains to calculate the sum

$$\sum_{j=0}^{m_2-1} C_j B^j.$$  

Now the direct application of the Paterson–Stockmeyer algorithm requires 18 matrix multiplications for $m_2 = 16$ because the coefficients $C_j$ are matrices. Therefore we use the Horner method [22, Section 4.2], which requires $m_2 - 1$ matrix multiplications.

We note the papers [1–3], in which the problem of calculating $f^{[1]}(A, A) \diamond H$ for $A = B$ is discussed in great detail.

7 The whole algorithm

In this section, we recall the sequence of all our manipulations.
0. Given a square matrix $R$ with a real spectrum and a function $f$ analytic in an open neighborhood of the spectrum of $R$. We assume that the entries of $R$ are known with the standard accuracy (IEEE double-precision binary floating-point format, which admits about 16 significant decimal digits) and $f$ can be calculated with any desirable accuracy. The aim is the approximate calculation of $f(R)$.

1. We construct the Schur decomposition $R = QTQ^H$, where $Q$ is unitary and $T$ is upper triangular. We denote by $F$ the matrix $f(T)$.

2. We find a half-open interval $[a, b)$ that contains the spectrum $\sigma(T)$ of $T$ (we recall that $\sigma(T)$ coincides with the set of the diagonal entries of $T$).

3. We choose a number $\rho > 0$. In our numerical experiments in Section 8 we use $\rho = 2$ (or sometimes $\rho = 1$). Then we divide $[a, b)$ into parts of the length $\rho$:

$$
[a, b) \subseteq [a, a+\rho) \cup [a+\rho, a+2\rho) \cup [a+2\rho, a+3\rho) \cup \cdots \cup [a+(n-1)\rho, a+n\rho).
$$

Here $n$ denotes the number of the parts.

4. We split the spectrum of $T$ into $n$ clusters: the $k$th cluster is contained in $[a + (k-1)\rho, a+k\rho)$, $k = 1, 2, \ldots, n$. If some clusters are empty, we skip them and enumerate the rest anew. Then we reorder (see Section 2) the diagonal entries of $T$ so that the entries of the first cluster precede those of the second, the entries of the second cluster precede those of the third, and so on. The order of entries inside a cluster does not matter. Thus, we come to the matrix $T' = UTU^T$, where $U$ is the product of unitary similarity transformations that reorder the neighboring diagonal entries, see Section 2. Below we denote $T'$ by the original symbol $T$.

5. For each $k = 1, 2, \ldots, n$, we calculate $\alpha_k$, the minimal eigenvalue of $T_{k,k}$, and $\beta_k$, the maximal one. We transform $\alpha_k$ and $\beta_k$ into numbers having a large number of significant digits (e.g., by appending additional digits after the already available reliable ones and rounding up or down), which will ensure 16 significant digits in the coefficients of $p$ in step 7. In our numerical experiments in Section 8, we use 30 significant decimal digits in $\alpha_k$ and $\beta_k$ for $\rho = 2$, and 36 significant digits for $\rho = 1$. We calculate $\gamma_k = \frac{\alpha_k+\beta_k}{2}$, the center of the segment $[\alpha_k, \beta_k]$, and its radius $\delta_k = \frac{\beta_k-\alpha_k}{2}$.

6. We calculate the Chebyshev nodes $z_1, z_2, \ldots, z_{16}$ on the segment $[-1, 1]$ with a large number of significant decimal digits (30 or 36, depending on $\rho$ in our numerical experiments in Section 8). For each $k = 1, 2, \ldots, n$, we calculate the Chebyshev nodes $\lambda_1, \lambda_2, \ldots, \lambda_{16}$ on the segment $[\tilde{a}, \tilde{b}] = [-\delta_k, \delta_k]$ by the scaling $\lambda_i = \delta_k z_i$; thus, the numbers $\lambda_i$ have about 30 or 36 significant decimal digits, respectively. Then we construct the Newton interpolating polynomial $p$ that agrees with the function $g(\lambda) = f(\lambda + \gamma_k)$ at $\lambda_1, \lambda_2, \ldots, \lambda_{16}$. We expand $p$ in powers of $\lambda$. Using the Paterson–Stockmeyer algorithm (see Section 5) we calculate the approximation $F_{k,k} = g(A) \approx f(T_{k,k})$, where $A = T_{k,k} - \gamma_k I$.

7. For each $k = 1, 2, \ldots, n-1$, we carry out the following actions. We denote by $\lambda_1, \lambda_2, \ldots, \lambda_{16}$ and $\mu_1, \mu_2, \ldots, \mu_{16}$ the Chebyshev nodes on the segments

---

Footnote: We recall that one can use a closed interval $[a, b]$. In this case, the notation becomes less symmetric but nothing changes essentially.
\[ \hat{\alpha}_1, \hat{\beta}_1 = [-\delta_k, \delta_k] \] and \[ \hat{\alpha}_2, \hat{\beta}_2 = [-\delta_{k+1}, \delta_{k+1}] \] respectively (calculated in step 6).

