Applied Mathematics, **2011, 2, 247-253**
doi:10.4236/am.2011.22028 Published Online February 2011 (http://www.SciRP.org/journal/am)

Matrix Padé-Type Method for Computing the Matrix Exponential

Chunjing Li[1], Xiaojing Zhu[1], Chuanqing Gu[2]
[1]*Department of Mathematics, Tongji University, Shanghai, China*
[2]*Department of Mathematics, Shanghai University, Shanghai, China*
E-mail: *cqgu@staff.shu.edu.cn*
Received October 31, 2010; *revised December* 23, 2010; *accepted December* 28, 2010

Abstract

Matrix Padé approximation is a widely used method for computing matrix functions. In this paper, we apply matrix Padé-type approximation instead of typical Padé approximation to computing the matrix exponential. In our approach the scaling and squaring method is also used to make the approximant more accurate. We present two algorithms for computing e^A and for computing e^{At} with many $t \geq 0$ respectively. Numerical experiments comparing the proposed method with other existing methods which are MATLAB's functions *expm* and *funm* show that our approach is also very effective and reliable for computing the matrix exponential e^A. Moreover, there are two main advantages of our approach. One is that there is no inverse of a matrix required in this method. The other is that this method is more convenient when computing e^{At} for a fixed matrix A with many $t \geq 0$.

Keywords: Matrix PadÉ-Type Approximation, Matrix Exponential, Scaling and Squaring, Backward Error

1. Introduction

The matrix exponential is the most studied and the most widely used matrix function. It plays a fundamental role in the solution of differential equations in many application problems such as nuclear magnetic resonance spectroscopy, Markov models, and Control theory. These problems often require the computation of e^{At} for a fixed matrix and many $t \geq 0$

For example, with some suitable assumptions on the smooth of a function f, the solution to the inhomogeneous system

$$\frac{dy}{dt} = Ay + f(t, y), y(0) = c, y \in \mathbb{C}^n, A \in \mathbb{C}^{n \times n},$$

is

$$y(t) = e^{At}c + \int_0^t e^{A(t-s)} f(s, y)\, ds.$$

A great number of methods for computing the matrix exponential have been proposed in the past decades. Moler and Van Loan's classic papers [1,2], studied nineteen dubious methods for computing the matrix exponential. Higham [3] which improved the traditional scaling and squaring method by a new backward error analysis is by far one of the most effective methods for computing the matrix exponential and his method has been implemented in MATLAB's *expm* function. A revisited version can be seen in [4]. In Higham's method and many other methods, Padé approximation is used to evaluate the matrix exponential because of its high accuracy.

In this paper we present a new approach to compute the matrix exponential. On one hand, we use the matrix Padé-type approximation given by Gu [5] instead of the typical Padé approximation to evaluate the matrix exponential. Padé-type approximation was first proposed by Brezinski [6,7] in the scalar case. Draux [8] explored this method for the case of matrix-valued function and proposed a matrix Padé-type approximation. On the other hand, the powerful scaling and squaring method is also applied in our approach. This method exploits the relation $e^A = \left(e^{A/2^s}\right)^{2^s}$. Thus we need to evaluate $e^{A/2^s}$ at first and then take squarings for s times to obtain the evaluation of e^A. In the spirit of [3], the scaling number s is determined by backward error analysis. Besides, the problem of overscaling is also considered in our approach.

We briefly introduce the definition and basic property of the matrix Padé-type approximation in Section 2.

Then we develop a specific analysis for our approach in Section 3. In Section 4, two algorithms and some numerical experiments compared with other existing methods are presented. Conclusions are made in Section 5.

2. Matrix Padé-Type Approximation

In this section, we give a brief introduction about Matrix Padé-type approximation or MPTA for short. There are various definitions for MPTA. We are concerned with those based on Brezinski [6,7] and Gu [5].

