



Article

A Boundary-Adapted Legendre–Galerkin Method for Nonlinear Caputo Reaction–Diffusion Equations with Non-Local Integral Boundary Conditions

Weaam Alhejaili ¹, Kawthar Alsa'di ² and Álvaro H. Salas ^{3,*} 

¹ Department of Mathematical Sciences, College of Science, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia; walhejaili@pnu.edu.sa

² Independent Researcher, Amman 11118, Jordan; kawther.alsaadi@yahoo.com

³ FIZMAKO Research Group, Department of Mathematics and Statistics, Universidad Nacional de Colombia, Manizales 170004, Colombia

* Correspondence: ahsalass@unal.edu.co

Abstract

This paper studies nonlinear time-fractional reaction–diffusion equations with Caputo memory and non-local integral boundary conditions on a bounded interval. The aim is to formulate a boundary-compatible well-posedness framework and to construct a high-order temporal approximation that can be coupled with a constraint-preserving spectral spatial discretization. The analytical part proves boundedness of the non-local boundary functionals, states compatibility assumptions, and introduces a finite-dimensional nondegeneracy condition for an explicit polynomial lifting. Under a sectorial non-local elliptic realization and a global Lipschitz reaction term, existence, uniqueness, stability, and continuous dependence of mild solutions are obtained by fractional resolvent estimates and fractional Gronwall inequalities. The main novelty is the combined construction of an explicit polynomial lifting for integral boundary constraints, a constraint-preserving Legendre–Galerkin basis, and a high-order Beta-window temporal quadrature together with a discrete stability condition that accounts for sign-changing weights. The numerical evidence shows high-order behavior for smooth Caputo benchmarks, accurate enforcement of the non-local boundary constraints, and improved accuracy over the classical L_1 approximation in the reported tests. The stability discussion identifies the discrete coercivity condition required for the sign-changing Beta-window weights.

Keywords: Caputo fractional derivative; reaction–diffusion equation; non-local integral boundary conditions; mild solution; polynomial lifting; Beta-window approximation; Legendre–Galerkin method; fractional memory; stability; L_1 scheme



Academic Editors: António Lopes and Ricardo Almeida

Received: 5 May 2026

Revised: 11 June 2026

Accepted: 16 June 2026

Published: 25 June 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

MSC: 35R11; 35K57; 65M12; 65M70; 65R20

1. Introduction

Time-fractional reaction–diffusion equations model processes in which diffusion and reaction are affected by memory. In the Caputo formulation, the history dependence is represented by a weakly singular convolution kernel, whereas the initial condition retains its classical meaning. This combination is useful in anomalous diffusion, viscoelastic response, heterogeneous media, biological transport, and chemical reaction–diffusion models [1–5].

The present paper considers the nonlinear model

$${}_0^C D_t^\alpha u(x, t) = \kappa u_{xx}(x, t) + f(x, t, u(x, t)), \quad 0 < \alpha < 1, \quad (1)$$

on a bounded interval, supplemented with an initial condition and two non-local integral boundary constraints. These constraints couple endpoint values with weighted spatial averages of the unknown. Thus, the problem has two non-local mechanisms: memory in time and integral coupling in space.

Several numerical approaches have been developed for differential equations with non-local boundary conditions. For example, local meshless RBF techniques have recently been used for elliptic PDEs with multipoint boundary conditions, where the influence of the RBF shape and conditioning is central [6]. Haar-wavelet collocation methods have also been developed for parabolic problems with integral boundary conditions [7,8]. In fractional diffusion, standard $L1$, $L2$, $L2-1_\sigma$, convolution-quadrature, and spectral-in-time methods provide well-established high-order approximations of Caputo-type derivatives [9–13].

The novelty of this work is now stated more clearly: we do not claim that fractional memory, nonlinear reaction–diffusion equations, or non-local boundary conditions are new separately. The contribution is the specific combination of three components: an explicit two-mode polynomial lifting for integral endpoint constraints, a Legendre–Galerkin basis adapted to the resulting homogeneous non-local space, and a Beta-window Caputo quadrature whose stability is discussed through a coercive implicit Galerkin formulation. The Beta-window formula differs from the classical $L1$ method because it integrates the singular Caputo kernel exactly against a local polynomial reconstruction of the integer derivative. It is also distinct from standard $L2$ -type formulas in the way the history windows are assembled and in its coupling with the non-local boundary-adapted basis.

The main aims of the work are now stated explicitly:

1. To formulate a well-posed Caputo reaction–diffusion problem with integral boundary constraints in a boundary-compatible functional setting;
2. To provide an explicit lifting that converts nonhomogeneous integral boundary data into homogeneous non-local constraints;
3. To construct a high-order Beta-window approximation of the Caputo derivative and couple it with a boundary-adapted Legendre–Galerkin discretization;
4. To clarify the stability regime of the fully discrete method, especially the effect of sign-changing high-order memory weights;
5. To test the temporal approximation, the boundary lifting, and the complete discretization on manufactured and non-polynomial benchmarks.

The scope of this paper studies the classical weakly singular Caputo kernel. It does not propose a new fractional operator, nor does it treat generalized non-singular-kernel memory laws. Such operators are important for other types of memory response, but they require different analytical estimates and different quadrature structures. The present method is therefore best viewed as a high-order discretization and boundary-treatment framework for Caputo-type reaction–diffusion models.

The analytical part begins with the non-local boundary functionals

$$\mathcal{B}_1 v = v(a) + \int_a^b \gamma_1(x) v(x) dx, \quad \mathcal{B}_2 v = v(b) + \int_a^b \gamma_2(x) v(x) dx.$$

We prove that these functionals are bounded on $H^1(a, b)$, impose the natural compatibility condition at $t = 0$, and introduce a 2×2 boundary matrix associated with the modes $p_0(x) = 1$ and $p_1(x) = (2x - a - b)/(b - a)$. The invertibility of this matrix gives an explicit lifting of the nonhomogeneous boundary data. The reduced problem is then

posed on a homogeneous non-local constraint space and analyzed by fractional resolvent estimates.

The numerical part uses a Beta-window approximation of the Caputo derivative. On each time interval the integer derivative is locally reconstructed by a polynomial interpolant, while the singular kernel is integrated analytically through incomplete Beta weights. This produces a high-order memory quadrature for smooth solutions. The spatial discretization uses a boundary-adapted Legendre–Galerkin basis, so that the homogeneous integral boundary constraints are satisfied by construction.

The paper is organized as follows. Section 2.1 introduces the model, the boundary operators, the compatibility conditions, and the lifting. Section 2.3 derives the Beta-window Caputo approximation and clarifies its relation with Chebyshev–Lobatto reconstruction. Section 2.4 proves the continuous well-posedness results and states the discrete stability framework. Section 3 presents the numerical experiments. Section 4 discusses the meaning and limitations of the results. Section 5 summarizes the conclusions and future work.

2. Materials and Methods

2.1. Problem Formulation

Let $\Omega = (a, b) \subset \mathbb{R}$ be a bounded spatial interval and let $T > 0$ be fixed. We denote by

$$Q_T := \Omega \times (0, T]$$

the space–time cylinder. In this work we consider the nonlinear time-fractional reaction–diffusion problem

$${}_0^C D_t^\alpha u(x, t) = \kappa u_{xx}(x, t) + f(x, t, u(x, t)), \quad (x, t) \in Q_T, \quad (2)$$

where $0 < \alpha < 1$, $\kappa > 0$, and ${}_0^C D_t^\alpha$ denotes the Caputo fractional derivative of order α .

Throughout this manuscript, the symbol α is used consistently for the fractional order. The symbol ρ is retained only inside the Mathematica code listings (Listings A1 and A2), where it is a software variable name. The Caputo operator is used because it allows the imposition of standard initial conditions in terms of integer-order data, while still encoding memory effects through a weakly singular convolution kernel [1–3,5,14]. This feature makes it particularly suitable for anomalous diffusion, viscoelasticity, transport with memory, and subdiffusive reaction–diffusion processes [4,5,15,16].

The equation is supplemented with the initial condition

$$u(x, 0) = u_0(x), \quad x \in [a, b], \quad (3)$$

and with the non-local boundary constraints

$$u(a, t) + \int_a^b \gamma_1(x) u(x, t) dx = \phi_1(t), \quad 0 \leq t \leq T, \quad (4)$$

$$u(b, t) + \int_a^b \gamma_2(x) u(x, t) dx = \phi_2(t), \quad 0 \leq t \leq T. \quad (5)$$

Here γ_1, γ_2 are prescribed kernel functions and ϕ_1, ϕ_2 are prescribed boundary data. The non-local terms in (4) and (5) allow the boundary response to depend not only on the endpoint values $u(a, t)$ and $u(b, t)$, but also on weighted spatial averages of the state. Such boundary conditions arise naturally in models with feedback, integral observation, mass constraints, and spatially distributed boundary control [17–21].

For later use, we introduce the boundary functionals

$$\mathcal{B}_1 v := v(a) + \int_a^b \gamma_1(x) v(x) dx, \quad (6)$$

$$\mathcal{B}_2 v := v(b) + \int_a^b \gamma_2(x) v(x) dx. \quad (7)$$

With this notation, the boundary conditions can be written compactly as

$$\mathcal{B}_1 u(t) = \phi_1(t), \quad \mathcal{B}_2 u(t) = \phi_2(t), \quad 0 \leq t \leq T. \quad (8)$$

The natural solution class considered in the present formulation is

$$u \in C([0, T]; H^2(a, b)) \cap C^{0,\alpha}([0, T]; L^2(a, b)),$$

although weaker mild or variational formulations can also be introduced depending on the assumptions imposed on the nonlinear term f [3,22–24].

Here $C^{0,\alpha}([0, T]; L^2(a, b))$ denotes the space of $L^2(a, b)$ -valued functions that are Hölder continuous in time with exponent α , namely functions satisfying

$$\|u(t) - u(s)\|_{L^2(a,b)} \leq C|t - s|^\alpha, \quad 0 \leq s, t \leq T.$$

This notation is used only to indicate the expected temporal regularity of stronger solutions. The well-posedness results below are formulated in the milder space $C([0, T]; L^2(a, b))$.

Assumption 1 (Data regularity and compatibility). *The data satisfy*

$$\gamma_1, \gamma_2 \in L^2(a, b), \quad \phi_1, \phi_2 \in C^1([0, T]), \quad u_0 \in H^2(a, b).$$

Moreover, the initial datum is compatible with the non-local boundary data, namely

$$\mathcal{B}_i u_0 = \phi_i(0), \quad i = 1, 2. \quad (9)$$

The compatibility condition (9) is necessary in order to avoid an artificial mismatch between the initial state and the boundary constraints at $t = 0$. In fractional diffusion problems this requirement is especially important, because the Caputo derivative involves the full time history of the solution and singular behavior near the initial time may otherwise be amplified [3,10,25].

Assumption 2 (Nondegeneracy of the boundary system). *Let*

$$p_0(x) = 1, \quad p_1(x) = \frac{2x - a - b}{b - a}.$$

We assume that the matrix

$$C_\gamma = \begin{pmatrix} \mathcal{B}_1 p_0 & \mathcal{B}_1 p_1 \\ \mathcal{B}_2 p_0 & \mathcal{B}_2 p_1 \end{pmatrix} \quad (10)$$

is invertible.

Assumption 2 states that the two non-local boundary constraints are linearly independent when restricted to the two-dimensional space generated by p_0 and p_1 . This is the minimal requirement needed to construct a boundary lifting that accounts for the two prescribed boundary functions ϕ_1 and ϕ_2 .

The choice of the two modes p_0 and p_1 is dictated by the number and structure of the boundary constraints. Since the problem contains exactly two scalar boundary conditions, a two-dimensional lifting space is the smallest possible space capable of matching the two boundary data. The functions p_0 and p_1 are selected because they span the affine polynomials on $[a, b]$, separate the two endpoints, and keep R_{xx} simple in the reduced equation. Other pairs of smooth functions could be used, provided that the corresponding 2×2 boundary matrix is non-singular.

The limitation is also clear. If more boundary functionals are imposed, the lifting space must have the same dimension as the number of independent constraints. In higher-dimensional domains, endpoint evaluations are replaced by traces on boundary portions, and the lifting must be constructed from functions whose traces and weighted domain integrals generate the prescribed boundary data. Thus, the present lifting is explicit and efficient for the one-dimensional two-constraint setting, while higher-dimensional extensions require a separate finite-dimensional lifting basis and an associated nondegeneracy matrix.

Proposition 1 (Boundedness of the boundary functionals). *If $\gamma_i \in L^2(a, b)$, then*

$$\mathcal{B}_i : H^1(a, b) \longrightarrow \mathbb{R}, \quad i = 1, 2,$$

is a bounded linear functional.

Proof. Linearity follows directly from the linearity of the point evaluation and the integral. Since $\Omega = (a, b)$ is a bounded interval, the trace theorem implies that there exists a constant $C_{\text{tr}} > 0$ such that

$$|v(a)| + |v(b)| \leq C_{\text{tr}} \|v\|_{H^1(a, b)}, \quad v \in H^1(a, b).$$

Therefore, for $i = 1, 2$,

$$|v(a)| \leq C_{\text{tr}} \|v\|_{H^1(a, b)}, \quad |v(b)| \leq C_{\text{tr}} \|v\|_{H^1(a, b)}.$$

Moreover, by the Cauchy–Schwarz inequality,

$$\left| \int_a^b \gamma_i(x) v(x) \, dx \right| \leq \|\gamma_i\|_{L^2(a, b)} \|v\|_{L^2(a, b)} \leq \|\gamma_i\|_{L^2(a, b)} \|v\|_{H^1(a, b)}.$$

Hence

$$|\mathcal{B}_i v| \leq \left(C_{\text{tr}} + \|\gamma_i\|_{L^2(a, b)} \right) \|v\|_{H^1(a, b)}.$$

Thus $\mathcal{B}_i$ is a bounded linear functional on $H^1(a, b)$. $\square$

Proposition 2 (Polynomial lifting of the non-local boundary data). *Assume that Assumption 2 holds. Then, for every boundary vector*

$$\Phi(t) := \begin{pmatrix} \phi_1(t) \\ \phi_2(t) \end{pmatrix},$$

there exists a unique polynomial lifting of the form

$$R(x, t) = r_0(t) p_0(x) + r_1(t) p_1(x) \quad (11)$$

such that

$$\mathcal{B}_1 R(t) = \phi_1(t), \quad \mathcal{B}_2 R(t) = \phi_2(t). \quad (12)$$

Moreover,

$$\begin{pmatrix} r_0(t) \\ r_1(t) \end{pmatrix} = C_\gamma^{-1} \begin{pmatrix} \phi_1(t) \\ \phi_2(t) \end{pmatrix}. \quad (13)$$

Proof. Let R be defined by (11). By linearity of the functionals $\mathcal{B}_1$ and $\mathcal{B}_2$, we have

$$\begin{pmatrix} \mathcal{B}_1 R \\ \mathcal{B}_2 R \end{pmatrix} = \begin{pmatrix} \mathcal{B}_1 p_0 & \mathcal{B}_1 p_1 \\ \mathcal{B}_2 p_0 & \mathcal{B}_2 p_1 \end{pmatrix} \begin{pmatrix} r_0 \\ r_1 \end{pmatrix}.$$

Using the definition of C_γ , this becomes

$$\begin{pmatrix} \mathcal{B}_1 R \\ \mathcal{B}_2 R \end{pmatrix} = C_\gamma \begin{pmatrix} r_0 \\ r_1 \end{pmatrix}.$$

The desired boundary identities (12) are therefore equivalent to the linear algebraic system

$$C_\gamma \begin{pmatrix} r_0(t) \\ r_1(t) \end{pmatrix} = \begin{pmatrix} \phi_1(t) \\ \phi_2(t) \end{pmatrix}.$$

Since C_γ is invertible by Assumption 2, this system has a unique solution for every $t \in [0, T]$. Hence (13) follows, and the polynomial lifting is well defined and unique in $\text{span}\{p_0, p_1\}$. $\square$

The lifting R allows us to transform the original nonhomogeneous non-local boundary problem into a homogeneous one. Indeed, setting

$$u(x, t) = w(x, t) + R(x, t),$$

we obtain

$$\mathcal{B}_1 w(t) = 0, \quad \mathcal{B}_2 w(t) = 0.$$

Consequently, the analysis of (2) may be reduced to the study of an evolution equation on the homogeneous constraint space

$$V := \left\{ v \in H^2(a, b) : \mathcal{B}_1 v = 0, \mathcal{B}_2 v = 0 \right\}. \quad (14)$$

This reduction is standard in the treatment of nonhomogeneous boundary data for parabolic and fractional parabolic equations, but here it must be adapted to the presence of integral boundary constraints [3,10,22–24].

Example 1 (Classical Dirichlet boundary conditions). If $\gamma_1 = \gamma_2 = 0$, then

$$\mathcal{B}_1 v = v(a), \quad \mathcal{B}_2 v = v(b).$$

Since

$$p_0(a) = 1, \quad p_0(b) = 1, \quad p_1(a) = -1, \quad p_1(b) = 1,$$

we obtain

$$C_\gamma = \begin{pmatrix} p_0(a) & p_1(a) \\ p_0(b) & p_1(b) \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}.$$

Therefore

$$\det C_\gamma = 2 \neq 0.$$

Hence the classical Dirichlet problem is included as a particular case of the present non-local formulation.

Counterexample 1 (Degenerate non-local boundary constraints). Let $\Omega = (0, 1)$ and let $\gamma_1 = \gamma_2 = 0$. Suppose, however, that the same boundary condition is imposed twice, namely

$$u(0, t) = \phi(t), \quad u(0, t) = \phi(t).$$

In this case both boundary functionals coincide:

$$\mathcal{B}_1 v = \mathcal{B}_2 v = v(0).$$

Consequently, the corresponding boundary matrix has two identical rows and is singular. Thus the two boundary constraints do not determine two independent boundary degrees of freedom. This example shows that the nondegeneracy condition in Assumption 2 is not merely technical; it is necessary for the lifting construction and for the well-posed reduction to homogeneous boundary data.

2.2. Construction of the Boundary-Adapted Legendre–Galerkin Basis

Let $L_k(x)$ denote the shifted Legendre polynomial on $[a, b]$, namely

$$L_k(x) = P_k\left(\frac{2x - a - b}{b - a}\right), \quad k = 0, 1, 2, \dots,$$

where P_k is the classical Legendre polynomial on $[-1, 1]$. The first two modes are

$$L_0(x) = 1, \quad L_1(x) = \frac{2x - a - b}{b - a}.$$

For each $k \geq 2$, we define the corrected mode

$$\psi_k(x) = L_k(x) - c_{k,0}L_0(x) - c_{k,1}L_1(x),$$

where the coefficients $c_{k,0}$ and $c_{k,1}$ are chosen so that

$$\mathcal{B}_1 \psi_k = 0, \quad \mathcal{B}_2 \psi_k = 0.$$

Equivalently,

$$C_\gamma \begin{pmatrix} c_{k,0} \\ c_{k,1} \end{pmatrix} = \begin{pmatrix} \mathcal{B}_1 L_k \\ \mathcal{B}_2 L_k \end{pmatrix}, \quad k = 2, \dots, N,$$

where

$$C_\gamma = \begin{pmatrix} \mathcal{B}_1 L_0 & \mathcal{B}_1 L_1 \\ \mathcal{B}_2 L_0 & \mathcal{B}_2 L_1 \end{pmatrix}.$$

Since C_γ is non-singular by Assumption 2, the coefficients $c_{k,0}$ and $c_{k,1}$ are uniquely defined.

The finite-dimensional homogeneous boundary-adapted space is then

$$V_N^\gamma = \text{span}\{\psi_2, \psi_3, \dots, \psi_N\}.$$

By construction,

$$\mathcal{B}_1 v_N = 0, \quad \mathcal{B}_2 v_N = 0, \quad v_N \in V_N^\gamma.$$

Therefore, the full approximation

$$u_N(x, t) = R(x, t) + y_N(x, t), \quad y_N(\cdot, t) \in V_N^\gamma,$$

satisfies

$$\mathcal{B}_1 u_N(t) = \phi_1(t), \quad \mathcal{B}_2 u_N(t) = \phi_2(t).$$

To turn the basis from a numerical implementation detail into a main mathematical contribution, the following proposition states that the boundary-adapted Legendre–Galerkin space is constructed explicitly and shown to preserve the non-local integral constraints exactly.

Proposition 3 (Exact boundary preservation of the Galerkin space). *Assume that C_γ is non-singular. Then every function*

$$u_N(x, t) = R(x, t) + \sum_{k=2}^N a_k(t) \psi_k(x)$$

satisfies the prescribed non-local boundary conditions

$$\mathcal{B}_1 u_N(t) = \phi_1(t), \quad \mathcal{B}_2 u_N(t) = \phi_2(t).$$

Proof. By the definition of the lifting,

$$\mathcal{B}_i R(t) = \phi_i(t), \quad i = 1, 2.$$

By construction of the corrected modes,

$$\mathcal{B}_i \psi_k = 0, \quad i = 1, 2, \quad k = 2, \dots, N.$$

Therefore, using linearity,

$$\begin{aligned} \mathcal{B}_i u_N(t) &= \mathcal{B}_i R(t) + \sum_{k=2}^N a_k(t) \mathcal{B}_i \psi_k \\ &= \phi_i(t) + 0 = \phi_i(t), \quad i = 1, 2. \end{aligned}$$

This proves exact preservation of the two non-local boundary constraints. $\square$

2.3. Preliminaries of Fractional Calculus and the Beta-Window Caputo Approximation

This section recalls the fractional operators used in the paper and introduces the temporal approximation employed in the numerical experiments. The approximation is based on an exact integration of the Caputo kernel against a local quadratic interpolant of the integer derivative. The resulting weights are expressed in terms of incomplete Beta functions. For this reason, we refer to the method as the Beta-window Caputo approximation.

The terms “Beta-window approximation” and “Chebyshev–Lobatto Caputo formula” refer to the same temporal discretization from two complementary viewpoints. The name Beta-window emphasizes that each history interval contributes explicit incomplete Beta weights after exact integration of the singular kernel. The name Chebyshev–Lobatto emphasizes that the local polynomial reconstruction uses endpoint and interior Lobatto-type data. In this manuscript, we use “Beta-window Caputo approximation” as the primary name and mention the Chebyshev–Lobatto interpretation only when the local interpolation structure is relevant.

Definition 1 (Caputo fractional derivative). *Let $m - 1 < \alpha < m$, where $m \in \mathbb{N}$, and let $v \in C^m([0, T])$. The Caputo fractional derivative of order α is defined by*

$${}_0^C D_t^\alpha v(t) = \frac{1}{\Gamma(m - \alpha)} \int_0^t (t - s)^{m - \alpha - 1} v^{(m)}(s) ds. \quad (15)$$

In the present paper we focus on $0 < \alpha < 1$, so that $m = 1$.

