Journal of Corrosion and Materials
www.corromat.com

J. Corros. Mater. (2026) 5(2), 163–180.

Algebraic Time Regularization for Product-Integration Discretizations of Nonlinear Caputo Evolution Equations

Shem Oyoo Wandiga[1,*]

Institute for Climate Change and Adaptation, University of Nairobi, P.O. Box 30197-00100, Nairobi, Kenya

*Corresponding author: wandigas@uonbi.ac.ke

(Received: 13 July 2026. Received in revised form: 14 September 2026. Accepted: 29 September 2026. Published online: 30 December 2026.)

Abstract

Weak initial regularity occurs quite commonly in nonlinear Caputo evolution equations even if the coefficient functions and boundary conditions are sufficiently smooth. The lack of regularity of the solution at the beginning is an essential numerical challenge since the error generated near the starting time propagates through the hereditary integration operation. The algebraic time transformation $t = s^{\eta/\alpha}$ with $\eta \in \mathbb{N}^+$ is employed to derive product-integration schemes for nonlinear models having singular terms $t^{j\alpha}$ with $0 < \alpha < 1$. In particular, the power of the Caputo derivatives can be transformed into a smoother function of the transformed time, which allows local interpolation of the nonlinearity. The weakly singular memory kernel is incorporated directly within the weights of the product integration. In the case of $\eta = 1$, piecewise constant interpolation yields first-order implicit and delayed-state rectangle methods. If $\eta = 2$, then linear interpolation results in implicit, Newton-linearized, and extrapolated trapezoidal schemes with order two temporal accuracy. The proof of convergence consists of a careful study of the transformed quadrature weights along with a weakly singular discrete Gronwall lemma. The rectangle methods show first-order convergence while the trapezoidal schemes achieve second-order temporal accuracy for $\alpha = 0.4, 0.6, 0.8$ for both the test examples considered, which are the exact solution $u(t) = t + t^\alpha$ for a nonlinear scalar Caputo reaction equation and the exact solution $u(x,t) = (t + t^\alpha)\sin x$ of a time-fractional Allen–Cahn equation. It turns out that for the Allen–Cahn model, the transformed scheme performs effectively not only for the pure nonlinear Caputo problem but also when combined with the diffusion and cubic reaction. Moreover, the finite differences are second-order accurate in space for this equation.

Keywords: Caputo derivative; product integration; initial-layer singularity; nonlinear fractional equation; Volterra integral equation; fractional Allen–Cahn equation; discrete Gronwall inequality; memory-dependent materials modelling.

1. Introduction

Time-fractional differential equations arise in processes where the current evolution involves a memory of all past states of the system. Classical fractional-calculus theory gives the general mathematical foundation for arbitrary-order differentiation and integration [1]. Podlubny's monograph provides a standard treatment of fractional differential equations and their solution methods [2]. Hilfer's edited volume illustrates the role of fractional models in physical applications [3]. Mainardi's work connects fractional calculus with waves, relaxation, and viscoelastic memory [4]. Such models have applications in anomalous diffusion, generalized relaxation, viscoelastic behavior, delayed transport, electrochemical memory, degradation kinetics, and phase dynamics. The common denominator of such models is the convolution kernel that retains the influence of the entire history on the current state. The advantage of such models is that they provide an economical description of processes that are not easily characterized using a finite number of classical first-order relaxation mechanisms.

A typical physical process responsible for fractional time dependence is the presence of traps, heterogeneity, broad distributions of waiting times, and nonexponential relaxation. Subdiffusive regimes can be derived from continuous-time random walk considerations and power-law waiting times [5]. Fractional viscoelasticity describes relaxation phenomena across a wide range of characteristic timescales [4]. Similarly, fractional behavior is expected for material computations involving the presence of passive films, porosity, multiphase regions, microstructural obstacles, and grain boundary networks that act to slow ionic transport or reactions. In these cases, a fractional model offers an efficient means of accounting for memory in the sense that the initial distribution of a state variable affects subsequent transport or reaction rates.

There are two main problems associated with solving Caputo models numerically. The first concerns the history integral involved. At each step, the computation depends on the history of previous steps, requiring a numerical approach to preserve a record of the past steps without losing stability. Analysis of fractional differential equations already shows why solution regularity must be treated carefully [6]. Classical convolution quadrature introduced a systematic way to discretize fractional memory [7]. Its operational-calculus formulation broadened this approach for evolution problems [8]. Fractional diffusion-wave analysis demonstrated how convolution quadrature interacts with weakly singular kernels [9]. Fully discrete difference schemes also identified temporal and spatial approximation issues for fractional diffusion-wave systems [10].

Finite difference and spectral approximations further confirmed order sensitivity in time-fractional diffusion equations [11]. Semidiscrete finite-element error estimates showed the same challenge in fractional parabolic problems [12]. Delay-type fractional equations added another setting where history and state dependence interact [13]. High-order compact schemes illustrated the continuing effort to recover accuracy under fractional subdiffusion [14]. Second, if the problem data are sufficiently smooth, the fractional solution may still exhibit weak regularity at $t = 0$, which implies a loss of accuracy when solving the equation directly in physical time.

An obvious issue is that the singularity cannot be limited to the lowest few steps. The Caputo derivative holds onto memory of the early times, and an erroneous characterization of the initial layer may affect the solution at all time levels during the evolution. Weakly singular discrete Gronwall inequalities provide one classical tool for controlling this propagation [15]. Implicit-explicit schemes for nonlinear fractional equations with nonsmooth solutions show how nonlinear response and initial layers interact [16]. Numerical analysis of nonlinear subdiffusion equations gives a related framework for stability and convergence [17]. A modern discrete Gronwall inequality further supports convergence proofs for subdiffusion schemes [18]. Linearized compact ADI schemes show that nonlinear time-fractional models require special treatment of the coupled state and memory terms [19]. Energy-dissipation analysis for time-fractional phase-field equations adds an important stability perspective [20]. Time-fractional Allen–Cahn analysis is especially relevant for the reaction–diffusion example considered later in this paper [21]. Nonuniform Alikhanov Galerkin finite-element methods show how linearization and nonuniform time stepping can be combined [22]. Second-order schemes for fractional KdV–Burgers equations provide another example where initial singularity must be captured directly [23].

A number of numerical methods have been proposed to mitigate the impact of low initial regularity. Nonuniform-mesh finite difference methods concentrate temporal resolution near $t = 0$ [24]. Alikhanov's difference scheme improves the temporal approximation for time-fractional diffusion equations [25]. Graded-mesh error analysis explains how mesh concentration can recover convergence for singular solutions [26]. Fast Caputo evaluation methods reduce the cost of history-dependent calculations [27]. Sharp estimates for nonuniform L1 formulas clarify the role of time-step variation [28]. Further L1 analysis on graded and uniform meshes confirms the sensitivity of accuracy to the starting singularity [29]. A novel approach for nonsmooth fractional parabolic solutions provides another route for handling weak initial data [30]. Recent schemes designed to capture dramatic initial evolutions of nonlinear subdiffusion equations are also closely related [31]. The present approach relies on a different technique. The physical time scale is converted to another time scale prior to application of the quadrature formula. With $t = s^{\eta/\alpha}$, the singular part $t^{j\alpha}$ is transformed to $s^{j\eta}$. Thus, the new unknown function becomes more regular with respect to the independent variable of the quadrature scheme. Moreover, the integration weights are calculated as usual, but for a transformed weakly singular kernel of Volterra integral operators.

It will be interesting to find out if such algebraic regularity suffices to recover the desired temporal order of accuracy for nonlinear Caputo equations with fractional initial layers. Our goal in the paper is to answer that question using analytical and computational means. First of all, we consider a scalar nonlinear reaction equation of the Caputo type with an exact solution $u(t) = t + t^{\alpha}$. This problem tests the temporal part of the solution without spatial discretization. Secondly, a fractional Allen-Cahn equation with an exact solution $u(x, t) = (t + t^{\alpha}) \sin x$ is analyzed. This example combines diffusion, nonlinear reaction, and a known temporal singularity part. The comparison with Euler, L1, and CQBDF1 formulas in physical time identifies the role of the transformation in order recovery.

However, the numerical accuracy of the fractional solver cannot be determined solely on the basis of its consistency on smooth test functions. Indeed, although a Caputo solution may have a continuous behavior, it might still lack the necessary derivatives needed for an accurate high-order quadrature approach. This point is crucial in memory-dominated problems where the earlier part of the simulation enters the computation through the singular kernel, and stays there for good in each following convolution step. Therefore, even if all other subsequent time steps are small, any initial error will propagate through future computations and change the final asymptotic dynamics. Thus, besides temporal resolution, the regularity class for the interpolation becomes crucial.

This observation makes the use of product integration especially attractive in our case because this method does not approximate the singular part of the integrand as if it were smooth. Namely, it uses the full kernel on each subinterval, while an approximation of the remaining nonlinear part is taken into account. In a linear problem, such a separation increases accuracy even in cases of weakly singular Volterra equations. However, in the nonlinear Caputo equation, the distinction between the nonlinear response and the kernel becomes crucial. They play entirely different roles analytically, since the former depends on the present state, while the latter describes the weighting of the past values. Hence, putting both parts into one integrand will blur their essential difference.

In contrast to simple meshing methods, product integration selects the algebraic transformation depending on the expansion at the point of singularity. For example, if the solution has the terms $t^{j\alpha}$, then the transformation $t = s^{\eta/\alpha}$ will transform them to $s^{j\eta}$. That is, we aim specifically at powers, whose presence deteriorates the temporal regularity of the

solution. It should be stressed that this methodology is distinct from the idea of clustering points close to $t = 0$. Although a graded mesh modifies the position of nodes, it uses approximation in the same variable as before. In particular, the transformation modifies the very function approximated in product integration. This is why the recovered order is visible in the slope of the convergence curves rather than only in a smaller constant.

Another point supporting the use of the Volterra representation is the clear way from continuous memory to the weights. The differential definition of the Caputo derivative is better for modelling, while the integral one demonstrates explicitly the kernel that needs approximation. Having seen the kernel, it becomes possible to split the numerical procedure into two stages, integration of the weakly singular memory factor and interpolation of the rest state-dependent function. This separation is the fundamental principle underlying the algorithms presented below. It also allows comparison with classical discretized fractional calculus [7]. The same comparison extends naturally to convolution quadrature based on operational calculus [8]. Fractional diffusion-wave discretization provides another reference for memory-weight construction [9].

It is known from the literature on nonsmooth fractional solutions that the order of accuracy cannot be improved simply using the high degree of the method. If the singular parts of the solution are excluded from regularity conditions, any nominal high-order algorithm becomes low-order. Such situations are usually solved by using graded meshes or correcting the formula used for the time grid, which changes either the distribution of temporal nodes or adds the singular corrections. Instead, the transformed method keeps the same approach, but the improvement of regularity is made in the process of equation transformation rather than as an additional correction to its solution.

