Journal of Scientific Computing (2023) 96:70
https://doi.org/10.1007/s10915-023-02287-2

Solving Time-Dependent PDEs with the Ultraspherical Spectral Method

Lu Cheng[1] · **Kuan Xu**[1]

Received: 13 February 2023 / Revised: 13 June 2023 / Accepted: 30 June 2023 /
Published online: 20 July 2023

Abstract

We apply the ultraspherical spectral method to solving time-dependent PDEs by proposing two approaches to discretization based on the method of lines and show that these approaches produce approximately same results. We analyze the stability, the error, and the computational cost of the proposed method. In addition, we show how adaptivity can be incorporated to offer adequate spatial resolution efficiently. Both linear and nonlinear problems are considered. We also explore time integration using exponential integrators with the ultraspherical spatial discretization. Comparisons with the Chebyshev pseudospectral method are given along the discussion and they show that the ultraspherical spectral method is a competitive candidate for the spatial discretization of time-dependent PDEs.

Keywords Spectral method · Time-dependent PDEs · Chebyshev polynomials · Ultraspherical polynomials

Mathematics Subject Classification 65L04 · 65M12 · 65M15 · 65M20 · 65M70

1 Introduction

In this article, we consider the one-dimension time-dependent PDE

$$\mathcal{T}u = \mathcal{F}(t, u(x,t)), \tag{1a}$$

$$\text{s.t.} \quad \mathcal{B}u = c, \tag{1b}$$

$$u(x,0) = f(x), \tag{1c}$$

where $\mathcal{T}$ is the first-order differential operator in time. $\mathcal{F}$ is a spatial operator that acts on the time t and the solution $u(x,t)$. For a fixed t, $u(x,t)$ becomes a univariate function of the

✉ Kuan Xu
kuanxu@ustc.edu.cn

[1] School of Mathematical Sciences, University of Science and Technology of China, 96 Jinzhai Road, Hefei 230026, Anhui, China

spatial variable x defined on $[-1, 1]$. $\mathcal{F}(t, u(x, t))$ can be further decomposed as

$$\mathcal{F}(t, u(x, t)) = \mathcal{L}u + \mathcal{N}(t, u(x, t)), \tag{2}$$

where $\mathcal{L}$ and $\mathcal{N}$ are the linear and nonlinear parts, respectively. Throughout this article, we follow the convention of writing $\mathcal{L}u$, instead of $\mathcal{L}(u)$, for $\mathcal{L}$ is linear. Without loss of generality, we assume that $\mathcal{L}$ is an Nth order linear differential operator in space for $x \in [-1, 1]$

$$\mathcal{L} = a^N(x)\frac{\mathrm{d}^N}{\mathrm{d}x^N} + \cdots + a^1(x)\frac{\mathrm{d}}{\mathrm{d}x} + a^0(x) \tag{3}$$

with $a^N(x) \neq 0$ so that $\mathcal{L}$ is non-singular. The side conditions $\mathcal{B}$ contains N linear functionals which are boundary conditions or constraints of other sorts and c is an N-vector. The function $f(x)$ gives the initial condition.

In [18] Olver and Townsend present a fast and stable spectral method enabled by the ultraspherical polynomials which solves linear ordinary differential equations of the form

$$\mathcal{L}u = g, \tag{4a}$$

$$\text{s.t. } \mathcal{B}u = c, \tag{4b}$$

where $\mathcal{L}$ is also defined as in (3). This ultraspherical spectral method assumes the solution is written in its Chebyshev expansion

$$u(x) = \sum_{k=0}^{\infty} u_k T_k(x),$$

where $T_k(x)$ is the Chebyshev polynomial of degree k. This way, $u(x)$ is identified by the coefficient vector $\boldsymbol{u} = [u_0, u_1, \ldots]^T$. With a change of basis, the λ-order differentiation operator is as sparse as

$$\mathcal{D}_\lambda = 2^{\lambda-1}(\lambda-1)! \begin{pmatrix} \overbrace{0 \cdots 0}^{\lambda \text{ times}} & \lambda & & & \\ & & \lambda+1 & & \\ & & & \lambda+2 & \\ & & & & \ddots \end{pmatrix}, \tag{5}$$

for $\lambda = 1, 2, \ldots$, where $\mathcal{D}_\lambda$ maps Chebyshev coefficients to ultraspherical $C^{(\lambda)}$ coefficients.[1]

If any of $a^\lambda(x)$ in (3) is not constant and written as

$$a^\lambda(x) = \sum_{k=0}^{\infty} a_j C_j^{(\lambda)}(x),$$

the differential operator $\mathcal{D}_\lambda$ should be pre-multiplied by the multiplication operator whose (j, k) entry reads

$$\mathcal{M}_\lambda[a^\lambda]_{j,k} = \sum_{s=\max(0,k-j)}^{k} a_{2s+j-k} c_s^\lambda(k, 2s+j-k) \tag{6}$$

[1] In [18], $\mathcal{D}_0 = \mathcal{D}_1$, while in this paper we let $\mathcal{D}_1$ maps from Chebyshev T to C^1 and $\mathcal{D}_0 = \mathcal{I}$, i.e., the identity operator, for notational consistency.

for $j, k \geq 0$, where

$$c_s^\lambda(j,k) = \frac{j+k+\lambda-2s}{j+k+\lambda-s}\frac{(\lambda)_s(\lambda)_{j-s}(\lambda)_{k-s}}{s!(j-s)!(k-s)!}\frac{(2\lambda)_{j+k-s}}{(\lambda)_{j+k-s}}\frac{(j+k-2s)!}{(2\lambda)_{j+k-2s}}.$$

Note that $\mathcal{M}_\lambda[a^\lambda]$ maps the ultraspherical space of $C^{(\lambda)}$ to itself. As long as $a^\lambda(x)$ possesses certain smoothness, it can be approximated by a finite series, that is,

$$a^\lambda(x) \approx \sum_{k=0}^{m} a_j C_j^{(\lambda)}(x).$$

This way, $\mathcal{M}_\lambda[a^\lambda]$ becomes banded since $a_j = 0$ for $j > m$. Another approach to calculating the entries of $\mathcal{M}_\lambda[a^\lambda]$ is given in [26], based on a recurrence relation for the multiplication operator.

When $\mathcal{D}_\lambda$ and $\mathcal{M}_\lambda[a^\lambda]$ are employed, each term in (3) maps to a different ultraspherical basis. So the following conversion operators $\mathcal{S}_\lambda$ are needed to map the coefficients in T to those in $C^{(1)}$ or $C^{(\lambda)}$ to $C^{(\lambda+1)}$ respectively

$$\mathcal{S}_0 = \begin{pmatrix} 1 & & -\frac{1}{2} & & & \\ & \frac{1}{2} & & -\frac{1}{2} & & \\ & & \frac{1}{2} & & -\frac{1}{2} & \\ & & & \ddots & & \ddots \end{pmatrix}, \tag{7a}$$

$$\mathcal{S}_\lambda = \begin{pmatrix} 1 & & -\frac{\lambda}{\lambda+2} & & & \\ & \frac{\lambda}{\lambda+1} & & -\frac{\lambda}{\lambda+3} & & \\ & & \frac{\lambda}{\lambda+2} & & -\frac{\lambda}{\lambda+4} & \\ & & & \ddots & & \ddots \end{pmatrix} \text{ for } \lambda \geq 1. \tag{7b}$$

In terms of (5), (6), and (7), the differential equation (4a) can be represented as

$$\left(\mathcal{M}_N[a^N]\mathcal{D}_N + \sum_{\lambda=0}^{N-1} \mathcal{S}_{N-1}\ldots\mathcal{S}_\lambda\mathcal{M}_\lambda[a^\lambda]\mathcal{D}_\lambda\right)\boldsymbol{u} = \mathcal{S}_{N-1}\ldots\mathcal{S}_0\boldsymbol{g}, \tag{8}$$

where $\boldsymbol{g}$ is the vector containing the Chebyshev coefficients of $g(x)$. To make (8) of finite dimension, the operators are truncated by the projection operator $\mathcal{P}_n = (I_n, \mathbf{0})$, where I_n is the $n \times n$ identity matrix, with the dimension n properly chosen. The truncated version of (8) reads

$$\begin{aligned} \mathcal{P}_{n-N}\left(\mathcal{M}_N[a^N]\mathcal{D}_N + \sum_{\lambda=0}^{N-1} \mathcal{S}_{N-1}\ldots\mathcal{S}_\lambda\mathcal{M}_\lambda[a^\lambda]\mathcal{D}_\lambda\right)\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u} \\ = \mathcal{P}_{n-N}\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{g}, \end{aligned} \tag{9}$$

where the unknown $\mathcal{P}_n\boldsymbol{u}$ and the (unconverted) right-hand side $\mathcal{P}_n\boldsymbol{g}$ are n-vectors and the differential operators on the left-hand side and the product of the conversion operators on the right-hand side are approximated by their truncated version of dimension $(n-N) \times n$ via *exact truncation*. The system (9) is finally squared up to form an $n \times n$ system by the first n columns of the discretized version of the boundary conditions (4b) and this is the system by solving which one obtains the Chebyshev coefficients u_k of the truncated version of the solution

$$\widetilde{u}_n(x) = \sum_{k=0}^{n-1} u_k T_k(x).$$

The ultraspherical spectral method recapitulated above enjoys a few important advantages over the collocation-based pseudospectral methods, including linear computational complexity, good conditioning, and adaptivity via optimal truncation.

In this article, we extend the ultraspherical spectral method to the solution of the time-dependent problem (1) within the method of lines (MOL) framework. Our investigation is by no means the first attempt to solve time-dependent PDEs by the ultraspherical spectral method. In [27], Townsend and Olver describe an extension of the ultraspherical spectral method to two spatial dimensions for the solution of linear PDEs with variable coefficients defined on bounded rectangular domains and their focus is on the automated manner of solution provided that the splitting rank of the partial differential operator (PDO) is known. When applied to an initial boundary value problem, this bivariate ultraspherical spectral method treats it as a boundary value problem of two spatial dimensions by deeming the time variable as a second spatial variable. Our motivation in this article, however, is to employ the ultraspherical spectral method in space while do the time-stepping using common time integration schemes. Moreover, we consider a more general setting where the problem may or may not have a sufficiently concise closed-form description or the spatial operator can only be evaluated via black-box routines, which is often the case in real-world problems.

The first and probably only existing works where the ultraspherical spectral method is used in conjunction with time-stepping schemes may be [9, 24], where the implicit-explicit method and the backward Euler method are employed, respectively. However, the application of these time-stepping schemes are not theoretically analyzed to give insights on their performance. On the software side, the DEDALUS package solves time-dependent PDEs using (a first-order variant of) the ultraspherical spectral method with the time-integration done by a range of ODE integrators including multistep and Runge–Kutta IMEX methods [3]. The success of these attempts suggests a pressing demand on the theoretical analysis of time stepping when the ultraspherical spectral method is used for the spatial discretization. This is exactly what the present article focuses on. By giving a rather complete treatment to solving time-dependent PDEs in one spatial dimension, this article may well serve as a foundation for migration to problems in higher spatial dimensions.

In the first part of this article, we concentrate on the linear case of (1), i.e.,

$$\mathcal{T}u = \mathcal{L}u, \tag{10a}$$

$$\text{s.t.} \quad \mathcal{B}u = c, \tag{10b}$$

$$u(x, 0) = f(x), \tag{10c}$$

by discussing the discretization of (10) via standard time stepping schemes (Sect. 2) and analyzing the stability (Sect. 3), the error (Sect. 4), and the computational cost (Sect. 5). The stepping nature of the method enables an adaptive implementation which we describe in Sect. 6. In the linear regime, our discussion will make frequent use of the one-dimensional transport equation

$$u_t(x, t) = u_x(x, t), \tag{11a}$$

$$u(1, t) = 0 \tag{11b}$$

and the heat equation

$$u_t(x,t) = u_{xx}(x,t), \tag{12a}$$

$$u(-1,t) = u(1,t) = 0, \tag{12b}$$

both subject to the initial condition $u(x,0) = f(x)$. Also, we shall simply take $f(x) = \exp(-200x^2)$ and $f(x) = \sin(2\pi x)$ for (11) and (12), respectively. In the study of the collocation-based pseudospectral method, much attention has been paid to these problems from various perspectives, particularly regarding the stability restrictions on time stepping and the eigenvalue distribution of the spatial discretization operators, see, e.g., [7, 11, 25, 34].

We close our discussion in the linear regime by briefly analyzing the problems with periodic boundary conditions (Sect. 7). The collocation-based pseudospectral method, for many years, has been taken as 'the' method, and the discussion and analysis for the linear case facilitate the comparison between the two methods. In addition, they lay the foundation for the analysis of nonlinear time-dependent problems (Sect. 8). In Sect. 9, we examine the application of the exponential integrator in conjunction with the ultraspherical spectral method. Conclusion and discussion are given in the final section.

Throughout this article, all the norms are taken to be the infinity norm. Calligraphy font is used for operators or infinite matrices and bold fonts for infinite vectors, whereas the truncated version of operators, infinite matrices, and vectors are in normal fonts.

All the numerical experiments in this article are performed in JULIA v1.5.3 on a desktop with a 4 core 2.1 Ghz AMD Ryzen 5 3500U CPU.

2 Discretization

We start by considering the discretization of (10). Suppose that the solution $u(x,t)$ is written as an infinite Chebyshev series

$$u(x,t) = \sum_{k=0}^{\infty} u_k(t) T_k(x),$$

where the coefficients $u_k(t)$ we are solving for are now dependent of time t. If the spatial operator $\mathcal{L}$ on the right-hand side of (10a) is expressed in terms of the operators reviewed in Sect. 1 as

$$\mathcal{L} = \mathcal{M}_N\left[a^N\right]\mathcal{D}_N + \sum_{\lambda=0}^{N-1} \mathcal{S}_{N-1}\cdots\mathcal{S}_\lambda\mathcal{M}_\lambda\left[a^\lambda\right]\mathcal{D}_\lambda, \tag{13}$$

the left-hand side of (10a) must be pre-multiplied by a series of conversion operators so that both the sides end up being in the $C^{(N)}$ basis, that is,

$$\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{T}\boldsymbol{u} = \mathcal{L}\boldsymbol{u}, \tag{14}$$

where $\boldsymbol{u} = [u_0(t), u_1(t), \ldots]^T$ is the infinite vector collecting the coefficients $u_k(t)$.

Now we bifurcate our discussion by presenting two ways to further discretize (14) and enforce the boundary condition (10b), both following the method of lines. They differ in how a square system is formed by solving which we obtain a truncated approximation to $\boldsymbol{u}$.

