fractal and fractional

MDPI

Article

Numerical Solution of Advection–Diffusion Equation of Fractional Order Using Chebyshev Collocation Method

Farman Ali Shah [1], Kamran [1,*], Wadii Boulila [2,3,*], Anis Koubaa [2] and Nabil Mlaiki [4]

[1] Department of Mathematics, Islamia College Peshawar, Jamrod Road, Peshawar 25120, Khyber PakhtunKhwa, Pakistan
[2] Robotics and Internet-of-Things Laboratory, Prince Sultan University, Riyadh 11586, Saudi Arabia; akoubaa@psu.edu.sa
[3] RIADI Laboratory, National School of Computer Sciences, University of Manouba, Manouba 2010, Tunisia
[4] Department of Mathematics and Sciences, Prince Sultan University, Riyadh 11586, Saudi Arabia; nmlaiki@psu.edu.sa or nmlaiki2012@gmail.com
* Correspondence: kamran.maths@icp.edu.pk (K.); wboulila@psu.edu.sa (W.B.)

Abstract: This work presents a highly accurate method for the numerical solution of the advection–diffusion equation of fractional order. In our proposed method, we apply the Laplace transform to handle the time-fractional derivative and utilize the Chebyshev spectral collocation method for spatial discretization. The primary motivation for using the Laplace transform is its ability to avoid the classical time-stepping scheme and overcome the adverse effects of time steps on numerical accuracy and stability. Our method comprises three primary steps: (i) reducing the time-dependent equation to a time-independent equation via the Laplace transform, (ii) employing the Chebyshev spectral collocation method to approximate the solution of the transformed equation, and (iii) numerically inverting the Laplace transform. We discuss the convergence and stability of the method and assess its accuracy and efficiency by solving various problems in two dimensions.

Keywords: advection–diffusion equation; Caputo derivative; Laplace transform; Chebyshev spectral collocation method; Stehfest's method; Talbot's method

Citation: Ali Shah, F.; Kamran; Boulila, W.; Koubaa, A.; Mlaiki, N. Numerical Solution of Advection–Diffusion Equation of Fractional Order Using Chebyshev Collocation Method. *Fractal Fract.* **2023**, *7*, 762. https://doi.org/10.3390/fractalfract7100762

Academic Editors: Jordan Hristov and Dimiter Prodanov

Received: 6 September 2023
Revised: 9 October 2023
Accepted: 13 October 2023
Published: 17 October 2023

1. Introduction

Fractional calculus (FC) is the generalization of classical calculus, which includes integrals and derivatives in non-integer order [1,2]. In recent years, the concept of fractional derivatives has been applied with great success to model various real-life phenomena in many scientific fields [3]. Fractional order operators include the history of a physical phenomenon from the initial state to the current state. Therefore, fractional order operators are often applied to model systems that describe the influence of memory effects [4]. Fractional order operators have broadened the concept of integer order operators [5]. In the last ten years, PDEs of fractional order have attracted the remarkable attention of the research community due to their ability to model various problems, such as neural networks [6], optimal voltage controllers for electronic vehicle charging stations [7], robotics [8], optimal controls for impulsive systems [9], hydrology [10], medical sciences [11], and many others [12].

The time-fractional advection–diffusion equations (TFADEs) have been widely used in many fields of science because of their ability to model memory and non-local properties [13]. ADEs combine diffusion and advection terms to describe physical events in which particles, energy, or other physical quantities are moved inside a physical system by two processes, i.e., diffusion and advection. The concentration of substances for mass and heat transfer is represented by the advection and diffusion models. This equation has been used to model heat transfer in draining films [14], flow in porous media [15], and air pollution [16], etc. There have been many attempts to solve the TFADEs accurately.

In [17], a second-order implicit difference method was proposed for solving a time–space-fractional ADE. Mardani et al. [18] studied a meshless method for the solution of ADEs with variable coefficients. A fully implicit finite difference scheme based on cubic B-splines was developed in [19] for solving TFADEs. Abbaszadeh and Amjadian [20] studied the spectral element method for the solution of TFADEs. In the current study, we investigate the numerical solution of two-dimensional TFADEs using the Chebyshev spectral collocation method (CSCM) coupled with the Laplace transform (LT).

Spectral collocation methods have attracted the attention of the research community due to their highly accurate solutions to partial differential equations. They converge exponentially fast as compared to the algebraic convergence rates for FEM and FDM. Spectral methods are global in nature. This means that the optimal accuracy can be obtained with few grid points. For the first time, spectral methods were introduced by Gottlieb and Orszag [21]. Their work paved the way for more advanced extensions of the method for a larger variety of problems [22,23]. The most important tool for spectral methods is the computation of differentiation matrices, discussed briefly by Welfert [24]. The spectral collocation methods have been used by many authors. For example, in [25], the authors discussed the Hermite spectral methods for unbounded domains. The solution of the fractional advection–dispersion equation by the Chebyshev spectral method was incorporated by Sweilam et al. [26]. In [27], the spectral Legendre–Chebyshev collocation method for variable coefficients was examined. Tian et al. [28] investigated the polynomial spectral collocation method for space-fractional ADEs. The spectral collocation method and the non-standard finite difference technique for the solution of TFADEs were introduced by [29].

In the aforementioned methods, the time discretization is carried out using the finite difference time-stepping method. The main shortcoming of the time-stepping technique is that it may not always result in a stable solution. A finite difference time-stepping scheme is stable if the errors calculated at a one-time step do not cause the errors to be increased during the calculations. In other words, the time-stepping scheme is stable whether the errors remain constant or decay during the computations. Furthermore, for optimal accuracy, we need to decrease the time step, which results in an increased computational time, and, thus, has low efficiency in simulating the time-fractional problems. To overcome this drawback of the finite difference time-stepping scheme, the LT has been used as one of the best alternative mathematical tools [30,31]. However, while using the LT, the main difficulty is its inversion. In the literature, many algorithms have been developed for the numerical inversion of the LT [32–34]. In the current study, we use Stehfest's method [35] and Talbot's method [32,36] for the numerical inversion of the LT.

2. Numerical Scheme

In our numerical scheme, a two-dimensional TFADE is taken into account. First, the LT is employed to reduce the TFADE to a time-independent problem in the LT domain. Then, the CSCM is employed to obtain the approximate solution to the transformed problem. Finally, the time domain solution is retrieved from the LT domain solution via the inverse LT.

2.1. Two-Dimensional TFADE

We consider a two-dimensional time-fractional advection–diffusion equation of the form

$$D_\tau^\alpha U(\bar{\mathbf{x}},\tau) = \lambda_1 \Delta U(\bar{\mathbf{x}},\tau) + (\overrightarrow{\mathbf{u}} \cdot \nabla) U(\bar{\mathbf{x}},\tau) - \lambda_2 U(\bar{\mathbf{x}},\tau) + \mathcal{Q}(\bar{\mathbf{x}},\tau),\ \bar{\mathbf{x}} \in \Omega,\ 0 \le \tau \le 1, \quad (1)$$

with boundary conditions

$$U(\bar{\mathbf{x}},\tau) = h(\bar{\mathbf{x}},\tau),\ \bar{\mathbf{x}} \in \partial\Omega,\ 0 \le \tau \le 1, \quad (2)$$

and the initial condition

$$U(\bar{\mathbf{x}},0) = U_0(\bar{\mathbf{x}}),\ \bar{\mathbf{x}} \in \Omega, \quad (3)$$

where $\overrightarrow{\mathbf{u}} = (u_1, u_2)$ is a known vector, λ_1 and λ_2 are arbitrary constants, $\mathcal{Q}(\bar{\mathbf{x}}, \tau)$, $U_0(\bar{\mathbf{x}})$, $h(\bar{\mathbf{x}}, \tau)$ are given continuous functions, $\Omega \subset \mathbb{R}^2$ is a bounded domain with a smooth boundary $\partial\Omega$, and D_τ^α represents Caputo's derivative, defined as [3]

$$D_\tau^\alpha U(\bar{\mathbf{x}}, \tau) = \frac{1}{\Gamma(1-\alpha)} \int_0^\tau \frac{\partial U(\bar{\mathbf{x}}, z)}{\partial z} (\tau - z)^{-\alpha} dz, \ 0 < \alpha \leq 1.$$

2.2. Time Discretization

In this section, the LT is used to reduce the TFADE to an equivalent time-independent problem in the LT domain. The LT of the function $U(\bar{\mathbf{x}}, \tau)$ is denoted by $\widehat{U}(\bar{\mathbf{x}}, s)$ and is defined as

$$\widehat{U}(\bar{\mathbf{x}}, s) = \int_0^\infty e^{-s\tau} U(\bar{\mathbf{x}}, \tau) d\tau,$$

and the LT of Caputo's derivative is defined as

$$\mathcal{L}\{D_\tau^\alpha U(\bar{\mathbf{x}}, \tau)\} = s^\alpha \widehat{U}(\bar{\mathbf{x}}, s) - s^{\alpha-1} U_0.$$

Applying the LT to (1)–(3) gives

$$s^\alpha \widehat{U}(\bar{\mathbf{x}}, s) - s^{\alpha-1} U_0 = \lambda_1 \Delta \widehat{U}(\bar{\mathbf{x}}, s) + (\overrightarrow{\mathbf{u}} \cdot \nabla) \widehat{U}(\bar{\mathbf{x}}, s) - \lambda_2 \widehat{U}(\bar{\mathbf{x}}, s) + \widehat{\mathcal{Q}}(\bar{\mathbf{x}}, s), \tag{4}$$

and

$$\widehat{U}(\bar{\mathbf{x}}, s) = \widehat{h}(\bar{\mathbf{x}}, s), \tag{5}$$

Equations (4) and (5) can be rearranged as

$$(s^\alpha I - \mathbf{L}) \widehat{U}(\bar{\mathbf{x}}, s) = \widehat{G}(\bar{\mathbf{x}}, s), \tag{6}$$

and

$$\widehat{U}(\bar{\mathbf{x}}, s) = \widehat{h}(\bar{\mathbf{x}}, s). \tag{7}$$

where $\mathbf{L} = \lambda_1 \Delta + (\overrightarrow{\mathbf{u}} \cdot \nabla) - \lambda_2 I$ is a linear differential operator, $\widehat{G}(\bar{\mathbf{x}}, s) = s^{\alpha-1} U_0 + \widehat{\mathcal{Q}}(\bar{\mathbf{x}}, s)$, and I denotes the identity operator. To obtain the approximate solution of the system defined in (6) and (7), first, the linear spatial operator $\mathbf{L}$ is discretized using the CSCM. Then, the fully discrete system is solved for each node s in the LT domain. Finally, the numerical inversion of the LT is performed to obtain the solution of the problem defined in Equations (1)–(3).

