International Journal of Mathematical, Engineering and Management Sciences
Vol. 5, No. 3, 432-446, 2020
https://doi.org/10.33889/IJMEMS.2020.5.3.036

Computationally Efficient Hybrid Method for the Numerical Solution of the 2D Time Fractional Advection-Diffusion Equation

Fouad Mohammad Salama
School of Mathematical Sciences,
Universiti Sains Malaysia, Penang, Malaysia.
Corresponding author: fuadmohd321@gmail.com

Norhashidah Hj. Mohd Ali
School of Mathematical Sciences,
Universiti Sains Malaysia, Penang, Malaysia.
E-mail: shidah_ali@usm.my

(Received September 12, 2019; Accepted January 4, 2020)

Abstract
In this paper, a hybrid method based on the Laplace transform and implicit finite difference scheme is applied to obtain the numerical solution of the two-dimensional time fractional advection-diffusion equation (2D-TFADE). Some of the major limitations in computing the numerical solution for fractional differential equations (FDEs) in multi-dimensional space are the huge computational cost and storage requirement, which are $O(N^2)$ cost and $O(MN)$ storage, N and M are the total number of time levels and space grid points, respectively. The proposed method reduced the computational complexity efficiently as it requires only $O(N)$ computational cost and $O(M)$ storage with reasonable accuracy when numerically solving the TFADE. The method's stability and convergence are also investigated. The Results of numerical experiments of the proposed method are obtained and found to compare well with the results of existing standard finite difference scheme.

Keywords- Fractional advection-diffusion equation, Laplace transform, Finite difference scheme, Stability, Convergence.

1. Introduction
Fractional differential equations (FDEs) have received much interest over the last two decades due to its wide applications in many scientific branches. A detailed discussion of fractional calculus and its applications can be found in (Oldham and Spanier, 1974; Samko et al., 1993; Miller and Ross, 1993; and Podlubny, 1999). Here, the TFADE is considered as a useful tool for describing the transport dynamics in complex systems which are governed by anomalous diffusion and non-exponential relaxation patterns (Zhuang et al., 2011).

In recent years, several finite difference methods have been developed to solve fractional advection-diffusion equations (FADEs). Chen et al. (2008) presented implicit and explicit finite difference schemes for solving the fractional reaction-subdiffusion equation. The schemes had been proven to be stable and convergent using a Fourier analysis. Shen et al. (2011) established numerical explicit and implicit difference schemes for the solution of the space-time Riesz-Caputo FADE. They proved that the explicit scheme is conditionally stable, whereas the implicit scheme is unconditionally stable. Later, Gao and Sun (2015) proposed a combined compact difference scheme to solve the TFADE in one dimension. The time fractional derivative was defined in the Caputo sense. The Fourier analysis technique was employed to prove the stability and convergence of the method. Balasim and Ali (2017) derived standard point and explicit group iterative finite difference schemes to compute the numerical solution of the TFADE. The group methods were proved to be stable and convergent using matrix analysis. Mardani et al. (2018) suggested a numerical method based on finite difference scheme and meshless method for solving the one-dimensional TFADE. Vong et al. (2018) presented a second order finite difference schemes to solve FADE. They utilized weighted and shifted Grunwald-Letnikov formulas to discretize the Riemann-Liouville fractional derivatives. Recently, Zhang et al. (2019) suggested implicit and explicit difference schemes to solve the time-space FADE in one dimension. Both schemes have first order accuracy in time and space.

In this paper, we study the following two-dimensional problem of the form

$$ {}_0^C D_t^\alpha u(x,y,t) = a_1 \frac{\partial^2 u}{\partial x^2} + a_2 \frac{\partial^2 u}{\partial y^2} - b_1 \frac{\partial u}{\partial x} - b_2 \frac{\partial u}{\partial y} + f(x,y,t) \tag{1} $$

over a region $\Omega = \{(x,y)|\, 0 \le x \le L, 0 \le y \le L, 0 \le t \le T\}$, with the initial and boundary conditions

$$ u(x,y,0) = p(x,y), \tag{2} $$

$$ \begin{aligned} u(x,0,t) &= p_1(x,t), \quad u(x,L,t) = p_2(x,t),\\ u(0,y,t) &= p_3(y,t), \quad u(L,y,t) = p_4(y,t), \end{aligned} \tag{3} $$

where, a_1, a_2, b_1 and b_2 are positive constants, and ${}_0^C D_t^\alpha$ is the Caputo fractional derivative of order α, $0 < \alpha < 1$.

One of the major challenges in computing the numerical solution for time fractional differential equations (TFDEs) is the nonlocal property of the fractional operator. This means that the computed values of the solution on all preceding time levels need to be stored to obtain the solution values on the present time level. This needs a substantial size of memory, especially for multi-dimensional problems. On the other hand, the computational cost of numerically solving the TFDEs using standard finite difference schemes is of $O(N^2)$ (Gong et al., 2014; Gong et al., 2015; Jiang et al., 2017) which is computationally very expensive and time consuming. For more information about the computational complexity of FDEs, please refer to (Gong et al., 2015). On the contrary, the computational cost of numerically solving time-dependent partial differential equations (PDEs) is of $O(N)$, and the solution outcomes need to be saved on a fixed number of time levels (Gong et al., 2014; Jiang et al., 2017). The main aim of this study is to obtain an economical computational solution for the 2D TFADE (1). The Laplace transform will be employed to approximate the time fractional derivative and reduce the TFADE (1) to its corresponding PDE. The implicit difference scheme will then be applied to solve the resulting PDE, which reduces the computational work and memory usage significantly.

The rest of this paper is arranged as follows. Basic preliminaries and definitions are given in section in section 2. In section 3, we explain the formulation of the hybrid method for the problem (1). Unconditional Stability and convergence of the numerical scheme are proven in section 4. In section 5, we briefly review an existing standard finite difference scheme for numerically solving the TFADE (1). The computational complexities of both standard and hybrid methods are also discussed in this section. Section 6 explores the performance of the hybrid method versus the standard scheme through numerical experiments. Finally, conclusions are illustrated in section 7.

2. Preliminaries

Here we introduce some basic definitions and preliminaries which are utilized further in this paper.
Definition 2.1 (Podlubny, 1999). The Caputo fractional derivative of order α of a function $f(\cdot)$ is given by

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

Definition 2.2 (Zill, 2012). For $t \geq 0$, the Laplace transform of a function $f(\cdot)$ is given by

$$ \mathcal{L}\{f(t)\} = F(s) = \int_0^\infty e^{-st} f(t) dt, $$

where, $F(s)$ is the Laplace transform of $f(t)$.