As was already mentioned above, the use of physical-time methods for analysis of singularity was never aimed at rejecting the fractional approaches. It rather allows explaining some peculiarities of their behaviour. Euler-type and convolution-quadrature formulas remain important references for fractional differential equations [7]. Alikhanov-type formulas give another high-accuracy route for time-fractional diffusion problems [25]. Nonuniform L1 formulas also provide sharp error control under suitable temporal meshes [28]. Numerical calculations demonstrate that a directly implemented formula suffers from the same limitation due to the lack of regularity of the combination $t + t^{\alpha}$. When the problem is transformed into the new variable and the memory kernel left in the weight, the desired order of accuracy of product rule formula is obtained.

The novelty is in the coupling of Volterra transformation approach, product-integration weights, nonlinear end-point methods, and a suitable convergent procedure that works with weakly singular memory kernels. First order methods use product-rectangular integration after $t = s^{1/\alpha}$. Second order methods use product-trapezoidal integration after $t = s^{2/\alpha}$. The developed methods involve implicit, delay-term, Newton-linearization, and extrapolation approaches. First and second order error estimates are rigorously established in the transformed variable, and the computed error tables justify the predicted convergence rates for 0.4, 0.6, and 0.8 fractional derivatives.

2. Caputo Type Nonlinear Problem with Weak Initial Data

For $0 < \alpha < 1$, the Caputo derivative of a sufficiently differentiable function y is

$$ {}^{C}D_{t}^{\alpha} y(t) = \frac{1}{\Gamma(1-\alpha)} \int_{0}^{t} (t-r)^{-\alpha} y'(r)\, dr. \tag{1} $$

The kernel $(t-r)^{-\alpha}$ is weakly singular and nonlocal. The Caputo form is convenient because the initial value is prescribed as $y(0) = y_0$, while the derivative carries the hereditary contribution of earlier states. The nonlinear equation considered in the analysis is

$$ {}^{C}D_{t}^{\alpha} y(t) = f(t, y(t)), \qquad y(0) = y_0, \qquad 0 < t \le T. \tag{2} $$

The function f is assumed to be Lipschitz continuous with respect to its second argument on the relevant solution set and sufficiently smooth in its variables. The right-hand side may represent reaction, relaxation, damage growth, phase evolution, or a nonlinear forcing relation.

Equation (2) is equivalent to the Volterra integral equation

$$ y(t) = y_0 + \frac{1}{\Gamma(\alpha)} \int_{0}^{t} (t-r)^{\alpha-1} f(r, y(r))\, dr. \tag{3} $$

This formulation is employed for product integration. The weakly singular kernel is handled analytically within the coefficient of the quadrature rule, while the nonlinear term is linearized using interpolation in the local sense. This is critical. The weakly singular kernel does not get approximated by some nice polynomial; instead, $f(r, y(r))$ gets approximated within each subinterval.

A distinct characteristic of the Caputo evolution is an approximation of the solution in the form of

$$y(t) = \psi(t) + \sum_{j=1}^{J} c_j t^{j\alpha} + \sum_{j=1}^{\bar{J}} d_j t^{1+j\alpha}, \qquad J = \lceil 2/\alpha \rceil - 1, \tag{4}$$

with $\psi \in C^2[0,T]$ and $c_j, d_j \in \mathbb{R}$. The fractional powers account for the lack of smoothness near the initial instant. With small values of α, the function t^α exhibits high curvature around $t = 0$, such that its derivatives diverge as $t \to 0^+$. In a straightforward uniform-time approach, this can cause convergence in terms of fractional smoothness instead of the expected polynomial order.

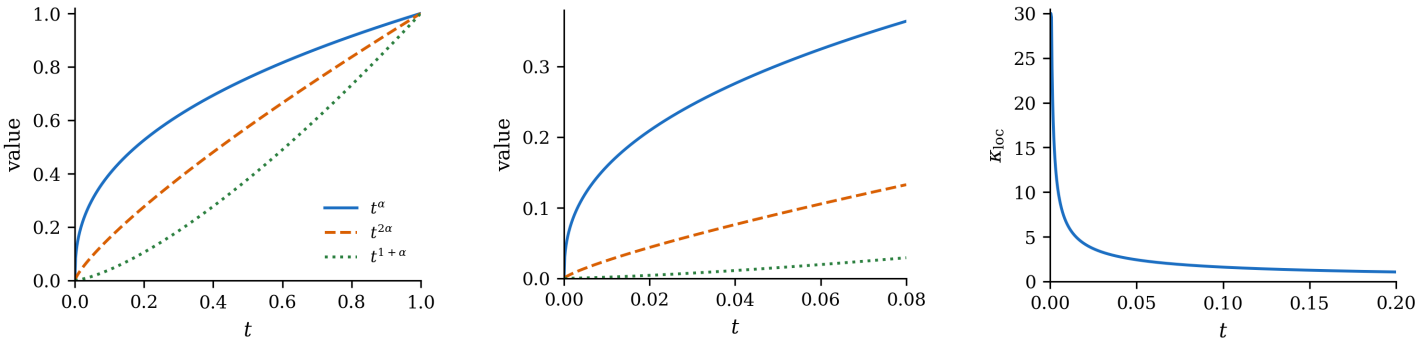

Figure 1: Initial-layer curvature for $\alpha = 0.4$.

Equation (2) is sufficiently general to describe both scalar problems and the spatial discretizations of fractional evolution equations. Upon the spatial discretization of a reaction–diffusion equation, the unknown becomes vector-valued, and the nonlinear term becomes composed of the diffusion matrices and the pointwise reaction terms. We use the scalar notation since the time discretization and memory properties are clearly identified within a single component. Analogously, the local properties of the solution and nonlinearities arise in each individual component, when the spatial part of the problem is discretized by means of the usual finite difference or finite element approach and when the corresponding local Lipschitz property holds.

It is worth mentioning here that the Caputo integral representation shows explicitly that the value y_0 is specified and not any fractional derivative. The singularity is introduced not through the initial condition but via the evolution law itself. This is a crucial issue to keep in mind while developing a solver, as a scheme which expects smooth time derivatives fails in spite of having smooth right-hand side and the classical initial value. In other words, the weak initial layer arises inherently in fractional differential equations.

The expansion (4) provides some hints on how the singularity can be approximated correctly. The powers $t^{j\alpha}$ provide information regarding possible singular derivatives and dominating powers within the memory error term. For α being close to zero, there exists significant curvature in the leading power. Due to the presence of the memory integral, the error caused locally at the beginning time points propagates rapidly to the later states. A good numerical integration rule should take care of these dominating powers in order to approximate them more accurately.

As shown in Figure 1, for $\alpha = 0.4$, the fast early growth of t^α produces the needed curvature. It should be noted that this term makes the interpolation of smooth functions impossible.

3. Algebraic Transformation of the Volterra Equation

The transformation

$$t = s^{\eta/\alpha}, \qquad \eta \in \mathbb{N}^+, \tag{5}$$

is introduced, and the transformed unknown is defined by

$$z(s) = y(s^{\eta/\alpha}). \tag{6}$$

Substitution of (5) into (3) gives

$$z(s) = y_0 + \frac{\eta}{\Gamma(1+\alpha)} \int_0^s \left(s^{\eta/\alpha} - r^{\eta/\alpha}\right)^{\alpha-1} r^{\eta/\alpha-1} f(r^{\eta/\alpha}, z(r))\, dr, \quad 0 \le s \le T^{\alpha/\eta}. \tag{7}$$

The Jacobian produces the factor $r^{\eta/\alpha-1}$. The memory kernel remains weakly singular, but the transformed unknown has improved behaviour near $s = 0$. Expansion (4) becomes

$$z(s) = \psi(s^{\eta/\alpha}) + \sum_{j=1}^{J} c_j s^{j\eta} + \sum_{j=1}^{\bar{J}} d_j s^{\eta(1+j\alpha)/\alpha}. \tag{8}$$

The term $t^{j\alpha}$ is therefore converted into $s^{j\eta}$. This is the algebraic reason that ordinary interpolation in s can approximate a solution whose physical-time representation is weakly regular.

Let

$$h = \frac{T^{\alpha/\eta}}{N}, \qquad s_i = ih, \qquad i = 0, 1, \ldots, N. \tag{9}$$

The numerical value z_i approximates $z(s_i)$. The transformed kernel appearing in (7) is

$$K_{\eta,\alpha}(s_i, r) = \left(s_i^{\eta/\alpha} - r^{\eta/\alpha}\right)^{\alpha-1} r^{\eta/\alpha-1}. \tag{10}$$

It can be seen from the expression $T^{\alpha/\eta}$ that the computation under transformation (7) is performed on another interval. With uniform sampling in the variable s, we have non-uniform sampling in the physical variable t. This clustering at the origin is a result of the transformation, but it does not fully explain why such high accuracy is achieved. The key point is that the transformed function has a new asymptotic behavior in the leading terms.

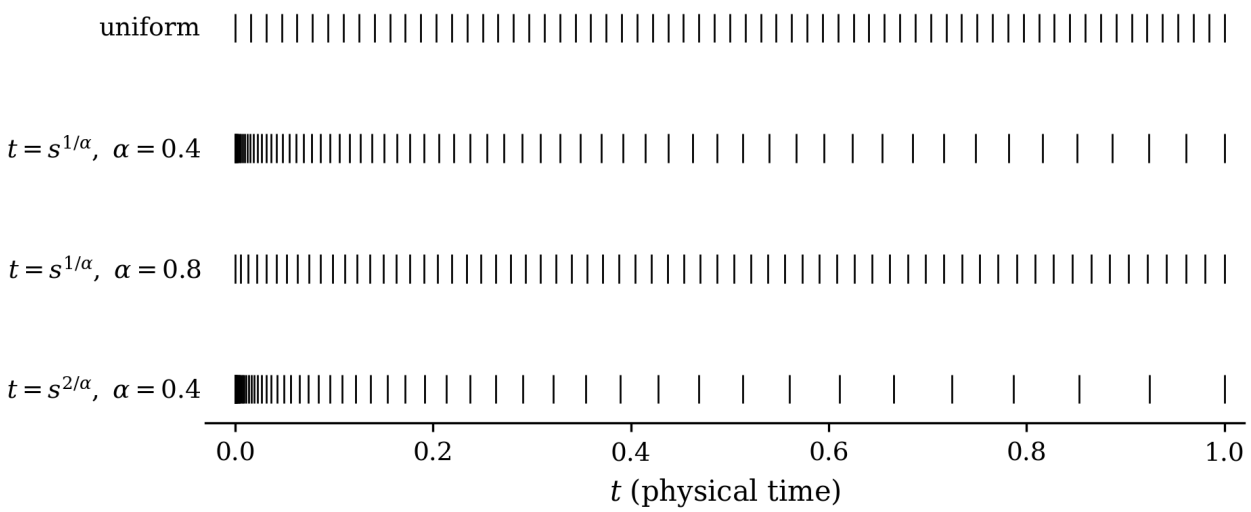

Figure 2: Physical-time nodes induced by algebraic regularization.