2.3. Chebyshev Spectral Collocation Method

In the CSCM, we interpolate the data $\left\{ \left(x_j, \widehat{U}(x_j) \right) \right\}$ by using the Lagrange interpolation polynomial (LIP) $l_j(x)$ of a degree of at most n [23,37]

$$\mathcal{I}_n(x) = \sum_{j=0}^{n} l_j(x) \widehat{U}_j$$

where $l_j(x)$ is the LIP at the point $x_j (j = 0, 1, \ldots, n)$, given as

$$l_j(x) = \frac{(x - x_0) \ldots (x - x_{j-1})(x - x_{j+1}) \ldots (x - x_n)}{(x_j - x_0) \ldots (x_j - x_{j-1})(x_j - x_{j+1}) \ldots (x_j - x_n)} \tag{8}$$

where $\widehat{U}_j = \widehat{U}(x_j)$. In the CSCM, the Chebyshev nodes are selected as the interpolation nodes. The domain $[-1, 1]$ is discretized using the Chebyshev points defined as

$$x_j = \cos\left(\frac{j\pi}{n}\right), \ \ j = 0, 1, \ldots, n. \tag{9}$$

The approximation of the first derivative $\frac{\partial \widehat{U}(x)}{\partial x}$ on the Chebyshev points can be obtained as

$$\frac{\partial \widehat{U}(x)}{\partial x} \approx \mathbf{D}_n \widehat{U},$$

where the entries of the matrix $\mathbf{D}_n$ are

$$[\mathbf{D}_n]_{i,j} = l_j'(x_i), \ i, j = 1, 2, \ldots, n.$$

The off-diagonal entries of $[\mathbf{D}_n]_{i,j}$ are obtained as

$$[\mathbf{D}_n]_{i,j} = \frac{\alpha_j}{\alpha_i(x_i - x_j)}, \ i \neq j,$$

where $\alpha_j^{-1} = \prod_{i \neq j}^{n}(x_i - x_j)$, and the diagonal elements are obtained as

$$[\mathbf{D}_n]_{i,j} = -\sum_{j=0,\ j\neq i}^{n} [\mathbf{D}_n]_{i,j}, \ i = 0, 1, 2, .., n.$$

Next, the entries of the μth-order differentiation matrix $\mathbf{D}_n^{\mu}$ can be obtained analytically as

$$[\mathbf{D}_n^{\mu}]_{i,j} = l_j^{\mu}(x_i), \ i, j = 1, 2, \ldots, n.$$

A more accurate and stable evaluation can be found in [24,38]. In [24], the author derived a useful recursion relation for the differentiation matrices, given as

$$[\mathbf{D}_n^{\mu}]_{kj} = \frac{\mu}{x_k - x_j}\left(\frac{\alpha_j}{\alpha_k}[\mathbf{D}_n^{(\mu-1)}]_{kk} - [\mathbf{D}_n^{(\mu-1)}]_{kj}\right), \ k \neq j$$

For square domain $[-1, 1]^2$, assume the points $\bar{\mathbf{x}}_{ij}$ are defined as

$$\bar{\mathbf{x}}_{ij} = \left(\cos\left(\frac{i\pi}{n}\right), \cos\left(\frac{j\pi}{n}\right)\right), \ i, j = 0, 1, 2, \ldots, n$$

The LIP corresponding to the above Chebyshev points are given as

$$l_{ij}(\bar{\mathbf{x}}) = l_i(x) l_j(y), \tag{10}$$

where $l_{ij}(\bar{\mathbf{x}}_{ij}) = \delta_{ij}, \ i, j = 0, 1, 2, \ldots, n$. In terms of x and y, the second derivatives of the LIPs (10) are given as

$$\frac{\partial^2 l_{ij}(\bar{\mathbf{x}}_{rs})}{\partial x^2} = l_i''(x_r) l_j(y_s) = [\mathbf{D}_n^2]_{ri}\delta_{js},$$
$$\frac{\partial^2 l_{ij}(\bar{\mathbf{x}}_{rs})}{\partial y^2} = l_i(x_r) l_j''(y_s) = \delta_{ri}[\mathbf{D}_n^2]_{sj},$$

where $\mathbf{D}_n^2$ denotes the second-order Chebyshev differentiation matrix. Applying the linear operator $\mathbf{L}$ on the LIP at the Chebyshev nodes, $\{\bar{\mathbf{x}}_{rs}\}$ gives

$$\mathbf{L}(l_{ij}(\bar{\mathbf{x}}_{rs})) = \lambda_1\left([\mathbf{D}_n^2]_{ri}\delta_{js} + \delta_{ri}[\mathbf{D}_n^2]_{sj}\right) + \left(u_1\left([\mathbf{D}_n]_{ri}\delta_{js}\right) + u_2\left(\delta_{ri}[\mathbf{D}_n]_{sj}\right)\right) - \lambda_2\delta_{ri}\delta_{sj}, \tag{11}$$

Therefore, the discretized form of the linear differential operator $\mathbf{L}$ via the CSCM can be expressed as

$$\mathbf{L_D} = \lambda_1\Big(I_n \otimes \mathbf{D}_n^2 + \mathbf{D}_n^2 \otimes I_n\Big) + \Big(u_1\Big(I_n \otimes \mathbf{D}_n\Big) + u_2\Big(\mathbf{D}_n \otimes I_n\Big)\Big) - \lambda_2\Big(I_n \otimes I_n\Big). \quad (12)$$

By using Equation (12) in Equation (6), we have

$$(s^\alpha I - \mathbf{L_D})\widehat{U}(\bar{\mathbf{x}}, s) = \widehat{G}(\bar{\mathbf{x}}, s), \quad (13)$$

where we incorporate the boundary conditions in Equation (7) by taking the matrix $\mathbf{L_D}$, based on all collocation points $\bar{\mathbf{x}}$, and then replace the rows of $\mathbf{L_D}$ corresponding to collocation at boundary points with unit vectors that have a one in the position corresponding to the diagonal of $\mathbf{L_D}$. Thus, the boundary condition $U(\bar{\mathbf{x}}, s) = \widehat{h}(\bar{\mathbf{x}}, s)$ in (7) will be explicitly enforced at this point as soon as we set the right-hand side to the corresponding value of $\widehat{h}(\bar{\mathbf{x}}, s)$ [23]. After incorporating the boundary conditions and solving Equation (13) for each node s, we obtain the approximate solution $\widehat{U}(\bar{\mathbf{x}}, s)$ in the LT domain. In Section 2.4, we describe the numerical inverse LT method used to evaluate the approximate time domain solution $U(\bar{\mathbf{x}}, \tau)$ for the original problem in Equations (1)–(3).

Stability and Error Analysis

Given the Chebyshev nodes in Equation (9) and the Lagrange interpolation polynomials defined in Equation (8), the interpolation operator is defined as follows [37]:

$$\mathcal{I}_n : C(\Omega) \to \mathcal{P}_n, \quad \mathcal{I}_n(U) = \sum_{j=0}^{n} U(x_j) l_j(x). \quad (14)$$

The steps proposed by Borm et al. [39] for constructing the error bound are utilized here. Let M_n be a constant with the stability estimate given as

$$\|\mathcal{I}_n(U)\|_\infty \le M_n \|U\|_\infty, \quad for\ all\ U \in C[-1, 1]. \quad (15)$$

$$\mathcal{I}_n(U) = U, \quad for\ \ all\ \ U \in \mathcal{P}_n. \quad (16)$$

For the Chebyshev interpolation, we have

$$M_n = \frac{ln(n+1)}{(\frac{\pi}{2})} + 1 \le (n+1). \quad (17)$$

Thus, the stability constant depends on n and develops extremely slowly. Additionally, the following approximation error bound [39] holds for a function $U \in C^{(n+1)}[-1, 1]$,

$$\|U - \mathcal{I}_n(U)\|_\infty \le \frac{2^{-n}}{(n+1)!} \|U^{(n+1)}\|_\infty. \quad (18)$$

Theorem 1 ([39]). *If (15) and (18) hold for $U \in C^{(n+1)}[-1, 1]$, and considering $k \in \{0, 1, 2, \ldots, n\}$, then*

$$\|U^{(k)} - \mathcal{I}_n(U)^{(k)}\|_\infty \le \frac{2(M_n^{(k)} + 1)}{(n-k+1)!} \left(\frac{1}{2}\right)^{(n-k+1)} \|U^{(n+1)}\|_\infty \quad (19)$$

depending on the stability constant

$$M_n^{(k)} = \frac{M_n}{k!}\left(\frac{n!}{(n-k)!}\right).$$

Now, use (18) and (19) for the TFADE (1) in one dimension. For the $1-D$ case, the linear differential operator $\mathbf{L}$ is of the form $\mathbf{L} = \lambda_1 \frac{\partial^2}{\partial x^2} + u_1 \frac{\partial}{\partial x} - \lambda_2 I$. Thus, we find the following error estimate

$$\begin{aligned}
\mathcal{E} &= \|(D_t^\alpha U - \mathbf{L}U) - (D_t^\alpha \mathcal{I}_n(U) - \mathbf{L}\mathcal{I}_n(U))\|_\infty \\
&= \|(D_t^\alpha (U - \mathcal{I}_n(U)) - \mathbf{L}(U - \mathcal{I}_n(U))\|_\infty \\
&\leq \|(D_t^\alpha (U - \mathcal{I}_n(U)))\|_\infty + \|\mathbf{L}(U - \mathcal{I}_n(U))\|_\infty \\
&\leq \|D_t^\alpha (U - \mathcal{I}_n(U))\|_\infty + |\lambda_1| \|U_{xx} - \mathcal{I}_n(U)_{xx}\|_\infty \\
&+ |u_1| \|U_x - \mathcal{I}_n(U)_x\|_\infty + |\lambda_2| \|U - \mathcal{I}_n(U)\|_\infty \\
\mathcal{E} &\leq \|D_t^\alpha (U - \mathcal{I}_n(U))\|_\infty + |\lambda_1| \frac{2(M_n^{(2)} + 1)}{(n-1)!} \left(\frac{1}{2}\right)^{n-1} \|U^{(n+1)}\|_\infty \\
&+ |u_1| \frac{2(M_n^{(1)} + 1)}{n!} \left(\frac{1}{2}\right)^{n} \|U^{(n+1)}\|_\infty + |\lambda_2| \frac{2^{-n}}{(n+1)!} \|U^{(n+1)}\|_\infty
\end{aligned}$$

The time derivative is accurately evaluated. Therefore, the error bound of $\|(D_t^\alpha (U - \mathcal{I}_n(U)))\|_\infty$ is the same order of $\|(U - \mathcal{I}_n(U))\|_\infty$. Finally, we have [37]:

$$\mathcal{E} \leq C \|U^{(n+1)}\|_\infty.$$

where C is constant, resulting from calculation of the coefficients of $\|U^{(n+1)}\|_\infty$. We can use the tensor product interpolation operators for two-dimensional problems.

2.4. *Numerical Inversion of Laplace Transform*

In this section, we implement the inverse Laplace transform method to convert the CSCM solution $\widehat{U}(\bar{\mathbf{x}}, s)$ from the Laplace domain to the time domain as follows:

$$U(\bar{\mathbf{x}}, \tau) = \frac{1}{2\pi i} \int_{\rho - i\infty}^{\rho + i\infty} e^{s\tau} \widehat{U}(\bar{\mathbf{x}}, s) ds, \ \rho > \rho_0. \tag{20}$$

Here the transform $\widehat{U}(\bar{\mathbf{x}}, s)$ needs to be inverted, ρ_0 is the converging abscissa, and $\rho > \rho_0$. This means that all of the singularities of $\widehat{U}(\bar{\mathbf{x}}, s)$ lie in the open half-plane $Re(s) < \rho$. We aim to approximate the integral defined in Equation (20). In most cases, it is quite difficult to evaluate the integral defined in Equation (20) analytically; thus, a numerical method must be used. There are several numerical algorithms in the literature that can be used to evaluate the integral defined in Equation (20). Among them, we can list the Fourier series method [40], the de Hoog method [41], Stehfest's method [35,42], Talbot's method [32,36], and the Weeks method [43,44], etc. Each approach has an identifiable use and is appropriate for a particular function. In this article, we use two popular inversion algorithms: the improved Talbot's and Stehfest's methods will be presented in the following sections.