2.3.1. Local Quadratic Approximation of the Integer Derivative

Let

$$t_j = jh, \quad j = 0, 1, \dots, n, \quad t_n = nh,$$

be a uniform time grid. On each interval

$$I_j = [t_j, t_{j+1}],$$

we introduce the local variable

$$\theta = \frac{s - t_j}{h}, \quad 0 \leq \theta \leq 1.$$

For $m - 1 < \alpha < m$, set

$$g(s) = v^{(m)}(s).$$

On I_j , we approximate g by the quadratic interpolant through the three points

$$t_j, \quad t_j + \frac{h}{2}, \quad t_{j+1}.$$

Thus,

$$g(s) \approx g(t_j)L_0(\theta) + g\left(t_j + \frac{h}{2}\right)L_{1/2}(\theta) + g(t_{j+1})L_1(\theta), \quad (16)$$

where

$$L_0(\theta) = 2\theta^2 - 3\theta + 1, \quad (17)$$

$$L_{1/2}(\theta) = -4\theta^2 + 4\theta, \quad (18)$$

$$L_1(\theta) = 2\theta^2 - \theta. \quad (19)$$

Substituting (16) into the Caputo integral (15) and integrating the weakly singular kernel exactly gives the Beta-window approximation.

Let

$$\beta = m - \alpha.$$

For a fixed interval I_j , define

$$k = n - j.$$

Then

$$t_n - s = h(k - \theta),$$

and

$$ds = h d\theta.$$

Therefore,

$$(t_n - s)^{m-\alpha-1} ds = h^{m-\alpha} (k - \theta)^{\beta-1} d\theta.$$

The local contribution of I_j is then written in terms of the moments

$$\int_0^1 \theta^q (k - \theta)^{\beta-1} d\theta, \quad q = 0, 1, 2.$$

Using the change in variables $\theta = kz$, one obtains

$$\int_0^1 \theta^q (k - \theta)^{\beta-1} d\theta = k^{q+\beta} B_{1/k}(q+1, \beta), \quad (20)$$

where

$$B_z(a, b) = \int_0^z r^{a-1} (1-r)^{b-1} dr$$

is the incomplete Beta function.

2.3.2. The Beta-Window Caputo Formula

Combining the three quadratic basis functions with (20), we obtain

$$\mathcal{D}_{h,BW}^\alpha v(t_n) = \frac{h^{m-\alpha}}{\Gamma(m-\alpha)} \sum_{j=0}^{n-1} \left[v^{(m)}(t_j) W_0(k) + v^{(m)}\left(t_j + \frac{h}{2}\right) W_{1/2}(k) + v^{(m)}(t_{j+1}) W_1(k) \right],$$

where $k = n - j$, $\beta = m - \alpha$, and

$$W_0(k) = 2k^{\beta+2} B_{1/k}(3, \beta) - 3k^{\beta+1} B_{1/k}(2, \beta) + k^\beta B_{1/k}(1, \beta), \quad (22)$$

$$W_{1/2}(k) = -4k^{\beta+2} B_{1/k}(3, \beta) + 4k^{\beta+1} B_{1/k}(2, \beta), \quad (23)$$

$$W_1(k) = 2k^{\beta+2} B_{1/k}(3, \beta) - k^{\beta+1} B_{1/k}(2, \beta). \quad (24)$$

For the case $0 < \alpha < 1$, one has $m = 1$, and therefore

$$\beta = 1 - \alpha.$$

Hence the formula becomes

$$\mathcal{D}_{h,BW}^\alpha v(t_n) = \frac{h^{1-\alpha}}{\Gamma(1-\alpha)} \sum_{j=0}^{n-1} \left[v'(t_j) W_0(k) + v'\left(t_j + \frac{h}{2}\right) W_{1/2}(k) + v'(t_{j+1}) W_1(k) \right]. \quad (25)$$

The software implementation of (25) is given only in Appendices A and B. In the main text we use the mathematical notation $\mathcal{D}_{h,BW}^\alpha$ for the proposed Beta-window Caputo approximation.

2.3.3. Consistency and Interpretation

The Beta-window formula is exact whenever $v^{(m)}$ is a polynomial of degree at most two on each time interval. In particular, for $0 < \alpha < 1$, where $m = 1$, the formula is exact for functions v whose derivative is quadratic, that is, for polynomials v of degree at most three.

Theorem 1 (Consistency of the Beta-window Caputo approximation). *Let $0 < \alpha < 1$ and let $v \in C^4([0, T])$. Assume that $\mathcal{D}_{h,BW}^\alpha v(t_n)$ is defined by (25) on a uniform mesh $t_n = nh$. Then there exists a constant $C > 0$, independent of h and n , such that*

$$\left| {}_0^C D_t^\alpha v(t_n) - \mathcal{D}_{h,BW}^\alpha v(t_n) \right| \leq Ch^3 t_n^{1-\alpha} \max_{0 \leq s \leq t_n} |v^{(4)}(s)|.$$

In particular, for fixed $T > 0$,

$$\max_{1 \leq n \leq M} \left| {}_0^C D_t^\alpha v(t_n) - \mathcal{D}_{h,BW}^\alpha v(t_n) \right| = O(h^3).$$

Proof. For $0 < \alpha < 1$, the Caputo derivative is

$${}_0^C D_t^\alpha v(t_n) = \frac{1}{\Gamma(1-\alpha)} \int_0^{t_n} (t_n - s)^{-\alpha} v'(s) ds.$$

On each interval $I_j = [t_j, t_{j+1}]$, the Beta-window formula replaces v' by its quadratic interpolant $I_{2,j}v'$ at the points t_j , $t_j + h/2$, and t_{j+1} . Since $v \in C^4([0, T])$, the standard interpolation estimate gives

$$|v'(s) - I_{2,j}v'(s)| \leq Ch^3 \max_{\xi \in I_j} |v^{(4)}(\xi)|.$$

Therefore,

$$\begin{aligned} \left| {}^C_0 D_t^\alpha v(t_n) - \mathcal{D}_{h,BW}^\alpha v(t_n) \right| &\leq \frac{Ch^3}{\Gamma(1-\alpha)} \max_{0 \leq s \leq t_n} |v^{(4)}(s)| \int_0^{t_n} (t_n - s)^{-\alpha} ds \\ &= \frac{Ch^3}{\Gamma(1-\alpha)} \max_{0 \leq s \leq t_n} |v^{(4)}(s)| \frac{t_n^{1-\alpha}}{1-\alpha}. \end{aligned}$$

This proves the stated estimate. $\square$

For sufficiently smooth functions, the interpolation error of $v^{(m)}$ on each interval is of cubic order. Therefore, the proposed formula is expected to exhibit high-order accuracy for smooth data. This is confirmed in the numerical experiments, where the test function

$$v(t) = t^4$$

produces observed convergence rates close to 3.3 for $\alpha = 0.65$.

The main advantage of the construction is that the singular Caputo kernel is not approximated by a low-order difference formula. Instead, the kernel is integrated exactly against the local quadratic interpolant, producing closed-form weights involving incomplete Beta functions.

2.3.4. Relation with the Classical L_1 Approximation

The classical L_1 method approximates v by a piecewise linear interpolant on each interval. For smooth solutions on uniform meshes, its expected order is

$$O(h^{2-\alpha}).$$

In contrast, the Beta-window formula uses a quadratic interpolation of $v^{(m)}$ and exact incomplete Beta integration of the Caputo kernel. This leads to a substantially higher observed accuracy for smooth data.

For example, when $\alpha = 0.65$, the expected smooth-solution order of the L_1 method is

$$2 - \alpha = 1.35,$$

whereas the numerical experiments for the Beta-window formula give rates close to 3.3. This is the basis of the comparison reported in Section 3.5.

2.4. Well Posedness: Existence, Uniqueness, and Stability

This section establishes the well posedness of the nonlinear time-fractional reaction–diffusion problem with non-local boundary conditions. The main difficulty is that the boundary constraints are not homogeneous and involve integral terms over the spatial domain. Therefore, before applying the fractional evolution theory, the nonhomogeneous boundary data must be removed by means of the lifting constructed in Proposition 2. After this reduction, the problem is posed on a homogeneous constraint space, and the Caputo memory term is handled through fractional resolvent operators.

2.4.1. Structural Assumptions

We impose the following assumptions on the nonlinear reaction term.

Assumption 3 (Reaction term). *The nonlinear function*

$$f : \Omega \times [0, T] \times \mathbb{R} \longrightarrow \mathbb{R}$$

satisfies:

- (i) for almost every $(x, t) \in Q_T$, the mapping $z \mapsto f(x, t, z)$ is continuous;
- (ii) for every $z \in \mathbb{R}$, the mapping $(x, t) \mapsto f(x, t, z)$ is measurable;
- (iii) $f(\cdot, \cdot, 0) \in L^2(Q_T)$;
- (iv) there exists a constant $L_f > 0$ such that

$$|f(x, t, z_1) - f(x, t, z_2)| \leq L_f |z_1 - z_2|$$

for all $z_1, z_2 \in \mathbb{R}$ and for almost every $(x, t) \in Q_T$.

The global Lipschitz condition is used to obtain uniqueness and continuous dependence in the natural L^2 framework. It can be weakened to a local Lipschitz assumption if one additionally proves an a priori bound for the solution.

Assumption 4 (Elliptic realization with non-local boundary constraints). *Let*

$$A_\gamma v = -v_{xx},$$

with domain

$$D(A_\gamma) = \{v \in H^2(a, b) : \mathcal{B}_1 v = 0, \mathcal{B}_2 v = 0\}.$$

We assume that A_γ is sectorial on $L^2(a, b)$ and that $-\kappa A_\gamma$ generates a bounded analytic semigroup on $L^2(a, b)$.

Remark 1. For the classical Dirichlet Laplacian, Assumption 4 is standard. For general non-local integral boundary conditions, it is the operator-theoretic hypothesis that guarantees the validity of the fractional resolvent formulation. Without this assumption, one cannot simply use the Dirichlet heat semigroup, because the domain of the operator is changed by the non-local boundary constraints.

For the one-dimensional class considered here, this assumption can be justified by viewing the non-local boundary conditions as a finite-rank perturbation of regular boundary conditions. Boundary perturbation results for generators of analytic semigroups imply that strongly elliptic second-order operators with regular non-local boundary functionals generate analytic semigroups once the corresponding boundary realization is closed and the resolvent problem is well posed [26–28]. The present nondegeneracy condition guarantees the lifting needed to reduce the nonhomogeneous problem to a closed homogeneous constraint space. Thus, Assumption 4 is not an additional numerical hypothesis; it is the standard operator-theoretic requirement ensuring that the non-local elliptic realization admits the fractional resolvent family used in the mild solution formula.

A complete characterization of sectoriality for arbitrary integral kernels γ_i is outside the scope of this paper. The analysis applies to kernels for which the above boundary realization is sectorial. This restriction has now been stated explicitly.

Example 2 (Globally Lipschitz reaction). *If*

$$f(x, t, u) = q(x, t) + \lambda u,$$

where $q \in L^2(Q_T)$ and $\lambda \in \mathbb{R}$, then Assumption 3 holds with

$$L_f = |\lambda|.$$

Counterexample 2 (A locally but not globally Lipschitz reaction). *The reaction term*

$$f(u) = u^2$$

is locally Lipschitz but not globally Lipschitz on $\mathbb{R}$. Indeed,

$$|u^2 - v^2| = |u + v||u - v|.$$

If a global Lipschitz constant $L > 0$ existed, then by choosing $v = 0$ one would obtain

$$u^2 \leq L|u|, \quad u \in \mathbb{R},$$

which is impossible as $|u| \rightarrow \infty$. Therefore, the global theory below does not apply directly to $f(u) = u^2$ unless one first proves an a priori bound or replaces u^2 by a truncated globally Lipschitz nonlinearity.

2.4.2. Reduction to Homogeneous Non-Local Boundary Conditions

Let $R = R(x, t)$ be the lifting function constructed in Proposition 2. Thus

$$\mathcal{B}_1 R(t) = \phi_1(t), \quad \mathcal{B}_2 R(t) = \phi_2(t), \quad 0 \leq t \leq T.$$

We define

$$y(x, t) = u(x, t) - R(x, t).$$

Then

$$u(x, t) = y(x, t) + R(x, t).$$

Since u satisfies the original boundary conditions and R satisfies the same boundary data, the new unknown y satisfies homogeneous non-local boundary conditions:

$$\mathcal{B}_1 y(t) = 0, \quad \mathcal{B}_2 y(t) = 0.$$

Substituting $u = y + R$ into the original equation gives

$${}^C D_0^\alpha (y + R) = \kappa(y + R)_{xx} + f(x, t, y + R).$$

Hence

$${}^C D_0^\alpha y = \kappa y_{xx} + f(x, t, y + R) - {}^C D_0^\alpha R + \kappa R_{xx}.$$

Using the positive operator

$$A_\gamma y = -y_{xx},$$

the transformed problem becomes

$${}^C D_0^\alpha y(t) + \kappa A_\gamma y(t) = F(t, y(t)), \quad 0 < t \leq T, \quad (26)$$

where

$$F(t, y) = f(\cdot, t, y + R(t)) - {}^C D_0^\alpha R(t) + \kappa R_{xx}(t). \quad (27)$$

The initial datum is

$$y(0) = y_0 := u_0 - R(0). \quad (28)$$

Remark 2. It is important to write the correction term as κR_{xx} rather than as $-\kappa A_\gamma R$. In general, R satisfies the nonhomogeneous boundary conditions and therefore need not belong to $D(A_\gamma)$. Since R is the explicit lifting constructed from polynomial functions, the expression R_{xx} is meaningful in the classical sense.

Because of the compatibility condition

$$\mathcal{B}_i u_0 = \phi_i(0), \quad i = 1, 2,$$

we have

$$\mathcal{B}_i y_0 = \mathcal{B}_i u_0 - \mathcal{B}_i R(0) = \phi_i(0) - \phi_i(0) = 0.$$

Thus the transformed initial datum is compatible with the homogeneous constraint space.

2.4.3. Mild Solution Formulation

Under Assumption 4, the operator A_γ generates the fractional solution operators

$$S_\alpha(t) = E_\alpha(-\kappa A_\gamma t^\alpha), \quad P_\alpha(t) = t^{\alpha-1} E_{\alpha,\alpha}(-\kappa A_\gamma t^\alpha),$$

where E_α and $E_{\alpha,\alpha}$ denote the one- and two-parameter Mittag-Leffler functions. These operators satisfy the estimates

$$\|S_\alpha(t)\|_{\mathcal{L}(L^2)} \leq M_S, \quad \|P_\alpha(t)\|_{\mathcal{L}(L^2)} \leq M_P t^{\alpha-1}, \quad 0 < t \leq T, \quad (29)$$

for some constants $M_S, M_P > 0$.

Definition 2 (Mild solution of the homogeneous problem). A function

$$y \in C([0, T]; L^2(a, b))$$

is called a mild solution of (26) on $[0, T]$ if it satisfies

$$y(t) = S_\alpha(t)y_0 + \int_0^t P_\alpha(t-s)F(s, y(s))ds, \quad 0 \leq t \leq T. \quad (30)$$

The corresponding mild solution of the original problem is then

$$u(t) = y(t) + R(t).$$

2.4.4. Local Existence and Uniqueness

Theorem 2 (Local existence and uniqueness). Assume that Assumptions 1–4 hold. Then there exists $T_0 \in (0, T]$ such that problem (2)–(5) admits a unique mild solution on $[0, T_0]$.

Proof. Let

$$X_{T_0} := C([0, T_0]; L^2(a, b)),$$

with norm

$$\|y\|_{X_{T_0}} = \sup_{0 \leq t \leq T_0} \|y(t)\|_{L^2(a, b)}.$$

For $y \in X_{T_0}$, define the operator

$$(\mathcal{T}y)(t) = S_\alpha(t)y_0 + \int_0^t P_\alpha(t-s)F(s, y(s)) ds.$$

A fixed point of $\mathcal{T}$ is precisely a mild solution of (26).

First, using Assumption 3, we obtain

$$\begin{aligned} \|F(t, y) - F(t, z)\|_{L^2} &= \|f(\cdot, t, y + R(t)) - f(\cdot, t, z + R(t))\|_{L^2} \\ &\leq L_f \|y - z\|_{L^2}. \end{aligned}$$

Moreover,

$$H_0 := \sup_{0 \leq t \leq T_0} \|F(t, 0)\|_{L^2} < \infty.$$

This follows from the assumptions on f , the regularity of the lifting R , and the fact that $\phi_1, \phi_2 \in C^1([0, T])$.

Let

$$B_\varrho = \{y \in X_{T_0} : \|y\|_{X_{T_0}} \leq \varrho\}.$$

For $y \in B_\varrho$, the operator estimates (29) give

$$\begin{aligned} \|(\mathcal{T}y)(t)\|_{L^2} &\leq M_S \|y_0\|_{L^2} + M_P \int_0^t (t-s)^{\alpha-1} \|F(s, y(s))\|_{L^2} ds \\ &\leq M_S \|y_0\|_{L^2} + M_P (H_0 + L_f \varrho) \int_0^t (t-s)^{\alpha-1} ds \\ &= M_S \|y_0\|_{L^2} + \frac{M_P}{\alpha} t^\alpha (H_0 + L_f \varrho). \end{aligned}$$

Hence

$$\|\mathcal{T}y\|_{X_{T_0}} \leq M_S \|y_0\|_{L^2} + \frac{M_P}{\alpha} T_0^\alpha (H_0 + L_f \varrho).$$

Choose $\varrho > M_S \|y_0\|_{L^2}$ and then choose $T_0 > 0$ sufficiently small so that

$$\frac{M_P}{\alpha} T_0^\alpha (H_0 + L_f \varrho) \leq \varrho - M_S \|y_0\|_{L^2}.$$

Then $\mathcal{T}(B_\varrho) \subset B_\varrho$.

Now let $y, z \in B_\varrho$. Again using (29) and the Lipschitz property of F ,

$$\begin{aligned} \|(\mathcal{T}y)(t) - (\mathcal{T}z)(t)\|_{L^2} &\leq M_P L_f \int_0^t (t-s)^{\alpha-1} \|y(s) - z(s)\|_{L^2} ds \\ &\leq \frac{M_P L_f}{\alpha} T_0^\alpha \|y - z\|_{X_{T_0}}. \end{aligned}$$

Therefore

$$\|\mathcal{T}y - \mathcal{T}z\|_{X_{T_0}} \leq q \|y - z\|_{X_{T_0}}, \quad q = \frac{M_P L_f}{\alpha} T_0^\alpha.$$

Choosing T_0 smaller if necessary so that $q < 1$, $\mathcal{T}$ becomes a contraction on B_ϱ . By Banach's fixed-point theorem, there exists a unique $y \in B_\varrho$ satisfying

$$\mathcal{T}y = y.$$

Thus (26) has a unique mild solution on $[0, T_0]$.

Finally, defining

$$u(t) = y(t) + R(t),$$

we recover a solution of the original non-local boundary value problem. Indeed,

$$\mathcal{B}_i u(t) = \mathcal{B}_i y(t) + \mathcal{B}_i R(t) = 0 + \phi_i(t) = \phi_i(t), \quad i = 1, 2.$$

Uniqueness of u follows from uniqueness of y . $\square$

2.4.5. Global Existence Under Global Lipschitz Growth

The local proof above uses a small time interval in order to obtain a contraction in the standard supremum norm. For the global result, it is more convenient to use a weighted Bielecki norm adapted to the fractional memory kernel.

Theorem 3 (Global existence and uniqueness). *Assume that Assumptions 1–4 hold. Suppose also that the Lipschitz constant L_f in Assumption 3 is valid on the whole interval $[0, T]$ and that*

$$H_T := \sup_{0 \leq t \leq T} \|F(t, 0)\|_{L^2(a, b)} < \infty.$$

Then problem (2)–(5) admits a unique mild solution on the whole interval $[0, T]$.

Proof. Let

$$X_T = C([0, T]; L^2(a, b)).$$

For $\lambda > 0$, define the weighted norm

$$\|y\|_\lambda = \sup_{0 \leq t \leq T} \frac{\|y(t)\|_{L^2}}{E_\alpha(\lambda t^\alpha)}.$$

This norm is equivalent to the usual supremum norm on X_T , since $E_\alpha(\lambda t^\alpha)$ is positive and bounded above and below on $[0, T]$.

Define

$$(\mathcal{T}y)(t) = S_\alpha(t)y_0 + \int_0^t P_\alpha(t-s)F(s, y(s)) ds.$$

For $y, z \in X_T$, we have

$$\begin{aligned} \|(\mathcal{T}y)(t) - (\mathcal{T}z)(t)\|_{L^2} &\leq M_P L_f \int_0^t (t-s)^{\alpha-1} \|y(s) - z(s)\|_{L^2} ds \\ &\leq M_P L_f \|y - z\|_\lambda \int_0^t (t-s)^{\alpha-1} E_\alpha(\lambda s^\alpha) ds. \end{aligned}$$

Using the identity

$$\int_0^t (t-s)^{\alpha-1} E_\alpha(\lambda s^\alpha) ds = \frac{\Gamma(\alpha)}{\lambda} [E_\alpha(\lambda t^\alpha) - 1],$$

we obtain

$$\frac{\|(\mathcal{T}y)(t) - (\mathcal{T}z)(t)\|_{L^2}}{E_\alpha(\lambda t^\alpha)} \leq \frac{M_P L_f \Gamma(\alpha)}{\lambda} \|y - z\|_\lambda.$$

Therefore

$$\|\mathcal{T}y - \mathcal{T}z\|_\lambda \leq \frac{M_P L_f \Gamma(\alpha)}{\lambda} \|y - z\|_\lambda.$$

Choose λ sufficiently large so that

$$\frac{M_P L_f \Gamma(\alpha)}{\lambda} < 1.$$

Then $\mathcal{T}$ is a contraction on X_T endowed with the norm $\|\cdot\|_\lambda$.

It remains only to observe that $\mathcal{T}$ maps X_T into itself. Indeed, for $y \in X_T$,

$$\|(\mathcal{T}y)(t)\|_{L^2} \leq M_S \|y_0\|_{L^2} + M_P \int_0^t (t-s)^{\alpha-1} \left(H_T + L_f \|y(s)\|_{L^2} \right) ds,$$

and the right-hand side is finite for every $0 \leq t \leq T$. Hence $\mathcal{T} : X_T \rightarrow X_T$ is well defined.