Let $f(z)$ be a given power series with $n\times n$ matrix coefficients, *i.e*.,

$$f(z)=\sum_{i=0}^{\infty}C_i z^i, C_i\in\mathbb{C}^{n\times n}, z\in\mathbb{C}.$$

Let $\phi: P\to\mathbb{C}^{n\times n}$ be a generalized linear function on a polynomial space P, defined by

$$\phi\left(x^l\right)=C_l, l=0,1,\cdots$$

Let v_m be a scalar polynomial of degree m

$$v_m(z)=\sum_{j=0}^{m}b_j z^j \tag{2.1}$$

and assume that the coefficient $b_m=1$. Then we define

$$W_k(z)=\phi\left(\frac{x^{k-m+1}v_m(x)-z^{k-m+1}v_m(z)}{x-z}\right), \tag{2.2}$$

$$\tilde{v}_m(z)=z^m v_m\left(z^{-1}\right), \tag{2.3}$$

$$\tilde{W}_k(z)=z^k W_k\left(z^{-1}\right). \tag{2.4}$$

Definition 1 *Let* $\tilde{W}_k(z)$ *and* $\tilde{v}_m(z)$ *be defined by* (2.4) *and* (2.3) *respectively. Then*

$$R_{km}(z)=\tilde{W}_k(z)/\tilde{v}_m(z)$$

is called a matrix Padé-type approximation to $f(z)$ *and is denoted by* $(k/m)_f(z)$.

The approximant $(k/m)_f(z)$. satisfies the error formula

$$f(z)-(k/m)_f(z)=O\left(z^{k+1}\right). \tag{2.5}$$

In fact, from (2.1)-(2.4), we have

$$\tilde{v}_m(z)=\sum_{j=0}^{m}b_j z^{m-j} \tag{2.6}$$

and

$$\begin{aligned}W_k(z)&=\phi\left(\sum_{j=0}^{m}b_j\sum_{i=0}^{k-m+j}x^{k-m+j-i}z^i\right)\\&=\sum_{j=0}^{m}b_j\sum_{i=0}^{k-m+j}C_i z^{k-m+j-i}.\end{aligned} \tag{2.7}$$

Then

$$\begin{aligned}\tilde{W}_k(z)&=\sum_{j=0}^{m}b_j\sum_{i=0}^{k-m+j}C_i z^{m-j+i}\\&=\sum_{j=0}^{m}b_j z^{m-j}\sum_{i=0}^{k-m+j}C_i z^i.\end{aligned} \tag{2.8}$$

It follows from (2.6)-(2.8) that

$$\tilde{v}_m(z)f(z)-\tilde{W}(z)=\sum_{i=0}^{\infty}\left(\sum_{j=0}^{m}b_j C_{k-m+1+j+i}\right)z^{k+1+i}. \tag{2.9}$$

Under the normalization condition $\tilde{v}_m(0)=b_m=1$, we obtain (2.5). Note that the numerator $\tilde{W}_k(z)$ is a matrix-valued polynomial and the denominator $\tilde{v}_m(z)$ is a scalar polynomial. For more theories on matrix Padé approximation and matrix Padé-type approximation see [5-10].

3. MPTA for the Matrix Exponential

Let *f* be a function having the Taylor series expansion with the radius of convergence *r* of the form,

$$f(z)=\sum_{i=0}^{\infty}\left(f^{(i)}/i!\right)z^i, |z|<r. \tag{3.1}$$

It follows from Higham [11] that for any $A\in\mathbb{C}^{n\times n}$ and $z\in\mathbb{C}$ with $\rho(Az)<r$, where ρ represents the spectral radius of a matrix,

$$f(Az)=\sum_{i=0}^{\infty}\left(f^{(i)}/i!\right)(Az)^i.$$

According to (2.8) and (2.9) in the previous section, the [k/m] matrix Padé-type approximant to $f(Az)$ can be expressed by the form

$$R_{km}(Az)=P_{km}(Az)/q_{km}(Az),$$

where

$$q_{km}(Az)=\sum_{j=0}^{m}b_j z^{m-j}=q_{km}(z), b_m=1, \tag{3.2}$$

and

$$P_{km}(Az)=\sum_{j=0}^{m}b_j z^{m-j}\sum_{i=0}^{k-m+j}\left(f^{(i)}/i!\right)(Az)^i. \tag{3.3}$$

In fact, the difference between the matrix Padé-type approximants and the typical matrix Padé approximants is that the denominator of the former is a scalar polynomial, as denoted by $q_{km}(z)$ in (3.2).

In the case of the matrix exponential, e^{At} can be defined as follows

$$e^{At}=I+At+\frac{A^2}{2!}t^2+\cdots$$

By (3.2) and (3.3), we immediately obtain the [k/m] matrix Padé-type approximant to e^{At} as follows,

$$R_{km}(At)=P_{km}(At)/q_{km}(t),$$

where

$$q_{km}(t)=\sum_{j=0}^{m}b_j t^{m-j}, b_m=1, \tag{3.4}$$

and

$$P_{km}(At)=\sum_{j=0}^{m}b_j t^{m-j}\left(\sum_{i=0}^{k-m+j}(t^i/i!)A^i\right). \quad (3.5)$$

Now we give an error expression in the following theorem.

