Fast and high-order approximation of parabolic equations using hierarchical direct solvers and implicit Runge-Kutta methods

Ke Chen[1*], Daniel Appelö[2], Tracy Babb[3], Per-Gunnar Martinsson[4]

[1*]Department of Mathematics, University of Maryland, 4176 Campus Drive, College Park, 20742, MD, USA.
[2]Department of Mathematics, Virginia Tech., 225 Stanger Street, Blacksburg, 24061, VA, USA.
[3]Applied Mathematics, University of Colorado Boulder, 2300 Colorado Avenue, Boulder, 80309, CO, USA.
[4]Department of Mathematics, University of Texas at Austin, 2515 Speedway, Austin, 78712, TX, USA.

*Corresponding author(s). E-mail(s): kechen@umd.edu;
Contributing authors: appelo@vt.edu; tracy.babb@colorado.edu; pgm@oden.utexas.edu;

Abstract

A stable and high-order accurate solver for linear and nonlinear parabolic equations is presented. An additive Runge-Kutta method is used for the time stepping, which integrates the linear stiff terms by an explicit singly diagonally implicit Runge-Kutta (ESDIRK) method and the nonlinear terms by an explicit Runge-Kutta (ERK) method. In each time step, the implicit solve is performed by the recently developed Hierarchical Poincaré-Steklov (HPS) method. This is a fast direct solver for elliptic equations that decomposes the space domain into a hierarchical tree of subdomains and builds spectral collocation solvers locally on the subdomains. These ideas are naturally combined in the presented method since the singly diagonal coefficient in ESDIRK and a fixed time-step ensures that the coefficient matrix in the implicit solve of HPS remains the same for all time stages. This means that the precomputed inverse can be efficiently reused, leading to a scheme with complexity (in two dimensions) $\boldsymbol{\mathcal{O}(N^{1.5})}$ for the precomputation where the solution operator to the elliptic problems is built, and then $\boldsymbol{\mathcal{O}(N \log N)}$ for the solve in each time step . The stability of the method is proved for first order in time and any order in space, and numerical evidence substantiates a claim of stability for a much broader class of time discretization methods. Numerical experiments supporting the accuracy of efficiency of the method in one and two dimensions are presented.

Keywords: ESDIRK, parabolic, direct solver, high-order, hierarchical

1 Introduction

In this paper we consider numerical methods for solving parabolic equations of the form

$$\begin{aligned} u_t &= \mathcal{L}u + q + g(u)\,, && \text{in } (0,T)\times\Omega \\ u &= f\,, && \text{on } (0,T)\times\partial\Omega \\ u &= u_0\,, && \text{on } \{0\}\times\Omega \end{aligned} \qquad (1)$$

where $\mathcal{L}$ denotes a general second order elliptic differential operator and Ω is a bounded domain in $\mathbb{R}^d$. The function $q = q(t,x)$ denotes an external source and $g = g(t,u)$ denotes a nonlinear term. We consider the case of Dirichlet boundary conditions, f and denote the initial data by u_0. The nonlinear term is assumed to contain derivative operators of degree no greater than one so that the elliptic term $\mathcal{L}u$ dominates g, i.e. the equation is of parabolic type. To solve this equation numerically, it is usually preferred to discretize the linear parabolic part by an implicit method to avoid numerical stiffness. In each timestep a linear elliptic problem must be solved and when there are many timesteps it can become advantageous to use a direct solver. This is of course especially true if the complexity (as it is here) of the linear solver is good. In fact, although the pre-computation needed to build a fast direct solver can be more expensive than solving the equation once with an iterative solver, once a solution operator has built, each subsequent solve is very fast. In this paper we use an efficient direct elliptic solver called "Hierarchical Poincaré-Steklov(HPS)" coupled with high order Runge-Kutta(RK) time discretization to develop a fast solver to the parabolic equation (1).

The HPS solver is a domain decomposition scheme with spectral collocation discretization on each subdomain. It is drew from the classical nested dissection and multifrontal methods [1, 2] often used for low order finite difference discertizations but differs from these in that it achieves very high order of accuracy of spatial derivatives. The HPS method was first proposed in [3] for elliptic and Helmholtz equations and later generalized in [4–6] for general elliptic equation and higher dimensions. The HPS is a highly efficient direct solver to elliptic equations with high order of convergence [7–9]. These advantages remain when HPS is used for the elliptic solve in combination with implicit Runge-Kutta discretizations of parabolic equations, and preliminary results [10, 11] indicated that the resulting scheme is stable and efficient. However, challenges remain for proving stability of explicit-implicit schemes. For example, there is usually an order barrier for bound preserving higher-order implicit schemes [12] and developing a second order in time for general partial differential equations is still open, though several second order schemes exist for kinetic equations [13]. In this manuscript we combine high-order Runge-Kutta schemes with HPS to develop a fast and accurate solver for general parabolic equations. Stability is rigorously proven for the case where HPS of any spatial order of accuracy is combined with a first order accurate time discretization (backward Euler). That the method remains stable for high order time discretizations as well is substantiated through extensive numerical examples in one and two dimensions.

In Section 2 we introduce the HPS method for elliptic equations. Section 3 combines HPS with high order Runge-Kutta time discretization and describes the proposed solvers for general parabolic equations. In Section 4 we investigate the stability of the scheme. Numerical examples for one and two dimensional problems are given in Section 5.

2 The HPS method for time-independent problems

In this section, we briefly review the HPS method [3–6, 14] that we use for the spatial discretization; for additional details, we refer to [9, Sec. 24–26]. For concreteness, let us consider the Dirichlet boundary value problem,

$$\begin{aligned} \mathcal{A}u(x) &= g(x)\,, \qquad x \in \Omega, \\ u(x) &= f(x)\,, \quad x \in \partial\Omega, \end{aligned} \tag{2}$$

where $\mathcal{A}$ is a general second order elliptic differential operator

$$\begin{aligned} \mathcal{A}u(x) = &- c_{11}(x)\partial_1^2 u(x) - 2c_{12}(x)\partial_1\partial_2 u(x) - c_{22}(x)\partial_2^2 u(x) \\ &+ c_1(x)\partial_1 u(x) + c_2(x)\partial_2 u(x) + c(x)u(x)\,. \end{aligned} \tag{3}$$

Here $g(x)$ is the source term and $f(x)$ the boundary data. The HPS method first partitions the domain $\Omega \subset \mathbb{R}^2$ into a hierarchical tree of subdomains Ω^τ , $\tau = 1, \ldots, N$ and uses spectral collocation methods to discretize the local elliptic operators $\mathcal{A}^\tau$. After the discretization, local solution operators and Dirichlet to Neumann (DtN) maps restricted in each subdomain are built at the bottom level, where only small matrix inversion is involved. These local operators are then used to build the global direct solvers via a hierarchical merge from the bottom level to the top level. Once the global solver is constructed, HPS can can solve (2) for multiple sources and boundary data via a top-to-bottom sweep. In the implementation of HPS, we assume the mixed term $c_{12}(x) = 0$ so one can ignore the corner points in Figure 1. For problems involving a nonzero mixed term $c_{12}(x)$, one can also ignore the corner points in HPS, and then apply extrapolation methods to estimate the values on corner points. More details can be found in Chapter 24 of [9].

2.1 Discretization and numerical schemes

For simplicity we assume the domain Ω is rectangular. The domain is first partitioned into two children subdomains and then each children subdomain is further partitioned in a similar manner. This hierarchical partition will stop until the leaf children reach the preset square size and consequently a hierarchical tree of domains are formed. Eventually the domain is partitioned into $n_1 \times n_2$ squares of the same size and each square is discretized by $p \times p$ Chebyshev nodes, see Figure 1. We denote all collocation

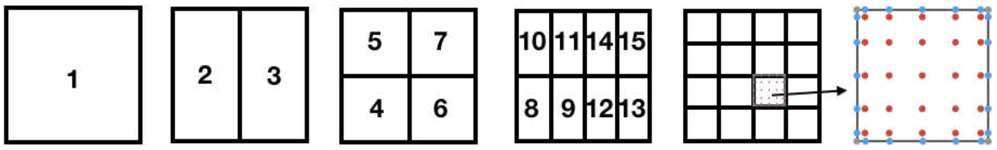


Fig. 1: The domain Ω is hierarchically halved and eventually is partitioned into squares. Then each square is discretized with $p \times p$ Chebyshev nodes.

points by $\mathsf{x} = \{x_i\}_{i=1}^N$ and define the discretized solution $\mathsf{u} = [u(x_i)]_{i=1}^N$ as the solution vector with collated values. The Chebyshev nodes on each square is classified into three groups: the interior nodes (marked in red), the boundary edge nodes (marked in blue) and the corner nodes (marked in gray). It turns out that the gray nodes does not contribute to any spectral derivatives to other nodes when there are no mixed derivatives in (3), so we dropped them off in the collocation points x. After excluding all the corner points, the total number of points in the grid equals

$$N = (p-2)(pn_1n_2 + n_1 + n_2) \approx p^2 n_1 n_2 \,.$$

The discretization procedure above would introduce a sparse $N \times N$ matrix A, and the value of $\mathsf{A}(i,:)\mathsf{u}$ may have different values depending on the type of point x_i

$$\mathsf{A}(i,:)\mathsf{u} \approx \begin{cases} [\mathcal{A}u](x_i), & \text{for any interior points (marked in red)}\,, \\ 0, & \text{for any edge point that is } \textit{not} \text{ on } \partial\Omega\,, \\ \pm\frac{\partial u}{\partial n}, & \text{for any edge point that is on } \partial\Omega\,. \end{cases} \tag{4}$$