8. We consider the shifted function \( h(\lambda, \mu) = f^{[1]}(\lambda + \gamma_k, \mu + \gamma_{k+1}) \), where \( f^{[1]}(\lambda, \mu) = \frac{f(\lambda) - f(\mu)}{\lambda - \mu} \). We construct the interpolating polynomial \( p \) in two variables that agrees with the function \( h \) at the points \((\lambda_i, \mu_j)\) as it is described in Section 6. (To ensure higher accuracy, we first construct the interpolating polynomial \( p \) in the Newton form (17), and then expand it in powers of \( \lambda \) and \( \mu \).) We substitute the matrices \( A = T_{k,k} - \gamma_k1 \), \( B = T_{k+1,k+1} - \gamma_{k+1}1 \), and \( H = T_{k,k+1} \) into \( p \) (see formula (18)) and obtain the approximation \( F_{k,k+1} = h(A, B) \circ H \approx p(A, B) \circ H \).

9. We fill in the remaining off-diagonal entries of \( F \) by formula (1) (diagonal by diagonal). Finally, we return to the matrix \( R \) by means of \( f(R) = QUFU^HQ^H \).

8 Numerical experiments

We carry out our numerical experiments using “Mathematica” [58]. In all experiments \( \rho = 2 \) or \( \rho = 1 \). Interpolating polynomials are always constructed by the Chebyshev nodes of degree 15.

Experiment 1 This experiment shows that our algorithm can be used to calculate the exponential function of a matrix with a spectrum uniformly distributed in \([-10, 0]\).

We take \( N = 256 \) for the order of the upper triangular matrix \( R = T \). We set \( \rho = 2 \) and \( n = 5 \) (\( n \) is the number of clusters). We create numbers \( di_i, i = 1, 2, \ldots, N \), generated by a random variable uniformly distributed in \([-n\rho, 0] = [-10, 0]\) and having 100 significant decimal digits, and arrange them in the increasing order. We create a diagonal matrix \( D \) of the size \( N \times N \) with the diagonal entries \( d_{ii} \).

We create an upper triangular matrix \( S \), consisting of units on the main diagonal and random numbers with 100 significant digits uniformly distributed in \([-1/3, 1/3]\) above the main diagonal.

We set \( T = SDS^{-1} \).

Thus, we obtain a random triangular (block) matrix \( T \) with diagonal entries arranged in the desired (i.e., as after step 4 in Section 7) order.

We take for \( f \) the function \( f(\lambda) = e^{\lambda} \). For the precise matrix \( \tilde{F} = f(T) \) we take \( \tilde{F} = Sf(D)S^{-1} \).

After that, we round the matrices \( T \) and \( \tilde{F} \) to 16 significant decimal digits.

We set\(^8\) \( a = d_{11} \) and divide the half-open interval \([a, a + n\rho]\) into parts of the length \( \rho \):

\[ [a, b) \subseteq [a, a + \rho) \cup [a + \rho, a + 2\rho) \cup [a + 2\rho, a + 3\rho) \cup \cdots \cup [a + (n-1)\rho, a + n\rho), \]

\(^8\)Here \( d_{11} \) is the entry of the rounded matrix \( D \).
which generates the partition of the spectrum $\sigma(T)$ into clusters and the matrices into blocks. We calculate $F = f(T)$ according to the algorithm described in items 5–8 of Section 7.

We repeat this experiment 100 times. The result is as follows. The mean value$^9$ of the condition number

$$\kappa(S) = \|S\| \cdot \|S^{-1}\|$$

is $1.95 \times 10^3$. The mean value of the operator norm

$$\|\tilde{F}\|_{2\to2} = \max_{\|x\|_2=1} \|Fx\|_2$$

of $\tilde{F} = f(T)$ is 5.37; here $\|\cdot\|_{2\to2}$ is the matrix norm induced by the Euclidean norm $\|\cdot\|_2$ on $\mathbb{C}^N$. The mean value of the max-norm

$$\|\tilde{F}\|_\infty = \max_{i,j=1,\ldots,N} |\tilde{f}_{ij}|$$

of $\tilde{F}$ is 0.965. The mean value of the operator norm

$$\|[F - \tilde{F}]\|_{2\to2} = \max_{\|x\|_2=1} \|(F - \tilde{F})x\|_2$$

of the error $F - \tilde{F}$ is $7.9 \times 10^{-4}$. The mean value of the Frobenius norm