Theorem 1 *Let* q_{km} *and* P_{km} *be expressed as* (3.4) *and* (3.5) *respectively, then*

$$\frac{P_{km}(At)}{q_{km}(t)}-e^{At}=\frac{t^{k+1}}{q_{km}(t)}\sum_{i=0}^{\infty}t^i\left(\sum_{j=0}^{m}b_j\frac{A^{k-m+1+i+j}}{(k-m+1+i+j)!}\right). \quad (3.6)$$

Proof: Take $C_i=A^i/i!$ in (2.9).

So far, the coefficients of the denominator q_{km} are arbitrary. These coefficients may affect the accuracy of the approximant. Sidi [12] proposed three procedures to choose these coefficients in the case of vector-valued rational approximation. We can generalize his procedures to the matrix case. We only introduce one procedure, which is called SMPE based on minimal polynomial extrapolation. In the spirit of SMPE in [12], we aim to minimize the norm of the coefficient of the first term of the numerator, *i.e*., the $i=0$ term in the error expression (3.6). Since this term can be viewed as the major part of the error. Under the condition $b_m=1$, other coefficients of q_{km} are the solution to the following minimization problem

$$\min_{b_0,\cdots,b_m}\left\|\sum_{j=0}^{m}b_j\frac{A^{k-m+1+j}}{(k-m+1+j)!}\right\|, \quad (3.7)$$

where we choose the Frobenius norm. The corresponding inner product of two matrices A and B is defined by

$$(A,B)=tr\left(A^HB\right)=\sum_{i=1}^{n}\sum_{j=1}^{n}\bar{A}_{ji}B_{ji}. \quad (3.8)$$

The next lemma can transform the minimization problem (3.7) into the solution of a linear system.

Lemma 1 *The minimization problem* (3.7) *is equivalent to the following linear system*

$$\begin{aligned}&\sum_{j=0}^{m-1}\left(D_{k-m+1+i},D_{k-m+1+j}\right)b_j\\&=-\left(D_{k-m+1+i},D_{k+1}\right),i=0,\cdots,m-1,\end{aligned} \quad (3.9)$$

where $D_l=A^l/l!$.

Proof: See Sidi [12].

Different from the series form error expression in (3.6), we present another form of error bound based on Taylor series truncation error in the following theorem.

Theorem 2 *Suppose f has the Taylor series expansion of the form as in* (3.1) *with radius of convergence r. Let* $R_{km}(Az)=P_{km}(Az)/q_{km}(z)$ *be the matrix Padé-type approximant to* $f(Az)$, *where* $A\in\mathbb{C}^{n\times n}$, $z\in\mathbb{R}$ *with* $\rho(Az)<r$ *and* q_{km} *and* P_{km} *have the form* (3.2) *and* (3.3) *respectively. If* q_{km} *is a real coefficient polynomial and*

$$1-\sum_{j=0}^{m-1}\left|b_jz^{m-j}\right|>0,$$

then for any matrix norm, it follows that

$$\begin{aligned}&\left\|f(Az)-R_{km}(Az)\right\|\\&\le\frac{|z|^{k+1}\left\|\sum_{j=0}^{m}b_j\frac{A^{k-m+1+j}}{(k-m+1+j)!}f^{(k-m+1+j)}\left(Az\xi_j\right)\right\|}{1-\sum_{j=0}^{m}\left|b_jz^{m-j}\right|},\end{aligned} \quad (3.10)$$

where ξ_j, $j=0,\cdots,m$ *are some real constants in* (0,1).

Proof: The result follows from a similar analysis used in [13].

Corollary 1 *Let* $R_{km}(Az)=P_{km}(Az)/q_{km}(z)$ *be the matrix Padé-type approximant to the matrix exponential* e^{At}, *where* $A\in\mathbb{C}^{n\times n},t\in\mathbb{R}$ *and* q_{km} *and* P_{km} *have the form* (3.4) *and* (3.5) *respectively and* q_{km} *is a real coefficient polynomial. If*

$$1-\sum_{j=0}^{m-1}\left|b_jz^{m-j}\right|>0,$$

then

$$\begin{aligned}&\left\|e^{At}-R_{km}(At)\right\|\\&\le\frac{|t|^{k+1}\left\|\sum_{j=0}^{m}b_j\frac{A^{k-m+1+j}}{(k-m+1+j)!}e^{At\xi_j}\right\|}{1-\sum_{j=0}^{m}\left|b_jt^{m-j}\right|},\end{aligned} \quad (3.11)$$

where ξ_j, $j=0,\cdots,m$ *are some real constants in* (0,1).