In the remainder of this article, we confine our discussion about the discretization of the temporal operator $\mathcal{T}$ to the standard time marching schemes for solving the ODE initial value problem $v_t = f(t, v)$. That is, we consider the linear multistep methods

$$\sum_{j=0}^{r} \alpha_j v^{k+j} = h \sum_{j=0}^{r} \beta_j f^{k+j}, \tag{15}$$

where $\alpha_r = 1$, and the explicit Runge–Kutta methods

$$y_j = hf(t_k + \theta_j h, v^k + \mu_j y_{j-1}), \text{ for } j = 1, 2, \cdots, s \tag{16a}$$

$$v^{k+1} = v^k + \sum_{j=1}^{s} \gamma_j y_j, \tag{16b}$$

where $\theta_1 = \mu_1 = 0$ and $\sum_{j=1}^{s} \gamma_j = 1$. In (15) and (16), h is the step size.

2.1 Approach 1

Our first approach enforces the main equation and the boundary conditions simultaneously. To this end, we truncate the operators and $\boldsymbol{u}$

$$\mathcal{P}_{n-N}\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{T}\mathcal{P}_n\boldsymbol{u} = \mathcal{P}_{n-N}\mathcal{L}\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}, \tag{17}$$

which amounts to taking the first $n - N$ rows and the first n columns of $\mathcal{S}_{N-1}\dots\mathcal{S}_0$ and $\mathcal{L}$ and approximating the solution by its n-term truncation

$$u_n(x, t) \approx \widetilde{u}_n(x, t) = \sum_{k=0}^{n-1} u_k(t)T_k(x).$$

Note that the truncation of $\mathcal{P}_{n-N}\mathcal{L}\mathcal{P}_n^\top$ is done *exactly* as

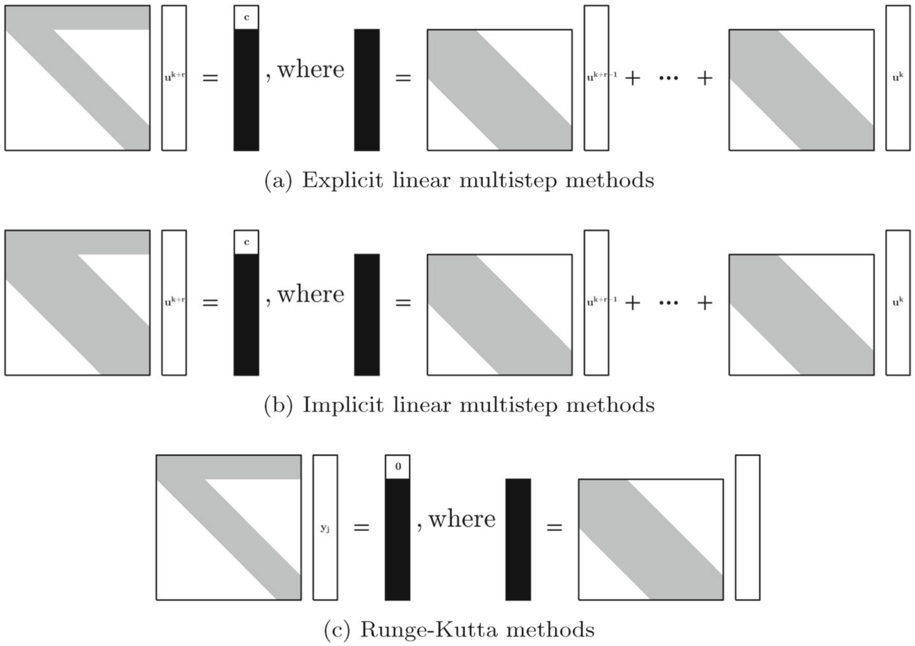
$$\begin{aligned}
\mathcal{P}_{n-N}\mathcal{L}\mathcal{P}_n^\top =& \mathcal{P}_{n-N}\left(\mathcal{M}_N[a^N]\mathcal{D}_N + \sum_{\lambda=0}^{N-1}\mathcal{S}_{N-1}\dots\mathcal{S}_\lambda\mathcal{M}_\lambda[a^\lambda]\mathcal{D}_\lambda\right)\mathcal{P}_n^\top \\
&= \left(\mathcal{P}_{n-N}\mathcal{M}_N[a^N]\mathcal{P}_{n-N}^\top\right)(\mathcal{P}_{n-N}\mathcal{D}_N\mathcal{P}_n) + \sum_{\lambda=0}^{N-1}\left(\mathcal{P}_{n-N}\mathcal{S}_{N-1}\mathcal{P}_{n-N+2}^\top\right) \\
&\quad \times \left(\prod_{i=2}^{N-\lambda}\mathcal{P}_{n-N+2(i-1)}\mathcal{S}_{N-i}\mathcal{P}_{n-N+2i}^\top\right)\left(\mathcal{P}_{n-N+2(N-\lambda)}\mathcal{M}_\lambda[a^\lambda]\mathcal{P}_{n-\lambda}^\top\right) \\
&\quad \times \left(\mathcal{P}_{n-\lambda}\mathcal{D}_\lambda\mathcal{P}_n^\top\right).
\end{aligned}$$

For exact truncations of operators, see [18, Remark 2] for details.

We truncate the operators in the boundary conditions analogously by taking the first n columns of $\mathcal{B}$

$$\mathcal{B}\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u} = c. \tag{18}$$

When (18) is laid on the top of (17), an $n \times n$ square system is formed despite that the temporal operator is not yet discretized.



(a) Explicit linear multistep methods

(b) Implicit linear multistep methods

(c) Runge-Kutta methods

Fig. 1 Sparsity patterns of the fully discretized systems in Approach 1 for linear multistep methods (21) and Runge–Kutta methods (22). **a**, **b** mainly differ in the matrix on the left-hand side in that the lower bandwidth of its banded part is zero for explicit schemes whereas the banded part could have nonzero sub-diagonals for implicit schemes

Now we turn to the discretization in time. When a multistep method is applied to (17), we have

$$\sum_{j=0}^{r} \alpha_j \mathcal{P}_{n-N} \mathcal{S}_{N-1} \dots \mathcal{S}_0 \mathcal{P}_n^\top \mathcal{P}_n \boldsymbol{u}^{k+j} = h \sum_{j=0}^{r} \beta_j \mathcal{P}_{n-N} \mathcal{L} \mathcal{P}_n^\top \mathcal{P}_n \boldsymbol{u}^{k+j}.$$

or, equivalently,

$$\begin{aligned}&(\mathcal{P}_{n-N} \mathcal{S}_{N-1} \dots \mathcal{S}_0 \mathcal{P}_n^\top - h\beta_r \mathcal{P}_{n-N} \mathcal{L} \mathcal{P}_n^\top) \mathcal{P}_n \boldsymbol{u}^{k+r} \\ &\quad = h \sum_{j=0}^{r-1} \beta_j \mathcal{P}_{n-N} \mathcal{L} \mathcal{P}_n^\top \mathcal{P}_n \boldsymbol{u}^{k+j} - \sum_{j=0}^{r-1} \alpha_j \mathcal{P}_{n-N} \mathcal{S}_{N-1} \dots \mathcal{S}_0 \mathcal{P}_n^\top \mathcal{P}_n \boldsymbol{u}^{k+j}.\end{aligned} \tag{19}$$

Here, $\boldsymbol{u}^k = [u_0(t_k), u_1(t_k), \dots]^T$ is the approximate solution at kth time step, and $\mathcal{P}_n \boldsymbol{u}^k$ is the n-vector with the trailing coefficients dropped.

When a Runge–Kutta method is applied to (17), each stage becomes

$$\mathcal{P}_{n-N} \mathcal{S}_{N-1} \dots \mathcal{S}_0 \mathcal{P}_n^\top y_j = h \mathcal{P}_{n-N} \mathcal{L} \mathcal{P}_n^\top (\mathcal{P}_n \boldsymbol{u}^k + \mu_j y_{j-1}) \tag{20}$$

for $j = 1, 2, \cdots, s$. Note that y_j is a finite vector (see (16a)).

We are finally in a position to form the fully discretized square system. Stacking the boundary conditions (18) and the main equation (19) gives

$$\begin{aligned}&\begin{pmatrix}\mathcal{B}\mathcal{P}_n^\top\\ \mathcal{P}_{n-N}\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top-h\beta_r\mathcal{P}_{n-N}\mathcal{L}\mathcal{P}_n\end{pmatrix}\mathcal{P}_n\boldsymbol{u}^{k+r}\\&=\begin{pmatrix}c\\ \sum_{j=0}^{r-1}(\beta_jh\mathcal{P}_{n-N}\mathcal{L}\mathcal{P}_n^\top-\alpha_j\mathcal{P}_{n-N}\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top)\mathcal{P}_n\boldsymbol{u}^{k+j}\end{pmatrix},\end{aligned}\tag{21}$$

by solving which we have $\mathcal{P}_n\boldsymbol{u}^{k+r}$. The sparsity structure of (21) is shown by Fig. 1a, b for explicit and implicit multistep methods, respectively.

When (20) is combined with the boundary conditions, we obtained a square system for the intermediate solutions y_j at each stage of a Runge–Kutta method

$$\begin{pmatrix}\mathcal{B}\mathcal{P}_n^\top\\ \mathcal{P}_{n-N}\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top\end{pmatrix}y_j=\begin{pmatrix}0\\ h\mathcal{P}_{n-N}\mathcal{L}\mathcal{P}_n^\top(\mathcal{P}_n\boldsymbol{u}^k+\mu_jy_{j-1})\end{pmatrix},\tag{22}$$

where $j=1,2,\cdots,s$, and we update $\mathcal{P}_n^\top\boldsymbol{u}^{k+1}$ as

$$\mathcal{P}_n\boldsymbol{u}^{k+1}=\mathcal{P}_n\boldsymbol{u}^k+\sum_{j=1}^{s}\gamma_jy_j.$$

The sparsity of (22) is shown by Fig. 1c.

2.2 Approach 2

Approach 2 ignores the boundary conditions in the first place and truncates (14) to form a square system:

$$\mathcal{P}_n\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{T}\mathcal{P}_n\boldsymbol{u}=\mathcal{P}_n\mathcal{L}\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u},\tag{23}$$

where the truncations of $\mathcal{P}_n\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top$ and $\mathcal{P}_n\mathcal{L}\mathcal{P}_n^\top$ are, again, carried out exactly. Stepping using multistep or Runge–Kutta methods, we end up with

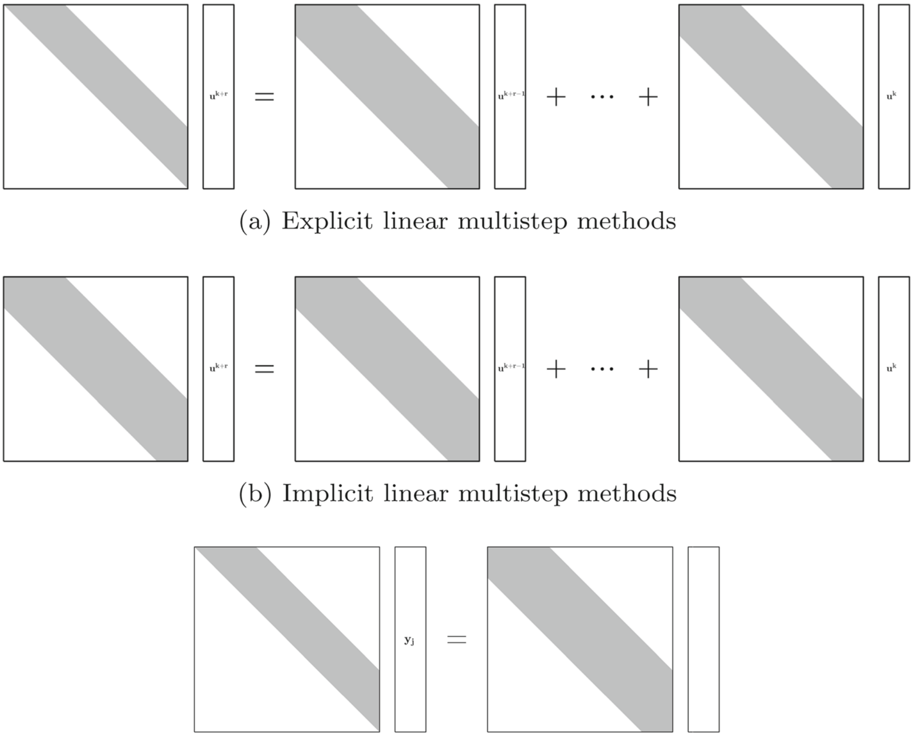
$$\begin{aligned}&(\mathcal{P}_n\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top-h\beta_r\mathcal{P}_n\mathcal{L}\mathcal{P}_n)\mathcal{P}_n\boldsymbol{u}^{k+r}\\&\quad=h\sum_{j=0}^{r-1}\beta_j\mathcal{P}_n\mathcal{L}\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+j}-\sum_{j=0}^{r-1}\alpha_j\mathcal{P}_n\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+j}\end{aligned}\tag{24}$$

and

$$\mathcal{P}_n\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top y_j=h\mathcal{P}_n\mathcal{L}\mathcal{P}_n^\top(\mathcal{P}_n\boldsymbol{u}^k+\mu_jy_{j-1}),\tag{25}$$

respectively. The sparsity patterns are shown in Fig. 2. Note that (24) and (25) differ from (19) and (20) by being square systems instead of rectangular. Since $\mathcal{P}_n\mathcal{S}_{N-1}\ldots\mathcal{S}_0\mathcal{P}_n^\top$ is non-singular, (24) and (25) can be solved for $\mathcal{P}_n\boldsymbol{u}^{k+r}$ and the intermediate result at jth stage, respectively. Obviously, $\mathcal{P}_n\boldsymbol{u}^{k+r}$ obtained this way rarely satisfies the boundary condition. To enforce the boundary condition, we free N components in $\mathcal{P}_n\boldsymbol{u}^{k+r}$ and allow them to be re-determined by

$$\mathcal{B}\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+r}=c.$$



(a) Explicit linear multistep methods

(b) Implicit linear multistep methods

(c) Runge-Kutta methods

Fig. 2 Sparsity patterns of the fully discretized system in Approach 2 for linear multistep methods (24) and Runge-Kutta methods (25)

This is, in fact, an $N \times N$ system. For example, if we choose to re-determine the last N components in $\mathcal{P}_n \boldsymbol{u}^{k+r}$, we end up with the $N \times N$ system in, for example, MATLAB's syntax

$$\mathcal{B}\texttt{(1:N,n-N+1:n)}\boldsymbol{u}^{k+r}\texttt{(n-N+1:n)} = c - \mathcal{B}\texttt{(1:N,1:n-N)}\boldsymbol{u}^{k+r}\texttt{(1:n-N)}.$$

One may wonder the difference between the solutions obtained via these two approaches, which we investigate now.

2.3 Approach 1 versus Approach 2

Assuming the solution of (10) is Lipschitz continuous in both space and time, we now show that the difference between the solutions obtained via the two approaches is bounded and vanishes when the discretization in both time and space becomes increasingly dense. Although what is furnished below is not a rigorous proof, it suffices to give an explanation why these two approaches return converging solutions.

We set out by considering three problems and assuming the solutions u^{k+j} are known for $j = 0, 1, \dots, r-1$.