$$\|F - \tilde{F}\|_2 = \left( \sum_{i=1}^N \sum_{j=1}^N |f_{ij} - \tilde{f}_{ij}|^2 \right)^{\frac{1}{2}}$$

is $1.2 \times 10^{-3}$. The mean value of the max-norm

$$\|[F - \tilde{F}]\|_\infty = \max_{i,j=1,\ldots,N} |f_{ij} - \tilde{f}_{ij}|$$

is $1.2 \times 10^{-4}$; here the maximum is taken over all $i, j = 1, 2, \ldots, N$. The mean value of the incomplete Frobenius norm

$$\|[F - \tilde{F}]\|_{2, \text{incomplete}} = \sqrt{\sum |f_{ij} - \tilde{f}_{ij}|^2}$$

is $9.1 \times 10^{-5}$; here the sum is taken only over the entries of the two main block diagonals. The mean value of the incomplete max-norm

$$\|[F - \tilde{F}]\|_{\infty, \text{incomplete}} = \max |f_{ij} - \tilde{f}_{ij}|$$

is $1.5 \times 10^{-5}$; here the maximum is taken over the entries of only the two main block diagonals.

**Experiment 2** This experiment shows that our algorithm is also applicable to a more complicated function $f(\lambda) = \log \lambda$.

We again take $N = 256$ for the order of the upper triangular matrix $R = T$, and take $\rho = 2$ and $n = 5$. We create a diagonal matrix $D$ of the size $N \times N$ with diagonal entries $d_{ii}, i = 1, 2, \ldots, N$, generated by a random variable uniformly distributed in

---

$^9$The *mean value* of the numbers $x_1, x_2, \ldots, x_{100}$ is defined as $\frac{1}{100} \sum_{i=1}^{100} x_i$. 

@ Springer
[1, 11] having 100 significant decimal digits and arranged in the increasing order. We set \(a = d_{11}\) and divide the half-open interval \([a, a + n\rho)\) into parts of the length \(\rho:\)

\([a, b) \subseteq [a, a + \rho) \cup [a + \rho, a + 2\rho) \cup [a + 2\rho, a + 3\rho) \cup \cdots \cup [a + (n - 1)\rho, a + n\rho)\).

We create an upper triangular matrix \(S\). We set \(s_{ii} = 1, i = 1, 2, \ldots, N\), and we take for the other entries of \(S\) above the main diagonal the values of a random variable uniformly distributed in \([-1/4, 1/4]\) with 100 significant decimal digits. The further actions are the same as in Experiment 1.

The result is as follows. The mean value of the condition number \(\kappa(S)\) is 191.5. The mean value of the operator norm \(\|\widetilde{F}\|_{2 \rightarrow 2}\) is 29.7. The mean value of the max-norm \(\|\widetilde{F}\|_{\infty}\) is 5.5. The mean value of the operator norm \(\|F - \widetilde{F}\|_{2 \rightarrow 2}\) is \(1.1 \times 10^{-3}\). The mean value of the max-norm \(\|F - \widetilde{F}\|_{\infty}\) is \(2.1 \times 10^{-3}\). The mean value of the incomplete max-norm \(\|F - \widetilde{F}\|_{\infty, \text{incomplete}}\) is \(5.9 \times 10^{-4}\).

**Experiment 3** In this experiment, we apply our algorithm to a matrix from a standard test set.

We take the tridiagonal Toeplitz matrix [21, section 28.5]

\[
R = \begin{pmatrix}
d & e & 0 & \ldots & 0 & 0 & 0 \\
c & d & e & \ldots & 0 & 0 & 0 \\
0 & c & d & e & \ldots & \ddots & \vdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & \ldots & c & d & e \\
0 & 0 & 0 & \ldots & 0 & c & d \\
\end{pmatrix}
\]

of the order \(N = 256\) with \(c = 2, e = 5/2,\) and \(d = 5\). The parameters ensure that \(\sigma(R) \subseteq [0.5, 9.5]\). We take \(f(\lambda) = \sqrt{\lambda}\). We calculate \(\widetilde{F} = f(R)\) with \(R\) rounded to 200 significant decimal digits (300 decimal digits give the same final precision, but 150 do not). After that, we round the matrices \(R\) and \(\widetilde{F}\) to 16 significant decimal digits. We consider the rounded \(\widetilde{F}\) as an exact matrix \(f(R)\).