Proof: The result follows directly from Theorem 2.

The result of corollary 1 shows that the MPTA to e^{At} is feasible. In fact, we do not have to worry about the denominator of the right side in (3.11) since our numerical experiments in the next section show that all b_j are much smaller than 1.

Now we turn to the implementation of the scaling and squaring method of our approach. If the norm of the matrix *At* is not small enough, we use the scaling and squaring technique which computes the matrix exponential indirectly. We want to reduce the norm of *At* to a sufficiently small number because the function can be well approximated near the origin. Thus we need to compute $R_{km}\left(At/2^s\right)$ and then take

$$e^{At}\approx\left(R_{km}\left(At/2^s\right)\right)^{2^s}.$$

The scaling and squaring method goes back at least to Lawson [14].

The scaling integer *s* should be carefully determined. We choose this number based on the error analysis. Higham [3] explored a new backward error analysis of the matrix exponential. According to the analysis in [3], we

give the following lemma.

Lemma 2 *If the approximant* R_{km} *satisfies*

$$e^{-2^{-s}At}R_{km}\left(2^{-s}At\right)=I+G,$$

where $\|G\|<1$ *and the norm is any consistent matrix norm, then*

$$R_{km}\left(2^{-s}At\right)^{2^s}=e^{At+E},$$

where E commutes with A and

$$\frac{\|E\|}{\|At\|}\le\frac{-\log\left(1-\|G\|\right)}{\left\|2^{-s}At\right\|}. \tag{3.12}$$

Proof: See the proof in [3].

The result of (3.12) shows, as Higham [3] points out, that the truncation error *E* in the approximant to e^{At} as equivalent to a perturbation in the original matrix At. So it is a backward error analysis. If we define

$$\Delta=R_{km}\left(2^{-s}At\right)-e^{2^{-s}At},$$

then $G=e^{-2^{-s}At}\Delta$. In order to obtain a clear error analysis for the perturbation *E*, we should give a practical estimation for Δ. Though (3.11) gives an upper bound for Δ, it is a rough estimation for us rather than a practical one. Let

$$A_T=\sum_{i=0}^{k}\left(2^{-s}At\right)^i\Big/i!-e^{2^{-s}At}$$

be the Taylor series truncation error of $e^{2^{-s}At}$. In terms of [11] or [13], the norm of Δ_T is bounded by the inequality

$$\|\Delta_T\|\le\frac{\left(2^{-s}\|At\|\right)^{k+1}e^{2^{-s}\|At\|}}{(k+1)!}. \tag{3.13}$$

Actually, according to the choice of b_j according to the minimization problem (3.7), we can assume that

$$\|\Delta\|\le\|\Delta_T\|.$$

It therefore follows from (3.13) that

$$\|G\|=e^{-2^{-s}\|At\|}\|\Delta\|\le e^{-2^{-s}\|At\|}\|\Delta_T\|\le\frac{(2^{-s}\|At\|)^{k+1}e^{2^{-s+1}\|At\|}}{(k+1)!}. \tag{3.14}$$

We want to choose such *s* that the backward relative error $\|E\|/\|At\|$ does not exceed $u=1.1\times10^{-16}$, which is the unit roundoff in IEEE double precision arithmetic. To this end, in accordance with (3.12), we require the following inequality hold,

$$-\log\left(1-\|G\|\right)\Big/\left\|2^{-s}At\right\|\le u,$$

that is,

$$\|G\|+e^{-2^{-s}\|At\|u}\le1. \tag{3.15}$$

Then (3.14) implies that

$$\frac{\left(2^{-s}\|At\|\right)^{k+1}e^{2^{-s+1}\|At\|}}{(k+1)!}+e^{-2^{-s}\|At\|u}\le1 \tag{3.16}$$

is a sufficient condition for (3.15).