In the HPS solver, this big sparse matrix A is not explicitly formed but instead the matrix-vector application of A^{-1} is constructed via a hierarchical sweep through all subdomains. To explain the details, we first restrict our scope to a local subdomain Ω^τ. Denote the interior nodes index in a square Ω^τ as J_i^τ and the boundary edge nodes index as J_b^τ, the operator $\mathcal{A}$ can be locally discretized as a spectral differential matrix and likewise equation ((2)) can be discretized in the following form for any leaf node τ:

$$\begin{aligned} \begin{bmatrix} \mathsf{A}_{i,i}^\tau & \mathsf{A}_{i,b}^\tau \end{bmatrix} \begin{bmatrix} \mathsf{u}_i^\tau \\ \mathsf{u}_b^\tau \end{bmatrix} &= \mathsf{g}_i^\tau\,, \\ \mathsf{u}_b^\tau &= \mathsf{f}_b^\tau\,, \end{aligned} \tag{5}$$

where the subscript denotes the values on the corresponding collocation nodes. For example, $\mathsf{A}_{i,b}^\tau = \mathsf{A}^\tau(J_i^\tau, J_b^\tau)$ and $\mathsf{g}_i^\tau = \mathsf{g}(J_i^\tau)$. The resulting local problem is of small size p and thus it is easy to construct the solution operator S^τ that maps the boundary data to the interior solution

$$\mathsf{u}_i^\tau = \mathsf{S}^\tau \mathsf{u}_b^\tau\,.$$

Additionally, we can build the DtN operator T^τ that maps the boundary data u_b^τ to a vector v^τ consisting of the boundary fluxes

$$\mathsf{v}^\tau = \mathsf{T}^\tau \mathsf{u}_b^\tau \,.$$

More precisely, if the collocation point x_i is on a vertical edge, then $\mathsf{v}^\tau(i) \approx \frac{\partial \mathsf{u}}{\partial x_1}(x_i)$ and if x_i is on a horizontal edge, then $\mathsf{v}^\tau(i) \approx \frac{\partial u}{\partial x_2}(x_i)$. These two operators can be computed directly from the local spectral differentiation matrix. For example, in the absence of source term g_i, the solution operator S^τ and DtN operators are

$$\mathsf{S}^\tau = -\mathsf{A}_{i,i}^{-1}\mathsf{A}_{i,b} \quad \text{and} \quad \mathsf{T}^\tau = \mathsf{D}\mathsf{S}^\tau, \tag{6}$$

where D consists of spectral differentiation operators on edge nodes corresponding to $\frac{\partial}{\partial x_1}$ and $\frac{\partial}{\partial x_2}$ respectively.

The full hierarchical HPS solver consists of two stages: a build stage that sweeps from leaf squares to its parent, and a solve stage that pass through the tree starting from the root to its leaves. In the building stage, the solution operator S^τ and DtN operator T^τ are built for each subdomain τ from leaves to roots. For leaf subdomains, they can be built directly by using equation (6). For a parent subdomain τ with children subdomains α and β, S^τ and T^τ can be built by "merging" the DtN operators T^α and T^β of the children subdomains α and β. These process is done by Schur complements of the linear system (5) with respect to that of its children via the continuity of solution and fluxes on the interface. See more details in Chapter 25 of [9]. For completeness, we briefly discuss below how to merge the local operators on children domains into the local operators on the parent domain. We first partition the boundary points on $\partial\Omega^\alpha$ and $\partial\Omega^\beta$ into the three sets: J_1, the boundary points on $\partial\Omega_\alpha/\partial\Omega_\beta$; J_2, the boundary points on $\partial\Omega_\beta/\partial\Omega_\alpha$; and J_3, the boundary points on both $\partial\Omega_\alpha$ and $\partial\Omega_\beta$ but not on $\partial\Omega_\tau$. See Figure 2 for an illustration of the boundary points. The goal of the merge process is to construct the solution operator S^τ and DtN operator T^τ on the parent node τ from the local operators S^α, T^α and S^β, T^β on its children nodes. These children local operators can be partitioned as the following.

$$\begin{bmatrix} \mathsf{v}_1 \\ \mathsf{v}_3 \end{bmatrix} = \begin{bmatrix} \mathsf{T}^\alpha_{1,1} & \mathsf{T}^\alpha_{1,3} \\ \mathsf{T}^\alpha_{3,1} & \mathsf{T}^\alpha_{3,3} \end{bmatrix} \begin{bmatrix} \mathsf{u}_1 \\ \mathsf{u}_3 \end{bmatrix} , \quad \begin{bmatrix} \mathsf{v}_2 \\ \mathsf{v}_3 \end{bmatrix} = \begin{bmatrix} \mathsf{T}^\alpha_{2,2} & \mathsf{T}^\alpha_{2,3} \\ \mathsf{T}^\alpha_{3,2} & \mathsf{T}^\alpha_{3,3} \end{bmatrix} \begin{bmatrix} \mathsf{u}_2 \\ \mathsf{u}_3 \end{bmatrix} . \tag{7}$$

By definition, the local operators on the parent nodes can also partitioned analogously.

$$\mathsf{u}_3 = \mathsf{S}^\tau \begin{bmatrix} \mathsf{u}_1 \\ \mathsf{u}_2 \end{bmatrix} , \quad \begin{bmatrix} \mathsf{v}_1 \\ \mathsf{v}_2 \end{bmatrix} = \mathsf{T}^\tau \begin{bmatrix} \mathsf{u}_1 \\ \mathsf{u}_2 \end{bmatrix} \tag{8}$$

Combining (7) and (8), we can solve for S^τ and T^τ with T^α and T^β as the following.

$$\mathsf{S}^\tau = \left(\mathsf{T}^\alpha_{3,3} - \mathsf{T}^\beta_{3,3}\right)^{-1} \left[-\mathsf{T}^\alpha_{3,1} \mid \mathsf{T}^\beta_{3,2}\right] , \quad \mathsf{T}^\tau = \begin{bmatrix} \mathsf{T}^\alpha_{1,1} & 0 \\ 0 & \mathsf{T}^\beta_{2,2} \end{bmatrix} + \begin{bmatrix} \mathsf{T}^\alpha_{1,3} \\ \mathsf{T}^\beta_{2,3} \end{bmatrix} \mathsf{S}^\tau \,.$$

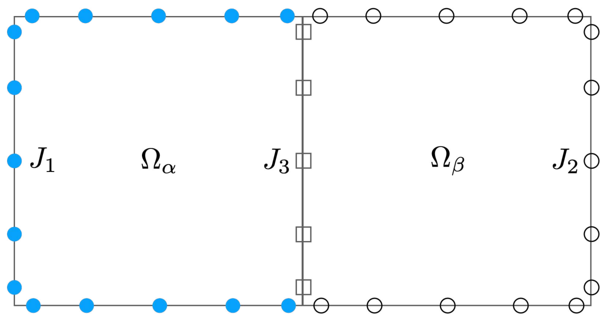


Fig. 2: An illustration of boundary points on the parent domain $\Omega^\tau = \Omega^\alpha \cup \Omega^\beta$.

In the solve stage, one starts from the root domain Ω and iteratively uses the solution operator to map boundary data to the interface of its children subdomains. This process would pass boundary information from parent domains to its children and consequently provide boundary data for all leaf subdomains. Then with the solution operator S^τ for all leaf subdomains, one can easily construct local solutions and glue them together into a global solution. We summarized the solve stage of HPS in the following algorithm 1 when no external force is present.

Algorithm 1 HPS solve stage with no body load.

```
1: $\mathsf{u}(k) = f(x_k)$ for all $k \in J_b^1$.
2: for $\tau = 1, 2, \dots, n$ do
3:     $\mathsf{u}(J_i^\tau) = \mathsf{S}^\tau \mathsf{u}(J_b^\tau)$.
4: end for
```

The time complexity to build all local matrices for leaves in 2D is about $(p^2)^3 = p^6$ because there are p^2 points for each leaf subdomains. As there are about N/p^2 leaf subdomains, the total cost to process all leaf subdomains is about $(p^6)(N/p^2) = p^4N$. For a parent subdomain τ at level l, the cost of merging process is about $2^{-2l}N^{1.5}$. Because there are 2^l subdomains at level l, the total cost at level l is about $2^{-l}N^{1.5}$ and thus the total build cost in 2D is $\sum_{l=0}^{L} 2^{-l}N^{1.5} \approx N^{1.5}$. For the solve stage, the cost of applying the solution operator S^τ is about $2^{-l}N$ so the total cost of the solve stage in 2D is $\sum_{l=0}^{L-1} 2^l 2^{-l} N \approx N \log N$. We numerically verify the solve stage time complexity in Section 5.3.

3 The RKHPS method for time-dependent problems

We now introduce the Runge-Kutta Hierarchical Poincaré-Steklov (RKHPS) schemes for the general parabolic equation (1). These schemes combine high order Runge-Kutta discretization and high order HPS schemes, thus they enjoy high accuracy, stability and efficiency. As mentioned above the HPS scheme is most efficient if the matrix A does not change throughout the computation and it is therefore natural to consider Explictly, Singly Diagonally Implicit Runge-Kutta (ESDIRK) methods.