We calculate the Schur decomposition \(R = QT Q^H\) (where \(Q\) is unitary and \(T\) is upper triangular) and carry out the reordering. Next, we divide the half-open interval \([a, b)\) (the interval that contains the spectrum \(\sigma(R)\) of the rounded \(R\)) into parts of the length \(\rho\), which generates the partition of the spectrum \(\sigma(T)\) into clusters. Since the diagonal entries of \(T\) may have small imaginary parts, we put an eigenvalue in a corresponding cluster if its real part belongs to the appropriate half-open interval. Then we split \(T\) into blocks in accordance with this division. We calculate \(f(T)\) according to the algorithm described in items 5–8 of Section 7. Finally, we set \(\widetilde{F} = Q f(T) Q^H\) (more precisely, \(F = U Q f(T) Q^H U^H\), where \(U\) is the unitary matrix that performs the reordering) and consider \(F\) as an approximation to \(f(R)\).

For \(\rho = 2\), the comparison with the precise result \(\widetilde{F}\) is as follows. The max-norm \(\|\widetilde{F}\|_{\infty}\) is 2.17. The operator norm \(\|F - \widetilde{F}\|_{2 \rightarrow 2}\) is \(4.01 \times 10^{-4}\). The Frobenius norm \(\|F - \widetilde{F}\|_2\) is \(4.02 \times 10^{-4}\). The max-norm \(\|F - \widetilde{F}\|_{\infty}\) is \(2.1 \times 10^{-5}\).
For $\rho = 1$, the comparison with the precise result $\tilde{F}$ is as follows. The operator norm $\|F - \tilde{F}\|_2 \to 2$ is $7.81 \times 10^{-6}$. The Frobenius norm $\|F - \tilde{F}\|_2$ is $7.82 \times 10^{-6}$. The max-norm $\|F - \tilde{F}\|_{\infty}$ is $5.34 \times 10^{-7}$.

**Experiment 4** We compare our algorithm with the Schur–Parlett algorithm. We recall that the Schur–Parlett algorithm uses a parameter $\delta$ which means the minimum allowable distance between clusters. We use the value $\delta = 0.1$ recommended in [11].

We note that if $|\lambda_i - \lambda_j| > \delta$ for all eigenvalues $\lambda_i \neq \lambda_j$, all clusters in the Schur–Parlett algorithm have the size $1 \times 1$ and the algorithm is reduced to the non-block Parlett recurrence (1). The differences $t_{jj} - t_{ii}$ in formulae (1) may lead to repeated loss of accuracy.

We take for $R = T$ the Opitz bidiagonal matrix [45] of the size $128 \times 128$ that has the entries $r_{ii} = 10 + 10j/131$, $j = 1, 2, \ldots, 128$, on the main diagonal and the entries $r_{i,i+1} = 1$ on the first upper diagonal (the rest of elements are zero). We substitute this matrix in the function $f(\lambda) = e^{i\lambda}$ by means of the two methods and compare with the precise (computed with 200 significant decimal digits) result $\tilde{F}$.

For the non-block Parlett algorithm we have (in usual notation)

$$\|F - \tilde{F}\|_2 \to 2 = 3.00 \times 10^{-7}, \|F - \tilde{F}\|_2 = 5.15 \times 10^{-7}, \|F - \tilde{F}\|_{\infty} = 4.38 \times 10^{-8}.$$  

For our algorithm with $\rho = 2$ we have

$$\|F - \tilde{F}\|_2 \to 2 = 2.94 \times 10^{-11}, \|F - \tilde{F}\|_2 = 3.78 \times 10^{-11}, \|F - \tilde{F}\|_{\infty} = 1.04 \times 10^{-11}.$$  

**Experiment 5** We again compare our algorithm with the Schur–Parlett algorithm. If the spectrum of $R$ cannot be divided into well-separated clusters, the Schur–Parlett algorithm calculates $f(R)$ by the Taylor formula applied to the whole $R$. First, calculating by the Taylor formula is time consuming. Second, and even worse, it may lead to a loss of accuracy. We give a corresponding example.

It is known that if the absolute value of a negative number $\lambda < 0$ is large, the sum of the Taylor series for $e^{\lambda}$ in floating-point arithmetic may essentially differ from the exact value of $e^{\lambda}$. For example, in the IEEE double-precision binary floating-point format, $e^{\lambda} - \sum_{i=0}^{200} \frac{\lambda^i}{i!} = 8.6 \times 10^{-7}$ for $\lambda = -25$ and $e^{\lambda} - \sum_{i=0}^{200} \frac{\lambda^i}{i!} = -0.56$ for $\lambda = -40$.

We take $f(\lambda) = e^{-\lambda^2}$. We take for $R = T$ the diagonal matrix of the size $128 \times 128$ with the diagonal entries

$$r_{ii} = 10(i - 1)/127 - 5, \quad i = 1, 2, \ldots, 128,$$

from the segments $[-5, 5]$.