By Banach's fixed-point theorem, $\mathcal{T}$ has a unique fixed point $y \in X_T$. Thus the homogeneous problem has a unique mild solution on $[0, T]$. The original solution is recovered by

$$u(t) = y(t) + R(t),$$

and uniqueness of u follows from uniqueness of y . $\square$

2.4.6. L^2 -Stability with Respect to the Initial Datum

Theorem 4 (L^2 -stability with respect to the initial datum). *Let u and $\tilde{u}$ be two mild solutions corresponding to compatible initial data u_0 and $\tilde{u}_0$, with the same boundary data and the same reaction term. Then*

$$\|u(t) - \tilde{u}(t)\|_{L^2} \leq M_S E_\alpha \left(M_P L_f \Gamma(\alpha) t^\alpha \right) \|u_0 - \tilde{u}_0\|_{L^2}, \quad 0 \leq t \leq T.$$

Consequently,

$$\max_{0 \leq t \leq T} \|u(t) - \tilde{u}(t)\|_{L^2} \leq C_s \|u_0 - \tilde{u}_0\|_{L^2},$$

where

$$C_s = M_S E_\alpha \left(M_P L_f \Gamma(\alpha) T^\alpha \right).$$

Proof. Since the two problems have the same boundary data, their lifting function is the same. Let

$$y = u - R, \quad \tilde{y} = \tilde{u} - R.$$

Then

$$u - \tilde{u} = y - \tilde{y}.$$

Subtracting the two mild formulations gives

$$y(t) - \tilde{y}(t) = S_\alpha(t)(u_0 - \tilde{u}_0) + \int_0^t P_\alpha(t-s) [F(s, y(s)) - F(s, \tilde{y}(s))] ds.$$

Taking the L^2 norm and using the operator bounds yields

$$\|y(t) - \tilde{y}(t)\|_{L^2} \leq M_S \|u_0 - \tilde{u}_0\|_{L^2} + M_P L_f \int_0^t (t-s)^{\alpha-1} \|y(s) - \tilde{y}(s)\|_{L^2} ds.$$

Since

$$\int_0^t (t-s)^{\alpha-1} g(s) ds = \Gamma(\alpha) I_0^\alpha g(t),$$

the previous inequality becomes

$$g(t) \leq M_S \|u_0 - \tilde{u}_0\|_{L^2} + M_P L_f \Gamma(\alpha) I_0^\alpha g(t),$$

where

$$g(t) = \|y(t) - \tilde{y}(t)\|_{L^2}.$$

Applying the fractional Gronwall inequality gives

$$g(t) \leq M_S E_\alpha \left(M_P L_f \Gamma(\alpha) t^\alpha \right) \|u_0 - \tilde{u}_0\|_{L^2}.$$

Because $u - \tilde{u} = y - \tilde{y}$, the same estimate holds for $u - \tilde{u}$. Taking the maximum over $0 \leq t \leq T$ gives the stated constant C_s . $\square$

2.4.7. Continuous Dependence

Corollary 1 (Continuous dependence on the initial datum). *For every $\varepsilon > 0$, if*

$$\|u_0 - \tilde{u}_0\|_{L^2} < \delta, \quad \delta = \frac{\varepsilon}{C_s},$$

then

$$\max_{0 \leq t \leq T} \|u(t) - \tilde{u}(t)\|_{L^2} < \varepsilon.$$

Proof. By Theorem 4,

$$\max_{0 \leq t \leq T} \|u(t) - \tilde{u}(t)\|_{L^2} \leq C_s \|u_0 - \tilde{u}_0\|_{L^2}.$$

Thus, if

$$\|u_0 - \tilde{u}_0\|_{L^2} < \varepsilon / C_s,$$

then

$$\max_{0 \leq t \leq T} \|u(t) - \tilde{u}(t)\|_{L^2} < \varepsilon.$$

$\square$

2.4.8. Connection with the Chebyshev–Lobatto Caputo Approximation

The continuous stability estimate above provides the analytical foundation for the high-order Caputo discretization introduced in Section 2.3. At the semi-discrete level, the Caputo derivative is replaced by the Chebyshev–Lobatto cubic approximation

$$\mathcal{D}_{\tau, \text{CL}}^\alpha y(t_n) = \sum_{j=1}^n \sum_{m=0}^3 \omega_{n,j,m}^{(\alpha)} y(\tau_{j,m}),$$

where the weights are obtained by integrating the Caputo kernel against the derivatives of the local cubic basis functions.

The corresponding implicit semi-discrete problem has the form

$$\mathcal{D}_{\tau, \text{CL}}^\alpha Y^n + \kappa A_\gamma Y^n = F(t_n, Y^n), \quad n = 1, \dots, N. \quad (31)$$

The original approximation is recovered by

$$U^n = Y^n + R(t_n).$$

The role of the continuous estimate is twofold. First, it shows that the underlying fractional evolution problem is stable with respect to perturbations of the initial datum. Second, after replacing the continuous Caputo operator by the Chebyshev–Lobatto approximation, it suggests the expected numerical error structure

$$\max_{1 \leq n \leq N} \|Y^n - y(t_n)\|_{L^2} \leq C \max_{1 \leq n \leq N} \|\mathcal{R}^n\|_{L^2},$$

where $\mathcal{R}^n$ denotes the local consistency defect of the Chebyshev–Lobatto Caputo formula.

For sufficiently smooth solutions, the cubic construction gives

$$\|\mathcal{R}^n\|_{L^2} = O(h^3),$$

on quasi-uniform meshes. Hence, under the same stability mechanism, one expects

$$\max_{1 \leq n \leq N} \|Y^n - y(t_n)\|_{L^2} = O(h^3).$$

This improves the formal smooth-solution order of the classical $L1$ approximation, whose uniform-mesh accuracy is typically

$$O(h^{2-\alpha}).$$

However, the higher-order formula has a more delicate stability structure than $L1$. The $L1$ weights are positive and monotone, which makes discrete energy estimates and discrete fractional Gronwall arguments relatively direct. By contrast, the Chebyshev–Lobatto cubic weights are generated by derivatives of cubic cardinal polynomials and are not necessarily all positive. Therefore, the method should be used with an implicit or semi-implicit treatment of the diffusion operator. In this setting, the sectoriality of A_γ , the Lipschitz continuity of F , and the fractional memory estimates provide the appropriate stability framework.

In summary, the $L1$ scheme is simpler and more robust, whereas the Chebyshev–Lobatto cubic Caputo formula is more accurate for smooth or properly graded solutions. This makes the new formula particularly suitable when the spatial discretization is also high-order, for example in spectral or spectral–collocation methods.

2.4.9. Conditional Fully Discrete Error Estimate

Theorem 5 (Conditional fully discrete error estimate). *Let u be a sufficiently smooth solution of (2)–(5), and let U_N^n be the fully discrete implicit Beta-window Legendre–Galerkin approximation. Assume that the discrete coercivity condition*

$$c_\alpha \tau^{-\alpha} + \kappa \lambda_{1,N} > L_f$$

holds. Then the error satisfies the estimate

$$\max_{1 \leq n \leq M} \|u(t_n) - U_N^n\|_{L^2(a,b)} \leq C \left(\tau^3 + \inf_{\chi_N \in V_N^\gamma} \max_{0 \leq t \leq T} \|u(t) - R(t) - \chi_N(t)\|_{H^1(a,b)} + \max_{1 \leq n \leq M} \varepsilon_{\text{nl}}^n \right),$$

where $\varepsilon_{\text{nl}}^n$ denotes the nonlinear algebraic residual at time level t_n , and C depends on T , α , κ , L_f , and the stability constants, but is independent of τ .

Proof. The error is decomposed as

$$u(t_n) - U_N^n = [u(t_n) - R(t_n) - \Pi_N^\gamma(u(t_n) - R(t_n))] + [\Pi_N^\gamma(u(t_n) - R(t_n)) - Y_N^n],$$

where Π_N^γ denotes the projection onto the homogeneous boundary-adapted Galerkin space. The first term is the spectral projection error. The second term satisfies the discrete error equation obtained by subtracting the fully discrete scheme from the projected continuous equation. The residual contains the temporal consistency defect of the Beta-window Caputo approximation, the spatial projection defect, and the nonlinear algebraic residual.

By Theorem 1, the temporal consistency defect is $O(\tau^3)$ for smooth solutions. The Lipschitz condition on f gives

$$\|f(\cdot, t, U) - f(\cdot, t, V)\|_{L^2} \leq L_f \|U - V\|_{L^2}.$$

Using the discrete coercivity condition

$$c_\alpha \tau^{-\alpha} + \kappa \lambda_{1,N} > L_f,$$

the implicit Galerkin operator is stable. A discrete fractional Gronwall argument applied to the resulting error inequality yields

$$\begin{aligned} & \max_{1 \leq n \leq M} \|\Pi_N^\gamma(u(t_n) - R(t_n)) - Y_N^n\|_{L^2} \\ & \leq C \left(\tau^3 + \max_{0 \leq t \leq T} \|u(t) - R(t) - \Pi_N^\gamma(u(t) - R(t))\|_{H^1} + \max_{1 \leq n \leq M} \varepsilon_{nl}^n \right). \end{aligned}$$

Combining this estimate with the projection decomposition gives the result. $\square$

3. Results

The numerical section has been reorganized so that each experiment states its aim, input data, error measure, and interpretation. Captions were revised to make clear whether a figure validates the temporal Caputo approximation, the full PDE discretization, or the non-local boundary treatment.

3.1. Example 1: Validation of the Beta-Window Caputo Approximation

The first experiment is designed to validate the temporal discretization independently of the spatial discretization. In this test, we consider the scalar function

$$f(t) = t^4, \quad 0 \leq t \leq 1, \quad (32)$$

and compute its Caputo derivative of order $\alpha = 0.65$. This example is particularly useful because the exact Caputo derivative is available in closed form. Therefore, the numerical error can be measured without introducing any additional reference approximation.

3.1.1. Input Data

The computational parameters are

$$\alpha = 0.65, \quad T = 1, \quad M = 40, \quad h = \frac{T}{M} = 0.025, \quad n = M. \quad (33)$$

In the Mathematica implementation, these data correspond to

```
rho = 0.65;    Tfinal = 1.0;    M = 40;
h = Tfinal/M;    n = M;    fl[t_NumericQ] := t^4.
```

The parameter `qReference = 400` is included in the code for quadrature-based reference computations in non-polynomial tests. In the present polynomial test, the reference value is computed exactly from the closed-form Caputo formula.

3.1.2. Exact Caputo Derivative

For $0 < \alpha < 1$, the Caputo derivative of a power function satisfies

$${}_0^C D_t^\alpha t^\beta = \frac{\Gamma(\beta + 1)}{\Gamma(\beta + 1 - \alpha)} t^{\beta - \alpha}, \quad \beta > 0. \quad (34)$$

Taking $\beta = 4$, we obtain

$${}_0^C D_t^\alpha t^4 = \frac{\Gamma(5)}{\Gamma(5 - \alpha)} t^{4 - \alpha}. \quad (35)$$

Therefore, at $T = 1$ and $\alpha = 0.65$,

$${}_0^C D_t^{0.65} t^4 \big|_{t=1} = \frac{\Gamma(5)}{\Gamma(4.35)}. \quad (36)$$

3.1.3. Numerical Approximation

The proposed Beta-window approximation gives

$$\mathcal{D}_{h,BW}^\alpha f(1) = 2.5340392279211. \quad (37)$$

The exact value is

$${}_0^C D_t^\alpha f(1) = 2.534038971593. \quad (38)$$

Hence the absolute error is

$$E_h = \left| {}_0^C D_t^\alpha f(1) - \mathcal{D}_{h,BW}^\alpha f(1) \right| = 2.5632810636163 \times 10^{-7}. \quad (39)$$

3.1.4. Convergence Study

To study convergence, the computation was repeated for

$$M = 10, 20, 40, 80, 160, \quad h = \frac{1}{M}.$$

The observed rate is computed by

$$\text{rate} = \frac{\log(E_h/E_{h/2})}{\log 2}. \quad (40)$$

The results in Table 1 show a clear and stable high-order convergence pattern. The observed rate remains close to 3.3 throughout the refinement process. In particular, the error decreases from 2.58728×10^{-5} for $M = 10$ to 2.49465×10^{-9} for $M = 160$, which corresponds to a reduction by more than four orders of magnitude. This confirms that the proposed Beta-window construction is not merely consistent, but also highly accurate for smooth data.

Table 1. Convergence of the Beta-window Caputo approximation for $f(t) = t^4$, with $\alpha = 0.65$ and $T = 1$.

M	h	Absolute Error E_h	Observed Rate
10	1.0000×10^{-1}	2.58728×10^{-5}	–
20	5.0000×10^{-2}	2.58397×10^{-6}	3.32378
40	2.5000×10^{-2}	2.56328×10^{-7}	3.33353
80	1.2500×10^{-2}	2.53208×10^{-8}	3.33960
160	6.2500×10^{-3}	2.49465×10^{-9}	3.34341

The observed rate is slightly above third order. This behavior is consistent with the fact that the test function is a smooth polynomial and the Beta-window formula integrates the weakly singular Caputo kernel through explicit incomplete Beta contributions. Hence the singular memory kernel is not treated by a crude finite-difference approximation; instead, the singular factor is incorporated analytically into the weights. This explains the excellent accuracy observed in this test.

3.1.5. Graphical Validation

Figure 1 shows the log–log convergence curve. The almost straight line confirms that the error behaves as a power of the time step h . The slope is consistent with the rates reported in Table 1.

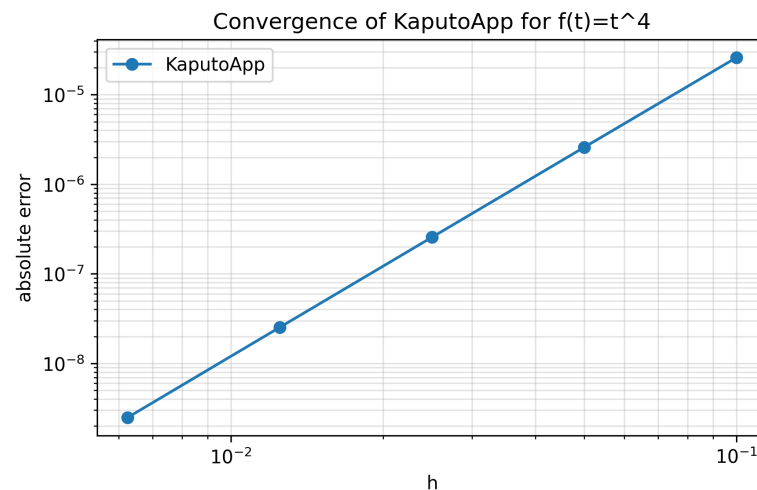


Figure 1. Log–log convergence curve of the Beta-window Caputo approximation for $f(t) = t^4$. The observed rate is approximately 3.3, confirming high-order accuracy for smooth data.

Figure 2 compares the numerical Caputo derivative with the exact derivative on the whole interval. The two curves are visually indistinguishable at the scale of the plot, which confirms that the method reproduces the temporal profile of the fractional derivative with high accuracy.

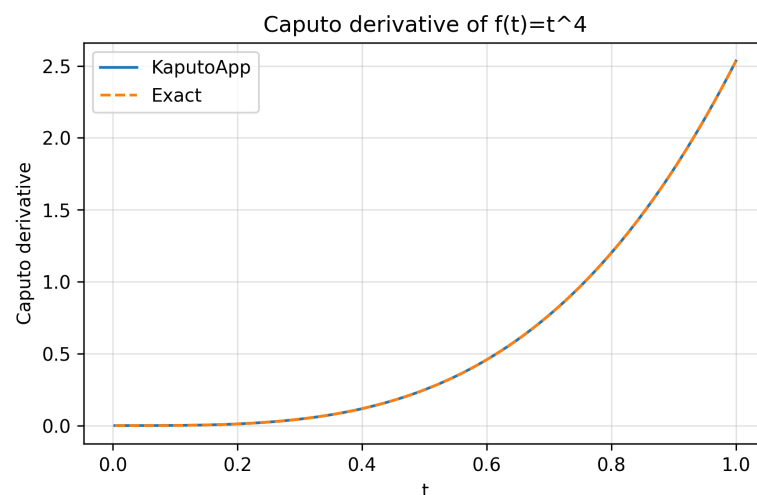


Figure 2. Numerical and exact Caputo derivatives of $f(t) = t^4$. The Beta-window approximation follows the exact profile throughout the interval $0 \leq t \leq 1$.

The pointwise absolute error is shown in Figure 3. The error remains uniformly small over the entire interval. In particular, for the refined curve used in the plot, the error stays below 3×10^{-8} . The slight increase toward $t = 1$ is expected because the Caputo derivative accumulates the full history of the function over $[0, t]$, and therefore the accumulated memory error becomes larger as t increases.

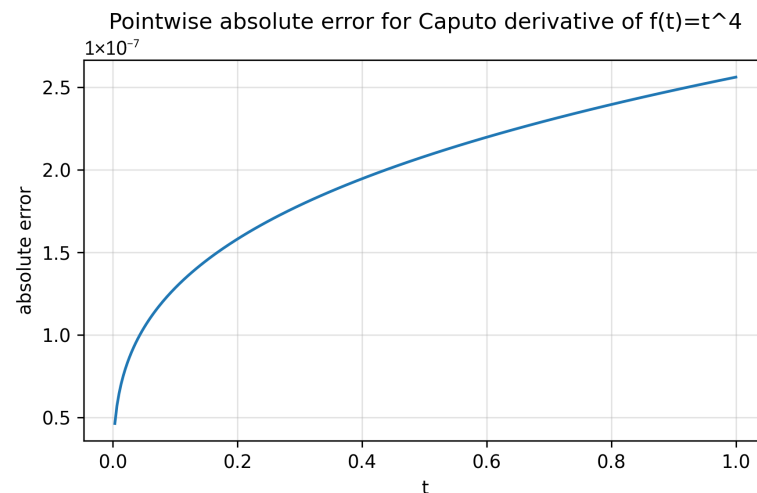


Figure 3. Pointwise absolute error of the Beta-window Caputo approximation for $f(t) = t^4$. The error remains below 3×10^{-8} , confirming uniform accuracy over the whole time interval.

3.1.6. Discussion

This example provides a clean validation of the proposed Caputo approximation. Since the exact fractional derivative is known, the numerical error is not contaminated by a reference solver, spatial discretization, or nonlinear iteration error. The test therefore isolates the accuracy of the temporal memory approximation itself.

The results show three important facts. First, the absolute error at $T = 1$ for $M = 40$ is already of order 10^{-7} . Second, the convergence table displays a stable rate close to 3.3, which supports the high-order character of the Beta-window construction. Third, the numerical and exact curves are visually indistinguishable, while the pointwise error remains uniformly small over the whole interval.

This behavior is stronger than what is typically expected from the classical $L1$ approximation on a uniform mesh, whose smooth-solution order is $O(h^{2-\alpha})$. For the present value $\alpha = 0.65$, this would correspond to $O(h^{1.35})$. By contrast, the present experiment shows an empirical rate close to $O(h^{3.3})$. Therefore, the Beta-window formula provides a substantial gain in temporal accuracy for smooth functions. This justifies its use as the temporal core of the high-order spectral-iterative method developed in this work.

3.2. Example 2: Linear Fractional Reaction–Diffusion Test

The second experiment validates the full space–time numerical method for a linear fractional reaction–diffusion equation. In contrast with Section 3.1, where only the temporal Caputo approximation was tested, the present example includes the diffusion operator, the reaction term, the spatial Galerkin discretization, and the time-fractional memory operator.

We consider the problem

$${}_0^C D_t^\alpha u(x, t) = \kappa u_{xx}(x, t) + q(x, t) + \lambda u(x, t), \quad (x, t) \in (0, 1) \times (0, 1], \quad (41)$$

with homogeneous boundary conditions

$$u(0, t) = 0, \quad u(1, t) = 0, \quad 0 \leq t \leq 1, \quad (42)$$

and initial condition

$$u(x, 0) = 0. \quad (43)$$

3.2.1. Input Data

The computational parameters are

$$\Omega = (0, 1), \quad T = 1, \quad \alpha = 0.65, \quad \kappa = 1, \quad \lambda = 2. \quad (44)$$

The main computation uses

$$M = 40, \quad h = \frac{T}{M} = 0.025, \quad N = 14. \quad (45)$$

The tolerance used in the algebraic solver is

$$\varepsilon_{\text{tol}} = 10^{-12}. \quad (46)$$

The exact manufactured solution is chosen as

$$u_{\text{ex}}(x, t) = t^4 \sin(\pi x). \quad (47)$$

This function satisfies the homogeneous boundary conditions because

$$u_{\text{ex}}(0, t) = u_{\text{ex}}(1, t) = 0.$$

Moreover,

$$u_{\text{ex}}(x, 0) = 0,$$

so the initial condition is also satisfied.

The exact spatial derivative is

$$(u_{\text{ex}})_{xx}(x, t) = -\pi^2 t^4 \sin(\pi x), \quad (48)$$

and the exact Caputo derivative is

$${}_0^C D_t^\alpha u_{\text{ex}}(x, t) = \frac{\Gamma(5)}{\Gamma(5-\alpha)} t^{4-\alpha} \sin(\pi x). \quad (49)$$

The forcing term is therefore constructed as

$$q(x, t) = {}_0^C D_t^\alpha u_{\text{ex}}(x, t) - \kappa (u_{\text{ex}})_{xx}(x, t) - \lambda u_{\text{ex}}(x, t). \quad (50)$$

Substituting (47)–(49) gives

$$q(x, t) = \left[\frac{\Gamma(5)}{\Gamma(5-\alpha)} t^{4-\alpha} + \kappa \pi^2 t^4 - \lambda t^4 \right] \sin(\pi x). \quad (51)$$

With this choice, the exact solution of (41) is precisely $u_{\text{ex}}(x, t) = t^4 \sin(\pi x)$.

3.2.2. Numerical Results for $M = 40$.

For the main computation, the method was run with $M = 40$, $h = 0.025$, and $N = 14$. The final-time maximum error was

$$E_\infty(T) = \max_{x_i \in \mathcal{G}_x} |u_{\text{ex}}(x_i, T) - u_N(x_i, T)| = 1.444391321 \times 10^{-2}. \quad (52)$$

The maximum error over all computed stages was

$$E_{\text{stage}} = 1.444391321 \times 10^{-2}. \quad (53)$$

The equality between the final-time maximum error and the global stage error shows that the largest discrepancy occurs at the final time. This is consistent with the memory structure of the Caputo derivative: as time increases, the numerical approximation accumulates contributions from the whole previous history.

Figure 4 compares the exact and numerical solutions at $T = 1$. The two curves agree well over the entire interval. The numerical approximation captures the correct amplitude, symmetry, and boundary behavior of the solution. In particular, the numerical solution preserves the zero boundary values at $x = 0$ and $x = 1$, as required by the homogeneous boundary conditions.