Definition 2.3 (Podlubny, 1999). The Laplace transform of Caputo fractional derivative is given by

$$ \mathcal{L}\{{}_0^C D_t^\alpha f(t)\} = \int_0^\infty e^{-st} \left({}_0^C D_t^\alpha f(t)\right) dt = s^\alpha F(s) - \sum_{k=0}^{n-1} s^{\alpha-k-1} f^{(k)}(0), n-1 \leq \alpha \leq n, \ \ n \in \mathbb{N} $$

In line with that, we have

$$ \mathcal{L}\{{}_0^C D_t^\alpha f(t)\} = s^\alpha F(s) - s^{\alpha-1} f(0), \qquad 0 < \alpha \leq 1. $$

Corollary 2.1 (Zill, 2012). Let $f'(t)$ be a continuous function on $[0,\infty)$, then the Laplace transform of the function $f'(t)$ is provided as follows

$$ \mathcal{L}\{f'(t)\} = sF(s) - f(0). $$

3. Formulation of the Hybrid Method

The nonlocal property of the time fractional derivative in (1) necessitates the solution outcomes on all the preceding time steps to be saved to obtain the solution on the current time step. To overcome this issue, we convert the TFADE (1) to a partial differential equation (PDE) by approximating the Caputo time fractional derivative using the Laplace transform method and linearization property proposed by Ren et al. (2016) and utilized by Bishehniasar et al. (2017) for an analytic solution as follows

$$ \begin{aligned} \mathcal{L}\{{}_0^C D_t^\alpha u(x,y,t)\} &= s^\alpha U(x,y,s) - s^{\alpha-1} u(x,y,0), \\ &= s^\alpha [U(x,y,s) - s^{-1} u(x,y,0)], \end{aligned} \tag{4} $$

where, $U(x,y,s)$ is the Laplace transform of $u(x,y,t)$.

Since $0 < \alpha < 1$, the term s^α can be linearized as follows

$$ s^\alpha \approx \alpha s + (1-\alpha). \tag{5} $$

Substituting (5) into (4), we obtain

$$ \begin{aligned} \mathcal{L}\{{}_0^C D_t^\alpha u(x,y,t)\} \approx\ & \alpha s [U(x,y,s) - s^{-1} u(x,y,0)] \\ & + (1-\alpha)[U(x,y,s) - s^{-1} u(x,y,0)]. \end{aligned} \tag{6} $$

Applying the inverse Laplace transform, Eq. (6) is reduced to

$$ {}_0^C D_t^\alpha u(x,y,t) \approx \alpha \frac{\partial u(x,y,t)}{\partial t} + (1-\alpha)[u(x,y,t) - u(x,y,0)]. \tag{7} $$

By utilizing (7), the original 2D TFADE (1) is simplified to the following PDE

$$\frac{\partial u}{\partial t} = A_1 \frac{\partial^2 u}{\partial x^2} + A_2 \frac{\partial^2 u}{\partial y^2} - B_1 \frac{\partial u}{\partial x} - B_2 \frac{\partial u}{\partial y} - (r-1)u(x,y,t) + (r-1)p(x,y) + rf(x,y,t), \tag{8}$$

over the region $\Omega = \{(x,y)|\ 0 \le x \le L, 0 \le y \le L, 0 \le t \le T\}$, with the initial and boundary conditions

$$u(x,y,0) = p(x,y), \tag{9}$$

$$\begin{aligned} &u(x,0,t) = p_1(x,t), \quad u(x,L,t) = p_2(x,t), \\ &u(0,y,t) = p_3(y,t), \quad u(L,y,t) = p_4(y,t), \end{aligned} \tag{10}$$

where, $A_1 = \frac{a_1}{\alpha}$, $A_2 = \frac{a_2}{\alpha}$, $B_1 = \frac{b_1}{\alpha}$, $B_2 = \frac{b_2}{\alpha}$ and $r = \frac{1}{\alpha}$.

Now, we obtain an approximate numerical solution for the original TFADE (1) by computing the numerical solution of the simplified PDE (8). Let $h_x = L/M_x$ and $h_y = L/M_y$ be the uniform spatial grid sizes in x and y directions, respectively, and $\tau = T/N$ be the time step size with M_x, M_y and N being positive integers. Define the mesh points $x_i = ih_x$, $y_j = jh_y$, $0 \le i \le M_x$, $0 \le j \le M_y$, and $t_k = k\tau$, $0 \le k \le N$.

Define $u_{i,j}^k$ as the numerical solution at the point (x_i, y_j, t_k). Based on the forward in time and centered in space discretizations about the point (x_i, y_j, t_k), the implicit finite difference scheme for solving (8) is given as

$$\begin{aligned} \frac{u_{i,j}^{k+1} - u_{i,j}^k}{\tau} = {} & A_1 \left(\frac{u_{i+1,j}^{k+1} - 2u_{i,j}^{k+1} + u_{i-1,j}^{k+1}}{h_x^2} \right) + A_2 \left(\frac{u_{i,j+1}^{k+1} - 2u_{i,j}^{k+1} + u_{i,j-1}^{k+1}}{h_y^2} \right) \\ & - B_1 \left(\frac{u_{i+1,j}^{k+1} - u_{i-1,j}^{k+1}}{2h_x} \right) - B_2 \left(\frac{u_{i,j+1}^{k+1} - u_{i,j-1}^{k+1}}{2h_y} \right) - (r-1)u_{i,j}^{k+1} \\ & + (r-1)u_{i,j}^0 + rf_{i,j}^{k+1} + O\left(\tau + h_x^2 + h_y^2\right). \end{aligned} \tag{11}$$

After some rearrangement and neglecting the higher order terms, the following implicit finite difference approximation is obtained

$$\begin{aligned} \left(1 + (r-1)\tau + 2s_x + 2s_y\right)u_{i,j}^{k+1} = {} & \left(s_x - \frac{c_x}{2}\right) u_{i+1,j}^{k+1} + \left(s_x + \frac{c_x}{2}\right) u_{i-1,j}^{k+1} + \left(s_y - \frac{c_y}{2}\right) u_{i,j+1}^{k+1} + \\ & \left(s_y + \frac{c_y}{2}\right) u_{i,j-1}^{k+1} + u_{i,j}^k + (r-1)\tau u_{i,j}^0 + r\tau f_{i,j}^{k+1}, \end{aligned} \tag{12}$$

$$u_{i,j}^0 = p(x_i, y_j), \tag{13}$$

$$\begin{aligned} &u_{i,0}^k = p_1(x_i, t_k), \quad u_{i,M_y} = p_2(x_i, t_k) \\ &u_{0,j}^k = p_3(y_j, t_k), \quad u_{M_x,j}^k = p_4(y_j, t_k) \end{aligned} \tag{14}$$

in which $1 \le i \le M_x - 1$, $1 \le j \le M_y - 1$, $0 \le k \le N - 1$, $s_x = \frac{A_1\tau}{h_x^2}$, $s_y = \frac{A_2\tau}{h_y^2}$, $c_x = \frac{B_1\tau}{h_x}$, $c_y = \frac{B_2\tau}{h_y}$.