First, we apply the Schur–Parlett algorithm, which is reduced to the substitution into the Taylor series. We denote by $\tilde{F}$ the matrix $e^{-R^2}$ calculated with 200 significant decimal digits. For the Schur–Parlett algorithm we have (in usual notation)

$$\|F - \tilde{F}\|_2 \to 2 = 2.33 \times 10^{-6}, \|F - \tilde{F}\|_2 = 3.43 \times 10^{-6}, \|F - \tilde{F}\|_{\infty} = 2.33 \times 10^{-6},$$  

$$\|F - \tilde{F}\|_2 \to 2 = 2.94 \times 10^{-11}, \|F - \tilde{F}\|_2 = 3.78 \times 10^{-11}, \|F - \tilde{F}\|_{\infty} = 1.04 \times 10^{-11}.$$
with $F$ being the matrix $p(R)$ of the rounded matrix $R$, where $p$ is the 200th Taylor polynomial (with 16 significant decimal digits) of the function $f$. For our algorithm with $\rho = 2$, we have
\[
\|F - \tilde{F}\|_2 = 3.81 \times 10^{-10}, \quad \|F - \tilde{F}\|_2 = 1.43 \times 10^{-9}, \quad \|F - \tilde{F}\|_\infty = 3.81 \times 10^{-10}.
\]
For our algorithm with $\rho = 1$, we have
\[
\|F - \tilde{F}\|_2 = 3.66 \times 10^{-15}, \quad \|F - \tilde{F}\|_2 = 1.50 \times 10^{-14}, \quad \|F - \tilde{F}\|_\infty = 3.66 \times 10^{-15}.
\]

**Experiment 6** In this Experiment, we compare our algorithm with the scaling and squaring method using the idea of [11, Experiment 3]. Let $R$ be the $128 \times 128$ two-diagonal matrix with the diagonal entries
\[
\begin{align*}
  r_{ii} & = 10i/128 - 5, & i = 1, 2 \ldots, 128, \\
  r_{1,125} = r_{2,126} = r_{3,127} = r_{4,128} = 10^{12}
\end{align*}
\]
(all other elements are zero). We calculate $e^R$. First, we apply our algorithm. The norm $\|F - \tilde{F}\|_2$ of the difference between the matrix $F = e^R$ calculated by our algorithm (with $\rho = 2$) and the matrix $\tilde{F} = e^R$ calculated with 200 significant decimal digits is $1.42 \times 10^{-14}$. Then, we apply the scaling and squaring method [22, section 10.3]. By the scaling and squaring method, we mean the approximate formula
\[
e^R \approx \left( r_m(R/2^k) \right)^{2^k} = \left( p_m(R/2^k)(q_m(R/2^k))^{-1} \right)^{2^k},
\]
where $r_m(\lambda) = \frac{p_m(\lambda)}{q_m(\lambda)}$ is the diagonal Padé approximant, $p_m$ and $q_m$ are polynomials of degree $m$. We calculate the minimum of the norm
\[
\|\tilde{F} - \left( r_m(R/2^k) \right)^{2^k}\|_2
\]
over all $k = 1, 2, \ldots, 100$ and $m = 1, 2, \ldots, 30$. The minimum is equal to $3.9 \times 10^{-3}$.

**9 Conclusions**

In this paper, we describe an algorithm for calculating an analytic function $f$ of a matrix $R$ with a real spectrum. The algorithm can be considered as a modification of the block Schur–Parlett algorithm. It is natural to use it when the block Schur–Parlett algorithm creates too large clusters.

The algorithm first calculates the Schur upper triangular form $T = Q^H R Q$, where $Q$ is unitary, and the diagonal elements of $T$ are arranged in order. Thus, the problem is reduced to calculating $F = f(T)$. The matrix $T$ splits into blocks so that the spectra of diagonal blocks $T_{k,k}$ are neither too large nor too small. The small spectra of $T_{k,k}$ allow us to calculate $F_{k,k}$ and $F_{k,k+1}$ approximating $f$ with a polynomial of relatively low degree (specifically, we use interpolating polynomials with the Chebyshev nodes of degree 15). Then we calculate the rest of the entries of $F$ using the non-block Parlett recurrence (1). The large distance between the spectra of $T_{k,k}$ and $T_{k+2,k+2}$ ensures that the Parlett recurrence does not lead to a large loss of accuracy.
Now it is convenient to say a few words about the choice of the parameter $\rho$. We calculate the entries of the matrix $F = f(T)$ by formula (1) if $f_{ij}$ does not belong to the main or the first upper block diagonal. It is known that the main loss of accuracy when using formula (1) is connected with the calculation of $t_{jj} - t_{ii}$ when $t_{jj}$ is close to $t_{ii}$. We recall that our algorithm ensures that $t_{jj} - t_{ii}$ is calculated only for $|t_{jj} - t_{ii}| \geq \rho$. From this point of view, the higher $\rho$, the better.