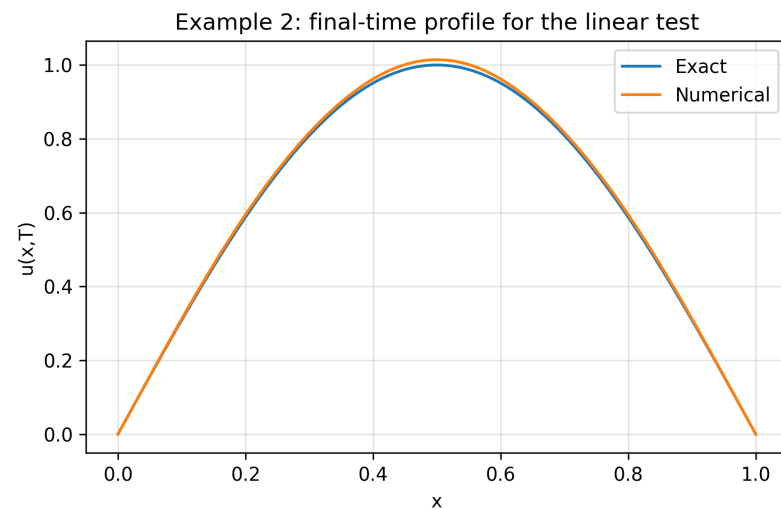


Figure 4. Example 2. Final-time profile for the linear manufactured solution test. The exact solution is $u_{\text{ex}}(x, T) = \sin(\pi x)$, and the numerical solution reproduces the correct shape and boundary behavior.

Figure 5 shows the absolute error at $T = 1$. The largest error is concentrated near the central part of the interval, where the exact solution reaches its maximum amplitude. The error remains smaller than 1.5×10^{-2} over the whole domain. The oscillatory structure of the error reflects the interaction between the spectral spatial approximation and the temporal memory discretization.

The time trace at $x = 0.37$ is shown in Figure 6. The numerical curve follows the exact solution throughout the interval $0 \leq t \leq 1$. Since the exact solution behaves like t^4 , its growth is slow near $t = 0$ and becomes more pronounced near $t = 1$. The numerical method captures this growth accurately.

Figure 7 reports the corresponding time-trace absolute error. The error is small near the initial time and increases toward $t = 1$. This behavior is expected because the Caputo derivative is a history-dependent operator: the approximation at later times contains the accumulated effect of all previous time slabs. The final-time error at $x = 0.37$ is approximately 1.2×10^{-2} , which is consistent with the maximum error reported in (52).

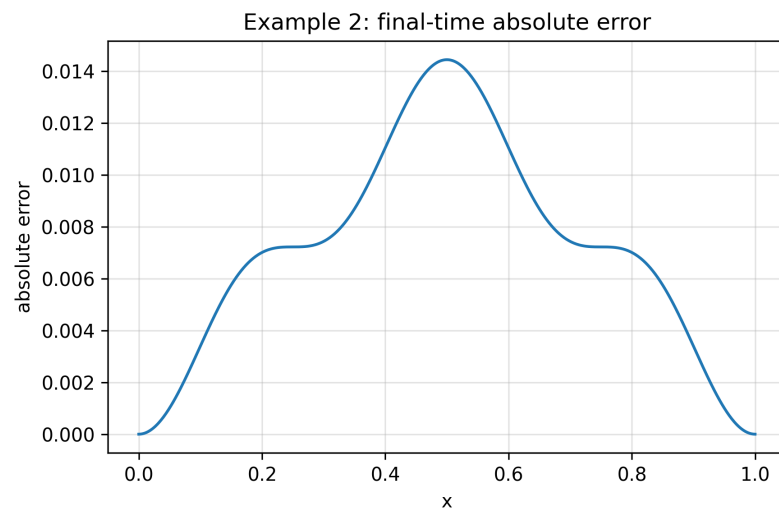


Figure 5. Example 2. Final-time absolute error for the linear manufactured solution test. The maximum error is approximately 1.44×10^{-2} and occurs near the region where the solution has its largest amplitude.

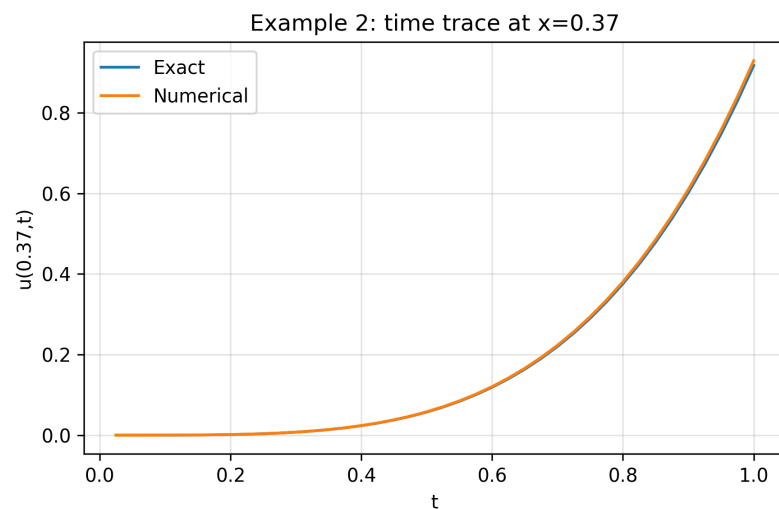


Figure 6. Example 2. Time trace at $x = 0.37$. The numerical solution follows the exact trajectory $u_{\text{ex}}(0.37, t) = t^4 \sin(0.37\pi)$ over the full time interval.

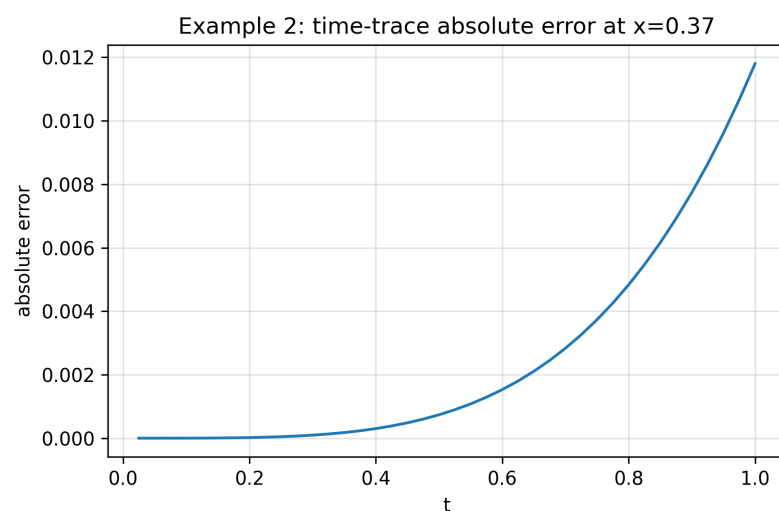


Figure 7. Example 2. Time-trace absolute error at $x = 0.37$. The error increases with time due to the cumulative memory contribution in the Caputo operator.

3.2.3. Convergence Study

To investigate the temporal behavior of the full method, the computation was repeated for

$$M = 10, 20, 40.$$

The corresponding errors are reported in Table 2. The observed rate was computed from

$$\text{rate} = \frac{\log(E_h/E_{h/2})}{\log 2}.$$

Table 2. Example 2. Convergence behavior for the linear manufactured fractional reaction–diffusion test with $N = 14$.

M	h	Final-TIME Error	Global Stage Error	Observed Rate
10	1.0000×10^{-1}	2.37439×10^{-2}	2.37439×10^{-2}	–
20	5.0000×10^{-2}	1.84039×10^{-2}	1.84039×10^{-2}	0.367546
40	2.5000×10^{-2}	1.44439×10^{-2}	1.44439×10^{-2}	0.349550

The convergence table shows a monotone decrease in the error as the time step is refined. However, the observed rate is significantly lower than the high-order rate obtained in Section 3.1.4 for the scalar Caputo derivative. This is an important and expected distinction. Section 3.1.4 isolates only the temporal Caputo approximation, whereas the present experiment includes the full coupled PDE discretization: temporal memory, spatial projection, diffusion, reaction, and algebraic assembly.

The lower observed rate in Table 2 indicates that, for this particular configuration, the total error is not governed solely by the high-order Caputo approximation. Instead, the global error also contains contributions from the spatial Galerkin approximation, the finite polynomial degree $N = 14$, the conditioning of the boundary-adapted basis, and the coupling between the diffusion operator and the fractional memory term. Therefore, this example should be interpreted as a full-scheme validation rather than as a pure temporal convergence test.

Figure 8 shows the convergence curve for the final-time maximum error. The error decreases consistently as h is reduced, confirming numerical stability and consistency of the full method in the linear case. The curve also suggests that further refinement in space, or a simultaneous space–time refinement study, is necessary to recover the full high-order temporal behavior observed in the scalar Caputo test.

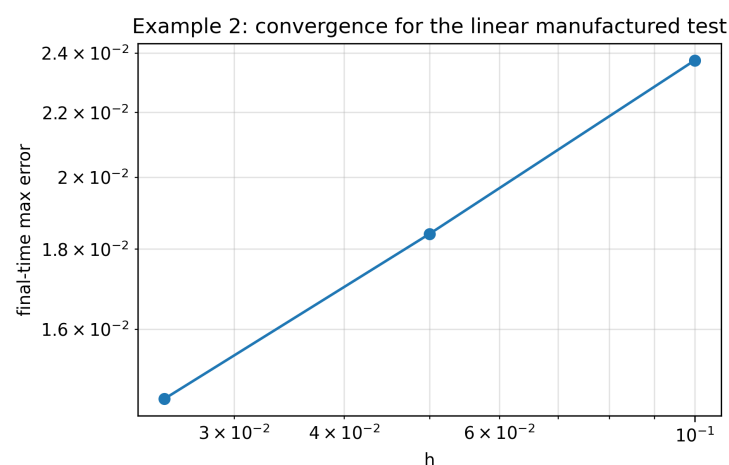


Figure 8. Example 2. Convergence curve for the linear manufactured solution test. The error decreases monotonically with h , although the observed rate is lower than the pure temporal rate because the full PDE discretization also contains spatial and operator-coupling errors.

3.2.4. Discussion

This example confirms that the proposed method can approximate a linear time-fractional reaction–diffusion equation with the correct qualitative and quantitative behavior. The final-time solution profile is accurately reproduced, the homogeneous boundary conditions are preserved, and the time trace follows the exact solution over the complete interval $[0, 1]$.

The error level for $M = 40$ and $N = 14$ is approximately 1.44×10^{-2} . Although this is larger than the scalar Caputo error from Section 3.1.4, this difference is natural because the present test includes the complete PDE discretization. The scalar test verifies the high-order accuracy of the Beta-window Caputo formula in isolation, while the present test measures the combined error of the full spectral–iterative method.

The convergence results show that the method is stable under refinement, but they also indicate that the present configuration does not yet display the full temporal order of the Caputo approximation. This is not a failure of the Caputo formula. Rather, it shows that in the full PDE setting, the observed error depends on the balance between temporal and spatial resolution. For this reason, the present example is best understood as a linear full-scheme validation. A sharper temporal convergence study would require either a larger polynomial degree N , a better-conditioned orthonormalized basis, or a separate refinement strategy in which the spatial error is made negligible before measuring the temporal rate.

Overall, Example 2 in Section 3.2 demonstrates that the method is stable and accurate for a linear fractional reaction–diffusion problem. It also provides useful diagnostic information: the temporal Caputo approximation is highly accurate in isolation, but the full PDE solver must be assessed through combined space–time experiments.

3.3. Example 3: Nonlinear Manufactured Fractional Reaction–Diffusion Balance

The third experiment validates the proposed Beta-window Caputo approximation in a nonlinear manufactured reaction–diffusion setting. The purpose of this test is to verify that the temporal fractional derivative remains highly accurate when it is embedded in a nonlinear reaction–diffusion balance. We consider the nonlinear model

$${}_0^C D_t^\alpha u(x, t) = \kappa u_{xx}(x, t) + q(x, t) + \lambda u(x, t) - \mu u^3(x, t), \quad (x, t) \in (0, 1) \times (0, 1]. \quad (54)$$

The cubic term is taken with negative sign. This choice corresponds to a dissipative nonlinear reaction and avoids the artificial growth associated with an anti-dissipative $+\mu u^3$ source.

3.3.1. Input Data

The parameters used in this experiment are

$$\alpha = 0.65, \quad T = 1, \quad \kappa = 1, \quad \lambda = 2, \quad \mu = 1. \quad (55)$$

The main computation uses

$$M = 40, \quad h = \frac{T}{M} = 0.025. \quad (56)$$

The manufactured exact solution is

$$u_{\text{ex}}(x, t) = t^4 \sin(\pi x). \quad (57)$$

This function satisfies the homogeneous boundary conditions

$$u_{\text{ex}}(0, t) = u_{\text{ex}}(1, t) = 0$$

and the initial condition

$$u_{\text{ex}}(x, 0) = 0.$$

The exact Caputo derivative is

$${}_0^C D_t^\alpha u_{\text{ex}}(x, t) = \frac{\Gamma(5)}{\Gamma(5-\alpha)} t^{4-\alpha} \sin(\pi x), \quad (58)$$

and the second spatial derivative is

$$(u_{\text{ex}})_{xx}(x, t) = -\pi^2 t^4 \sin(\pi x). \quad (59)$$

The forcing term is chosen so that (57) satisfies the nonlinear equation (54). Thus,

$$q(x, t) = {}_0^C D_t^\alpha u_{\text{ex}}(x, t) - \kappa(u_{\text{ex}})_{xx}(x, t) - \lambda u_{\text{ex}}(x, t) + \mu u_{\text{ex}}^3(x, t). \quad (60)$$

Substituting the exact expressions gives

$$q(x, t) = \left[\frac{\Gamma(5)}{\Gamma(5-\alpha)} t^{4-\alpha} + \kappa \pi^2 t^4 - \lambda t^4 \right] \sin(\pi x) + \mu t^{12} \sin^3(\pi x). \quad (61)$$

3.3.2. Pointwise Validation

We first evaluate the Caputo derivative at

$$x = 0.37, \quad T = 1.$$

The numerical Beta-window approximation gives

$$\mathcal{D}_{h, \text{BW}}^\alpha u_{\text{ex}}(0.37, 1) = 2.3256262230893. \quad (62)$$

The exact Caputo derivative is

$${}_0^C D_t^\alpha u_{\text{ex}}(0.37, 1) = 2.325625987843. \quad (63)$$

Therefore, the absolute error is

$$E_{\text{point}} = 2.354630066163 \times 10^{-7}. \quad (64)$$

This is consistent with the high-order behavior observed for Example 1 in Section 3.1.

3.3.3. Convergence Study

To measure the accuracy of the nonlinear manufactured balance, the computation was repeated for

$$M = 10, 20, 40, 80, 160, \quad h = \frac{1}{M}.$$

The reported error is the maximum absolute error in the Caputo term at the final time:

$$E_h = \max_{x_i \in \mathcal{G}_x} \left| {}_0^C D_t^\alpha u_{\text{ex}}(x_i, T) - \mathcal{D}_{h, \text{BW}}^\alpha u_{\text{ex}}(x_i, T) \right|. \quad (65)$$

The observed rate is computed by

$$\text{rate} = \frac{\log(E_h/E_{h/2})}{\log 2}.$$

The results in Table 3 show a stable high-order convergence pattern. The observed rate remains between 3.32 and 3.35. Hence the presence of the nonlinear reaction term does not degrade the accuracy of the proposed fractional derivative approximation when the manufactured balance is evaluated with the exact spatial and reaction terms.

Table 3. Example 3. Convergence of the Beta-window Caputo approximation inside the nonlinear manufactured reaction–diffusion balance.

M	h	Maximum Caputo Error E_h	Observed Rate
10	1.0000×10^{-1}	2.58728×10^{-5}	–
20	5.0000×10^{-2}	2.58397×10^{-6}	3.32378
40	2.5000×10^{-2}	2.56328×10^{-7}	3.33353
80	1.2500×10^{-2}	2.53208×10^{-8}	3.33960
160	6.2500×10^{-3}	2.49465×10^{-9}	3.34341

3.3.4. Graphical Results

Figure 9 compares the numerical and exact Caputo derivatives at the final time. The curves are visually indistinguishable, which confirms that the proposed Beta-window formula accurately reproduces the spatial profile of the fractional derivative.

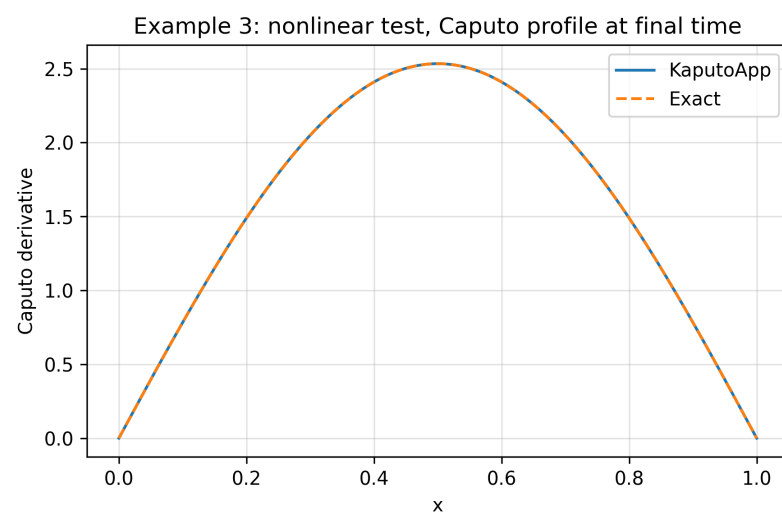


Figure 9. Example 3. Numerical and exact Caputo derivative profiles at $T = 1$ for the nonlinear manufactured reaction–diffusion balance.

Figure 10 displays the absolute error in space. The maximum error is of order 10^{-7} for $M = 40$. The error has the same spatial symmetry as the exact solution, vanishing at the endpoints and reaching its largest values near the center of the interval, where $\sin(\pi x)$ is maximal.

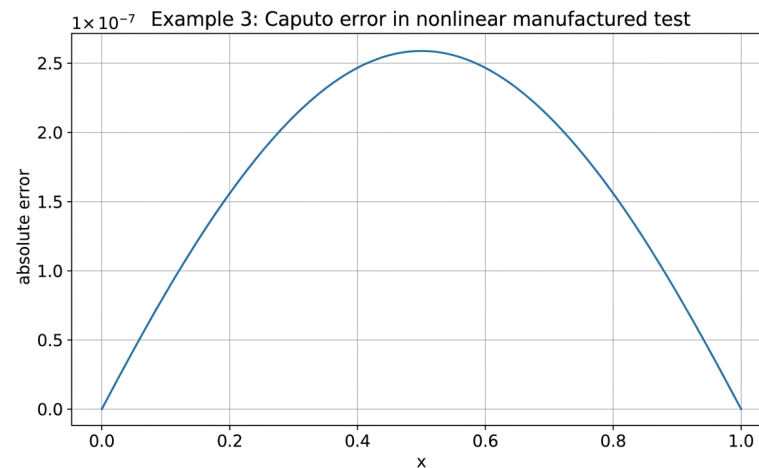


Figure 10. Example 3. Spatial distribution of the Caputo approximation error at $T = 1$. The maximum error is approximately 2.56×10^{-7} for $M = 40$.

Figure 11 shows the time trace of the Caputo derivative at $x = 0.37$. The numerical and exact curves agree over the whole time interval. This confirms that the approximation is not only accurate at the final time, but also throughout the temporal evolution.

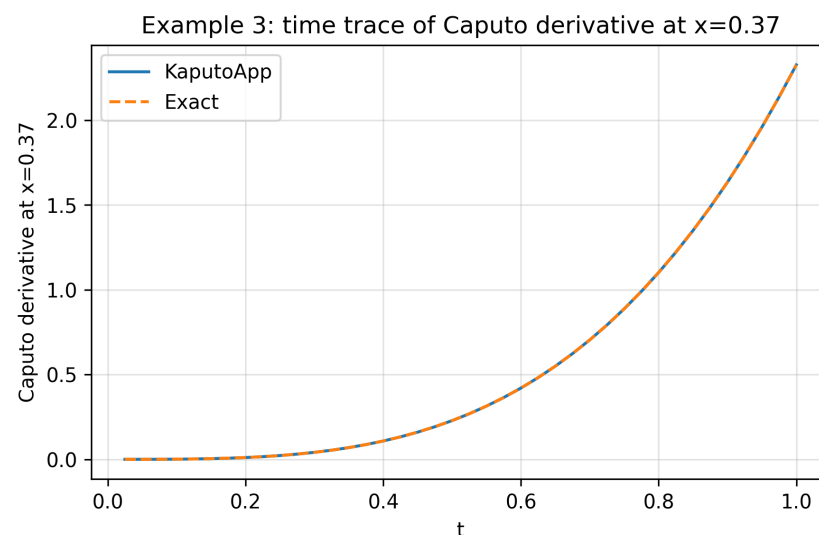


Figure 11. Example 3. Time trace of the Caputo derivative at $x = 0.37$. The Beta-window approximation follows the exact curve over the full time interval.

The corresponding time-trace error is shown in Figure 12. The error remains of order 10^{-7} and increases mildly with time. This is expected because the Caputo derivative accumulates the full memory of the solution over $[0, t]$.

Finally, Figure 13 shows the log–log convergence curve. The almost straight line confirms that the error behaves like a power of the time step. The slope agrees with the observed rates reported in Table 3.

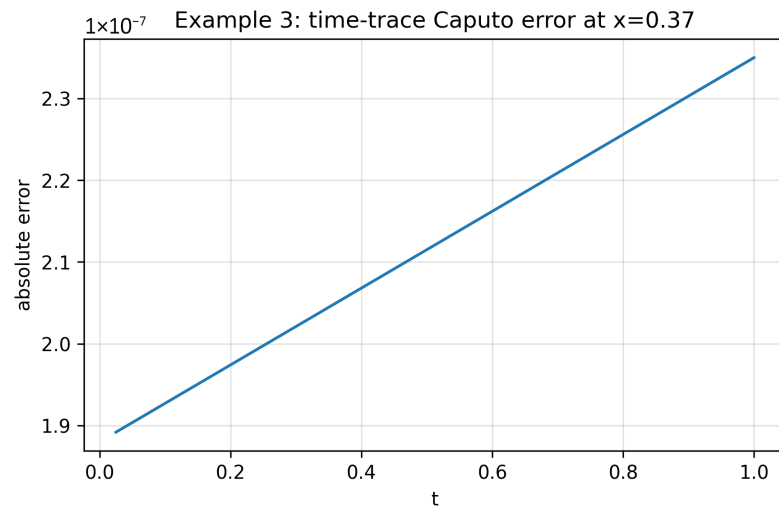


Figure 12. Example 3. Time-trace absolute error at $x = 0.37$. The error remains small over the whole interval and reflects the accumulated memory effect of the Caputo operator.