4. Stability and Convergence

In this section, we utilize the Fourier method (Ali et al., 2017) to prove the stability and convergence of the numerical scheme (12). Suppose that $u_{i,j}^k$ is the approximate solution of the exact solution $U_{i,j}^k$ of Eq. (12). Then the error is represented as

$$\mathcal{E}_{i,j}^k = U_{i,j}^k - u_{i,j}^k. \tag{15}$$

Next, Eq. (12) can be written by error form as

$$-\left(s_x - \frac{c_x}{2}\right)\mathcal{E}_{i+1,j}^{k+1} - \left(s_x + \frac{c_x}{2}\right)\mathcal{E}_{i-1,j}^{k+1} + \left(1 + (r-1)\tau + 2s_x + 2s_y\right)\mathcal{E}_{i,j}^{k+1}$$
$$-\left(s_y - \frac{c_y}{2}\right)\mathcal{E}_{i,j+1}^{k+1} - \left(s_y + \frac{c_y}{2}\right)\mathcal{E}_{i,j-1}^{k+1} = \mathcal{E}_{i,j}^k + (r-1)\tau\mathcal{E}_{i,j}^0. \tag{16}$$

For $k = 0,1,\dots,N-1$, define the following grid function

$$\mathcal{E}^k(x,y) = \begin{cases} \mathcal{E}_{i,j}^k, & x_{i-\frac{\Delta x}{2}} < x < x_{i+\frac{\Delta x}{2}}, y_{j-\frac{\Delta y}{2}} < y < y_{j+\frac{\Delta y}{2}}, \\ 0, & 0 \le x \le \frac{\Delta x}{2} \ or \ L - \frac{\Delta x}{2} \le x \le L, \\ 0, & 0 \le y \le \frac{\Delta y}{2} \ or \ L - \frac{\Delta y}{2} \le y \le L, \end{cases} \tag{17}$$

where, $\mathcal{E}^k(x,y)$ can be expanded in Fourier series as

$$\mathcal{E}^k(x,y) = \sum_{l_1,l_2=-\infty}^{\infty} \phi^k(l_1,l_2) e^{2I\pi\left(\frac{l_1 x}{L} + \frac{l_2 y}{L}\right)}, \tag{18}$$

in which $I = \sqrt{-1}$, and

$$\phi^k(l_1,l_2) = \frac{1}{L}\int_0^L \int_0^L \mathcal{E}^k(x,y)\, e^{-2I\pi(l_1 x/L + l_2 y/L)} dxdy. \tag{19}$$

From the Parseval equality and l^2 norm, we have

$$\left\|\mathcal{E}^k\right\|_2^2 = \sum_{i=1}^{M_x-1}\sum_{j=1}^{M_y-1} h_x h_y \left|\mathcal{E}_{i,j}^k\right|^2 = \sum_{l_1,l_2=-\infty}^{\infty} \left|\phi^k(l_1,l_2)\right|^2. \tag{20}$$

Based on the analysis above, suppose that

$$\mathcal{E}_{i,j}^k = \phi^k e^{I\beta i h_x} e^{I\gamma j h_y}, \tag{21}$$

where, $\beta = 2\pi l_1/L, \gamma = 2\pi l_2/L$.

Substituting (21) in (16), yields

$$\phi^{k+1} = \frac{1}{1+(r-1)\tau+\rho_1+I\rho_2}\phi^k + \frac{(r-1)\tau}{1+(r-1)\tau+\rho_1+I\rho_2}\phi^0, \tag{22}$$

where,

$$\rho_1 = 4s_x \sin^2\left(\frac{\beta h_x}{2}\right) + 4s_y \sin^2\left(\frac{\gamma h_y}{2}\right), \tag{23}$$

$$\rho_2 = c_x \sin(\beta h_x) + c_y \sin(\gamma h_y). \tag{24}$$

Proposition 1. Suppose ϕ^{k+1} $(k = 0,1,\dots,N-1)$ satisfy (22), then

$$\left|\phi^{k+1}\right| \le |\phi^0|, \quad k = 0,1,\dots,N-1. \tag{25}$$

Proof. Using mathematical induction, take $k = 0$ in (22)

$$|\phi^1| = \left|\frac{1+(r-1)\tau}{1+(r-1)\tau+\rho_1+I\rho_2}\right| |\phi^0| \le |\phi^0|$$

Now, assume that

$$|\phi^s| \le |\phi^0|, \quad s = 1,2,\dots,k \tag{26}$$

From (22) and (26), we obtain

$$\begin{aligned}
|\phi^{k+1}| &\le \left|\frac{1}{1+(r-1)\tau+\rho_1+I\rho_2}\right| |\phi^k| + \left|\frac{(r-1)\tau}{1+(r-1)\tau+\rho_1+I\rho_2}\right| |\phi^0| \\
&\le \frac{1}{|1+(r-1)\tau+\rho_1+I\rho_2|}|\phi^0| + \frac{(r-1)\tau}{|1+(r-1)\tau+\rho_1+I\rho_2|}|\phi^0| \\
&= \frac{1+(r-1)\tau}{|1+(r-1)\tau+\rho_1+I\rho_2|}|\phi^0|,
\end{aligned}$$

and as $1+(r-1)\tau \le |1+(r-1)\tau+\rho_1+I\rho_2|$, then
$|\phi^{k+1}| \le |\phi^0|$.