This is illustrated by the node distribution shown in Figure 2. Here the effect of the transformation on the node distribution is apparent; the nodes cluster in physical time near $t = 0$, with greater cluster density in the presence of a stronger singularity. The second-order case $\eta = 2$ shows even greater clustering. These pictures demonstrate the point analytically made earlier; the early-time layer of the Volterra integral equation will be handled by the method prior to inclusion in the memory terms.

The factor $J(s)$ appearing in Eq. (7) is crucial to the formulation. It is multiplied with the weak-kernel function $K_{\eta,\alpha}(s)$ in order to generate the weights in the product-integration quadrature formula. If omitted, the integral equation becomes inconsistent. The quadrature weights generated will not correspond to those of a simple rectangle- or trapezoidal-like rule with a transformed grid, since they arise from the weak kernel through the Volterra equation.

For $\eta = 1$, the leading singular power t^α reduces to s, making possible a first-order constant approximation in the interpolating polynomial. If $\eta = 2$, the reduction is to s^2, allowing linear interpolation and a second-order accurate approximation. Higher powers of η could be considered, but the two cases presented have already demonstrated the fundamental concept; the degree of transformation must be appropriate for the chosen interpolating order.

The product integration step is accomplished by integrating the product of the transformed weak kernel, $K_{\eta,\alpha}(s)$ and the local polynomial interpolation of the nonlinear factor. The singularity therefore appears only in the quadrature coefficient, not as a node value.

Figure 3 shows that while the recent past has significant weight, older states continue to be components of the Volterra memory. The coefficient matrix is lower-triangular in time, and the density of states around $r/s_i = 1$ accounts for the weak singularity of the Caputo kernel. The product integration formulas preserve the memory effect rather than smoothing out the integrand with a quadrature weight.

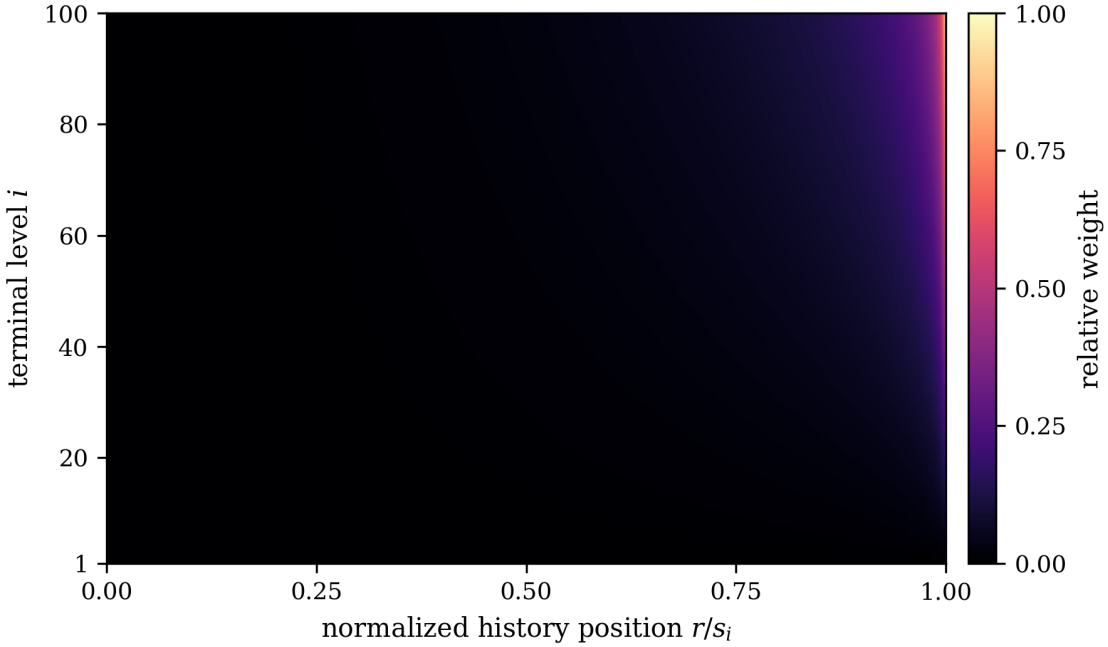

Figure 3: Transformed kernel mass for $\eta = 2$.

3.1 Product Rectangle Formulas

For the first-order schemes, $\eta = 1$. Equation (7) reduces to

$$z(s) = z_0 + \frac{1}{\Gamma(1+\alpha)} \int_0^s \left(s^{1/\alpha} - r^{1/\alpha}\right)^{\alpha-1} r^{1/\alpha-1} f(r^{1/\alpha}, z(r))\, dr. \tag{11}$$

For $1 \le m \le i$, define

$$a_m^i = \int_{s_{m-1}}^{s_m} \left(s_i^{1/\alpha} - r^{1/\alpha}\right)^{\alpha-1} r^{1/\alpha-1}\, dr. \tag{12}$$

The coefficient a_m^i measures the contribution of the m-th transformed-time interval to the value at s_i. The product-rectangle rule gives the implicit formula

$$z_i = z_0 + \frac{1}{\Gamma(1+\alpha)} \sum_{m=1}^{i} a_m^i f(s_m^{1/\alpha}, z_m). \tag{13}$$

The terminal value appears inside the nonlinear response, so a nonlinear equation may be solved at each time level. This treatment is accurate for strong nonlinear response because it does not delay the terminal state.

A delayed-state variant is

$$z_i = z_0 + \frac{1}{\Gamma(1+\alpha)} \sum_{m=1}^{i} a_m^i f(s_m^{1/\alpha}, z_{m-1}). \tag{14}$$

This construction decreases the cost of nonlinear evaluation by reusing the previous transformed-time value. The drawback is increased error constant due to the stronger influence of the memory effect. Both constructions use the piecewise constant representation of the nonlinear function since the nonlinear correction is given by a constant over the interval in either case.

The difference between the two formulae (13) and (14) is practical in addition to analytical. With (13), the terminal nonlinear value influences the calculation, hence requiring nonlinear solves at every time step. This cost may be tolerable for small systems and is advantageous for stiff nonlinear response. Formula (14), however, performs all evaluations at the previous transformed-time point. Thus, (14) does not require any terminal nonlinear solves; it is a simpler formula, but with decreased current information available. The numerical tests show that this leads primarily to higher error constant.

This example further shows why regularization alone does not provide second-order accuracy. Once the singular t^α has been transformed into s, the principal singularity disappears. But now, since the local representation of f is piecewise constant, the method cannot accurately resolve the transformed nonlinear response in its first derivative over the interval. Therefore, first-order accuracy is the correct result for the method, and the numerical experiments merely confirm this fact.

On the other hand, the construction of lower order is very helpful for practical implementation, when robustness and simplicity are prioritized over high-order accuracy. Second, this scheme gives us a baseline against which the accuracy of other methods can be assessed. The transformation $t = s^{1/\alpha}$ maps t^α into s, thus removing the initial singularity from the problem. As mentioned above, however, this singularity must be removed completely, i.e., using linear interpolation.

4. Second-Order Product-Trapezoidal Discretization

For the second-order schemes, $\eta = 2$. The transformed Volterra equation is

$$z(s) = z_0 + \frac{2}{\Gamma(1+\alpha)} \int_0^s \left(s^{2/\alpha} - r^{2/\alpha}\right)^{\alpha-1} r^{2/\alpha-1} f(r^{2/\alpha}, z(r))\, dr. \tag{15}$$

Let $\ell_m(r)$ denote the standard nodal hat function associated with s_m. The product-trapezoidal weights are

$$b_m^i = \int_0^{s_i} \left(s_i^{2/\alpha} - r^{2/\alpha}\right)^{\alpha-1} r^{2/\alpha-1} \ell_m(r)\, dr. \tag{16}$$

The fully implicit second-order formula is

$$z_i = z_0 + \frac{2}{\Gamma(1+\alpha)} \sum_{m=0}^{i} b_m^i f(s_m^{2/\alpha}, z_m). \tag{17}$$

The transform changes the leading power terms $t^{j\alpha}$ to s^{2j}, which means that the highest order terms will work well for the linear interpolation procedure. The weight b_m^i is the result of integrating the weak singularity with the nodal basis function, and hence the product integration technique is used instead of the usual trapezoidal rule.

A Newton-linearized terminal treatment is

$$\begin{aligned} z_i = z_0 &+ \frac{2}{\Gamma(1+\alpha)} \sum_{m=0}^{i-1} b_m^i f(s_m^{2/\alpha}, z_m) \\ &+ \frac{2}{\Gamma(1+\alpha)} b_i^i \left[f(s_i^{2/\alpha}, z_{i-1}) + f_2(s_i^{2/\alpha}, z_{i-1})(z_i - z_{i-1}) \right], \end{aligned} \tag{18}$$

where $f_2 = \partial f / \partial y$. This expression preserves the transformed memory weights and changes only the terminal nonlinear closure. It reduces the cost of the terminal nonlinear solve while retaining second-order temporal behaviour under the stated assumptions.

The extrapolated variant begins with

$$\begin{aligned} z_1 = z_0 &+ \frac{2}{\Gamma(1+\alpha)} b_0^1 f(s_0^{2/\alpha}, z_0) \\ &+ \frac{2}{\Gamma(1+\alpha)} b_1^1 \left[f(s_1^{2/\alpha}, z_0) + f_2(s_1^{2/\alpha}, z_0)(z_1 - z_0) \right], \end{aligned} \tag{19}$$

and for $i \geq 2$ uses

$$z_i = z_0 + \frac{2}{\Gamma(1+\alpha)} \sum_{m=0}^{i-1} b_m^i f(s_m^{2/\alpha}, z_m) + \frac{2}{\Gamma(1+\alpha)} b_i^i f(s_i^{2/\alpha}, 2z_{i-1} - z_{i-2}). \tag{20}$$

Extrapolated state $2z_{i-1} - z_{i-2}$ does not require a nonlinear terminal solve following initialization. While there may be more coarse-grid error compared to the implicit scheme, the transformed trapezoidal memory representation is still second order.

