J. Math. Computer Sci., 23 (2021), 110–123

Online: ISSN 2008-949X

Journal of Mathematics and Computer Science

Journal Homepage: www.isr-publications.com/jmcs

Fast O(N) hybrid Laplace transform-finite difference method in solving 2D time fractional diffusion equation

Fouad Mohammad Salama*, Norhashidah Hj. Mohd Ali, Nur Nadiah Abd Hamid

School of Mathematical Sciences, Universiti Sains Malaysia, 11800 Penang, Malaysia.

Abstract

It is time-memory consuming when numerically solving time fractional partial differential equations, as it requires $O(N^2)$ computational cost and $O(MN)$ memory complexity with finite difference methods, where, N and M are the total number of time steps and spatial grid points, respectively. To surmount this issue, we develop an efficient hybrid method with $O(N)$ computational cost and $O(M)$ memory complexity in solving two-dimensional time fractional diffusion equation. The presented method is based on the Laplace transform method and a finite difference scheme. The stability and convergence of the proposed method are analyzed rigorously by the means of the Fourier method. A comparative study drawn from numerical experiments shows that the hybrid method is accurate and reduces the computational cost, memory requirement as well as the CPU time effectively compared to a standard finite difference scheme.

Keywords: Caputo fractional derivative, fractional diffusion equation, Laplace transform, finite difference scheme, stability and convergence analyses.

2020 MSC: 35R11, 44A10, 65M12.

1. Introduction

Fractional calculus which entails integrals and derivatives of fractional orders is one of the essential topics of applied mathematics. In particular, much attention has been directed recently to fractional differential equations (FDEs). This growing interest in FDEs is obviously attributed to their ability in modeling many problems in engineering [6, 32], viscoelasticity [20], hydrology [4], bio science [19] and other sciences [17, 22, 31].

Attempting to seek solutions of FDEs, several approaches including analytical and numerical methods are suggested. Adomian decomposition [7], homotopy analysis [8] and tau method [16] are such examples of analytical and approximate methods that can be found in literature. However, analytic solutions for most fractional differential equations can not be obtained explicitly [10]. Consequently, significant effort has been invested in developing numerical methods for solving these equations. Of numerical methods, finite difference methods are prominently plentiful [1, 3, 11, 21, 23, 33].

In this study, we consider the following two-dimensional time fractional diffusion equation (TFDE)

$$ {}_0^C D_t^\alpha u(x,y,t) = a_1 \frac{\partial^2 u(x,y,t)}{\partial x^2} + a_2 \frac{\partial^2 u(x,y,t)}{\partial y^2} + f(x,y,t), \quad 0 < \alpha < 1, \tag{1.1} $$

where $a_1, a_2 > 0$, and $f(x,y,t)$ is the source term with the initial and boundary conditions

*Corresponding author
Email address: fuadmohd321@gmail.com (Fouad Mohammad Salama)

doi: 10.22436/jmcs.023.02.04

Received: 2020-05-12 *Revised:* 2020-08-15 *Accepted:* 2020-09-15

$$\begin{aligned} u(x,y,0) &= g(x,y), \quad x,y \in \Omega, \\ u(x,0,t) &= g_1(x,t), \quad u(x,L,t) = g_2(x,t), \\ u(0,y,t) &= g_3(y,t), \quad u(L,y,t) = g_4(y,t), \qquad 0 \leqslant t \leqslant T, \end{aligned}$$

in which $\Omega = \{(x,y)|0 \leqslant x \leqslant L, 0 \leqslant y \leqslant L\}$, and ${}_0^C D_t^\alpha u(x,y,t)$ is the Caputo fractional derivative given as follows:

$${}_0^C D_t^\alpha u(x,y,t) = \frac{1}{\Gamma(n-\alpha)} \int_0^t \frac{u^{(n)}(x,y,\tau)}{(t-\tau)^{\alpha+1-n}} d\tau.$$

The existing discretization schemes constructed based on finite difference approximations for solving Eq. (1.1) require large computational cost and memory requirement. Such numerical schemes necessitate $O(N^2)$ computational cost and $O(MN)$ memory complexity [12, 14]. These considerable computational challenges are mainly caused by the non-local property of the fractional derivative which employs the solution values on all previous time steps to compute the solution at the current time level [26]. This is in contrast with the classical diffusion equation, where the solution need to be stored at a fixed number of time steps and the computational cost is linear with respect to N. To surmount such computational challenges in solving FDEs numerically, several methods including parallel computing [9, 13], short memory principle [34], and multigrid method [24] have been suggested. In this study, we propose a hybrid method which is based on Laplace transform technique and Crank-Nicolson difference scheme to achieve an economical computational solution of the time fractional diffusion equation (1.1). To summarize, the Laplace transform method is used at first to approximate the Caputo-type time fractional derivative and reduce the original TFDE (1.1) to its corresponding partial differential equation (PDE). Afterwards, a Crank-Nicolson difference scheme is employed for solving the obtained PDE which result in a low cost and fast solution method of the original TFDE. The stability and convergence of the developed numerical scheme are discussed via the Fourier method. The advantages of the proposed method are demonstrated in terms of the computational cost, memory complexity and execution times when compared to an existing standard finite difference scheme.

The rest of this manuscript is arranged as follows. In Section 2, we show the formulation of the proposed method. The stability and convergence analyses are then discussed in Sections 3 and 4, respectively. An existing standard finite difference scheme is presented in Section 5. The computational cost and memory complexity of both standard and hybrid methods are also analyzed in the same section. The computational superiority of the hybrid method over the standard scheme has been illustrated through numerical experiments in Section 6. Eventually, conclusions are made in Section 7.

2. Construction of the hybrid method

Due to the non local property of the fractional derivative, solving problem (1.1) using standard finite difference methods (such as explicit or implicit scheme with a certain discretization formula for the time fractional derivative), necessitates the storage of the solution outcomes at all preceding time levels to compute the solution at the current time level. To overcome this issue, we use the Laplace transform method and linearization property proposed by Ren et al. [27] and utilized in [5, 26, 28–30] to approximate the Caputo fractional derivative and convert the TFDE (1.1) to its corresponding PDE. The resulting PDE will then be solved using Crank-Nicolson difference scheme, which generates a numerical solution close to the exact solution of the original problem (1.1) with significant savings in computational cost and memory requirement.