According to (20) and (25), we obtain

$$\|\mathcal{E}^{k+1}\|_2^2 = \sum_{l_1,l_2=-\infty}^{\infty} |\phi^k(l_1,l_2)|^2 \le \sum_{l_1,l_2=-\infty}^{\infty} |\phi^0(l_1,l_2)|^2 = \|\mathcal{E}^0\|_2^2.$$

Therefore, we have

$$\|\mathcal{E}^{k+1}\|_2 \le \|\mathcal{E}^0\|_2, \quad k = 0,1,\dots,N-1,$$

which means that the implicit scheme is unconditionally stable.

Next, we utilize an analogous approach as that used to discuss the stability to prove the convergence of the implicit scheme.

Let the truncation error at the point (x_i, y_j, t_k) be denoted by $R_{i,j}^k$. From (11) there is a positive constant C_1 such that for all i, j and k, we have

$$|R_{i,j}^k| \le C_1(\tau + h_x^2 + h_y^2). \tag{27}$$

Suppose that $u_{i,j}^k$ is the approximate solution of the exact solution $U_{i,j}^k$. The exact solution at time level $k+1$ can be written as

$$\begin{aligned}
&-\left(s_x - \frac{c_x}{2}\right)U_{i+1,j}^{k+1} - \left(s_x + \frac{c_x}{2}\right)U_{i-1,j}^{k+1} + \left(1+(r-1)\tau+2s_x+2s_y\right)U_{i,j}^{k+1} \\
&-\left(s_y - \frac{c_y}{2}\right)U_{i,j+1}^{k+1} - \left(s_y + \frac{c_y}{2}\right)U_{i,j-1}^{k+1} \\
&= U_{i,j}^k + (r-1)\tau U_{i,j}^0 + r\tau f_{i,j}^{k+1} + \tau R_{i,j}^{k+1}.
\end{aligned} \tag{28}$$

The error can be defined as

$$E_{i,j}^k = U_{i,j}^k - u_{i,j}^k. \tag{29}$$

By subtracting (12) from (28), the following error equation is obtained

$$-\left(s_x-\frac{c_x}{2}\right)E_{i+1,j}^{k+1}-\left(s_x+\frac{c_x}{2}\right)E_{i-1,j}^{k+1}+\left(1+(r-1)\tau+2s_x+2s_y\right)E_{i,j}^{k+1}$$
$$-\left(s_y-\frac{c_y}{2}\right)E_{i,j+1}^{k+1}-\left(s_y+\frac{c_y}{2}\right)E_{i,j-1}^{k+1}$$
$$=E_{i,j}^{k}+(r-1)\tau E_{i,j}^{0}+\tau R_{i,j}^{k+1}, \tag{30}$$

where,

$$\begin{aligned}&E_{i,j}^{0}=0, \quad 0\le i\le M_x, 0\le j\le M_y,\\ &E_{0,j}^{k}=E_{M_x,j}^{k}=0, \quad 0\le j\le M_y, 0\le k\le N,\\ &E_{i,0}^{k}=E_{i,M_y}^{k}=0, \quad 0\le i\le M_x, 0\le k\le N.\end{aligned} \tag{31}$$

Then, we define the following discrete functions for $k=0,1,\dots,N-1$

$$E^k(x,y)=\begin{cases}E_{i,j}^{k}, & x_{i-\frac{h_x}{2}}<x<x_{i+\frac{h_x}{2}}, y_{j-\frac{h_y}{2}}<y<y_{j+\frac{h_y}{2}},\\ 0, & 0\le x\le\frac{h_x}{2} \ or\ L-\frac{h_x}{2}\le x\le L,\\ 0, & 0\le y\le\frac{h_y}{2} \ or\ L-\frac{h_y}{2}\le y\le L,\end{cases} \tag{32}$$

and

$$R^k(x,y)=\begin{cases}R_{i,j}^{k}, & x_{i-\frac{h_x}{2}}<x<x_{i+\frac{h_x}{2}}, y_{j-\frac{h_y}{2}}<y<y_{j+\frac{h_y}{2}},\\ 0, & 0\le x\le\frac{h_x}{2} \ or\ L-\frac{h_x}{2}\le x\le L,\\ 0, & 0\le y\le\frac{h_y}{2} \ or\ L-\frac{h_y}{2}\le y\le L,\end{cases} \tag{33}$$

$i=1,2,\dots,M_x-1, j=1,2,\dots,M_y-1, k=0,1,\dots,N-1.$

$E^k(x,y)$ and $R^k(x,y)$ can be expanded in Fourier series as

$$E^k(x,y)=\sum_{l_1,l_2=-\infty}^{\infty}\lambda^k(q_1,q_2)e^{2I(l_1x/L+l_2y/L)}, \qquad k=0,1,\dots,N-1 \tag{34}$$

$$R^k(x,y)=\sum_{l_1,l_2=-\infty}^{\infty}\psi^k(q_1,q_2)e^{2I(l_1x/L+l_2y/L)}, \qquad k=0,1,\dots,N-1 \tag{35}$$

in which

$$\lambda^k(l_1,l_2)=\frac{1}{L}\int_0^L\int_0^L E^k(x,y)\,e^{-2I(l_1x/L+l_2y/L)}dxdy, \tag{36}$$

$$\psi^k(l_1,l_2)=\frac{1}{L}\int_0^L\int_0^L R^k(x,y)\,e^{-2I(l_1x/L+l_2y/L)}dxdy. \tag{37}$$

By utilizing the l^2 norm and the Parseval equality, we have

$$\left\|E^k\right\|_{l^2}^2=\sum_{l_1,l_2=-\infty}^{\infty}\left|\lambda^k(l_1,l_2)\right|^2, \tag{38}$$

$$\left\|R^k\right\|_{l^2}^2=\sum_{l_1,l_2=-\infty}^{\infty}\left|\psi^k(l_1,l_2)\right|^2. \tag{39}$$

Based on the analysis above, we assume that the solution of Eq. (30) has the following forms

$$E_{i,j}^{k}=\rho^k e^{I\beta i h_x}e^{I\gamma j h_y}, \ R_{i,j}^{k}=\phi^k e^{I\beta i h_x}e^{I\gamma j h_y}. \tag{40}$$

Substituting (40) into (30), yields

$$\lambda^{k+1} = \frac{1}{1+(r-1)\tau+\rho_1+I\rho_2}\lambda^k + \frac{(r-1)\tau\lambda^0+\tau\psi^{k+1}}{1+(r-1)\tau+\rho_1+I\rho_2}, \tag{41}$$

where. ρ_1 and ρ_2 are mentioned before.

