Research Article

Lihui Tu and Huanjie Hong*

Validity and error analysis of calculating matrix exponential function and vector product

https://doi.org/10.1515/nleng-2024-0007

received December 4, 2023; accepted March 31, 2024

Abstract: In this study, an efficient calculation method of matrix exponential function and vector product is proposed to reduce the calculation error. This method is based on the flexible exponential integral scheme; using B-series theory and two-color tree theory, the numerical series expression is obtained, and the exponential integral of the matrix exponential function is solved. By constructing a specific numerical integration scheme and analyzing its convergence under the theoretical framework of analytic semi-groups, the convergence analysis of numerical schemes is studied and perfected. The numerical results show that the proposed method of matrix exponential function and vector product has a small error, which shows obvious advantages in reducing the calculation error.

Keywords: matrix exponential function, vector product, efficient calculation, numerical algorithm

1 Introduction

The matrix exponential function \( f(A) \) or matrix function times vector \( f(A)v \) has a wide range of application background; its effective and reliable numerical calculation is one of the hotspots and difficulties in the field of numerical algebra in recent years [1]. In this study, the efficient calculation of matrix empirical function and vector product method will be studied.

The formulation and theoretical study of the matrix empirical function \( f(A) \) started almost at the same time as linear algebra. In 1850, the British mathematician Sylvester first proposed the concept of matrix, and then, Cayley, Sylvester, Frobenius, and other scientists began to study the algebraic properties of matrix, which promoted the development of the whole linear algebra [2]. The study of the matrix empirical function began in 1858 when Cayley discussed the square root function of the second- and third-order matrices. Then, researchers began to devote themselves to the research of pure mathematical theory, such as the definition and basic properties of the general matrix empirical function, and made a lot of progress. With the continuous development of applied science, the matrix empirical function is applied in different fields, and the attention of the matrix empirical function has gradually shifted from the field of pure mathematics to the numerical calculation of the vector product. A series of achievements have been made in the calculation condition number, algorithm analysis and design, and calculation stability of the matrix empirical function and vector product [3].

With the rapid development of computer performance, new problems are emerging in the application field. In recent 20 years, more and more attention has been paid to the numerical algorithm of Eq. (1) and its corresponding theoretical analysis, such as error analysis, stability analysis, space and time complexity analysis

\[
f(A)v, \ v \in \mathbb{C}^{n \times n}, f(z) \text{ is complex-valued function. (1)}
\]

For example, the solution of linear equations \( Ax = b \) is one of the core problems in the field of scientific and engineering, which is equivalent to the reciprocal function of \( f(z) \), namely, \( f(z) = z^{-1} \). In addition, many mathematical models established by physics, chemistry, or economics include the following:

\[
\begin{align*}
\begin{cases}
y'(t) = Ay(t), \ t \geq 0 \\
y(0) = v.
\end{cases}
\end{align*}
\] (2)

In fact, the efficient calculation of matrix empirical function and vector product has always been an active research topic in the field of numerical algebra. In addition to exponential integral, in many fields of science and engineering applications, it is often necessary to solve the matrix exponential function and vector product of the form:

\[
f(A)b, \ f : C \rightarrow C, \ A \in \mathbb{C}^{n \times n}, \ b \in \mathbb{C}^{n},
\] (3)

Besides \( \phi-1111 \) function, \( f \) common forms include symbolic function \( \text{sign}(A) \), logarithmic function \( \log(A) \), and square
root function $\sqrt{A}$. For the application and calculation of the matrix empirical functions, please refer to the latest monograph of Higham [4], which introduces the basic methods and main results for solving the matrix empirical function and vector product.

In a sense, there are many mature and effective algorithms for the calculation of the matrix empirical function and vector product. However, in many practical applications, especially in the spatial discretization of PDEs (partial differential equations), matrix $A$ is usually large and sparse or has some special structural characteristics. For such a matrix, direct calculation of the matrix empirical function $f(A)$ will destroy the structural characteristics of the matrix, which leads to a large number of non-zero elements being filled and a large amount of storage and calculation. The required solution is the product of the matrix exponential function and the vector instead of the matrix exponential function alone. At present, the widely accepted method is to construct a numerical method based on the operation of the matrix $A$ and the vector product, which is usually simpler than calculating the product of the matrix $A$ and the vector $b$ directly after calculating $f(A)$. So far, there have been some effective algorithms designed in this method, which do not need to be solved $f(A)$ separately, but directly calculate the product of the matrix empirical function and the vector. In this process, it mainly betrays the product operation of the matrix and the vector [5].

2 Materials and methods

2.1 Exponential integral of the matrix exponential function

The exponential integral of the matrix exponential function is constructed by the constant variation formula based on the continuous linearization of some vector fields $f$ at the numerical nodes.