2.4.1. TheImproved Talbot's Method

Here, we use Talbot's method to approximate the $U(\bar{\mathbf{x}}, \tau)$

$$U(\bar{\mathbf{x}}, \tau) = \frac{1}{2\pi i} \int_{\rho - i\infty}^{\rho + i\infty} e^{s\tau} \widehat{U}(\bar{\mathbf{x}}, s) ds = \frac{1}{2\pi i} \int_{C} e^{s\tau} \widehat{U}(\bar{\mathbf{x}}, s) ds, \ Re(s) > 0. \tag{21}$$

where C is a suitably chosen line. It is very difficult to solve the above integral due to the highly oscillatory exponential function $e^{s\tau}$ and the slow-decaying transform $\widehat{U}(\bar{\mathbf{x}}, s)$. Talbot [36] suggested that by utilizing contour integration, this problem can be resolved. He further said that the integration might be carried out so that the real parts of the contours begin and stop on the left side of the complex plane, which contains all of the singularities in the transformed function $\widehat{U}(\bar{\mathbf{x}}, s)$. Due to the exponential element, the integrand decays quickly on such contours, making integral Equation (21) suitable for approximation with the midpoint rule. Consider a contour of the form [32]:

$$\Gamma : s = s(\eta), \quad -\pi \leq \eta \leq \pi, \tag{22}$$

where $Res(\pm\pi) = -\infty$, and $s(\eta)$ is given as

$$s(\eta) = \frac{M_T}{\tau}\zeta(\eta),\ \zeta(\eta) = -\delta + \varrho\eta cot(\mu\eta) + \gamma i\eta, \tag{23}$$

where $\mu, \varrho, and\ \gamma$ will be chosen precisely. Using Equation (23) in Equation (21) , we obtain

$$U(\bar{\mathbf{x}}, \tau) = \frac{1}{2\pi i}\int_{-\pi}^{\pi} e^{s(\eta)\tau}\widehat{U}(\bar{\mathbf{x}}, s(\eta))s'(\eta)d\eta. \tag{24}$$

The integral defined in (24) is approximated via the midpoint rule with spacing $h = \frac{2\pi}{M_T}$ as

$$U_{App}(\bar{\mathbf{x}}, \tau) \approx \frac{1}{M_T i}\sum_{j=1}^{M_T} e^{s(\eta_j)\tau}\widehat{U}(\bar{\mathbf{x}}, s(\eta_j))s'(\eta_j), \quad \eta_j = -\pi + (j - \frac{1}{2})h. \tag{25}$$

Convergence

The approximate value of the defined integral in Equation (24) depends on the contour of integration Γ; different rates of convergence are attained for the suggested numerical scheme. Additionally, the quadrature step h determines that the suggested scheme will converge. In order to have optimal results, we need to search for the optimal contour of integration, which can be found by using the optimal values for the parameters involved in (23). In [32], the authors have proposed optimal values for the parameters as

$$\delta = 0.6122, , \gamma = 0.2645, \mu = 0.6407, \text{ and } \varrho = 0.5017,$$

For the improved Talbot's method, the error estimate is given as

$$Error_{est} = |\ U_{App}(\bar{\mathbf{x}}, \tau) - U(\bar{\mathbf{x}}, \tau)| = O(e^{-1.35800M_T}).$$

2.4.2. TheStehfest's Method

The Gaver–Stehfest method is one of the most important techniques for the numerical inversion of the Laplace transform. It was designed in the second half of the 1960s. It has gained prominence in a number of disciplines, such as chemistry, finance, economics, and computational physics, due to its effectiveness and ease. The series of Gaver approximants, as determined by Gaver [42], is the foundation of the Gaver–Stehfest method. Acceleration was required because the Gaver approximants' convergence was basically logarithmic. Stehfest [35] used the Salzer acceleration method to offer a linear acceleration method. The Gaver–Stehfest approach uses a series of functions to approximate $U(\bar{\mathbf{x}}, \tau)$ as

$$U_{App}(\bar{\mathbf{x}}, \tau) = \frac{ln2}{\tau}\sum_{j=1}^{M_S}\alpha_j\widehat{U}\left(\bar{\mathbf{x}}, \frac{ln2}{\tau}j\right), \tag{26}$$

where the coefficients α_j are defined by

$$\alpha_j = (-1)^{\frac{M_S}{2}+j}\sum_{\ell=\lfloor\frac{j+1}{2}\rfloor}^{min(j,\frac{M_S}{2})}\frac{\ell^{\frac{M_S}{2}}(2\ell)!}{(\frac{M_S}{2}-\ell)!\ell!(\ell-1)!(j-\ell)!(2\ell-j)!}. \tag{27}$$

Solve Equation (6) for the corresponding Laplace parameters $s = \frac{ln2}{\tau}j,\ j = 1, 2, 3, \ldots,$ M_S. The solution to the given problem in Equation (1) can be obtained by (26). There are a few appealing features of the Gaver–Stehfest algorithm: (i) $U_{App}(\bar{\mathbf{x}}, \tau)$ are linear in terms of the values of $U(\bar{\mathbf{x}}, s)$; (ii) the values of $U(\bar{\mathbf{x}}, s)$ are needed only for the real value of s; (iii) the process of computing the coefficients is quite simple; and (iv) for constant functions, this approach yields highly accurate approximations. In the literature, this technique has

been used by many authors in [45,46], where it has been demonstrated that this method converges very fast to $U_{App}(\bar{\mathbf{x}}, \tau)$ (provided $U_{App}(\bar{\mathbf{x}}, \tau)$ is non-oscillatory).

Convergence

In [45], the author has derived two conditions for $U(\bar{\mathbf{x}}, \tau)$, which guarantee the convergence of $U_{App}(\bar{\mathbf{x}}, \tau)$. The conditions are given in the following theorem.

Theorem 2. *Let $U : (0, \infty) \to \mathbb{R}$ be a locally integrable function and the approximate solution $U_{App}(\bar{\mathbf{x}}, \tau)$ be defined by (26), and let its LT $\widehat{U}(\bar{\mathbf{x}}, s)$ be defined for $s > 0$:*

1. *$U_{App}(\bar{\mathbf{x}}, \tau)$ converges for the values of $U_{App}(\bar{\mathbf{x}}, \tau)$ in the neighborhood of τ.*
2. *Let for some real number m and some $0 < \varepsilon < 1/4$*

$$\int_0^{\varepsilon} |U(-\tau log_2(1/2 + \xi)) + U(-\tau log_2(1/2 - \xi)) - 2m|\xi^{-1} d\xi < \infty.$$

   Then, $U \to m$ as $M_S \to +\infty$.
3. *Allow that $U_{App}(\bar{\mathbf{x}}, \tau)$ has a bounded variation in the neighborhood of τ. Then,*

$$U_{App}(\tau) \to \frac{U(\tau + 0) + U(\tau - 0)}{2} \text{ as } M_S \to +\infty.$$

Corollary 1. *Under the above assumptions, if*

$$U(\tau + \xi) - U(\tau) = O(|\xi|^{v}),$$

$\forall\ \xi$ and some v in the neighborhood of τ, then $U_{App}(\bar{\mathbf{x}}, \tau) \to U(\bar{\mathbf{x}}, \tau)$, as $M_S \to +\infty$.

Additionally, the authors in [47] performed various experiments for the parameter effect on the accuracy of the numerical scheme, and they concluded in their findings that "If p significant digits are required, then let M_S be the positive integer $\lceil 2.2p \rceil$. Set the system precision to $q = \lceil 1.1 M_S \rceil$ and, for a given M_S, calculate $\alpha_i, 1 \leq i \leq M_S$, using (27). Then, for the given transform $\widehat{U}(\bar{\mathbf{x}}, s)$ and the argument τ, calculate the $U_{App}(\bar{\mathbf{x}}, \tau)$ in (26)". According to these conclusions, if the error of input data is $10^{-(q+1)} \leq \frac{\|\widehat{U} - U\|}{\|U\|} \leq 10^{-q}$, with an even positive integer M_S via $q = \lceil 1.1 M_S \rceil$, then the final error is $10^{-(p+1)} \leq \frac{\|\widehat{U} - U\|}{\|U\|} \leq 10^{-p}$, where $M_S = \lceil 2.2p \rceil$ [48]. Next, we present Algorithm 1 for the proposed numerical scheme.

Algorithm 1 : Algorithm for the proposed numerical scheme

1: **Input:** The computational domain, the fractional order derivative, the final time, the optimal parameters for the inverse laplace transform methods, the inhomogeneous function, and other conditions.
2: **Step i:** Apply the Laplace transform to problems (1) and (2), and obtain the time-independent problems (6) and (7).
3: **Step ii:** Obtain $\mathbf{L_D}$ by discretizing the linear differential operator $\mathbf{L}$ using the CSCM via (12).
4: **Step iii:** Solve Equation (13) with the boundary conditions given in Equation (7) for each point in the LT domain.
5: **Step iv:** Compute the approximate solution to the problem in Equations (1)–(3) using Equation (21) or Equation (26).
6: **Step v:** The approximate solution is $U_{App}(\bar{\mathbf{x}}, \tau)$.

3. Numerical Results and Discussion

The numerical results and discussions are addressed in this section. Three examples are used to evaluate and validate the effectiveness and accuracy of the Laplace-transformed CSCM. We performed our experiments in MATLAB R2019a on a Windows 10 (64-bit) PC equipped with an Intel(R) Core TM i5-4310M CPU @ 2.70 GHz with 4 GB of RAM. Three error norms—the relative L_2 error, the maximum absolute error, and the RMS error—are used to evaluate the accuracy of the proposed method and are defined below:

$$er_2 = \sqrt{\frac{\sum_{j=1}^{n}(U(\bar{\mathbf{x}}_j,\tau) - U_{App}(\bar{\mathbf{x}}_j,\tau))^2}{\sum_{j=1}^{n}(U(\bar{\mathbf{x}}_j,\tau))^2}},$$

$$er_\infty = \max_{1\le j\le n} |U(\bar{\mathbf{x}}_j,\tau) - U_{App}(\bar{\mathbf{x}}_j,\tau)|,$$

$$er_{rms} = \sqrt{\frac{\sum_{j=1}^{n}(U(\bar{\mathbf{x}}_j,\tau) - U_{App}(\bar{\mathbf{x}}_j,\tau))^2}{n}},$$

where $U(\bar{\mathbf{x}}_j,\tau)$ and $U_{App}(\bar{\mathbf{x}}_j,\tau)$ represent the analytic and numerical solutions, respectively.