Proposition 2. Suppose λ^{k+1} $(k = 0,1,\dots,N-1)$ form the solution of (41), then there is a positive constant C so that

$$|\lambda^{k+1}| \le C_2(k+1)\tau|\psi^1|. \tag{42}$$

Proof. Since $E^0 = 0$, and using (35) we have

$$\lambda^0 = \lambda^0(l_1,l_2) = 0. \tag{43}$$

According to (37) and (39), there is a positive constant C_2 such that

$$|\psi^k| \le C_2|\psi^1|. \tag{44}$$

We use mathematical induction to prove (42). For $k = 0$ in (41) and using (43), we get

$$\lambda^1 = \frac{1}{1+(r-1)\tau+\rho_1+I\rho_2}\tau\psi^1.$$

As $1 < |1+(r-1)\tau+\rho_1+I\rho_2|$, and from (44), we have

$$|\lambda^1| \le \tau|\psi^1| \le C_2\tau|\psi^1|.$$

Now, assume that

$$|\lambda^s| \le C_2 s\tau|\psi^1|, \qquad s = 1,2,\dots,k \tag{45}$$

From (41), (44) and (45) we have

$$\begin{aligned}|\lambda^{k+1}| &\le \left|\frac{1}{1+(r-1)\tau+\rho_1+I\rho_2}\right||\lambda^k| + \left|\frac{1}{1+(r-1)\tau+\rho_1+I\rho_2}\right|\tau|\psi^{k+1}| \\ &\le \frac{1}{|1+(r-1)\tau+\rho_1+I\rho_2|}C_2k\tau|\psi^1| + \frac{1}{|1+(r-1)\tau+\rho_1+I\rho_2|}C_2\tau|\psi^1| \\ &= \frac{1}{|1+(r-1)\tau+\rho_1+I\rho_2|}C_2(k+1)\tau|\psi^1|.\end{aligned}$$

As $1 < |1+(r-1)\tau+\rho_1+I\rho_2|$, then for all values

$$|\lambda^{k+1}| \le C_2(k+1)\tau|\psi^1|.$$

According to (27) and (39), we obtain

$$\|R^k\|_2 \le \sqrt{M_xM_yh_xh_y}C_1(\tau+h_x^2+h_y^2) = LC_1(\tau+h_x^2+h_y^2). \tag{46}$$

For $k = 0,1,\dots,N-1$, and in view of proposition 2, (38), (39) and (46)

$$\|E^{k+1}\|_2 \le C_2(k+1)\tau\|R^1\|_2 \le LC_1C_2(k+1)\tau(\tau+h_x^2+h_y^2),$$

observing that $(k+1)\tau \le T, k = 0,1,\dots,N-1$, we have

$$\|E^{k+1}\|_2 \le C(\tau+h_x^2+h_y^2),$$

in which $C = C_1 C_2 LT$. This means that the implicit scheme (12) is convergent with order of convergence $O(\tau + h_x^2 + h_y^2)$.

5. Review of a Standard Scheme

In this section, an existing standard implicit finite difference scheme for solving (1) is introduced for comparison. As stated in section 1, solving TFDEs using standard finite difference methods is time-memory consuming, especially for multi-dimensional problems. A significant reason for this is the considerable computational complexity of those methods. Consequently, the computational complexity for both standard and present methods would be analyzed in this section. The computational complexity is determined by the total number of arithmetic operations to be performed accompanied by the memory requirement for each method.

Balasim (2017) derived a standard implicit finite difference scheme for solving problem (1). The author utilized the following discretization formula proposed by Lin and Xu (2007) to approximate the time fractional derivative

$$ {}_0^C D_t^\alpha u\big(x_i, y_j, t_{k+1}\big) = \frac{\tau^{-\alpha}}{\Gamma(2-\alpha)} \sum_{s=0}^{k} b_s\big(u_{i,j}^{k-s+1} - u_{i,j}^{k-s}\big) + O(\tau^{2-\alpha}), $$

where, $b_s = (s+1)^{1-\alpha} - s^{1-\alpha}$.

So, the fully discrete scheme for solving the TFADE (1) can be given as

$$ \begin{aligned} \frac{\tau^{-\alpha}}{\Gamma(2-\alpha)} \sum_{s=0}^{k} & b_s\big[u_{i,j}^{k-s+1} - u_{i,j}^{k-s}\big] \\ &= a_1\left(\frac{u_{i+1,j}^{k+1} - 2u_{i,j}^{k+1} + u_{i-1,j}^{k+1}}{h_x^2}\right) + a_2\left(\frac{u_{i,j+1}^{k+1} - 2u_{i,j}^{k+1} + u_{i,j-1}^{k+1}}{h_y^2}\right) \\ &\quad -b_1\left(\frac{u_{i+1,j}^{k+1} - u_{i-1,j}^{k+1}}{2h_x}\right) - b_2\left(\frac{u_{i,j+1}^{k+1} - u_{i,j-1}^{k+1}}{2h_y}\right) + f_{i,j}^{k+1} + O\big(\tau^{2-\alpha} + h_x^2 + h_y^2\big). \end{aligned} $$

For simplification, the standard scheme above can be simplified as (Balasim, 2017)

$$ \begin{aligned} (1 + 2s_1 + 2s_2)u_{i,j}^{k+1} = {} & \left(s_1 - \frac{c_1}{2}\right)u_{i+1,j}^{k+1} + \left(s_1 + \frac{c_1}{2}\right)u_{i-1,j}^{k+1} + \left(s_2 - \frac{c_2}{2}\right)u_{i,j+1}^{k+1} \\ & + \left(s_2 + \frac{c_2}{2}\right)u_{i,j-1}^{k+1} + u_{i,j}^{k} + \sum_{s=1}^{k} b_s\big[u_{i,j}^{k-s} - u_{i,j}^{k-s+1}\big] \\ & + v f_{i,j}^{k+1}, \end{aligned} \tag{47} $$

where. $v = \tau^\alpha \Gamma(2-\alpha)$, $s_1 = \frac{a_1 v}{h_x^2}$, $s_2 = \frac{a_2 v}{h_y^2}$, $c_1 = \frac{b_1 v}{h_x}$, $c_2 = \frac{b_2 v}{h_y}$. The standard implicit scheme was also proven to be unconditionally stable and convergent in the same study using Fourier analysis method.