3.1 Time discretization

In general, the right hand side of (1) can be split into a stiff part $\mathcal{L}u$ and a non-stiff part $q + g(u)$. The natural choice is to adopt an implicit-explicit RK (IMEX-RK) method, in which $\mathcal{L}u$ is discretized by a method defined through an implicit Butcher table A, b and c whereas $g(u)$ is treated by a method defined by an explicit Butcher table $\hat{A}, \hat{b}$ and $\hat{c}$. In particular, we use the ESDIRK scheme for the stiff term $\mathcal{L}u$. These methods usually have an explicit first stage and put the same constant on the diagonal entries a_{ii}. Such structure in the Butcher table A allows to have the same sparse matrix A in (4) in all iterations. Therefore, we can build a HPS solver once in the offline and apply the same HPS solve for all iterations. The nonstiff term is dealt with an Explict Runge-Kutta (ERK) method, which only has non-zero entries in the lower diagonal part in the Butcher table. See Table 1. In particular, we used the tableau data **ARK4(3)6L[2]SA** and **ARK5(4)8L[2]SA** from the additive Runge-Kutta (ARK) methods by Carpenter and Kennedy [15]. It is assumed that the time step coefficients c_j and $\hat{c}_j$ are the same in both ESDIRK and ERK method.

0	0	0		$\dots$	0
c_2	a_{21}	γ	0	$\dots$	0
$\vdots$	$\vdots$	$\ddots$	$\ddots$	$\ddots$	$\vdots$
c_{s-1}	$a_{s-1,1}$	$a_{s-1,2}$	$\dots$	γ	0
1	b_1	b_2	$\dots$	b_{s-1}	γ
	b_1	b_2	$\dots$	b_{s-1}	γ

0	0	0		$\dots$	0
$\hat{c}_2$	$\hat{a}_{21}$	0	0	$\dots$	0
$\vdots$	$\vdots$	$\ddots$	$\ddots$	$\ddots$	$\vdots$
$\hat{c}_{s-1}$	$\hat{a}_{s-1,1}$	$\hat{a}_{s-1,2}$	$\dots$	0	0
1	$\hat{a}_{s,1}$	$\hat{a}_{s,2}$	$\dots$	$\hat{a}_{s,s-1}$	0
	$\hat{b}_1$	$\hat{b}_2$	$\dots$	$\hat{b}_{s-1}$	$\hat{b}_s$

Table 1: Butcher tables of Runge-Kutta methods. **Left:** ESDIRK. **Right**: ERK

The general RK method can be usually formulated in two ways: the stage formulation or the slope formulation. These two formulations are algebraically equivalent for ODE systems but not necessarily equivalent for PDEs. In this manuscript, we implement and compare both formulations. The stage formulation consists of s intermediate stage solves:

$$u_i^n = u^n + \Delta t \sum_{j=1}^{i} a_{ij} \mathcal{L} u_j^n + \Delta t \sum_{j=1}^{i-1} \hat{a}_{ij} \left(q_j^n + g_j^n \right) , \quad i = 1, \dots, s, \tag{9}$$

where $t_j^n = t^n + c_i \Delta t$, $q_j^n = q(t_j^n, x)$ and $g_j^n = g(u_j^n)$. With a ESDIRK method shown in Table 1, we can assume that $a_{ii} = \gamma, i = 2, \dots, s$ and rewrite equation (9) as the following

$$\begin{aligned} u_1^n &= u^n \\ (\mathsf{I} - \Delta t \gamma \mathcal{L}) u_i^n &= u^n + \Delta t \sum_{j=1}^{i-1} a_{ij} \mathcal{L} u_j^n + \Delta t \sum_{j=1}^{i-1} \hat{a}_{ij} \left(q_j^n + g_j^n \right) , \quad i = 2, \dots, s. \end{aligned} \tag{10}$$

It is clear that the first stage is explicit and for other stages one needs to invert an elliptic operator $(\mathsf{I} - \Delta t \gamma \mathcal{L})$. The equations above hold in the interior of Ω and are equipped with boundary conditions

$$u_i^n = f(t_i^n)\,, \quad \text{on } \partial\Omega\,, \quad i = 1, \ldots, s. \tag{11}$$

In comparison, the slope formulations, if there is no nonlinear term $g = 0$, is composed of multiple stages where slope variables k_i^n are calculated as follows

$$\begin{aligned} k_1^n &= \mathcal{L}u^n + q_j^n, \\ (\mathsf{I} - \Delta t\gamma\mathcal{L})k_i^n &= \mathcal{L}u^n + \Delta t \sum_{j=1}^{i-1} a_{ij}\mathcal{L}k_j^n\,, \quad i = 2, \ldots, s. \end{aligned} \tag{12}$$

The semi-discretization above need to be augmented with suitable boundary conditions.

Let E be the diagonal matrix that is 1 at boundary DOF and zero everywhere else. Suppose we want to solve the PDE $v_t = v_{xx} + F^{\mathrm{E}}(v)$, with boundary conditions $v = v_{\mathrm{BC}}(t)$ using a semi-discretization

$$u_t = D_2 u + \tau E(u - v_{\mathrm{BC}}(t)) + F^{\mathrm{E}}(u).$$

Here the boundary conditions are enforced weakly by the penalty term. Denoting $F^{\mathrm{I}}(u) = D_2 u + \tau E(u - v_{\mathrm{BC}}(t))$ we have that

$$F^{\mathrm{I}}(u) = u_t - F^{\mathrm{E}}(u).$$

We consider the first stage in an IMEX method. Given the current solution u_{n-1} it is:

1. Set $K_1^{\mathrm{E}} = F^{\mathrm{E}}(u_{n-1})$.
2. Solve
$$K_1^{\mathrm{I}} = F^{\mathrm{I}}(u_1) \equiv F^{\mathrm{I}}(u_{n-1} + \Delta t A_{1,1}^{\mathrm{I}} K_1^{\mathrm{I}} + \Delta t A_{2,1}^{\mathrm{E}} K_1^{\mathrm{E}}).$$
3. Using the explicit expression for F^{I} we only use the penalty term for K_1^{I} and find
$$K_1^{\mathrm{I}} = D_2 u_1 + \Delta t A_{1,1}^{\mathrm{I}} D_2 K_1^{\mathrm{I}} + \tau E(K_1^{\mathrm{I}} - R) + \Delta t A_{2,1}^{\mathrm{E}} D_2 K_1^{\mathrm{E}}.$$

   Here we can use the PDE, $F^{\mathrm{I}}(u) = u_t - F^{\mathrm{E}}(u)$, to find
$$R = \frac{dv_{\mathrm{BC}}(t)}{dt} - EF^{\mathrm{E}}(u_i).$$

   As we now have additional unknowns on the boundary from $EF^{\mathrm{E}}(u_i)$ we must add the equations
$$E(K_2^E - F^{\mathrm{E}}(u_i)) = 0.$$

A natural choice when $q = 0$ is to set $k_i^n = u_t(t_i^n)$ at the boundary as the slope k_i^n can be interpreted as an approximation to the time derivatives of the solution. At the end, the one step approximation u^{n+1} can be calculated explicitly by assembling the slope variables k_i^n

$$u^{n+1} = u^n + \Delta t \sum_{j=1}^{n} b_j k_j^n .$$

When the nonlinear term g is present, another explicit slope variable l_i^n is introduced. The intermediate slope variables k_i^n and l_i^n are computed as follows:

$$\begin{aligned}
k_1^n &= \mathcal{L}u^n + q_j^n , \\
l_1^n &= g_j^n , \\
(\mathsf{I} - \Delta t\gamma\mathcal{L})k_i^n &= \mathcal{L}u^n + \Delta t \sum_{j=1}^{i-1} a_{ij}\mathcal{L}k_j^n + \Delta t \sum_{j=1}^{i-1} \hat{a}_{ij}\mathcal{L}l_j^n + q_i^n \, , \quad i = 2, \ldots, s, \\
l_i^n &= g(u^n + \Delta t \sum_{j=1}^{i} a_{ij}k_j^n + \Delta t \sum_{j=1}^{i-1} \hat{a}_{ij}l_j^n) \, , \quad i = 2, \ldots, s.
\end{aligned} \tag{13}$$

Similarly, the one step approximation can be assembled as

$$u^{n+1} = u_n + \Delta t \sum_{j=1}^{n} b_j k_j^n + \Delta t \sum_{j=1}^{n} \hat{b}_j l_j^n .$$

However, a major challenge is to design suitable boundary conditions for the intermediate slopes k_i^n and l_i^n. As there are no clear interpretation of these individual slope variables on the boundary, it is only possible to assign suitable boundary conditions for limited situations. For instance, zero boundary conditions can be assigned for both k_i^n and l_i^n if BC of the PDE is time-independent.

3.1.1 Implicit and explicit computation

Notice that in either the stage or slope formulations above, there are two types of equations that need to be solved, i.e. implicit elliptic equations in the form

$$(\mathsf{I} - \Delta t\gamma\mathcal{L})u^\tau = \text{known RHS} \quad \text{on } \Omega^\tau ,$$

or explicit equations in the form

$$u = \text{known RHS} \quad \text{on } \Omega^\tau ,$$

where u denotes the stage variables u_i^n or slope variables k_i^n, l_i^n respectively. Both the implicit and explicit equations are equipped with Dirichlet boundary conditions and hold on all subdomains. As the identity operator can be interpreted as an elliptic operator, both the implicit and explicit equations can be solved on the hierarchical tree by following standard HPS methods. However, for explicit equations alone, one can explicitly form the global linear sparse system and directly solve it without using HPS method. We have tested and compare both implementations in the arXiv report and found that there are only machine precision difference between directly solving sparse system and using HPS method.