Example 1. *Consider the 2D time-fractional advection–diffusion Equation (1) with an exact solution given as*

$$U(x,y,\tau) = x^2 + y^2 + \tau^2,\ \Omega = [-1,1]^2,\ \tau \in [0,1],$$

where $\lambda_1 = 1$, $\overrightarrow{\mathbf{u}} = (1,1)^T$, *and* $\lambda_2 = 0$. *The forcing term* $\mathcal{Q}(x,y,\tau)$, *and the initial-boundary data are extracted from the exact solution. The numerical results obtained via Talbot's and Stehfest's methods are presented in Tables 1 and 2 by using different values of* α, M_S, M_T, *and* n *at* $\tau = 1$. *The approximate solution obtained by the proposed method is shown in Figure 1a. The plots for the comparison of errors* er_2, er_∞, *and* er_{rms} *using Stehfest's method for several values of* α, M_S, *and* τ *at* $n = 25$ *are shown in Figure 1b–d, respectively. Figure 2a–c show the plots of errors* er_2, er_∞, *and* er_{rms} *obtained using Talbot's method with different values of* α, M_T, *and* τ *at* $n = 25$. *Figure 2d presents the contour plot of the absolute error using Talbot's method for different values of* α *with* $M_T = 26, \tau = 1$, *and* $n = 25$. *The accuracy of Stehfest's method enhances for* $M_S = 8, 10, 12, 14$ *and then gradually diminishes for values exceeding* $M_S = 14$. *From all of the obtained results presented in the figures and tables, we conclude that the proposed method is stable, accurate, and efficient. Clearly, the numerical results show that the accuracy of Talbot's method is better than that of Stehfest's method.*

Example 2. *Consider the 2D time-fractional advection–diffusion Equation (1) with an exact solution given as*

$$U(x,y,\tau) = sin(x+y) + \tau^2,\ \Omega = [-1,1]^2,\ \tau \in [0,1],$$

where $\lambda_1 = 1$, $\overrightarrow{\mathbf{u}} = (1,1)^T$, *and* $\lambda_2 = 0$. *The forcing term* $\mathcal{Q}(x,y,\tau)$, *and the initial-boundary data are extracted from the exact solution. The numerical results obtained via Talbot's and Stehfest's methods are presented in Tables 3 and 4 by using different values for* α, M_S, M_T, *and* n *at* $\tau = 1$. *The approximate solution obtained by the proposed method is shown in Figure 3a. The plots of the comparison of errors* er_2, er_∞, *and* er_{rms} *using Stehfest's method for several values of* α, M_S, *and* τ *at* $n = 25$ *are shown in Figure 3b–d, respectively. Figure 4a–c show the plots of errors* er_2, er_∞, *and* er_{rms} *by using Talbot's method with different values of* α, M_T, *and* τ *at* $n = 25$. *Figure 3e presents the contour plot of the absolute error obtained by using Stehfest's method for different values of* α *with* $M_S = 12, \tau = 1$, *and* $n = 25$. *The accuracy of Stehfest's method enhances for* $M_S = 8, 10, 12, 14$ *and then gradually diminishes for values exceeding* $M_S = 14$. *From all of the obtained results presented in the figures and tables, we conclude that the proposed method is stable, accurate, and efficient. Clearly, the results show that the accuracy of Talbot's method is better than that of Stehfest's method.*

Table 1. The er_2, er_∞, *and* er_{rms} obtained using Stehfest's method for different values of α, n, *and* M_S at $\tau = 1$ for Example 1.

$\alpha = 0.30$	n	M_S	er_2	er_∞	er_{rms}	CPU(s)
	21	12	9.7994×10^{-4}	6.0556×10^{-5}	4.6664×10^{-5}	0.233261
	23		1.1084×10^{-3}	8.2038×10^{-5}	4.8190×10^{-5}	0.200611
	25		1.1581×10^{-3}	7.1935×10^{-5}	4.6325×10^{-5}	0.266045
	25	08	1.0812×10^{-1}	4.1584×10^{-3}	4.3246×10^{-3}	0.196162
		10	1.4553×10^{-3}	5.8142×10^{-5}	5.8213×10^{-5}	0.249590
		12	1.1581×10^{-3}	7.1935×10^{-5}	4.6325×10^{-5}	0.255773
$\alpha = 0.50$						
	21	12	9.8850×10^{-4}	5.6517×10^{-5}	4.7071×10^{-5}	0.167258
	23		1.0923×10^{-3}	7.1906×10^{-5}	4.7492×10^{-5}	0.204245
	25		1.2186×10^{-3}	1.3279×10^{-4}	4.8744×10^{-5}	0.302795
	25	08	1.0812×10^{-1}	4.1584×10^{-4}	4.3246×10^{-3}	0.196037
		10	1.4539×10^{-3}	5.7437×10^{-5}	5.8156×10^{-5}	0.241286
		12	1.2186×10^{-3}	1.3279×10^{-4}	4.8744×10^{-5}	0.377014
$\alpha = 0.80$						
	21	12	1.0540×10^{-3}	9.2987×10^{-5}	5.0191×10^{-5}	0.142862
	23		9.8247×10^{-4}	6.2113×10^{-5}	4.2716×10^{-5}	0.235252
	25		1.2741×10^{-3}	1.5227×10^{-4}	5.0962×10^{-5}	0.287223
	25	08	1.0811×10^{-1}	4.1585×10^{-3}	4.3246×10^{-5}	0.214762
		10	1.4514×10^{-3}	5.7719×10^{-5}	5.8056×10^{-5}	0.234821
		12	1.2741×10^{-3}	1.5227×10^{-4}	5.0962×10^{-5}	0.287223

Table 2. The er_2, er_∞, *and* er_{rms} obtained via Talbot's method for different values of α, n, and M_T at $\tau = 1$ for Example 1.

$\alpha = 0.30$	n	M_T	er_2	er_∞	er_{rms}	CPU(s)
	21	26	3.6788×10^{-10}	9.4858×10^{-11}	1.7518×10^{-11}	0.477112
	23		5.1649×10^{-10}	1.5900×10^{-10}	2.2456×10^{-11}	0.710826
	25		8.1265×10^{-10}	3.4762×10^{-10}	3.2506×10^{-11}	1.118694
	25	22	3.6227×10^{-8}	1.5662×10^{-9}	1.4491×10^{-9}	0.955487
		24	3.0969×10^{-9}	3.1978×10^{-10}	1.2388×10^{-10}	0.997716
		26	8.1265×10^{-10}	3.4762×10^{-10}	3.2506×10^{-11}	1.082287
$\alpha = 0.50$						
	21	26	4.5560×10^{-10}	1.7228×10^{-10}	2.1695×10^{-11}	0.468985
	23		5.0871×10^{-10}	1.8159×10^{-10}	2.2118×10^{-11}	0.741330
	25		8.4173×10^{-10}	3.3289×10^{-10}	3.3669×10^{-11}	1.042339
	25	22	3.6042×10^{-8}	1.4790×10^{-9}	1.4417×10^{-9}	0.952621
		24	2.8473×10^{-9}	3.0892×10^{-10}	1.1389×10^{-10}	0.987540
		26	8.4173×10^{-10}	3.3289×10^{-10}	3.3669×10^{-11}	1.042339
$\alpha = 0.80$						
	21	26	3.9392×10^{-10}	8.8192×10^{-11}	1.8758×10^{-11}	0.479675
	23		4.8220×10^{-10}	1.9319×10^{-10}	2.0965×10^{-11}	0.761089
	25		1.1098×10^{-9}	7.7091×10^{-10}	4.4393×10^{-11}	1.094016
	25	22	3.6116×10^{-8}	1.5182×10^{-9}	1.4446×10^{-9}	0.917780
		24	2.9420×10^{-9}	2.3727×10^{-10}	1.1768×10^{-10}	1.010576
		26	1.1098×10^{-9}	7.7091×10^{-10}	4.4393×10^{-11}	1.094016

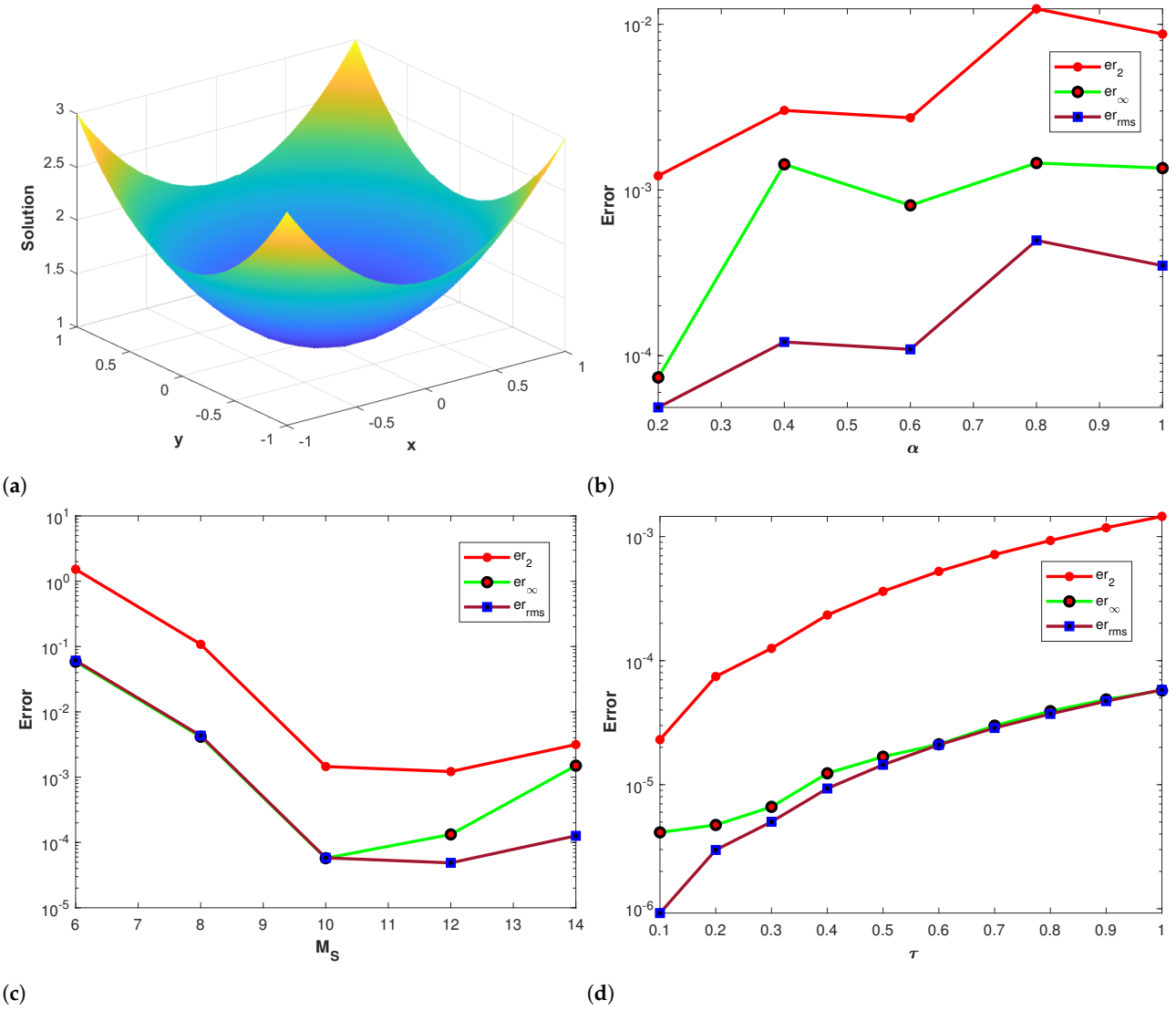

Figure 1. (**a**) Numerical solution of Example 1. (**b**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Stehfest's method for different α, with $M_S = 10$ and $n = 25$ at $\tau = 1$ for Example 1. (**c**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Stehfest's method for different M_S with $\alpha = 0.3$ and $n = 25$ at $\tau = 1$ for Example 1. (**d**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Stehfest's method for different τ with $M_S = 10$, $\alpha = 0.3$, and $n = 25$ for Example 1.