As can be seen from Eq. (47), computing the numerical solution at each unknown time level necessitates the storage of the solution values at all previous time levels. This means that there are NM_xM_y solution values to be saved in the memory. With 8 bytes required to store each value, and disregard the memory usage of the coefficients, initial condition and source term, the standard method (47) requires about $8NM_xM_y$ bytes memory space. For instance, it requires 800 GB with

$M_x = 10240$, $M_y = 10240$ and $N = 1024$. According to Eq. (12), conversely, one can see that the solution requires to be stored only at two-time levels, k and $k+1$. This indicates that we have only $2M_xM_y$ solution outcomes to be stored in the memory. Therefore, the hybrid method needs only $16M_xM_y$ bytes of memory space. So, the hybrid method's memory requirement is of $O(M)$ which is much smaller than the standard method of $O(NM)$, where $M = M_xM_y$.

On the other hand, the expensive computational cost for implementing standard finite difference schemes is another issue that needs to be addressed. In order to obtain $u_{i,j}^{k+1}$, the computation of the right-hand side of (47) must be performed. Assuming that the coefficients $(1+2s_1+2s_2)$, $(s_1-c_1/2)$, $(s_1+c_1/2)$, $(s_2-c_2/2)$, $(s_2+c_2/2)$ and b_s are computed and stored in advance, it requires $(6+k)$ additions and $(6+k)$ multiplications to obtain each grid point $u_{i,j}^{k+1}$, $k = 0,1,2,\dots,N-1$. Each time level involves $(M_x-1)(M_y-1)$ grid points. So, for each time level k, $(12+2(k))(M_x-1)(M_y-1)$ arithmetic operations are required to be implemented. As $k = 0 \to N-1$, and based on (47), the total computational cost for the standard method is

$$\begin{aligned}
&12N(M_x-1)(M_y-1) + 2(1+2+\cdots+N-1)(M_x-1)(M_y-1)\\
&\quad = 12N(M_x-1)(M_y-1) + 2\left(\frac{N(N-1)}{2}\right)(M_x-1)(M_y-1)\\
&\quad = (N^2+11N)(M_x-1)(M_y-1).
\end{aligned}$$

On the contrary, the (ith,jth) grid point of time level t_{k+1} in (12) needs 6 additions and 9 multiplications ignoring the computation of the coefficients. With $(M_x-1)(M_y-1)$ spatial points at each time level, there are $15(M_x-1)(M_y-1)$ arithmetic operations. Since there are N time levels, the whole computational cost of the present method is about $15N(M_x-1)(M_y-1)$. Thus, the hybrid method's computational cost is of $O(N)$ which is much less expensive than standard method of $O(N^2)$. With the assumption that the times taken for addition and multiplication operations to be executed are approximately the same, the hybrid method is expected to perform faster than the standard method. The memory requirement and computational cost for the standard and hybrid methods are summarized in Table 1.

Table 1. Memory requirement and computational cost for the standard and hybrid methods

Method	Memory requirement (bytes)	Computational cost
Standard method [1]	$8NM_xM_y$	$(N^2+11N)(M_x-1)(M_y-1)$
Hybrid method	$16M_xM_y$	$15N(M_x-1)(M_y-1)$

6. Numerical Experiments

In this section, computational experiments are provided to illustrate our considerations. Standard scheme (47) and hybrid method developed in section 3 are considered to solve problem (1). The accuracy of these methods is evaluated by maximum absolute error and average absolute error. The hybrid method is shown to be precise and computationally efficient tool for computing the numerical solution of the TFADE (1). The numerical experiments are implemented in Mathematica 11.3 software and it is run on Windows 10 operated by Intel Quad Core Processor with 8 GB of RAM.

The maximum error between the numerical solution and the exact solution of (1) is defined as follows:

$$Max\ Error = \max_{1\le i\le M_x-1,1\le j\le M_y-1}\left|U_{i,j}^N - u_{i,j}^N\right|. \tag{48}$$

Example 1. We solve the following two-dimensional problem

$$\begin{gathered}{}_0^C D_t^{\alpha} u(x,y,t) = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - \frac{\partial u}{\partial x} - \frac{\partial u}{\partial y} + \frac{t^{1-\alpha}(\sin x + \sin y)}{\Gamma(2-\alpha)} \\ +t(\sin x + \cos x + \sin y + \cos y),\end{gathered}$$

with the initial and boundary conditions

$$\begin{aligned}&u(x,y,0) = 0, \\ &u(x,0,t) = t\sin x, \qquad u(x,1,t) = t(\sin x + \sin 1), \\ &u(0,y,t) = t\sin y, \qquad u(1,y,t) = t(\sin y + \sin 1),\end{aligned}$$

and the exact solution is $u(x,y,t) = t(\sin x + \sin y)$.

We use Laplace transform to approximate Caputo time fractional derivative. Hence, the above problem is converted to the following system:

$$\begin{gathered}\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - \frac{\partial u}{\partial x} - \frac{\partial u}{\partial y} - (r-1)u(x,y,t) \\ + r\left(\frac{t^{1-\alpha}(\sin x + \sin y)}{\Gamma(2-\alpha)} + t(\sin x + \cos x + \sin y + \cos y)\right),\end{gathered}$$

$$\begin{aligned}&u(x,y,0) = 0, \\ &u(x,0,t) = t\sin x, \qquad u(x,1,t) = t(\sin x + \sin 1), \\ &u(0,y,t) = t\sin y, \qquad u(1,y,t) = t(\sin y + \sin 1),\end{aligned}$$

For the numerical solution of this problem, the solution domain is discretized for different time steps of 10, 15, 20 and 25 for the time discretization $(0 < t < 1)$, and for space discretization, we assume $h_x = h_y = 1/40$ in both x and y directions. Table 2 shows the computed values of CPU time (in seconds), memory space requirement (in bytes), maximum absolute error (Max) and average absolute error (Ave) in both standard and hybrid methods when $\alpha = 0.15, 0.35$. The numerical results demonstrate that the hybrid method has $(24.64 - 47.79)\%$ less computational time than the standard method, whereas the memory space requirement was $(8 - 20)\%$ less, with almost the same degree of accuracy. Figure 1 shows the computational time consumed by both standard and hybrid methods when $\alpha = 0.15, 0.35$. It can be observed that the hybrid method's results show a significantly less CPU time compared to the standard method at various values of time steps and is in addition to the considerable savings in the memory space usage too. This is in line with the theoretical analysis of computational complexity in the previous section.