If we denote $\theta=2^{-s}\|At\|$, then (3.16) can be rewritten as follows

$$\frac{\theta^{k+1}e^{2\theta}}{(k+1)!}+e^{-\theta u}\le1.$$

Since $e^{-\theta u}\approx1-\theta u$, finally we obtain

$$g_k(\theta):=\frac{\theta^k e^{2\theta}}{(k+1)!}\le u. \tag{3.17}$$

Then we define this value $\theta_{\max}=\left\{\theta:g_k(\theta)\le u\right\}$. Thus $\|E\|\le u\|At\|$ if *s* satisfies $\left\|2^{-s}At\right\|\le\theta_{\max}$, which means we choose

$$s=\left\lceil\log_2\left(\|At\|/\theta_{\max}\right)\right\rceil. \tag{3.18}$$

However, in practice, the choice of *s* in (3.18) may lead to overscaling, so we need to take measures to reduce the effect caused by overscaling. Therefore the method in Al-Mohy and Higham [15] can be applied.

Lemma 3 *Define* $h_l(x)=\sum_{j=l}^{\infty}a_jx^j$ *with the radius of convergence r and*

$$\tilde{h}_l(x)=\sum_{j=l}^{\infty}|a_j|x^j$$

and suppose $\rho(A)<r$ *and* $p\in\mathbb{N}$. *Then if* $l\ge p(p-1)$,

$$\|h_l(A)\|\le\tilde{h}_l\left(\max\left\{\left\|A^p\right\|^{1/p},\left\|A^{p+1}\right\|^{1/(p+1)}\right\}\right).$$

Proof: See [15].

Note that

$$g_k(\theta)=\sum_{j=k}^{\infty}\frac{2^{j-k}}{(j-k)!(k+1)!}x^j.$$

Then according to lemma 3, we have $g_{16}(\theta)\le g_{16}(\alpha)$, where

$$\alpha=\max\left\{\left\|\left(2^{-s}At\right)^4\right\|^{1/4},\min\left\{\left\|\left(2^{-s}At\right)^3\right\|^{1/3},\left\|\left(2^{-s}At\right)^5\right\|^{1/5}\right\}\right\}. \tag{3.19}$$

So *s* is chosen so that $\alpha\le\theta_{\max}=0.744$.

For the numerator degree *k*, we have no clear theoretical analysis or strategy for the best choice. Based on our numerical experiments, we find $k=16$ may be a

good choice and correspondingly have $\theta_{\max} = 0.744$. Since if k is too small, a large s is required which is potentially dangerous and if k is too large, evaluating matrix multiplications may require excessive computations and storages.

Now we need to consider the total computational cost. The computation of choosing b_j consists of two parts. One part is computing the inner products in the above linear system (3.9). It needs k matrix multiplications to obtain $A^{k-m+1},\cdots,A^{k+1}$ and $m(m+1)$ computations of those inner products. Each inner product defined by (3.8) costs $2n^2$ flops. So the cost of computing inner products is $2m(m+1)n^2$ flops. We choose $m \le n^{1/2}$ to make the cost do not exceed $O(n^3)$. The other part is solving the $m \times m$ linear system (3.9) to obtain the coefficients b_j. It costs $2m^3$ flops which is not expensive since m is much smaller than n. When we evaluate $P_{km}(At)$, we rewrite (3.5) as the following form

$$P_{km}(At) = \sum_{j=0}^{k}\left(\sum_{i=\max\{j-k+m,0\}}^{m} b_i t^{m-i}\right)\frac{t^j}{j!}A^j, \qquad (3.20)$$

where the powers of A have already been obtained in the previous steps. So there are no extra matrix multiplications. Finally, we should take s matrix multiplications for squaring which costs $2sn^3$ flops.

4. Algorithms and Numerical Experiments

Based on the analysis above, we present the following algorithm for computing the matrix Padé-type approximation to the matrix exponential.

Algorithm 1 *This algorithm evaluates the matrix exponential using Padé-type approximation together with scaling and squaring method. The algorithm is intended for IEEE double precision arithmetic.*

1) *Compute and Store* $A^2, A^3, \cdots, A^{17}$.

2) *Repeat scaling* $A \leftarrow A/2^s$ *until s satisfies* $\alpha \le 0.744$, *where* α *is defined by* (3.19) *with* 1*-norm.*

3) *Set* $m = \min\left\{m_0, \left\lfloor n^{1/2} \right\rfloor\right\}$.

4) *Compute the inner products in* (3.9) *via* (3.8).