On the other hand, $\rho$ is the length (more precisely, its upper estimate) of the spectrum of the diagonal blocks $T_{k,k}$. We calculate $F_{k,k} = f(T_{k,k})$ replacing $f$ with an interpolating polynomial $p$. We calculate $F_{k,k}+1$ in a similar manner. It is clear that the larger the $\rho$, the higher the degree of $p$ is required and the larger the block sizes become, which slows down the computation. Nevertheless, we note some details that show that the influence of the magnitude of $\rho$ is not very crucial. (i) The polynomial $p$ is applied to matrices $T_{k,k}$ of a middle size (comparative to the size of $T$). (ii) We use the Paterson–Stockmeyer algorithm to substitute a matrix into a polynomial, which requires about $\sqrt{m}$ matrix multiplications, where $m$ is the degree of the polynomial. (iii) In our numerical experiments, the calculation of the two main block diagonals (including the calculation of interpolating polynomials) was about 5 times faster (for $n = 5$ diagonal blocks of approximately same order and $N = 256$) than the calculation of the remaining entries according to (1); thus, at least for these examples, the computation time is comparable to the time required for the Parlett algorithm. The optimal and automatic choice of $\rho$ needs special research.

Some possible modifications and generalizations of the algorithm are obvious. First, instead of interpolation by polynomials with the Chebyshev nodes, other methods of approximate calculation of $F_{k,k}$ and $F_{k,k+1}$ can be used. Second, the spectrum of $R$ may be contained in a non-real segment of $\mathbb{C}$ or in a non-closed curve without self-intersections. The algorithm can also be modified to the case when the spectrum is located in a small neighborhood of a segment or a curve, see Experiment 3.

The calculation of $f(R)$ is an open problem when the spectrum of $R$ densely fills a large circumference.

Acknowledgements The authors would like to thank the anonymous referee for valuable comments, which helped to considerably improve the exposition of the paper.

Funding This work was supported by the joint project of the Czech Science Foundation and Russian Foundation for Basic Research (grants nos. 20-10591J and 19-52-26006).