Example 3. *Consider the 2D time-fractional advection–diffusion Equation* (1) *with an exact solution given as*

$$U(x,y,\tau) = \tau^2 e^{(x+y)},\ \Omega = [-1,1]^2,\ \tau \in [0,1],$$

where $\lambda_1 = 1$, $\overrightarrow{\mathbf{u}} = (1,1)^T$, *and* $\lambda_2 = 0$. *The forcing term* $\mathcal{Q}(x,y,\tau)$, *and the initial-boundary data are extracted from the exact solution. The numerical results obtained via Talbot's and Stehfest's methods are presented in Tables 5 and 6 by using different values of* $\alpha, M_S, and M_T$, *with n at* $\tau = 1$. *The approximate solution obtained by the proposed method is shown in Figure 5a. The plots of the comparison of the errors* er_2, er_∞, *and* er_{rms} *using Stehfest's method for several values of* α, M_S, *and* τ *at* $n = 25$ *are shown in Figure 5b–d, respectively. Figure 6a–c show the plots of errors* er_2, er_∞, *and* er_{rms} *via Talbot's method with different values of* α, M_T, *and* τ *at* $n = 25$. *Figure 6d presents the contour plot of the absolute error by using Talbot's method for different values of* α *with* $M_T = 26, \tau = 1$, *and* $n = 25$. *The accuracy of Stehfest's method enhances for* $M_S = 8, 10, 12, 14$ *and then gradually diminishes for values exceeding than* $M_S = 14$. *From all the obtained results presented in the figures and tables, we conclude that the proposed method is stable,*

accurate, and efficient. Clearly, the results show that the accuracy of Talbot's method is better than that of Stehfest's method.

Table 3. The $er_2, er_\infty, ander_{rms}$ obtained via Stehfest's method for different values of α, n, and M_S at $\tau = 1$ of Example 2.

$\alpha = 0.30$	n	M_S	er_2	er_∞	er_{rms}	CPU(s)
	21	12	9.9430×10^{-4}	5.2055×10^{-5}	4.7348×10^{-5}	0.192349
	23		1.0940×10^{-3}	6.1845×10^{-5}	4.7567×10^{-5}	0.333385
	25		1.1758×10^{-3}	6.1712×10^{-5}	4.7031×10^{-5}	0.303723
	25	08	1.0812×10^{-1}	4.1584×10^{-3}	4.3246×10^{-3}	0.271822
		10	1.4554×10^{-3}	5.8032×10^{-5}	5.8217×10^{-5}	0.257647
		12	1.1758×10^{-3}	6.1712×10^{-5}	4.7031×10^{-5}	0.303723
$\alpha = 0.50$						
	21	12	9.9254×10^{-4}	5.2330×10^{-5}	4.7264×10^{-5}	0.170534
	23		1.0888×10^{-3}	5.9124×10^{-5}	4.7338×10^{-5}	0.257113
	25		1.1700×10^{-3}	5.4659×10^{-5}	4.6800×10^{-5}	0.360627
	25	08	1.0812×10^{-1}	4.1583×10^{-3}	4.3246×10^{-3}	0.197690
		10	1.4551×10^{-3}	5.6736×10^{-5}	5.8205×10^{-5}	0.383466
		12	1.1700×10^{-3}	5.4659×10^{-5}	4.6800×10^{-5}	0.360627
$\alpha = 0.80$						
	21	12	9.9170×10^{-4}	5.0410×10^{-5}	4.7224×10^{-5}	0.178846
	23		1.0836×10^{-3}	5.0672×10^{-5}	4.7115×10^{-5}	0.276298
	25		1.2056×10^{-3}	6.7927×10^{-5}	4.8224×10^{-5}	0.292158
	25	08	1.0812×10^{-1}	4.1583×10^{-3}	4.3246×10^{-5}	0.274343
		10	1.4550×10^{-3}	5.6956×10^{-5}	5.8199×10^{-5}	0.269435
		12	1.2056×10^{-3}	6.7927×10^{-5}	4.8224×10^{-5}	0.292158

Table 4. The $er_2, er_\infty, ander_{rms}$ obtained using Talbot's method for different values of α, n, and M_T at $\tau = 1$ of Example 2.

$\alpha = 0.30$	n	M_T	er_2	er_∞	er_{rms}	CPU(s)
	21	26	2.3790×10^{-10}	6.3914×10^{-11}	1.1328×10^{-11}	0.508399
	23		2.6664×10^{-10}	5.2105×10^{-11}	1.1593×10^{-11}	0.829189
	25		5.6968×10^{-10}	3.8928×10^{-10}	2.2787×10^{-11}	1.151650
	25	22	3.6179×10^{-8}	1.4491×10^{-9}	1.4472×10^{-9}	0.977947
		24	2.9202×10^{-9}	2.3436×10^{-10}	1.1681×10^{-10}	1.044065
		26	5.6968×10^{-10}	3.8928×10^{-10}	2.2787×10^{-11}	1.151650
$\alpha = 0.50$						
	21	26	2.7669×10^{-10}	7.5912×10^{-11}	1.3176×10^{-11}	0.545475
	23		2.7187×10^{-10}	6.4024×10^{-11}	1.1820×10^{-11}	0.806244
	25		4.6716×10^{-10}	2.0896×10^{-10}	1.8687×10^{-11}	1.225075
	25	22	3.6229×10^{-8}	1.4880×10^{-9}	1.4492×10^{-9}	0.939474
		24	2.8311×10^{-9}	1.5034×10^{-10}	1.1324×10^{-10}	1.027609
		26	4.6716×10^{-10}	2.0896×10^{-10}	1.8687×10^{-11}	1.225075
$\alpha = 0.80$						
	21	26	2.4234×10^{-10}	4.0678×10^{-11}	1.1540×10^{-11}	0.512461
	23		2.4004×10^{-10}	7.5801×10^{-11}	1.0437×10^{-11}	0.820029
	25		4.3673×10^{-10}	2.3799×10^{-10}	1.7469×10^{-11}	1.162087
	25	22	3.6188×10^{-8}	1.4465×10^{-9}	1.4475×10^{-9}	0.988509
		24	2.8035×10^{-9}	1.4878×10^{-10}	1.1214×10^{-10}	1.047553
		26	4.3673×10^{-9}	2.3799×10^{-10}	1.7469×10^{-11}	1.162087

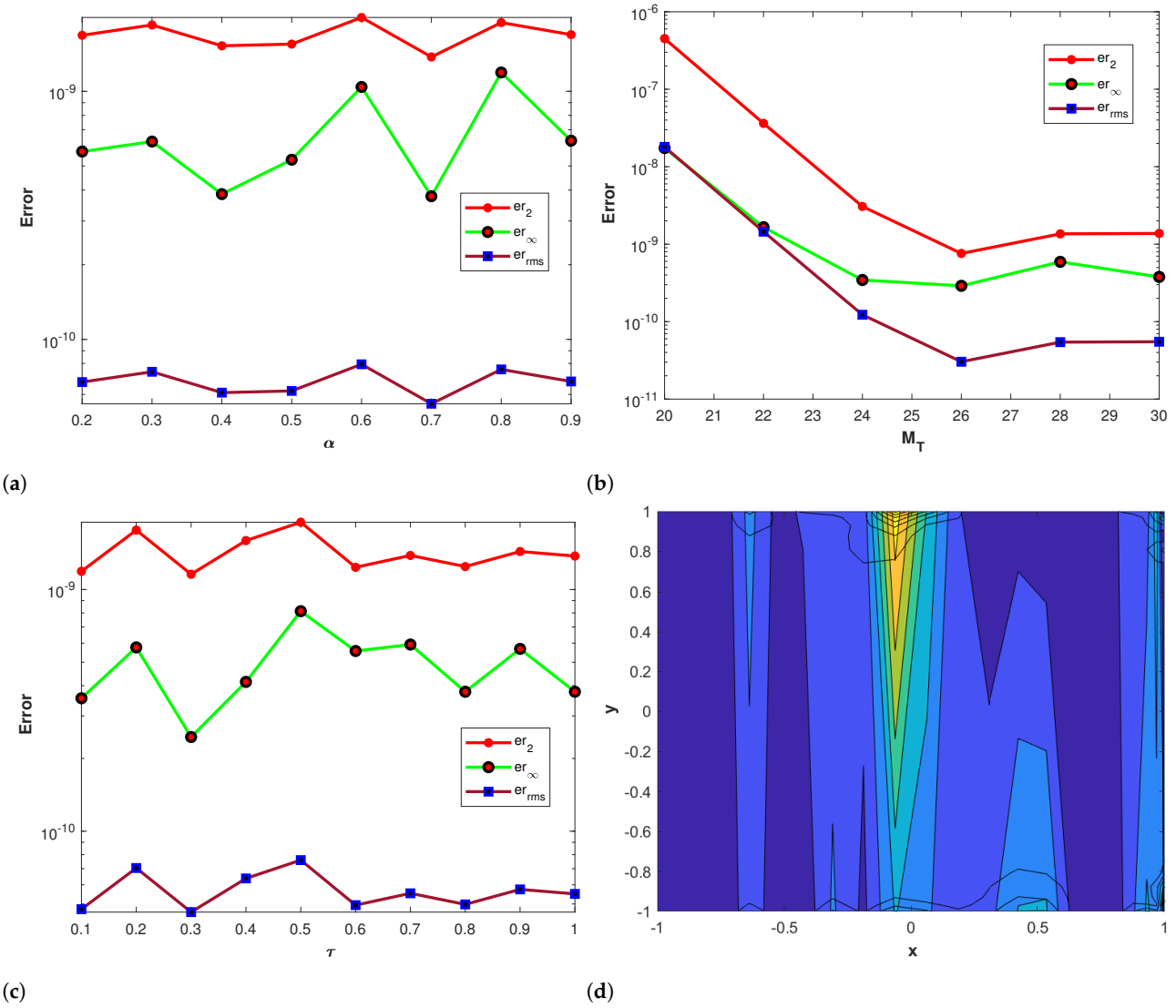

Figure 2. (**a**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Talbot's method for different α with $M_T = 22$ and $n = 25$ at $\tau = 1$ for Example 1. (**b**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Talbot's method for different M_T with $\alpha = 0.3$ and $n = 25$ at $\tau = 1$ for Example 1. (**c**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Talbot's method for different τ with $M_T = 22$, $alpha = 0.3$, and $n = 25$ for Example 1. (**d**) Contour plot of the absolute error obtained with $M_T = 26$, $\tau = 1$ $alpha = 0.3$, and $n = 25$ using Talbot's method for Example 1.