3.1.2 Boundary conditions

As the unknowns u_i^n, k_i^n and l_i^n are approximating the solutions and the slope variables respectively, we need to assign boundary conditions in different ways in different formulations correspondingly. For the stage formulation, a natural choice is to assign $u_i^n = u(x, t^n + c_i\Delta t)$ for $x \in \partial\Gamma$. However, such treatment may suffer from order reduction [16] as shown in the numerical results. For the slope formulation, if only k_i^n is present, the natural choice is to assign $k_i^n = u_t(x, t^n + c_i\Delta t)$ for $x \in \partial\Gamma$ because k_i^n is approximating the time derivative of the solutions. In the case when both the implicit slope k_i^n and explicit slope l_i^n are both present, there is no clear relation between them and the solution, thus only limited cases are applicable. For example, if equation (1) is equipped with time-independent BC, then zero boundary Dirichlet conditions can be assigned to the slope variables.

To deal with Neumann and Robin boundary conditions, the HPS method uses the pre-computed DtN operators to map them to Dirichlet boundary conditions. More details can be found in [11].

3.1.3 Penalization in the slope formulation

The HPS method inherently enforces continuity of the fluxes across the interfaces of children subdomains. Such feature, however in the slope formulation, does not guarantee that the solution has a continuous flux across the interface. As a trivial example, if only k_i^n is present, then the updating formula

$$u^{n+1} = u^n + \Delta t \sum_{j=1}^{s} b_j k_j^n \,,$$

will pass any flux mismatch from the previous solution u^n to the next step.

An easy fix of this can be projecting the solution into the continuous flux space. However, for greater generality, we enforce an penalization on flux jump in the slope formulation. That is, the zero flux jump condition in HPS

$$[[Tk + h^k]] = 0 \,,$$

is replaced by a penalized version

$$[[Tk + h^k - \Delta t^{-1} h^u]] = 0 \,,$$

where Tk denotes the derivative from the homogeneous part, h^k denotes the flux of particular slope and h^u denotes the flux of the solution u. This new penalized condition modifies the merging process by adding an extra term.

4 Stability of RKHPS

In this section, we seek to shed light on the stability properties of RKHPS. We establish that the time-stepping map $u^n \to u^{n+1}$ is stable for the particular case of the heat equation discretized with HPS in space, and backwards Euler in time. For higher order discretization, analysis appears to be challenging, but we present numerical evidence that point strongly towards the conclusion that RKHPS is stable up to order five.

4.1 Eigenvalues of the local differentiation matrix

The HPS uses spectral collocation method to discretize each subdomain and the corresponding local differentiation matrix are approximating the second order elliptic operator whose eigenvalues are negative. In particular, the eigenvalues of the continuous second derivative with zero boundary conditions are defined as

$$\begin{aligned} D^2 u(x) &= \lambda u(x)\,, \quad -1 \leq x \leq 1 \\ u(\pm 1) &= 0\,. \end{aligned}$$

The eigenvalues of this continuous problem are known to be $\lambda_k = -\frac{k^2\pi^2}{4}$. The Chebychev collocation methods considers u as $(p-1)$-th order polynomials such that the above equation holds at Chebyshev points $x_j = \cos(\frac{j\pi}{p-1})$. Such discretization yields the spectral differentiation matrix, which approximates the first $\frac{2}{\pi}$ portion of the eigenvalues very well but there is an $\mathcal{O}(p^4)$ error for the remaining eigenvalues. It is shown in [17, 18] that the eigenvalues of spectral differentiation matrix are real, negative and distinct.

4.2 Stability of RKHPS

We now prove the stability of RKHPS for linear heat equation in the following form:

$$\begin{cases} u_t = \Delta u + f\,, & x \in \Omega, \\ u = g\,, & x \in \Gamma, \end{cases} \tag{14}$$

where f is the external force.

Theorem 1. *In dimension two, the eigenvalues of the time-stepping map $\mathsf{M}^n : \mathsf{u}^n \mapsto \mathsf{u}^{n+1}$ of backward Euler HPS method for heat equation (14) has modulus bounded by 1 for any time step size h. In particular, backward Euler is Lax-Richtmyer stable.*

Proof. For stability proof of (14), it suffices to assume $f = g = 0$. The Backward Euler time discretization for a certain Runge-Kutta scheme yields the following semi-continuous PDE:

$$\begin{cases} (\mathsf{I} + h\Delta)u^{n+1} = u^n\,, & x \in \Omega, \\ u^{n+1} = 0\,, & x \in \Gamma. \end{cases} \tag{15}$$

In HPS method of the above semi-continous equation, the domain Ω is decomposed into a hierarchical sequence of subdomains:

$$\begin{cases} (\mathsf{I}+h\Delta)u^{n+1,\tau} = u^n\,, & x \in \Omega^\tau, \\ u^{n+1,\tau} = f^\tau\,, & x \in \Gamma^\tau, \end{cases} \tag{16}$$

where f^τ is the unknown boundary condition over each subdomain. Consider the local problem (16) with fixed τ, we decompose the local solutions into particular solution $w^{n+1,\tau}$ and homogeneous solution $\phi^{n+1,\tau}$,

$$u^{n+1,\tau} = w^{n+1,\tau} + \phi^{n+1,\tau}\,.$$

They satisfy the following equations respectively.

$$\begin{cases} (\mathsf{I}+h\Delta)w^\tau = u^n\,, & x \in \Omega^\tau, \\ w^\tau = 0\,, & x \in \Gamma^\tau, \end{cases} \quad \text{and} \quad \begin{cases} (\mathsf{I}+h\Delta)\phi^\tau = u^n\,, & x \in \Omega^\tau, \\ \phi^\tau = f^\tau\,, & x \in \Gamma^\tau. \end{cases} \tag{17}$$

We omit the supscript $^{n+1}$ in above equations and the proof below when no confusion occurs. In the HPS method, each leaf node τ is discretized with $p \times p$ Chebyshev points. We denote the indices of all Chebyshev points in Ω^τ as J^τ, the interior points as J_i^τ and the boundary points as J_b^τ. Then the operator $(\mathsf{I}+h\Delta)$ is discretized as the 2nd order Chebyshev differentiation matrix L and globally it is discretized as a large sparse matrix A. (see details in section 2.1).

Now consider a parent node τ with children α and β, equation (17) implies that the particular solution on the left children must satisfy that

$$\mathsf{A}(J_i^\alpha, J^\alpha)\mathsf{w}^\alpha = \mathsf{u}^n(J^\alpha)\,.$$

Similarly, equation (16) implies that

$$\mathsf{A}(J_i^\alpha, J^\tau)\mathsf{w}^\tau = \mathsf{u}^n(J^\alpha)\,.$$

Therefore, we must have

$$\mathsf{A}(J_i^\alpha, J^\alpha)\mathsf{w}^\alpha = \mathsf{A}(J_i^\alpha, J^\tau)\mathsf{w}^\tau = \mathsf{A}(J_i^\alpha, J^\alpha)\mathsf{w}^\tau(J^\alpha),$$

where the last equality holds because of the sparsity of matrix A. In fact, the values of $\mathsf{A}(J_i^\alpha, J^\tau)\mathsf{w}^\tau$ depends only on the nodal points within Ω^α rather than that in Ω^τ. Decomposing the index $J^\alpha = [J_i^\alpha, J_b^\alpha]$, we can rewrite the above equation as

$$\mathsf{A}(J_i^\alpha, J_i^\alpha)\mathsf{w}^\alpha(J_i^\alpha) = \mathsf{A}(J_i^\alpha, J_i^\alpha)\mathsf{w}^\tau(J_i^\alpha) + \mathsf{A}(J_i^\alpha, J_b^\alpha)\mathsf{w}^\tau(J_b^\alpha),$$

where we used the fact that $\mathsf{w}^\alpha(J_b^\alpha) = 0$. Invert the square matrix $\mathsf{A}(J_i^\alpha, J_i^\alpha)$, we have

$$\mathsf{w}^\alpha(J_i^\alpha) = \mathsf{w}^\tau(J_i^\alpha) + \left(\mathsf{A}(J_i^\alpha, J_i^\alpha)\right)^{-1}\mathsf{A}(J_i^\alpha, J_b^\alpha)\mathsf{w}^\tau(J_b^\alpha).$$

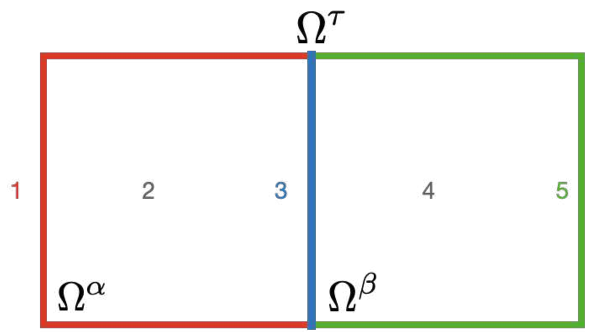


Fig. 3: The vector J^τ is partitioned into five blocks.