5) *Solve* (3.9) *to obtain the coefficients of* q_{km}.

6) *Compute* q_{km} *via* (3.4).

7) *Compute* P_{km} *via* (3.20).

8) $R_{km} = P_{km}/q_{km}$.

9) $R_{km} \leftarrow (R_{km})^{2^s}$.

Note that the $m \times m$ linear system in (3.9) maybe ill-conditioned. To avoid this difficulty, we can use a fast and stable iterative method to solve it. The iteration will stop if it fails to converge after the maximum number of iterations. We can also modify the algorithm by resetting b_j or choosing $b_m = 1,\ b_j = 0,\ j = 0,\cdots,m-1,$ when the system is ill-conditioned. The latter case is actually a truncated Taylor series.

Now we present some numerical experiments that compare the accuracy of three methods for computing the matrix exponential.

First we took 53 test matrices obtained from MATLAB R2009b's *gallery* function in the Matrix Computation Toolbox and most of them are real and 8×8. **Figure 1** shows the relative error in the Frobenius-norm of *expmpt* (Algorithm 1) and MATLAB R2009b's functions *expm* and *funm*. For details about *expm* and *funm* or other algorithms for computing the matrix exponential see [12]. The exact matrix exponential e^A is obtained by computing a 150 order Taylor polynomial using MATLAB's Symbolic Math Toolbox. We can make the observations from **Figure 1** as follows.

- MATLAB's function *expm* is still the best code in general.
- The proposed algorithm *expmpt* is also very effective. Precisely, *expmpt* is more accurate than *expm* in 16 examples.
- Both *expm* and *expmpt* are more reliable than *funm* in general.

Second we concentrate on a class of 2×2 matrix of the following form

$$A = \begin{bmatrix} a & b \\ 0 & c \end{bmatrix},$$

where a, c is generated randomly from (–1, –0.5) and b is generated randomly from (–1000, –500). We tested 20 matrices of this form and plotted the results in **Figure 2**. We can make the observations from **Figure 2** as follows.

- Algorithm 1 and MATLAB's function *funm* perform similarly in most examples.

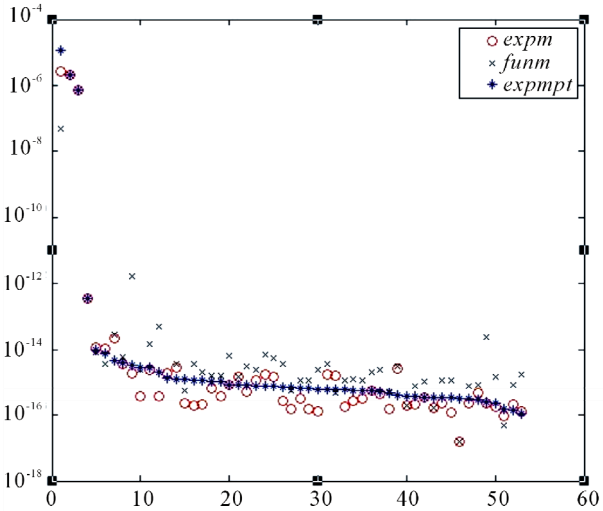

Figure 1. Normwise relative errors for MATLAB's *expm*, *funm* and Algorithm 1 on testing matrices from MATLAB's gallery function.

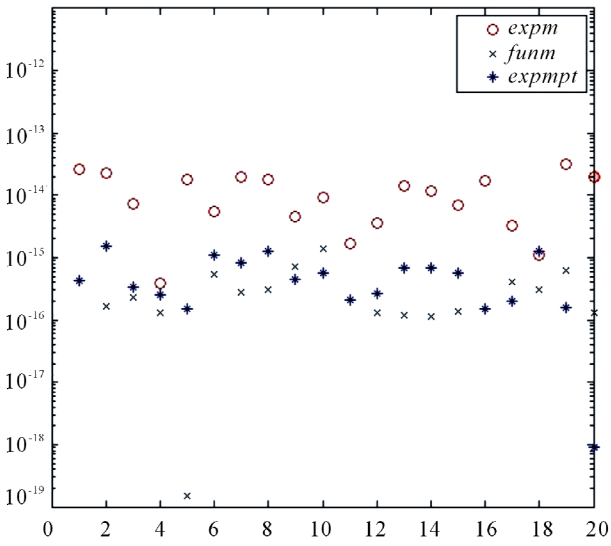

Figure 2. Normwise relative errors for MATLAB's *expm*, *funm* and Algorithm 1 on testing 2/times 2 matrices described above.