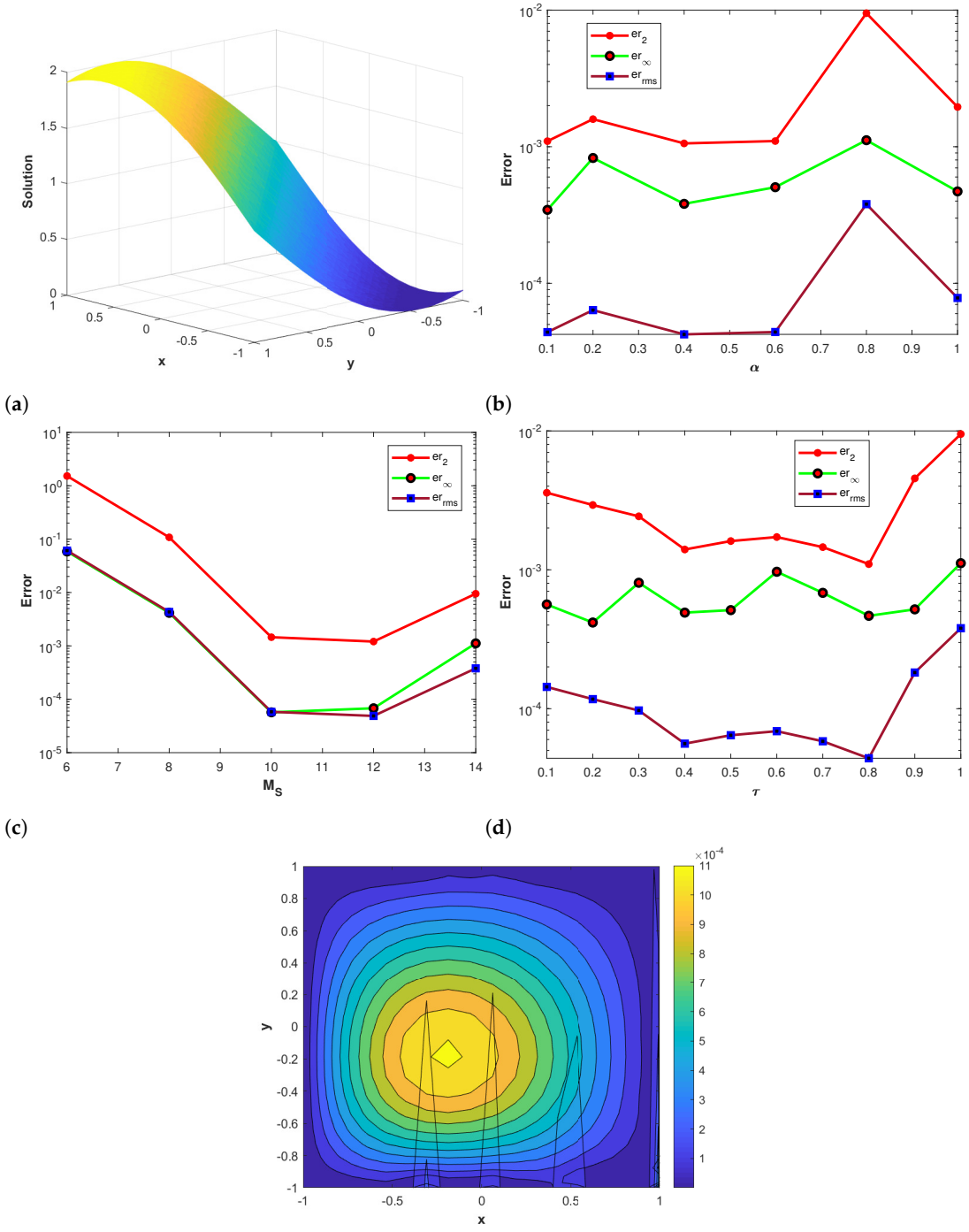

Figure 3. (**a**) Numerical solution of Example 2. (**b**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Stehfest's method for different α, with $M_S = 10$ and $n = 25$ at $\tau = 1$ for Example 2. (**c**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Stehfest's method for different M_S with $\alpha = 0.3$ and $n = 25$ at $\tau = 1$ for Example 2. (**d**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Stehfest's method for different τ with $M_S = 10$, $\alpha = 0.3$, and $n = 25$ for Example 2. (**e**) Contour plot of the absolute error obtained using Stehfest's method with $\alpha = 0.5$, $M_S = 12$, $\tau = 1$, and $n = 25$ for Example 2.

Table 5. The $er_2, er_\infty, ander_{rms}$ obtained via Stehfest's method for different values of α, n, and M_S at $\tau = 1$ of Example 3.

$\alpha = 0.30$	n	M_S	er_2	er_∞	er_{rms}	CPU(s)
	21	14	1.9041×10^{-4}	3.5446×10^{-5}	9.0670×10^{-6}	0.142019
	23		1.7304×10^{-4}	2.7673×10^{-5}	7.5234×10^{-6}	0.214226
	25		2.2300×10^{-4}	4.3858×10^{-5}	8.9200×10^{-6}	0.278411
	25	10	3.3997×10^{-3}	4.1349×10^{-4}	1.3599×10^{-4}	0.234626
		12	2.7481×10^{-3}	3.3443×10^{-4}	1.0992×10^{-4}	0.275799
		14	2.2300×10^{-4}	4.3858×10^{-5}	8.9200×10^{-6}	0.334381
$\alpha = 0.50$						
	21	14	1.7886×10^{-4}	3.4616×10^{-5}	8.5172×10^{-6}	0.173237
	23		1.9875×10^{-4}	4.7601×10^{-5}	8.6413×10^{-6}	0.238699
	25		1.8379×10^{-4}	2.8417×10^{-5}	7.3515×10^{-6}	0.321624
	25	10	3.3995×10^{-3}	4.1349×10^{-4}	1.3598×10^{-4}	0.246361
		12	2.7522×10^{-3}	3.3443×10^{-4}	1.1009×10^{-4}	0.256780
		14	1.8379×10^{-4}	2.8417×10^{-5}	7.3515×10^{-6}	0.320022
$\alpha = 0.80$						
	21	14	2.1292×10^{-4}	2.4907×10^{-5}	1.0139×10^{-5}	0.160703
	23		2.8307×10^{-4}	4.5577×10^{-5}	1.2307×10^{-5}	0.228405
	25		4.0222×10^{-4}	4.9547×10^{-5}	1.6089×10^{-5}	0.323644
	25	10	3.3995×10^{-3}	4.1349×10^{-4}	1.3598×10^{-4}	0.253768
		12	2.7525×10^{-3}	3.3443×10^{-4}	1.1010×10^{-5}	0.318281
		14	4.0222×10^{-4}	4.9547×10^{-5}	1.6089×10^{-5}	0.323644

Table 6. The $er_2, er_\infty, ander_{rms}$ obtained using Talbot's method for different values of α, n, and M_T at $\tau = 1$ for Example 3.

$\alpha = 0.30$	n	M_S	er_2	er_∞	er_{rms}	CPU(s)
	21	30	2.4416×10^{-11}	8.9271×10^{-12}	1.1627×10^{-12}	0.623102
	23		4.5926×10^{-11}	2.4259×10^{-11}	1.9968×10^{-12}	0.893166
	25		1.0575×10^{-10}	3.6040×10^{-11}	4.2300×10^{-12}	1.424708
	25	26	5.3511×10^{-10}	8.2624×10^{-11}	2.1404×10^{-11}	1.154428
		28	7.2756×10^{-11}	3.0986×10^{-11}	2.9102×10^{-12}	1.290241
		30	1.0575×10^{-10}	3.6040×10^{-11}	4.2300r $\times 10^{-12}$	1.304660
$\alpha = 0.50$						
	21	30	2.4307×10^{-11}	8.8405×10^{-12}	1.1575×10^{-12}	0.590850
	23		4.5760×10^{-11}	1.3785×10^{-11}	1.9896×10^{-12}	0.895766
	25		6.7767×10^{-11}	3.3908×10^{-11}	2.7107×10^{-12}	1.342481
	25	26	5.0284×10^{-10}	7.5070×10^{-11}	2.0114×10^{-11}	1.118816
		28	8.2316×10^{-11}	3.8009×10^{-11}	3.2926×10^{-12}	1.221575
		30	6.7767×10^{-11}	3.3908×10^{-11}	2.7107×10^{-12}	1.282603
$\alpha = 0.80$						
	21	30	2.6976×10^{-11}	8.4350×10^{-12}	1.2846×10^{-12}	0.569593
	23		4.9832×10^{-11}	1.3129×10^{-11}	2.1666×10^{-12}	0.901900
	25		8.9002×10^{-11}	5.0524×10^{-11}	3.5601×10^{-12}	1.338635
	25	26	5.0864×10^{-10}	7.6800×10^{-11}	2.0346×10^{-11}	1.152948
		28	6.4788×10^{-11}	2.4306×10^{-11}	2.5915×10^{-12}	1.197204
		30	8.9002×10^{-11}	5.0524×10^{-11}	3.5601×10^{-12}	1.295479

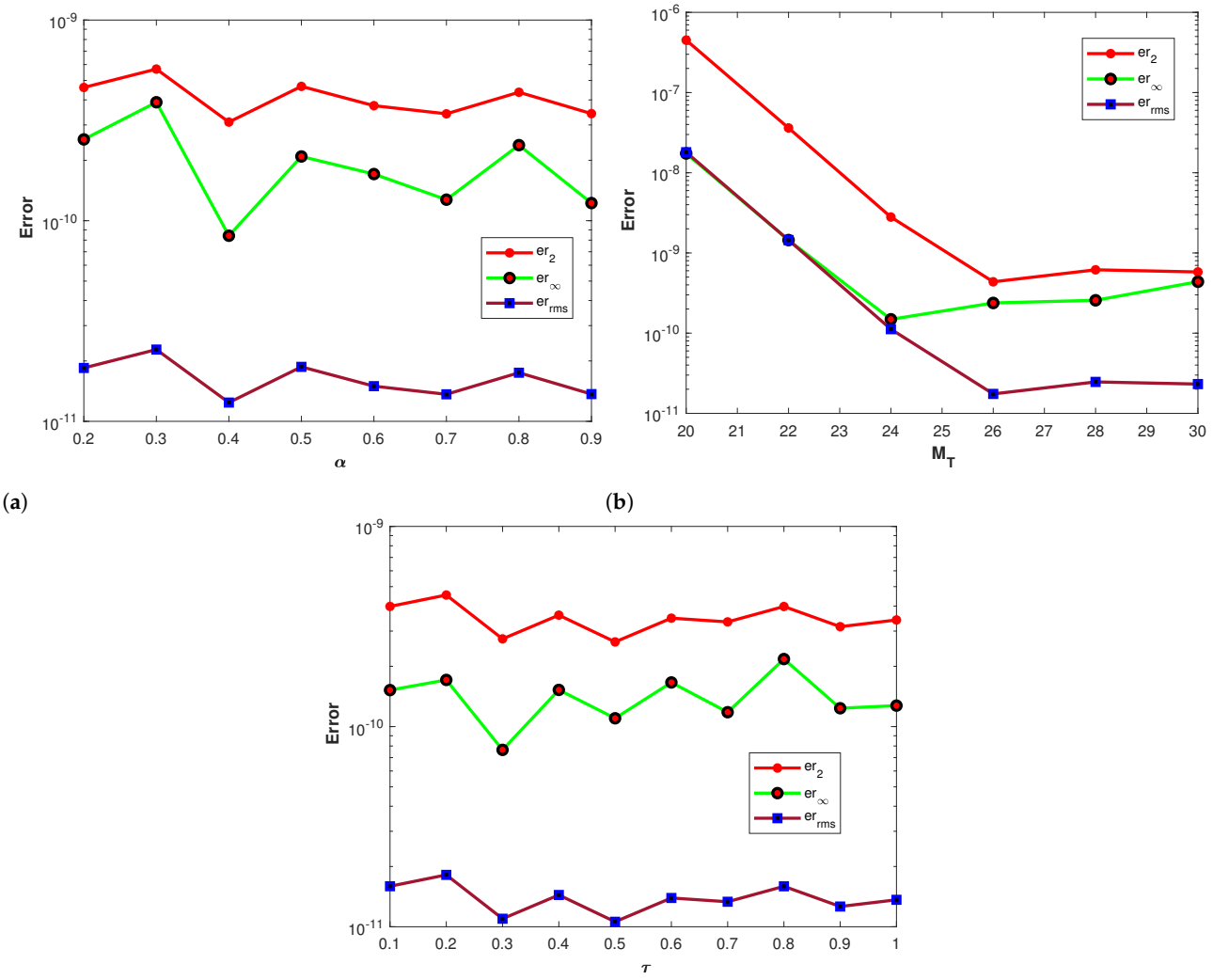

Figure 4. (**a**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Talbot's method for different α, with $M_T = 22$ and $n = 25$ at $\tau = 1$ for Example 2. (**b**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Talbot's method for different M_T with $\alpha = 0.3$ and $n = 25$ at $\tau = 1$ for Example 2. (**c**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Talbot's method for different τ with $M_T = 22$, $alpha = 0.3$, and $n = 25$ for Example 2.