Given value nodes $u_n$, assume:

$$J_n = J(u_n) = \frac{\partial f}{\partial u}(u_n), \quad T_n(u(t)) = f(u(t)) - J_n u(t),$$

then the matrix exponential function can be regarded as

$$u(t) = J_n u(t) + r_n(u(t)) + g(u(t)), u(t_0) = u_0,$$

$$t_{n+1} = t_n + h_n, \quad n = 0, 1, \ldots$$

According to the variation formula of the matrix empirical function, the analytical solution of this function at time node (6) can be expressed as follows:

$$u(t_n + h_n) = \int_{t_n}^{t_n + h_n} e^{\int_{t_n}^{t} J(u(t))} dt,$$

(7)

the nonlinear terms $t_n + h_n - t$ in the expression are interpolated and discretized on the interval $[0, 1]$, such as Lagrange interpolation or Newton polynomial interpolation [6], then the following s-level exponential integral schemes with general form can be obtained:

$$
\begin{cases}
U_{n+1} = h_n \sum_{j=1}^{i-1} \hat{a}_j(h_n f_n) T_0(U_{nj}) + h_n \sum_{j=1}^{i-1} \hat{a}_j(h_n f_n) g(U_{nj}), \\
U_{n+1} = h_n \sum_{j=1}^{i} \hat{b}_j(h_n f_n) r_0(U_{nj}) + h_n \sum_{j=1}^{i} \hat{b}_j(h_n f_n) g(U_{nj}).
\end{cases}
$$

(8)

In the process of solving the exponential integral of the matrix exponential function, an important problem is to calculate the product of the exponential matrix function and the series. In fact, through direct verification, it can be found that the product of these matrix functions defined on $hf'$ and the series is still series [7]. Therefore, the remaining key problem is to express the coefficients of the newly generated TB (Taylor-Baker)-series.

In the aforementioned analysis, the relevant operations for solving semilinear equations are defined. These operations, with simple modifications, can be used directly to calculate the case corresponding to the FEI format [8]. Now, introduce the revised definition.

Suppose $a \in G_0, \ h f'$ is represented by $f$, and then it is defined as follows:

$$fa = \sum_{t \in \Gamma} a(t)[t]f,$$

(9)

where $\Gamma$ is a mapping belonging to $G_0$, which needs to meet the following requirements:

$$t(x) = \begin{cases}
1, & x = t \\
0, & x \neq t
\end{cases}$$

(10)

More generally, suppose $p(j)$ be the power series function of $f$:

$$p(j) = p_0 + p_1 f + p_2 f^2 + \ldots,$$

(11)

then,

$$p(t)a = \sum_{t \in \Gamma} a(t)(p_0 t + p_1 f t + \ldots),$$

(12)

When $a \in G_0$,

$$\varphi(hf')TB(a, u_0) = TB(\varphi(j) a, u_0),$$

(13)

where $\varphi(\cdot)$ is an exponential matrix function, which can be expanded into power series.

Based on the aforementioned preparations, it is easy to establish the recursive relationship between the intermediate
stage value \( u_0 \) of the numerical formats and the TB-coefficient of the output vector \( u_t \). If \( \eta_i \in G_i \) is the TB-coefficient corresponding to \( U_0 \) and \( \eta_i \in G_i \) is the output term \( u_t \), then the TB-series of numerical format can be expressed as follows:

\[
\begin{align*}
U_0 &= TB(\eta_i, u_0), 1 \leq i \leq s \\
u_t &= TB(\eta_i, u_0).
\end{align*}
\]

The corresponding TB-coefficient meets the following requirements:

\[
\begin{align*}
\eta_i &= 1 + c_1f(c_1, f)D_f \\
\eta_i &= 1 + c_1f(c_1, f)D_f,
\end{align*}
\]

Once the value of the obtained coefficient is compared with the exact solution, the classical order condition satisfied by the numerical scheme can be obtained [9]. If it is true for \( t \in 2t \) in matrix A, when \( \eta(t) = E(t) \), then the classical order of the scheme is \( p \).

The above-mentioned method is considered to solve the initial value problem of nonlinear ordinary differential equation, which can be decomposed into rigid term and nonrigid term by the right-end term:

\[
\dot{u} = F(u) = f(u) + g(u),
\]

where \( f \) corresponds to the rigid term and \( g \) corresponds to the nonrigid term. It is a natural idea to construct an effective numerical method using the structural characteristics of the right-hand term, which can be decomposed into initial value problems with different functions. The general solution method often ignores the structural characteristics of this kind of problems. The matrix exponential function and vector product are solved by constructing a flexible exponential integral based on exponential time differencing. These schemes are general and flexible and can make more effective use of the structural characteristics of the matrix. This method combines the characteristics of the exponential method [10]. Compared with the exponential method, its construction is based on the linearization of partial vector field \( F \) rather than the whole vector field. There are at least two advantages in this method: first, it is simpler to linearize partial vector field than to linearize the whole vector field, the linearization of partial vector field \( F \) is simpler than that of the whole vector field \( F \), i.e., the matrix of \( F \) is simpler than that of the whole vector field.

At the same time, since \( F \) is a rigid term, this does not affect its stability. Second, in many applications, the right-hand term \( F \) often has a special structure, and its exponential function matrix also has a special form, such as symmetric matrix, antisymmetric matrix, or Hamiltonian matrix. The special structure of matrix plays an important role in the calculation of matrix function, which can greatly reduce the workload of calculation in many cases.