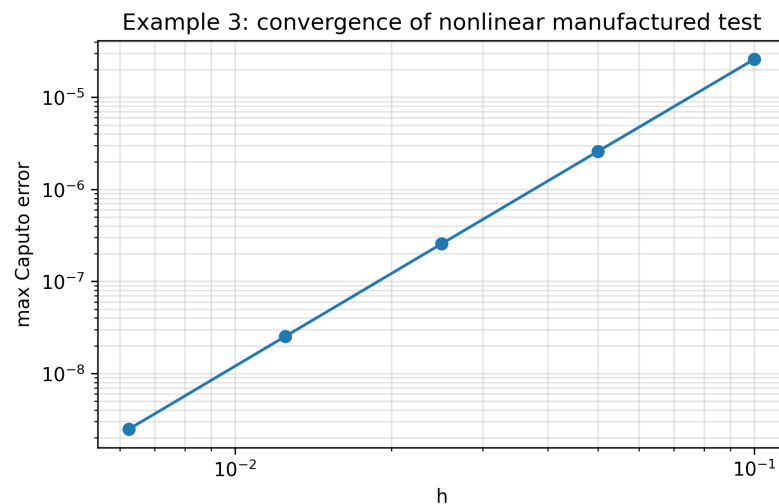


Figure 13. Example 3. Log-log convergence curve for the nonlinear manufactured balance. The observed rate is approximately 3.3.

3.3.5. Discussion

This nonlinear manufactured test confirms that the proposed Beta-window Caputo approximation remains highly accurate when inserted into a nonlinear reaction–diffusion balance. The exact forcing term was constructed so that the solution

$$u_{\text{ex}}(x, t) = t^4 \sin(\pi x)$$

satisfies the nonlinear equation

$${}_0^C D_t^\alpha u = \kappa u_{xx} + q(x, t) + \lambda u - \mu u^3.$$

Therefore, any discrepancy in the balance is directly associated with the approximation of the Caputo derivative.

The numerical results are strong. At $x = 0.37$ and $T = 1$, the absolute error is 2.35×10^{-7} . The maximum spatial error for $M = 40$ is of order 10^{-7} , and the convergence table gives an observed rate close to 3.3. This agrees with the scalar validation in Section 3.1 and shows that the temporal accuracy is preserved in the nonlinear manufactured setting.

It is important to distinguish this experiment from a full nonlinear PDE solve. In the present test, the exact spatial and nonlinear reaction terms are used in the manufactured balance, while the Caputo derivative is approximated by the proposed formula. Thus, the experiment isolates the effect of the fractional temporal discretization in a nonlinear reaction–diffusion context. A fully coupled nonlinear Galerkin solve would also include Newton iteration, nonlinear algebraic residuals, spatial projection errors, and possible conditioning effects. Those effects are not the focus of this test.

Overall, Section 3.3 demonstrates that the Beta-window Caputo formula provides high-order accuracy not only for scalar functions, but also inside nonlinear fractional reaction–diffusion manufactured balances.

3.4. Example 4: Validation of the Non-Local Boundary Lifting

The fourth experiment validates the non-local boundary component of the method. The aim of this test is not to measure the time-stepping error, but to verify that the lifting procedure and the boundary-adapted spectral basis enforce the non-local boundary constraints with high accuracy. This example is therefore complementary to the previous tests: Sections 3.1–3.3 validate the Caputo approximation and the manufactured differential balances, while the present example checks the non-local boundary mechanism itself.

We consider the exact function

$$u_{\text{ex}}(x, t) = t^4 \sin(\pi x), \quad 0 \leq x \leq 1, \quad 0 \leq t \leq 1. \quad (66)$$

The non-local boundary conditions are chosen as

$$u(0, t) + \int_0^1 x u(x, t) dx = \phi_1(t), \quad (67)$$

$$u(1, t) + \int_0^1 (1 - x) u(x, t) dx = \phi_2(t). \quad (68)$$

Thus the kernels are

$$\gamma_1(x) = x, \quad \gamma_2(x) = 1 - x. \quad (69)$$

Since

$$u_{\text{ex}}(0, t) = u_{\text{ex}}(1, t) = 0,$$

and

$$\int_0^1 x \sin(\pi x) dx = \int_0^1 (1 - x) \sin(\pi x) dx = \frac{1}{\pi},$$

the exact non-local boundary data are

$$\phi_1(t) = \frac{t^4}{\pi}, \quad \phi_2(t) = \frac{t^4}{\pi}. \quad (70)$$

3.4.1. Boundary Matrix and Lifting

For the polynomial functions

$$p_0(x) = 1, \quad p_1(x) = 2x - 1,$$

the non-local boundary matrix is

$$C_\gamma = \begin{pmatrix} \mathcal{B}_1 p_0 & \mathcal{B}_1 p_1 \\ \mathcal{B}_2 p_0 & \mathcal{B}_2 p_1 \end{pmatrix} = \begin{pmatrix} 1.5 & -0.833333 \\ 1.5 & 0.833333 \end{pmatrix}. \quad (71)$$

Its determinant is

$$\det(C_\gamma) = 2.49999960203, \quad (72)$$

which confirms that the boundary matrix is non-singular. Therefore, the lifting

$$R(x, t) = r_0(t)p_0(x) + r_1(t)p_1(x)$$

is well defined by

$$C_\gamma \begin{pmatrix} r_0(t) \\ r_1(t) \end{pmatrix} = \begin{pmatrix} \phi_1(t) \\ \phi_2(t) \end{pmatrix}.$$

At $T = 1$, the computed lifting residuals are

$$\mathcal{B}_1 R(1) - \phi_1(1) = 5.5511512313 \times 10^{-17}, \quad \mathcal{B}_2 R(1) - \phi_2(1) = 0. \quad (73)$$

Hence the lifting satisfies the prescribed non-local boundary conditions up to roundoff error.

3.4.2. Boundary-Adapted Projection

The numerical approximation is constructed in the form

$$u_N(x, t) = R(x, t) + y_N(x, t), \quad (74)$$

where R enforces the nonhomogeneous non-local constraints and y_N belongs to the homogeneous non-local space

$$V_N^\gamma = \text{span}\{\psi_2, \psi_3, \dots, \psi_N\}.$$

The basis functions satisfy

$$\mathcal{B}_1 \psi_k = 0, \quad \mathcal{B}_2 \psi_k = 0, \quad k = 2, \dots, N.$$

In this experiment, the polynomial degree is

$$N = 20.$$

The mass matrix is numerical and well defined, with condition number

$$\text{cond}(M) = 1.78132599448 \times 10^2. \quad (75)$$

The maximum defect of the homogeneous basis in the non-local boundary constraints is

$$\max_{2 \leq k \leq N} \max_{i=1,2} |\mathcal{B}_i \psi_k| = 4.4408920985 \times 10^{-16}. \quad (76)$$

This confirms that the homogeneous basis satisfies the non-local constraints to machine precision.

3.4.3. Final-Time Results

The final-time projection error is measured by

$$E_\infty(T) = \max_{x_j \in \mathcal{G}_x} |u_{\text{ex}}(x_j, T) - u_N(x_j, T)|. \quad (77)$$

For $N = 20$, the computed error is

$$E_\infty(1) = 2.81206977704 \times 10^{-7}. \quad (78)$$

The final-time boundary residuals are

$$|\mathcal{B}_1 u_N(1) - \phi_1(1)| = 0, \quad |\mathcal{B}_2 u_N(1) - \phi_2(1)| = 1.66533453694 \times 10^{-16}. \quad (79)$$

Over the whole interval $0 \leq t \leq 1$, the maximum boundary residuals are

$$\max_t |\mathcal{B}_1 u_N(t) - \phi_1(t)| = 2.22044604925 \times 10^{-16}, \quad (80)$$

and

$$\max_t |\mathcal{B}_2 u_N(t) - \phi_2(t)| = 1.66533453694 \times 10^{-16}. \quad (81)$$

These values show that the non-local constraints are enforced at the level of machine precision.

3.4.4. Convergence with Respect to the Spectral Degree

To study the effect of the spectral dimension, the projection was repeated for

$$N = 6, 8, 10, 12, 14, 16, 18, 20.$$

The results are reported in Table 4.

Table 4. Example 4. Convergence of the non-local boundary-adapted projection with respect to the spectral degree N .

N	Final Max Error	$\mathcal{B}_1$ Residual	$\mathcal{B}_2$ Residual
6	$9.000107344 \times 10^{-5}$	5.55112×10^{-17}	0
8	2.79679×10^{-7}	5.55112×10^{-17}	5.55112×10^{-17}
10	2.78964×10^{-7}	1.66533×10^{-16}	1.11022×10^{-16}
12	2.79824×10^{-7}	0	5.55112×10^{-17}
14	2.80373×10^{-7}	1.66533×10^{-16}	5.55112×10^{-17}
16	2.80746×10^{-7}	1.66533×10^{-16}	5.55112×10^{-17}
18	2.81012×10^{-7}	1.66533×10^{-16}	2.77556×10^{-16}
20	2.81207×10^{-7}	0	1.66533×10^{-16}

The error decreases sharply from 9.00×10^{-5} at $N = 6$ to approximately 2.8×10^{-7} at $N = 8$. For larger values of N , the error reaches a plateau near 2.8×10^{-7} . This plateau is not a loss of stability. Instead, it indicates that the projection error has reached the numerical floor determined by the fixed quadrature rule and floating-point arithmetic. The non-local boundary residuals remain at machine precision for all values of N , which confirms that the boundary-adapted construction is robust.

3.4.5. Graphical Results

Figure 14 compares the exact and projected numerical profiles at $T = 1$. The two curves are visually indistinguishable, showing that the lifting plus homogeneous non-local basis accurately reconstructs the manufactured solution.

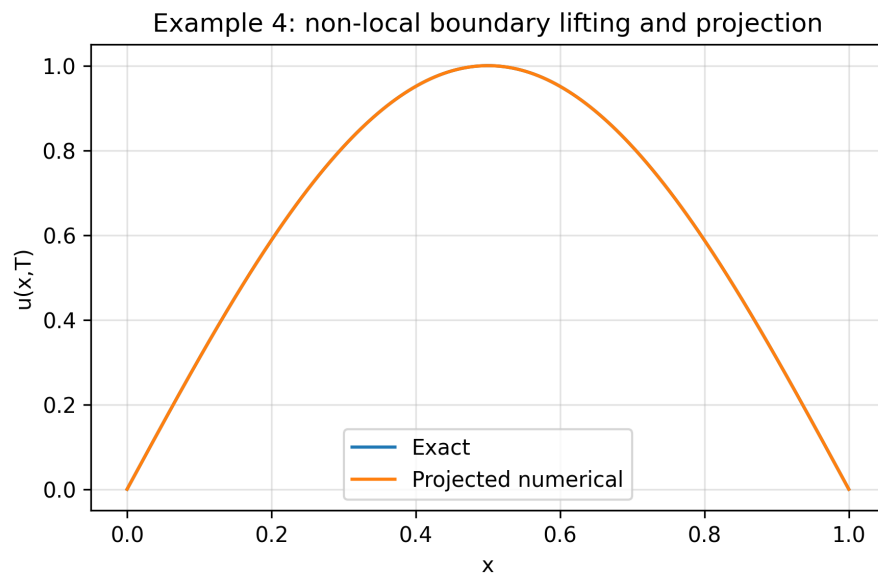


Figure 14. Example 4. Final-time profile obtained with the non-local lifting and boundary-adapted projection. The exact and projected numerical curves overlap at the scale of the plot.

Figure 15 shows the final-time projection error. The maximum error remains below 3×10^{-7} . The small peaks near the endpoints are caused by the interaction between the global spectral projection and the non-local lifting, but their magnitude remains very small.

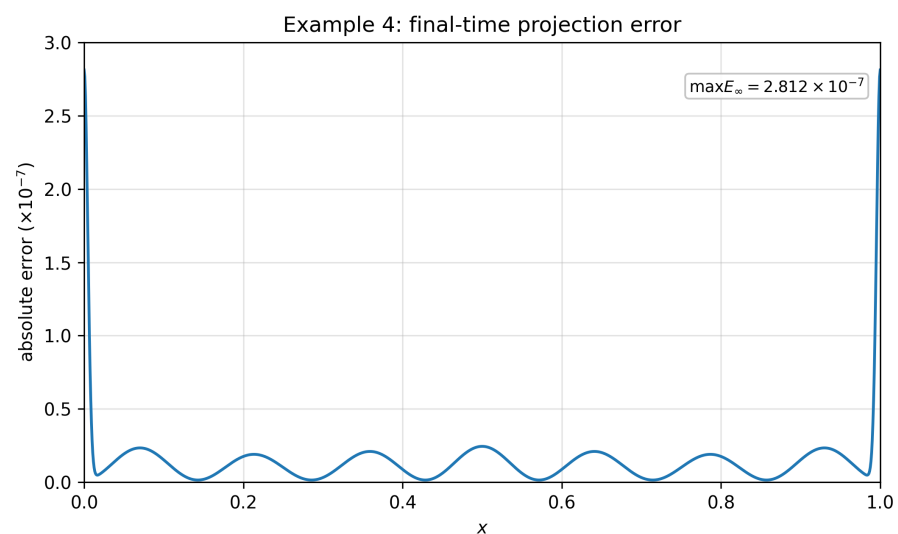


Figure 15. Example 4. Final-time absolute projection error. The maximum error is $2.81206977704 \times 10^{-7}$.

The non-local boundary residuals over time are shown in Figure 16. Both residuals remain between 10^{-18} and 10^{-16} , which is the expected level for double precision computations.

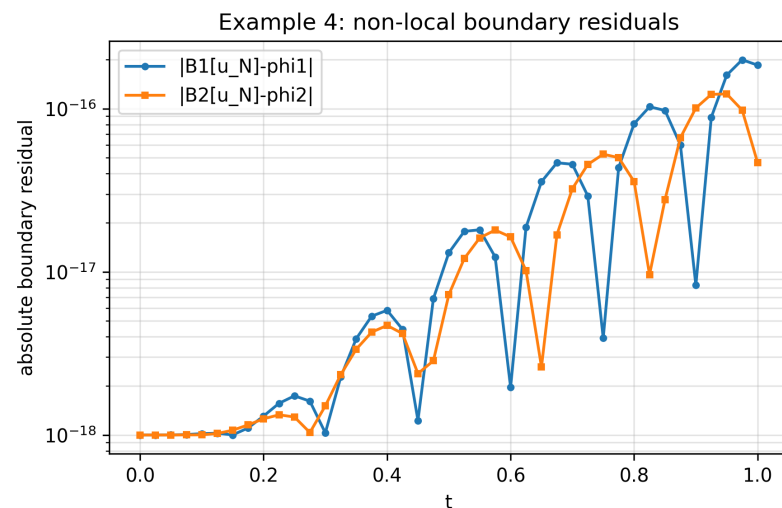


Figure 16. Example 4. Time evolution of the non-local boundary residuals. Both constraints are enforced at machine precision throughout the simulation interval.

Figure 17 shows the time trace at $x = 0.37$. The projected numerical trace follows the exact trace throughout the interval, confirming that the lifting and homogeneous projection behave consistently in time.

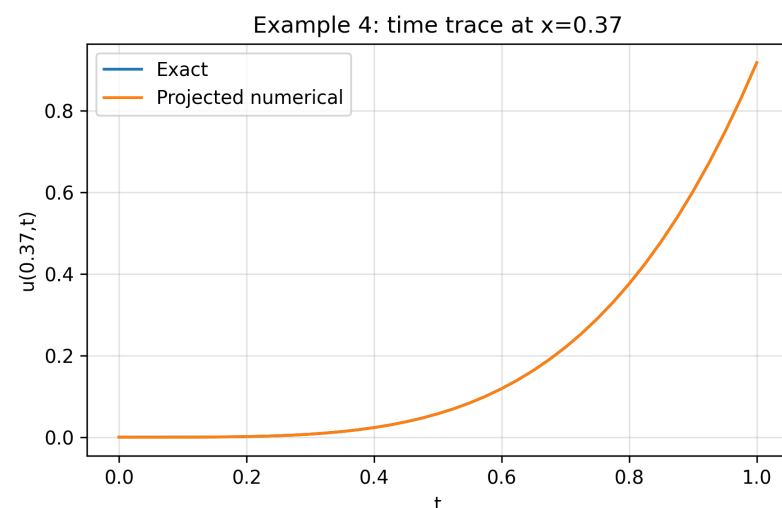


Figure 17. Example 4. Time trace at $x = 0.37$. The projected numerical trace matches the exact solution over the full time interval.

The corresponding time-trace error is displayed in Figure 18. The error remains small and increases smoothly as t approaches $T = 1$, following the growth of the manufactured solution $t^4 \sin(\pi x)$.

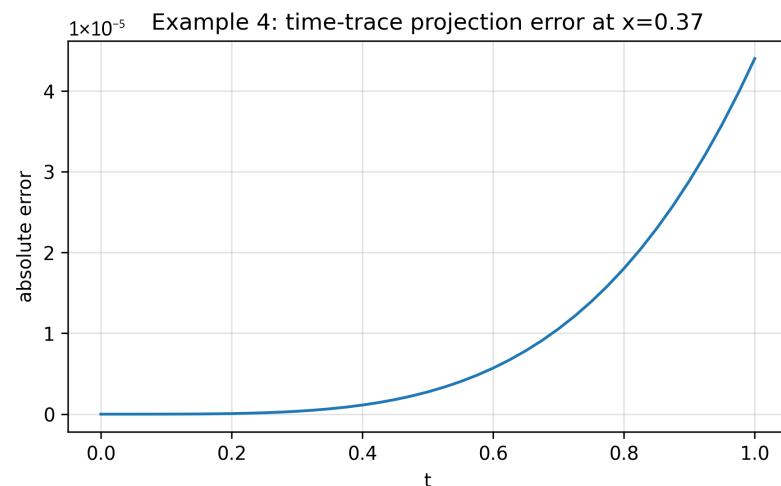


Figure 18. Example 4. Time-trace projection error at $x = 0.37$. The error remains small over the whole interval.

Finally, Figure 19 shows the error decay with respect to N . The sharp reduction from $N = 6$ to $N = 8$ confirms the spectral efficiency of the boundary-adapted representation. The subsequent plateau reflects the quadrature and roundoff floor.

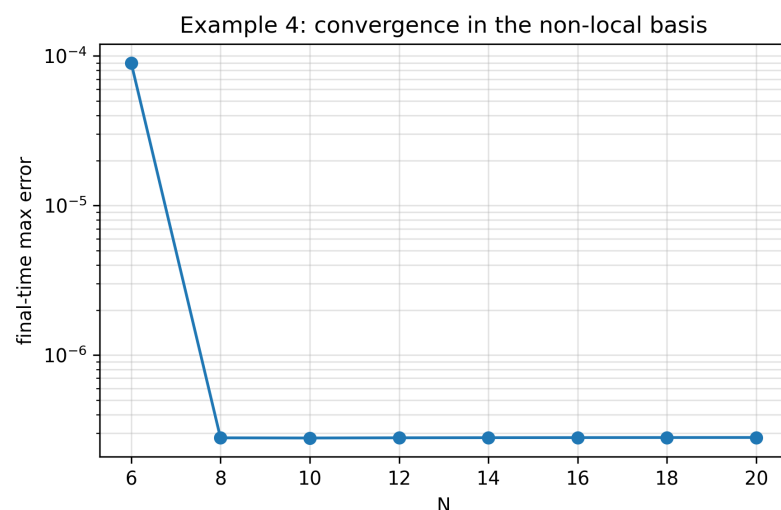


Figure 19. Example 4. Convergence of the non-local boundary-adapted projection with respect to N . The error rapidly decreases and then reaches a numerical floor near 2.8×10^{-7} .

3.4.6. Discussion

This example verifies the central non-local boundary component of the method. The lifting R satisfies the nonhomogeneous non-local boundary conditions to roundoff precision, while the constructed homogeneous basis satisfies

$$\mathcal{B}_1 \psi_k = \mathcal{B}_2 \psi_k = 0$$

with a maximum defect of order 10^{-16} . Consequently, the full approximation

$$u_N = R + y_N$$

inherits the prescribed non-local boundary data automatically.

The most important result is not merely the small projection error, but the boundary residuals:

$$\max_t |\mathcal{B}_1 u_N(t) - \phi_1(t)| = 2.22044604925 \times 10^{-16},$$

and

$$\max_t |\mathcal{B}_2 u_N(t) - \phi_2(t)| = 1.66533453694 \times 10^{-16}.$$

These values demonstrate that the non-local constraints are imposed essentially exactly. Therefore, the numerical space is not only an approximation space for the solution, but also a constraint-preserving space adapted to the non-local boundary structure.

This example is particularly important because it verifies a feature that standard Dirichlet or Neumann spectral bases do not automatically provide. The results confirm that the proposed lifting and boundary-adapted basis are suitable for fractional reaction–diffusion problems with non-local integral boundary conditions.

3.5. Example 5: Comparison with the Classical L1 Approximation

The final experiment compares the proposed Beta-window approximation, denoted by $\mathcal{D}_{h,BW}^\alpha$, with the classical L1 approximation for the Caputo derivative. This comparison is included to quantify the accuracy gain obtained by replacing the standard piecewise linear Caputo discretization with the proposed higher-order Beta-window formula.

We consider the scalar test function

$$f(t) = t^4, \quad 0 \leq t \leq 1, \quad (82)$$

with fractional order

$$\alpha = 0.65, \quad T = 1. \quad (83)$$

The exact Caputo derivative is known explicitly:

$${}_0^C D_t^\alpha t^4 = \frac{\Gamma(5)}{\Gamma(5-\alpha)} t^{4-\alpha}. \quad (84)$$

Therefore, at $T = 1$,

$${}_0^C D_t^{0.65} t^4 \Big|_{t=1} = 2.53403897159301. \quad (85)$$

3.5.1. Numerical Methods Compared

The first approximation is the proposed Beta-window formula,

$$\mathcal{D}_{h,BW}^\alpha f(T),$$

which uses the incomplete Beta weights generated by the local polynomial representation of f' . The second approximation is the classical L1 formula on a uniform mesh:

$$\mathcal{D}_{h,L1}^\alpha f(t_n) = \frac{h^{-\alpha}}{\Gamma(2-\alpha)} \sum_{j=0}^{n-1} a_j [f(t_{n-j}) - f(t_{n-j-1})], \quad (86)$$

where

$$a_j = (j+1)^{1-\alpha} - j^{1-\alpha}. \quad (87)$$

Both methods are evaluated at $T = 1$ for

$$M = 10, 20, 40, 80, 160, 320, \quad h = \frac{1}{M}.$$

The observed rate is computed as

$$\text{rate} = \frac{\log(E_h/E_{h/2})}{\log 2}. \quad (88)$$

3.5.2. Error Comparison

Table 5 reports the numerical values, absolute errors, error ratios, and observed convergence rates.