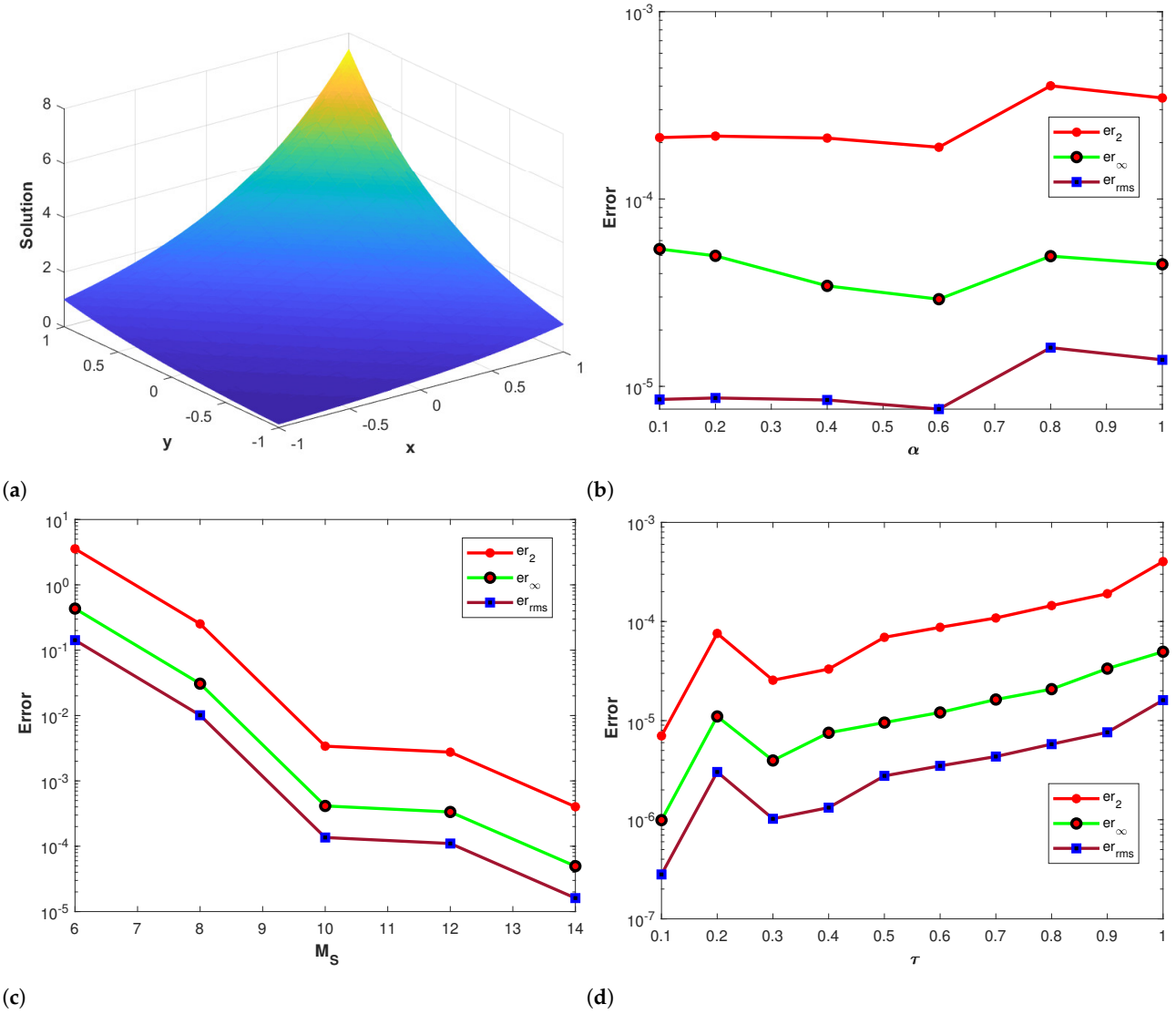

Figure 5. (**a**) Numerical solution of Example 3. (**b**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Stehfest's method for different α, with $M_S = 10$ and $n = 25$ at $\tau = 1$ for Example 3. (**c**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Stehfest's method for different M_S with $\alpha = 0.3$ and $n = 25$ at $\tau = 1$ for Example 3. (**d**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Stehfest's method for different τ with $M_S = 10$, $\alpha = 0.3$, and $n = 25$ for Example 3.

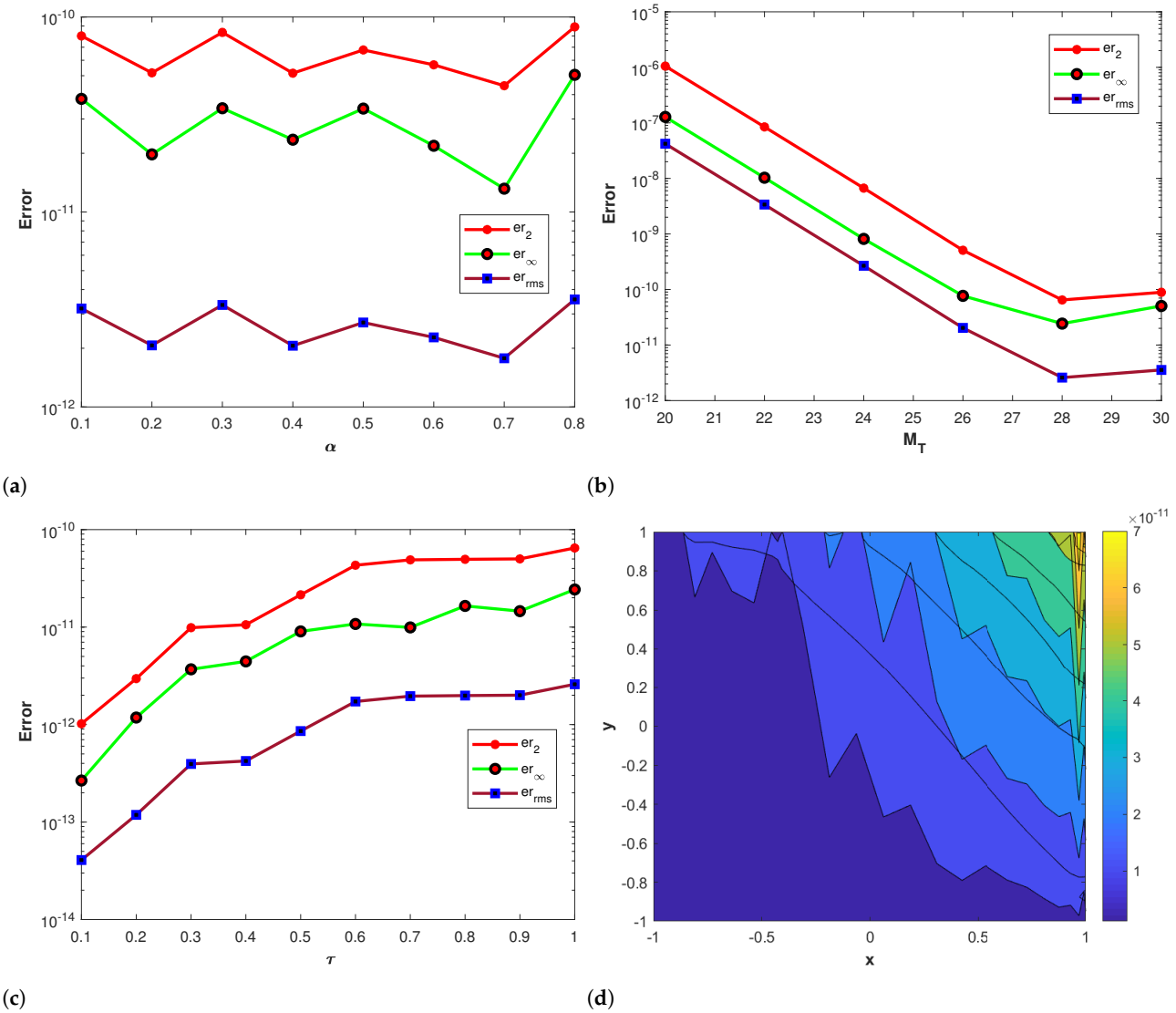

Figure 6. (**a**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Talbot's method for different α, with $M_T = 26$ and $n = 25$ at $\tau = 1$ for Example 3. (**b**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Talbot's method for different M_T with $\alpha = 0.3$ and $n = 25$ at $\tau = 1$ for Example 3. (**c**) The plot shows a comparison of er_2, er_∞, and er_{rms} using Talbot's method for different τ with $M_T = 26$, $alpha = 0.3$, and $n = 25$ for Example 3. (**d**) Contour plot of the absolute error obtained with $M_T = 26$, $\tau = 1$ $alpha = 0.5$, and $n = 25$ using Talbot's method for Example 3.

4. Conclusions

In this article, a numerical method based on the LT coupled with the CSCM for the numerical solution of two-dimensional TFADEs in the Caputo sense has been developed. It first uses the LT to convert a TFADE into a time-independent inhomogeneous equation in the LT space. Then, it uses the CSCM to discretize the spatial derivatives of this Laplace-transformed inhomogeneous equation. Finally, it chooses Stehfest's method and Talbot's method for the numerical inversion of the LT to retrieve the numerical solutions of the TFADE from the corresponding CSCM solutions in the LT domain. Compared with the finite difference time-stepping scheme, the proposed method introduces the LT technique to avoid the calculation of a costly convolution integral in the approximation of time-fractional derivation and to avoid the effect of the time step on the stability and accuracy. Hence, it can successfully solve the long time history TFADEs. The proposed advection–diffusion equation of fractional order presents significant potential for extension into uncertainty

propagation and sensitivity analysis, enabling a more comprehensive and nuanced understanding of system responses to parameter variations and initial conditions [49–51].

Author Contributions: Conceptualization, F.A.S. and K.; methodology, F.A.S. and K.; software, F.A.S. and K.;validation, W.B., A.K. and N.M.; formal analysis, W.B., A.K. and N.M.; investigation, K., W.B., A.K. and N.M.; resources, W.B., A.K. and N.M.; data curation, F.A.S., K., W.B., A.K. and N.M.; writing—original draft preparation, F.A.S., K.; writing—review and editing, F.A.S. and K.; visualization, W.B., N.M.; supervision, W.B., A.K. and N.M.; project administration, W.B. and N.M.; funding acquisition, W.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: All data required for this research are included within the paper.