### 2.2 Calculation algorithm of the matrix empirical function and vector product

Schur–Parlett algorithm [11] is the best choice for calculating the general matrix empirical function \( f(A) \), which is based on Schur decomposition of matrix and is forward stable in most cases. Next, the design idea and implementation details of the algorithm will be given. According to the property of the matrix empirical function, \( f(XAX^{-1}) = Xf(A)X^{-1} \), so theoretically, if the form of \( B = XAX^{-1} \) is simple (such as diagonal matrix), \( f(B) \) is also easy to obtain. However, in the case of finite precision operation, it is necessary to require a small number of X's conditions \( k(X) = ||X|| \cdot ||X^{-1}|| \) to ensure the stability of the algorithm. In particular, if \( x \) is a unitary matrix, \( k(X) = 1 \) is an ideal case. Therefore, in order to ensure the reliability of the results of general matrix \( A \), it is necessary to find its simplest form (Schur decomposition \( A = QTQ^H \)), under unitary similar transformation, where \( Q \) is unitary matrix and \( T = (t_{ij}) \) is upper triangular matrix. At the moment, \( f(A) = Qf(T)Q^H \), and the calculation of the upper triangular matrix function \( f(T) \) becomes the key of the problem. \( F = f(T) \) is also the upper triangular matrix function, its diagonal element \( T_{ij} = f(t_{ij}) \), and \( FT = TF \). According to these properties, if the matrix \( T \) is an N-order upper triangular matrix and the diagonal elements are different from each other, and the function \( f(z) \) is defined on the spectrum of \( T \), then \( F = (f(t_{ij})) = f(T) \) can be obtained from the Parlett recurrence formula [12].

There are two points to explain about the algorithm of the matrix empirical function and vector product:

1. The calculation amount of the algorithm is \( 2N^3/3 \)flops (N is the order of matrix \( A \));
2. If \( t_{ij} = t_{ji}, i \neq j \), the algorithm will be interrupted.

In order to avoid the aforementioned problems, the algorithm can be extended to the calculation of block upper triangular matrix functions. By unitary similar transformation, \( A = QTQ^H \) is rearranged so that \( T \) becomes a block upper triangular matrix \( T = (T_{i,j}) \), and when \( i \neq j \), diagonal blocks \( T_{ii} \) and \( T_{ij} \) have no same eigenvalue. \( F = (F_{ij}) = f(T) \) is a block upper triangular matrix, which has the same block structure as matrix \( T \), and satisfies \( F_{ii} = f(T_{ii}) \). According to \( TF = FT \), the recurrence formula for calculating \( F_{ij} \) is as follows:

\[
\begin{align*}
T_{i,i}F_{i,j} - F_{i,i}T_{i,j} & = F_{i,i}T_{i,j} - T_{i,i}F_{i,j} + \sum_{k=1}^{i-1} (F_{i,k}T_{k,j} - T_{i,i}F_{k,j}), \quad i < j.
\end{align*}
\]

According to the theory of function approximation, there are polynomials or rational polynomials \( r(z) \), so in a certain region, \( f(z) \approx r(z) \), and then, the matrix function can be calculated approximately by \( f(A) = r(A) \). However,
even if \( r(z) \) is a good approximation of function \( f(z) \), the approximation effect of \( r(A) \) and \( f(A) \) is questionable, which is closely related to the spectral properties of matrix \( A \). If \( A = XDX^{-1} \), then

\[
f(A) - r(A) = X(f(D) - r(D))X^{-1}.
\]

(18)

It can be seen from the aforementioned section that when the norm of matrix \( A \) is small, it is possible to obtain a better approximation of function \( f(A) \) by truncated Taylor approximation. In most cases, the generated matrix cannot meet this requirement. Fortunately, for some special functions, the problem that \( \|A\| \) is bigger can be resolved by scaling the square algorithm. Take the matrix exponential function \( e^A \) as the example, and there is:

\[
e^A = (e^{A/2})^2, \quad s \in N,
\]

(19)

The selection of parameter \( s \) determines that the value of \( \|A\|/2^2 \) (in general, \( \|A\|/2^2 \) and \( 1 \) are of the same order) can be calculated stably and accurately. Set \( B = R_{m,m}(A/2^2) = e^{A/2^2} \), then:

\[
e^A \approx B^2, \quad s \in N.
\]

(20)

The backward error analysis of the scaling square algorithm shows that if \( \|A\|/2^2 \leq 0.5 \), there is \( B^{1/2} = e^{A/2} \), among which:

\[
\|E\|/\|A\| \leq 8\left(\frac{\|A\|}{2^2}\right)^{2m}\left(\frac{(m!)^2}{(2m)!/(2m + 1)!}\right).
\]

(21)

The calculation amount of \( R_{m,m}(A/2^2)^2 \) is about:

\[(m + s + 1/3)N^3\text{flops},\]

(22)

where \( N \) is the order of matrix \( A \), so if you want to calculate the relative error reach of \( e^A \), you can obtain the method to determine the “optimal” parameters \( m \) and \( s \). Here, the “optimal” parameter refers to the \( m \) and \( s \) that make the calculation amount of the scaling square algorithm minimum. Similarly, the truncated Taylor expansion \( T_k(A/2^2) \) can be used to calculate \( e^{A/2} \). According to the same analysis method, the method to determine the “optimal” parameters \( k \) and \( s \) can also be obtained [13]. Two kinds of scaling square algorithm give the determination value of the optimal parameter and point out when \( \|A\| \) is small, the calculation of function approximation is only half of the truncated Taylor expansion. With the increase of \( \|A\| \), the difference between the two is narrowing. As a whole, the efficiency of function approximation is higher.