The time fractional derivative ${}_0^C D_t^\alpha u(x,y,t)$ can be approximated using Laplace transform as follows [25]

$$L\{{}_0^C D_t^\alpha u(x,y,t)\} = s^\alpha u(x,y,s) - s^{\alpha-1} u(x,y,0) = s^\alpha [u(x,y,s) - s^{-1} u(x,y,0)], \tag{2.1}$$

in which $u(x,y,s)$ is the Laplace transform of $u(x,y,t)$. As we have $0 < \alpha < 1$, then the term s^α is linearized as

$$s^\alpha \approx \alpha s^1 + (1-\alpha)s^0 = \alpha s + (1-\alpha). \tag{2.2}$$

By setting (2.2) into (2.1), we get the following expression

$$\begin{aligned} L\{{}_0^C D_t^\alpha u(x,y,t)\} &\approx [\alpha s + (1-\alpha)][u(x,y,s) - s^{-1}u(x,y,0)] \\ &= \alpha s[u(x,y,s) - s^{-1}u(x,y,0)] + (1-\alpha)[u(x,y,s) - s^{-1}u(x,y,0)]. \end{aligned}$$

Now, the inverse Laplace transform may be applied to yield

$${}_0^C D_t^\alpha u(x,y,t) \approx \alpha \frac{\partial u(x,y,t)}{\partial t} + (1-\alpha)[u(x,y,t) - u(x,y,0)]. \tag{2.3}$$

By taking Eq. (2.3) into consideration, the original TFDE (1.1) is simplified to the following PDE

$$\begin{aligned} &\frac{\partial u(x,y,t)}{\partial t} = A_1 \frac{\partial^2 u(x,y,t)}{\partial x^2} + A_2 \frac{\partial^2 u(x,y,t)}{\partial y^2} - (r-1)u(x,y,t) + (r-1)g(x,y) + rf(x,y,t), \\ &u(x,y,0) = g(x,y), \quad x,y \in \Omega, \\ &u(x,0,t) = g_1(x,t), \quad u(x,L,t) = g_2(x,t), \\ &u(0,y,t) = g_3(y,t), \quad u(L,y,t) = g_4(y,t), \qquad 0 \leqslant t \leqslant T, \end{aligned} \tag{2.4}$$

where $A_1 = \frac{a_1}{\alpha}$, $A_2 = \frac{a_2}{\alpha}$ and $r = \frac{1}{\alpha}$. This approach removes the fractional derivative of the original problem (1.1) and turns it into a PDE, which can minimize the computational cost and memory complexity. Next, we utilize a finite difference scheme to attain the numerical solution of the PDE (2.4), which leads to obtaining an approximate-numerical solution of the TFDE (1.1). Define a uniform solution domain with $t_k = k\Delta t$, $k = 0,1,2,\ldots,N$, $x_i = i\Delta x$, $i = 0,1,2,\ldots,M_x$, and $y_j = j\Delta y$, $j = 0,1,2,\ldots,M_y$, where M_x, M_y and N are positive integers. Based on the central difference approximations in time and space directions about the point $(x_i, y_j, t_{k+1/2})$, the following Crank-Nicolson difference scheme is utilized to discretize the simplified problem (2.4)

$$\begin{aligned} \frac{u_{i,j}^{k+1} - u_{i,j}^k}{\Delta t} &= \frac{A_1}{2}\left(\frac{u_{i+1,j}^{k+1} - 2u_{i,j}^{k+1} + u_{i-1,j}^{k+1}}{(\Delta x)^2} + \frac{u_{i+1,j}^k - 2u_{i,j}^k + u_{i-1,j}^k}{(\Delta x)^2}\right) \\ &+ \frac{A_2}{2}\left(\frac{u_{i,j+1}^{k+1} - 2u_{i,j}^{k+1} + u_{i,j-1}^{k+1}}{(\Delta y)^2} + \frac{u_{i,j+1}^k - 2u_{i,j}^k + u_{i,j-1}^k}{(\Delta y)^2}\right) \\ &- (r-1)\left(\frac{u_{i,j}^{k+1} + u_{i,j}^k}{2}\right) + (r-1)u_{i,j}^0 + rf_{i,j}^{k+1/2} + O\left((\Delta t)^2 + (\Delta x)^2 + (\Delta y)^2\right). \end{aligned} \tag{2.5}$$

Upon simplification and neglecting higher order terms, the following fully-discrete Crank-Nicolson scheme is obtained:

$$\begin{aligned} u_{i,j}^{k+1} = \frac{1}{1 + 0.5(r-1)\Delta t + q_1 + q_2}\Bigg[&\frac{q_1}{2}\left(u_{i+1,j}^{k+1} + u_{i-1,j}^{k+1} + u_{i+1,j}^k + u_{i-1,j}^k\right) \\ &+ \frac{q_2}{2}\left(u_{i,j+1}^{k+1} + u_{i,j-1}^{k+1} + u_{i,j+1}^k + u_{i,j-1}^k\right) + (1 - 0.5(r-1)\Delta t - q_1 - q_2)u_{i,j}^k \\ &+ (r-1)\Delta t u_{i,j}^0 + r\Delta t f_{i,j}^{k+1/2}\Bigg], \end{aligned} \tag{2.6}$$

in which $q_1 = \frac{A_1\Delta t}{\Delta x^2}$, $q_2 = \frac{A_2\Delta t}{\Delta y^2}$ with the initial and boundary conditions

$$u_{i,j}^0 = g(x_i, y_j), \quad u_{i,0}^k = g_1(x_i, t_k), \quad u_{i,L}^k = g_2(x_i, t_k), \quad u_{0,j}^k = g_3(y_j, t_k), \quad u_{L,j}^k = g_4(y_j, t_k),$$

for $i = 1,2,\ldots,M_x - 1$, $j = 1,2,\ldots,M_y - 1$ and $k = 0,1,\ldots,N-1$.

3. Stability analysis

In this section, we analyze the stability of the difference scheme (2.6) by the means of the Fourier analysis method. Assume that $U_{i,j}^k$ is the approximate solution of Eq. (2.6), then the error can be defined as follows:

$$\xi_{i,j}^k = u_{i,j}^k - U_{i,j}^k, \quad 1 \leqslant i \leqslant M_x - 1, \quad 1 \leqslant j \leqslant M_y - 1, \quad 0 \leqslant k \leqslant N.$$

Utilizing the above definition along with the Eq. (2.6), we can easily obtain the following round-off error equation:

$$\begin{aligned} &\frac{-q_1}{2}(\xi_{i+1,j}^{k+1} + \xi_{i-1,j}^{k+1}) + (1 + 0.5(r-1)\Delta t + q_1 + q_2)\xi_{i,j}^{k+1} - \frac{q_2}{2}(\xi_{i,j+1}^{k+1} + \xi_{i,j-1}^{k+1}) \\ &\quad = \frac{q_1}{2}(\xi_{i+1,j}^{k} + \xi_{i-1,j}^{k}) + (1 - 0.5(r-1)\Delta t - q_1 - q_2)\xi_{i,j}^{k} + \frac{q_2}{2}(\xi_{i,j+1}^{k} + \xi_{i,j-1}^{k}) + (r-1)\Delta t \xi_{i,j}^{0}. \end{aligned} \tag{3.1}$$