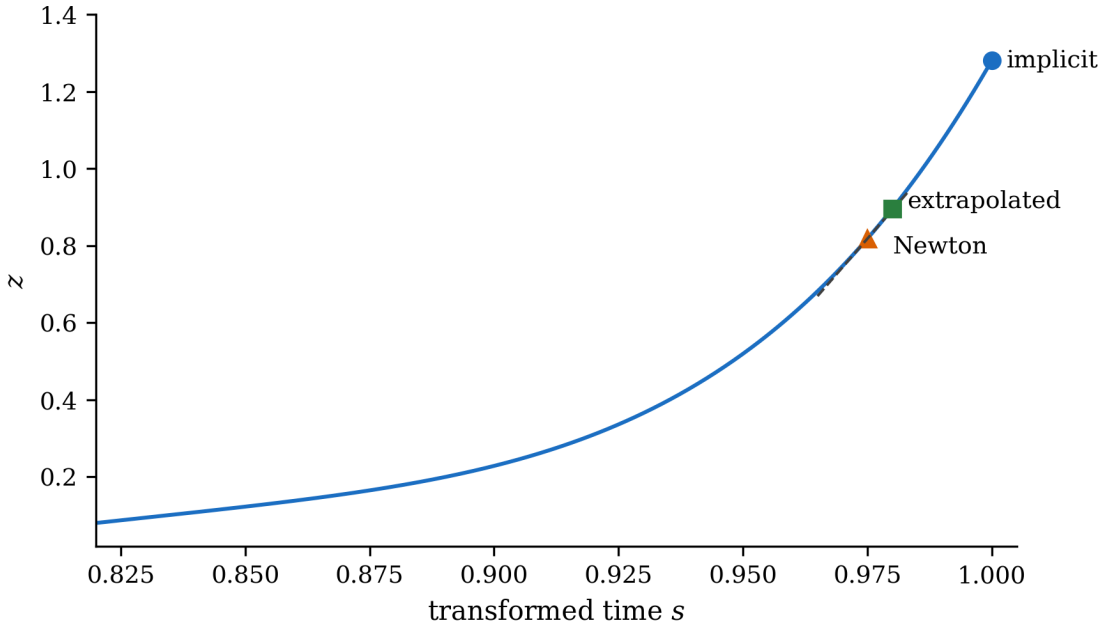

Figure 4: Terminal nonlinear closure in transformed time.

There is increased sensitivity to terminal nonlinear solves with these second-order methods since linear interpolation involves the nodal state. Implicit terminal state yields the most direct nonlinear term, while replacing this state through a Newton approximation or extrapolation results in an altered nonlinear terminal state using identical memory coefficients. Thus, the second-order schemes maintain the transformed-memory form but with varying computational cost per step.

The closure method presented in Figure 4 discriminates among the three second-order schemes based on their terminal-level closure behaviour. While the implicit approach relies directly on the current value, the Newton approach introduces a tangent correction, and the extrapolation approach uses the last two values. As these three approaches employ the same transformed memory weights, Figure 4 explains why their orders coincide asymptotically while their coarse-grid error constants do not match.

The benefit of using the Newton-linearized expression lies in retaining a local derivative of the nonlinear operator. That is relevant in case f_2 is available and the nonlinear solve can be reduced to a linear problem. The extrapolation formula remains cheaper since it does not involve even an evaluation of the derivative at the terminal point starting from the second step. Accuracy of this approximation relies upon the smoothness of the transformed solution between consecutive grid nodes. Numerical results support the predicted asymptotic order and verify that the implicit method yields more accurate error values than the other two approaches.

Thus, the only difference among the three second-order schemes is in treatment of the terminal nonlinear value. The formula for the fully implicit approach is straightforward. In contrast, the Newton-linearized formula reduces the nonlinear solve at the terminal point to a linear one. Finally, the extrapolation formula approximates the terminal value using the information about earlier values. Since all three formulas involve the same transformed product-trapezoidal weights, their asymptotic order is dictated by the transformed interpolation error.

5. Convergence Argument

Let

$$e_i = z(s_i) - z_i \tag{21}$$

denote the nodal error in transformed time. For the first-order implicit rectangle formula, subtraction of the numerical equation from the exact product-integral identity gives

$$|e_i| \le |e_0| + \frac{L}{\Gamma(1+\alpha)} \sum_{m=1}^{i} a_m^i |e_m| + |R_1^i|, \tag{22}$$

where L is the Lipschitz constant of f in its second argument and $R_1^i = O(h)$. For sufficiently small h, the terminal contribution can be absorbed into the left-hand side. The transformed weights satisfy, for $1 \le m \le i-1$,

$$a_m^i \le h\, m^{1/\alpha - 1} \left(i^{1/\alpha} - m^{1/\alpha} \right)^{\alpha - 1}. \tag{23}$$

The weakly singular form of this coefficient permits use of a discrete Gronwall inequality and yields

$$\max_{1 \le i \le N} |e_i| \le Ch. \tag{24}$$

The same first-order bound applies to the delayed-state rectangle formula because the nonlinear delay contributes a term of the same transformed-time order under the same smoothness assumptions.

For the second-order product-trapezoidal formula, the local remainder satisfies $R_2^i = O(h^2)$. The weights obey

$$b_0^i \le Ch^2, \tag{25}$$

and, for $1 \le m \le i-1$,

$$b_m^i \le Ch^2 \left(i^{2/\alpha} - m^{2/\alpha} \right)^{\alpha - 1} m^{2/\alpha - 1}. \tag{26}$$

The terminal term containing $b_i^i |e_i|$ is absorbed into the left-hand side for a sufficiently fine grid. The discrete Gronwall inequality then gives

$$\max_{1 \le i \le N} |e_i| \le Ch^2. \tag{27}$$

Newton linearization and extrapolation methods share the same order due to terminal substitution errors that correspond to transformed linear interpolation error order.

The values of these constants depend on T, α, the Lipschitz bound for f, and bounds for the regularity of the transformed exact solution. These constants do not depend on the number of the time-step once the grid becomes sufficiently fine. This

behavior agrees with the numerical experiments: with smaller values of α we observe higher error levels for coarse grids; however, the error rate converges towards the theoretical order for fine meshes.

This proof also explains why the terminal term should be dealt with separately. The terminal coefficient is multiplied by the current error term and therefore cannot be bounded by using an approach similar to the previous history terms. Bounding such a term on the left-hand side is the conventional procedure for solving Volterra equations with Lipschitz nonlinearity. After bounding the last term, the summation becomes a weakly singular sum that can be bounded with the help of the discrete Gronwall lemma. Thus, we obtain a stability result with the memory kernel.

It is clear that the proof of the convergence contains contributions of both the regularity and memory effect. On one hand, the transformation enhances the regularity of the unknown in the transformed time variable; on the other hand, the weights keep the weakly singular kernel in the desired form that enables summation estimates. Therefore, our approach does not treat the kernel as a smooth function and thus avoids physical-time smoothness which is not a property of our solution.

6. Numerical Results and Discussion

All calculations are carried out on $0 \leq t \leq 1$. The error in the maximum norm is provided for $\alpha = 0.4, 0.6, 0.8$. These parameters describe strong, moderate, and mild memory influence as the parameter tends to one. The numerical results are analyzed according to four aspects: the performance of the rectangle family of first-order methods, the order recovery of the trapezoidal family of second-order schemes, the degradation of the convergence rate for physical-time methods, and the compatibility of the transformed rule in time with a reaction–diffusion problem.

6.1 Scalar nonlinear Caputo reaction equation

The scalar test problem is

$$ {}^{C}D_t^{\alpha} u - (u^2 - u) = g(t), \qquad 0 < t \leq 1, \tag{28}$$

with exact solution

$$ u(t) = t + t^{\alpha}. \tag{29}$$

Since

$$ {}^{C}D_t^{\alpha} t = \frac{t^{1-\alpha}}{\Gamma(2-\alpha)}, \qquad {}^{C}D_t^{\alpha} t^{\alpha} = \Gamma(1+\alpha), \tag{30}$$

the forcing function is

$$ g(t) = \frac{t^{1-\alpha}}{\Gamma(2-\alpha)} + \Gamma(1+\alpha) - \left[(t+t^{\alpha})^2 - (t+t^{\alpha})\right]. \tag{31}$$

The scalar equation provides an additional test of nonlinear consistency as well. If we put $u(t) = t + t^{\alpha}$ into the nonlinear term, we generate terms of the form t^2, $t^{1+\alpha}$, and $t^{2\alpha}$, meaning that our forcing contains both a smooth term and a fractional one. In other words, this time we need to solve a problem with more than just the leading term. We should integrate our transformed kernel and evaluate the nonlinearity where individual terms evolve at different rates.

Therefore, our manufactured equation with a nonlinearity provides extra information compared to a simple manufactured linear equation.

We can interpret entries in the table by dividing the rate, magnitude, and nonlinear cost. First, rate shows whether we actually achieve the theoretical transformed interpolation order. Second, magnitude takes into account the initial layer strength (the α value itself), and the nonlinear treatment near the end of the domain. Finally, nonlinear cost depends on what type of value we use for our nonlinear evaluation at every step – current implicit, Newton expanded or extrapolated. Three values above are connected yet distinct. One algorithm can have the same rate as another one yet have larger errors or fewer nonlinear calculations per step.

Our manufactured solution consists of two parts playing different numerical roles. Our term t generates a smooth part with the Caputo derivative that equals $ct^{1-\alpha}$. Meanwhile, t^{α} generates the initial layer and the constant value of the Caputo derivative equals $\Gamma(1+\alpha)$. Coupling these two parts through the nonlinear term $u^2 - u$ makes sure that our problem cannot be reduced to a linear memory quadrature. As we know the exact solution in advance, any error in the tables below can be traced to a time discretization and nonlinear approximation.

Finally, using three different values of α is meaningful too. For example, at $\alpha = 0.4$ our singularity is particularly strong and the transformed grid needs to resolve fast evolution in the initial layer. Meanwhile, at $\alpha = 0.6$ our singularity becomes weaker while we still see the memory effect. Lastly, $\alpha = 0.8$ corresponds to a solution close to classical smoothness and better performance of the direct and transformed methods.

Thus, the variability of α helps us detect the true robustness to fractional regularity or good performance in the mild case.

This equation isolates our temporal singularity and includes the nonlinear reaction term.