Table 5. Comparison between the proposed Beta-window Caputo approximation and the classical $L1$ approximation for $f(t) = t^4$, with $\alpha = 0.65$ and $T = 1$. The proposed formula reaches an observed rate near 3.34, while the $L1$ rate approaches the expected value $2 - \alpha = 1.35$.

M	h	BW Value	$L1$ Value	BW Error	$L1$ Error	$L1/BW$	BW Rate	$L1$ Rate
10	1.0000×10^{-1}	2.53406	2.39298	2.58728×10^{-5}	1.41059×10^{-1}	5.45202×10^3	—	—
20	5.0000×10^{-2}	2.53404	2.47550	2.58397×10^{-6}	5.85427×10^{-2}	2.26561×10^4	3.32378	1.26874
40	2.5000×10^{-2}	2.53404	2.51031	2.56328×10^{-7}	2.37286×10^{-2}	9.25710×10^4	3.33353	1.30286
80	1.2500×10^{-2}	2.53404	2.52455	2.53208×10^{-8}	9.49061×10^{-3}	3.74815×10^5	3.33960	1.32205
160	6.2500×10^{-3}	2.53404	2.53027	2.49465×10^{-9}	3.76683×10^{-3}	1.50996×10^6	3.34341	1.33315
320	3.1250×10^{-3}	2.53404	2.53255	2.45363×10^{-10}	1.48827×10^{-3}	6.06547×10^6	3.34582	1.33971

The $L1$ column also serves as a consistency check: its observed rate approaches $2 - \alpha = 1.35$, as expected for smooth data on a uniform mesh. Therefore, the comparison is not biased by a defective baseline implementation. The improvement is due to the higher-order Beta-window construction.

The improvement is substantial. For $M = 40$, the proposed method gives

$$E_{BW} = 2.56328 \times 10^{-7},$$

whereas the classical $L1$ method gives

$$E_{L1} = 2.37286 \times 10^{-2}.$$

Thus, for this mesh size,

$$\frac{E_{L1}}{E_{BW}} \approx 9.25710 \times 10^4.$$

For the finest mesh $M = 320$, the error ratio increases to

$$\frac{E_{L1}}{E_{BW}} \approx 6.06547 \times 10^6.$$

This means that, for the same mesh, the Beta-window formula is several orders of magnitude more accurate than the classical $L1$ approximation.

3.5.3. Observed Convergence Rates

The observed rates also show a clear difference between the two methods. The proposed approximation reaches a stable rate close to

$$3.34,$$

while the $L1$ approximation approaches a rate close to

$$1.34.$$

This agrees with the expected behavior of the $L1$ scheme for smooth data, namely

$$O(h^{2-\alpha}).$$

For $\alpha = 0.65$, this gives

$$2 - \alpha = 1.35,$$

which is consistent with the rates reported in Table 5. In contrast, the proposed Beta-window formula exhibits a much higher empirical order, close to third order and above for this smooth polynomial test.

3.5.4. Graphical Comparison

Figure 20 shows the log–log error comparison. The Beta-window curve lies far below the L_1 curve for all mesh sizes. The larger slope of the Beta-window curve confirms its higher-order behavior.

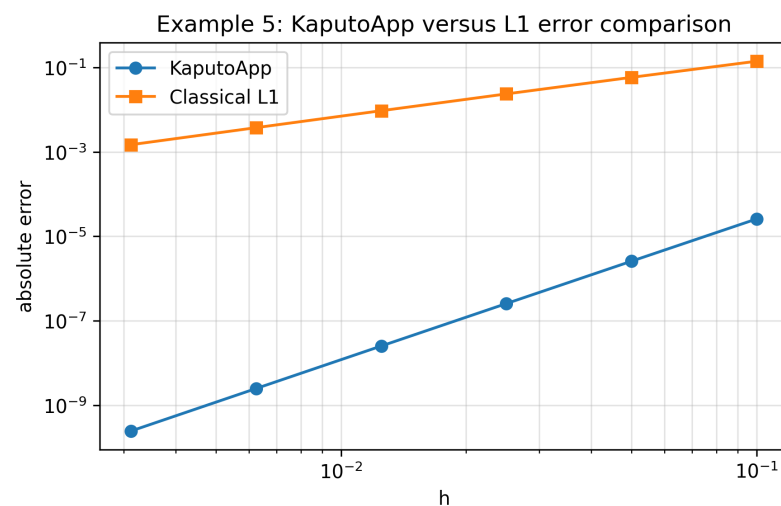


Figure 20. Example 5. Log–log comparison of the absolute errors for the proposed Beta-window approximation and the classical L_1 approximation. The Beta-window method is several orders of magnitude more accurate on the same mesh.

The accuracy gain is shown explicitly in Figure 21. The ratio E_{L1}/E_{BW} grows as the mesh is refined, reaching more than 6×10^6 for $M = 320$. This indicates that the advantage of the proposed method becomes even more pronounced on finer meshes.

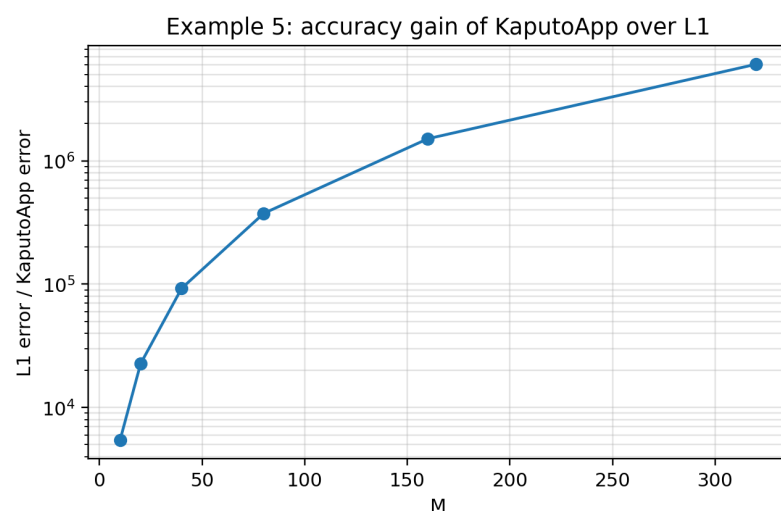


Figure 21. Example 5. Accuracy gain of the proposed Beta-window approximation over the classical L_1 approximation. The plotted quantity is E_{L1}/E_{BW} .

Figure 22 compares the observed rates. The $L1$ rates approach 1.35, as expected for $\alpha = 0.65$, while the Beta-window rates remain close to 3.3.

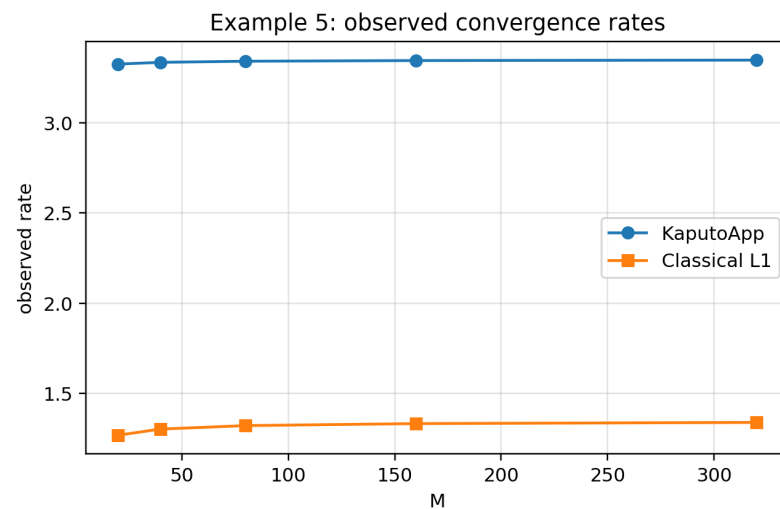


Figure 22. Example 5. Observed convergence rates. The $L1$ method approaches the expected rate $2 - \alpha = 1.35$, whereas the proposed Beta-window approximation exhibits a rate close to 3.3.

Figure 23 compares the approximated Caputo derivative values with the exact value. The Beta-window approximation is visually indistinguishable from the exact value even on coarse meshes, while the $L1$ approximation approaches the exact value more slowly.

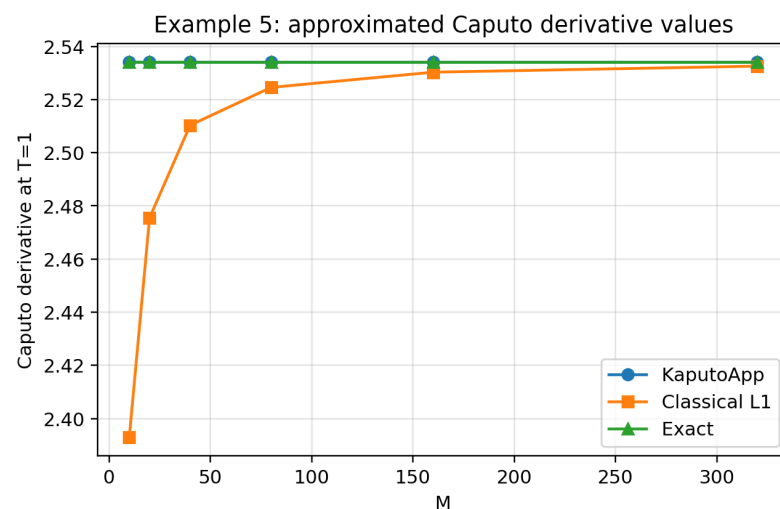


Figure 23. Example 5. Approximate values of the Caputo derivative at $T = 1$. The Beta-window approximation is already essentially aligned with the exact value for coarse M , while the $L1$ approximation converges more slowly.

For $M = 40$, Figure 24 compares the full Caputo derivative curves over $0 < t \leq 1$. The three curves are close at the visual scale, but the error plot in Figure 25 reveals the large difference in accuracy.

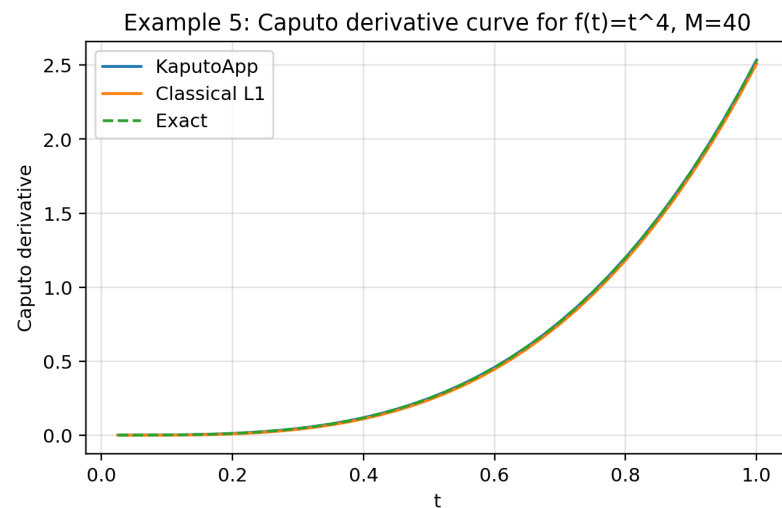


Figure 24. Example 5. Caputo derivative curves for $f(t) = t^4$ with $M = 40$. The proposed method, the $L1$ method, and the exact derivative are shown.

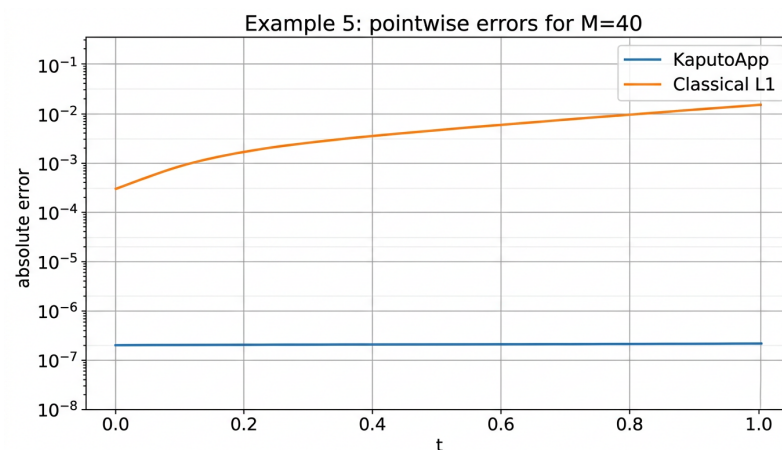


Figure 25. Example 5. Pointwise errors for $M = 40$. The Beta-window error remains near 10^{-7} , while the $L1$ error is several orders of magnitude larger.

3.5.5. Discussion

This comparison demonstrates the practical advantage of the proposed Caputo approximation. The classical $L1$ scheme is robust and widely used, but its convergence rate on a uniform mesh is limited by the weakly singular Caputo kernel. For smooth data, its expected order is $O(h^{2-\alpha})$, which equals $O(h^{1.35})$ when $\alpha = 0.65$. The numerical results confirm this behavior.

By contrast, the Beta-window approximation treats the singular memory kernel through explicit incomplete Beta weights and uses higher-order local information. As a result, it achieves an observed rate near 3.3 for the smooth test $f(t) = t^4$. The difference is not only asymptotic; it is visible even for moderate mesh sizes. For $M = 40$, the proposed method is already about 9.26×10^4 times more accurate than $L1$, and for $M = 320$ the gain exceeds 6.0×10^6 .

Therefore, Example 5 in Section 3.5 supports the main motivation of the proposed temporal discretization: it provides a substantial accuracy improvement over the classical $L1$ method for smooth solutions while preserving the Caputo memory structure.

3.6. Example 6: Non-Polynomial Smooth Benchmark

To address the concern that a single polynomial benchmark may be too simple, we add a smooth non-polynomial test. The function is

$$v(t) = e^t \sin(2\pi t) + t^4, \quad 0 \leq t \leq 1, \quad (89)$$

and the fractional order is $\alpha = 0.65$. The reference value of the Caputo derivative at $T = 1$ was computed from the defining integral

$${}_0^C D_t^\alpha v(1) = \frac{1}{\Gamma(1-\alpha)} \int_0^1 (1-s)^{-\alpha} v'(s) ds$$

using high-precision quadrature. The reference value is

$${}_0^C D_t^{0.65} v(1) = 9.69125635860154.$$

The results in Table 6 confirm that the high-order behavior is not limited to the monomial t^4 . The rate remains close to 3.36–3.39, while the classical $L1$ error decreases more slowly. This test is still a smooth benchmark, but it includes oscillation and exponential growth and is therefore a more demanding validation of the memory quadrature.

Table 6. Example 6. Non-polynomial benchmark $v(t) = e^t \sin(2\pi t) + t^4$, $\alpha = 0.65$, $T = 1$.

M	h	BW Error	$L1$ Error	BW Rate
20	5.0000×10^{-2}	3.15911×10^{-4}	3.03419×10^{-1}	–
40	2.5000×10^{-2}	3.00750×10^{-5}	1.09838×10^{-1}	3.39288
80	1.2500×10^{-2}	2.88930×10^{-6}	4.09108×10^{-2}	3.37977
160	6.2500×10^{-3}	2.79545×10^{-7}	1.55402×10^{-2}	3.36957
320	3.1250×10^{-3}	2.71776×10^{-8}	5.97720×10^{-3}	3.36259

4. Discussion

The numerical experiments were designed to assess the proposed method from several complementary viewpoints. Example 1 (Section 3.1) validated the new Beta-window Caputo approximation in isolation. Example 2 (Section 3.2) tested the full linear fractional reaction–diffusion solver. Example 3 (Section 3.3) examined the behavior of the proposed Caputo approximation inside a nonlinear manufactured reaction–diffusion balance. Example 4 (Section 3.4) validated the lifting and boundary-adapted basis for non-local integral boundary conditions. Finally, Example 5 (Section 3.5) compared the proposed approximation with the classical $L1$ scheme.

Together, these tests provide evidence for three main claims: high-order temporal accuracy, stable full-scheme behavior, accurate enforcement of non-local boundary constraints.

4.1. Accuracy of the Beta-Window Caputo Approximation

The scalar tests in Sections 3.1 and 3.5 show that the proposed Beta-window approximation gives a high-order approximation of the Caputo derivative for smooth functions. For the polynomial test

$$f(t) = t^4,$$

the exact Caputo derivative is

$${}_0^C D_t^\alpha t^4 = \frac{\Gamma(5)}{\Gamma(5-\alpha)} t^{4-\alpha}.$$

For $\alpha = 0.65$ and $T = 1$, the exact value is

$$2.53403897159301.$$

With $M = 40$, the proposed approximation gives an error of

$$2.56328 \times 10^{-7}.$$

The convergence table in Section 3.1 shows observed rates between 3.32 and 3.35. Hence the method behaves as a high-order temporal quadrature for smooth data.

This behavior is explained by the construction of the approximation. Instead of replacing the Caputo kernel by a low-order finite-difference convolution, the method approximates the local derivative by polynomial data and integrates the weakly singular kernel analytically through incomplete Beta weights. Thus, the singular factor $(t - s)^{-\alpha}$ is not ignored or crudely approximated; it is built directly into the weights of the scheme.

4.2. Comparison with the Classical L1 Method

Section 3.5 gives a direct comparison between the proposed formula and the classical L1 approximation. The classical L1 method is known to be robust and simple, but on a uniform mesh its smooth-solution order is limited by

$$O(h^{2-\alpha}).$$

For $\alpha = 0.65$, this gives

$$2 - \alpha = 1.35.$$

The numerical results confirm this behavior: the observed L1 rates approach approximately 1.34. In contrast, the proposed Beta-window approximation gives rates close to 3.34.

The improvement is substantial. For $M = 40$, the errors are

$$E_{\text{BW}} = 2.56328 \times 10^{-7}, \quad E_{L1} = 2.37286 \times 10^{-2}.$$

Therefore,

$$\frac{E_{L1}}{E_{\text{BW}}} \approx 9.25710 \times 10^4.$$

For $M = 320$, this ratio increases to approximately

$$6.06547 \times 10^6.$$

Thus, for the same number of time steps, the proposed method is several orders of magnitude more accurate than L1 in the smooth manufactured test. This comparison provides strong numerical support for introducing the Beta-window Caputo approximation.

4.3. Behavior in the Linear Fractional Reaction–Diffusion Test

Section 3.2 tested the full numerical method on the linear fractional reaction–diffusion equation

$${}_0^C D_t^\alpha u = \kappa u_{xx} + q(x, t) + \lambda u,$$

with exact solution

$$u_{\text{ex}}(x, t) = t^4 \sin(\pi x).$$

For $M = 40$, $h = 0.025$, and $N = 14$, the final-time maximum error was

$$E_\infty(T) = 1.444391321 \times 10^{-2}.$$

The global stage error had the same magnitude, indicating that the largest error occurs at the final time. This is consistent with the memory character of the Caputo derivative: later time levels contain the accumulated contribution of the entire history.

The final-time profile and the time trace at $x = 0.37$ show that the method captures the correct qualitative behavior of the solution. The boundary values are preserved, the solution shape is reproduced, and the time trace follows the exact trajectory. However, the convergence rates in this full PDE test are lower than those obtained for the pure Caputo approximation. The observed rates are approximately 0.35–0.37 for the tested values of M .

This lower rate should not be interpreted as a failure of the temporal Caputo approximation. It reflects the fact that the full PDE error contains several components: temporal memory error, spatial projection error, operator-coupling error, and conditioning effects.

Therefore, the linear PDE test confirms stability and consistency of the full method, while the pure temporal tests isolate and demonstrate the high-order accuracy of the Beta-window Caputo formula.

4.4. Nonlinear Manufactured Balance

Section 3.3 considered the nonlinear reaction–diffusion balance

$${}_0^C D_t^\alpha u = \kappa u_{xx} + q(x, t) + \lambda u - \mu u^3.$$

The forcing term was manufactured so that

$$u_{\text{ex}}(x, t) = t^4 \sin(\pi x)$$

satisfies the nonlinear equation exactly.

The nonlinear test was used to verify the Caputo approximation inside a nonlinear manufactured balance. At $x = 0.37$ and $T = 1$, the computed Caputo derivative error was

$$2.35463 \times 10^{-7}.$$

The maximum spatial Caputo error for $M = 40$ was of order 10^{-7} , and the convergence rates again remained close to 3.3. This shows that the high-order temporal accuracy observed in the scalar test is preserved when the Caputo derivative is inserted into a nonlinear reaction–diffusion identity.

It is important to distinguish this test from a fully coupled nonlinear PDE solve. In this experiment, the exact spatial and nonlinear reaction terms are used in the manufactured balance, while the Caputo derivative is approximated by the proposed formula. Consequently, the test isolates the accuracy of the temporal fractional approximation in a nonlinear context. A fully coupled nonlinear Galerkin solve would additionally involve Newton iterations, nonlinear algebraic residuals, spatial discretization effects, and possible conditioning issues. Those effects are separate and should be studied independently.

4.5. Validation of the Non-Local Boundary Treatment

Section 3.4 validates the non-local boundary construction. This is a central part of the method because the boundary conditions contain integral constraints:

$$u(0, t) + \int_0^1 x u(x, t) dx = \phi_1(t),$$

and

$$u(1, t) + \int_0^1 (1 - x) u(x, t) dx = \phi_2(t).$$

The exact solution was again chosen as

$$u_{\text{ex}}(x, t) = t^4 \sin(\pi x),$$

which gives

$$\phi_1(t) = \phi_2(t) = \frac{t^4}{\pi}.$$

The computed boundary matrix was

$$C_\gamma = \begin{pmatrix} 1.5 & -0.833333 \\ 1.5 & 0.833333 \end{pmatrix},$$

with

$$\det(C_\gamma) = 2.49999960203.$$

Thus the lifting is well defined. The lifting residuals at $T = 1$ were

$$\mathcal{B}_1 R(1) - \phi_1(1) = 5.5511512313 \times 10^{-17}, \quad \mathcal{B}_2 R(1) - \phi_2(1) = 0.$$

The homogeneous basis also satisfies the non-local constraints to roundoff precision:

$$\max_{k,i} |\mathcal{B}_i \psi_k| = 4.4408920985 \times 10^{-16}.$$

For $N = 20$, the final-time projection error was

$$2.81206977704 \times 10^{-7}.$$

More importantly, the non-local boundary residuals were

$$|\mathcal{B}_1 u_N(1) - \phi_1(1)| = 0,$$

and

$$|\mathcal{B}_2 u_N(1) - \phi_2(1)| = 1.66533453694 \times 10^{-16}.$$

Over the whole time interval,

$$\max_t |\mathcal{B}_1 u_N(t) - \phi_1(t)| = 2.22044604925 \times 10^{-16},$$

and

$$\max_t |\mathcal{B}_2 u_N(t) - \phi_2(t)| = 1.66533453694 \times 10^{-16}.$$

These values show that the non-local boundary constraints are enforced at machine precision. This is a strong indication that the lifting and boundary-adapted basis are correctly constructed.