For other matrix functions (such as logarithmic function \( \log(A) \), trigonometric function \( \sin(A) \) and \( \cos(A) \)), the algorithm expression of the matrix empirical function and vector product can be obtained as follows:

\[
\log(A) = 2^s \log(A^{1/2}),
\]

\[
\sin(2A) = 2 \sin(A) \cos(A),
\]

(23)

\[
\cos(2A) = 2 \cos^2(A) - I.
\]

2.3 Error analysis of the matrix empirical function and vector product

In the aforementioned algorithm, the main step is to truncate the first finite term of the series of exponential function \( e^T \) of matrix \( T \). This process has the following error results:

Suppose that \( M \) is the \( k \)-step iteration of the aforementioned algorithm for the nonnegative matrix defined above, and \( T_k(M) \) is the calculated solution of \( M \), then there is:

\[
|T_k(M) - M| \leq [(kn + k - n)u + O(u^2)]T_k(M).
\]

(24)

Proof: For the convenience of writing, \( T_k \) is used instead. The aforementioned algorithms are as follows:

\[
T_k = fl(T_{k-1} + p_{k-1}).
\]

(25)

First,

\[
\hat{p}_k - p_k \leq [(k(n + k - n)u + O(u^2))p_k,
\]

(26)

as \( \hat{p}_0 = p_0 = M \), when \( k = 1, \hat{p}_1 = fl\left(M \frac{M}{2}\right) \), it can be known that

\[
|\hat{p}_1 - p_1| \leq [(n + n)u + O(u^3)]p_1.
\]

(27)

That is, when \( k = 1 \), the conclusion is true. If \( k = i \), it is true, then when \( k = i + 1 \), the following formula can be obtained from \( \hat{p}_{i+1} = fl\left(\hat{p}_{i} + \frac{M}{2}\right) \):

\[
\hat{p}_{i+1} \leq \left( \frac{1 + nu + O(u^2)}{1 + n} \right)\hat{p}_{i+1}.
\]

(28)

In the same method, it can be obtained that

\[
\hat{p}_{i+1} \geq \left( 1 - (i + 1)(n + 1)u + O(u^2) \right)p_{i+1},
\]

(29)

and then,

\[
|\hat{p}_{i+1} - p_{i+1}| \leq \left[(i + 1)(n + 1)u + O(u^2)\right]|p_{i+1}|
\]

(30)

Therefore, when \( k = i + 1 \), the conclusion is also true. When \( k = 1 \), since \( \hat{T}_0 = T_0 = I \), the following can be given:

\[
|\hat{T}_1 - T_1| \leq uT_1.
\]

(31)

Now, if \( k = i \), it is true. When \( k = i + 1 \), form \( \hat{T}_{i+1} = fl(\hat{T}_i + \hat{p}_i) \), it can see that

\[
\hat{T}_{i+1} \leq (1 + u)(\hat{T}_i + \hat{p}_i) \leq (1 + u)[1 + (in + i - n)u + O(u^2)]T_i.
\]

(32)

\[
\leq (1 + u)(1 + (n + 1)u + O(u^2))
\]
In the same method, the following formula can be obtained:

$$\hat{T}_{i+1} \geq [1 - ((i + 1)n + (i + 1) - n)u + O(u^2)]T_{i+1}. \quad (33)$$

That is, when \( k = i + 1 \), the conclusion is also true.

In fact, the aforementioned algorithm includes a process of shifting and a process of truncation of \( e^z \) series. For the truncation process, the influence of the calculation error on the calculation accuracy of the process is clearly explained here. Therefore, it is easy to get the influence of the error of the matrix empirical function and vector product on the algorithm. Here is the rounding error of our algorithm.

It can be seen from the error analysis results that since the maximum number of iteration steps of the algorithm is \( m = j + n - 1 \) 1111, the relative error bound of each element of the calculated solution obtained by the algorithm with respect to the accurate solution is not more than \((j + n - 1)m + j)u + tol + O(u^2)\) at most. It can be seen that the maximum relative bound is \( O(u^2)u \), which is very small. Therefore, the aforementioned algorithm can accurately calculate every element of the matrix index of the essentially nonnegative upper triangular matrix with similar diagonal elements, which cannot be solved by traditional methods.

In addition, the minimum \( j \) that makes \( \sum_{k=j+1}^{\infty} \delta^k \leq tol \) hold increases with the increase of \( \delta (\delta \leq 1) \). When \( tol = 100u \) is taken in the actual calculation and \( u = 2^{-53} \) is double machine accuracy, the minimum \( j \) that makes \( \sum_{k=j+1}^{\infty} \delta^k \leq tol \) hold is 16. Therefore, the maximum number of iteration steps of this algorithm satisfies \( m \leq 16 + n - 1 = n + 15 \), so the speed of this algorithm is self-evident.