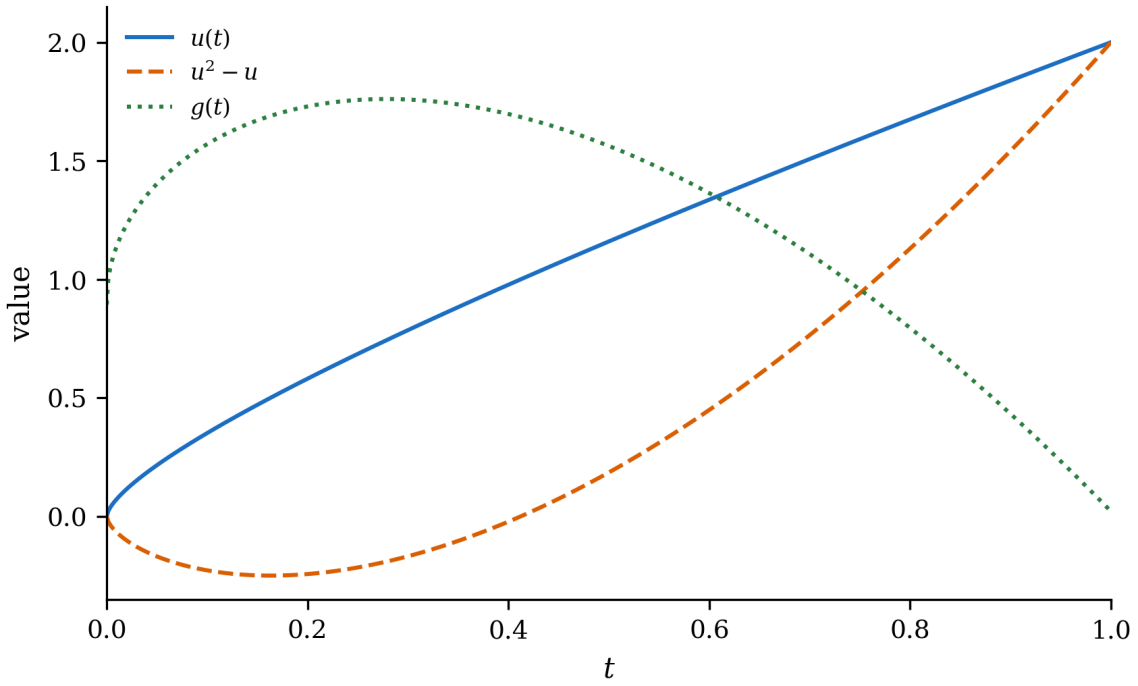

Figure 5: Scalar manufactured solution and forcing components.

It can be seen from Figure 5 that the problem of scalar verification involves both the singularity in the fractional response and the nonlinearity of the reaction term. It should be noted that the forcing is no arbitrary source term but the one necessary for the equation $u(t) = t + t^{\alpha}$. Thus, the error tabulations that follow are a real test of the transformation itself.

Table 1: Scheme I scalar errors.

N	$\alpha = 0.4$		$\alpha = 0.6$		$\alpha = 0.8$	
	Error	Order	Error	Order	Error	Order
1000	3.4639×10^{-1}	–	6.7655×10^{-3}	–	2.2863×10^{-3}	–
2000	1.3343×10^{-1}	1.3763	3.3359×10^{-3}	1.0201	1.8669×10^{-3}	1.0731
3000	8.3082×10^{-2}	1.1685	2.2133×10^{-3}	1.0120	7.0564×10^{-4}	1.0649
4000	6.0391×10^{-2}	1.1088	1.6559×10^{-3}	1.0086	5.2011×10^{-4}	1.0643

The errors for the implicit scheme from Table 1 exhibit consistent convergence as N increases. They reach maximum values when $\alpha = 0.4$, since this is when the solution has the lowest level of regularity in its fractional part. With $\alpha = 0.6$ and $\alpha = 0.8$, the orders do not deviate much from the value one. As regards $\alpha = 0.4$, the orders converge to one slower, since there is a stronger initial curvature.

Table 2: Scheme II scalar errors.

N	$\alpha = 0.4$		$\alpha = 0.6$		$\alpha = 0.8$	
	Error	Order	Error	Order	Error	Order
1000	4.3014×10^{-1}	–	2.7604×10^{-2}	–	7.1714×10^{-3}	–
2000	2.7749×10^{-1}	0.6324	1.4006×10^{-2}	0.9788	3.5936×10^{-3}	0.9968
3000	2.0582×10^{-1}	0.7368	9.3831×10^{-3}	0.9879	2.3972×10^{-3}	0.9985
4000	1.6379×10^{-1}	0.7941	1.6559×10^{-3}	1.0086	1.7983×10^{-3}	0.9991

The delayed state rectangle errors in Table 2 are higher than in Table 1. This trend is especially clear for $\alpha = 0.4$, where the nonlinear delayed evaluation does poorly in matching the initial quick variation of the exact solution. However, for $\alpha = 0.6$ and $\alpha = 0.8$, there is only an insignificant difference from first order. Thus, comparing the two rectangles, one sees that nonlinear end behavior influences the error constant and not the transformed order.

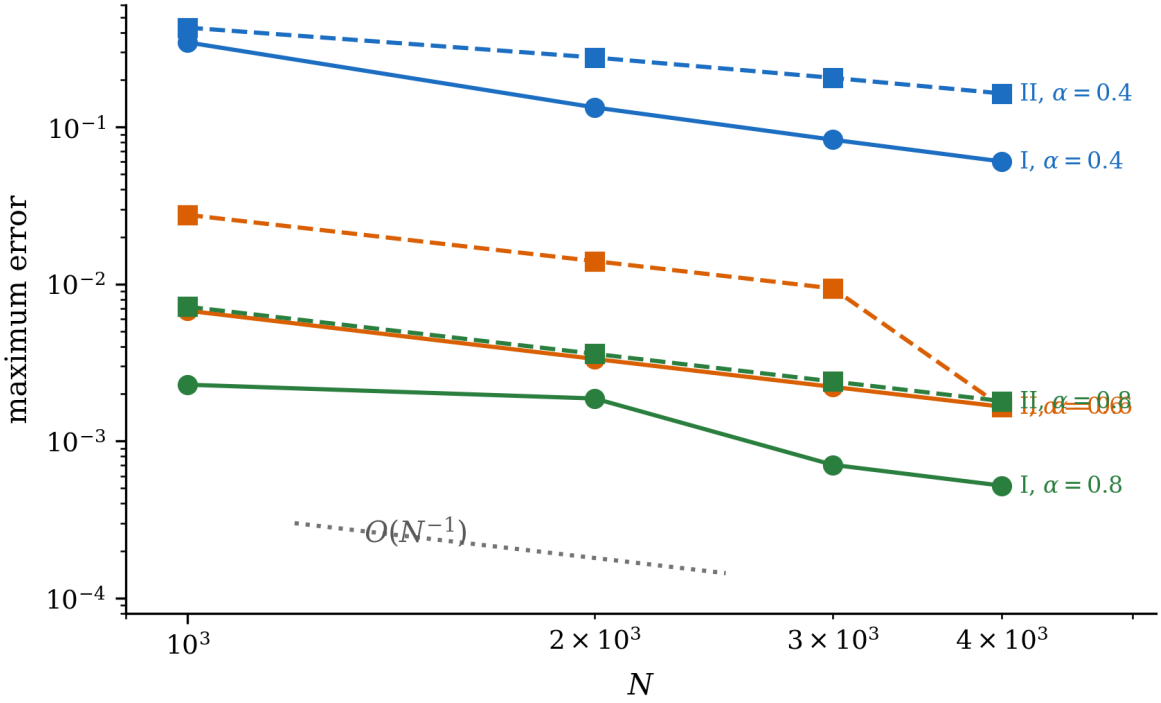

Figure 6: First-order rectangle-scheme convergence.

The first-order nature of the curves also shows that an absolute smaller error is not necessarily indicative of a more effective approach. Scheme I is normally below Scheme II since it uses the current nonlinear values, yet the two slopes are similar as soon as the asymptotic region is reached. Therefore, deciding among the two rectangle approaches can be reduced to choosing between efficiency and accuracy considerations. For example, in a big nonlinear system, Scheme II may turn out to be a favourable option if first-order dynamics in time is enough. In a stiff scalar ODE or strongly coupled system, Scheme I may be preferable because it gives a more current nonlinear evaluation.

The case $\alpha = 0.4$ requires a closer examination since the errors are significantly higher and the order of convergence is achieved later. However, this should not contradict the theoretical results. In such a way, while the transformed scheme modifies the main order, it does not alter the parameters of the estimates which depend on the fractional order and smoothness of the rest of the terms. Thus, the strong memory problem remains difficult even after formally achieving the original order.

Figure 6 demonstrates similar first-order convergence. Delayed-state approaches produce higher curves for the majority of the fractional orders due to less accurate current nonlinearity. The main slope will be still dictated by the product-rule construction with respect to the transformed time. Although the transformation weakens the influence of the singular layer well enough to restore the first-order, it does not reduce the error constant because of the lower order of the fractional derivative.

Table 3: Scheme III scalar errors.

N	$\alpha = 0.4$		$\alpha = 0.6$		$\alpha = 0.8$	
	Error	Order	Error	Order	Error	Order
100	1.8761×10^{-2}	–	7.2753×10^{-5}	–	2.6841×10^{-5}	–
200	3.7660×10^{-3}	2.3167	1.8112×10^{-5}	2.0060	6.6615×10^{-6}	2.0105
400	9.0495×10^{-4}	2.0571	4.5335×10^{-6}	1.9983	1.6600×10^{-6}	2.0046
800	2.2606×10^{-4}	2.0012	1.1354×10^{-6}	1.9974	4.1439×10^{-7}	2.0021

For the fully implicit trapezoidal scheme in Table 3, one can observe a clear order of two for all values of α, even for $\alpha = 0.4$. This is remarkable since the true solution has the expression t^α, which makes second-order convergence impossible for any direct smooth-time approximation. The key lies in the variable substitution, where $t = s^{2/\alpha}$, which transforms the dominant singular power from s^α to s^2.

Table 4: Scheme IV scalar errors.

N	$\alpha = 0.4$		$\alpha = 0.6$		$\alpha = 0.8$	
	Error	Order	Error	Order	Error	Order
100	1.5948×10^{-1}	–	2.2079×10^{-3}	–	2.4690×10^{-4}	–
200	3.1524×10^{-2}	2.3388	3.6106×10^{-4}	2.6123	3.8513×10^{-5}	2.6805
400	5.7871×10^{-3}	2.4455	5.8291×10^{-5}	2.6309	6.2528×10^{-6}	2.6228
800	1.0401×10^{-3}	2.4761	9.2534×10^{-6}	2.6552	1.0753×10^{-6}	2.5998

As seen in Table 4, the errors of the Newton-linearized scheme decrease quite fast upon refinement. In some cases, the orders seem to be higher than two on the listed meshes since they correspond to cancellation before reaching the asymptotic regime and not to any order that is formally above two. The errors for coarse grids are slightly larger than for the fully implicit case because the last step nonlinearity has been approximated locally around the previously computed solution.

Table 5: Scheme V scalar errors.