Acknowledgments: The authors W. Boulila, A. Koubaa and N. Mlaiki would like to thank the Prince Sultan University for paying the publication fees for this work through RIOTU LAB.

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

References

1. Khan, Z.; Shah, K.; Abdalla, B.; Abdeljawad, T. A numerical Study of Complex Dynamics of a Chemostat Model Under Fractal-Fractional Derivative. *Fractals* **2023**, *31*, 2340181. [CrossRef]
2. Shah, K.; Amin, R.; Abdeljawad, T. Utilization of Haar wavelet collocation technique for fractal-fractional order problem. *Heliyon*. **2023**, *9*, e17123. [CrossRef] [PubMed]
3. Podlubny, I. *Fractional Differential Equations: An Introduction to Fractional Derivatives, Fractional Differential Equations, to Methods of Their Solution and Some of Their Applications*; Elsevier: Amsterdam, The Netherlands, 1998.
4. Oldham, K.; Spanier, J. *The Fractional Calculus Theory and Applications of Differentiation and Integration to Arbitrary Order*; Elsevier: Amsterdam, The Netherlands, 1974.
5. Tarasov, V.E. Geometric interpretation of fractional-order derivative. *Fract. Calc. Appl. Anal.* **2016**, *19*, 1200–1221. [CrossRef]
6. Roohi, M.; Zhang, C.; Taheri, M.; Basse-ÓConnor, A. Synchronization of Fractional-Order Delayed Neural Networks Using Dynamic-Free Adaptive Sliding Mode Control. *Fractal Fract.* **2023**, *7*, 682. [CrossRef]
7. Zaid, S.A.; Bakeer, A.; Albalawi, H.; Alatwi, A.M.; AbdelMeguid, H.; Kassem, A.M. Optimal Fractional-Order Controller for the Voltage Stability of a DC Microgrid Feeding an Electric Vehicle Charging Station. *Fractal Fract.* **2023**, *7*, 677. [CrossRef]
8. Gharab, S.; Feliu Batlle, V. Fractional Control of a Class of Underdamped Fractional Systems with Time Delay-Application to a Teleoperated Robot with a Flexible Link. *Fractal Fract.* **2023**, *7*, 646. [CrossRef]
9. Guechi, S.; Dhayal, R.; Debbouche, A.; Malik, M. Analysis and Optimal Control of ϕ-Hilfer Fractional Semilinear Equations Involving Nonlocal Impulsive Conditions. *Symmetry* **2021**, *13*, 2084. [CrossRef]
10. Benson, D.A.; Wheatcraft, S.W.; Meerschaert, M.M. Application of a fractional advection-dispersion equation. *Water Resour. Res.* **2000**, *36*, 1403–1412. [CrossRef]
11. Magin, R.L. *Fractional Calculus in Bioengineering*; Begell House Publishers: Danbury, CT, USA, 2006.
12. Sun, H.; Zhang, Y.; Baleanu, D.; Chen, W.; Chen, Y. A new collection of real world applications of fractional calculus in science and engineering. *Commun. Nonlinear Sci. Numer. Simul.* **2018**, *64*, 213–231. [CrossRef]
13. Owolabi, K.M. High-dimensional spatial patterns in fractional reaction diffusion system arising in biology. *Chaos Solitons Fractals* **2020**, *134*, 109723. [CrossRef]
14. Isenberg, J.; Gutfinger, C. Heat transfer to a draining film. *Int. J. Heat Mass Transf.* **1973**, *16*, 505–512. [CrossRef]
15. Kumar, N. Unsteady flow against dispersion in finite porous media. *J. Hydrol.* **1983**, *63*, 345–358. [CrossRef]
16. Lanser, D.; Verwer, J.G. Analysis of operator splitting for advection–diffusion–reaction problems from air pollution modelling. *J. Comput. Appl. Math.* **1999**, *111*, 201–216. [CrossRef]
17. Zhao, Y.L.; Huang, T.Z.; Gu, X.M.; Luo, W.H. A fast second-order implicit difference method for time space fractional advection diffusion equation. *Numer. Funct. Anal. Optim.* **2020**, *41*, 257–293. [CrossRef]
18. Mardani, A.; Hooshmandasl, M.R.; Heydari, M.H.; Cattani, C. A meshless method for solving the time fractional advection diffusion equation with variable coefficients. *Comput. Math. Appl.* **2018**, *75*, 122–133. [CrossRef]
19. Mohyud-Din, S.T.; Akram, T.; Abbas, M.; Ismail, A.I.; Ali, N.H. A fully implicit finite difference scheme based on extended cubic B-splines for time fractional advection-diffusion equation. *Adv. Differ. Equations* **2018**, *2018*, 1–17. [CrossRef]
20. Abbaszadeh, M.; Amjadian, H. Second order finite difference or spectral element formulation for solving the fractional advection diffusion equation. *Commun. Appl. Math. Comput.* **2020**, *2*, 653–669. [CrossRef]
21. Gottlieb, D.; Orszag, S.A. *Numerical Analysis of Spectral Methods: Theory and Applications*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 1977.
22. Canuto, C.; Hussaini, M.Y.; Quarteroni, A.; Zang, T.A. *Spectral Methods, Evolution to Complex Geometries and Applications to Fluid Dynamics*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2007.

23. Trefethen, L.N. *Spectral Methods in MATLAB, Volume 10 of Software, Environments, and Tools*; Society for Industrial and Applied Mathematics (SIAM): Philadelphia, PA, USA, 2000; Volume 24, p. 57.
24. Welfert, B.D. Generation of pseudospectral differentiation matrices I. *Siam J. Numer. Anal.* **1997**, *34*, 1640–1657. [CrossRef]
25. Mao, Z.; Shen, J. Hermite spectral methods for fractional PDEs in unbounded domains. *Siam J. Sci. Comput.* **2017**, *39*, A1928–A1950. [CrossRef]
26. Sweilam, N.H.; Khader, M.M.; Adel, M. Chebyshev pseudo-spectral method for solving fractional advection-dispersion equation. *Appl. Math.* **2014**, *5*, 3240. [CrossRef]
27. Bhrawy, A.H.; Baleanu, D. A spectral Legendre-Gauss-Lobatto collocation method for a space-fractional advection diffusion equations with variable coefficients. *Rep. Math. Phys.* **2013**, *72*, 219–233. [CrossRef]
28. Tian, W.; Deng, W.; Wu, Y. Polynomial spectral collocation method for space fractional advection-diffusion equation. *Numer. Methods Partial. Differ. Equ.* **2014**, *30*, 514–535. [CrossRef]
29. Sweilam, N.H.; El Sayed, A.A.E.; Boulaaras, S. Fractional order advection dispersion problem solution via the spectral collocation method and the non standard finite difference technique. *Chaos Solitons Fractals* **2021**, *144*, 110736. [CrossRef]
30. Kamran; Khan, S.; Alhazmi, S.E.; Alotaibi, F.M.; Ferrara, M.; Ahmadian, A. On the Numerical Approximation of Mobile-Immobile Advection-Dispersion Model of Fractional Order Arising from Solute Transport in Porous Media. *Fractal Fract.* **2022**, *6*, 445. [CrossRef]
31. Davies, A.J.; Crann, D.; Kane, S.J.; Lai, C.H. A hybrid Laplace transform/finite difference boundary element method for diffusion problems. *Comput. Model. Eng. Sci.* **2007**, *18*, 79–86.
32. Dingfelder, B.; Weideman, J.A.C. An improved Talbot method for numerical Laplace transform inversion. *Numer. Algorithms* **2015**, *68*, 167–183. [CrossRef]
33. Kamran; Kamal, R.; Rahmat, G.; Shah, K. On the Numerical Approximation of Three-Dimensional Time Fractional Convection-Diffusion Equations. *Math. Probl. Eng.* **2021**, *2021*, 1–16. [CrossRef]
34. Kamal, R.; Kamran; Rahmat, G.; Ahmadian, A.; Arshad, N.I.; Salahshour, S. Approximation of linear one dimensional partial differential equations including fractional derivative with non-singular kernel. *Adv. Differ. Equ.* **2021**, *2021*, 1–15. [CrossRef]
35. Stehfest, H. Algorithm 368: Numerical inversion of Laplace transforms [D5]. *Commun. Acm* **1970**, *13*, 47–49. [CrossRef]
36. Talbot, A. The accurate numerical inversion of Laplace transforms. *Ima J. Appl. Math.* **1979**, *23*, 97–120. [CrossRef]
37. Shokri, A.; Mirzaei, S. A pseudo-spectral based method for time-fractional advection-diffusion equation. *Comput. Methods Differ. Equ.* **2020**, *8*, 454–467.
38. Baltensperger, R.; Trummer, M.R. Spectral differencing with a twist. *Siam J. Sci. Comput.* **2003**, *24*, 1465–1487. [CrossRef]
39. Börm, S.; Grasedyck, L.; Hackbusch, W. Introduction to hierarchical matrices with applications. *Eng. Anal. Bound. Elem.* **2003**, *27*, 405–422. [CrossRef]
40. Crump, K.S. Numerical inversion of Laplace transforms using a Fourier series approximation. *J. ACM* **1976**, *23*, 89–96. [CrossRef]
41. De Hoog, F.R.; Knight, J.H.; Stokes, A.N. An improved method for numerical inversion of Laplace transforms. *SIAM J. Sci. Stat. Comput.* **1982**, *3*, 357–366. [CrossRef]
42. Gaver, D.P., Jr. Observing stochastic processes, and approximate transform inversion. *Oper. Res.* **1966**, *14*, 444–459. [CrossRef]
43. Weeks, W.T. Numerical inversion of Laplace transforms using Laguerre functions. *J. ACM* **1966**, *13*, 419–429. [CrossRef]
44. Kamran; Khan, S.U.; Haque, S.; Mlaiki, N. On the Approximation of Fractional-Order Differential Equations Using Laplace Transform and Weeks Method. *Symmetry* **2023**, *15*, 1214. [CrossRef]
45. Kuznetsov, A. On the Convergence of the Gaver–Stehfest Algorithm. *Siam J. Numer. Anal.* **2013**, *51*, 2984–2998. [CrossRef]
46. Davies, B.; Martin, B. Numerical inversion of the Laplace transform: A survey and comparison of methods. *J. Comput. Phys.* **1979**, *33*, 1–32. [CrossRef]
47. Abate, J.; Whitt, W. A unified framework for numerically inverting Laplace transforms. *Informs J. Comput.* **2006**, *18*, 408–421. [CrossRef]
48. Fu, Z.J.; Chen, W.; Yang, H.T. Boundary particle method for Laplace transformed time fractional diffusion equations. *J. Comput. Phys.* **2013**, *235*, 52–66. [CrossRef]
49. Boulila, W.; Ayadi, Z.; Farah, I. Sensitivity analysis approach to model epistemic and aleatory imperfection: Application to Land Cover Change prediction model. *J. Comput. Sci.* **2017**, *23*, 58–70. [CrossRef]
50. Ferchichi, A.; Boulila, W.; Farah, I. Propagating aleatory and epistemic uncertainty in land cover change prediction process. *Ecol. Inform.* **2017**, *37*, 24–37. [CrossRef]
51. Ferchichi, A.; Boulila, W.; Farah, I. Reducing uncertainties in land cover change models using sensitivity analysis. *Knowl. Inf. Syst.* **2018**, *55*, 719–740. [CrossRef]