Thai Journal of Mathematics : 103-131
Special Issue : Annual Meeting in Mathematics 2019

http://thaijmath.in.cmu.ac.th
Online ISSN 1686-0209

Finite Integration Method via Chebyshev Polynomial Expansion for Solving 2-D Linear Time-Dependent and Linear Space-Fractional Differential Equations

Ratinan Boonklurb[†,1], Ampol Duangpan[†] and Arnont Saengsiritongchai[‡]

[†]Department of Mathematics and Computer Science, Faculty of Science,
Chulalongkorn University, Bangkok 10330, Thailand
e-mail : ratinan.b@chula.ac.th (R. Boonklurb)
ty_math@hotmail.com (A. Duangpan)
[‡]Airport Slot Allocation Group,
The Civil Aviation Authority of Thailand, Bangkok 10210, Thailand
e-mail : arnont.plug.s@gmail.com (A. Saengsiritongchai)

Abstract : In this paper, we modify the finite integration method (FIM) using Chebyshev polynomial, to construct a numerical algorithm for finding approximate solutions of two-dimensional linear time-dependent differential equations. Comparing with the traditional FIMs using trapezoidal and Simpson's rules, the numerical results demonstrate that our proposed algorithm give a better accuracy even for a large time step. In addition, we also devise a numerical algorithm based on the idea of FIM using Chebyshev polynomial to find approximate solutions of a linear space-fractional differential equations under Riemann-Liouville definition of fractional order derivative. Several numerical examples are given and accuracy of our numerical method is demonstrated comparing to their exact solutions. It is shown that the our proposed method can give a good accuracy as high as 10^{-5} even with a few numbers of computational node.

Keywords : finite integration method; Chebyshev polynomial expansion; linear time-dependent differential equation; linear space-fractional differential equation.
2010 Mathematics Subject Classification : 65L05; 65L10; 65N30.

[1]Corresponding author.

1 Introduction

The finite integration method (FIM) is one of the recently developed numerical techniques for finding approximate solutions to boundary value problems for linear differential equations. Similar to the idea of the finite difference method (FDM), we replace the solution domain with a finite number of points, known as grid points and obtain the solution at these points. The grids are generally spaced along the independent coordinates.

The important tool in FIM is an integration matrix. Traditionally, the integration matrix can be obtained by the directly numerical integration, provided by Wen et al. [1], using both of the trapezoidal rule and radial basis functions which have demonstrated that an approximate solution derived from these methods gave the higher accuracy than the FDM. Next, Li et al. [2] used the FIM to solve multi-dimensional problems. After that, Li et al. [3] developed the FIM for solving multi-dimensional partial differential equations (PDEs) by using the Simpson's rule, Newton-Cotes and Lagrange interpolation. They illustrate that their FIM give a lot better result comparing to the FDM. In 2018, Boonklurb et al. [4] modified the traditional FIM by using Chebyshev polynomial to approximate the solution of linear differential equations and the steady state PDEs. Their modification gave a significant better results comparing to the traditional FIM. However, their modified method cannot be applied directly to the problems depending on time and problems involving linear fractional order derivatives. Recently, Saengsiritongchai and Boonklurb [5] developed the algorithm for solving one-dimensional linear time-dependent differential equations via FIM using Chebyshev polynomial.

In this paper, we consider the two-dimensional linear time-dependent PDEs with prescribed initial and boundary conditions. We construct our numerical algorithm for seeking approximate solutions which is combining the modified FIM using Chebyshev polynomial proposed by Boonklurb et al. [4] together with a difference quotient formula to estimate the time derivative and a technique of the Crank-Nicolson method [6]. Moreover, we also construct an algorithm for finding numerical results of linear fractional differential equations (FDEs) based on the FIM via shifted Chebyshev polynomials. For each nodal point, we use the zeros of the shifted Chebyshev polynomial of some degree. We test our algorithms throughout several examples by using the MatLab program to compare our results with the results obtained by other methods and their analytical solutions.

2 Preliminaries

In this section, background knowledge on the definition and properties of the Chebyshev polynomials and shifted Chebyshev polynomials are provided in order to construct the integration matrices which are the main materials for the modified FIM proposed by Boonklurb et al. [4]. Moreover, The defnitions for fractional order derivatives and the form of linear FDEs are presented.

2.1 Chebyshev Polynomials

First, let us introduce the definition and some important properties of the Chebyshev polynomial for constructing the first and higher order integration matrices for solving differential equations.