N	$\alpha = 0.4$		$\alpha = 0.6$		$\alpha = 0.8$	
	Error	Order	Error	Order	Error	Order
100	8.7281×10^{-2}	–	1.9757×10^{-3}	–	2.0034×10^{-4}	–
200	2.7388×10^{-2}	1.6721	3.2896×10^{-4}	2.5864	3.1694×10^{-5}	2.6602
400	6.2592×10^{-3}	2.1295	5.3119×10^{-5}	2.6302	5.2614×10^{-6}	2.5907
800	1.2079×10^{-3}	2.3735	8.3960×10^{-6}	2.6615	9.3199×10^{-7}	2.4971

The extrapolated errors shown in Table 5 confirm that the prediction from earlier points is consistent with the trapezoidal rule transformation. It is less smooth for the most singular problem on coarse meshes, but when refining, it pushes the errors to second-order magnitude. The procedure is valuable due to the reduction of nonlinear iterations required for convergence.

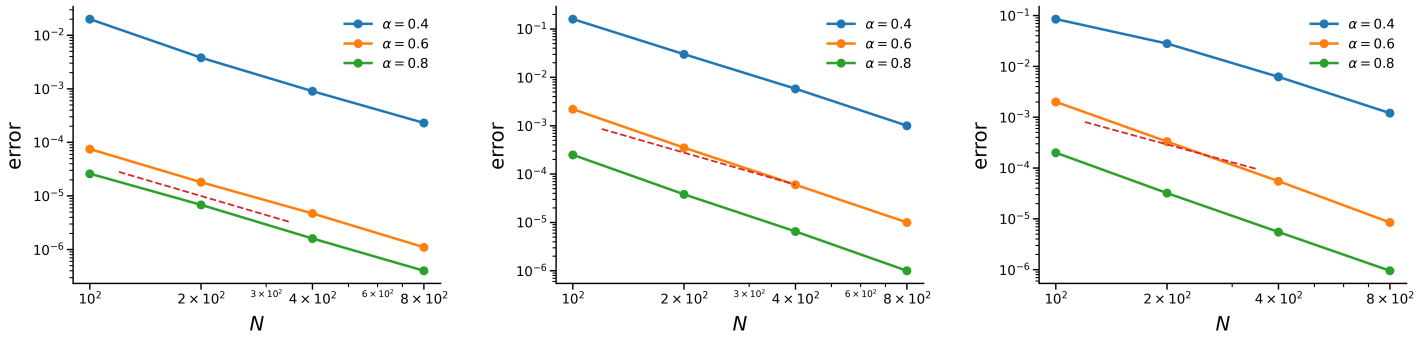

Figure 7: Second-order trapezoidal-scheme convergence.

The contrast between Tables 3, 4, and 5 demonstrates the role of nonlinear algebra in this computation. Scheme III provides the smallest and the most uniform errors due to its implicit evaluation of the last nonlinear term. Scheme IV relies on the derivative f_2 evaluated at the previous level and thus includes more local information than the pure extrapolation. Scheme V is the least expensive scheme after initialization and stays accurate also after refinement, although it suffers somewhat stronger pre-asymptotic effects for the first few refinement levels. These features are quite typical for nonlinear time stepping schemes: the memory quadrature defines the order of accuracy, and the terminal evaluation of nonlinear terms influences the constants.

The second-order behavior is the strongest proof that the transformation does something other than refine near the origin. In the case where there are additional physical-time nodes near $t = 0$, the accuracy order could be still restricted by the interpolation of a weakly regular function. In this particular case, the function in the quadrature variable was modified. The leading term here is quadratic in s if $\eta = 2$, and the linear product rule is used with the smooth response. The corresponding second-order behavior can be explained in such terms.

Figure 7 presents the effect of reducing the error constant on one hand, and that of recovering the slope on the other hand. The convergence behavior for the trapezoidal schemes in the transformed time is governed by the second-order convergence dictated by the regularity of the response in the new coordinates; no longer determined by the singular behavior in physical time.

6.2 Direct physical-time comparison

The scalar equation (28) is also solved using the same discretizations, but directly in the physical time, namely the Euler, L1, and CQBDF1 discretizations. This provides some insight into the convergence when there is no regularization of the

singularity prior to the quadrature. The direct formulas still converge, but their rates are controlled by the fractional smoothness of $u(t) = t + t^\alpha$.

Table 6: Direct physical-time errors.

Method	N	$\alpha = 0.4$		$\alpha = 0.6$		$\alpha = 0.8$	
		Error	Order	Error	Order	Error	Order
Euler	1000	5.8753×10^{-3}	–	2.5839×10^{-3}	–	5.9704×10^{-4}	–
Euler	2000	4.9689×10^{-3}	0.2418	1.7779×10^{-3}	0.5394	3.5089×10^{-4}	0.7668
Euler	4000	4.0802×10^{-3}	0.2843	1.2407×10^{-3}	0.5614	2.0410×10^{-4}	0.7817
Euler	8000	3.2811×10^{-3}	0.3145	8.0860×10^{-4}	0.5752	1.1806×10^{-4}	0.7898
L1	1000	8.3161×10^{-3}	–	2.8395×10^{-3}	–	6.2471×10^{-4}	–
L1	2000	6.4074×10^{-3}	0.3762	1.8880×10^{-3}	0.5888	3.5977×10^{-4}	0.7961
L1	4000	4.9231×10^{-3}	0.3802	1.2522×10^{-3}	0.5924	2.0696×10^{-4}	0.7977
L1	8000	3.7727×10^{-3}	0.3840	8.2909×10^{-4}	0.5949	1.1897×10^{-4}	0.7987
CQBDF1	10	6.6267×10^{-3}	–	1.5370×10^{-3}	–	1.8332×10^{-4}	–
CQBDF1	20	5.1092×10^{-3}	0.3752	1.0393×10^{-3}	0.5646	1.1208×10^{-4}	0.7099
CQBDF1	40	3.9323×10^{-3}	0.3810	6.9808×10^{-4}	0.5741	6.7741×10^{-5}	0.7264
CQBDF1	80	3.0031×10^{-3}	0.3856	4.6667×10^{-4}	0.5810	4.0584×10^{-5}	0.7391

However, the orders of the formulas using the direct approach resemble the fractional orders more than those of standard second-order methods. This pattern corresponds to the behavior in the case of an unresolved initial layer. Reducing the number of steps by increasing N does not eliminate the singularity at zero. Thus, the formula approximates a function of fractional regularity. On the other hand, the transformed formula uses the smoother function $t^\alpha s^{N-\alpha}$.

The comparison of the error in the physical-time coordinate is important from the point of view of explaining the phenomenon. It is tempting to believe that the main reason for the poor convergence rate is the nonlinearity of $u^2 - u$. However, the rates of the direct method are consistent with the fractional regularity. Moreover, this nonlinear differential equation transforms into a second order equation under the action of the transformed product-trapezoidal formula. The key is thus the interaction of the initial layer with the time variable used in approximation.

As seen from the values shown in Table 6, the order loss occurs because of the difference between the orders in direct and transformed methods. The Euler, L1 and CQBDF1 formulas demonstrate much worse rates than the trapezoidal scheme. They approximate the solution of the nonlinear equation sampled in the physical time coordinate, which has fractional regularity t^α. Reducing the step size decreases the error, but it does not affect the regularity of the function.

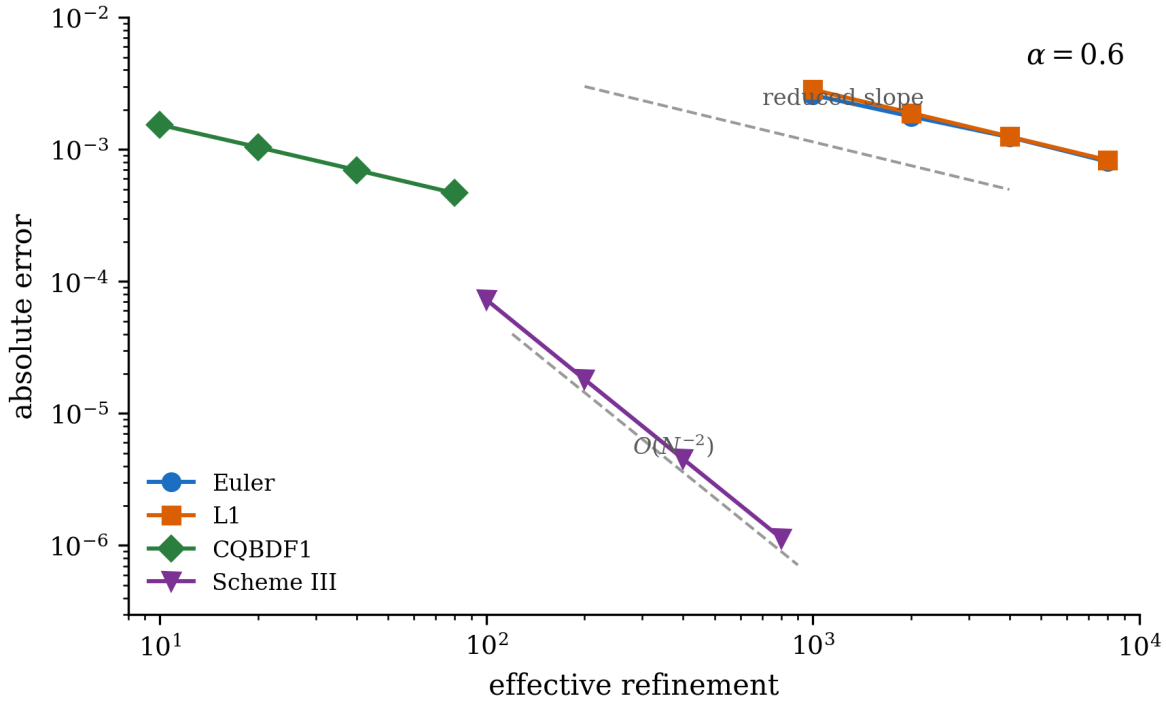

Figure 8: Direct physical-time comparison for $\alpha = 0.6$.

The physical-time comparison also explains why the transformed data do not necessarily indicate the result of using more or less levels. Indeed, the direct computations employ much larger values of N for both Euler and L1 schemes, whereas the slope of their approximation error stays low. In contrast, a transformed trapezoidal rule reaches higher slope using

fewer levels since the dominant singular term has been preprocessed. Thus, the key feature here is a dependent variable, whose smoothness is being improved.

It can be helpful in mesh selection. Suppose you employ the direct physical-time approximation and encounter slow convergence. You will most likely try to refine the mesh uniformly. It will help reduce the error but still not provide optimal efficiency due to approximation of the same weakly regular function on each step. The algebraic transformation leads to a more concentrated approximation around zero and, what is especially crucial, smoother leading initial powers. Now, a uniform refinement works with an easier time approximation.

As seen in Figure 8, the transformed second-order scheme falls into another convergence domain. In contrast to the direct curves with their shallow slopes, the transformed approximation has a very high slope. The conclusion here is that initial regularity, and not nonlinear reaction alone, sets the main restriction for the direct computation.