- Problem 1: use Approach 1 to obtain a solution vector of length $n + N$, which can be computed by solving the following system (see (21), (22), and Fig. 1):

$$\begin{pmatrix} B_{(1)} & B_{(2)} \\ S_{(1)} & S_{(2)} \\ S_{(3)} & S_{(4)} \end{pmatrix} u_{P_1}^{k+r} = \begin{pmatrix} c \\ \sum_{j=0}^{r-1} \begin{pmatrix} L_{(1)}^j & L_{(2)}^j \\ L_{(3)}^j & L_{(4)}^j \end{pmatrix} u^{k+j} \end{pmatrix}. \tag{26}$$

  Note that (26) covers both linear multistep and Runge–Kutta methods. On the left-hand side, the top N rows are partitioned as an $N \times n$ part $B_{(1)}$ and an $N \times N$ part $B_{(2)}$. Along with the N-vector c, the first N equations represent the boundary conditions. The banded part of the coefficient matrix is partitioned into $S_{(1)}$, $S_{(2)}$, $S_{(3)}$, and $S_{(4)}$, whose dimensions are $(n - N) \times n$, $(n - N) \times N$, $N \times n$, and $N \times N$, respectively. The solution, denoted by $u_{P_1}^{k+r}$, is an $(n + N)$-vector. The $(n + N)$-vector u^{k+j} represents either the solution at the previous steps or the initial condition. The banded matrices which u^{k+j} multiplies with are partitioned into $L_{(1)}^j$, $L_{(2)}^j$, $L_{(3)}^j$, and $L_{(4)}^j$, whose dimensions are conformal with $S_{(1)}$, $S_{(2)}$, $S_{(3)}$, and $S_{(4)}$, respectively.
- Problem 2: use Approach 1 to obtain a solution vector of length n, which amounts to solving the $n \times n$ system

$$\begin{pmatrix} B_{(1)} \\ S_{(1)} \end{pmatrix} u_{P_2}^{k+r} = \begin{pmatrix} c \\ \sum_{j=0}^{r-1} L_{(1)}^j u^{k+j}\,\texttt{(1:n)} \end{pmatrix}.$$

- Problem 3: use Approach 2 to obtain a solution vector of length n by first solving the $n \times n$ system

$$\begin{pmatrix} S_{(1)} \\ S_{(3)} \end{pmatrix} \widehat{u}_{P_3}^{k+r} = \sum_{j=0}^{r-1} \begin{pmatrix} L_{(1)}^j \\ L_{(3)}^j \end{pmatrix} u^{k+j}\,\texttt{(1:n)}, \tag{27a}$$

  which produces the intermediate solution $\widehat{u}_{P_3}^{k+r}$. This is then followed by the correction step which re-determines the last N components of $\widehat{u}_{P_3}^{k+r}$. If the corrected solution is denoted by $u_{P_3}^{k+r}$, the boundary conditions are satisfied

$$B_{(1)} u_{P_3}^{k+r} = c. \tag{27b}$$

Now we assume that n is large enough so that the solution is fully resolved and spectral accuracy is achieved in space. Thus, there exists a small number $\epsilon > 0$ so that $\left\| u^{k+j}\,\texttt{(n-N+1:n+N)} \right\| < \epsilon$ for $j = 0, 1, \ldots, r - 1$ and $\left\| u_{P_1}^{k+r}\,\texttt{(n-N+1:n+N)} \right\| < \epsilon$. Also, we assume h is small enough to stabilize whatever time stepper we are using.

In the following argument, Problem 1 serves as a bridge connecting Problems 2 and 3 whose solutions are what we try to show to be close. Thus, we bound the difference between $u_{P_1}^{k+r}$ and $u_{P_3}^{k+r}$ (Step 1) and that between $u_{P_1}^{k+r}$ and $u_{P_2}^{k+r}$ (Step 2) first, and these results, when combined, give the difference between $u_{P_2}^{k+r}$ and $u_{P_3}^{k+r}$.

Step 1 We first look at the difference between the first n components of $u_{P_1}^{k+r}$ and the uncorrected solution $\widehat{u}_{P_3}^{k+r}$, i.e., $e_1 = ||u_{P_1}^{k+r}\,\texttt{(1:n)} - \widehat{u}_{P_3}^{k+r}||$. From (26), we have

$$\begin{pmatrix} S_{(1)} & S_{(2)} \\ S_{(3)} & S_{(4)} \end{pmatrix} u_{P_1}^{k+r} = \sum_{j=0}^{r-1} \begin{pmatrix} L_{(1)}^j & L_{(2)}^j \\ L_{(3)}^j & L_{(4)}^j \end{pmatrix} u^{k+j},$$

that is,

$$\begin{pmatrix} S_{(1)} \\ S_{(3)} \end{pmatrix} u_{P_1}^{k+r}\,\texttt{(1:n)} + \begin{pmatrix} S_{(2)} \\ S_{(4)} \end{pmatrix} u_{P_1}^{k+r}\,\texttt{(n+1:n+N)}$$
$$= \sum_{j=0}^{r-1} \begin{pmatrix} L_{(1)}^{j} \\ L_{(3)}^{j} \end{pmatrix} u^{k+j}\,\texttt{(1:n)} + \sum_{j=0}^{r-1} \begin{pmatrix} L_{(2)}^{j} \\ L_{(4)}^{j} \end{pmatrix} u^{k+j}\,\texttt{(n+1:n+N)}.$$

Combining the last equation with (27a), we have

$$\begin{aligned} e_1 &= \left\| \begin{pmatrix} S_{(1)} \\ S_{(3)} \end{pmatrix}^{-1} \begin{pmatrix} S_{(2)} u_{P_1}^{k+r}\,\texttt{(n+1:n+N)} - \sum_{j=0}^{r-1} L_{(2)}^{j} u^{k+j}\,\texttt{(n+1:n+N)} \\ S_{(4)} u_{P_1}^{k+r}\,\texttt{(n+1:n+N)} - \sum_{j=0}^{r-1} L_{(4)}^{j} u^{k+j}\,\texttt{(n+1:n+N)} \end{pmatrix} \right\| \\ &\le \left\| \begin{pmatrix} S_{(1)} \\ S_{(3)} \end{pmatrix}^{-1} \right\| \max_j \{\|S_{(2)}\|, \|L_{(2)}^{j}\|, \|S_{(4)}\|, \|L_{(4)}^{j}\|\} 2\epsilon = C_1 \epsilon. \end{aligned} \tag{28}$$

Now we bound the correction due to the enforcement of the boundary conditions, i.e., $e_2 = \left\| \widehat{u}_{P_3}^{k+r} - u_{P_3}^{k+r} \right\|$.

For a multistep method (15), we have

$$B_{(1)} \widehat{u}_{P_3}^{k+r} - B_{(1)} u_{P_3}^{k+r} = B_{(1)} \widehat{u}_{P_3}^{k+r} - c = B_{(1)} \widehat{u}_{P_3}^{k+r} + B_{(1)} \sum_{j=0}^{r-1} \frac{\alpha_j}{\alpha_r} u^{k+j}, \tag{29}$$

where we have used the fact that all $u_{P_3}^{k+j}$ satisfy the boundary conditions, i.e., $B^{(1)} u_{P_3}^{k+j} = c$ for $j = 0, \ldots, r-1$, and the consistency condition

$$\sum_{j=0}^{r} \alpha_j = 0.$$

Substituting (15) into (29) gives

$$\begin{aligned} &B_{(1)} \widehat{u}_{P_3}^{k+r} - B_{(1)} u_{P_3}^{k+r} \\ &\quad = \frac{B_{(1)}}{\alpha_r} h \left(\sum_{j=0}^{r-1} \beta_j \begin{pmatrix} S_{(1)}^{j} \\ S_{(3)}^{j} \end{pmatrix}^{-1} \begin{pmatrix} L_{(1)}^{j} \\ L_{(3)}^{j} \end{pmatrix} u^{k+j} + \beta_r \begin{pmatrix} S_{(1)}^{r} \\ S_{(3)}^{r} \end{pmatrix}^{-1} \begin{pmatrix} L_{(1)}^{r} \\ L_{(3)}^{r} \end{pmatrix} \widehat{u}_{P_3}^{k+r} \right), \end{aligned}$$

which further leads to

$$e_2 = \left\| \frac{h}{\alpha_r} \left(\sum_{j=0}^{r-1} \beta_j \begin{pmatrix} L_{(1)}^{j} \\ L_{(3)}^{j} \end{pmatrix} u^{k+j} + \beta_r \begin{pmatrix} L_{(1)}^{r} \\ L_{(3)}^{r} \end{pmatrix} \widehat{u}_{P_3}^{k+r} \right) \right\| \le C_2 h, \tag{30}$$

Analogously, for Runge–Kutta methods (16)

$$B_{(1)} \widehat{u}_{P_3}^{k+r} - B_{(1)} u_{P_3}^{k+r} = B_{(1)} \widehat{u}_{P_3}^{k+r} - c = B_{(1)} \widehat{u}_{P_3}^{k+r} - B_{(1)} u^{k+r-1} = B_{(1)} \sum_{j=1}^{r} \gamma_j y_j,$$

implying

$$e_2 = \left\| \sum_{j=1}^{r} \gamma_j y_j \right\| \leq C_3 h. \tag{31}$$

Finally, (28) and any one of (30) and (31) give

$$\begin{aligned} e_3 &= \left\| \begin{pmatrix} u_{P_3}^{k+r} \\ 0 \end{pmatrix} - u_{P_1}^{k+r} \right\| \\ &\leq \left\| \begin{pmatrix} \widehat{u}_{P_3}^{k+r} \\ 0 \end{pmatrix} - \begin{pmatrix} u_{P_1}^{k+r}\,\texttt{(1:n)} \\ 0 \end{pmatrix} \right\| + \left\| \begin{pmatrix} u_{P_1}^{k+r}\,\texttt{(1:n)} \\ 0 \end{pmatrix} - u_{P_1}^{k+r} \right\| \qquad (32) \\ &\quad + \left\| \begin{pmatrix} u_{P_3}^{k+r} \\ 0 \end{pmatrix} - \begin{pmatrix} \widehat{u}_{P_3}^{k+r} \\ 0 \end{pmatrix} \right\| \\ &\leq e_1 + \epsilon + e_2 = C_4 h + C_5 \epsilon. \qquad (33) \end{aligned}$$

Step 2 Since n is large enough to resolve the solution, the difference between $u_{P_2}^{k+r}$ (prolonged by padding with zeros) and $u_{P_1}^{k+r}$ should be small:

$$e_4 = \left\| \begin{pmatrix} u_{P_2}^{k+r} \\ 0 \end{pmatrix} - u_{P_1}^{k+r} \right\| \leq \epsilon. \tag{34}$$

Step 3 Inequalities (33) and (34) give

$$\begin{aligned} ||u_{P_2}^{k+r} - u_{P_3}^{k+r}|| &\leq \left\| \begin{pmatrix} u_{P_2}^{k+r} \\ 0 \end{pmatrix} - u_{P_1}^{k+r} \right\| + \left\| u_{P_1}^{k+r} - \begin{pmatrix} u_{P_3}^{k+r} \\ 0 \end{pmatrix} \right\| \\ &= e_4 + e_3 = C_4 h + C_6 \epsilon. \qquad (35) \end{aligned}$$

The message conveyed by (35) is that the solutions obtained using Approaches 1 and 2 differ only by a quantity of $\mathcal{O}(h)$ plus a multiple of ϵ. In fact, the actual differences observed in all of our experiments are rather minuscule. In Fig. 3, the differences between the computed solutions via the two approaches are displayed versus the number of time steps for the one-dimensional transport equation (11) and the heat equation (12). To have the initial conditions and the solutions at the subsequent time steps fully resolved, we let $n = 300$ for both problems. For the one-dimensional transport equation, 4th-order Adam-Bashforth method is used with $h = 0.1/n^2$, while for the heat equation, 3rd-order Runge–Kutta method is used with $h = 1/n^4$. These h's are chosen to stabilize the time steppers and the derivation of these restrictions is discussed in the next section. For both problems, the computed solution via the two approaches differ only by an amount of virtually machine epsilon after the first 50, 000 steps.

3 Stability

The primary task now is to understand when the approaches proposed in the last section lead to stable calculation if a standard time stepping scheme is employed. In a general application of the method of lines, we consider the semi-discretized problem $\mathcal{T}u = Au$, where A is a matrix that approximates the spatial operator. The rule of thumb for stability is that the MOL is stable if the eigenvalues of A, scaled by the step size h, lie in the stability region of the time stepper [28]. The same conclusion is drawn from our extensive experiments with the

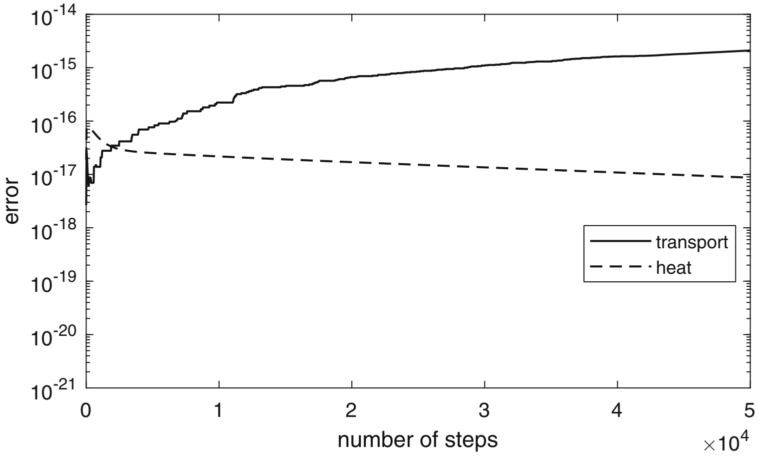


Fig. 3 Difference between solutions obtained via the two approaches

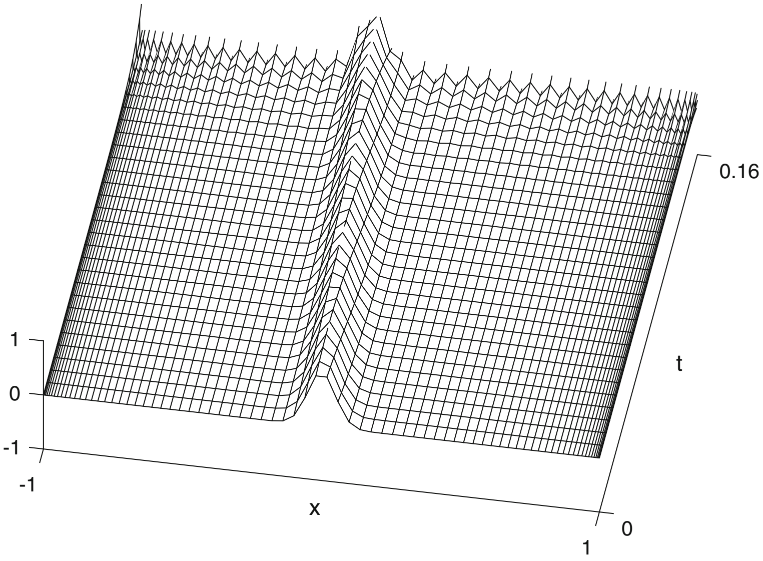


Fig. 4 Modal instability incited when solving (11) with $n = 80$ and $h = 3.45/n^2$

ultraspherical spectral method — for many problems in the form of (10), the two proposed approaches lead to discretization whose stability is mainly determined by the spectra of A. For example, if we apply the forward Euler method to the one-dimensional transport equation (11), both of the approaches begin to yield unstable results when $h > 3.41/n^2$, as shown in Fig. 4. This instability is *modal* [30, section 31], as it sets in globally and never ceases to grow — if we carry on to $t = 0.3$, the unstable solution would be $\mathcal{O}(10^5)$.