Now, let us consider the influence of the calculation error of the algorithm to calculate the general essential nonnegative triangular matrix. Obviously, the aforementioned algorithm only includes a scaling process and the process of invoking the algorithm. Through the aforementioned theorems, the calculation errors generated in the process of calling the algorithm are studied. Similarly, the error bound of the whole algorithm process can be easily obtained. In the face of different types of matrices, for sparse matrices, special preprocessing technology and Krylov subspace method are used to maintain the sparsity of the calculation, so as to improve the efficiency. For symmetric matrices, the method takes advantage of their unique properties, such as a numerical integration scheme that preserves symmetry, to ensure the accuracy of the calculation. As for non-square matrices, although this method is primarily concerned with square matrices, it can also be applied to non-square matrices in principle, although the algorithm may need to be adjusted appropriately to handle this type of matrix structure. Future research will continue to explore these questions in depth and make the necessary modifications to the method to ensure that it can be effectively applied to these special types of matrices.

### 2.4 Calculation of the matrix empirical function and vector product

The solution of block Krylov subspace is as the calculation of the matrix empirical function and vector product in:

$$\varphi_{\ell_i}(A)b_0 + \varphi_{\ell_i}(A)b_1 + \varphi_{\ell_i}(A)b_2 + \cdots + \varphi_{\ell_i}(A)b_p, \quad (0 \leq p < < n), \quad \quad (34)$$

where \( A \in \mathbb{R}^{n \times n} \) is a large sparse matrix, \( i = 0, 1, \cdots, p \), and \( b_i \in \mathbb{R}^n \). \( \varphi \)-functions have been introduced in the aforementioned analysis.

Among them,

$$\varphi_{\ell_i}(z) = e^z, \quad \varphi_{\ell_0}(z) = \int_0^1 e^{(1-\delta)z} \frac{\theta^{j-1}}{(j-1)!} \, d\theta, \quad j \geq 1. \quad (35)$$

They satisfy the recursion relation as follows:

$$\varphi_{\ell_{j+1}}(z) = \varphi_{\ell_j}(z) \frac{1}{z}, \quad \varphi_{\ell_0}(z) = e^z, \quad j \geq 0 \quad (36)$$

They can be represented by exponential functions \( e^z \), such as:

$$\varphi_{\ell_0}(z) = \frac{e^z - 1}{z}, \quad \varphi_{\ell_j}(z) = \frac{e^z - 1 - z}{z^2}, \quad \varphi_{\ell_1}(z) = \frac{e^z - 1 - z - \frac{z^2}{2}}{z^3}, \quad \cdots \quad (37)$$

Several expressions need to be calculated for each time step of exponential integration. Therefore, the efficient solution of these expressions directly determines the calculation efficiency of the exponential integral.

When dealing with singular matrices or matrices with eigenvalues close to zero, various strategies are adopted to ensure the robustness and accuracy of the proposed method. First of all, for singular matrix, we study to improve the condition number of the matrix by introducing small regularization terms, which can make the matrix invertible without significantly affecting the characteristics of the matrix. In addition, for matrices with eigenvalues close to zero, we adopt preprocessing technology to improve the numerical stability of the matrix by appropriate transformation. In practical applications, these challenges can lead to
numerical instability in the calculation process, especially when performing calculations of matrix exponential functions and vector products. Therefore, the adaptive selection of eigenvalues by the Krylov subspace method is used to avoid calculation errors caused by eigenvalues approaching zero.

In the past 20 years, the calculation of exponential matrix function and vector product has attracted the attention of many scholars, especially the calculation of the matrix exponential function and vector product. As pointed out earlier, Krylov subspace method is one of the most effective methods to solve large exponential matrix functions [14]. Many scholars at home and abroad are devoted to the application research of this method in matrix index, including the design and analysis of algorithm, the improvement of W and standard subspace method, etc. At present, the Krylov subspace has been solved by a wealth of algorithm skills and theoretical analysis, but for the general form of the problem, the relevant research is relatively less.

In Section 2.3, the calculation error of the matrix exponential function and vector product is analyzed in detail. The initial value problem of the matrix exponential function is given by:

\[ y(t) = \varphi_0(tA)b_0 + \sum_{j=0}^{p-1} t^j \varphi_{j+1}(tA)b_{j+1}, \quad (38) \]
\[ y'(t) = Ay(t) + \sum_{j=0}^{p-1} t^j b_{j+1}, \quad y(0) = b_0. \quad (39) \]

Similar to the standard block Krylov subspace method, this study still use the aforementioned relationship to complete the construction.