Table 2. Comparison between standard and hybrid methods at $h_x = h_y = 1/40$ for Example 1

N	Method	CPU time (sec)	Memory usage (bytes)	Ave	Max
$\alpha = 0.15$					
10	Standard	408	128000	1.2450E-3	2.9435E-3
	Hybrid	161	25600	1.1977E-3	2.4731E-3
15	Standard	672	192000	1.2458E-3	2.9460E-3
	Hybrid	217	25600	1.2019E-3	2.4819E-3
20	Standard	954	256000	1.2469E-3	2.9495E-3
	Hybrid	266	25600	1.2016E-3	2.4810E-3
25	Standard	1250	320000	1.2466E-3	2.9500E-3
	Hybrid	308	25600	1.1996E-3	2.4766E-3
$\alpha = 0.35$					
10	Standard	295	128000	1.2528E-3	2.9618E-3
	Hybrid	141	25600	2.9037E-3	6.0473E-3
15	Standard	489	192000	1.2452E-3	2.9444E-3
	Hybrid	196	25600	2.9036E-3	6.0475E-3
20	Standard	672	256000	1.2461E-3	2.9475E-3
	Hybrid	237	25600	2.9056E-3	6.0521E-3
25	Standard	926	320000	1.2495E-3	2.9566E-3
	Hybrid	262	25600	2.9049E-3	6.0504E-3

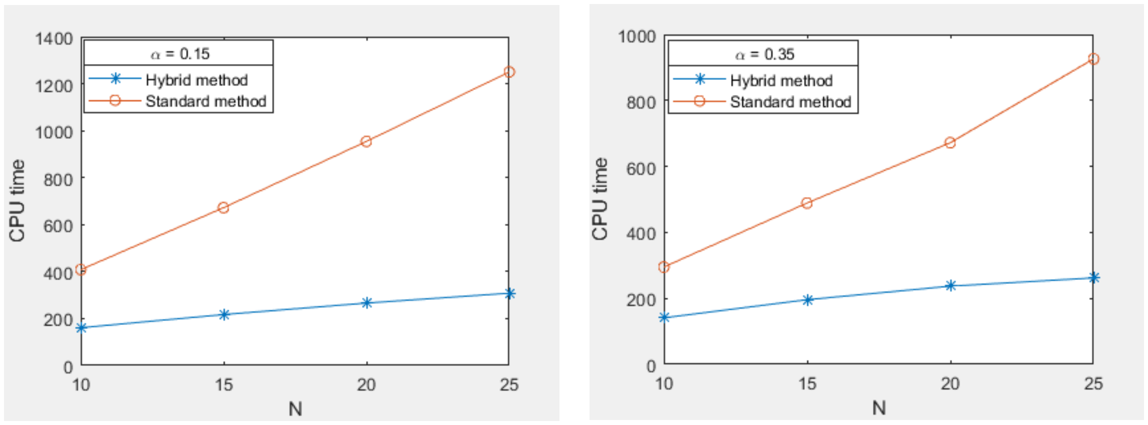

Figure 1. CPU time versus total number of time levels N of Example 1

Example 2. Let us consider the following two-dimensional problem

$$ {}_{0}^{C}D_{t}^{\alpha} u(x,y,t) = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - \frac{\partial u}{\partial x} - \frac{\partial u}{\partial y} + 0.5\Gamma(3+\alpha)t^2 e^{x+y}, $$

with the initial and boundary conditions

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

and the exact solution $u(x,y,t) = t^{2+\alpha}e^{x+y}$.

Again, we utilize Laplace transform to approximate Caputo time fractional derivative. Hence, the above problem is converted to the following system:

$$\frac{\partial u}{\partial t}=\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}-\frac{\partial u}{\partial x}-\frac{\partial u}{\partial y}-(r-1)u(x,y,t)+0.5r\Gamma(3+\alpha)t^2e^{x+y},$$
$$u(x,y,0)=0,$$
$$u(x,0,t)=t^{2+\alpha}e^{x},\qquad u(x,1,t)=t^{2+\alpha}e^{x+1},$$
$$u(0,y,t)=t^{2+\alpha}e^{y},\qquad u(1,y,t)=t^{2+\alpha}e^{y+1}.$$

In the second example, we use various time steps of 12, 16, 20 and 24 for the time discretization $(0<t<1)$, and for space discretization, we assume $h_x=h_y=1/45$ in both x and y directions. Table 3 exhibit the computed values of CPU time, memory space usage, maximum absolute error and average absolute error by both standard and hybrid methods when $\alpha=0.75$, 0.95. The numerical results demonstrate that the hybrid method has $(28.07-45.45)\%$ less computational time than the standard method, whereas the memory space usage is almost $(8.33-16.66)\%$ less, without jeopardizing the accuracy of the numerical solutions. In view of Figure 2, it is noted that the CPU time taken for the hybrid method to be implemented is significantly less compared to the standard method. This verifies that the hybrid method has less computational complexity than the standard method.

Table 3. Comparison between standard and hybrid methods at $h_x=h_y=1/45$ for Example 2

N	Method	CPU time (sec)	Memory usage (bytes)	Ave	Max
$\alpha=0.75$					
12	Standard	550	194400	4.1777E-3	8.7862E-3
	Hybrid	250	32400	3.1003E-3	6.3767E-3
16	Standard	783	259200	2.0939E-3	4.7569E-3
	Hybrid	295	32400	4.5832E-4	1.3858E-3
20	Standard	972	324000	9.0666E-4	2.7076E-3
	Hybrid	339	32400	2.5940E-3	5.7599E-3
24	Standard	1318	388800	4.6009E-4	1.5967E-3
	Hybrid	370	32400	4.0356E-3	8.7668E-3
$\alpha=0.95$					
12	Standard	531	194400	1.5750E-2	3.2758E-2
	Hybrid	235	32400	1.7109E-2	3.5792E-2
16	Standard	733	259200	1.1017E-2	2.2900E-2
	Hybrid	292	32400	1.1852E-2	2.4761E-2
20	Standard	886	324000	8.1787E-3	1.6989E-2
	Hybrid	306	32400	8.6629E-3	1.8067E-2
24	Standard	1009	388800	6.3019E-3	1.3121E-2
	Hybrid	323	32400	6.5240E-3	1.3579E-2