Analogously for child β, we have

$$\mathsf{w}^\beta(J_i^\beta) = \mathsf{w}^\tau(J_i^\beta) + \left(\mathsf{A}(J_i^\beta, J_i^\beta)\right)^{-1} \mathsf{A}(J_i^\beta, J_b^\beta)\mathsf{w}^\tau(J_b^\beta).$$

To illustrate the structure of the above two equations, we partition the indices J^τ into five blocks (see an illustration in Figure 3)

$$\begin{bmatrix} \mathsf{w}^\alpha(J_i^\alpha) \\ \mathsf{w}^\beta(J_i^\beta) \end{bmatrix} = \begin{bmatrix} \mathsf{T}_{11} & \mathbf{I} & \mathsf{T}_{13} & 0 & 0 \\ 0 & 0 & \mathsf{T}_{23} & \mathbf{I} & \mathsf{T}_{25} \end{bmatrix} \mathsf{w}^\tau := \mathsf{T}\mathsf{w}^\tau\,, \tag{18}$$

where $\mathsf{T}_{11}, \mathsf{T}_{13}$ are submatrices of $(\mathsf{A}(J_i^\alpha, J_i^\alpha))^{-1}\mathsf{A}(J_i^\alpha, J_b^\alpha)$ and analogously T_{23}, T_{25} are submatrices of $\left(\mathsf{A}(J_i^\beta, J_i^\beta)\right)^{-1} \mathsf{A}(J_i^\beta, J_b^\beta)$. Such structure of matrix T guarantees that itself has singular values bounded below by 1. In fact, we have

$$\mathsf{T}\mathsf{T}^\top - \mathsf{I} = \begin{bmatrix} \mathsf{T}_{11}\mathsf{T}_{11}^\top + \mathsf{T}_{13}\mathsf{T}_{13}^\top & \mathsf{T}_{13}\mathsf{T}_{23}^\top \\ \mathsf{T}_{23}\mathsf{T}_{13}^\top & \mathsf{T}_{23}\mathsf{T}_{23}^\top + \mathsf{T}_{25}\mathsf{T}_{25}^\top \end{bmatrix} = \begin{bmatrix} \mathsf{T}_{11}\mathsf{T}_{11}^\top & \\ & \mathsf{T}_{25}\mathsf{T}_{25}^\top \end{bmatrix} + \begin{bmatrix} \mathsf{T}_{13} \\ \mathsf{T}_{23} \end{bmatrix} \begin{bmatrix} \mathsf{T}_{13}^\top & \mathsf{T}_{23}^\top \end{bmatrix},$$

which is the sum of two semi-positive definite matrices. Therefore (18) implies that for any parent node τ with children α and β:

$$\|\mathsf{w}^\alpha\|^2 + \|\mathsf{w}^\beta\|^2 = \|\mathsf{w}^\alpha(J_i^\alpha)\|^2 + \|\mathsf{w}^\beta(J_i^\beta)\|^2 \geq \|\mathsf{w}^\tau\|^2.$$

Apply the above inequality hierarchically for all parent nodes τ, we have

$$\|\mathsf{u}^1\|^2 = \|\mathsf{w}^1\|^2 \leq \|\mathsf{w}^2\|^2 + \|\mathsf{w}^3\|^2 \leq \cdots \leq \sum_{\tau \text{ is leaf}} \|\mathsf{w}^\tau\|^2, \tag{19}$$

where the summation in the last inequality is over all leaf nodes τ. For any leaf τ, the Chebyshev discreization of equation (17) implies that

$$\mathsf{w}^\tau(J_i^\tau) = (1 + h\mathsf{L}(J_i^\tau, J_i^\tau))^{-1}\mathsf{u}^n(J_i^\tau)\,, \tag{20}$$

where L is the 2D second order Chebyshev differentiation matrix. It is shown in [17] that the 1D Chebyshev differentiation matrix E has real, distinct and negative eigenvalues $\sigma_{E,i}$. Notice that $\mathsf{L}(J_i^\tau, J_i^\tau) = \mathsf{E} \otimes \mathsf{I} + \mathsf{I} \otimes \mathsf{E}$, we conclude that $\mathsf{L}(J_i^\tau, J_i^\tau)$ is also diagonalizable. In fact, assuming the eigen-decomposition $\mathsf{E} = \mathsf{V}_E \Sigma_E \mathsf{V}_E^{-1}$, we must have

$$\begin{aligned}(\mathsf{V}_\mathsf{E}^{-1} \otimes \mathsf{V}_\mathsf{E}^{-1})\mathsf{L}(J_i^\tau, J_i^\tau)(\mathsf{V}_\mathsf{E} \otimes \mathsf{V}_\mathsf{E}) &= (\mathsf{V}_\mathsf{E}^{-1} \otimes \mathsf{V}_\mathsf{E}^{-1})(\mathsf{E} \otimes \mathsf{I} + \mathsf{I} \otimes \mathsf{E})(\mathsf{V}_\mathsf{E} \otimes \mathsf{V}_\mathsf{E}) \\ &= (\mathsf{V}_\mathsf{E}^{-1} \otimes \mathsf{V}_\mathsf{E}^{-1})\left((\mathsf{V}_\mathsf{E}\Sigma_E\mathsf{V}_\mathsf{E}^{-1}) \otimes \mathsf{I} + \mathsf{I} \otimes (\mathsf{V}_\mathsf{E}\Sigma_E\mathsf{V}_\mathsf{E}^{-1})\right)(\mathsf{V}_\mathsf{E} \otimes \mathsf{V}_\mathsf{E}) \\ &= \Sigma_E \otimes \mathsf{I} + \mathsf{I} \otimes \Sigma_E .\end{aligned} \tag{21}$$

This implies the eigenvalues of $\mathsf{L}(J_i^\tau, J_i^\tau)$ are pairwise sum $\sigma_{E,i} + \sigma_{E,j} < 0$ and the eigenvectors are pairwise Kronecker product $\mathsf{V}_{\mathsf{E},i} \otimes \mathsf{V}_{\mathsf{E},j}$.

Consequently, there exists the eigen-decomposition of $\mathsf{L}(J_i^\tau, J_i^\tau)$ for any leaf τ,

$$\mathsf{L}(J_i^\tau, J_i^\tau) = \mathsf{V}_i^\tau \Sigma_i^\tau (\mathsf{V}_i^\tau)^{-1} ,$$

where Σ_i^τ is a diagonal matrix with negative entries and V_i^τ contains the eigenvectors. Plug it into equation (20), we have

$$\mathsf{w}^\tau(J_i^\tau) = \mathsf{V}_i^\tau (1 + h\Sigma_i^\tau)^{-1} (\mathsf{V}_i^\tau)^{-1} \mathsf{u}^n(J_i^\tau) . \tag{22}$$

Because Σ_i^τ are negative, the L-stability of Euler methods implies that the entries of $(1 + h\Sigma_i^\tau)^{-1}$ must have modulus smaller than 1, therefore

$$\|\mathsf{w}^\tau\|^2 = \|\mathsf{w}^\tau(J_i^\tau)\|^2 \leq \|\mathsf{u}^n(J_i^\tau)\|^2 .$$

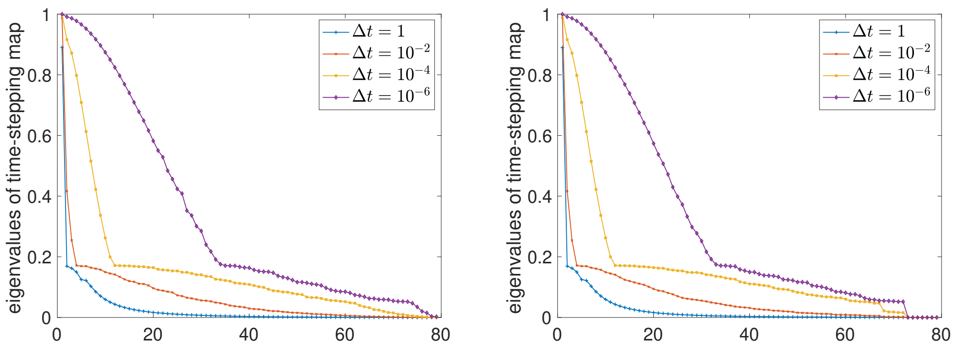
Now combine the above equation with (19), we have

$$\|\mathsf{u}^{n+1}\|^2 = \|\mathsf{u}^1\|^2 \leq \sum_{\tau \text{ is leaf}} \|\mathsf{u}^n(J_i^\tau)\|^2 \leq \|\mathsf{u}^n\|^2,$$

which implies $\|\mathsf{M}^n\| \leq 1$ and thus the stability of RKHPS. □

Remark 1. For general parabolic equations, unfortunately there is no guarantee that spectral approximation to the elliptic operator has real negative eigenvalues. For example, in the strong convection regime or high frequency regime of Helmholtz type operator, the spectral approximation may have imaginary or real positive eigenvalues and in those cases there is no guarantee for stability of RKHPS. Numerical result shows that the RKHPS method is stable for partial differential equations in which the elliptic operator is dominating though.

In Figure 4a and 4b we have plotted the eigenvalues of the time-stepping map for 1D variable coefficient parabolic equation (25) without or with a HPS hierarchical tree structure. We see that in both cases the RKHPS has eigenvalues bounded by 1, regardless of the time-step size $\Delta t = 1, 10^{-2}, 10^{-4}, 10^{-6}$. Moreover, for the RKHPS with hierarchical tree, we set the depth of tree $L = 3$ and consequently there are 8 leaf nodes. By comparing Figure 4a and 4b, we see that the tree structure introduces 7 zero eigenvalues. As demonstrated in the proof stability, such phenomena stems from the continuity flux assumptions in the HPS methods and the number of zero eigenvalues equals to the total number of interfaces. In fact, it can shown that the null space of the time-stepping map consists of functions that are supported on the interfaces.