References

1. Al-Mohy, A.H., Higham, N.J.: Computing the Fréchet derivative of the matrix exponential, with an application to condition number estimation. SIAM J. Matrix Anal. Appl. 30(4), 1639–1657 (2008/09). https://doi.org/10.1137/080716426
2. Al-Mohy, A.H., Higham, N.J.: The complex step approximation to the Fréchet derivative of a matrix function. Numer. Algorithm. 53(1), 113–148 (2010). https://doi.org/10.1007/s11075-009-9323-y
3. Al-Mohy, A.H., Higham, N.J., Relton, S.D.: Computing the Fréchet derivative of the matrix logarithm and estimating the condition number. SIAM J. Sci. Comput. 35(4), C394–C410 (2013). https://doi.org/10.1137/120885991
4. Babenko, K.I. Osnovy chislennogo analiza [Fundamentals of Numerical Analysis], second edn. Regulyarnaya i Khaolicheskaia Dinamika, Moscow–Izhevsk (2002). (in Russian)
5. Bader, P., Blanes, S.F.C.: Computing the matrix exponential with an optimized Taylor polynomial approximation. Mathematics 7(12), 1174 (2019)
6. Bai, Z., Demmel, J.: Using the matrix sign function to compute invariant subspaces. SIAM J. Matrix Anal. Appl. 19(1), 205–225 (1998). https://doi.org/10.1137/S0895479896927179
7. Bai, Z., Demmel, J.W.: On swapping diagonal blocks in real Schur form. Linear Algebra Appl. 186, 73–95 (1993). https://doi.org/10.1016/0024-3795(93)90286-W
8. Burrage, K., Hale, N., Kay, D.: An efficient implicit FEM scheme for fractional-in-space reaction-diffusion equations. SIAM J. Sci. Comput. 34(4), A2145–A2172 (2012). https://doi.org/10.1137/110847007
9. Carbonell, F., Jimenez, J.C., Pedroso, L.M.: Computing multiple integrals involving matrix exponents. J. Comput. Appl. Math. 213(1), 300–305 (2008)
10. Cardoso, J.R., Sadeghi, A.: Computation of matrix gamma function. BIT 59(2), 343–370 (2019). https://doi.org/10.1007/s10543-018-00744-1
11. Davies, P.I., Higham, N.J.: A Schur-Parlett algorithm for computing matrix functions. SIAM J. Matrix Anal. Appl. 25(2), 464–485 (electronic). https://doi.org/10.1137/S0895479802410815 (2003)
12. Davis, C.: Explicit functional calculus. Linear Algebra Appl. 6, 193–199 (1973)
13. Dzjadyk, V.K., Ivanov, V.V.: On asymptotics and estimates for the uniform norms of the Lagrange interpolation polynomials corresponding to the Chebyshev nodal points. Anal. Math. 9(2), 85–97 (1983). https://doi.org/10.1007/BF01982005
14. Gel’fond, A.O.: Calculus of Finite Differences. International Monographs on Advanced Mathematics and Physics. Hindustan Publishing Corp., Delhi. Translation of the third Russian edition (1971)
15. Gil’, M.I.: Norm estimates for functions of two operators on tensor products of Hilbert spaces. Math. Nachr. 281(8), 1129–1141 (2008). https://doi.org/10.1002/mana.200510665
16. Gil’, M.I.: Norm estimates for functions of two non-commuting matrices. Electron. J. Linear Algebra 22, 504–512 (2011). https://doi.org/10.13001/1081-3810.1453
17. Godunov, S.K.: Ordinary Differential Equations with Constant Coefficient, Translations of Mathematical Monographs, Vol. 169. American Mathematical Society, Providence. Translated from the Russian original (1994)
18. Golub, G.H., Van Loan, C.F. Matrix Computations, 4th edn. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore (2013)
19. Grimm, V., Hochbruck, M.: Rational approximation to trigonometric operators. BIT 48(2), 215–229 (2008). https://doi.org/10.1007/s10543-008-0185-9
20. Hargreaves, G.I., Higham, N.J.: Efficient algorithms for the matrix cosine and sine. Numer. Algorithm. 40(4), 383–400 (2005). https://doi.org/10.1007/s11075-005-8141-0
21. Higham, N.J.: Accuracy and Stability of Numerical Algorithms, 2nd edn. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (2002). https://doi.org/10.1137/1.9780898718027
22. Higham, N.J.: Functions of matrices: theory and computation. Society for Industrial and Applied Mathematics (SIAM), Philadelphia. https://doi.org/10.1137/1.9780898717778 (2008)
23. Higham, N.J., Lin, L.: A Schur-Padé algorithm for fractional powers of a matrix. SIAM J. Matrix Anal. Appl. 32(3), 1056–1078 (2011). https://doi.org/10.1137/10081232X
24. Higham, N.J., Liu, X.: A multiprecision derivative-free Schur–Parlett algorithm for computing matrix functions. Tech. Rep. MIMS EPrint 2020.19, Manchester Institute for Mathematical Sciences, The University of Manchester, Manchester, (2020) Revised (2021)
25. Householder, A.S.: The Theory of Matrices in Numerical Analysis. Blaisdell Publishing Co, Ginn and Co., New York (1964)
26. Ikramov, K.D.: Numerical Solution of Matrix Equations. Orthogonal Methods. Izdat. “Nauka”, Moscow (1984). (in Russian)
27. Ikramov, K.D.: Matrix pencils — Theory, applications, numerical methods. Itochi Nauki Tehk., Ser. Mat. Anal. 29, 3–106 (1991). (in Russian); English translation in J. Soviet Math. 64(2), 783–853 (1993)
28. Jarlebring, E., Damm, T.: The Lambert W function and the spectrum of some multidimensional time-delay systems. Automatica 43(12), 2124–2128 (2007). https://doi.org/10.1016/j.automatica.2007.04.001
29. Jódar, L., Cortés, J.C.: Some properties of gamma and beta matrix functions. Appl. Math. Lett. 11(1), 89–93 (1998). https://doi.org/10.1016/S0893-9659(97)00139-0