For $k = 0, 1, \ldots, N$, we define the discrete function

$$\xi^k(x,y) = \begin{cases} \xi_{i,j}^k, & x_{i-\frac{\Delta x}{2}} < x \leqslant x_{i+\frac{\Delta x}{2}}, y_{j-\frac{\Delta y}{2}} < y \leqslant y_{j+\frac{\Delta y}{2}}, \\ 0, & 0 \leqslant x \leqslant \frac{\Delta x}{2} \text{ or } L - \frac{\Delta x}{2} \leqslant x \leqslant L, \\ 0, & 0 \leqslant y \leqslant \frac{\Delta y}{2} \text{ or } L - \frac{\Delta y}{2} \leqslant y \leqslant L. \end{cases}$$

Expanding $\xi^k(x,y)$ in a Fourier series, yields

$$\xi^k(x,y) = \sum_{l_1=-\infty}^{\infty} \sum_{l_2=-\infty}^{\infty} \lambda^k(l_1, l_2) e^{2\pi\sqrt{-1}(l_1 x/L + l_2 y/L)},$$

in which

$$\lambda^k(l_1, l_2) = \frac{1}{L^2} \int_0^L \int_0^L \xi^k(x,y) e^{-2\pi\sqrt{-1}(l_1 x/L + l_2 y/L)} dx dy.$$

Utilizing the l^2 norm definition together with the Parseval's equality, we obtain

$$\|\xi^k\|_2^2 = \sum_{i=1}^{M_x-1} \sum_{j=1}^{M_y-1} \Delta x \Delta y |\xi_{i,j}^k|^2 = \sum_{l_1=-\infty}^{\infty} \sum_{l_2=-\infty}^{\infty} |\lambda^k(l_1, l_2)|^2.$$

Next, assume the solution of Eq. (3.1) is of the following form:

$$\xi_{i,j}^k = \lambda^k e^{\sqrt{-1}(\beta_1 i \Delta x + \beta_2 j \Delta y)}, \tag{3.2}$$

where $\beta_1 = 2\pi l_1/L$ and $\beta_2 = 2\pi l_2/L$. Substituting (3.2) into (3.1), yields

$$\lambda^{k+1} = \frac{1 - 0.5(r-1)\Delta t - \rho_1 - \rho_2}{1 + 0.5(r-1)\Delta t + \rho_1 + \rho_2} \lambda^k + \frac{(r-1)\Delta t}{1 + 0.5(r-1)\Delta t + \rho_1 + \rho_2} \lambda^0, \tag{3.3}$$

where

$$\rho_1 = 2q_1 \sin^2\left(\frac{\beta_1 \Delta x}{2}\right), \quad \rho_2 = 2q_2 \sin^2\left(\frac{\beta_2 \Delta y}{2}\right).$$

Proposition 3.1. *If* $2 - (r-1)\Delta t \geqslant 0$ *and* λ^k $(k = 0, 1, \ldots, N-1)$ *satisfies Eq.* (3.3), *then* $|\lambda^{k+1}| \leqslant |\lambda^0|$.

Proof. We use mathematical induction to prove the above result. For $k = 0$ and according to Eq. (3.3), we have

$$|\lambda^1| \leqslant \left|\frac{1+0.5(r-1)\Delta t-\rho_1-\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\right||\lambda^0|.$$

Since $\rho_1 \geqslant 0$ and $\rho_2 \geqslant 0$, we get

$$|\lambda^1| \leqslant |\lambda^0|.$$

Next, supposing that $|\lambda^s| \leqslant |\lambda^0|$ for $s = 1, 2, \ldots, k$. From Eq. (3.3) we obtain

$$\begin{aligned}|\lambda^{k+1}| &\leqslant \left|\frac{1-0.5(r-1)\Delta t-\rho_1-\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\right||\lambda^k| + \left|\frac{(r-1)\Delta t}{1+(r-1)\Delta t+\rho_1+\rho_2}\right||\lambda^0| \\ &\leqslant \left|\frac{1-0.5(r-1)\Delta t-\rho_1-\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\right||\lambda^0| + \frac{(r-1)\Delta t}{1+(r-1)\Delta t+\rho_1+\rho_2}|\lambda^0|.\end{aligned}$$

If $1-0.5(r-1)\Delta t-\rho_1-\rho_2 \geqslant 0$ and since $\rho_1, \rho_2 \geqslant 0$, then

$$|\lambda^{k+1}| \leqslant \frac{1-0.5(r-1)\Delta t-\rho_1-\rho_2+(r-1)\Delta t}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}|\lambda^0| = \frac{1+0.5(r-1)\Delta t-\rho_1-\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}|\lambda^0| \leqslant |\lambda^0|.$$

If $1-0.5(r-1)\Delta t-\rho_1-\rho_2 < 0$, then

$$|\lambda^{k+1}| \leqslant \frac{-1+0.5(r-1)\Delta t+\rho_1+\rho_2+(r-1)\Delta t}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}|\lambda^0| = \frac{-1+1.5(r-1)\Delta t+\rho_1+\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}|\lambda^0|.$$

Here,

$$\begin{aligned}&|\lambda^{k+1}| \leqslant |\lambda^0| \\ &\quad\Leftrightarrow \frac{-1+1.5(r-1)\Delta t+\rho_1+\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2} \leqslant 1 \\ &\quad\Leftrightarrow -1+1.5(r-1)\Delta t+\rho_1+\rho_2 \leqslant 1+0.5(r-1)\Delta t+\rho_1+\rho_2 \\ &\quad\Leftrightarrow 2-(r-1)\Delta t \geqslant 0.\end{aligned}$$

□

Now, using Proposition 3.1 along with the Parseval's equality, we obtain the following result:

$$\|\xi^k\|_2^2 = \sum_{l_1=-\infty}^{\infty}\sum_{l_2=-\infty}^{\infty}|\lambda^k(l_1,l_2)|^2 \leqslant \sum_{l_1=-\infty}^{\infty}\sum_{l_2=-\infty}^{\infty}|\lambda^0(l_1,l_2)|^2 = \|\xi^0\|_2^2,$$

that is

$$\|\xi^k\|_2^2 \leqslant \|\xi^0\|_2^2.$$

According to the above analysis, it can be seen that the difference scheme (2.6) is stable given that $2-(r-1)\Delta t \geqslant 0$ is satisfied.