(a) with no hierarchical tree structure (b) with 8 leaf nodes in the hierarchical tree

Fig. 4: Eigenvalues of the time-stepping map for 1D variable coefficient parabolic equation

5 Numerical tests

5.1 1D convection diffusion equation

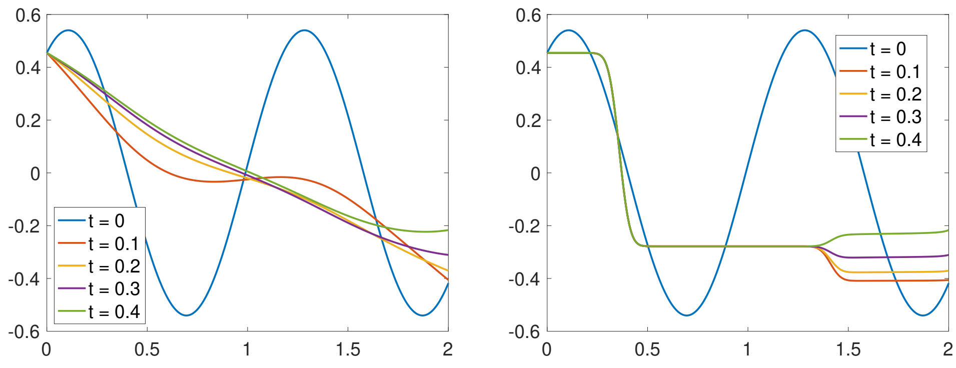
We then test the convection-diffusion equation with variable convection-coefficient.

$$u_t = u_{xx} - k\sin(1 + 1.9\pi x)u_x + q \tag{23}$$

with initial and boundary conditions

$$\begin{aligned} u(x,0) &= \sin(1 + 1.7\pi x)\cos(1)\,, && \text{for all } x \in [0,2] \\ u(x,t) &= \sin(1 + 1.7\pi x)\cos(1 + t^2 x)(1 + t^3 x)\,, && \text{for all } t \in [0,0.5] \text{ and } x = 0 \text{ or } 2 \end{aligned} \tag{24}$$

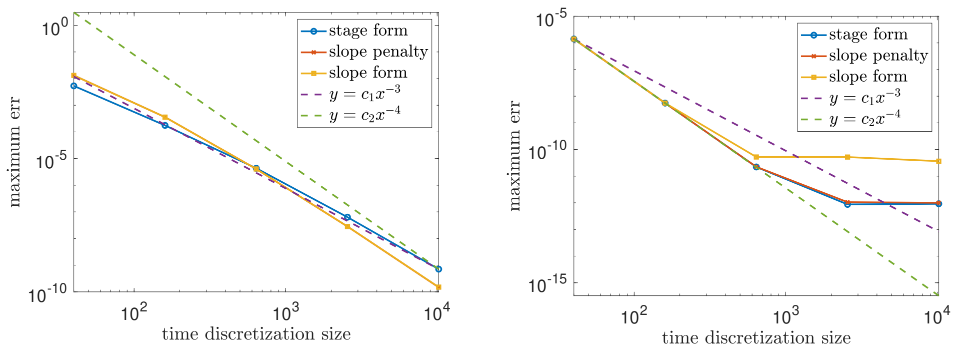
We first consider the case with no external source $q = 0$. We discretize the domain with 32 leaves and $p = 21$ on each leaf node. The approximate solution is computed using a ARK4(3)6L[2]SA-ESDIRK method in [15]. In Figure 5a and 5b we plot time stamps of approximate solution with $k = 1$ and $k = 100$. In the strong diffusion regime $k = 1$, the solution profiles 5a are similar to that of the heat equation. In comparison, in the strong convection regime $k = 100$, the solution in Figure 5b quickly forms shocks at point $x = 0.3588$ and $x = 1.4114$. These two points are exactly where the convection coefficient $\sin(1+1.9\pi x)$ changes from positive to negative. We first set $k = 1$ and test the diffusion dominated case. We plot the case of inhomogeneous and homogeneous BC in Figure 6a and 6b. The plots are again similar to that of the heat equation upto



(a) strong diffusion regime (b) strong convection regime

Fig. 5: time stamps of convection diffusion equation

some minor difference. Then we set $k = 100$ and plot in Figure 7a and 7b. In this case, the order of convergence drops to 3rd order regardless of different formulations or type of boundary conditions.



(a) inhomogeneous BC (b) homogeneous BC

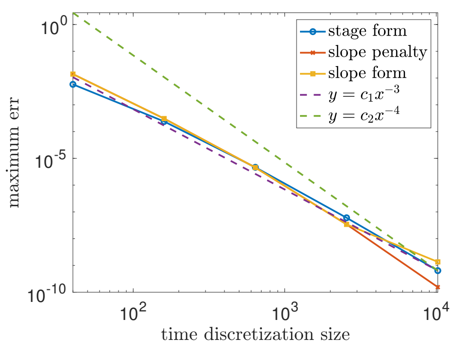
Fig. 6: convergence test of convection diffusion equation with $k = 1$

5.2 1D variable coefficient parabolic equation

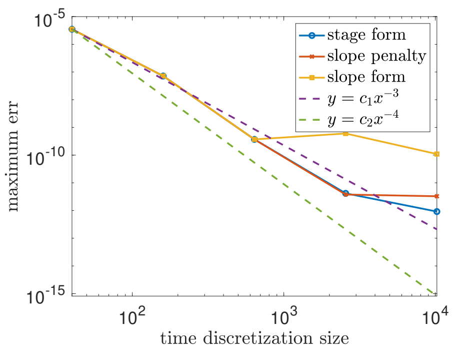
We test variable coefficient parabolic equation in this section with same discretization and ESDIRK method as the previous subsection.

$$u_t = \partial_x(a\partial_x u) + \kappa^2 u + q \tag{25}$$

where $a(x) = 1 + 0.9\sin(1 + 1.9\pi x)$ is the inhomogeneous medium conductivity. We set $\kappa = 1$ so the equation is diffusion dominated. We first calculate a typical solution



(a) inhomogeneous BC



(b) homogeneous BC

Fig. 7: convergence test of convection diffusion equation with $k = 100$

with no external source $q = 0$ and with initial and boundary conditions set as in (24). The time stamps of the approximate solution is plotted in Figure 8. One can see that the solution quickly changes from the sine profile to a monotone temperature diffusion profile. Also notice that the solution is nearly a constant on regions where the medium conductivity achieves large values.

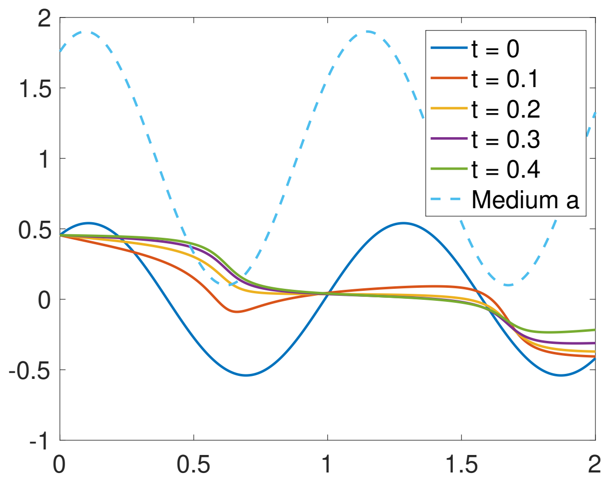
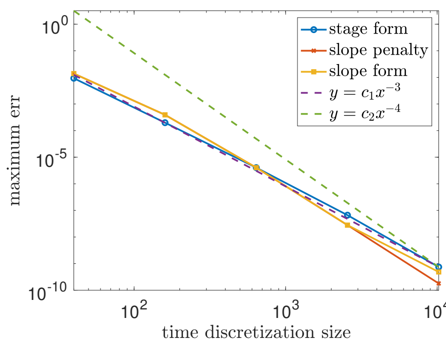
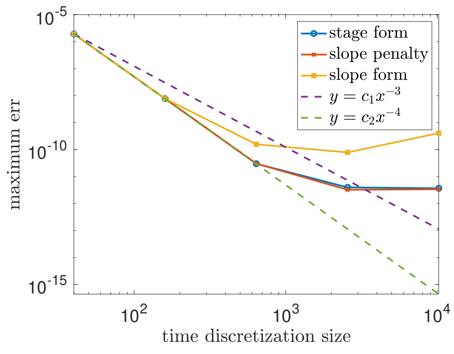


Fig. 8: time stamps of approximate solution from $t = 0$ to $t = 0.5$.

In Figure 9a and 9b we plot the convergence rate for inhomogeneous BC and homogeneous BC cases respectively. The plots more or less resembles that of the heat equation. For inhomogeneous case, we obtain asymptotically 3rd order convergence for stage formulation and nearly 4th order for slope formulations. For homogeneous case, all three methods obtains 4th order convergence before it gets saturated at magnitude of 10^{-10} to 10^{-12}.