6.3 Fractional Allen–Cahn reaction–diffusion equation

The spatially distributed test problem is

$$ {}^{C}D_{t}^{\alpha}u - u_{xx} - (u - u^{3}) = g(x,t), \qquad (x,t) \in [0,\pi] \times (0,1], \tag{32} $$

with homogeneous boundary conditions. The forcing function is chosen so that

$$ u(x,t) = (t + t^{\alpha}) \sin x. \tag{33} $$

The Allen–Cahn equation may also prove relevant due to the requirement of long-time simulations in phase-field models. In a memory-dependent situation, an initial error in time will affect future interface amplitudes or chemical reaction balances. While the chosen manufactured solution has a particularly straightforward sinusoidal structure in the space variable, it nevertheless preserves the critical feature of time-fractional equations. The numerical method must maintain accuracy of the convolution as the diffusion and cubic reaction operators are calculated at nodal states. The achieved order of convergence in time reported in Table 7 thus demonstrates that the transformed weights can be applied beyond isolated ordinary differential equations.

The goal of the spatial test is similar. Here, one checks that the transformation of time does not negatively affect a classical process of spatial discretization. The second derivative of $\sin x$ is differentiable. According to theory, the central differences should yield a second-order spatial error. As shown in Table 8, the observed errors obey this order in all cases of time fractionality. This means that the temporal singularity does not spoil spatial convergence in this example problem.

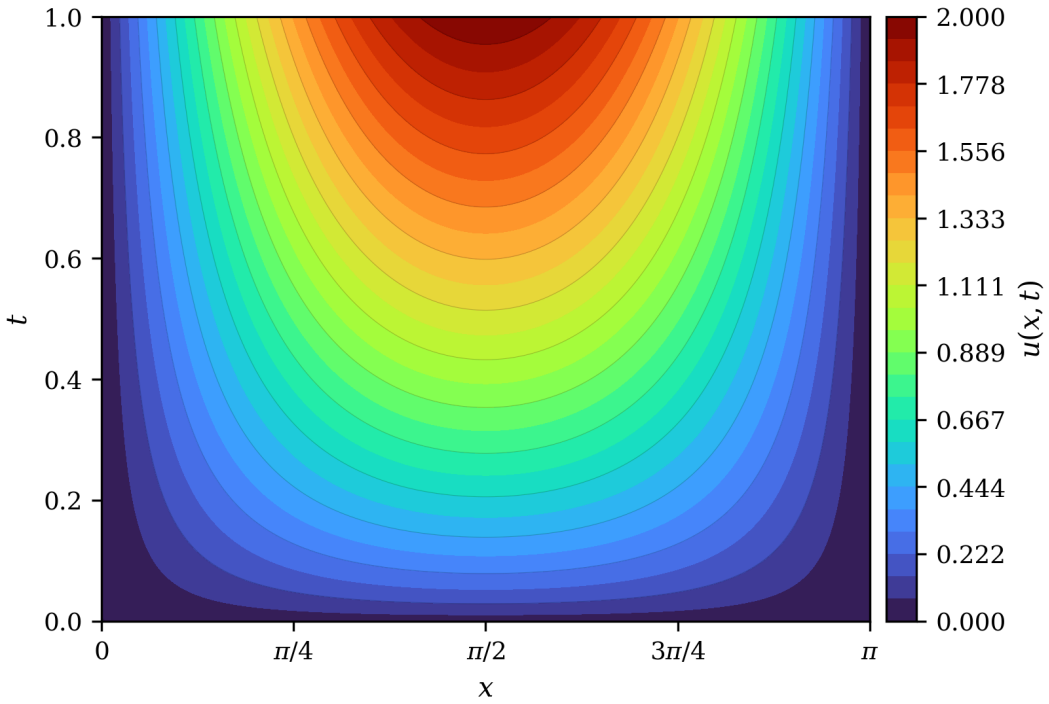

Figure 9: Manufactured Allen–Cahn space-time state.

As mentioned above, the Allen–Cahn equation is particularly challenging since it incorporates a memory operator, a spatial second derivative, and a cubic nonlinearity. The choice of a manufactured solution maintains the simplicity of

the spatial dependence but involves a reaction operator within the source term. Thus the test verifies that the temporal approximation performs well once the scalar equation was included in a reaction–diffusion model.

The errors presented in the table on temporal discretization come from sufficiently fine spatial mesh such that they reflect only the temporal contributions. Conversely, those in the spatial table were estimated on sufficiently refined temporal meshes, so that the pure spatial discretization error dominates. The distinction between time and space is crucial for this type of analysis; without it, a dominant temporal error would obscure the rate of spatial convergence and vice versa.

The spatial derivatives are computed using the common second-order central differences. The solution is differentiable in space and has a weak temporal singularity.

The space-time domain shown in Figure 9 represents the pattern of the constructed Allen-Cahn solution employed in the reaction-diffusion test. The Dirichlet conditions with zero values on both boundaries, $x = 0$ and $x = \pi$, remain fixed at all times, but the amplitude increases as $t + t^{\alpha}$. This enables us to distinguish between smoothness in space and weak regularity in time and treat the two separately.

Table 7: Allen–Cahn temporal errors.

N	$\alpha = 0.4$		$\alpha = 0.6$		$\alpha = 0.8$	
	Error	Order	Error	Order	Error	Order
8	1.5343×10^{-3}	–	3.4241×10^{-4}	–	7.3544×10^{-4}	–
16	4.3389×10^{-4}	1.8222	8.7884×10^{-5}	1.9459	1.6378×10^{-4}	2.1669
32	1.1709×10^{-4}	1.8897	2.2818×10^{-5}	1.9616	3.8450×10^{-5}	2.0907
64	3.0965×10^{-5}	1.9189	5.8943×10^{-6}	1.9527	9.2644×10^{-6}	2.0532

It can be noted from the results presented in Table 7 that the modified trapezoidal rule continues to demonstrate high accuracy even under the influence of diffusion and nonlinear reaction. Moreover, the convergence order is almost two for each of the considered fractional values. The slowest process in terms of convergence comes closest to an optimal value, similar to the scalar problem.

Table 8: Allen–Cahn spatial errors.

M	$\alpha = 0.4$		$\alpha = 0.6$		$\alpha = 0.8$	
	Error	Order	Error	Order	Error	Order
8	3.4258×10^{-3}	–	3.3368×10^{-3}	–	3.2843×10^{-3}	–
16	8.6220×10^{-4}	1.9903	8.4019×10^{-4}	1.9897	8.2728×10^{-4}	1.9891
32	2.1600×10^{-4}	1.9970	2.1043×10^{-4}	1.9974	2.0720×10^{-4}	1.9974
64	5.4114×10^{-5}	1.9969	5.2643×10^{-5}	1.9990	5.1812×10^{-5}	1.9997

Spatial errors presented in Table 8 are of second order and not significantly dependent on α. Indeed, such behavior was expected since the singular term is $t + t^{\alpha}$, but the spatial term $\sin(x)$ is differentiable. Thus, this example shows that there is a good separation of numerical phenomena. Memory integral is handled via the transformed product integration, and spatial derivative is controlled using central differences.

Temporal errors are larger for the highest memory power but stay within the second-order bounds of the transformed problem. This demonstrates that neither diffusion nor cubic reaction bring back the loss of order observed when directly discretizing the problem in physical time. The transformation is beneficial even for the time-discretization scheme. This result is of significance for practical applications because fractional memory laws are often coupled to transport and nonlinear reactions.

Spatial errors show no difference across memory powers since the exact spatial dependence remains the same $\sin(x)$ for all α. The effect of time-fractionality is to scale the temporal amplitude of the function, not its regularity in space. It follows that the choice of spatial approximation can be independent of the temporal one in this particular manufactured solution. In general geometries, an equivalent freedom could exist to combine a transformed time rule with various spatial schemes.

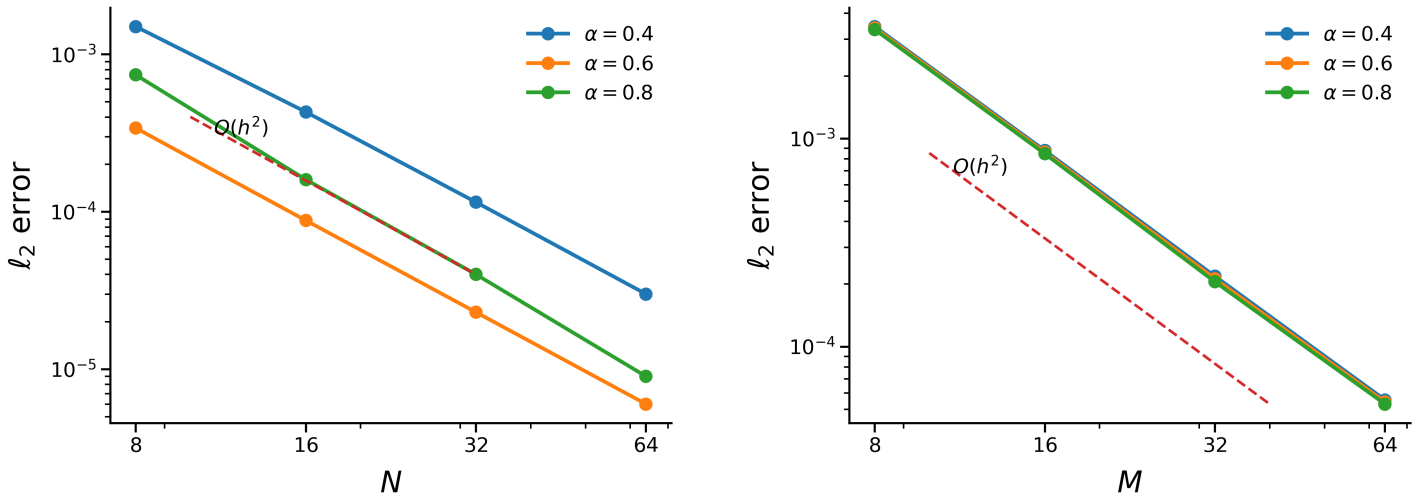

Figure 10: Separated Allen–Cahn temporal and spatial convergence.

Curves in Figure 10 reflect the trends already observed in Table **??**. In temporal direction, there is agreement with the pattern of the transformed second-order rule. In spatial direction, there is agreement with the central-difference second-order rule. Memory power influences only the magnitude of the temporal error while leaving the spatial convergence rate unchanged. This feature is significant for independent refinements of temporal memory and spatial transport in simulations.

7. Materials-Modelling Significance

Numerical problems analyzed above have the abstract form, yet they capture the essential mechanisms frequently encountered in materials modelling. Scalar Caputo equation may model the time-evolution of concentration, damage state, passive-film thickness, relaxation level, or reaction extent. Fractional Allen–Cahn equation may capture diffusion-reaction dynamics with nonlinearity and history effects. For both classes of models, initial evolution can affect the further history of the process due to presence of the Caputo memory integral.