4. Convergence analysis

This section investigates the convergence of the difference scheme (2.6) by following an analogous approach to that of Section 3. Suppose that the truncation error at the grid point $(x_i, y_j, t_{k+1/2})$ is denoted by $R_{i,j}^{k+1/2}$. From Eq. (2.5) there is a positive constant C_1 such that for all the values of i, j and k, we have

$$R_{i,j}^{k+1/2} \leqslant C_1((\Delta t)^2+(\Delta x)^2+(\Delta y)^2), \tag{4.1}$$

where

$$C_1 = \max_{1\leqslant i\leqslant M_x-1, 1\leqslant j\leqslant M_y-1, 0\leqslant k\leqslant N} \left\{C_{i,j}^k\right\}.$$

Suppose that the exact solution is denoted by $U_{i,j}^k$ while the approximate solution is denoted by $u_{i,j}^k$. Consequently, the exact solution at the time level $k+1/2$ is given as follows:

$$\begin{aligned} U_{i,j}^{k+1} = \frac{1}{1+0.5(r-1)\Delta t+q_1+q_2}\Bigg[&\frac{q_1}{2}\left(U_{i+1,j}^{k+1}+U_{i-1,j}^{k+1}+U_{i+1,j}^{k}+U_{i-1,j}^{k}\right)\\ &+\frac{q_2}{2}\left(U_{i,j+1}^{k+1}+U_{i,j-1}^{k+1}+U_{i,j+1}^{k}+U_{i,j-1}^{k}\right)+(1-0.5(r-1)\Delta t-q_1-q_2)U_{i,j}^{k}\\ &+(r-1)\Delta t U_{i,j}^{0}+r\Delta t f_{i,j}^{k+1/2}+\Delta t R_{i,j}^{k+1/2}\Bigg]. \end{aligned} \tag{4.2}$$

The error can be represented as $E_{i,j}^k = U_{i,j}^k - u_{i,j}^k$. By Subtracting (2.6) form (4.2), the next error equation follows immediately as

$$\begin{aligned} E_{i,j}^{k+1} = \frac{1}{1+0.5(r-1)\Delta t+q_1+q_2}\Bigg[&\frac{q_1}{2}\left(E_{i+1,j}^{k+1}+E_{i-1,j}^{k+1}+E_{i+1,j}^{k}+E_{i-1,j}^{k}\right)\\ &+\frac{q_2}{2}\left(E_{i,j+1}^{k+1}+E_{i,j-1}^{k+1}+E_{i,j+1}^{k}+E_{i,j-1}^{k}\right)+(1-0.5(r-1)\Delta t-q_1-q_2)E_{i,j}^{k}\\ &+(r-1)\Delta t E_{i,j}^{0}+\Delta t R_{i,j}^{k+1/2}\Bigg] \end{aligned} \tag{4.3}$$

and

$$\begin{aligned} &E_{i,j}^0 = 0, \quad 0\leqslant i\leqslant M_x, 0\leqslant j\leqslant M_y,\\ &E_{0,j}^k = E_{M_x,j}^k = 0, \quad 0\leqslant j\leqslant M_y, 0\leqslant k\leqslant N,\\ &E_{i,0}^k = E_{i,M_y}^k = 0, \quad 0\leqslant i\leqslant M_x, 0\leqslant k\leqslant N. \end{aligned}$$

For $k = 0, 1, \ldots, N$, we define the discrete functions

$$E^k(x,y) = \begin{cases} E_{i,j}^k, & x_{i-\frac{\Delta x}{2}} < x \leqslant x_{i+\frac{\Delta x}{2}}, y_{j-\frac{\Delta y}{2}} < y \leqslant y_{j+\frac{\Delta y}{2}},\\ 0, & 0\leqslant x\leqslant \frac{\Delta x}{2} \text{ or } L-\frac{\Delta x}{2}\leqslant x\leqslant L,\\ 0, & 0\leqslant y\leqslant \frac{\Delta y}{2} \text{ or } L-\frac{\Delta y}{2}\leqslant y\leqslant L, \end{cases}$$

and

$$R^k(x,y) = \begin{cases} R_{i,j}^k, & x_{i-\frac{\Delta x}{2}} < x \leqslant x_{i+\frac{\Delta x}{2}}, y_{j-\frac{\Delta y}{2}} < y \leqslant y_{j+\frac{\Delta y}{2}},\\ 0, & 0\leqslant x\leqslant \frac{\Delta x}{2} \text{ or } L-\frac{\Delta x}{2}\leqslant x\leqslant L,\\ 0, & 0\leqslant y\leqslant \frac{\Delta y}{2} \text{ or } L-\frac{\Delta y}{2}\leqslant y\leqslant L. \end{cases}$$

Next, Fourier series expansions of the functions $E^k(x,y)$ and $R^k(x,y)$ can be written as follows:

$$E^k(x,y) = \sum_{l_1=-\infty}^{\infty}\sum_{l_2=-\infty}^{\infty} \psi^k(l_1,l_2) e^{2\pi\sqrt{-1}(l_1x/L+l_2y/L)},$$

$$R^k(x,y) = \sum_{l_1=-\infty}^{\infty}\sum_{l_2=-\infty}^{\infty} \phi^k(l_1,l_2) e^{2\pi\sqrt{-1}(l_1x/L+l_2y/L)},$$

in which

$$\psi^k(l_1,l_2) = \frac{1}{L^2}\int_0^L\int_0^L E^k(x,y) e^{-2\pi\sqrt{-1}(l_1x/L+l_2y/L)}dxdy, \tag{4.4}$$

$$\phi^k(l_1,l_2)=\frac{1}{L^2}\int_0^L\int_0^L R^k(x,y)e^{-2\pi\sqrt{-1}(l_1x/L+l_2y/L)}dxdy. \tag{4.5}$$

Utilizing the l^2 norm definition together with the Parseval's equality, we obtain

$$\|E^k\|_2^2=\sum_{i=1}^{M_x-1}\sum_{j=1}^{M_y-1}\Delta x\Delta y\left|E_{i,j}^k\right|^2=\sum_{l_1=-\infty}^{\infty}\sum_{l_2=-\infty}^{\infty}\left|\psi^k(l_1,l_2)\right|^2, \tag{4.6}$$

$$\|R^k\|_2^2=\sum_{i=1}^{M_x-1}\sum_{j=1}^{M_y-1}\Delta x\Delta y\left|R_{i,j}^k\right|^2=\sum_{l_1=-\infty}^{\infty}\sum_{l_2=-\infty}^{\infty}\left|\phi^k(l_1,l_2)\right|^2. \tag{4.7}$$

Based on the above analysis, we assume that the solution of Eq. (4.3) has the following forms:

$$E_{i,j}^k=\psi^k e^{\sqrt{-1}(\beta_1 i\Delta x+\beta_2 j\Delta y)},\quad R_{i,j}^k=\phi^k e^{\sqrt{-1}(\beta_1 i\Delta,x+\beta_2 j\Delta y)}, \tag{4.8}$$

where $\beta_1=2\pi l_1/L$ and $\beta_2=2\pi l_2/L$. Substituting Eq. (4.8) into Eq. (4.3), we get

$$\psi^{k+1}=\frac{1-0.5(r-1)\Delta t-\rho_1-\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\psi^k+\frac{(r-1)\Delta t\psi^0+\Delta t\phi^{k+1/2}}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}, \tag{4.9}$$

where ρ_1 and ρ_2 are as defined in the previous section.