Without losing generality, the initial value problem with zero initial condition is considered:

\[ y'(t) = Ay(t) + \sum_{j=0}^{p-1} t^j b_{j+1}, \quad y(0) = 0. \quad (40) \]

Suppose that \( B = [b_1, b_2, \ldots, b_p] \), \( P(t) = \left[ 1, t, \frac{t^2}{2!}, \ldots, \frac{t^{p-1}}{(p-1)!} \right]^T \), the matrix exponential function can be written in the form of matrix as follows:

\[ y'(t) = Ay(t) + BP(t), \quad y(0) = 0. \quad (41) \]

Multiply both sides of the matrix empirical function \( y'(t) \) by \( \hat{A} = (I + yA)^{-1} \) at the same time, and there is

\[ \hat{A}y'(t) = \frac{(I - A)}{y}y(t) + \hat{A}BP(t), \quad y(0) = 0. \quad (42) \]

where \( y \) is predetermined.

QR decomposition of matrix \( B : B = V_lR \), where \( V_l \) is a \( n \times p \)-order orthogonal normal matrix, and \( R \) is a \( p \)-order upper triangular matrix. Applying block calculation to Krylov’s subspace \( K_{nl}(A, V_l) \), a set of orthogonal bases of linear space \( K_{nl}(A, V_l) \) can be generated, which satisfies the product relationship between the matrix empirical function and the vector:

\[ (I + yA)^{-1} V_m = V_m H_m + V_{m+1} H_{m+1}, \quad E_m^T = V_m^T H_m, \quad (43) \]

where \( V_m = [V_1, V_2, \ldots, V_m] \in \mathbb{R}^{mp \times p} \), \( E_m \in \mathbb{R}^{mp \times p} \), and \( H_m = (H_m) \) is a matrix generated by the first \( mp \) lines of \( H_m \) on a \( (m + 1)p \times mp \)-order block.

It should be noted that when the algorithm of the matrix exponential function and vector product is applied to a matrix \( (I + yA)^{-1} \), the product of the matrix exponential function and vector function of (44) needs to be calculated in each step:

\[ (I + yA)^{-1} V. \quad (44) \]

That is to solve the linear combination of the following formula:

\[ (I + yA)x = V. \quad (45) \]

In this study, a sparse solution is used to calculate the product of the matrix exponential function and the vector. In the entire calculation process, only one calculation is required.

Suppose that \( T_m = \frac{(H_m^{-1} - I)}{y} \), then the matrix empirical function is rewritten as:

\[ V_m T_m = A V_m + \left( \frac{(I + yA)}{y} \right) V_m H_m + E_m^T H_m^{-1}. \quad (46) \]

In summary, the block Krylov subspace method is used to solve the product of exponential matrix functions and vectors, the error of the algorithm is analyzed, and a simple and reliable posterior error estimate is established. In order to keep the dimension of the Krylov subspace within a reasonable range, the block Krylov subspace method based on time step is introduced. By optimizing the time step \( h \) and the dimension \( m \) of the Krylov subspace, the minimum workload of the algorithm can meet the accuracy requirements to complete the calculation [15]. In addition, in order to solve the problem of potential numerical instability in the calculation of large matrices, the proposed method adopts a variety of techniques to ensure stability. In particular, when dealing with matrices with specific properties such as sparsity or symmetry, algorithms adapted to these properties are employed to reduce computational errors and enhance stability. For sparse matrix, Krylov subspace method combined with preprocessing technology is studied, which can not only effectively use the sparse structure of matrix to accelerate the calculation, but also reduce the numerical instability caused by
the large number of matrix conditions. For symmetric matrices, the numerical integration scheme of symmetry preservation is adopted to ensure that the symmetry of the matrix is maintained in the integration process, which is crucial to maintain the stability and accuracy of the algorithm.

### 3 Results

The validity of the calculation method of the matrix exponential function and vector product is verified by numerical experiments. All experiments were performed on a laptop computer with an operating system of Windows 8 (Intel core i5, 2.6 GHz, and 6 GB of memory) using MATLAB software.

The relative errors used in the experiment are as follows:

$$\text{Error} = \frac{\|y - y_m\|}{\|y\|},$$  

(47)

where $y$ is the exact solution, which is the numerical solution. If there is no special explanation, the exact solution of each experiment is always calculated by the function ode45 of MATLAB.

Relative posterior error estimation is also used in error estimation. For convenience, this study still uses $\varepsilon^1_m$, $\varepsilon^2_m$, $\varepsilon^3_m$, and $\varepsilon^4_m$ to represent the relative form of posterior error estimation, namely,

$$\varepsilon^i_m = \frac{\varepsilon^i_m}{\|y\|}, \quad y = 1, 2, 3, 4.\quad (48)$$

For calculations of medium and small ($n < 1000$) matrix exp ($H$), use MATLAB’s own function expm to directly solve.