Modal instability also occurs with $h > 7.2/n^4$ when the proposed approaches and the forward Euler method are used to solve the heat equation (12).

We now show that these restrictions are indeed attributable to eigenvalues. The discussion below will mainly be based on Approach 2 as it offers an easier form for analysis. To facilitate our discussion, we adopt the following notations in this section for the truncated version of the solution vector and the relevant operators:

$$\begin{aligned} &u = \mathcal{P}_n\boldsymbol{u},\ S_\lambda = \mathcal{P}_n\mathcal{S}_\lambda\mathcal{P}_n^\top,\ D_\lambda = \mathcal{P}_n\mathcal{D}_\lambda\mathcal{P}_n^\top,\ L = \mathcal{P}_n\mathcal{L}\mathcal{P}_n^\top, \\ &\Theta_\lambda = \left(\mathcal{P}_n\mathcal{S}_{N-1}\mathcal{P}_{n+2}^\top\right)\left(\prod_{i=2}^{N-\lambda}\mathcal{P}_{n+2(i-1)}\mathcal{S}_{N-i}\mathcal{P}_{n+2i}^\top\right), \\ &M_\lambda = \mathcal{P}_{n+2(N-\lambda)}\mathcal{M}_\lambda[a^\lambda]\mathcal{P}_n^\top \text{ for } \lambda = 0, 1, \ldots, N. \end{aligned} \tag{36}$$

Let us first look at the one-dimensional transport equation (11). When Approach 2 is applied to (11), the system we end up solving can be written as

$$S_0\mathcal{T}Wu = D_1u, \tag{37}$$

where, other than the temporal differential operator $\mathcal{T}$, all the operators and the solution vector are replaced by their discretized and truncated counterparts. Here, $W = \left(\begin{array}{c|c} I_{n-1} & 0 \\ \hline \multicolumn{2}{c}{\mathcal{B}\mathcal{P}_n^\top} \end{array}\right)$ is an $n \times n$ matrix. The Dirichlet boundary condition is represented by the following $1 \times \infty$ functional $\mathcal{B} = \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots \end{pmatrix}$.

Note that the last row of (37) simplifies to (see (7a))

$$\frac{1}{2}\mathcal{T}\mathcal{B}\mathcal{P}_n^\top u = 0. \tag{38}$$

Since any of the multistep and Runge–Kutta methods represents u^{k+r} as a linear combination of u^{k+j} for $j = 0, 1, \ldots, r-1$ and the boundary condition is satisfied by u^{k+j} for $j = 0, 1, \ldots, r-1$, (38) amounts to the statement that the boundary condition is also satisfied, that is,

$$\mathcal{B}\mathcal{P}_n^\top u = 0. \tag{39}$$

On the other hand, the top $n-1$ rows of (37) are

$$S_0\texttt{(1:n-1,1:n-1)}\mathcal{T}u\texttt{(1:n-1)} + S_0\texttt{(1:n-1,n)}\mathcal{T}\mathcal{B}\mathcal{P}_n^\top u = D_1\texttt{(1:n-1,:)}u. \tag{40}$$

Because of (39), the second term on the left-hand side of (40) can be dropped and (40) coincides with the first $n-1$ rows of (23) for $N = 1$ and $\mathcal{L} = \mathcal{D}$. Hence, (37) represents the semi-discretized system for which the largest eigenvalue(s) of $W^{-1}S_0{}^{-1}D_1$ may determine the step size of a time-stepping method for stability. The following theorem gives an upper bound for the spectral radius of $W^{-1}S_0{}^{-1}D_1$.

Theorem 3.1 *The spectral radius of $Q = W^{-1}S_0{}^{-1}D_1$ satisfies*

$$\rho(Q) \le (n-1)^2\sqrt{\frac{1}{3} + \frac{2}{3(n-1)^2}}.$$

Proof We assume that n is an odd number; the proof for the even case follows analogously. Note that

$$S_0 = (I - B)A,$$

where $A = \operatorname{diag}\left(1, \frac{1}{2}, \frac{1}{2}, \cdots, \frac{1}{2}\right)$ and $B = \begin{pmatrix} 0_{(n-2)\times 2} & I_{n-2} \\ 0_{2\times 2} & 0_{2\times(n-2)} \end{pmatrix}$. Since B is a double-shift matrix, the inverse of S_0 can be represented as a finite series

$$S_0^{-1} = A^{-1}(I-B)^{-1} = A^{-1}\sum_{j=0}^{n/2} B^j,$$

which, when spelled out, reads

$$S_0^{-1} = \begin{pmatrix} 1 & & 1 & & 1 & \cdots & 1 \\ & 2 & & 2 & & \ddots & \\ & & 2 & & 2 & & \vdots \\ & & & \ddots & & \ddots & \\ & & & & \ddots & & 2 \\ & & & & & 2 & \\ & & & & & & 2 \end{pmatrix}. \tag{41}$$

and it is easy to show

$$W^{-1} = \left(\frac{I_{n-1}\big|0}{-\mathcal{B}\mathcal{P}_n^{\top}}\right).$$

A simple calculation gives

$$Q = \begin{pmatrix} & 1 & & 3 & & 5 & & & n-2 & \\ & & 4 & & 8 & & \cdots & 2(n-3) & & 2(n-1) \\ & & & 6 & & 10 & & & 2(n-2) & \\ & & & & \ddots & & \ddots & & & \vdots \\ & & & & & \ddots & & \ddots & & \vdots \\ & & & & & & \ddots & & \ddots & \vdots \\ & & & & & & & \ddots & & \vdots \\ & & & & & & & & \ddots & \vdots \\ & & & & & & & & & 2(n-1) \\ 0 & -1^2 & -2^2 & -3^2 & -4^2 & -5^2 & \cdots & -(n-3)^2 & -(n-2)^2 & -(n-1)^2 \end{pmatrix}.$$

The characteristic polynomial $\det(\lambda I - Q) = \lambda^n + a_{n-1}\lambda^{n-1} + \ldots + a_1\lambda + a_0$ has concise expressions for the coefficients of the leading terms,[2]

$$\begin{aligned} a_{n-1} &= -tr(Q) = (n-1)^2, \\ a_{n-2} &= E_2(Q) = \sum_{1\le i\ne j\le n} \det(Q[\{i,j\}]) = \frac{(n-1)^2}{3}[(n-1)^2 - 1], \end{aligned}$$

[2] For an $n \times n$ matrix A, $E_k(A)$ denotes the sum of A's principal minor of size k [15, section 1.2] and we use the notation $A[\alpha] = A[\alpha, \alpha]$, where α is a set of indices, to denote a principal submatrix of A [15, section 0.7.1].

which, by Vièta's theorem [32, section 5.7], imply

$$\sum_{k=1}^{n} \lambda_k = -(n-1)^2,$$

$$\sum_{i<j} \lambda_i \lambda_j = \frac{(n-1)^2}{3}[(n-1)^2 - 1],$$

where $\{\lambda_k\}_{k=1}^n$ are the n roots of $\det(\lambda I - Q)$, i.e., the eigenvalues of Q. We then have

$$\sum_{k=1}^{n} \lambda_k^2 = \left(\sum_{k=1}^{n} \lambda_k\right)^2 - 2\sum_{i \neq j, i<j} \lambda_i \lambda_j = \frac{(n-1)^4}{3} + \frac{2(n-1)^2}{3},$$

which gives

$$|\lambda_{max}| \leq \sqrt{\frac{(n-1)^4}{3} + \frac{2(n-1)^2}{3}} = (n-1)^2 \sqrt{\frac{1}{3} + \frac{2}{3(n-1)^2}}.$$

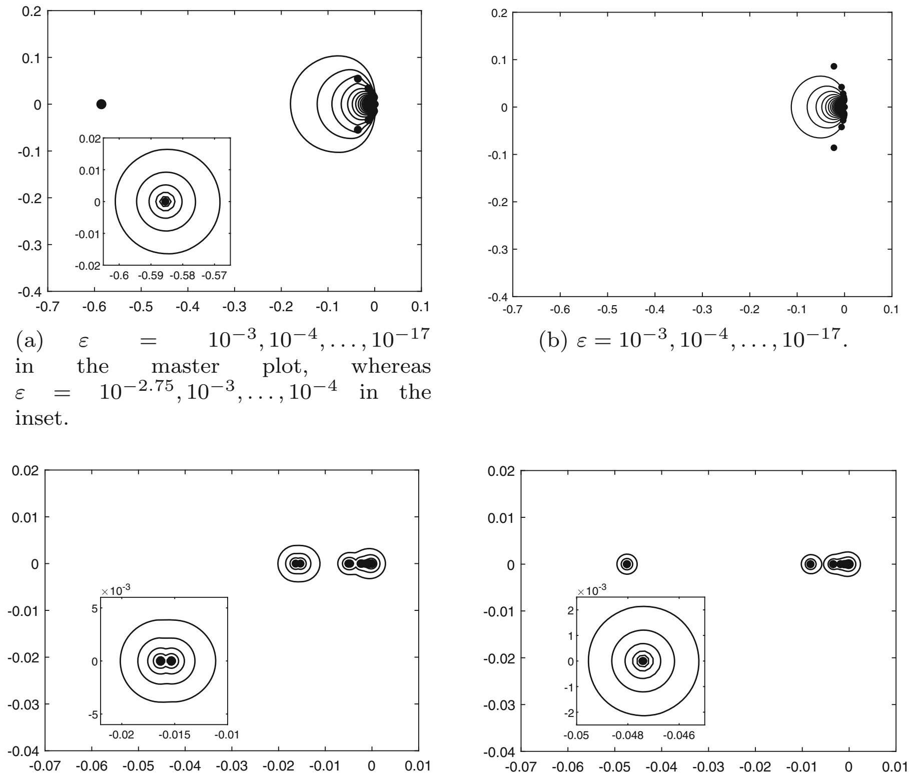
□

The necessary condition of the step size can be readily derived from Theorem 3.1. For example, for the forward Euler method to be stable, it is required that $|h\lambda_{max} + 1| \leq 1$, that is,

$$h \leq \frac{3.41}{(n-1)^2}. \tag{44}$$

This is exactly the threshold beyond which we see the modal instability as in Fig. 4. In Fig. 5a, the eigenvalues of $n^{-2}Q$ for $n = 64$ are plotted using dots, where the rescaling factor of n^{-2} helps remove the dependence of the entries of Q on the dimension n. The largest eigenvalue is an outlier located on the real axis that breaks away from the main cohort formed by the rest of the spectra and juts out into the left half plane. Along with the spectra, we also display the ε-pseudospectra in Fig. 5a. The pseudospectra clearly show the importance of the outlier. Although the spatial discretization matrix of the one-dimensional transport equation is nonnormal, as indicated by the pseudospectra around the main cohort, it is not far from normal in that its behavior is largely determined by the outlier. More precisely, it is only the magnitude of this outlier, not the pseudospectra around it, that matters. This can be seen from the facts that (1) the outlier is much larger in modulus than the ε-pseudospectra around the main cohort, (2) even in a plot for ε as large as 10^{-3} we do not see the pseudospectra contours around the outlier, and (3) the pseudospectra around the outlier (see the close-up) consist of a few concentric circles whose radii shrink proportionally with the order of ε. In fact, this outlier is almost a *normal eigenvalue* [30, §52] — its condition number $\kappa(\lambda_{outlier}) \approx 9.2$ (calculated in ∞-norm). In contrast, the two most outlying eigenvalues in the main cohort (symmetric about the real axis) have a condition number approximately 1378.8, which is the smallest among all the eigenvalues in the main cohort.[3] That is, the eigenvalues in the main cohort are nonnormal or significantly so. In a word, the outlier is of physical significance, and it is this outlier that restricts the step size of a time stepper with a bounded stability region.

[3] The condition number of other eigenvalues in the main cohort could be much larger. The closer they are to the origin, the greater the condition numbers become. The eigenvalues near the origin can hardly be numerically calculated to any satisfactory accuracy due to the extremely poor conditioning.



(a) $\varepsilon = 10^{-3}, 10^{-4}, \ldots, 10^{-17}$ in the master plot, whereas $\varepsilon = 10^{-2.75}, 10^{-3}, \ldots, 10^{-4}$ in the inset.

(b) $\varepsilon = 10^{-3}, 10^{-4}, \ldots, 10^{-17}$.

(c) $\varepsilon = 10^{-2.75}, 10^{-3.00}, \ldots, 10^{-4.25}$ in both plots.

(d) $\varepsilon = 10^{-2.75}, 10^{-3.00}, \ldots, 10^{-4.25}$ in both plots.

Fig. 5 The spectra and the ε-pseudospectra of the spatial discretization matrices due to the ultraspherical spectral method (left panes) and the Chebyshev pseudospectral method (right panes), rescaled by n^{-2} and n^{-4} respectively for the one-dimensional transport equation (top panes) and the heat equation (bottom panes). The insets show close-ups in the neighborhood of the outlier(s)

Hence, we can say that for the proposed approaches the stability of a time marching scheme, when applied to (11), is mainly determined by the spectra.

For comparison purposes, we show in Fig. 5b the spectra and pseudospectra of the rescaled first-order differentiation matrix from the Chebyshev pseudospectral method. As the largest eigenvalues in Fig. 5b are smaller than the outlier in Fig. 5a, one might think that the Chebyshev pseudospectral method is superior to the ultraspherical spectral method. This is, in fact, not the case. First, it is not true that ultraspherical spectral method always results in greater spectral radius than the collocation pseudospectral method, e.g., the second-order differentiation operator (see below), or if boundary conditions of other types are enforced (see Remark 3.1). Second, the Chebyshev pseudospectral method, in this particular case, allows a step size only of a constant times larger than the ultraspherical spectral method, since for both methods the spectral radius is $\mathcal{O}(n^2)$. Third, the ultraspherical spectral method is cheaper stepwise than the Chebyshev pseudospectral method (see Sect. 5), thanks to its sparsity structures.