Proposition 4.1. *Suppose that* ψ^k $(k=0,1,\ldots,N-1)$ *form the solution of* (4.9)*, then there is a positive constant* C_2 *such that* $\left|\psi^{k+1}\right|\leqslant C_2(k+1)\Delta t\left|\phi^{1/2}\right|$.

Proof. Since $E^0=0$ and form (4.4), we obtain

$$\psi^0=\psi^0(l_1,l_2)=0. \tag{4.10}$$

From (4.5) and (4.7), there exists a positive constant C_2 such that

$$\left|\phi^k\right|\leqslant C_2\left|\phi^{1/2}\right|,\quad k=0,1,\ldots,N. \tag{4.11}$$

Next, we complete the proof by using mathematical induction. For $k=0$ and from (4.9) and (4.10), we have

$$\psi^1=\frac{1}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\Delta t\phi^{1/2}.$$

Since $\rho_1,\ \rho_2\geqslant 0$ and from (4.11), we obtain

$$\left|\psi^1\right|\leqslant\Delta t\left|\phi^{1/2}\right|\leqslant C_2\Delta t\left|\phi^{1/2}\right|.$$

Now, supposing that $\left|\psi^{s+1}\right|\leqslant C_2(s+1)\Delta t\left|\phi^{1/2}\right|$, $s=0,1,\ldots,k-1$. Then according to equations (4.9)-(4.11), we get

$$\begin{aligned}\left|\psi^{k+1}\right|&\leqslant\left|\frac{1-0.5(r-1)\Delta t-\rho_1-\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\right|\left|\psi^k\right|+\left|\frac{1}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\right|\left|\Delta t\phi^{k+1/2}\right|\\&\leqslant\left|\frac{1-0.5(r-1)\Delta t-\rho_1-\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\right|C_2k\Delta t\left|\phi^{1/2}\right|+\frac{1}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}C_2\Delta t\left|\phi^{1/2}\right|.\end{aligned}$$

If $1-0.5(r-1)\Delta t-\rho_1-\rho_2\geqslant 0$ and since $\rho_1,\ \rho_2\geqslant 0$, then

$$\left|\psi^{k+1}\right|\leqslant\left[\frac{1-0.5(r-1)\Delta t-\rho_1-\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}k+\frac{1}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\right]C_2\Delta t\left|\phi^{1/2}\right|\leqslant C_2(k+1)\Delta t\left|\phi^{1/2}\right|.$$

If $1-0.5(r-1)\Delta t-\rho_1-\rho_2<0$ and since $\rho_1,\ \rho_2\geqslant 0$, then

$$\left|\psi^{k+1}\right|\leqslant\left[\frac{-1+0.5(r-1)\Delta t+\rho_1+\rho_2}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}k+\frac{1}{1+0.5(r-1)\Delta t+\rho_1+\rho_2}\right]C_2\Delta t\left|\phi^{1/2}\right|\leqslant C_2(k+1)\Delta t\left|\phi^{1/2}\right|.$$

□

Theorem 4.2. *The proposed difference scheme* (2.6) *is* l_2 *convergent with convergence order of* $O((\Delta t)^2+(\Delta x)^2+(\Delta y)^2)$.

Proof. Applying Proposition 4.1 along with Eqs. (4.6) and (4.7), yields

$$\|E^k\|_2^2=\sum_{l_1=-\infty}^{\infty}\sum_{l_2=-\infty}^{\infty}\left|\psi^k(l_1,l_2)\right|^2\leqslant\sum_{l_1=-\infty}^{\infty}\sum_{l_2=-\infty}^{\infty}C_2^2(k+1)^2(\Delta t)^2\left|\phi^{1/2}(l_1,l_2)\right|^2=C_2^2(k+1)^2(\Delta t)^2\|R^{1/2}\|_2^2.$$

Utilizing the inequality (4.1), there exists a positive constant C_1 such that

$$\|E^k\|_2\leqslant C_2(k+1)\Delta t\|R^{1/2}\|_2\leqslant C_1C_2(k+1)\Delta t((\Delta t)^2+(\Delta x)^2+(\Delta y)^2)\leqslant C((\Delta t)^2+(\Delta x)^2+(\Delta y)^2),$$

where $C=C_1C_2T$ as $(k+1)\Delta t\leqslant T$. This completes the proof. □

5. Review of existing standard finite difference scheme

There are various finite difference methods such as explicit and implicit schemes that can be used to solve problem (1.1) with a particular approximation formula for the Caputo fractional derivative. Here, we recall a standard Crank-Nicolson finite difference scheme proposed by Balasim and Ali [2] for solving problem (1.1) to compare it with the proposed hybrid method in the previous section. Utilizing the discretization formula in [15] to approximate the time fractional derivative in Eq. (1.1), and replacing the partial space derivatives by central difference approximations about the point $(x_i,y_j,t_{k+1/2})$, the standard Crank-Nicolson difference scheme for solving (1.1) yields immediately as follows [2]:

$$\begin{aligned}u_{i,j}^{k+1}=\frac{1}{1+r_1+r_2}\Bigg[&\frac{r_1}{2}\left(u_{i+1,j}^{k+1}+u_{i-1,j}^{k+1}+u_{i+1,j}^{k}+u_{i-1,j}^{k}\right)\\&+\frac{r_2}{2}\left(u_{i,j+1}^{k+1}+u_{i,j-1}^{k+1}+u_{i,j+1}^{k}+u_{i,j-1}^{k}\right)+\left(1-2^{1-\alpha}w_1-r_1-r_2\right)u_{i,j}^{k}\\&+2^{1-\alpha}\sum_{s=1}^{k-1}[w_{k-s}-w_{k-s+1}]u_{i,j}^{s}+2^{1-\alpha}w_ku_{i,j}^{0}+m_0f_{i,j}^{k+1/2}\Bigg],\end{aligned}\tag{5.1}$$

in which $m_0=2^{1-\alpha}\Gamma(2-\alpha)(\Delta t)^\alpha$, $r_1=\frac{a_1m_0}{(\Delta x)^2}$, $r_2=\frac{a_2m_0}{(\Delta y)^2}$, $w_s=(s+\frac{1}{2})^{1-\alpha}-(s-\frac{1}{2})^{1-\alpha}$ and the truncation error at the point $(x_i,y_j,t_{k+1/2})$ is of $O((\Delta t)^{2-\alpha}+(\Delta x)^2+(\Delta y)^2)$.

Next, the memory space usage and the computational cost of both standard and hybrid methods shall be discussed. From (5.1), it can be observed that we need to store the solution outcomes at all preceding time steps to obtain the solution at the present time step. This indicates that there are NM_xM_y outcomes must be stored in the memory. Disregard the memory requirement of the source term, initial condition and coefficients, it takes 8 bytes (without loss of generality) to store each outcome, meaning that the standard scheme defined in (5.1) allocates around $8NM_xM_y$ bytes of memory space. By way of illustration, with $M_x=10240$, $M_y=10240$ and $N=1024$, it needs 800 GB. On the contrary, and from (2.6), it can be observed that the solution outcomes need to be saved only at two time steps, k and $k+1$ until the required time level N is attained. So, only $2M_xM_y$ solution values need to be stored in the memory. Thus, the hybrid method requires memory space of only $16M_xM_y$ bytes. Depending on what have been discussed, the hybrid method allocates total memory of $O(M)$, which is far cheaper than the standard method of $O(NM)$, where $M=M_xM_y$.