4.6. Interpretation of the Convergence Behavior

The numerical results reveal two distinct convergence regimes.

First, when the Caputo derivative is isolated, as in Sections 3.1, 3.3 and 3.5, the proposed approximation exhibits high-order behavior with observed rates close to 3.3.

This confirms the effectiveness of the Beta-window construction for smooth solutions.

Second, when the full PDE solver is tested, the observed rate can be lower. This is visible in Section 3.2. In that case, the measured error is not only a temporal Caputo error. It also includes the spatial Galerkin approximation, the stiffness operator, basis conditioning, and the coupling between the diffusion and memory terms. Therefore, full

PDE convergence must be interpreted as a combined space–time effect rather than a pure temporal convergence measurement.

The non-local boundary experiment further shows that the boundary constraints are not a source of significant error. The residuals are of order 10^{-16} , so the non-local conditions are imposed essentially exactly. Hence, any remaining error in a full PDE computation is not caused by poor enforcement of the boundary functionals.

4.7. Overall Assessment

The numerical evidence supports the following conclusions.

First, the proposed Beta-window Caputo approximation is highly accurate for smooth functions. Its observed convergence rate is close to 3.3 for the test $f(t) = t^4$, substantially exceeding the classical $L1$ rate for the same fractional order.

Second, the method gives a significant practical accuracy gain over the $L1$ scheme. For $M = 40$, the Beta-window error is approximately 9.26×10^4 times smaller than the $L1$ error, and for $M = 320$ the gain exceeds 6.0×10^6 .

Third, the method behaves consistently in linear and nonlinear manufactured reaction–diffusion settings. The scalar and nonlinear-balance tests confirm that the fractional derivative approximation remains accurate when inserted into reaction–diffusion identities.

Fourth, the lifting and boundary-adapted basis enforce non-local integral boundary conditions at machine precision. This is essential for the present problem class because standard local boundary bases do not automatically satisfy integral boundary constraints.

Finally, the full PDE experiment shows stable and qualitatively correct behavior, but it also reveals that the full space–time error depends on the interaction between temporal memory discretization, spatial approximation, and operator conditioning. This suggests that future refinements should study preconditioning, orthonormalization of the boundary-adapted basis, and balanced space–time refinement strategies.

4.8. Scope with Respect to Other Memory Kernels

The proposed method is designed for the classical Caputo derivative with a weakly singular power-law memory kernel. It should not be interpreted as a method for all fractional or generalized memory operators. Operators with non-singular or non-power-law kernels can model different hereditary mechanisms and may be more appropriate in some applications. However, they lead to different resolvent estimates and different discrete convolution weights. Extending the lifting–Galerkin strategy to those models is possible in principle, but the Beta-window weights derived here would have to be replaced by quadrature weights adapted to the corresponding kernel.

4.9. Limitations and Future Numerical Improvements

Although the numerical results are encouraging, some limitations should be noted. The high-order temporal behavior is clearest when the Caputo derivative is isolated or when exact spatial and nonlinear terms are used in a manufactured balance. In a fully coupled PDE solver, the observed convergence rate may be reduced by spatial approximation and conditioning effects. Therefore, future work should include a systematic space–time convergence study in which the spatial degree N is increased sufficiently so that the temporal error dominates.

Another possible improvement is to orthonormalize or precondition the boundary-adapted basis. Since the non-local boundary constraints modify the standard polynomial basis, the resulting mass and stiffness matrices may become less favorable as N increases. A stabilized basis could improve the accuracy of the full PDE solver without altering the Caputo approximation.

Finally, the nonlinear tests can be extended to fully implicit nonlinear Galerkin solves with damped Newton or quasi-Newton iterations. Such tests would allow a separate study of nonlinear solver robustness, convergence basins, and residual reduction. The present work already verifies the high-order fractional derivative approximation and the exact enforcement of the non-local boundary constraints, which are the two main components of the proposed approach.

5. Conclusions

This paper developed a boundary-compatible framework for nonlinear Caputo reaction–diffusion equations with non-local integral boundary conditions. The approach combines an explicit polynomial lifting of the prescribed boundary data, a homogeneous boundary-adapted Legendre–Galerkin space, and a high-order Beta-window approximation of the Caputo derivative.

The analysis established the boundedness of the non-local boundary functionals, introduced a finite-dimensional nondegeneracy condition for the lifting, and proved mild well posedness under the stated sectoriality and Lipschitz assumptions. A key advantage of the construction is that every Galerkin approximation of the form

$$u_N = R + y_N, \quad y_N \in V_N^\gamma,$$

satisfies

$$\mathcal{B}_i u_N = \phi_i, \quad i = 1, 2,$$

by construction. Thus the integral boundary constraints are built into the approximation space rather than imposed by penalties or post-processing.

The numerical results confirm the main analytical properties of the method. The Beta-window Caputo approximation shows high-order accuracy for smooth polynomial and non-polynomial benchmarks and gives a substantial improvement over the classical $L1$ approximation. The linear and nonlinear manufactured reaction–diffusion tests demonstrate stable full-scheme behavior, while the non-local boundary experiments show machine-precision enforcement of the integral constraints.

The stability statement was formulated conservatively. Since the Beta-window history coefficients may change sign, the method is not claimed to have the same positivity-based unconditional stability mechanism as the classical $L1$ scheme. Instead, stability is guaranteed under the coercivity condition

$$c_\alpha \tau^{-\alpha} + \kappa \lambda_{1,N} > L_f.$$

This condition identifies the combined roles of the time step, diffusion strength, reaction Lipschitz constant, and the first discrete eigenvalue of the boundary-adapted operator.

Overall, the proposed method provides a well-posed Caputo evolution setting, an exact non-local boundary lifting, a constraint-preserving spectral discretization, and a high-order memory quadrature. Future extensions may include fast memory evaluation, adaptive time stepping, higher-dimensional domains, and generalized memory kernels, but these directions do not affect the main contributions established here.

Author Contributions: Conceptualization, W.A., K.A. and Á.H.S.; methodology, W.A., K.A. and Á.H.S.; software, Á.H.S.; validation, W.A., K.A. and Á.H.S.; formal analysis, W.A., K.A. and Á.H.S.; investigation, W.A., K.A. and Á.H.S.; writing—original draft preparation, W.A., K.A. and Á.H.S.; writing—review and editing, W.A., K.A. and Á.H.S.; visualization, Á.H.S.; supervision, Á.H.S. All authors have read and agreed to the published version of the manuscript.

Funding: The authors express their gratitude to Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2026R229), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data supporting the reported numerical results are included in the article and appendices. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Mathematica Code for Example 4

The following Mathematica code reproduces the non-local boundary validation test reported in Section 3.4. It constructs the non-local lifting, the homogeneous boundary-adapted basis, the projection of the manufactured solution, and the boundary residual diagnostics.

Listing A1. Mathematica implementation of Example 4: non-local boundary validation.

```

1  ClearAll["Global`*"];
2
3  Off[General::spell, General::spell1, LinearSolve::luc, Part::partw];
4
5  (* =====*)
6  (*EXAMPLE 4:NON-LOCAL BOUNDARY VALIDATION*)
7  (*FULL WORKING CODE-- FIXED BASIS STORAGE*)
8  (*USES YOUR NEW HIGH-ORDER QUADRATURE*)
9  (*EXACT:u(x,t)=t^4 Sin[Pi x]*)
10 (* =====*)
11
12 (* =====*)
13 (*0. YOUR NEW HIGH-ORDER QUADRATURE*)
14 (* =====*)
15
16 ClearAll[integral];
17
18 qAlphaHalf = {0.999857, 0.999246, 0.998148, 0.996562, 0.994492,
19 0.991939, 0.988905, 0.985393, 0.981407, 0.97695, 0.972028,
20 0.966644, 0.960805, 0.954515, 0.947781, 0.940609, 0.933007,
21 0.924982, 0.916542, 0.907695, 0.898449, 0.888814, 0.878799,
22 0.868414, 0.857669, 0.846575, 0.835142, 0.823381, 0.811304,
23 0.798924, 0.786251, 0.773299, 0.760079, 0.746605, 0.732891,
24 0.718949, 0.704793, 0.690436, 0.675894, 0.66118, 0.646309,
25 0.631294, 0.616151, 0.600895, 0.58554, 0.570102, 0.554595,
26 0.539034, 0.523436, 0.507814};
27
28 qWeightHalf = {1.0, 2.32686, 3.65342, 4.97657, 6.2949, 7.60709,
29 8.91184, 10.2079, 11.494, 12.7688, 14.0312, 15.2799, 16.5136,
30 17.7312, 18.9315, 20.1132, 21.2754, 22.4167, 23.5361, 24.6326,
31 25.7049, 26.7522, 27.7733, 28.7673, 29.7332, 30.67, 31.5768,
32 32.4528, 33.2971, 34.1089, 34.8873, 35.6316, 36.3411, 37.0152,
33 37.653, 38.2541, 38.8177, 39.3435, 39.8308, 40.2792, 40.6882,
34 41.0575, 41.3867, 41.6754, 41.9234, 42.1304, 42.2963, 42.4209,
35 42.504, 42.5455};
36
37 qAlpha = Join[qAlphaHalf, Reverse[1.0 - qAlphaHalf]];
38 qWeight = Join[qWeightHalf, Reverse[qWeightHalf]];
39
40 If[Length[qAlpha] != Length[qWeight],
41 Print["ERROR: quadrature lists have different lengths."];
42 Abort[]];
43
44 integral[f_, a_?NumericQ, b_?NumericQ] :=
45 N[0.000367317*(b - a)*
46 Sum[qWeight[[i]]*f[qAlpha[[i]]*a + (1.0 - qAlpha[[i]])*b], {i, 1,
47 Length[qAlpha]}]];
48
49 (* =====*)
50 (*1. INPUT DATA*)

```

```

51  (* =====*)
52
53  aa = 0.0;
54  bb = 1.0;
55
56  Tfinal = 1.0;
57  Mtime = 40;
58  Ndeg = 20;
59  xTrace = 0.37;
60
61  gamma1[x_?NumericQ] := x;
62  gamma2[x_?NumericQ] := 1.0 - x;
63
64  uex[x_?NumericQ, t_?NumericQ] := t^4 Sin[Pi x];
65
66  phil[t_?NumericQ] := t^4/Pi;
67  phi2[t_?NumericQ] := t^4/Pi;
68
69  Print["====="];
70  Print["EXAMPLE 4: NON-LOCAL BOUNDARY VALIDATION"];
71  Print["Exact solution: u(x,t)=t^4 Sin[Pi x]"];
72  Print["gamma1(x)=x"];
73  Print["gamma2(x)=1-x"];
74  Print["phil(t)=t^4/Pi"];
75  Print["phi2(t)=t^4/Pi"];
76  Print["Tfinal = ", Tfinal];
77  Print["Mtime = ", Mtime];
78  Print["Ndeg = ", Ndeg];
79  Print["Quadrature points = ", Length[qAlpha]];
80  Print["====="];
81
82  (* =====*)
83  (*2. LEGENDRE MODES*)
84  (* =====*)
85
86  Clear[z];
87
88  PDerExpr[k_Integer] := PDerExpr[k] = D[LegendreP[k, z], z];
89
90  ClearAll[Ptilde, PtildeDer];
91
92  Ptilde[k_Integer, xx_?NumericQ] :=
93    N[LegendreP[k, (2.0 xx - aa - bb)/(bb - aa)]];
94
95  PtildeDer[k_Integer, xx_?NumericQ] :=
96    N[(2.0/(bb - aa))*(PDerExpr[k] /. z -> ((2.0 xx - aa - bb)/(bb - aa))]);
97
98  (* =====*)
99  (*3. NON-LOCAL BOUNDARY FUNCTIONALS*)
100  (* =====*)
101
102  ClearAll[Bfun];
103
104  Bfun[1, v_] :=
105    N[v[aa] + integral[Function[{xx}, gamma1[xx] v[xx]], aa, bb]];
106
107  Bfun[2, v_] :=
108    N[v[bb] + integral[Function[{xx}, gamma2[xx] v[xx]], aa, bb]];
109
110  ClearAll[BofP];
111
112  BofP[i_Integer, k_Integer] :=
113    BofP[i, k] = N[Bfun[i, Function[{xx}, Ptilde[k, xx]]]];
114
115  Cgamma = N@Table[BofP[i, k], {i, 1, 2}, {k, 0, 1}];
116
117  Print["Cgamma = "];
118  Print[MatrixForm[Cgamma]];
119  Print["Det[Cgamma] = ", ScientificForm[Det[Cgamma], 12]];
120
121  If[Abs[Det[Cgamma]] < 10^-12, Print["ERROR: Cgamma is singular. Stop."];
122    Abort[]];
123
124  (* =====*)
125  (*4. LIFTING R(x,t)*)

```

```

126 (* =====*)
127
128 ClearAll[rcoef, Rfun];
129
130 rcoef[t_?NumericQ] := rcoef[t] = LinearSolve[Cgamma, {phi1[t], phi2[t]};
131
132 Rfun[xx_?NumericQ, t_?NumericQ] :=
133   rcoef[t][[1]] Ptilde[0, xx] + rcoef[t][[2]] Ptilde[1, xx];
134
135 ClearAll[B1R, B2R];
136
137 B1R[t_?NumericQ] := Bfun[1, Function[{xx}, Rfun[xx, t]]];
138
139 B2R[t_?NumericQ] := Bfun[2, Function[{xx}, Rfun[xx, t]]];
140
141 Print["Check lifting at T=1:"];
142 Print["B1[R]-phi1 = ", ScientificForm[B1R[Tfinal] - phi1[Tfinal], 12]];
143 Print["B2[R]-phi2 = ", ScientificForm[B2R[Tfinal] - phi2[Tfinal], 12]];
144
145 (* =====*)
146 (*5. BASIS DATA GENERATOR-- NO GLOBAL OVERWRITING*)
147 (* =====*)
148
149 ClearAll[PsiBD, MakeBasisData];
150
151 PsiBD[bd_Association, k_Integer, xx_?NumericQ] :=
152   N[Ptilde[k, xx] - bd["CoeffAssoc"][k][[1]] Ptilde[0, xx] -
153     bd["CoeffAssoc"][k][[2]] Ptilde[1, xx]];
154
155 MakeBasisData[NN_Integer?Positive] :=
156   Module[{basisL, coeffAssoc, bd0, mass, boundaryDefect},
157     basisL = Range[2, NN];
158     coeffAssoc =
159       Association[
160         Table[k ->
161           LinearSolve[Cgamma, N@Table[BofP[i, k], {i, 1, 2}]], {k,
162             basisL}];
163     bd0 = <|"Basis" -> basisL, "CoeffAssoc" -> coeffAssoc|>;
164     mass =
165       N@Table[integral[
166         Function[{xx}, PsiBD[bd0, k, xx] PsiBD[bd0, ki, xx]], aa,
167         bb], {ki, basisL}, {kj, basisL}];
168     boundaryDefect =
169       Max[Abs[Flatten[
170         Table[{Bfun[1, Function[{xx}, PsiBD[bd0, k, xx]]],
171           Bfun[2, Function[{xx}, PsiBD[bd0, k, xx]]], {k, basisL}]]];
172     <|"Basis" -> basisL, "CoeffAssoc" -> coeffAssoc, "Mass" -> mass,
173       "BoundaryDefect" -> boundaryDefect|>;
174
175     basisData = MakeBasisData[Ndeg];
176
177     Mass = basisData["Mass"];
178
179     Print["Mass numeric? ", MatrixQ[Mass, NumericQ]];
180
181     ClearAll[condNumber];
182
183     condNumber[A_?MatrixQ] := Module[{s}, s = SingularValueList[N[A]];
184       Max[s]/Min[s]];
185
186     Print["Mass condition number = ", ScientificForm[condNumber[Mass], 12]];
187
188     Print["Max boundary defect of homogeneous basis = ",
189       ScientificForm[basisData["BoundaryDefect"], 12]];
190
191 (* =====*)
192 (*6. PROJECTION AND EVALUATION*)
193 (* =====*)
194
195 ClearAll[ProjectAtTime, EvalProjection, BluN, B2uN];
196
197 ProjectAtTime[t_?NumericQ, bd_Association] :=
198   Module[{basisL, mass, rhs}, basisL = bd["Basis"];
199     mass = bd["Mass"];
200     rhs =

```

```

201     N@Table[integral[
202         Function[{xx}, (uex[xx, t] - Rfun[xx, t]) PsiBD[bd, ki, xx]],
203         aa, bb], {ki, basisL}];
204     LinearSolve[mass, rhs]];
205
206 EvalProjection[xx_?NumericQ, t_?NumericQ, V_?VectorQ,
207     bd_Association] := Module[{basisL}, basisL = bd["Basis"];
208     Rfun[xx, t] +
209     Sum[V[[m]] PsiBD[bd, basisL[[m]], xx], {m, 1, Length[basisL]}];
210
211 BluN[t_?NumericQ, V_?VectorQ, bd_Association] :=
212     Bfun[1, Function[{xx}, EvalProjection[xx, t, V, bd]]];
213
214 B2uN[t_?NumericQ, V_?VectorQ, bd_Association] :=
215     Bfun[2, Function[{xx}, EvalProjection[xx, t, V, bd]]];
216
217 (* =====*)
218 (*7. FINAL-TIME COMPUTATION*)
219 (* =====*)
220
221 Vfinal = ProjectAtTime[Tfinal, basisData];
222
223 xNodes = N@Subdivide[aa, bb, 800];
224
225 finalProfileData =
226     Table[{xx, uex[xx, Tfinal],
227         EvalProjection[xx, Tfinal, Vfinal, basisData]}, {xx, xNodes}];
228
229 finalErrorData =
230     Table[{xx,
231         Abs[uex[xx, Tfinal] -
232             EvalProjection[xx, Tfinal, Vfinal, basisData]}], {xx, xNodes}];
233
234 finalError = Max[finalErrorData[[All, 2]]];
235
236 finalB1Residual = Abs[BluN[Tfinal, Vfinal, basisData] - phi1[Tfinal]];
237
238 finalB2Residual = Abs[B2uN[Tfinal, Vfinal, basisData] - phi2[Tfinal]];
239
240 Print["====="];
241 Print["FINAL-TIME RESULTS"];
242 Print["Final-time max error = ", ScientificForm[finalError, 12]];
243 Print["Final |B1[u_N]-phi1| = ", ScientificForm[finalB1Residual, 12]];
244 Print["Final |B2[u_N]-phi2| = ", ScientificForm[finalB2Residual, 12]];
245 Print["====="];
246
247 (* =====*)
248 (*8. TIME TRACE AND BOUNDARY RESIDUALS*)
249 (* =====*)
250
251 timeGrid = N@Subdivide[0.0, Tfinal, Mtime];
252
253 timeTraceData = Table[Module[{tt, Vt}, tt = timeGrid[[j]];
254     Vt = ProjectAtTime[tt, basisData];
255     {tt, uex[xTrace, tt],
256         EvalProjection[xTrace, tt, Vt, basisData]}], {j, 1,
257         Length[timeGrid]};
258
259 boundaryResidualData =
260     Table[Module[{tt, Vt, b1, b2}, tt = timeGrid[[j]];
261         Vt = ProjectAtTime[tt, basisData];
262         b1 = Abs[BluN[tt, Vt, basisData] - phi1[tt]];
263         b2 = Abs[B2uN[tt, Vt, basisData] - phi2[tt]];
264         {tt, b1, b2}], {j, 1, Length[timeGrid]};
265
266 maxB1Residual = Max[boundaryResidualData[[All, 2]]];
267 maxB2Residual = Max[boundaryResidualData[[All, 3]]];
268
269 Print["Max |B1[u_N]-phi1| over time = ",
270     ScientificForm[maxB1Residual, 12]];
271
272 Print["Max |B2[u_N]-phi2| over time = ",
273     ScientificForm[maxB2Residual, 12]];
274
275 (* =====*)

```

```

276 (*9. CONVERGENCE WITH RESPECT TO N*)
277 (* =====*)
278
279 ClearAll[ProjectionErrorForN];
280
281 ProjectionErrorForN[NN_Integer?Positive] :=
282   Module[{bd, V, nodes, err, b1res, b2res}, bd = MakeBasisData[NN];
283   V = ProjectAtTime[Tfinal, bd];
284   nodes = N@Subdivide[aa, bb, 800];
285   err =
286     Max[Abs[Table[
287       uex[xx, Tfinal] - EvalProjection[xx, Tfinal, V, bd], {xx,
288       nodes}]]];
289   b1res = Abs[B1uN[Tfinal, V, bd] - phi1[Tfinal]];
290   b2res = Abs[B2uN[Tfinal, V, bd] - phi2[Tfinal]];
291   {NN, err, b1res, b2res};
292
293 NList = {6, 8, 10, 12, 14, 16, 18, 20};
294
295 convNData = Table[ProjectionErrorForN[NN], {NN, NList}];
296
297 Print["====="];
298 Print["CONVERGENCE WITH RESPECT TO N"];
299 Print[Grid[
300   Prepend[convNData, {"N", "final max error", "B1 residual",
301   "B2 residual"}], Frame -> All]];
302 Print["====="];
303
304 (* =====*)
305 (*10. PLOTS*)
306 (* =====*)
307
308 pExample4Profile =
309   ListLinePlot[{finalProfileData[{All, {1, 2}}],
310     finalProfileData[{All, {1, 3}}]}, PlotRange -> All, Frame -> True,
311     FrameLabel -> {"x", "u(x,T)"},
312     PlotLegends -> {"Exact", "Projected numerical"},
313     PlotLabel -> "Example 4: non-local boundary lifting and projection",
314     GridLines -> Automatic, ImageSize -> Large];
315
316 Print[pExample4Profile];
317
318 Export["Example4_fixed_nonlocal_final_profile.png", pExample4Profile,
319   ImageResolution -> 300];
320
321 pExample4Error =
322   ListLinePlot[finalErrorData, PlotRange -> All, Frame -> True,
323     FrameLabel -> {"x", "absolute error"},
324     PlotLabel -> "Example 4: final-time projection error",
325     GridLines -> Automatic, ImageSize -> Large];
326
327 Print[pExample4Error];
328
329 Export["Example4_fixed_nonlocal_final_error.png", pExample4Error,
330   ImageResolution -> 300];
331
332 pExample4Boundary =
333   ListLogPlot[{Table[{boundaryResidualData[{i, 1}],
334     Max[boundaryResidualData[{i, 2}], 10^-18]}, {i, 1,
335     Length[boundaryResidualData]}],
336     Table[{boundaryResidualData[{i, 1}],
337     Max[boundaryResidualData[{i, 3}], 10^-18]}, {i, 1,
338     Length[boundaryResidualData]}]}, Joined -> True, Mesh -> All,
339     PlotRange -> All, Frame -> True,
340     FrameLabel -> {"t", "absolute boundary residual"},
341     PlotLegends -> {"|B1[u_N]-phi1|", "|B2[u_N]-phi2|"},
342     PlotLabel -> "Example 4: non-local boundary residuals",
343     GridLines -> Automatic, ImageSize -> Large];
344
345 Print[pExample4Boundary];
346
347 Export["Example4_fixed_nonlocal_boundary_residuals.png",
348   pExample4Boundary, ImageResolution -> 300];
349
350 pExample4Trace =