Definition 2.1. ([7]) The Chebyshev polynomial of degree $n \geq 0$ is defined as

$$T_n(x) = \cos(n \arccos x), \text{ where } x \in [-1, 1].$$

Lemma 2.2. (i) *For $n \in \mathbb{N}$, the zeros of Chebyshev polynomial $T_n(x)$ are*

$$x_k = \cos\Big(\frac{2k-1}{2n}\Big)\pi, \text{ where } k \in \{1, 2, 3, ..., n\}. \tag{2.1}$$

(ii) *The single layer integrations of Chebyshev polynomial $T_n(x)$ are*

$$\begin{aligned}
\overline{T}_0(x) &= \int_{-1}^{x} T_0(\xi)d\xi = x + 1, \\
\overline{T}_1(x) &= \int_{-1}^{x} T_1(\xi)d\xi = \frac{x^2-1}{2}, \\
\overline{T}_n(x) &= \int_{-1}^{x} T_n(\xi)d\xi = \frac{T_{n+1}(x)}{2(n+1)} - \frac{T_{n-1}(x)}{2(n-1)} - \frac{(-1)^n}{n^2-1}, \text{ where } n \geq 2.
\end{aligned}$$

(iii) *The Chebyshev matrix $\mathbf{T}$ at each node $\{x_k\}_{k=1}^n$ defined by (2.1) be*

$$\mathbf{T} = \begin{bmatrix} T_0(x_1) & T_1(x_1) & \dots & T_{n-1}(x_1) \\ T_0(x_2) & T_1(x_2) & \dots & T_{n-1}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ T_0(x_M) & T_1(x_n) & \dots & T_{n-1}(x_n) \end{bmatrix}.$$

Then, it has the multiplicative inverse as $\mathbf{T}^{-1} = \frac{1}{n}\text{diag}(1, 2, 2, ..., 2)\mathbf{T}^\top$.

2.2 Shifted Chebyshev Polynomials

In some applications, the interval $[0, 1]$ is more convenient to use than $[-1, 1]$. Thus, we transform the independent variable of $T_n(x)$ into $[0, 1]$ which is denoted by $T_n^*(x)$ and called a shifted Chebyshev polynomial of degree n for $x \in [0, 1]$. It has the definition and properties as follows.

Definition 2.3. The shifted Chebyshev polynomial of degree $n \geq 0$ is defined as

$$T_n^*(x) = T_n(2x - 1), \text{ where } x \in [0, 1].$$

Lemma 2.4. (i) *For $n \in \mathbb{N}$, the zeros of shifted Chebyshev polynomial $T_n^*(x)$ are*

$$x_k = \frac{1}{2}\Big[\cos\Big(\frac{2k-1}{2n}\Big)\pi + 1\Big], \text{ where } k \in \{1, 2, 3, ..., n\}. \tag{2.2}$$

(ii) *The single layer integrations of shifted Chebyshev polynomial* $T_n^*(x)$ *are*

$$\overline{T}_0^*(x) = \int_0^x T_0(\xi)d\xi = x,$$
$$\overline{T}_1^*(x) = \int_0^x T_0(\xi)d\xi = x^2 - x,$$
$$\overline{T}_n^*(x) = \int_0^x T_0(\xi)d\xi = \frac{T_{n+1}^*(x)}{4(n+1)} - \frac{T_{n-1}^*(x)}{4(n-1)} - \frac{(-1)^n}{2(n^2-1)}, \text{ where } n \geq 2.$$

(iii) *The shifted Chebyshev matrix* $\mathbf{T}^*$ *at each node* $\{x_k\}_{k=1}^n$ *defined by* (2.2) *be*

$$\mathbf{T}^* = \begin{bmatrix} T_0^*(x_1) & T_1^*(x_1) & \dots & T_{n-1}^*(x_1) \\ T_0^*(x_2) & T_1^*(x_2) & \dots & T_{n-1}^*(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ T_0^*(x_n) & T_1^*(x_n) & \dots & T_{n-1}^*(x_n) \end{bmatrix}.$$

Then, it has the multiplicative inverse given by $(\mathbf{T}^*)^{-1} = \frac{1}{n}\text{diag}(1,2,2,...,2)(\mathbf{T}^*)^\top$.