Now, we turn to the heat equation (12) which features the spatial differentiation of a second order. Applying Approach 2 to the heat equation (12), but leaving the temporal operator nondiscretized, gives

$$S_1 S_0 \mathcal{T} H u = D_2 u, \tag{45}$$

where $H = \left(\frac{I_{n-2} | 0}{\mathcal{B}\mathcal{P}_n^\top}\right)$, and the functional $\mathcal{B} = \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots \\ 1 & -1 & 1 & -1 & \cdots \end{pmatrix}$ represents the Dirichlet boundary conditions in Chebyshev space. The equivalence of (45) and the Approach 2 discretization follows exactly the same reasoning for that of (37) where W is involved instead.

The following theorem gives a bound on the spectral radius of $H^{-1} S_0^{-1} S_1^{-1} D_2$.

Theorem 3.2 *The spectral radius of* $G = H^{-1} {S_0}^{-1} {S_1}^{-1} D_2$ *is bounded by*

$$\rho(G) \le \frac{2}{3} n(n-2)(n-1)^2.$$

Proof S_0^{-1} and D_2 are given by (41) and (5), respectively, and S_1^{-1} can be derived analogously to S_0^{-1}. Also, we have $H^{-1} = \left(\frac{I_{n-2} | 0}{H'}\right)$, where $H' = \begin{pmatrix} & -1 & & -1 & \ldots & 0 & \frac{1}{2} & -\frac{1}{2} \\ -1 & & -1 & & \ldots & -1 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$. It can be shown that $G_{nn} = -\frac{2}{3} n(n-2)(n-1)^2$ is the entry with the largest magnitude. Noting this, we can further show that for any $\lambda < -\frac{2}{3} n(n-2)(n-1)^2$ the determinant $\det(\lambda I - G) \neq 0$. Hence, all eigenvalues of G are smaller than $\frac{2}{3} n(n-2)(n-1)^2$ in modulus. □

The spectra and the pseudospectra of $n^{-4} G$ are shown in Fig. 5c, where the eigenvalues are lined up on the real axis, due to the parity of the order of the spatial differentiation. Once again, there are (two) outliers which detach themselves from the rest of the spectra and reside far in the left half plane. Though we can see the pseudospectra for both the outliers and the rest of the spectra, the relatively large ε, the relatively small scale of the axes, the shape of the pseudospectra contours around the outliers, and the fact that the condition numbers of these two outliers are small (both approximately 2.3) suggest that the outliers are physically significant, governing the behavior of the matrix.[4] Therefore, the outlier of largest modulus determines the maximum step size if a time stepper with bounded stability region is used. The spectra and the pseudospectra of the rescaled Chebyshev second-order differentiation matrix are shown in Fig. 5d for comparison.

Again, we can derive from Theorem 3.2 a threshold value below which the step size of the time marching scheme leads to a stable solution to (12). Although this bound is not sharp, i.e., a step size bigger than this value may well stabilize the computation, the key point is not missed — the largest eigenvalue of the ultraspherical discretization matrix behaves like $\mathcal{O}(n^4)$. This echoes [34], which gives a same result for the second-order pseudospectral differentiation matrix. Such an agreement is not a coincidence. Furthermore, the last two theorems suggest that the largest eigenvalues of the Nth order spatial differentiation operator, when truncated to $n \times n$ and converted back to Chebyshev space, scale like $\mathcal{O}(n^{2N})$, the same as in the Chebyshev pseudospectral methods.[5] Indeed, this is exactly what we show in Theorem 3.3

[4] In fact, the rest of the eigenvalues are all normal with $\mathcal{O}(1)$ condition numbers.

[5] It is well known that the largest eigenvalues of the Nth order Chebyshev pseudospectral differentiation matrix scale like $\mathcal{O}(n^{2N})$. Surprisingly, however, this assertion is not found in the literature and it seems that no one has ever given a proof of it.

below. To do so, we look at (23). Inverting the product of the conversion matrices on the left-hand side of (23) gives

$$\mathcal{T}u = (S_{N-1}S_{N-2}\dots S_0)^{-1}\, Lu, \tag{46}$$

where each conversion matrix is truncated exactly before the inversion. The following lemma gives an upper bound for the norm of S_λ^{-1}.

Lemma 3.1 *For $\lambda = 1, 2, \dots$, $\left\| S_\lambda^{-1} \right\| \le C_\lambda n^2$ for some constant C_λ and $\left\| S_0^{-1} \right\| \le n$.*

Proof Following a derivation analogous to the one for S_0^{-1}, we find

$$S_\lambda^{-1} = \begin{pmatrix} 1 & & 1 & & 1 & & \cdots & 1 \\ & \frac{\lambda+1}{\lambda} & & \frac{\lambda+1}{\lambda} & & \frac{\lambda+1}{\lambda} & & \\ & & \frac{\lambda+2}{\lambda} & & \frac{\lambda+2}{\lambda} & & \ddots & \vdots \\ & & & \frac{\lambda+3}{\lambda} & & \frac{\lambda+3}{\lambda} & & \vdots \\ & & & & \ddots & & \ddots & \\ & & & & & \ddots & & \frac{\lambda+n-3}{\lambda} \\ & & & & & & \frac{\lambda+n-2}{\lambda} & \\ & & & & & & & \frac{\lambda+n-1}{\lambda} \end{pmatrix}.$$

Hence, we have

$$\begin{aligned} \left\| S_\lambda^{-1} \right\| &= \max_i \left(\frac{n+1}{2}, \frac{n-1}{2}\frac{\lambda+1}{\lambda}, \dots, \left(\frac{n+1}{2} - \left\lceil \frac{i}{2} \right\rceil \right) \frac{\lambda+i}{\lambda}, \dots \frac{\lambda+n-1}{\lambda} \right) \\ &\le C_\lambda n^2, \end{aligned}$$

and $\left\| S_0^{-1} \right\| \le n$ follows from (41). □

Now we are in a position to bound the norm of the matrix on the right-hand side of (46).

Lemma 3.2 *Suppose that each of M_λ is of a finite bandwidth independent of the degrees of freedom n for $\lambda = 0, 1, \dots, N$, then*

$$\left\| \left(S_{N-1} \dots S_0 \right)^{-1} L \right\| \le C n^{2N} \tag{47}$$

for some constant C.

Proof From (5), it is easy to see that

$$\| D_\lambda \| \le C_N n$$

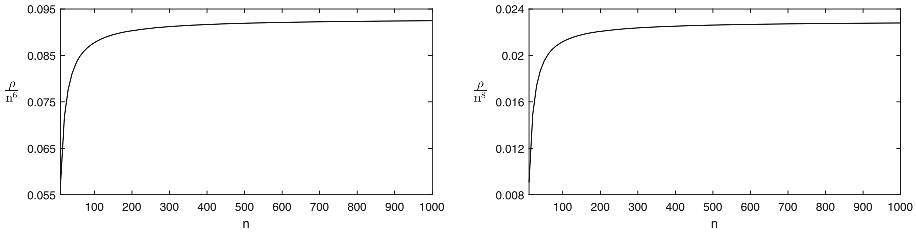
for all λ.

Since $M_\lambda[a^\lambda]$ has a finite bandwidth, $\left\| M_\lambda[a^\lambda] \right\|$ is bounded by a constant regardless the dimension n. Similarly, this is the case for $\| S_\lambda \|$ for all λ.

By the triangle inequality and the submultiplicativity of matrix norms, it follows from (36) that

$$\| L \| \le n \left(\| M_N \| + \sum_{\lambda=1}^{N-1} \| \Theta_\lambda M_\lambda \| \right) \le C_L n,$$

for some C_L and this, along with Lemma 3.1, gives (47). □



(a) $N = 3$ (b) $N = 4$

Fig. 6 The spectral radius, normalized by n^{-2N}, of the $n \times n$ spatial discretization matrices for Nth-order differentiation operators versus n

A direct consequence of Lemma 3.2 is an upper bound for the spectral radius of the matrix on the right-hand side of (46).

Theorem 3.3 *When Approach 2 is used for solving* (10) *where $\mathcal{L}$ is an Nth order differential operator with smooth variable coefficients given by* (3)*, there exists a constant C independent of the degrees of freedom n for the spatial discretization such that*

$$\rho(S_0^{-1}S_1^{-1}\dots S_{N-1}^{-1}L) \le Cn^{2N}. \tag{48}$$

Proof The smoothness of the variable coefficients implies finite bandwidth for each M_λ. Hence, this is a standard result led to by (47) which can be found, for example, in [15, Theorem 5.6.9]. □

Theorem 3.3 is numerically verified for the cases of $N = 3, 4$ in Fig. 6, where the spectral radius of Nth-order differentiation matrices is normalized by n^{-2N} and plotted versus different n. It can be seen that the normalized spectral radii indeed tend to be a constant.

In fact, the ε-pseudospectra radius of the matrix on the right-hand side of (46) is bounded by the same quantity plus ε.

Theorem 3.4 *If the assumption holds as in* Theorem 3.3*,*

$$\rho_\epsilon(S_0^{-1}S_1^{-1}\dots S_{N-1}^{-1}L) \le Cn^{2N} + \epsilon, \tag{49}$$

where the constant C is the one given in (48)*.*

Proof For any $\|E\| \le \epsilon$,

$$\rho(S_0^{-1}S_1^{-1}\dots S_{N-1}^{-1}L + E) \le \left\|S_0^{-1}S_1^{-1}\dots S_{N-1}^{-1}L\right\| + \|E\| \le Cn^{2N} + \epsilon,$$

which, by the second definition of pseudospectra [28, §2], gives (49). □

The bounds in the last two theorems give the worst case scenario of how the spectra and the pseudospectra scale with n for N. If we use the quantity n^{2N} as a guidance for choosing the step size, the stability is guaranteed.

Since the largest eigenvalue(s) of a spatial discretization matrix also grows like $\mathcal{O}(n^{2N})$ for the Chebyshev pseudospectral method, the ultraspherical spectral method and the Chebyshev pseudospectral method roughly tie in terms of the largest step that can be taken for a time marching scheme with a bounded stability region. The fact that the largest eigenvalues match for these two methods can also be seen by premultiplying both sides of (46) by an inverse discrete cosine transform (iDCT) matrix and ignoring the first and last rows, as this reproduces the discretization led to by the Chebyshev pseudospectral method [28, chapter 10]. Because the iDCT matrix is unitary, the norms of the spatial discretization matrices due to these two methods should be roughly same.

Remark 3.1 Our discussion in this section is based on the one-dimensional transport equation and the heat equation subject to homogeneous Dirichlet boundary conditions. However, the use of homogeneous Dirichlet boundary conditions is unimportant. Although other boundary conditions may lead to different constants in bounds such as (44), it would not change the main result given by Theorem 3.3. In addition, homogeneous Dirichlet boundary conditions were adopted in the study of the collocation-based pseudospectral methods [11, 28, 30, 34]. It is for comparative purposes that the use of the same boundary conditions seems natural.

4 Error

The error in the computed solution of PDEs comes mainly from two sources: discretization and rounding, where the former, in the present context, consists of those in space and time. That is,

$$\text{Total error} = \begin{array}{c}\text{spatial}\\ \text{discretization}\\ \text{error}\end{array} + \begin{array}{c}\text{temporal}\\ \text{discretization}\\ \text{error}\end{array} + \begin{array}{c}\text{rounding}\\ \text{error}\end{array}.$$

Like any other spectral method, the ultraspherical spectral method offers spectral accuracy, that is, the accuracy is limited not by the order of the discretization, but by the smoothness of the solution being approximated. When the degrees of freedom are sufficiently large, the solution can be adequately resolved in space, thereby bringing no spatial discretization error. The temporal discretization error introduced by the standard time marching schemes is usually of an algebraic order and its quantification and analysis can be found in standard texts like [1, 4, 12]. When the time step is small enough, the temporal discretization error can essentially be restricted to or below the level of machine epsilon. Hence, it is possible to completely annihilate the discretization error and this is a common working paradigm adopted by spectral methods for PDEs. This way, one is only left with the errors introduced by rounding. We now give an analysis of the rounding error for the proposed method, assuming the discretization error is absent.

We consider the iterative model

$$AU^{k+1} = BU^k, \tag{50}$$

which can be deemed as the prototype of the discretized systems obtained by the proposed method. Here, U^k and U^{k+1} are the computed solutions at two successive steps[6].

The key to our analysis is the quantity

$$\Delta^{k+1} = \underline{U}^{k+1} - A^{-1}B\underline{U}^k,$$

where $\underline{U}^k$ and $\underline{U}^{k+1}$, stored as floating point numbers, are the computed solutions at kth and $(k+1)$th step, respectively. Here, the matrix $A^{-1}B$ is assumed to be exact, not in its floating

[6] For a r-step linear multistep method, such a relation can be derived by forming A and B as $rn \times rn$ block matrices and U^k and U^{k+1} as vectors that incorporate the computed solution at r successive time steps.

point representation, so that $\varDelta^{k+1}$ quantifies the amount of error introduced by rounding at a single step. We shall find an upper bound for its magnitude as follows.

$$\left\|\varDelta^{k+1}\right\| = \left\|\underline{U}^{k+1} - A^{-1}B\underline{U}^{k}\right\| = \left\|fl(\underline{A}^{-1}\underline{B}\,\underline{U}^{k}) - A^{-1}B\underline{U}^{k}\right\|,$$

where $fl(x)$ denotes the function producing the closest floating point approximation to a given number x. There exists ϵ with $|\epsilon| \leq \epsilon_{mach}$ such that $fl(x) = x(1+\epsilon)$ [20]. Here, ϵ_{mach} is the machine epsilon and in IEEE double precision arithmetic ϵ_{mach} is $2^{-53} \approx 1.11 \times 10^{-16}$. Hence,

$$\begin{aligned}
\left\|\varDelta^{k+1}\right\| &= \left\|\underline{A}^{-1}\underline{B}\,\underline{U}^{k}(1+\epsilon) - A^{-1}B\underline{U}^{k}\right\| \\
&\leq \left\|\underline{A}^{-1}\underline{B} - A^{-1}B\right\| \left\|\underline{U}^{k}\right\| + \epsilon \left\|\underline{A}^{-1}\underline{B}\right\| \left\|\underline{U}^{k}\right\| \\
&\leq \left\|\underline{A}^{-1}\right\| \left\|\underline{B} - B\right\| \left\|\underline{U}^{k}\right\| + \|B\| \left\|\underline{A}^{-1} - A^{-1}\right\| \left\|\underline{U}^{k}\right\| + \epsilon \left\|\underline{A}^{-1}\underline{B}\right\| \left\|\underline{U}^{k}\right\|,
\end{aligned}$$

By Theorem 2.3.9 in [33], we have

$$\left\|\underline{A}^{-1} - A^{-1}\right\| \leq C_1 \epsilon_{mach},$$

where $C_1 = \dfrac{\left\|A^{-1}\right\| + \kappa(A)}{1 - \epsilon_{mach}\left\|A^{-1}\right\|}\left\|A^{-1}\right\|$ and $\kappa(A)$ is the condition number of A in the infinity norm. A little algebraic work gives

$$\left\|\varDelta^{k+1}\right\| \leq C_2 \left\|\underline{U}^{k}\right\| \epsilon_{mach}, \tag{51}$$

where $C_2 = \left\|A^{-1}\right\|\left(n + \dfrac{\left\|A^{-1}\right\| + \kappa(A)}{1 - \epsilon_{mach}\left\|A^{-1}\right\|}\|B\| + \|B\|\right)$.