Besides the additional memory space required using the standard scheme, the considerable computational cost that makes the scheme implementation time consuming is the other challenge. The right hand side of Eq. (5.1) need to be computed in order to attain $u_{i,j}^{k+1}$, where each grid point of time step t_{k+1} requires $11+(k-1)$ additions and $8+(k-1)$ multiplications, presuming that the coefficients are evaluated and saved beforehand. Since each time step include $(M_x-1)(M_y-1)$ grid points, there are $[19+2(k-1)](M_x-1)(M_y-1)$ arithmetic operations to be performed at each time step. Since $k=0\rightarrow N-1$, the whole computational cost using the standard scheme (5.1) is given by

$$\begin{aligned}
&19N(M_x-1)(M_y-1)+2(1+2+\cdots+N-2)(M_x-1)(M_y-1)\\
&\quad=19N(M_x-1)(M_y-1)+2\left(\frac{(N-2)(N-1)}{2}\right)(M_x-1)(M_y-1)\\
&\quad=(N^2+16N+2)(M_x-1)(M_y-1).
\end{aligned}$$

As an alternative solution method, every grid point $u_{i,j}^{k+1}$ evaluation using (2.6) involves 10 additions and 8 multiplications in which the coefficients are computed and stored in advance. Since each time step contain $(M_x-1)(M_y-1)$ grid points, there are $18(M_x-1)(M_y-1)$ arithmetical operations to be implemented at each time step. Thus, the total computational cost utilizing the hybrid method is around $18N(M_x-1)(M_y-1)$, since there are N time levels. Depending on this analysis, the computational work of using the hybrid method is of only $O(N)$ which is significantly cheaper than the standard scheme of $O(N^2)$. Assuming that the execution times of addition and multiplication operations are roughly the same, the proposed hybrid method is expected to exhibit more rapid convergence than the standard method. Table 1 highlights the computational cost and memory requirement when applying each of the standard and hybrid methods.

Table 1: Memory requirement and computational cost of the standard [2] and proposed hybrid methods.

Method	Memory requirement (byte)	Computational cost
Standard	$8NM_xM_y$	$(N^2+16N+2)(M_x-1)(M_y-1)$
Hybrid	$16M_xM_y$	$18N(M_x-1)(M_y-1)$

6. Numerical experiments and results

Two model problems are employed to investigate the performance of the proposed hybrid method developed in Section 3 in comparison to the existing standard finite difference method in [2] for solving the two-dimensional time fractional diffusion equation. To confirm our considerations, both standard and hybrid methods are implemented using Gauss-Seidel method and run on laptop with core 4, 8 GB of RAM with Windows 10 operating system and Mathematica 11.3 software. A tolerance factor of $\epsilon=10^{-5}$ and l_∞ norm are utilized for the convergence criteria. In both examples, the solution domain is discretized for various time steps of 10, 15, 20 and 25 and for space discretization, we assume a fixed uniform mesh size $h=\Delta x=\Delta y=1/50$.

Example 6.1. Consider the following two-dimensional TFDE [35]:

$$ {}_0^C D_t^\alpha u(x,y,t)=\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\left(\frac{2}{\Gamma(3-\alpha)}t^{2-\alpha}+2t^2\right)(\sin(x)\sin(y)),$$

subject to the initial-boundary conditions given by

$$\begin{aligned}
&u(x,y,0)=0,\\
&u(x,0,t)=0,\quad u(x,1,t)=t^2\sin(x)\sin(1),\\
&u(0,y,t)=0,\quad u(1,y,t)=t^2\sin(1)\sin(y),
\end{aligned}$$

in which $0\leqslant x,y\leqslant 1, 0\leqslant t\leqslant 1$ and the exact solution is $u(x,y,t)=t^2\sin(x)\sin(y)$.

Example 6.2. Here we apply both standard and hybrid methods on the two-dimensional TFDE [18]:

$$ {}_0^C D_t^\alpha u(x,y,t) = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \left[\frac{2t^{2-\alpha}}{\Gamma(3-\alpha)} - 2t^2 \right] e^{x+y}, $$

subject to the initial-boundary conditions

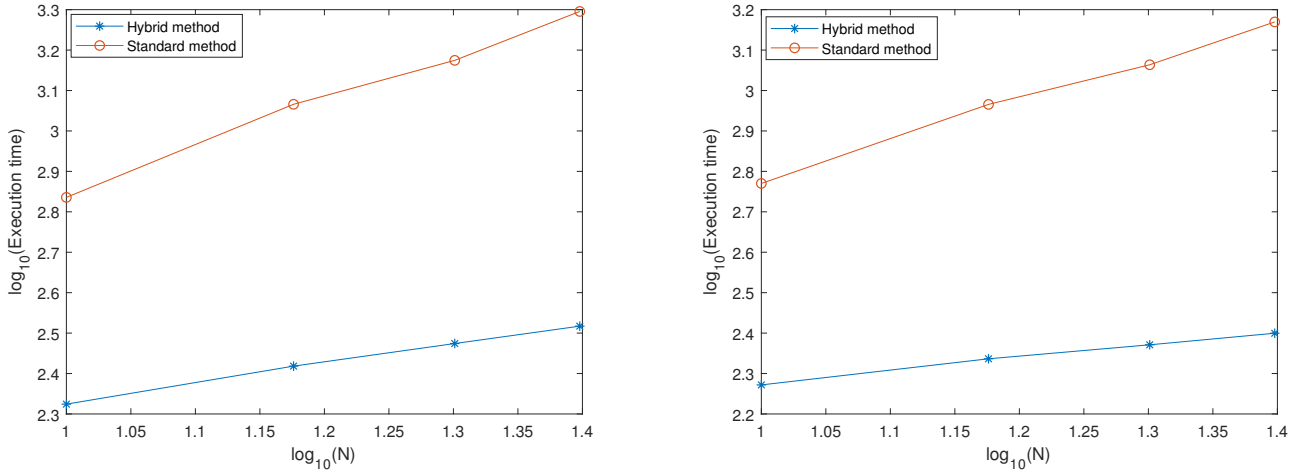
$$ u(x,y,0) = 0, \quad u(x,0,t) = t^2 e^x, \quad u(x,1,t) = t^2 e^{x+1}, u(0,y,t) = t^2 e^y, \quad u(1,y,t) = t^2 e^{1+y}, $$

in which $0 \leqslant x, y \leqslant 1, 0 \leqslant t \leqslant 1$ and the exact solution is $u(x,y,t) = t^2 e^{x+y}$. The accuracy and execution times (in seconds) of the standard and hybrid methods are compared in Tables 2 and 3, where the accuracy is determined by the maximum absolute error (Max) between the exact and numerical solutions. In view of Table 2, it can be observed that the hybrid method performs 3.14-6.00 faster than the standard method without jeopardize the accuracy of numerical solution of Example 6.1. For instance, with $N = 25$ and $\alpha = 0.3$, the execution time of the standard method is 1975 s while the execution time of the hybrid method is 329 s. A similar performance improvement in which the hybrid method runs 2.36-6.38 faster than the standard method for Example 6.2 is shown in Table 3. As an illustration, with $N = 25$ and $\alpha = 0.3$, the execution time of the standard method is 4044 s while the execution time of the hybrid method is 633 s.