This experiment is mainly to test the effectiveness of backward error estimation. To calculate the value of $\varphi_i(tA)b_0 + \varphi_i(tA)b_1 + \cdots + \varphi_i(tA)b_5$ at $t = \pm 1$ using the algorithm of the matrix exponential function and vector product, this study chose $b_0 = 0$, $b_i = e_i$, and $i = 1, 2, 3, 4, 5$. In this experiment, two commonly used test matrices are selected. The first matrix is a Markov chains matrix, from the package Expokit, recorded as cl024. The matrix is an asymmetric matrix with order $n = 1.024$ and nnz = 11,264 non-zero factors. The second matrix is a block-symmetric triangular matrix of 9801-order, which is discretely obtained by a two-dimensional Laplacian operator through standard finite difference methods, and can be generated by MATLAB commands [16]. In determining the convergence of the proposed numerical scheme, a strict set of criteria is used, including the analysis of the local truncation error, the monitoring of the spectral radius of the matrix, and the evaluation of the stability of the algorithm through various test cases. These criteria have been carefully chosen to ensure that they are consistent with the fundamental mathematical properties of matrices, exponential functions, and vector products. The robustness of the numerical scheme is reflected in its ability to consistently provide accurate results on different sets of matrices, including those with properties such as sparsity and symmetry, and even non-square matrices. This robustness is critical to the reliability of the proposed method and has been verified by a large number of numerical experiments. Therefore, convergence criteria are integral to the credibility of research methods, as they guarantee that the results are not only accurate but also stable, even with the potential numerical challenges inherent in large-scale matrix computation.

First, the problem of calculating matrix exponential function and vector product is defined as an abstract initial value problem. In this framework, matrix exponential functions are treated as generators of analytic semigroups, while vector products correspond to initial values. Using the theory of analytic semigroups, we can ensure the existence and uniqueness of analytic solutions when time tends to infinity. Next, by introducing B-series theory and two-color tree theory, we study the extraction of sequence expressions from numerical formats, in which the specific properties of matrices (such as sparsity and symmetry) are considered. The application of these theories allows accurate control of local truncation errors of numerical schemes, thus ensuring the convergence of numerical solutions. In addition, in order to verify the convergence of the numerical scheme, the stability and consistency conditions satisfied by the numerical scheme are analyzed. The stability condition ensures the stability of the numerical solution over time, while the consistency condition ensures that the error between the numerical solution and the analytical solution decreases as the step size decreases. By combining these theoretical analyses and conditions, the convergence of the numerical scheme is established: the proposed numerical scheme can converge to the analytical solution as long as the appropriate stability and consistency conditions are met. Figure 1 shows a graph of the true relative error of the first test matrix and the function of the relative posterior error estimate with respect to the number of iteration steps. The test results of the second test matrix are shown in Figure 2. It can be seen from the figure that first, in all cases, the algorithm of the matrix exponential function and vector product has a super-linear convergence speed and has completed a high calculation accuracy, reaching about $10^{14}$. Second, the extracted backward error estimation is also very effective, and it is basically completely consistent with the change trend of the real error. In some cases, these error estimates almost coincide with real errors. Among them, the error estimation is more effective, and the
modified algorithm of the matrix exponential function and vector product has slightly higher accuracy, which is completely consistent with the previous discussion. It is worth pointing out that when $t = -1$, it is $< 10$ in the first few iteration steps, and the error estimation performance is not ideal. This is mainly because the vectors generated in the first few steps contain less effective information. The true error is almost horizontal in the first few steps, without noticeable changes. As the number of iteration steps increases, the effective information carried by the vector also gradually increases, and the error estimate gradually approaches the true error, which accurately reflects the changing characteristics of the true error. Similar phenomena have appeared in some other calculations of the matrix exponential function. In the initial iteration process, especially when the numerical solution does not change significantly, some more effective error estimates can be constructed to be used in conjunction. In this experiment, two commonly used test matrices are selected. The first matrix is a Markov chain matrix from the package exokit, recorded as c1024. The matrix is an asymmetric matrix with an order of $n = 1.024$ with $nnz = 1,264$ nonzero vultures. The second matrix is a block symmetric triangular matrix of order 9801, which is derived from a two-dimensional Laplacian operator discretized by a standard finite difference method and can be generated by the matlab command. Using [1] algorithm and this algorithm to test the error, and compare its error, the results are shown in Figures 1–4.

In Figures 1 and 2, the function image of [1] algorithm and the algorithm in this paper for the true relative error of the first test matrix as well as the relative posterior error estimate for the number of iterative steps. The test fruit for the second test matrix is placed in Figures 3 and 4. It can be seen in the figure: first of all, in all cases, [1] algorithm and the present algorithm have a superlinear convergence speed, all of which have completed a higher
calculation accuracy of about 10−12; second, the extracted backward error estimation is also very effective, which is completely consistent with the actual error changing trend. In some cases, these error estimates are almost and the actual error is heavy. In this study, the source of error in matrix exponential function and vector product calculation methods is analyzed in detail, especially the error introduced by exponential function series truncation of matrix $T$. Through careful selection of the truncated series, the error caused by the truncated series is minimized. The experimental results show that although the series truncation will introduce some errors, this error can be effectively controlled by carefully selecting the truncation points and will not have a significant impact on the accuracy of the final calculation results. In addition, in practical applications, the propagation of errors is mainly affected by the cumulative effect of errors during calculation. To this end, the research adopts a variety of strategies to reduce error propagation, including the use of high-precision numerical integration schemes and optimization algorithms, which can reduce the accumulation of errors during iteration. Therefore, experimental verification ensures the stability and reliability of the proposed method in a wide range of application scenarios.

Four 10,000-order matrices were selected to compare the proposed method with the existing advanced algorithms, including [1,5] and [6], to verify the accuracy and computational complexity of the algorithm. The results are shown in Table 1.