- MATLAB's function *expm* is less reliable than the other two methods.

Therefore we can conclude from our experiments that the proposed method has certain value in practice.

Compared with typical Padé approximation which is used in traditional methods for the matrix exponential, Padé-type approximation we used in this paper has both some advantages and disadvantages.

Recall that the [k/m] Padé approximation, denoted by $r_{km}(x)=p_{km}(x)/q_{km}(x)$, to the function f is defined by the properties that that p_{km} and q_{km} are polynomials of degrees k and m, respectively, and that

$$f(x)-r_{km}(x)=O\left(x^{k+m+1}\right).$$

Replacing x with A, we have

$$f(A)-r_{km}(A)=O\left(A^{k+m+1}\right). \tag{4.1}$$

So Padé-type approximation requires a polynomial of higher degree to achieve the same accuracy as Padé approximation does. Another disadvantage of our approach is that each $A^j, j=2,\cdots,k$ has to be evaluated in our approach because we need them to determine the coefficients b_j. In [10], Paterson-Stockmeyer (PS) method was used to evaluate the matrix polynomials. For example, a the matrix polynomial such as

$$p_{15}(A)=b_{15}A^{15}+\cdots+b_1A+b_0I, \tag{4.2}$$

can be evaluated by only 7 matrix multiplications. Therefore, our approach costs more than *expm* if we only compute e^{At} for one or few t.

But in another point of view, our method has some advantages. Real problems usually require e^{At} for many $t\geq 0$, where t represents time. For example, these t are $t_i\in(0,1], i=1,\cdots,r,$ where r is not a small number. We should first divide these t into different intervals according to the criterion that points in the same interval have the same scaling integer. According to (3.20), when computing $P_{km}(At)$ for t in the same interval, we only evaluate $A^j, j=2,\cdots,k$ for one time. However, using PS method to evaluate a matrix polynomial such as $p_{15}(At)$ needs to update these powers of matrix At $At,(At)^2,(At)^4,(At)^6,(At)^8$ and take three extra matrix multiplications for each t. Therefore, when the number of t is very large, PS method may be worthless. Moreover, in our method q_{km} is a scalar polynomial and therefore no matrix division is required except for solving the $m\times m$ linear system with m much smaller than n. Nevertheless, Padé approximation requires the matrix division $q_{km}^{-1}(At)\,p_{km}(At)$ for each t. Matrix division with large dimension is not what we expect.

In the end of this section, we present a modified version of Algorithm 1 to compute e^{At} for many $t\geq 0$. Note that in the following algorithm, we preprocessed the given matrix A by the Schur decomposition to reduce the computational cost since only triangle matrices are involved after the decomposition.

Algorithm 2 *This algorithm based on Algorithm* 1 *evaluates* e^{At} *for* $0<t_1<\cdots<t_r$. *The algorithm is intended for IEEE double precision arithmetic.*

1) *Schur decomposition:* $A=QTQ^H$, *where* Q *is unitary and* T *is upper triangular.*

2) *Compute and store* $T_2,T_3,\cdots,T_{17}$.

3) *Divide all of the points* t_i *into several intervals which are* $U_0,U_1,\cdots,U_w$. *The corresponding scaling integer is* s_i.

4) *Set* $m=\min\left\{m_0,\lfloor n^{1/2}\rfloor\right\}$ *with some positive integer* m_0.

5) For $i=0:w$
   $T\leftarrow T/2^{s_i}$.
   Compute inner products in (3.9) *via* (3.8).
   Solve (3.9) *to obtain the coefficient of* q_{km}.
   For $t_j\in U_i$
      Compute q_{km} *via* (3.4).
      Compute P_{km} *via* (3.20) *where* A *is replaced by* T.
      $R_{km}=P_{km}/q_{km_{s_i}}$.
      $R_{km}\leftarrow(R_{km})^{2^{s_i}}$.
      $R_{km}\leftarrow QR_{km}Q^H$.
   end
end

Note that the purposes of algorithm 1 and of algorithm 2 are different. Algorithm 2 aims to computing e^{At} for many different t and algorithm 1 is designed to compute e^A only. But we emphasize here that in each inner loop,

algorithm uses the same approach as algorithm 1 to compute each e^{At}. In addition, only Schur decomposition is added at the beginning of the algorithm to reduce the computational cost when the number t is very much. Therefore, we do not present any numerical results of algorithm 2.