Now we add up the error introduced in each and every step to have the accumulated error at $(K+1)$th step in the form of a discrete convolution

$$E^{K+1} = \sum_{j=1}^{K+1} \left(A^{-1}B\right)^{K+1-j} \varDelta^{j},$$

whose magnitude can be bounded as

$$\begin{align}
\left\|E^{K+1}\right\| &= \left\|\sum_{j=1}^{K+1} \left(A^{-1}B\right)^{K+1-j} \varDelta^{j}\right\| \tag{52} \\
&\leq (K+1) \sup_{0 \leq r \leq K} \left\|\left(A^{-1}B\right)^{r}\right\| \sup_{1 \leq j \leq K+1} \left\|\varDelta^{j}\right\| \notag \\
&\leq (K+1) \sup_{0 \leq r \leq K} \left\|\left(A^{-1}B\right)^{r}\right\| \sup_{1 \leq j \leq K+1} \left\|\underline{U}^{j}\right\| C_2 \epsilon_{mach} \notag \\
&= C_3 (K+1) \epsilon_{mach}, \tag{53}
\end{align}$$

where we have used (51) to come up with the constant coefficient

$$C_3 = \sup_{0 \leq r \leq K} \left\|\left(A^{-1}B\right)^{r}\right\| \sup_{1 \leq j \leq K+1} \left\|\underline{U}^{j}\right\| \left\|A^{-1}\right\| \left(n + \frac{\left\|A^{-1}\right\| + \kappa(A)}{1 - \epsilon_{mach}\left\|A^{-1}\right\|}\|B\| + \|B\|\right),$$

independent of K.

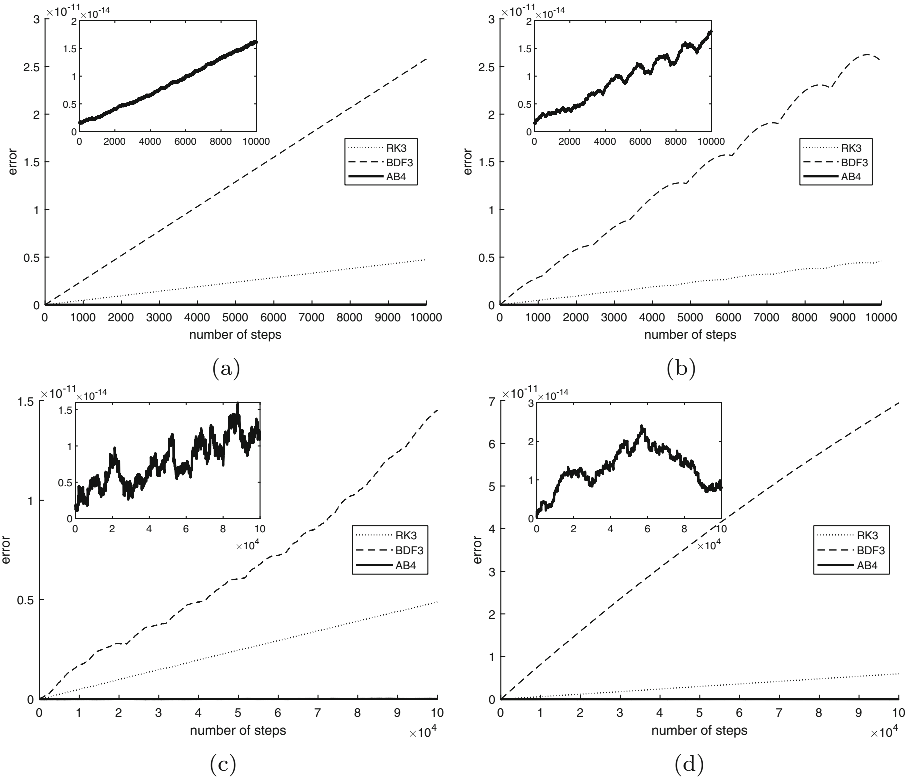


Fig. 7 The growth of the rounding error when solving (11) (top panes) and (12) (bottom panes) using ultraspherical spectral method (left panes) and Chebyshev pseudospectral method (right panes)

What (53) shows is that the accumulated error grows at worst *linearly* with the number of the time steps. We solve the one-dimensional transport equation (11) and the heat equation (12) using three different time marching schemes, i.e., RK3, AB4, and BDF3, and compare the computed solution with the exact solution. Sufficiently large n and small enough Δt are used so that there is no discretization error and the observed error is solely due to rounding. The error is plotted in Fig. 7 (left panes) to show its growth versus the number of time steps.

As shown in Fig. 7a, the errors grow exactly linearly for all three methods. The error of the AB4 method is relatively negligible compared to those of RK3 and BDF3, so its curve is indistinguishable from the x-axis in the same plot. The inset shows the linear growth of the error of AB4 using a different y-scale. The results shown in Fig. 7c look similar, only except that the error curves are somewhat more oscillatory, especially in the inset plot for AB4.

The different slopes of the curves are attributed to C_3 in (53). In fact, our calculation shows that it is the factor $\sup_{0\le r\le K}\left\|\left(A^{-1}B\right)^r\right\|$ that really makes a difference for these three methods. For example, this quantity is 3.9×10^{10}, 8.9×10^{6}, and 5.0 for BDF3, RK3, and AB4, respectively. Note that this factor is partly attributed to our analysis on the norms of the spatial discretization matrices in Sect. 3 and the Kreiss matrix theorem [30, Chapter 18].

What we also show in Fig. 7 (right panes) is how the rounding errors grow when Chebyshev pseudospectral method is used to solve (11) (Fig. 7b) and (12) (Fig. 7d). It is not surprising that they grow too at most linearly, since the model (50) and the analysis given above are also applicable to the Chebyshev pseudospectral method. We can see that the rounding errors are comparable in these two methods and this should also be expected by the reasoning right above Remark 3.1.

5 Computational Cost

As pointed out in [18], solving an almost banded system involves two steps: the QR factorization and the back substitution. They cost $\mathcal{O}(m^2 n)$ and $\mathcal{O}(mn)$ respectively, where n is the degrees of freedom and m is the bandwidth of the almost banded matrix.

The sparsity shown by Figs. 1 and 2 readily implies the same strategy for solving the resulting systems and, therefore, a computational cost of $\mathcal{O}(n)$ too for both the discretization approaches regardless of the time marching scheme.[7] However, when Approach 2 is employed we solve an upper triangular banded system (see Fig. 2) for which only the back substitution is needed. Moreover, note that since the boundary condition (10b) and the coefficients on the right-hand side of (10a) are independent of time, the QR factorization can be done once and for all at the beginning and at the subsequent steps only the back substitution is carried out.

The $\mathcal{O}(n)$ complexity is in stark contrast to the computational cost for solving (10) using the collocation-based pseudospectral method [28]. If an explicit time marching scheme is used, the cost to calculate the derivatives on the right-hand side of (10a) is $\mathcal{O}(n \log_2 n)$ with the aid of FFTs.[8] If an implicit method is used, a dense system with no particular structure needs to be solved by a direct method such as LU factorization at a cost of $\mathcal{O}(n^3)$ flops. Even though this cost is paid only once at the start of the time stepping and can be amortized over the subsequent steps, the cost of the backward substitution is still as high as $\mathcal{O}(n^2)$ since the system is dense. Furthermore, for an adaptive implementation similar to the one introduced below in Sect. 6, multiple or even a large number of LU decomposition may be needed, raising the cost significantly. These certify the great advantage of the ultraspherical spectral method in solving time-dependent PDEs.

6 Adaptivity

As time evolves, the solution to (10) may become spatially simpler or more complicated. It would be ideal if the method can take this into account and adapt the implementation for better efficiency but at the same time ensure that the degrees of freedom is large enough to guarantee an adequate resolution of the solution. This requires deciding a proper length of the solution vector at each time step.

Aurentz and Trefethen [2] propose an automated procedure in the context of function approximation for determining where to chop a Chebyshev series so that the truncated Chebyshev series is accurate and economical. The key of their chopping algorithm is the detection of a plateau, where the Chebyshev coefficients stay below a threshold and are sufficiently level. It is then based on this plateau that a chopping strategy is formulated. Algorithm 1 summarizes the plateau detection part of their chopping algorithm. As we can see, when we approximate a given function by Chebyshev series the emergence of a plateau signals sufficient resolution, and a large portion of the plateau and all the trailing coefficients beyond the plateau are discarded for efficiency (not indicated in Algorithm 1).

[7] For simplicity, we have omitted here the implied factor dependent of the bandwidth in the big-oh notation.

[8] In pseudospectral methods, variable coefficients are represented by diagonal matrices.

Algorithm 1 Detection of a plateau [2].

```
procedure PLATEAU($u$, $tol$)
    Step 1: Compute the normalized upper envelope of $u$.
    $envelope_j = \max_{j \le k \le n} |u_k|$
    if $envelop_1 \neq 0$ then
        $envelope = envelope/envelope_1$
    end if
    Step 2: Search for a plateau.
    for $j \leftarrow 2, n$ do
        $j_2 = round(1.25j + 5)$
        $e_1 = envelope(j)$
        $e_2 = envelope(j_2)$
        $r = 3\,(1 - \log(e_1)/\log(tol))$
        if $(e_1 == 0$ or $e_2/e_1 > r)$ then
            return $j, j_2$
        end if
    end for
end procedure
```

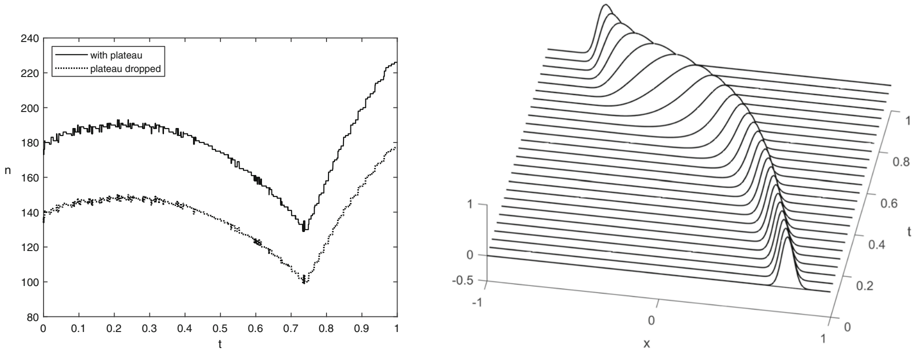
In the context of solving time-dependent PDEs, a plateau also serves as an indicator of adequate resolution. However, we only chop off the trailing coefficients beyond the plateau at each step.

Suppose we are marching to the $(k + r)$th step using the information at $t^k, t^{k+1}, \ldots, t^{k+r-1}$ by a multistep scheme or a Runge-Kutta method (for which $r = 1$). If there is no plateau in the computed solution u^{k+r}, we keep doubling the lengths of $u^k, u^{k+1}, \ldots, u^{k+r-1}$ by prolonging them with zeros and then re-calculate u^{k+r} until a plateau emerges. This way, we come up with the following algorithm which allows adaptivity for the solution — the solution vector is lengthened when an improved resolution may be effected or truncated when keeping some of the coefficients would not improve the resolution.

Algorithm 2 Adaptive stepping from $t^k, t^{k+1}, \ldots, t^{k+r-1}$ to t^{k+r}.

```
Stepping by a linear multistep or Runge-Kutta method to obtain the computed solution $u^{k+r}$. In case
$u^k, u^{k+1}, \ldots, u^{k+r-1}$ are not of the same length, extend the shorter vectors to the length of the longest
vector by prolonging them with zeros before stepping.
procedure ADAPT($u^k$, $u^{k+1}$, ..., $u^{k+r}$)
    L = LENGTH($u^{k+r}$)                    ▷ Function LENGTH returns the length of a vector.
    Call PLATEAU($u^{k+r}$, $tol$).
    if there is a plateau formed by $\{u_i^{k+r}\}_{i=j}^{j_2}$ then
        Drop $\{u_i^{k+r}\}_{i=j_2+1}^{L}$, use $u^{k+r} = \{u_i^{k+r}\}_{i=0}^{j_2}$ for computation at future steps.
    else
        $u^k = [u^k, 0, \ldots, 0]$ (padding with zeros so that the lengths of $u^k$ is $2L$) for $k = 0, 1, \ldots, r-1$
        Re-calculate $u^{k+r}$ by the same time marching scheme
        Call ADAPT($u^k$, $u^{k+1}$, ..., $u^{k+r}$)
    end if
end procedure
```

Note that the computed solution vectors fed into the calculation of the future steps are the ones with the plateau coefficients kept, i.e., only the trailing coefficients beyond the plateau are discarded. However, when a solution vector is no longer used for stepping, its plateau part can be safely dropped for saving storage, since keeping the plateau coefficients would not be of any help in improving the accuracy of the solution.



(a) Lengths of the solution vectors: plateau coefficients included (solid) and plateau discarded (dotted).

(b) Solution of (54).

Fig. 8 Solving transport equation (54) of variable speed with adaptivity

To demonstrate how Algorithm 2 works, we solve

$$u_t = c(x)u_x \quad \text{s.t. } u(1,t) = 0, \quad u(x,0) = e^{-400(x-0.75)^2}, \tag{54}$$

where $c(x) = 3/5 + 3\sin^2(x-1)^2$ is a variable propagation speed depending on x which results in a deformation of the left-travelling wave, as displayed in Fig. 8b. The solid line in Fig. 8a shows the evolution of the length n of the solution vector at each step, up to final time $t = 1$. This length includes the coefficients forming the plateau, whereas the dotted line shows the length if the plateau coefficients are discarded.

A noteworthy point is that the systems with different dimensions due to an adaptive implementation are not unrelated. Suppose that we solve with adaptivity and systems of dimensions $n_1 \times n_1$ and $n_2 \times n_2$ are solved by the QR factorization at two occasions with $n_2 > n_1$. Since the $n_2 \times n_2$ system is plainly an augment of the $n_1 \times n_1$ one by $n_2 - n_1$ more rows and columns, we can simply cache the QR factorization for whichever system comes first to speed up the calculation for the other. Hence, for an adaptive implementation with systems of various sizes, the actual cost could be as little as doing the QR factorization once — only for the system with the largest dimension.

Finally, we note that the CHEBFUN system [6], particularly its PDE solver PDE15S, offers a similar adaptivity in space. However, it is much more basic than the proposed one in that it does not have a mechanism for reducing the degrees of freedom when it is larger than it needs to be. Therefore, over-resolution may cause unnecessary drag in speed when the solution becomes spatially smoother.