On the other hand, the memory space usage of both standard and hybrid methods for various time steps are illustrated in Table 4. We note that the memory requirement of the hybrid method is only about 8-20% of the standard method. Figures 1 and 2 show the execution time for both methods versus N. Clearly, we can observe that the hybrid method requires much less computational time than the standard method, which is in agreement with the theoretical computational complexity analysis in Section 3.

Table 2: Comparison between the standard [2] and proposed hybrid methods at $h = 1/50$ for Example 6.1.

N	Method	Execution time (sec.)	Speedup	Max error
$\alpha = 0.3$				
10	Standard	685	3.24	1.2259E-03
	Hybrid	211		1.4781E-03
15	Standard	1164	4.44	1.2101E-03
	Hybrid	262		1.3833E-03
20	Standard	1495	5.01	1.3293E-03
	Hybrid	298		1.3672E-03
25	Standard	1975	6.00	1.2158E-03
	Hybrid	329		1.3605E-03
$\alpha = 0.7$				
10	Standard	589	3.14	1.4432E-03
	Hybrid	187		1.2438E-03
15	Standard	924	4.25	1.3235E-03
	Hybrid	217		1.1814E-03
20	Standard	1158	4.92	1.2869E-03
	Hybrid	235		1.1592E-03
25	Standard	1478	5.88	1.2726E-03
	Hybrid	251		1.1373E-03

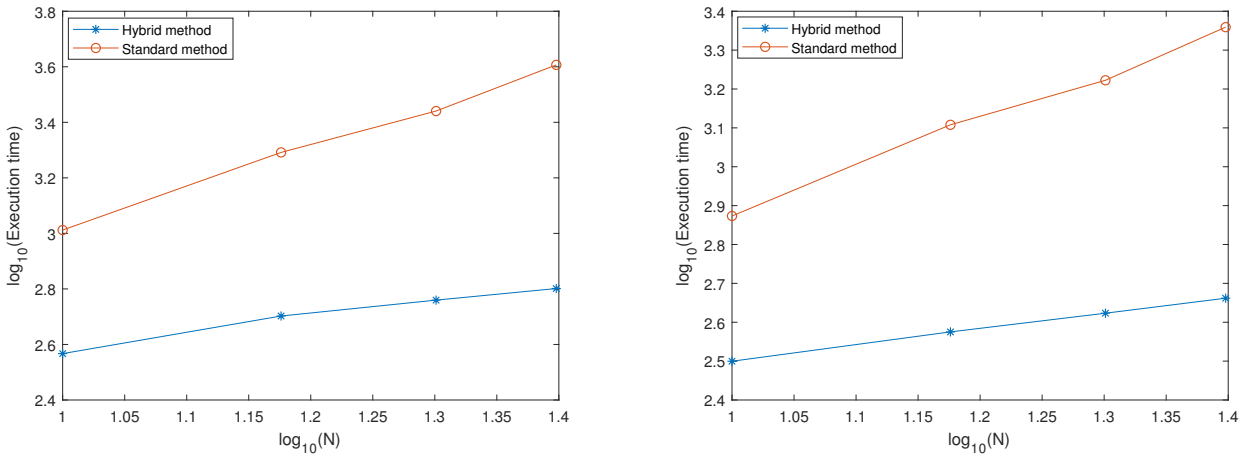


(a) $\alpha = 0.3$ (b) $\alpha = 0.7$

Figure 1: The Execution time (in seconds) against the total number of time steps N for Example 6.1 (log-log in base 10 plot).

Table 3: Comparison between the standard [2] and proposed hybrid methods at $h = 1/50$ for Example 6.2.

N	Method	Execution time (sec.)	Speedup	Max error
$\alpha = 0.3$				
10	Standard	1028	2.78	5.1109E-03
	Hybrid	369		4.1613E-03
15	Standard	1957	3.88	3.8115E-03
	Hybrid	504		2.6622E-03
20	Standard	2758	4.79	2.4789E-03
	Hybrid	575		2.7489E-03
25	Standard	4044	6.38	2.0563E-03
	Hybrid	633		2.7995E-03
$\alpha = 0.7$				
10	Standard	747	2.36	2.4564E-03
	Hybrid	316		1.4119E-03
15	Standard	1282	3.4	1.5626E-03
	Hybrid	376		1.0533E-03
20	Standard	1669	3.97	1.5009E-03
	Hybrid	420		9.3061E-04
25	Standard	2286	4.98	1.4626E-03
	Hybrid	459		8.7441E-04



(a) $\alpha = 0.3$ (b) $\alpha = 0.7$

Figure 2: The Execution time (in seconds) against the total number of time steps N for Example 6.2 (log-log in base 10 plot).

7. Conclusion

In this paper, we derived a hybrid method using Laplace transform and Crank-Nicolson finite difference scheme for the solution of two-dimensional TFDE. It has been shown that the derived method has much less memory complexity of $O(M)$ and computational work of $O(N)$, compared to $O(MN)$ memory complexity and $O(N^2)$ cost when utilizing standard finite difference scheme. The stability and convergence of the proposed method were discussed with complete details. Numerical results revealed that the hybrid method is cheaper than the standard difference scheme in terms of the computational cost, memory requirement as well as the CPU time, making it an effective method in solving the time fractional diffusion problem.

Table 4: Memory space usage of the standard [2] and proposed hybrid methods.

N	Method	Memory usage (byte)
10	Standard	200000
	Hybrid	40000
15	Standard	300000
	Hybrid	40000
20	Standard	400000
	Hybrid	40000
25	Standard	500000
	Hybrid	40000

Acknowledgment

The authors extend their sincere appreciation to the editor and referees for their time and valuable comments. The authors also gratefully acknowledge the financial support from Universiti Sains Malaysia (USM) Research University Grant (203.PMATHS.6711805).