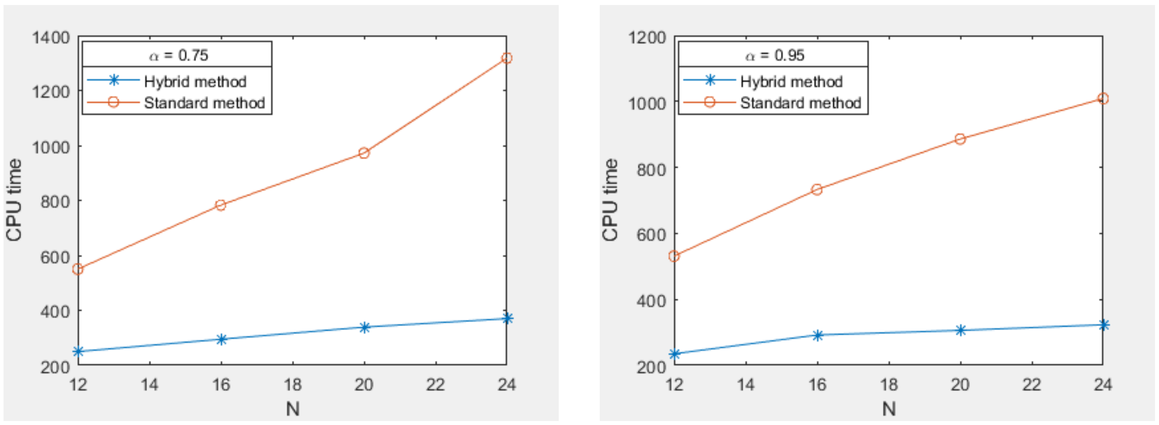

Figure 2. CPU time versus total number of time levels N of Example 2

7. Conclusion

In this paper, the Laplace transform is combined with a fully implicit finite difference scheme for solving the 2D-TFADE. The proposed method has the utility of low computational complexity as it requires only $O(M)$ memory space and $O(N)$ computational cost, where N and M are the total number of time levels and space grid points, respectively. A comparison between a standard implicit scheme and the proposed method has revealed that the proposed method is accurate and superior to the standard method, in terms of the computational time and storage requirement. The unconditional stability and convergence of the numerical scheme are proved using the Fourier series analysis

Conflict of Interest

The authors confirm that there is no conflict of interest to declare for this publication.

Acknowledgements

The authors extend their sincere appreciation to the editor and reviewers for their time and valuable suggestions. The authors also gratefully acknowledge the financial support from Universiti Sains Malaysia Research University Grants (1001/PMATHS/8011016).

References

Ali, U., Abdullah, F.A., & Mohyud-Din, S.T. (2017). Modified implicit fractional difference scheme for 2D modified anomalous fractional sub-diffusion equation. *Advances in Difference Equations*, *2017*(1), 185. doi:10.1186/s13662-017-1192-4.

Balasim, A.T., & Ali, N.H.M. (2017). New group iterative schemes in the numerical solution of the two-dimensional time fractional advection-diffusion equation. *Cogent Mathematics & Statistics*, *4*(1), 1412241.

Balasim, A.T. (2017). *Fractional group iterative methods for two dimensional time-fractional differential equations.* PhD Thesis. Universiti Sains Malaysia, Penang, Malaysia.

Bishehniasar, M., Salahshour, S., Ahmadian, A., Ismail, F., & Baleanu, D. (2017). An accurate approximate-analytical technique for solving time-fractional partial differential equations. *Complexity*, *2017*, 1-12.

Chen, C.M., Liu, F., & Burrage, K. (2008). Finite difference methods and a Fourier analysis for the fractional reaction–subdiffusion equation. *Applied Mathematics and Computation*, *198*(2), 754-769.

Gao, G.H., & Sun, H.W. (2015). Three-point combined compact difference schemes for time-fractional advection–diffusion equations with smooth solutions. *Journal of Computational Physics*, *298*, 520-538.

Gong, C., Bao, W., Tang, G., Jiang, Y. & Liu, J. (2014). A parallel algorithm for the two-dimensional time fractional diffusion equation with implicit difference method. *The Scientific World Journal*, *2014*, 1-8.

Gong, C., Bao, W., Tang, G., Jiang, Y., & Liu, J. (2015). Computational challenge of fractional differential equations and the potential solutions: a survey. *Mathematical Problems in Engineering*, *2015*, 1-13.

Jiang, S., Zhang, J., Zhang, Q., & Zhang, Z. (2017). Fast evaluation of the Caputo fractional derivative and its applications to fractional diffusion equations. *Communications in Computational Physics*, *21*(3), 650-678.

Lin, Y., & Xu, C. (2007). Finite difference/spectral approximations for the time-fractional diffusion equation. *Journal of Computational Physics*, *225*(2), 1533-1552.

Mardani, A., Hooshmandasl, M.R., Heydari, M.H., & Cattani, C. (2018). A meshless method for solving the time fractional advection–diffusion equation with variable coefficients. *Computers & Mathematics with Applications*, *75*(1), 122-133.

Miller, K.S., & Ross, B. (1993). *An introduction to the fractional calculus and fractional differential equations*. Wiley, New York.

Oldham, K.B., & Spanier, J. (1974). *The Fractional Calculus*. Academic Press, New York.

Podlubny, I. (1999). *Fractional differential equations*. Academic Press, New York.

Ren, J., Sun, Z.Z., & Dai, W. (2016). New approximations for solving the Caputo-type fractional partial differential equations. *Applied Mathematical Modelling*, *40*(4), 2625-2636.

Samko, S.G., Kilbas, A.A., & Marichey, O.I. (1993). *Fractional integrals and derivatives: theory and applications*. Gordon & Breach, Yverdon.

Shen, S., Liu, F., & Anh, V. (2011). Numerical approximations and solution techniques for the space-time Riesz–Caputo fractional advection-diffusion equation. *Numerical Algorithms*, *56*(3), 383-403.

Vong, S., Shi, C., & Lyu, P. (2018). A study on a second order finite difference scheme for fractional advection–diffusion equations. *Numerical Methods for Partial Differential Equations*, *35*(2), 493-508.

Zhang, F., Gao, X., & Xie, Z. (2019). Difference numerical solutions for time-space fractional advection diffusion equation. *Boundary Value Problems*, *2019*(1), 14.

Zhuang, P., Gu, Y., Liu, F., Turner, I., & Yarlagadda, P.K.D.V. (2011). Time-dependent fractional advection–diffusion equations by an implicit MLS meshless method. *International Journal for Numerical Methods in Engineering*, *88*(13), 1346-1362.

Zill, D.G. (2012). *A first course in differential equations with modeling applications*. Brooks Cole, Boston.