7 Spatially Periodic Problems

Up to this point, our discussion has been concentrated on spatially non-periodic problems. For (4) subject to periodic boundary conditions, one can simply follow the same framework of [18] but take $\left\{e^{\pm ikx}\right\}_{k=0}^{\infty}$ as the basis functions, reproducing the tau-method [19]. With the Fourier basis, the λ-order differentiation operator remains sparse as

$$\mathcal{D}_\lambda = \operatorname{diag}\left(0, i^\lambda, (-i)^\lambda, (2i)^\lambda, (-2i)^\lambda, \ldots, (2k)^\lambda, (-2k)^\lambda, \ldots\right)$$

and there is no more need for conversion operators $\mathcal{S}_\lambda$, resulting in an even simpler implementation of the ultraspherical spectral method in the periodic case. However, for time-dependent PDEs with periodic boundary conditions, i.e., (10) with (10b) replaced by periodic boundary conditions, time marching is not as easy as in the non-periodic case. For the simple cases where the right-hand side of (10a) is an odd-order spatial derivative of u, all the eigenvalues of the spatial discretization matrix reside on the imaginary axis for which only the schemes with a stability region enclosing the origin and its neighborhood along the imaginary axis are applicable. For example, for the one-dimensional transport equation (11) with periodic boundary conditions, this immediate disqualifies all the explicit Runge-Kutta methods, the first two Adams-Bashforth methods, the Adams-Moulton methods of 2, 3, and 4 steps, and the BDF methods with more than 2 steps.

8 Nonlinearity

So far, the discussion has been concentrated on linear problems, which help simplify the analysis substantially. We now return to (1) where $\mathcal{F}$ also includes a nonlinear part as in (2). In the remainder of this article, we slightly abuse the notation by assuming that the nonlinear operator $\mathcal{F}$ takes in and returns Chebyshev and $C^{(\lambda)}$ coefficients respectively, instead of function values. This way, the input and the output of $\mathcal{F}$ are consistent with those of the linear part $\mathcal{L}$. For Approach 1, the fully discretized system reads

$$\begin{aligned}&\begin{pmatrix}\mathcal{B}\mathcal{P}_n^\top\\ \mathcal{P}_{n-N}\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\end{pmatrix}\mathcal{P}_n\boldsymbol{u}^{k+r}\\ &\quad=\begin{pmatrix}c\\ \sum_{j=0}^{r-1}(\beta_j h\mathcal{P}_{n-N}\mathcal{F}(t_{k+j},\mathcal{P}_n\boldsymbol{u}^{k+j})-\alpha_j\mathcal{P}_{n-N}\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+j})\end{pmatrix},\end{aligned}\tag{55}$$

if an explicit multistep method ($\beta_r = 0$) is used. For an implicit multistep method ($\beta_r \neq 0$), we end up with the nonlinear equation

$$\begin{aligned}&\begin{pmatrix}\mathcal{B}\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+r}\\ \mathcal{P}_{n-N}\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+r}-h\beta_r\mathcal{P}_{n-N}\mathcal{F}(t_{k+r},\mathcal{P}_n\boldsymbol{u}^{k+r})\end{pmatrix}\\ &\quad=\begin{pmatrix}c\\ \sum_{j=0}^{r-1}(\beta_j h\mathcal{P}_{n-N}\mathcal{F}(t_{k+j},\mathcal{P}_n\boldsymbol{u}^{k+j})-\alpha_j\mathcal{P}_{n-N}\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+j})\end{pmatrix}.\end{aligned}\tag{56}$$

The last two equations should be compared with (21). If a Runge–Kutta method is used, (22) should be adapted to become

$$\begin{pmatrix}\mathcal{B}\mathcal{P}_n^\top\\ \mathcal{P}_{n-N}\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\end{pmatrix}y_j=\begin{pmatrix}0\\ h\mathcal{P}_{n-N}\mathcal{F}(t_k+\theta_j h,\mathcal{P}_n\boldsymbol{u}^k+\mu_j y_{j-1})\end{pmatrix},\tag{57}$$

For Approach 2, explicit and implicit multistep methods leads to

$$\begin{aligned}&\mathcal{P}_n\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+r}\\ &\quad=h\sum_{j=0}^{r-1}\beta_j\mathcal{P}_n\mathcal{F}(t_{k+j},\mathcal{P}_n\boldsymbol{u}^{k+j})-\sum_{j=0}^{r-1}\alpha_j\mathcal{P}_n\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+j}\end{aligned}\tag{58}$$

and

$$\begin{aligned}&\mathcal{P}_n\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+r}-h\beta_r\mathcal{P}_n\mathcal{F}(t_{k+r},\mathcal{P}_n\boldsymbol{u}^{k+r})\\&\quad=h\sum_{j=0}^{r-1}\beta_j\mathcal{P}_n\mathcal{F}(t_{k+j},\mathcal{P}_n\boldsymbol{u}^{k+j})-\sum_{j=0}^{r-1}\alpha_j\mathcal{P}_n\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top\mathcal{P}_n\boldsymbol{u}^{k+j},\end{aligned} \tag{59}$$

respectively, where Runge–Kutta methods gives

$$\mathcal{P}_n\mathcal{S}_{N-1}\dots\mathcal{S}_0\mathcal{P}_n^\top y_j=h\mathcal{P}_n\mathcal{F}(t_k+\theta_jh,\mathcal{P}_n\boldsymbol{u}^k+\mu_jy_{j-1}), \tag{60}$$

which should be contrasted with (25). The nonlinear part $\mathcal{N}(t,u)$ of $\mathcal{F}(t,u)$ at specific t and u is usually evaluated by plugging in the value of t,[9] sampling $\mathcal{N}(u(x))$ at Chebyshev grids in x of increasing size, calculating the Chebyshev coefficients by FFT until complete resolution, and converting them to $C^{(\lambda)}$ coefficients. The total cost is dominated by the few FFTs for the value-to-coefficient transform. Thus, the nominal complexities for solving (55), (57), (58), and (60) are all $\mathcal{O}(n\log_2 n)$, where the linear complexity of system solving is prevailed over by the complexity of the evaluation of the nonlinear terms. Note that (56) and (59) are nonlinear equations of $\mathcal{P}_n\boldsymbol{u}^{k+r}$ and the cost of solution may be the greatest concern since the multiplication operators lose bandedness. However, it has been shown that fast solution to the nonlinear systems obtained from ultraspherical discretization can still be effected with $\mathcal{O}(n\log_2 n)$ flops per iteration by an inexact Newton-GMRES method [21]. Since the solution at the previous time step can always serve as a good initial iterate for the next step, Newton's method usually skips the global stage and converges to machine precision in very few iterations. Thus, ultraspherical spectral method guarantees fast solution for virtually all the scenarios – explicit and implicit schemes, linear and nonlinear equations. This is in marked contrast to solving time-dependent PDEs with the collocation-based pseudospectral method as the corresponding differentiation matrices are dense and much less structured.

The convergence of the solutions obtained by the two approaches in the nonlinear case is guaranteed if $\mathcal{F}(t,u)$ satisfies Lipschitz conditions. This is met by virtually all the real-world problems.

To see how rounding errors accumulate, we replace the iterative model (50) by

$$U^{k+1}=g(U^k),$$

where g is the nonlinear map corresponding to $\mathcal{F}$. It can be shown that the modulus of the rounding error

$$\left\|\Delta^{k+1}\right\|=\left\|\underline{U}^{k+1}-g(\underline{U}^k)\right\|=\left\|fl(\underline{g}(\underline{U}^k))-g(\underline{U}^k)\right\|\le C_1\epsilon,$$

where $\underline{g}$ denotes the floating point approximation to g and $C_1=(2+\epsilon)\left\|g(\underline{U}^k)\right\|$. The accumulative error E^{K+1} at $(K+1)$th step is bounded by

$$\left\|E^{K+1}\right\|=\left\|\sum_{j=1}^{K+1}g^{K+1-j}\left(\Delta^j\right)\right\|\le C_2(K+1)\epsilon_{mach},$$

where $C_2=(2+\epsilon)\sup_{0\le r\le K}\|g^r(\cdot)\|\sup_{1\le j\le K+1}\left\|g(\underline{U}^{j-1})\right\|$. This constant C_2 is, again, solely determined by the nonlinear map g. The conclusion that the rounding error, in the worst possible scenario, renders a linear growth is unchanged.

How the adaptivity described in (6) is implemented is not affected by the nonlinearity and, thus, stays the same. For nonlinear periodic problems, the evaluation of the nonlinear

[9] In practice, $\mathcal{N}$ is often independent of t, being a univariate function of u.

term $\mathcal{F}$ is done with the Fourier coefficients, analogous to their Chebyshev counterpart in a straightforward manner.

The implementation of more advanced methods, such as the implicit-explicit differencing method, shares substantial similarities with those of the multistep and the Runge–Kutta methods. We choose to omit the discussion here.

9 Exponential Integrators

We could have closed this article at the end of last section. But the exponential integrators, also known as exponential time differencing, deserve a detailed discussion – it is arguably the most powerful method for solving stiff ODE initial value problems. More importantly, the combination of the exponential integrators and the ultraspherical spectral method turns out to be extremely efficient, as we shall see below.

Consider the main equation (1a), that is,

$$\mathcal{T}u = \mathcal{L}u + \mathcal{N}(t, u).$$

Suppose the linear operator $\mathcal{L}$ is expressed as in (13). To ensure that the coefficients produced on both sides are in the same space, we premultiply the right-hand side by $\mathcal{S}_0^{-1} \ldots \mathcal{S}_{N-1}^{-1}$ to obtain

$$\mathcal{T}\boldsymbol{u} = \mathcal{S}_0^{-1} \ldots \mathcal{S}_{N-1}^{-1}\mathcal{L}\boldsymbol{u} + \mathcal{N}(t, \boldsymbol{u}).$$

Following Approach 2 in Sect. 2, we ignore the boundary conditions momentarily and integrate the last equation on both sides from t_k to t_{k+1} to have the variation-of-constant formula in terms of the ultraspherical spectral operators

$$\begin{aligned} \boldsymbol{u}(t_{k+1}) = & e^{h\mathcal{S}_0^{-1}\ldots\mathcal{S}_{N-1}^{-1}\mathcal{L}}\boldsymbol{u}(t_k) + e^{h\mathcal{S}_0^{-1}\ldots\mathcal{S}_{N-1}^{-1}\mathcal{L}} \\ & \times \int_0^h e^{\tau\mathcal{S}_0^{-1}\ldots\mathcal{S}_{N-1}^{-1}\mathcal{L}}\mathcal{N}(t_k + \tau, \boldsymbol{u}(t_k + \tau))\, d\tau. \end{aligned} \tag{61}$$

Different approximations to the integral in (61) lead to various classes of exponential integrator [14]. If the integrand in (61) is replaced by its polynomial interpolant at certain distinct points in $[t_k, t_{k+1}]$, we have the exponential multistep methods

$$\boldsymbol{u}^{k+1} = e^{h\mathcal{S}_0^{-1}\ldots\mathcal{S}_{N-1}^{-1}\mathcal{L}}\boldsymbol{u}^k + h\sum_{j=0}^{p-1}\zeta_j(h\mathcal{S}_0^{-1}\ldots\mathcal{S}_{N-1}^{-1}\mathcal{L})\nabla^j\boldsymbol{v}^k, \tag{62}$$

where $\boldsymbol{u}^k = \boldsymbol{u}(t_k)$, $\boldsymbol{v}^k = \mathcal{N}(t_k, \boldsymbol{u}^k)$, and $\nabla^j\boldsymbol{v}^k$ denotes the jth backward difference defined recursively by $\nabla^0\boldsymbol{v}^k = \boldsymbol{v}^k$ and $\nabla^{j+1}\boldsymbol{v}^k = \nabla^j\boldsymbol{v}^k - \nabla^j\boldsymbol{v}^{k-1}$. The weights ζ_j can be calculated via the recurrence relation

$$\begin{aligned} \zeta_0(z) &= \varphi_1(z), \\ z\zeta_j(z) + 1 &= \sum_{i=0}^{j-1}\frac{1}{j-i}\zeta_i(z), \end{aligned}$$

where $\varphi_1(z) = \frac{e^z - 1}{z}$ is one of the so called φ-functions. These φ-functions can be generated from $\varphi_0(z) = e^z$ and the recurrence relation [13]

$$\varphi_{j+1}(z) = \frac{\varphi_j(z) - \varphi_j(0)}{z}.$$

Similarly, replacing the integrand in (61) by its Taylor expansion at t_k gives the exponential Runge–Kutta methods

$$\boldsymbol{u}^{k+1} = \exp(h\mathcal{S}_0^{-1} \ldots \mathcal{S}_{N-1}^{-1}\mathcal{L})\boldsymbol{u}^k + h\sum_{i=1}^{s} b_i(h\mathcal{S}_0^{-1} \ldots \mathcal{S}_{N-1}^{-1}\mathcal{L})\boldsymbol{v}^{ki}, \tag{63a}$$

$$\boldsymbol{u}^{ki} = \exp(c_i h\mathcal{S}_0^{-1} \ldots \mathcal{S}_{N-1}^{-1}\mathcal{L})\boldsymbol{u}^k + h\sum_{j=1}^{s} a_{ij}(h\mathcal{S}_0^{-1} \ldots \mathcal{S}_{N-1}^{-1}\mathcal{L})\boldsymbol{v}^{kj}, \tag{63b}$$

where $\boldsymbol{u}^{ki} = \boldsymbol{u}(t_k + c_i h)$, $\boldsymbol{v}^{ki} = \mathcal{N}(t_k + c_i h, \boldsymbol{u}^{ki})$. Like the Runge-Kutta methods, the weights a_{ij} and b_i satisfy $\sum_{j=1}^{s} b_j(z) = \varphi_1(z)$ and $\sum_{j=1}^{s} a_{ij}(z) = c_i\varphi_1(c_i z)$ for $i = 1, 2, \ldots, s$. For (63) to be explicit, it is also required that $c_1 = 0$ and $a_{ij}(z) = 0$ for $1 \le i \le j \le s$.

Various exponential Runge–Kutta methods have been constructed and some of the most commonly used higher-order schemes are those proposed by Cox and Matthews [5], Krogstad [17], and Hochbruck and Ostermann [13]. For example, the method by Krogstad is given by the following Butcher tableau

$$\begin{array}{c|cccc}
c_1 = 0 & & & & \\
c_2 = \frac{1}{2} & a_{21} = \frac{1}{2}\varphi_{1,2} & & & \\
c_3 = \frac{1}{2} & a_{31} = \frac{1}{2}\varphi_{1,3} - \varphi_{2,3} & a_{32} = \varphi_{2,3} & & \\
c_4 = 1 & a_{41} = \varphi_{1,4} - 2\varphi_{2,4} & & a_{43} = 2\varphi_{2,4} & \\
\hline
 & b_1 = \varphi_1 - 3\varphi_2 + 4\varphi_3 & b_2 = 2\varphi_2 - 4\varphi_3 & b_3 = b_2 & b_4 = -\varphi_2 + 4\varphi_3
\end{array},$$

where $\varphi_{i,j}(z) = \varphi_i(c_j z)$.