30. Jordan, C. Calculus of Finite Differences, 3rd edn. Chelsea Publishing Co., New York (1965)
31. Kenney, C.S., Laub, A.J.: The matrix sign function. IEEE Trans. Automat. Control 40(8), 1330–1348 (1995)
32. Kenney, C.S., Laub, A.J.: A Schur–Fréchet algorithm for computing the logarithm and exponential of a matrix. SIAM J. Matrix Anal. Appl. 19(3), 640–663 (electronic) (1998). https://doi.org/10.1137/S0895479896300334
33. Kressner, D.: Bivariate matrix functions. Oper. Matrices 8(2), 449–466 (2014). https://doi.org/10.7153/oam-08-23
34. Kressner, D.: A Krylov Subspace Method for the Approximation of Bivariate Matrix Functions. In: Structured Matrices in Numerical Linear Algebra, Springer INdAM Ser., Vol. 30, pp. 197–214. Springer, Cham (2019)
35. Kurbatov, V.G., Kurbatova, I.V.: Computation of a function of a matrix with close eigenvalues by means of the Newton interpolating polynomial. Linear Multilinear Algebra 64(2), 111–122 (2016). https://doi.org/10.1080/03081087.2015.1024243
36. Kurbatov, V.G., Kurbatova, I.V.: Computation of Green’s function of the bounded solutions problem. Comput. Methods Appl. Math. 18(4), 673–685 (2018). https://doi.org/10.1515/cmam-2017-0042
37. Kurbatov, V.G., Kurbatova, I.V.: Green’s function of the problem of bounded solutions in the case of a block triangular coefficient. Oper. Matrices 13(4), 981–1001 (2019). https://doi.org/10.7153/oam-2019-13-69
38. Kurbatov, V.G., Kurbatova, I.V.: An estimate of approximation of a matrix-valued function by an interpolation polynomial. Eur. Math. J. 11(1), 86–94 (2020). https://doi.org/10.32523/2077-9879-2020-11-1-86-94
39. Kurbatov, V.G., Kurbatova, I.V., Oreshina, M.N.: Analytic functional calculus for two operators. Adv. Oper. Theory 6(4), 60 (2021). https://doi.org/10.1007/s43036-021-00156-z
40. Kurbatov, V.G., Oreshina, M.N.: Interconnect macromodelling and approximation of matrix exponent. Analog Integr. Circ. Sig. Process 40(1), 5–19 (2004)
41. Kurbatov, I.V., Pechkurov, A.V.: Representations of Green’s function of the bounded solutions problem for a differential-algebraic equation. Banach J. Math. Anal. 14(3), 707–736 (2020). https://doi.org/10.1007/s43037-019-00036-y
42. McCurdy, A., Ng, K.C., Parlett, B.N.: Accurate computation of divided differences of the exponential function. Math. Comp. 43(168), 501–528 (1984). https://doi.org/10.1090/S0025-5718-1984-0748746-3
43. Moler, C., Van Loan, C.F.: Nineteen dubious ways to compute the exponential of a matrix. SIAM Rev. 20(4), 801–836 (1978). https://doi.org/10.1137/1020098
44. Moler, C., Van Loan, C.F.: Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. SIAM Rev. 45(1), 3–49 (2003). https://doi.org/10.1137/S00361445024180
45. Opitz, G.: Steigungsmatrizen. Z. Angew. Math. Mech. 44, T52–T54 (1964)
46. Parlett, B.N.: Computation of functions of triangular matrices. DTIC Document Memorandum ERL-m481, Berkeley, Electronics Research Laboratory, College of Engineering University of California (1974)
47. Parlett, B.N.: A recurrence among the elements of functions of triangular matrices. Linear Algebra Appl. 14(2), 117–121 (1976)
48. Paterson, M.S., Stockmeyer, L.J.: On the number of nonscalar multiplications necessary to evaluate polynomials. SIAM J. Comput. 2, 60–66 (1973). https://doi.org/10.1137/0202007
49. Phillips, G.G.: Interpolation and Approximation by Polynomials CMS Books in Mathematics/Ouvrages De Mathématiques De La SMC, vol. 14. Springer, New York (2003). https://doi.org/10.1007/b97417
50. Sastre, J.: Efficient evaluation of matrix polynomials. Linear Algebra Appl. 539, 229–250 (2018). https://doi.org/10.1016/j.laa.2017.11.010
51. Sastre, J., Ibáñez, J., Alonso, P., Peinado, J., Defez, E.: Two algorithms for computing the matrix cosine function. Appl. Math. Comput. 312, 66–77 (2017). https://doi.org/10.1016/j.amc.2017.05.019
52. Schmelzer, T., Trefethen, L.N.: Computing the gamma function using contour integrals and rational approximations. SIAM J. Numer. Anal. 45(2), 558–571 (2007). https://doi.org/10.1137/050646342
53. Seshu, S., Reed, M.B.: Linear Graphs and Electrical Networks, Addison–Wesley Series in the Engineering Sciences. Addison-Wesley Publishing Co., Inc., Reading (1961)
54. Varga, R.S., Karpenter, A.D.: A conjecture of S. Bernstein in approximation theory. Mat. Sb. (N.S.) 129(171)(4), 535–548. https://doi.org/10.1070/SM1987v057n02ABEH003086 (1986)
55. Vlach, J., Singhal, K. Computer Methods for Circuit Analysis and Design (Electrical Engineering), 2nd edn. Van Nostrand Reinhold Electrical/Computer Science and Engineering Series. Kluwer Academic Publishers, New York (1993)
56. Voevodin, V.V., Kuznetsov, Y.A.: Matritsy i vychisleniya [Matrices and computations]. “Nauka” Moscow. (in Russian) (1984)
57. Wilkinson, J.H.: The algebraic eigenvalue problem. Monographs on Numerical Analysis. The Clarendon Press, Oxford University Press, New York (1988)
58. Wolfram, S. The Mathematica Book, 5th edn. Wolfram Media, New York (2003)

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.