(a) inhomogeneous BC



(b) homogeneous BC

Fig. 9: convergence test of variable coefficient parabolic equation

5.3 2D heat equation

We consider the 2D heat equation in this section

$$u_t = u_{xx} + u_{yy} + q\,, \tag{26}$$

where a suitable source q, initial and boundary conditions that are compatible with the exact solution

$$u(t,x,y) = \sin(\pi x)\exp(-t(y-\frac{1}{2})^2)\,,$$

in the inhomogeneous boundary condition case, or

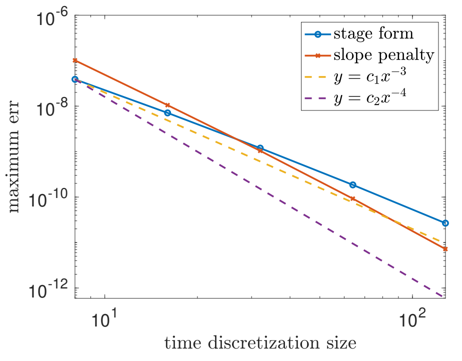
$$u(t,x,y) = \sin(2\pi x)\sin(2\pi y)\exp(-t(x+y))\,,$$

in the homogeneous boudnary condition case.

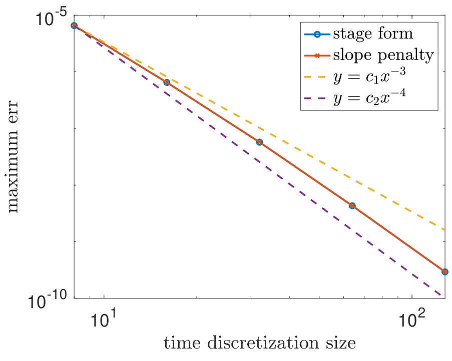
The domain $\Omega = [0,1]^2$ is divided into 8×8 nodes with $p = 21$. The approximate solution is computed using the ARK4(3)6L[2]SA-ESDIRK method in [15]. For both the stage and slope formulation, we plot the maximum error with different time discretizations in Figure 10a and 10b. In the case with inhomogeneous BC, the (penalized) slope formulation has minor order reduction while the stage formulation lost about one order of accuracy. In comparison, in the homogeneous BC cases, both formulations have same order of accuracy.

$O(N\log(N))$ time complexity

We plotted the CPU time of each time step for solving the heat equation in Figure 11. In particular, the computational domain is discretized with 4×4, 8×8,..., 64×64 subdomains, each with 11×11 Chebyshev nodes. The total number of degrees of freedom ranges from 1681 to 410,881. We then compare the proposed method with the finite element method (FEM). For the FEM method, the space is discretized with the same number of degree of freedom with no hierarchical structure. For simplicity, we used the backward-Euler method for the time discretization. The FEM solution



(a) inhomogeneous BC



(b) homogeneous BC

Fig. 10: convergence test of heat equation

is obtained using LAPACK DGB-FA and DGB-SL with a PLU decomposition to take advantage of the banded structure of the stiffness matrix. We note that there are more advanced ways to solve the linear system associated with high order FEM discretizations. A recent paper discussing such techniques is [19]. In Figure 11, it is shown that the solve stage time of the proposed method scales as $O(N \log(N))$ while that of FEM is slightly better than $O(N^2)$.

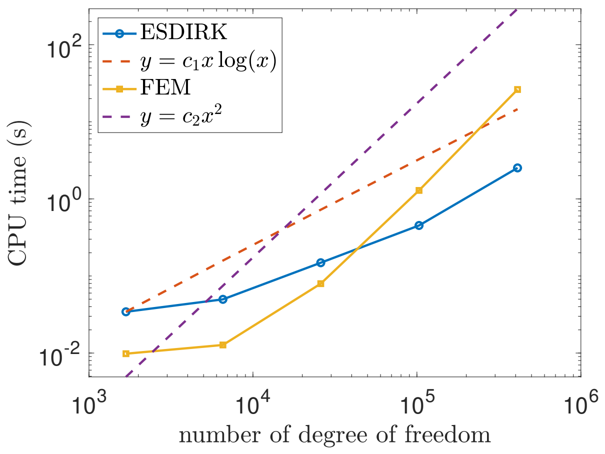


Fig. 11: CPU time comparison between our method and FEM for solving one time step for the heat equation

5.4 2D Burgers equation

In this section, we consider the following 2D Burgers equations:

$$\begin{aligned} u_t &= \varepsilon(u_{xx} + u_{yy}) - (uu_x + vu_y)\,, \\ v_t &= \varepsilon(u_{xx} + u_{yy}) - (uv_x + vv_y)\,. \end{aligned} \tag{27}$$

We use the same discretization in this example and ARK4(3)6L[2]SA-ESDIRK for the Analytic solution test. A higher order method ARK5(4)8L[2]SA-ESDIRK ([15]) is used for the rotating flow test to compute the approximate soluitons. We investigate the convergence rate for different typical solution to the Burgers equation and show that RKHPS has high order convergence and stability, though loss of order convergence occurs for inhomogeneous boundary condition problem. A rotating flow example is also provided to demonstrate that RKHPS can capture a sharp transition region in the fluid.

5.4.1 Analytic solution test

The Burgers equations are solved from $t = 0$ and $T = 2$ on a unit square $[0, 1]^2$. We partition the domain $[0, 1]^2$ into $8 \times 8 = 64$ smaller squares and each of them is further discretized with order $p = 21$ Chebyshev nodes. In total there are 25921 space grid points. The maximum error $\|u(T, \cdot) - u_{\text{exact}}(T, \cdot)\|_\infty$ are calculated on all grid points except those corner points. The viscosity is set as $\varepsilon = 0.1$. In this test, the slope formulation is not applicable and only stage formulation is used here. We study the convergence rate for different type of exact solutions at the terminal time $T = 2$.

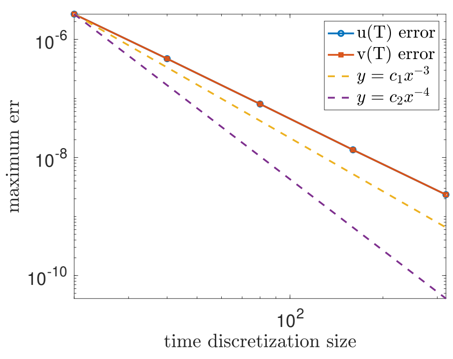
Traveling wave solution The exact solution is of the following form:

$$\begin{aligned} u_{\text{travel}}(x, y, t) &= \frac{3}{4} - \frac{1}{4\left(1 + \frac{\exp(4y-4x-t)}{32\varepsilon}\right)}\,, \\ v_{\text{travel}}(x, y, t) &= \frac{3}{4} + \frac{1}{4\left(1 + \frac{\exp(4y-4x-t)}{32\varepsilon}\right)}\,. \end{aligned} \tag{28}$$

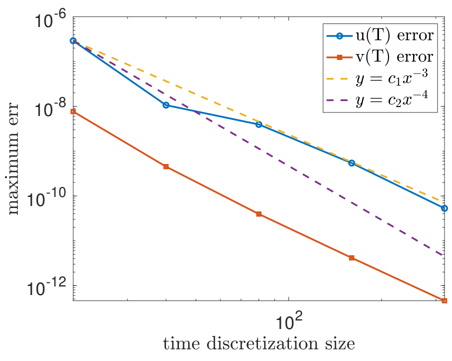
In Figure 12a, we plot the maximum error for both u and v, the convergence rate is near 3rd order due to the order reduction in inhomogeneous BC solutions.
Highly diffusive solution In this case, we consider the following exact solution:

$$\begin{aligned} u_{\text{diffusive}}(x, y, t) &= -\frac{4\pi\varepsilon \exp(-5\pi^2\varepsilon t)\cos(2\pi x)\sin(\pi y)}{2 + \exp(-5\pi^2\varepsilon t)\sin(2\pi x)\sin(\pi y)}\,, \\ v_{\text{diffusive}}(x, y, t) &= -\frac{2\pi\varepsilon \exp(-5\pi^2\varepsilon t)\sin(2\pi x)\cos(\pi y)}{2 + \exp(-5\pi^2\varepsilon t)\sin(2\pi x)\sin(\pi y)}\,. \end{aligned} \tag{29}$$

In Figure 12b, we plot the maximum error for u and v. The convergence rate of both are close to 4th order and gradually drop to 3rd order as the time discretization get finer.



(a) traveling wave solution



(b) highly diffusive solution

Fig. 12: convergence test of Burgers equation

5.4.2 Rotating flow test

In this test, we consider a rotating flow problem on the domain $[-\pi, \pi]^2$ with terminal time $T = 1.5$. The viscosity is set to be low $\varepsilon = 0.005$ and we set initial condition as

$$u = -5y \exp(-3(x^2 + y^2)) \,,$$
$$v = 5x \exp(-3(x^2 + y^2)) \,.$$

and set no-slip boundary conditions. The domain is discretized with 24×24 leaf nodes with $p = 24$ and the time is discretized with $\tau = 0.01$. To capture the shock like transition region, we used a 5th order ESDIRK method. In Figure 13, we plot the contour of velocity $[u, v]^\top$ at time $t = 0.01, 0.51$ and 1.01. We can see that two semicircle fluid are rotating and gradually expanding to a larger circle, eventually they form a shock near the circle.