References

[1] A. Atangana, *Fractional discretization: the African's tortoise walk*, Chaos Solitons Fractals, **130** (2020), 24 pages. 1

[2] A. T. Balasim, N. H. M. Ali, *Group iterative methods for the solution of two-dimensional time-fractional diffusion equation*, Advances in Industrial and Applied Mathematics (AIP Conference Proceedings, Johor Bahru, Malaysia), **1750** (2016), 7 pages. 5, 1, 6, 2, 3, 4

[3] P. M. Basha, V. Shanthi, *A robust second order numerical method for a weakly coupled system of singularly perturbed reaction-diffusion problem with discontinuous source term*, Int. J. Comput. Sci. Math., **11** (2020), 63–80. 1

[4] D. A. Benson, S. W. Wheatcraft, M. M. Meerschaert, *Application of a fractional advection-dispersion equation*, Water Resour. Res., **36** (2000), 1403–1412. 1

[5] M. Bishehniasar, S. Salahshour, A. Ahmadian, F. Ismail, D. Baleanu, *An accurate approximate-analytical technique for solving time-fractional partial differential equations*, Complexity, **2017** (2017), 12 pages. 2

[6] A. Carpinteri, F. Mainardi, *Fractals and fractional calculus in continuum mechanics*, Springer, New York, (2014). 1

[7] J.-F. Cheng, Y.-M. Chu, *Solution to the linear fractional differential equation using adomian decomposition method*, Math. Probl. Eng., **2011** (2011), 14 pages. 1

[8] M. Dehghan, J. Manafian, A. Saadatmandi, *Solving nonlinear fractional partial differential equations using the homotopy analysis method*, Numer. Methods Partial Differential Equations, **26** (2010), 448–479. 1

[9] K. Diethelm, *An efficient parallel algorithm for the numerical solution of fractional differential equations*, Fract. Calc. Appl. Anal., **14** (2011), 475–490. 1

[10] R. Du, W. Cao, Z. Z. Sun, *A compact difference scheme for the fractional diffusion-wave equation*, Appl. Math. Model., **34** (2010), 2998–3007. 1

[11] B. Ghanbari, A. Atangana, *An efficient numerical approach for fractional diffusion partial differential equations*, Alex. Eng. J., **59** (2020), 2171–2180. 1

[12] C. Y. Gong, W. M. Bao, G. J. Tang, Y. W. Jiang, J. Liu, *Computational challenge of fractional differential equations and the potential solutions: a survey*, Math. Probl. Eng., **2015** (2015), 13 pages. 1

[13] C. Gong, W. N. Bao, G. J. Tang, B. Yang, J. Liu, *An efficient parallel solution for caputo fractional reaction–diffusion equation*, J. Supercomput., **68** (2014), 1521–1537. 1

[14] S. D. Jiang, J. Zhang, Q. Zhang, Z. M. Zhang, *Fast evaluation of the caputo fractional derivative and its applications to fractional diffusion equations*, Commun. Comput. Phys., **21** (2017), 650–678. 1

[15] I. Karatay, N. Kale, S. R. Bayramoglu, *A new difference scheme for time fractional heat equations based on the crank-nicholson method*, Fract. Calc. Appl. Anal., **16** (2013), 892–910. 5

[16] S. Karimi Vanani, A. Aminataei, *Tau approximate solution of fractional partial differential equations*, Comput. Math. Appl., **62** (2011), 1075–1083. 1

[17] D. Kumar, J. Singh, D. Baleanu, S. Rathore, *Analysis of a fractional model of the ambartsumian equation*, Eur. Phys. J. Plus, **133** (2018), 13 pages. 1

[18] Q. Liu, Y. T. Gu, P. Zhuang, F. Liu, Y. F. Nie, *An implicit RBF meshless approach for time fractional diffusion equations*, Comput. Mech., **48** (2011), 1–12. 6.2

[19] R. L. Magin, *Fractional calculus in bioengineering*, Begell House, Redding, (2006). 1

[20] F. Mainardi, *Fractional calculus and waves in linear viscoelasticity*, Imperial College Press, London, (2010). 1

[21] S. T. Mohyud-Din, T. Akram, M. Abbas, A. I. Ismail, N. H. M. Ali, *A fully implicit finite difference scheme based on extended cubic B-splines for time fractional advection-diffusion equation*, Adv. Difference Equ., **2018** (2018), 17 pages. 1

[22] M. Moshrefi-Torbati, J. K. Hammond, *Physical and geometrical interpretation of fractional operators*, J. Franklin Inst., **335** (1998), 1077–1086. 1

[23] K. M. Owolabi, A. Atangana, *Robustness of fractional difference schemes via the Caputo subdiffusion-reaction equations*, Chaos Solitons Fractals, **111** (2018), 119–127. 1

[24] H.-K. Pang, H.-W. Sun, *Multigrid method for fractional diffusion equations*, J. Comput. Phys., **231** (2012), 693–703. 1

[25] I. Podlubny, *Fractional differential equations*, Academic press, San Diego, (1999). 2

[26] M. Ranjbar, H. Ghafouri, A. Khani, *Application of cubic* B*-spline quasi-interpolation for solving time-fractional partial differential equation*, Comput. Methods Differ. Equ., **2020** (2020), 1–13. 1, 2

[27] J. C. Ren, Z.-Z. Sun, W. Z. Dai, *New approximations for solving the caputo-type fractional partial differential equations*, Appl. Math. Model., **40** (2016), 2625–2636. 2

[28] F. M. Salama, N. H. M Ali, *Fast O(N) Hybrid method for the solution of two dimensional time fractional cable equation*, Compusoft, **8** (2019), 3453–3461. 2

[29] F. M. Salama, N. H. M. Ali, *Computationally efficient hybrid method for the numerical solution of the 2D time fractional advection-diffusion equation*, Int. J. Math. Eng. Manag. Sci., **5** (2020), 432–446.

[30] F. M. Salama, N. H. M. Ali, N. N. Abd Hamid *Efficient hybrid group iterative methods in the solution of two-dimensional time fractional cable equation*, Adv. Difference Equ., **2020** (2020), 20 pages. 2

[31] J. Singh, D. Kumar, D. Baleanu, S. Rathore, *An efficient numerical algorithm for the fractional drinfeld-sokolov-wilson equation*, Appl. Math. Comput., **335** (2018), 12–24. 1

[32] P. J. Torvik, R. L. Bagley, *On the appearance of the fractional derivative in the behavior of real materials*, J. Appl. Mech., **51** (1984), 294–298. 1

[33] S. Vong, P. Lyu, X. Chen, S.-L. Lei, *High order finite difference method for time-space fractional differential equations with caputo and riemann-liouville derivatives*, Numer. Algorithms, **72** (2016), 195–210. 1

[34] Y. F. Xu, Z. M. He, *The short memory principle for solving abel differential equation of fractional order*, Comput. Math. Appl., **62** (2011), 4796–4805. 1

[35] Y.-N. Zhang, Z.-Z. Sun, *Alternating direction implicit schemes for the two-dimensional fractional sub-diffusion equation*, J. Comput. Phys., **230** (2011), 8713–8728. 6.1