5. Conclusions

In this paper, we develop a new approach based on matrix Padé-type approximation mixed with the scaling and squaring method to compute the matrix exponential instead of typical Padé approximation. Two numerical algorithms for computing e^{A} and for e^{At} with many $t \geq 0$ respectively are proposed. Our approach is established closely relative to the backward error analysis and computational considerations. Numerical results comparing the proposed algorithm with existing functions *expm* and *funm* in MATLAB have shown the proposed algorithm is effective and reliable for computing the matrix exponential. Compared with typical Padé approximation, the most significant advantages of matrix Padé-type approximation lie in two aspects. One is the convenience for computing e^{At} for a large amount of *t*. The other is its avoiding $n \times n$ matrix divisions, where n is the size of the given matrix *A*.

6. Acknowledgements

The authors gratefully acknowledge the fund support from Applied Mathematics.

7. References

[1] C. B. Moler and C. F. Van Loan, "Nineteen Dubious Ways to Compute the Exponential of a Matrix," *SIAM Review*, Vol. 20, No. 4, 1978, pp. 801-836. doi:10.1137/1020098

[2] C. B. Moler and C. F. Van Loan, "Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later," *SIAM Review*, Vol. 45, No. 1, 2003, pp. 3-49. doi:10.1137/S00361445024180

[3] N. J. Higham, "The Scaling and Squaring Method for the Matrix Exponential Revisited," *SIAM Journal on Matrix Analysis and Application*, Vol. 26, No. 4, 2005, pp. 1179-1193. doi:10.1137/04061101X

[4] N. J. Higham, "The Scaling and Squaring Method for the Matrix Exponential Revisited," *SIAM Review*, Vol. 51, No. 4, 2009, pp. 747-764. doi:10.1137/090768539

[5] C. Gu, "Matrix Padé-Type Approximant and Directional Matrix Padé Approximant in the Inner Product Space," *Journal of Computational and Applied Mathematics*, Vol. 164-165, No. 1, 2004, pp. 365-385. doi:10.1016/S0377-0427(03)00487-4

[6] C. Brezinski, "Rational Approximation to Formal Power Series," *Journal of Approximation Theory*, Vol. 25, No. 4, 1979, pp. 295-317. doi:10.1016/0021-9045(79)90019-4

[7] C. Brezinski, "Padé-Type Approximation and General Orthogonal Polynomials," Birkh Äauser-Verlag, Basel, 1980.

[8] A. Draux, "Approximants de Type Padé et de Table," *Little*: *Publication A*, University of Lille, Lille, No. 96, 1983.

[9] C. Gu, "Generalized Inverse Matrix Padé Approximation on the Basis of Scalar Products," *Linear Algebra and Its Applications*, Vol. 322, No. 1-3, 2001, pp. 141-167. doi:10.1016/S0024-3795(00)00230-5

[10] C. Gu, "A Practical Two-Dimensional Thiele-Type Matrix Padé Approximation," *IEEE Transactions on Automatic Control*, Vol. 48, No. 12, 2003, pp. 2259-2263. doi:10.1109/TAC.2003.820163

[11] N. J. Higham, "Functions of Matrices: Theory and Computation," SIAM Publisher, Philadelphia, 2008. doi:10.1137/1.9780898717778

[12] A. Sidi, "Rational Approximations from Power Series of Vector-Valued Meromorphic Functions," *Journal of Approximation Theory*, Vol. 77, No. 1, 1994, pp. 89-111. doi:10.1006/jath.1994.1036

[13] R. Mathias, "Approximation of Matrix-Valued Functions," *SIAM Journal on Matrix Analysis and Application*, Vol. 14, No. 4, 1993, pp. 1061-1063. doi:10.1137/0614070

[14] J. D. Lawson, "Generalized Runge-Kutta Processes for Stable Systems with Large Lipschitz Constants," *SIAM Journal on Numerical Analysis*, Vol. 4, No. 3, 1967, pp. 372-380. doi:10.1137/0704033

[15] A. H. Al-Mohy and N. J. Higham, "A New Scaling and Squaring Algorithm for the Matrix," *SIAM Journal on Matrix Analysis and Application*, Vol. 31, No. 3, 2009, pp. 970-989. doi:10.1137/09074721X