As can be seen from the table, in the comparison results of estimation errors, [1] the calculation method is quite different from the matrix exponential function and vector product methods proposed in the research. The stability of the proposed method is improved by 13.6% compared with the algorithm [1]. In addition, the calculation method of matrix exponential function and vector product can reduce the calculation error of matrix exponential function and vector product. The experimental results prove the validity of this study.

### 4 Conclusions

The exponential function method has a long history, but it is considered to be impractical for a long time. The need to compute the product of the dimension of the exponential matrix and the dimension of the vector is the main reason. In recent years, with the development of numerical algebra, especially the emergence of related algorithms for solving matrix functions and vector products based on the Krylov subspace method, the effective implementation of the exponential integration has become possible. First, the exponential integration of the problem is solved, and an exponential integration is constructed to solve the initial value of the large ordinary differential equation with a right-end term composed of a combination of non-linear functions with different characteristics. The general method for extracting these format order conditions is given by the two-color tree theory, and then, several specific numerical integration formats are constructed. Numerical experiments verify that these formats are valid. In addition, the numerical theoretical orders are basically the same. It is worth noting that the main advantage of this method is its robustness and accuracy when solving large-scale problems. This method is especially effective for equations composed of various non-linear functions and has practical applicability in exponential integration of matrix functions. However, while the computational overhead has been shown to be minimal, the method does have its limitations. One of the limitations is its potential sensitivity to the choice of parameters and initial conditions, which can affect the speed and accuracy of the solution convergence. Although the method has been shown to be effective, the study was limited by experimental

<table>
<thead>
<tr>
<th>Algorithm name</th>
<th>Error</th>
<th>Time</th>
<th>Error</th>
<th>Time</th>
<th>Error</th>
<th>Time</th>
<th>Error</th>
<th>Time</th>
<th>Error</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1] Algorithm</td>
<td>$8.11 \times 10^{-12}$</td>
<td>0.28 s</td>
<td>$1.19 \times 10^{-11}$</td>
<td>7.95 s</td>
<td>$1.80 \times 10^{-11}$</td>
<td>0.69 s</td>
<td>$7.85 \times 10^{-12}$</td>
<td>1.15 s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Algorithm of this paper</td>
<td>$7.47 \times 10^{-12}$</td>
<td>0.27 s</td>
<td>$2.26 \times 10^{-11}$</td>
<td>0.16 s</td>
<td>$1.02 \times 10^{-11}$</td>
<td>0.18 s</td>
<td>$3.92 \times 10^{-13}$</td>
<td>0.65 s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[5] Algorithm</td>
<td>$2.84 \times 10^{-11}$</td>
<td>0.93 s</td>
<td>$4.14 \times 10^{-10}$</td>
<td>5.64 s</td>
<td>$1.63 \times 10^{-10}$</td>
<td>1.64 s</td>
<td>$4.24 \times 10^{-12}$</td>
<td>2.41 s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[6] Algorithm</td>
<td>$5.28 \times 10^{-11}$</td>
<td>1.53 s</td>
<td>$7.53 \times 10^{-10}$</td>
<td>4.65 s</td>
<td>$2.05 \times 10^{-10}$</td>
<td>3.41 s</td>
<td>$1.34 \times 10^{-12}$</td>
<td>4.24 s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reference [1] Algorithm</td>
<td>F.c</td>
<td></td>
<td>F.c</td>
<td></td>
<td>F.c</td>
<td></td>
<td>F.c</td>
<td></td>
<td>F.c</td>
<td></td>
</tr>
<tr>
<td>Algorithm of this paper</td>
<td>$9.60 \times 10^{-11}$</td>
<td>0.42 s</td>
<td>$1.83 \times 10^{-11}$</td>
<td>0.26 s</td>
<td>$5.27 \times 10^{-12}$</td>
<td>0.28 s</td>
<td>$9.34 \times 10^{-13}$</td>
<td>1.33 s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[5] Algorithm</td>
<td>F.c</td>
<td></td>
<td>F.c</td>
<td></td>
<td>F.c</td>
<td></td>
<td>F.c</td>
<td></td>
<td>F.c</td>
<td></td>
</tr>
</tbody>
</table>
conditions and did not test its performance on different computational conditions such as high-performance computing systems, so adjustments may be required when applied to certain types of matrices that exhibit special spectral properties or when dealing with boundary conditions that deviate from standard assumptions. Future research will focus on the aforementioned aspects. In addition, regarding the inherent limitations of series truncation of exponential functions for a matrix T, the study realizes that truncation may introduce numerical errors. Although the appropriate truncation order is used to minimize the error, the number of series terms required may increase when the spectral radius of the matrix is large, thus affecting the computational efficiency. The numerical experiments of the study show that for most cases, this error is manageable and does not have a significant impact on the accuracy of the solution. Future work will explore strategies to reduce such errors.

Funding information: None.

Author contributions: Lihui Tu wrote original draft; Huanjie Hong conducted data analysis and method guidance.

Conflict of interest: The authors declare that they have no conflict of interest.

Data availability statement: The data that support the findings of this study are available from the corresponding author upon reasonable request. I confirm that all images are original.

References