To make the exponential multistep method (62) and exponential Runge-Kutta method (63) practical, we still need to truncate all the operators and infinite vectors to finite dimensions. This is done by replacing $h\mathcal{S}_0^{-1} \ldots \mathcal{S}_{N-1}^{-1}\mathcal{L}$ by $G = h\mathcal{P}_n\mathcal{S}_0^{-1} \ldots \mathcal{S}_{N-1}^{-1}\mathcal{L}\mathcal{P}_n^{\top}$ and only retaining the first n components of $\boldsymbol{u}^k$, $\boldsymbol{v}^k$, and $\boldsymbol{v}^{ki}$. For convenience, we denote by u^k, v^k, and v^{ki} respectively the vectors formed by the first n components of $\boldsymbol{u}^k$, $\boldsymbol{v}^k$, and $\boldsymbol{v}^{ki}$.

The implementation of the exponential multistep and Runge–Kutta methods reviewed above boils down to the calculation of the product $\varphi_j(G)\xi$, where we use ξ to denote any of u^k, v^k, and v^{ki}. Since evaluating $\varphi_j(G)$ directly usually suffers from large cancellation errors for G of small magnitude, the evaluation of $\varphi_j(G)$ should be done via the Dunford-Taylor integral [16]. It is further shown that

$$\varphi_j(z) = \frac{1}{2\pi i}\int_{\Gamma} \frac{e^s}{s^j}\frac{1}{s - z}\mathrm{d}s,$$

where Γ is a closed contour enclosing all the eigenvalues of G [23]. Replacing Γ by a θ-parameterized Hankel contour $\phi(\theta)$, such as a Talbot's contour [31], leads to

$$\varphi_j(z) = \frac{1}{2\pi i}\int_{-\infty}^{+\infty} \frac{e^{\phi(\theta)}}{\phi(\theta)^j}\frac{1}{\phi(\theta) - z}\phi'(\theta)\mathrm{d}\theta.$$

By truncating the integration interval to $[-\pi, \pi]$ and approximating the integral by q-point trapezoidal rule, we have a q-term sum-of-pole approximation of $\varphi_j(z)$

$$r_{(j)}^{CI}(z) = \sum_{l=1}^{q} \frac{w_l^{CI}}{z - z_l^{CI}}, \tag{64}$$

where $w_l^{CI} = iq^{-1}e^{\phi_l}\phi_l'/\phi_l^j$, $\phi_l = z_l^{CI} = \phi(\theta_l)$, $\phi_l' = \phi'(\theta_l)$, and $\theta_l = \pi(2l-q-1)/(q-1)$ for $l = 1, 2, \ldots, q$.

One can also use the Carathéodory-Fejér approximation [29, 31] to obtain a near-best rational approximation to $\varphi_j(z)$

$$r_{(j)}^{CF}(z) = \sum_{l=1}^{q} \frac{w_l^{CF}}{z - z_l^{CF}}, \tag{65}$$

which is also in the sum-of-pole form as the one found by contour integral. The poles z_l^{CF} and weights w_l^{CF} of the CF approximation to $\varphi_j(z)$ usually differ for different j.

Note that when (64) or (65) are used, the calculation of $\varphi_j(G)\xi$ turns to solving linear systems $(G - z_l I)x_l = \xi$, or equivalently

$$(hL - z_l S_{N-1} \ldots S_0)x_l = S_{N-1} \ldots S_0 \xi, \tag{66}$$

for $l = 1, 2, \ldots, q$. What makes the exponential integrator even more powerful in the current context is the fact that (66) is a banded system as L and $S_{N-1} \ldots S_0$ are both banded. When the poles and weights are known from pre-computation, the total cost $\mathcal{O}(qn)$ for computing each of $\varphi_j(G)\xi$ is significantly less than if the collocation-based pseudospectral method were used, for which (66) is dense. Note that the convergence rate of $r_{(j)}^{CF}(z)$ is twice of that of $r_{(j)}^{CI}(z)$ and further speed-up for the CF method can be achieved by using common poles for all φ_j, whereas the contour-based method can compute the weights and poles cheaply. For comparisons of the contour-based and the CF methods, see [23, 31].

Here is a quick example of exponential integrating the Fisher equation

$$u_t = 0.001u_{xx} + u - u^2, \quad x \in [-1, 1],$$

subject to the homogeneous Dirichlet boundary conditions and the initial condition $u(0, x) = (1 - \tanh(40x/\sqrt{6}))/4$ using the ultraspherical and the pseudospectral spectral methods. For both the methods, we choose $n = 512$ and integrate up to $t = 10$ with steps of size $1/n^2$, contrasted with the $\mathcal{O}(1/n^4)$ restriction derived in Sect. 3. It takes 2.7 seconds for the ultraspherical spectral method to finish the simulation, which is compared with 3.9 seconds using the collocation-based pseudospectral method. Two methods have comparable accuracy of $\mathcal{O}(10^{-9})$ in this experiment. The acceleration is more substantial when the degrees of freedom is greater.

Remark 9.1 Different exponential integrators vary by how the integral of the nonlinear term is approximated. For a linear problem, i.e., $\mathcal{N} = 0$, all the exponential integrators coincide and give the same solution

$$u^{k+1} = \varphi_0(G)u^k. \tag{67}$$

Since this solution is exact, the step size is unlimited,[10] and the exponential integrators are also superb in solving linear problems. For example, exponential integrating the heat equation

$$u_t = 0.1u_{xx}, \;\; u(0,1) = u(0,-1) = 0, \;\; u(x,0) = \sin(2\pi x)$$

by the ultraspherical spectral method with $n = 32$ allows the step size to be as large as 0.1. With this step size, the absolute error of the computed solution at $t = 10$ is about $1.1352e{-}14$.

CHEBFUN has an `expm` function for calculating operator exponentials, which overrides the MATLAB function with the same name but working on matrices. It can exactly be used for evaluating $\varphi_0(G)$ in (67). However, CHEBFUN does not offer any more functionality in exponential integration beyond the linear case. Additionally, the CHEBFUN `expm` explicitly forms the matrix that approximates $\varphi_0(G)$ before it is applied to the vector u^k, therefore the sparsity seen in (66) is not taken advantage of and the storage cost becomes $\mathcal{O}(n^2)$ instead of $\mathcal{O}(n)$.

10 Conclusion and Remarks

We have applied the ultraspherical spectral method to solving time-dependent PDEs by offering two approaches for discretization and have examined a few key aspects of the proposed method, including the stability of stepping, the error accumulation, and the computational cost, for both the linear and nonlinear cases. Careful comparison shows that the new method ties with the Chebyshev pseudospectral method in terms of stability and error and has a clear advantage in speed and adaptivity.

So far, we have seen banded or almost-banded systems in two scenarios – the implicit multi-step methods like the Adam-Moulton and BDF methods and the exponential integrator. Since the sparsity is a consequence of the employment of the ultraspherical spectral method for the spatial discretization, many more time marching schemes can also enjoy the fast linear algebra when used for solving time-dependent PDEs. More advanced examples include the spectral deferred correction method [8] for obtaining high accuracy solutions, the parareal method for time integration in parallel [10], the symplectic integrator for Hamiltonian systems [22], just to name a few. When stiffness requires the use of basic implicit methods as the underlying driving schemes, the method benefits from the resulting sparse linear systems.

The speed-up that we have seen could be even more conspicuous for problems in higher spatial dimensions since the degrees of freedom n is squared or cubed.

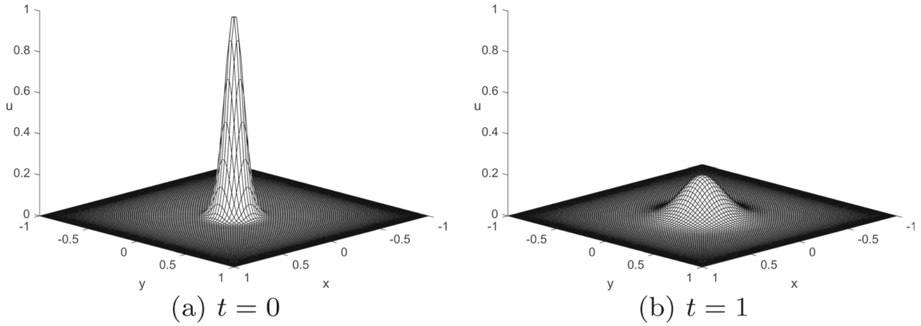
One thing we have left out but worth mentioning is the handling of a second derivative in time. If high accuracy is not required, it is usually approximated by the simple leap frog formula. A more general approach is to reduce an equation with a second-order temporal derivative to a system of two equations with first-order derivatives in time. For instance, $u_{tt} = \mathcal{F}(t, u(x,t))$ is reduced to

$$\begin{aligned} v_t &= \mathcal{F}(t, u(x,t)), \\ u_t &= v(x,t), \end{aligned}$$

where the methods covered in the previous sections can be applied.

Another possibility that is beyond the scope of this work is the extension of the ultraspherical spectral method to time-dependent problems in multiple spatial dimensions. The analysis may be more complicated and subtler than the present one, partly due to the boundary conditions. However, our initial numerical experiments show that the discretization approaches

[10] In practice, φ_0 can hardly be evaluated accurately for extremely large argument due to the conditioning.



(a) $t = 0$ (b) $t = 1$

Fig. 9 Solving a two dimensional heat equation by Approach 1 with $n_x = n_y = 128$ and the backward Euler with $h = 0.001$

discussed in Sect. 2 work well as expected. Figure 9b displays the solution at $t = 1$ to the two-dimensional heat equation in a square domain subject to homogeneous boundary conditions

$$\begin{aligned} &u_t = 0.01\left(u_{xx} + u_{yy}\right), \quad (x, y) \in [-1, 1] \times [-1, 1], \\ &\text{s.t.} \quad u|_{\Gamma} = 0 \quad \text{and} \quad u(x, 0) = e^{-100(x^2+y^2)}, \end{aligned}$$

where initial profile is shown in Fig. 9a.

As the ultraspherical spectral method has been widely accepted in the last decade, we believe the methods proposed in this article can serve as a natural companion of the ultraspherical spectral method for solving time-dependent problems and the analysis we have carried out can help understand and interpret the numerical results obtained from using these methods.

Acknowledgements The authors are grateful to Marco Fasondini, Sheehan Olver, and Ouyuan Qin for reading the manuscript and providing extremely helpful suggestions.

References

1. Ascher, U.M., Petzold, L.R.: Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations, vol. 61. SIAM, Philadelphia (1998)
2. Aurentz, J.L., Trefethen, L.N.: Chopping a Chebyshev series. ACM Trans. Math. Softw. **43**(4), 1–21 (2017)
3. Burns, K.J., Vasil, G.M., Oishi, J.S., Lecoanet, D., Brown, B.P.: Dedalus: a flexible framework for numerical simulations with spectral methods. Phys. Review Res. **2**(2), 023068 (2020)
4. Butcher, J.C.: Numerical Methods for Ordinary Differential Equations. Wiley, New York (2016)
5. Cox, S.M., Matthews, P.C.: Exponential time differencing for stiff systems. J. Comput. Phys. **176**(2), 430–455 (2002)
6. Driscoll, T.A., Hale, N., Trefethen, L.N.: Chebfun guide (2014)
7. Dubiner, M.: Asymptotic analysis of spectral methods. J. Sci. Comput. **2**(1), 3–31 (1987)
8. Dutt, A., Greengard, L., Rokhlin, V.: Spectral deferred correction methods for ordinary differential equations. BIT Numer. Math. **40**(2), 241–266 (2000)
9. Fortunato, D., Hale, N., Townsend, A.: The ultraspherical spectral element method. J. Comput. Phys. **436**, 110087 (2021)
10. Gander, M.J., Vandewalle, S.: Analysis of the parareal time-parallel time-integration method. SIAM J. Sci. Comput. **29**(2), 556–578 (2007)
11. Gottlieb, D., Lustman, L.: The spectrum of the Chebyshev collocation operator for the heat equation. SIAM J. Numer. Anal. **20**(5), 909–921 (1983)

12. Hairer, E., Nørsett, S.P., Wanner, G.: Solving Ordinary Differential Equations. I, Nonstiff Problems. Springer, Berlin (1993)
13. Hochbruck, M., Ostermann, A.: Exponential Runge-Kutta methods for parabolic problems. Appl. Numer. Math. **53**(2–4), 323–339 (2005)
14. Hochbruck, M., Ostermann, A.: Exponential integrators. Acta Numer. **19**, 209–286 (2010)
15. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (2012)
16. Kassam, A.K., Trefethen, L.N.: Fourth-order time-stepping for stiff PDEs. SIAM J. Sci. Comput. **26**(4), 1214–1233 (2005)
17. Krogstad, S.: Generalized integrating factor methods for stiff PDEs. J. Comput. Phys. **203**(1), 72–88 (2005)
18. Olver, S., Townsend, A.: A fast and well-conditioned spectral method. SIAM Rev. **55**(3), 462–489 (2013)
19. Ortiz, E.L.: The tau method. SIAM J. Numer. Anal. **6**(3), 480–492 (1969)
20. Overton, M.L.: Numerical Computing with IEEE Floating Point Arithmetic. SIAM, Philadelphi (2001)
21. Qin, O., Xu, K.: Solving nonlinear ODEs with the ultraspherical spectral method. submitted (2022)
22. Ruth, R.D.: A canonical integration technique. IEEE Trans. Nucl. Sci. (CERN-LEP-TH-83-14) **30**, 2669–2671 (1983)
23. Schmelzer, T., Trefethen, L.N.: Evaluating matrix functions for exponential integrators via Carathéodory-Fejér approximation and contour integrals. Electron. Trans. Numer. Anal. **29**, 1–18 (2007)
24. Słomka, J., Townsend, A., Dunkel, J.: Stokes' second problem and reduction of inertia in active fluids. Phys. Review Fluids **3**(10), 103304 (2018)
25. Sneddon, G.: Second-order spectral differentiation matrices. SIAM J. Numer. Anal. **33**(6), 2468–2487 (1996)
26. Townsend, A.: Computing with functions in two dimensions. Ph.D. thesis, Oxford University, UK (2014)
27. Townsend, A., Olver, S.: The automatic solution of partial differential equations using a global spectral method. J. Comput. Phys. **299**, 106–123 (2015)
28. Trefethen, L.N.: Spectral Methods in MATLAB. SIAM, Philadelphia, PA (2000)
29. Trefethen, L.N.: Approximation Theory and Approximation Practice, Extended SIAM, Philadelphia, PA (2019)
30. Trefethen, L.N., Embree, M.: Spectra and Pseudospectra. Princeton University Press, Princeton (2020)
31. Trefethen, L.N., Weideman, J.A.C., Schmelzer, T.: Talbot quadratures and rational approximations. BIT Numer. Math. **46**(3), 653–670 (2006)
32. Van der Waerden, B.L.: Algebra, vol. 2. Springer, Berlin (2003)
33. Watkins, D.S.: Fundamentals of Matrix Computations. Wiley, New York (2010)
34. Weideman, J., Trefethen, L.N.: The eigenvalues of second-order spectral differentiation matrices. SIAM J. Numer. Anal. **25**(6), 1279–1298 (1988)

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