5.5 Multiple filamentation

As a final example we simulate the regularized non-linear Schrödinger equation [20]

$$iu_t + (1 - ia\varepsilon)\Delta u + (1 + ic\varepsilon)|u|^2 u = ib\varepsilon u. \tag{30}$$

Here the terms with a, b and c corresponds to, linear wave-number dependent absorption, linear gain, and two-photon absorption, respectively. This is a challenging equation to simulate as its solution approximates collapse of filaments, a process which requires accurate numerics in both space and time. When $\varepsilon = 0$ and the optical power is beyond a critical threshold localized filementation occurs. Each such collapse is well approximated by a self-similar radially symmetric solution

$$|u(r, t)| \approx R_0(\rho)/L(t),$$

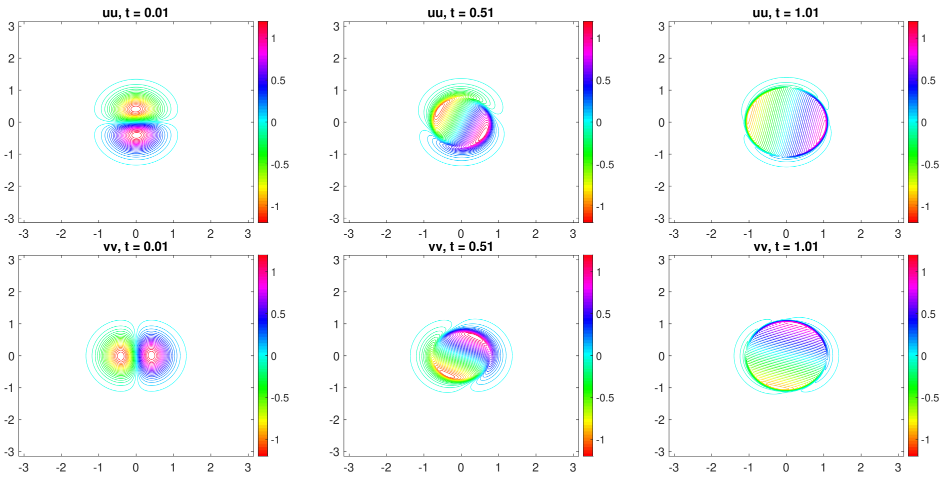


Fig. 13: rotating flow with low viscosity

with $\rho = r/L(t)$ and

$$L(t) \approx \sqrt{(2\pi)}\frac{\sqrt{t_0 - t}}{\sqrt{\ln|\ln(t_0 - t)|}},$$

where t_0 is the time of collapse.

We solve this problem on the domain $\Omega = [0, 25.6]^2$ from $t = 0$ to $t = 5s$. The domain is divided into 8×8 leaf nodes with $p = 13$, and timestep $\Delta t = 0.005$. The problem is equipped homogeneous Dirichlet boundary condition and with initial data

$$u = 5\exp\left(-(x - 12.8)^2 - (y - 12.8)^2\right).$$

We plot the contour of the modulus of the solution at different time stamps $t = 0, 1.66, 3.325$ and $t = 4.995$ in Figure 14. We see that the solution has sharp derivatives and our method qualitatively captures the collapse behavior of filaments of the solution.

Acknowledgements

This work was supported in part by NSF Grants DMS-2208164, DMS-2210286 (DA) and DMS-1952735, DMS-2012606 (PGM). The work of PGM was also supported by the Office of Naval Research (N00014-18-1-2354) and by the Department of Energy ASCR (DE-SC0022251).

Conflict of Interest

On behalf of all authors, the corresponding author states that there is no conflict of interest. All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject

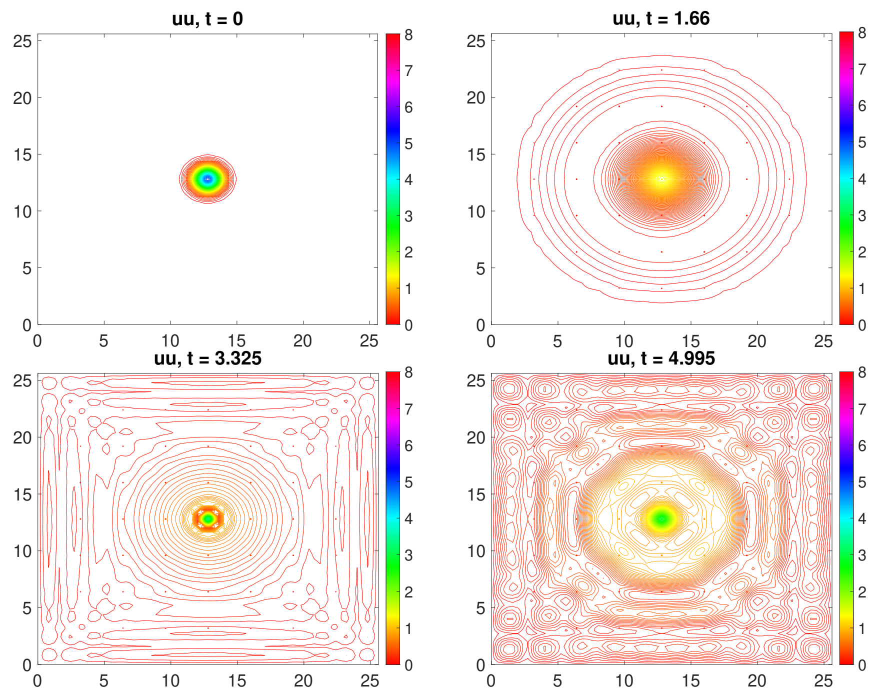


Fig. 14: solution stamps of the regularized non-linear Schrödinger equation.

matter or materials discussed in this manuscript. The authors have no financial or proprietary interests in any material discussed in this article.

Ethical Approval

N/A

Informed Consent

N/A

References

[1] Davis, T.A.: Direct Methods for Sparse Linear Systems. Society for Industrial and Applied Mathematics, Philadelphia (2006)

[2] George, A.: Nested dissection of a regular finite element mesh. SIAM Journal on Numerical Analysis **10**(2), 345–363 (1973)

[3] Martinsson, P.-G.: A direct solver for variable coefficient elliptic pdes discretized via a composite spectral collocation method. Journal of Computational Physics **242**, 460–479 (2013)

[4] Gillman, A., Barnett, A.H., Martinsson, P.-G.: A spectrally accurate direct solution technique for frequency-domain scattering problems with variable media. BIT Numerical Mathematics **55**(1), 141–170 (2015)

[5] Gillman, A., Martinsson, P.-G.: A direct solver with o(n) complexity for variable coefficient elliptic pdes discretized via a high-order composite spectral collocation method. SIAM Journal on Scientific Computing **36**(4), 2023–2046 (2014)

[6] Hao, S., Martinsson, P.-G.: A direct solver for elliptic pdes in three dimensions based on hierarchical merging of Poincaré–Steklov operators. Journal of Computational and Applied Mathematics **308**, 419–434 (2016)

[7] Martinsson, P.-G.: A direct solver for variable coefficient elliptic pdes discretized via a composite spectral collocation method. Journal of Computational Physics **242**, 460–479 (2013) https://doi.org/10.1016/j.jcp.2013.02.019

[8] Gillman, A., Martinsson, P.-G.: A direct solver with O(N) complexity for variable coefficient elliptic pdes discretized via a high-order composite spectral collocation method. SIAM Journal on Scientific Computing **36**(4), 2023–2046 (2014) https://doi.org/10.1137/130918988

[9] Martinsson, P.-G.: Fast Direct Solvers for Elliptic PDEs. Society for Industrial and Applied Mathematics, Philadelphia (2019)

[10] Babb, T., Martinsson, P.-G., Appelo, D.: HPS accelerated spectral solvers for time dependent problems. arXiv preprint arXiv:1811.04555, 155 (2018)

[11] Babb, T., Gillman, A., Hao, S., Martinsson, P.-G.: An accelerated poisson solver based on multidomain spectral discretization. BIT Numerical Mathematics **58**(4), 851–879 (2018)

[12] Gottlieb, S., Ketcheson, D.I., Shu, C.-W.: Strong Stability Preserving Runge-Kutta and Multistep Time Discretizations. World Scientific, Singapore (2011)

[13] Hu, J., Shu, R., Zhang, X.: Asymptotic-preserving and positivity-preserving implicit-explicit schemes for the stiff BGK equation. SIAM Journal on Numerical Analysis **56**(2), 942–973 (2018)

[14] Martinsson, P.-G.: A composite spectral scheme for variable coefficient Helmholtz problems. arXiv preprint arXiv: 1206.4136 (2012)

[15] Kennedy, C.A., Carpenter, M.H.: Additive Runge–Kutta schemes for convection–diffusion–reaction equations. Applied numerical mathematics **44**(1-2), 139–181 (2003)

[16] Rosales, R., Seibold, B., Shirokoff, D., Zhou, D.: Order reduction in high-order Runge–Kutta methods for initial boundary value problems. arXiv preprint

arXiv:1712.00897 (2017)

[17] Gottlieb, D., Lustman, L.: The spectrum of the chebyshev collocation operator for the heat equation. SIAM Journal on Numerical Analysis **20**(5), 909–921 (1983) https://doi.org/10.1137/0720063

[18] Weideman, J., Trefethen, L.N.: The eigenvalues of second-order spectral differentiation matrices. SIAM Journal on Numerical Analysis **25**(6), 1279–1298 (1988)

[19] Pazner, W., Kolev, T., Dohrmann, C.: Low-order preconditioning for the high-order finite element de rham complex. SIAM Journal on Scientific Computing **45**(2), 675–702 (2023)

[20] Lushnikov, P.M., Vladimirova, N.: Non-gaussian statistics of multiple filamentation. Opt. Lett. **35**(12), 1965–1967 (2010)