In materials-degradation modelling, the initial deposition of a surface layer and initial redistribution of ionic concentration may influence the further transport resistance. In phase-field computations, the early evolution of the interface motion may impact the further morphology development due to memory effects. Direct time discretization will underestimate the early layer and carry the corresponding error over into the history integration. Product-quadrature schemes transform the integrand to avoid this issue.

The two examples illustrate the effects of the proposed schemes on the dynamics of materials where transient layers arise early in the process and affect subsequent diffusion properties or reaction rates. A passive interface may form rapidly within an early period of exposure and affect the resistance to subsequent diffusion processes. A reacting interface may move rapidly early on and become slowed due to memory effects on its kinetics. In such situations, a lower order or order-reduced discretization can affect later dynamics despite later changes being smooth. The transformed product rule was formulated precisely to address this case. The method captures a smoother initial behavior of the solution before the power influences accumulate through memory dependence.

The formulation also works well with conventional spatial discretizations. The example using the Allen-Cahn reaction–diffusion equation features central differences in space, but the scheme is independent of the specific spatial discretization technique. Similarly, the temporal scheme can be coupled to finite elements, finite volumes, or spectral methods, as long as the resulting nonlinearity fulfills proper Lipschitz and stability conditions. The method thus treats the memory effect in a modular way, controlling the temporal fraction while the spatial discretization is chosen for the particular problem physics.

The practical implication of this theoretical result concerns simulations in which time-discretization order is lost. The calculations demonstrate that a nonlinear memory differential equation of Caputo type is solvable to the right order of accuracy when the time coordinate is transformed prior to product integration. This is important in materials modeling and parameter studies because the loss of order affects the numerical simulation in a manner indistinguishable from physical sensitivity or model uncertainty.

8. Conclusion

Transformation of the Volterra time coordinate allows for product-integration schemes of the intended order to be applied to nonlinear Caputo differential equations. The substitution $t = s^{\eta/\alpha}$ changes the leading terms of $t^{j\alpha}$ to $s^{j\eta}$. The

interpolation is thus performed using a regularized version of the solution. The weak singularity remains in the integration kernel, ensuring the required memory of the Caputo differential equation.

The rectangle rules of first order, achieved by choosing $\eta = 1$, exhibit one-order temporal convergence. The implicit scheme is less sensitive to errors, whereas the delayed state rule reduces nonlinearity costs, yet increases the error constant. The trapezoidal schemes of second order, achieved by selecting $\eta = 2$, lead to the main accuracy result. All three versions: fully implicit, Newton-linearized, and extrapolated schemes recover second-order temporal accuracy for $\alpha = 0.4$, 0.6, and 0.8. The schemes based directly on the Euler, L1, and CQBDF1 approaches suffer order reduction due to their direct use of time steps.

The fractional Allen–Cahn problem demonstrates the utility of the method. The transformed temporal scheme maintains accuracy at second order, despite the added diffusion and reaction in the problem formulation. The spatial central difference yields second-order accuracy for the solution. Thus, the method is effective in handling nonlinear fractional problems that include initial-layer singularity, memory accumulation, and spatial transport.

For implementation purposes, the new weights have to be calculated only once, at the beginning of the calculation, given α, η, and a fixed time-grid size. The memory sum remains a history process, making the complexity increase with the number of steps. This problem is characteristic of most fractional time-discretization methods. The improved accuracy gained due to regularization complements fast memory computation schemes. Accuracy and computation can thus be enhanced separately.

As for the parameter η, it has to be chosen according to the needed order of the interpolant and the expected orders of singularities. The current results suggest that values of $\eta = 1$ and 2 are adequate for rectangle and trapezoidal formulas respectively. However, if a higher-order product integration is used, a higher η value might be desirable. Such choice will however affect the transformed weights and interval behavior. The present examples illustrate that the two lowest values are enough to observe the effectiveness of transformation for recovery of order.

The calculated scalar examples show that the key aspect of the proposed regularization takes place in the simplest setup possible. The rectangle formulas exhibit first-order convergence if transformed with $t = s^{1/\alpha}$. The trapezoidal methods produce second-order convergence if transformed with $t = s^{2/\alpha}$. The Allen–Cahn example demonstrates that this result is not constrained to a scalar equation. It shows that temporal convergence is recovered with the inclusion of diffusion and cubic reaction terms. Spatial convergence is defined by the central difference. The comparison to physical-time schemes reveals that the same exact solution produces lower order of convergence without transformation.

The results presented above confirm the conclusion that, for nonlinear fractional problems of Caputo type and with powers $t^{j\alpha}$ as initial singularities, a preliminary transformation of the time coordinate recovers the expected convergence rate. Transformation does not remove memory and leaves the influence of small α values unchanged. Nevertheless, it alters the problem sufficiently enough to recover the desired order of accuracy for the tested cases.

Possible extensions for future research include: adaptive choice of η, fast algorithms for transformed-memory sum, extension to multiple-dimensional reaction–diffusion systems, and incorporation into measurements-based parameter studies on memory-dependent degradation and corrosion.

Conflict of Interest

The author declares no conflict of interest.

References

[1] Oldham, K., & Spanier, J. (1974). The fractional calculus theory and applications of differentiation and integration to arbitrary order (Vol. 111). Elsevier.

[2] Podlubny, I. (1998). Fractional differential equations: an introduction to fractional derivatives, fractional differential equations, to methods of their solution and some of their applications (Vol. 198). elsevier.

[3] Hilfer, R. (Ed.). (2000). Applications of fractional calculus in physics. World scientific.

[4] Mainardi, F. (2010). Fractional calculus and waves in linear viscoelasticity, Imperial College Press, London.

[5] Metzler, R., & Klafter, J. (2000). The random walk's guide to anomalous diffusion: a fractional dynamics approach. Physics reports, 339(1), 1-77.

[6] Diethelm, K., & Ford, N. J. (2002). Analysis of fractional differential equations. Journal of Mathematical Analysis and Applications, 265(2), 229-248.

[7] Lubich, C. (1986). Discretized fractional calculus. SIAM Journal on Mathematical Analysis, 17(3), 704-719.

[8] Lubich, C. (1988). Convolution quadrature and discretized operational calculus. I. Numerische Mathematik, 52(2), 129-145.

[9] Cuesta, E., Lubich, C., & Palencia, C. (2006). Convolution quadrature time discretization of fractional diffusion-wave equations. Mathematics of Computation, 75(254), 673-696.

[10] Sun, Z. Z., & Wu, X. (2006). A fully discrete difference scheme for a diffusion-wave system. Applied Numerical Mathematics, 56(2), 193-209.

[11] Lin, Y., & Xu, C. (2007). Finite difference/spectral approximations for the time-fractional diffusion equation. Journal of computational physics, 225(2), 1533-1552.

[12] Jin, B., Lazarov, R., & Zhou, Z. (2013). Error estimates for a semidiscrete finite element method for fractional order parabolic equations. SIAM Journal on Numerical Analysis, 51(1), 445-466.

[13] Morgado, M. L., Ford, N. J., & Lima, P. M. (2013). Analysis and numerical methods for fractional differential equations with delay. Journal of computational and applied mathematics, 252, 159-168.
[14] Ji, C. C., & Sun, Z. Z. (2015). A high-order compact finite difference scheme for the fractional sub-diffusion equation. Journal of Scientific Computing, 64(3), 959-985.
[15] Dixon, J., & McKee, S. (1986). Weakly singular discrete Gronwall inequalities. ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik, 66(11), 535-544.
[16] Cao, W., Zeng, F., Zhang, Z., & Karniadakis, G. E. (2016). Implicit-explicit difference schemes for nonlinear fractional differential equations with nonsmooth solutions. SIAM Journal on Scientific Computing, 38(5), A3070-A3093.
[17] Jin, B., Li, B., & Zhou, Z. (2018). Numerical analysis of nonlinear subdiffusion equations. SIAM Journal on Numerical Analysis, 56(1), 1-23.
[18] Liao, H. L., McLean, W., & Zhang, J. (2019). A discrete Gronwall inequality with applications to numerical schemes for subdiffusion problems. SIAM Journal on Numerical Analysis, 57(1), 218-237.
[19] Chen, X., Di, Y., Duan, J., & Li, D. (2018). Linearized compact ADI schemes for nonlinear time-fractional Schrödinger equations. Applied Mathematics Letters, 84, 160-167.
[20] Tang, T., Yu, H., & Zhou, T. (2019). On energy dissipation theory and numerical stability for time-fractional phase-field equations. SIAM Journal on Scientific Computing, 41(6), A3757-A3778.
[21] Du, Q., Yang, J., & Zhou, Z. (2020). Time-fractional Allen–Cahn equations: analysis and numerical methods. Journal of Scientific Computing, 85(2), 42.
[22] Zhou, B., Chen, X., & Li, D. (2020). Nonuniform Alikhanov linearized Galerkin finite element methods for nonlinear time-fractional parabolic equations. Journal of Scientific Computing, 85(2), 39.
[23] Cen, D., Wang, Z., & Mo, Y. (2021). Second order difference schemes for time-fractional KdV–Burgers' equation with initial singularity. Applied Mathematics Letters, 112, 106829.
[24] Zhang, Y. N., Sun, Z. Z., & Liao, H. L. (2014). Finite difference methods for the time fractional diffusion equation on non-uniform meshes. Journal of Computational Physics, 265, 195-210.
[25] Alikhanov, A. A. (2015). A new difference scheme for the time fractional diffusion equation. Journal of Computational Physics, 280, 424-438.
[26] Stynes, M., O'Riordan, E., & Gracia, J. L. (2017). Error analysis of a finite difference method on graded meshes for a time-fractional diffusion equation. SIAM Journal on Numerical Analysis, 55(2), 1057-1079.
[27] Jiang, S., Zhang, J., Zhang, Q., & Zhang, Z. (2017). Fast evaluation of the Caputo fractional derivative and its applications to fractional diffusion equations. Communications in Computational Physics, 21(3), 650-678.
[28] Liao, H. L., Li, D., & Zhang, J. (2018). Sharp error estimate of the nonuniform L1 formula for linear reaction-subdiffusion equations. SIAM Journal on Numerical Analysis, 56(2), 1112-1133.
[29] Kopteva, N. (2019). Error analysis of the L1 method on graded and uniform meshes for a fractional-derivative problem in two and three dimensions. Mathematics of Computation, 88(319), 2135-2155.
[30] Li, D., Sun, W., & Wu, C. (2021). A novel numerical approach to time-fractional parabolic equations with nonsmooth solutions. Numerical Mathematics: Theory, Methods and Applications, 14(2), 355-376.
[31] Qin, H., Li, D., & Zhang, Z. (2021). A novel scheme to capture the initial dramatic evolutions of nonlinear subdiffusion equations. Journal of Scientific Computing, 89(3), 65.