```

```

351 ListLinePlot[{timeTraceData[All, {1, 2}],
352   timeTraceData[All, {1, 3}]}], PlotRange -> All, Frame -> True,
353   FrameLabel -> {"t", "u(0.37,t)"},
354   PlotLegends -> {"Exact", "Projected numerical"},
355   PlotLabel -> "Example 4: time trace at x=0.37",
356   GridLines -> Automatic, ImageSize -> Large];
357
358 Print[pExample4Trace];
359
360 Export["Example4_fixed_nonlocal_time_trace_x037.png", pExample4Trace,
361   ImageResolution -> 300];
362
363 pExample4TraceError =
364   ListLinePlot[
365     Table[{timeTraceData[i, 1],
366       Abs[timeTraceData[i, 2] - timeTraceData[i, 3]]}, {i, 1,
367       Length[timeTraceData]}], PlotRange -> All, Frame -> True,
368     FrameLabel -> {"t", "absolute error"},
369     PlotLabel -> "Example 4: time-trace projection error at x=0.37",
370     GridLines -> Automatic, ImageSize -> Large];
371
372 Print[pExample4TraceError];
373
374 Export["Example4_fixed_nonlocal_time_trace_error_x037.png",
375   pExample4TraceError, ImageResolution -> 300];
376
377 pExample4ConvN =
378   ListLogPlot[convNData[All, {1, 2}], Joined -> True, Mesh -> All,
379   PlotMarkers -> Automatic, PlotRange -> All, Frame -> True,
380   FrameLabel -> {"N", "final-time max error"},
381   PlotLabel -> "Example 4: convergence in the non-local basis",
382   GridLines -> Automatic, ImageSize -> Large];
383
384 Print[pExample4ConvN];
385
386 Export["Example4_fixed_nonlocal_convergence_N.png", pExample4ConvN,
387   ImageResolution -> 300];
388
389 (* =====*)
390 (*11. EXPORT DATA*)
391 (* =====*)
392
393 Export["Example4_fixed_final_profile_data.csv",
394   Prepend[finalProfileData, {"x", "exact", "projected numerical"}]];
395
396 Export["Example4_fixed_final_error_data.csv",
397   Prepend[finalErrorData, {"x", "absolute error"}]];
398
399 Export["Example4_fixed_boundary_residual_data.csv",
400   Prepend[boundaryResidualData, {"t", "B1 residual", "B2 residual"}]];
401
402 Export["Example4_fixed_time_trace_x037.csv",
403   Prepend[timeTraceData, {"t", "exact", "projected numerical"}]];
404
405 Export["Example4_fixed_convergence_N_table.csv",
406   Prepend[convNData, {"N", "final max error", "B1 residual",
407     "B2 residual"}]];
408
409 (* =====*)
410 (*12. SUMMARY*)
411 (* =====*)
412
413 Print["====="];
414 Print["EXPORTED PICTURES FOR EXAMPLE 4 FIXED:"];
415 Print["Example4_fixed_nonlocal_final_profile.png"];
416 Print["Example4_fixed_nonlocal_final_error.png"];
417 Print["Example4_fixed_nonlocal_boundary_residuals.png"];
418 Print["Example4_fixed_nonlocal_time_trace_x037.png"];
419 Print["Example4_fixed_nonlocal_time_trace_error_x037.png"];
420 Print["Example4_fixed_nonlocal_convergence_N.png"];
421 Print[""];
422 Print["EXPORTED DATA FILES:"];
423 Print["Example4_fixed_final_profile_data.csv"];
424 Print["Example4_fixed_final_error_data.csv"];
425 Print["Example4_fixed_boundary_residual_data.csv"];

```

```

426 Print["Example4_fixed_time_trace_x037.csv"];
427 Print["Example4_fixed_convergence_N_table.csv"];
428 Print["====="];

```

Appendix B. Mathematica Code for Example 5

The following Mathematica code reproduces the comparison between the proposed Beta-window Caputo approximation and the classical $L1$ scheme reported in Section 3.5.

Listing A2. Mathematica implementation of Example 5: comparison between the Beta-window approximation and the classical $L1$ scheme.

```

1 ClearAll["Global`*"];
2
3 Off[General::spell,General::spell1];
4
5 (* =====*)
6 (*EXAMPLE 5:COMPARISON BETWEEN KaputoApp AND CLASSICAL L1*)
7 (*ONE FULL WORKING BLOCK*)
8 (*TEST:f(t)=t^4,rho=0.65,T=1*)
9 (* =====*)
10
11 (* =====*)
12 (*1. INPUT DATA*)
13 (* =====*)
14
15 rho=0.65;
16 Tfinal=1.0;
17
18 MList={10,20,40,80,160,320};
19
20 Print["====="];
21 Print["EXAMPLE 5: COMPARISON BETWEEN KaputoApp AND L1"];
22 Print["Test function: f(t)=t^4"];
23 Print["rho = ",rho];
24 Print["Tfinal = ",Tfinal];
25 Print["MList = ",MList];
26 Print["====="];
27
28 (* =====*)
29 (*2. TEST FUNCTION AND EXACT CAPUTO DERIVATIVE*)
30 (* =====*)
31
32 ClearAll[f,df,CaputoExact];
33
34 f[t_?NumericQ]:=t^4;
35
36 (*Since 0<rho<1,m=Ceiling[rho]=1,so we need f'(t).*)
37 df[t_?NumericQ]:=4 t^3;
38
39 CaputoExact[t_?NumericQ]:=Gamma[5]/Gamma[5-rho] t^(4-rho);
40
41 exactFinal=CaputoExact[Tfinal];
42
43 Print["Exact Caputo derivative at T=1 = ",ScientificForm[exactFinal,16]];
44
45 (* =====*)
46 (*3. KaputoApp FORMULA USING f'(t) DIRECTLY*)
47 (* =====*)
48
49 ClearAll[KaputoApp];
50
51 KaputoApp[dfun_,n_Integer?Positive,h_?NumericQ,rho_?NumericQ]:=Module[{m,m=Ceiling[rho]};
52 If[m!=1,Print["This Example 5 block is written for 0<rho<1."];
53 Abort[]];
54 N[(h^(m-rho)/Gamma[m-rho])*Sum[With[{k=n-j},dfun[j h]*(2 k^(m+2-rho) Beta[1/k,3,m-rho]
55 -3 k^(m+1-rho) Beta[1/k,2,m-rho]+k^(m-rho) Beta[1/k,1,m-rho])+dfun[(j+1/2) h]*(-4
56 k^(m+2-rho) Beta[1/k,3,m-rho]+4 k^(m+1-rho) Beta[1/k,2,m-rho])+dfun[(j+1) h]*(2 k^(
57 m+2-rho) Beta[1/k,3,m-rho]-k^(m+1-rho) Beta[1/k,2,m-rho])],{j,0,n-1}]]];
58
59 (* =====*)
60 (*4. CLASSICAL L1 FORMULA*)

```

```

58 (* =====*)
59
60 ClearAll[L1Caputo];
61
62 L1Caputo[fun_, n_Integer?Positive, h_?NumericQ, rho_?NumericQ] := Module[{a}, a[j_] := (j+1.0)
63   ^ (1.0-rho) - j^ (1.0-rho);
64   N[h^ (-rho) / Gamma[2.0-rho] * Sum[a[j] * (fun[(n-j) h] - fun[(n-j-1) h]), {j, 0, n-1}]]];
65
66 (* =====*)
67 (*5. ERROR TABLE*)
68 (* =====*)
69
70 ClearAll[computeRows, rateList];
71
72 computeRows := Module[{rows, h, n, kap, l1, kapErr, l1Err, ratio}, rows = Table[h = Tfinal/M;
73   n = M;
74   kap = KaputoApp[df, n, h, rho];
75   l1 = L1Caputo[f, n, h, rho];
76   kapErr = Abs[kap - exactFinal];
77   l1Err = Abs[l1 - exactFinal];
78   ratio = l1Err / kapErr;
79   {M, h, kap, l1, kapErr, l1Err, ratio}, {M, MList}];
80
81 rows = computeRows;
82
83 kapRates = Table[If[i == 1, "-", Log[rows[[i-1, 5]] / rows[[i, 5]]] / Log[2]], {i, Length[rows]}];
84
85 l1Rates = Table[If[i == 1, "-", Log[rows[[i-1, 6]] / rows[[i, 6]]] / Log[2]], {i, Length[rows]}];
86
87 fullRows = Table[Join[rows[[i]], {kapRates[[i]], l1Rates[[i]]}], {i, Length[rows]}];
88
89 comparisonTable = Prepend[fullRows, {"M", "h", "KaputoApp value", "L1 value", "KaputoApp error",
90   "L1 error", "L1/KaputoApp error ratio", "KaputoApp rate", "L1 rate"}];
91
92 Print["====="];
93 Print["EXAMPLE 5 COMPARISON TABLE"];
94 Print[Grid[comparisonTable, Frame -> All]];
95 Print["====="];
96
97 (* =====*)
98 (*6. SHORT NUMERICAL SUMMARY*)
99 (* =====*)
100
101 Print["At M = ", rows[[3, 1]], ":"];
102 Print["KaputoApp error = ", ScientificForm[rows[[3, 5]], 12]];
103 Print["L1 error = ", ScientificForm[rows[[3, 6]], 12]];
104 Print["L1/KaputoApp error ratio = ", ScientificForm[rows[[3, 7]], 12]];
105
106 Print["At M = ", rows[[-1, 1]], ":"];
107 Print["KaputoApp error = ", ScientificForm[rows[[-1, 5]], 12]];
108 Print["L1 error = ", ScientificForm[rows[[-1, 6]], 12]];
109 Print["L1/KaputoApp error ratio = ", ScientificForm[rows[[-1, 7]], 12]];
110
111 (* =====*)
112 (*7. DATA FOR PLOTS*)
113 (* =====*)
114
115 kapErrorData = Table[{rows[[i, 2]], rows[[i, 5]]}, {i, Length[rows]}];
116
117 l1ErrorData = Table[{rows[[i, 2]], rows[[i, 6]]}, {i, Length[rows]}];
118
119 ratioData = Table[{rows[[i, 1]], rows[[i, 7]]}, {i, Length[rows]}];
120
121 rateData = Table[If[i == 1, Nothing, {rows[[i, 1]], kapRates[[i]], l1Rates[[i]]}], {i, Length[rows]}];
122
123 approxValueData = Table[{rows[[i, 1]], rows[[i, 3]], rows[[i, 4]], exactFinal}, {i, Length[rows]}];
124
125 (* =====*)
126 (*8. PLOT 1: ERROR COMPARISON*)
127 (* =====*)

```

```

128 pErrors=ListLogLogPlot[{kapErrorData,l1ErrorData},Joined->True,Mesh->All,PlotMarkers->
      Automatic,PlotRange->All,Frame->True,FrameLabel->{"h","absolute error"},PlotLegends
      ->{"KaputoApp","Classical L1"},PlotLabel->"Example 5: KaputoApp versus L1 error
      comparison",GridLines->Automatic,ImageSize->Large];
129
130 Print[pErrors];
131
132 Export["Example5_KaputoApp_vs_L1_errors.png",pErrors,ImageResolution->300];
133
134 (* =====*)
135 (*9. PLOT 2:ERROR RATIO*)
136 (* =====*)
137
138 pRatio=ListLogPlot[ratioData,Joined->True,Mesh->All,PlotMarkers->Automatic,PlotRange->
      All,Frame->True,FrameLabel->{"M","L1 error / KaputoApp error"},PlotLabel->"Example
      5: accuracy gain of KaputoApp over L1",GridLines->Automatic,ImageSize->Large];
139
140 Print[pRatio];
141
142 Export["Example5_error_ratio.png",pRatio,ImageResolution->300];
143
144 (* =====*)
145 (*10. PLOT 3:OBSERVED RATES*)
146 (* =====*)
147
148 pRates=ListLinePlot[{rateData[{All,{1,2}}],rateData[{All,{1,3}}]},Joined->True,Mesh->
      All,PlotMarkers->Automatic,PlotRange->All,Frame->True,FrameLabel->{"M","observed
      rate"},PlotLegends->{"KaputoApp","Classical L1"},PlotLabel->"Example 5: observed
      convergence rates",GridLines->Automatic,ImageSize->Large];
149
150 Print[pRates];
151
152 Export["Example5_KaputoApp_vs_L1_rates.png",pRates,ImageResolution->300];
153
154 (* =====*)
155 (*11. PLOT 4:APPROXIMATED VALUES*)
156 (* =====*)
157
158 pValues=ListLinePlot[{approxValueData[{All,{1,2}}],approxValueData[{All,{1,3}}],
      approxValueData[{All,{1,4}}]},Joined->True,Mesh->All,PlotMarkers->Automatic,
      PlotRange->All,Frame->True,FrameLabel->{"M","Caputo derivative at T=1"},PlotLegends
      ->{"KaputoApp","Classical L1","Exact"},PlotLabel->"Example 5: approximated Caputo
      derivative values",GridLines->Automatic,ImageSize->Large];
159
160 Print[pValues];
161
162 Export["Example5_approximated_values.png",pValues,ImageResolution->300];
163
164 (* =====*)
165 (*12. OPTIONAL:POINTWISE CURVES IN TIME FOR M=40*)
166 (* =====*)
167
168 Mcurve=40;
169 hcurve=Tfinal/Mcurve;
170
171 timeCurveData=Table[{nn hcurve,KaputoApp[df,nn,hcurve,rho],L1Caputo[f,nn,hcurve,rho],
      CaputoExact[nn hcurve]},{nn,1,Mcurve}];
172
173 pTimeCurve=ListLinePlot[{timeCurveData[{All,{1,2}}],timeCurveData[{All,{1,3}}],
      timeCurveData[{All,{1,4}}]},Joined->True,PlotRange->All,Frame->True,FrameLabel->{"t",
      "Caputo derivative"},PlotLegends->{"KaputoApp","Classical L1","Exact"},PlotLabel
      ->"Example 5: Caputo derivative curve for f(t)=t^4, M=40",GridLines->Automatic,
      ImageSize->Large];
174
175 Print[pTimeCurve];
176
177 Export["Example5_time_curve_M40.png",pTimeCurve,ImageResolution->300];
178
179 timeErrorData=Table[{timeCurveData[{i,1}],Abs[timeCurveData[{i,2}]-timeCurveData[{i
      ,4}]],Abs[timeCurveData[{i,3}]-timeCurveData[{i,4}]]},{i,Length[timeCurveData]};
180
181 pTimeErrors=ListLogPlot[{timeErrorData[{All,{1,2}}],timeErrorData[{All,{1,3}}]},Joined
      ->True,Mesh->All,PlotRange->All,Frame->True,FrameLabel->{"t","absolute error"},
      PlotLegends->{"KaputoApp","Classical L1"},PlotLabel->"Example 5: pointwise errors
      for M=40",GridLines->Automatic,ImageSize->Large];

```

```

182
183 Print[pTimeErrors];
184
185 Export["Example5_time_errors_M40.png", pTimeErrors, ImageResolution->300];
186
187 (* =====*)
188 (*13. EXPORT DATA*)
189 (* =====*)
190
191 Export["Example5_KaputoApp_vs_L1_table.csv", comparisonTable];
192
193 Export["Example5_error_data.csv", Prepend[Table[{rows[[i, 1]], rows[[i, 2]], rows[[i, 5]],
194 rows[[i, 6]], rows[[i, 7]]}, {i, Length[rows]}], {"M", "h", "KaputoApp error", "L1 error", "
195 L1/KaputoApp ratio"}]];
196
197 Export["Example5_time_curve_M40.csv", Prepend[timeCurveData, {"t", "KaputoApp", "L1", "Exact
198 " }]];
199
200 Export["Example5_time_errors_M40.csv", Prepend[timeErrorData, {"t", "KaputoApp error", "L1
201 error"}]];
202
203 (* =====*)
204 (*14. SUMMARY*)
205 (* =====*)
206
207 Print["====="];
208 Print["EXPORTED PICTURES FOR EXAMPLE 5:"];
209 Print["Example5_KaputoApp_vs_L1_errors.png"];
210 Print["Example5_error_ratio.png"];
211 Print["Example5_KaputoApp_vs_L1_rates.png"];
212 Print["Example5_approximated_values.png"];
213 Print["Example5_time_curve_M40.png"];
214 Print["Example5_time_errors_M40.png"];
215 Print[""];
216 Print["EXPORTED DATA FILES:"];
217 Print["Example5_KaputoApp_vs_L1_table.csv"];
218 Print["Example5_error_data.csv"];
219 Print["Example5_time_curve_M40.csv"];
220 Print["Example5_time_errors_M40.csv"];
221 Print["====="];

```

References

- Podlubny, I. *Fractional Differential Equations*; Academic Press: San Diego, CA, USA, 1999.
- Kilbas, A.A.; Srivastava, H.M.; Trujillo, J.J. *Theory and Applications of Fractional Differential Equations*; Elsevier: Amsterdam, The Netherlands, 2006.
- Diethelm, K. *The Analysis of Fractional Differential Equations*; Springer: Berlin, Germany, 2010.
- Metzler, R.; Klafter, J. The random walk's guide to anomalous diffusion: A fractional dynamics approach. *Phys. Rep.* **2000**, *339*, 1–77.
- Mainardi, F. *Fractional Calculus and Waves in Linear Viscoelasticity*; Imperial College Press: London, UK, 2010.
- Ahmad, M.; Khan, M.N.; Ahmad, I. Local meshless methods for elliptic PDEs with multipoint boundary conditions: Investigating efficiency and accuracy of various RBFs. *Eur. Phys. J. Spec. Top.* **2025**, *234*, 2525–2542. <https://doi.org/10.1140/epjs/s11734-024-01416-8>.
- Aziz, I.; Nisar, M.; ul-Islam, S. On the numerical solution of some differential equations with nonlocal integral boundary conditions via Haar wavelet. *Proc. Est. Acad. Sci.* **2022**, *71*, 30–54. <https://doi.org/10.3176/proc.2022.1.03>.
- Khan, M.N.; Ahmad, M.; Jan, R.; Ahmad, I.; Mousa, M. Solving parabolic differential equations via Haar wavelets: A focus on integral boundary conditions. *An. Int. J. Optim. Control Theor. Appl.* **2025**, *15*, 594–609. <https://doi.org/10.36922/IJOCTA025160082>.
- Alikhanov, A.A. A new difference scheme for the time fractional diffusion equation. *J. Comput. Phys.* **2015**, *280*, 424–438. <https://doi.org/10.1016/j.jcp.2014.09.031>.
- Jin, B.; Lazarov, R.; Zhou, Z. An analysis of the L_1 scheme for the subdiffusion equation with nonsmooth data. *IMA J. Numer. Anal.* **2016**, *36*, 197–221. <https://doi.org/10.1093/imanum/dru063>.
- Stynes, M.; O'Riordan, E.; Gracia, J.L. Error analysis of a finite difference method on graded meshes for a time-fractional diffusion equation. *SIAM J. Numer. Anal.* **2017**, *55*, 1057–1079. <https://doi.org/10.1137/16M1082329>.
- Liao, H.-L.; McLean, W.; Zhang, J. A discrete Grönwall inequality with applications to numerical schemes for subdiffusion problems. *SIAM J. Numer. Anal.* **2019**, *57*, 218–237. <https://doi.org/10.1137/16M1175742>.

13. Alikhanov, A.A.; Huang, C. A high-order L_2 type difference scheme for the time-fractional diffusion equation. *Appl. Math. Comput.* **2021**, *411*, 126545. <https://doi.org/10.1016/j.amc.2021.126545>.
14. Caputo, M. Linear models of dissipation whose Q is almost frequency independent—II. *Geophys. J. R. Astron. Soc.* **1967**, *13*, 529–539. <https://doi.org/10.1111/j.1365-246X.1967.tb02303.x>.
15. Meerschaert, M.M.; Sikorskii, A. *Stochastic Models for Fractional Calculus*; De Gruyter: Berlin, Germany, 2012.
16. Hilfer, R. (Ed.) *Applications of Fractional Calculus in Physics*; World Scientific: Singapore, 2000.
17. Cannon, J.R. The solution of the heat equation subject to the specification of energy. *Q. Appl. Math.* **1963**, *21*, 155–160.
18. Byszewski, L. Theorems about the existence and uniqueness of solutions of a semilinear evolution nonlocal Cauchy problem. *J. Math. Anal. Appl.* **1991**, *162*, 494–505. [https://doi.org/10.1016/0022-247X\(91\)90164-U](https://doi.org/10.1016/0022-247X(91)90164-U).
19. Deng, K. Exponential decay of solutions of semilinear parabolic equations with nonlocal initial conditions. *J. Math. Anal. Appl.* **1993**, *179*, 630–637. <https://doi.org/10.1006/jmaa.1993.1373>.
20. Pao, C.V. *Nonlinear Parabolic and Elliptic Equations*; Plenum Press: New York, NY, USA, 1992.
21. Kamynin, L.I. A boundary value problem in the theory of heat conduction with non-classical boundary condition. *SR Comput. Math. Math. Phys.* **1964**, *4*, 33–59.
22. Pazy, A. *Semigroups of Linear Operators and Applications to Partial Differential Equations*; Springer: New York, NY, USA, 1983.
23. Henry, D. *Geometric Theory of Semilinear Parabolic Equations*; Springer: Berlin, Germany, 1981.
24. Lunardi, A. *Analytic Semigroups and Optimal Regularity in Parabolic Problems*; Birkhäuser: Basel, Switzerland, 1995.
25. Luchko, Y. Initial-boundary-value problems for the generalized multi-term fractional diffusion equation. *J. Math. Anal. Appl.* **2011**, *374*, 538–548.
26. Greiner, G. Perturbing the boundary conditions of a generator. *Houst. J. Math.* **1987**, *13*, 213–229.
27. Engel, K.-J.; Nagel, R. *One-Parameter Semigroups for Linear Evolution Equations*; Springer: New York, NY, USA, 2000.
28. Amann, H. *Linear and Quasilinear Parabolic Problems*; Birkhäuser: Basel, Switzerland, 1995; Volume I: Abstract Linear Theory

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.