2.3 Modified FIM Using Chebyshev Polynomial

In this section, for ease of reference, we brief a construction of integration matrices from the modified FIM using Chebyshev expansion proposed by Boonklurb et al. [4]. First, let $M \in \mathbb{Z}^+$ and $u(x)$ be a linear combination of the Chebyshev polynomials at node x_k as defined by (2.1), which is

$$u(x_k) = \sum_{n=0}^{M-1} c_n T_n(x_k).$$

For $k \in \{1,2,3,...,M\}$, it can be written in a matrix form: $\mathbf{u} = \mathbf{Tc}$ or $\mathbf{c} = \mathbf{T}^{-1}\mathbf{u}$, where $\mathbf{T}$ and $\mathbf{T}^{-1}$ are defined by Lemma 2.2 (iii), $\mathbf{u} = [u(x_1), u(x_2), u(x_3), ..., u(x_M)]^\top$ and $\mathbf{c} = [c_0, c_1, c_2, ..., c_{M-1}]^\top$. Next, we consider a single layer integration of u from -1 to x_k as

$$U^{(1)}(x_k) = \int_{-1}^{x_k} u(\xi)d\xi = \sum_{n=0}^{M-1} c_n \int_{-1}^{x_k} T_n(\xi)d\xi = \sum_{n=0}^{M-1} c_n \overline{T}_n(x_k),$$

where each $\overline{T}_n(x)$ is defined by Lemma 2.2 (ii). For $k \in \{1,2,3,...,M\}$, the above equation can be written in the matrix form as $\mathbf{U}^{(1)} = \overline{\mathbf{T}}\mathbf{c} = \overline{\mathbf{T}}\mathbf{T}^{-1}\mathbf{u} := \mathbf{Au}$, where $\mathbf{U}^{(1)} = [U^{(1)}(x_1), U^{(1)}(x_2), ..., U^{(1)}(x_M)]^\top$ and

$$\overline{\mathbf{T}} = \begin{bmatrix} \overline{T}_0(x_1) & \overline{T}_1(x_1) & \dots & \overline{T}_{M-1}(x_1) \\ \overline{T}_0(x_2) & \overline{T}_1(x_2) & \dots & \overline{T}_{M-1}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \overline{T}_0(x_M) & \overline{T}_1(x_M) & \dots & \overline{T}_{M-1}(x_M) \end{bmatrix}.$$

This $\mathbf{A} = \overline{\mathbf{T}}\mathbf{T}^{-1} := [a_{ki}]_{M\times M}$ is called the *first order integration matrix*. Next, for $k \in \{1, 2, 3, ..., M\}$, let us consider a double layer integration of u from -1 to x_k as

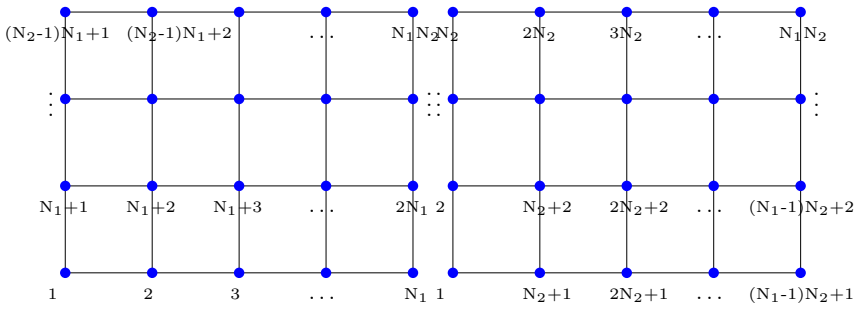
$$U^{(2)}(x_k) = \int_{-1}^{x_k}\int_{-1}^{\xi_2} u(\xi_1)d\xi_1 d\xi_2 = \sum_{i=1}^{M} a_{ki}\int_{-1}^{x_i} u(\xi_1)d\xi_1 = \sum_{i=1}^{M}\sum_{j=1}^{M} a_{ki}a_{ij}u(x_j),$$

which can write in matrix form: $\mathbf{U}^{(2)} = \mathbf{A}^2\mathbf{u}$, where $\mathbf{A}^2$ is called the *second order integration matrix*. Similarly, for the multi-layer integration of u from -1 to x_k as

$$U^{(m)}(x_k) = \int_{-1}^{x_k}\cdots\int_{-1}^{\xi_2} u(\xi_1)d\xi_1 \dots d\xi_m = \sum_{i_m=1}^{M}\cdots\sum_{j=1}^{M} a_{ki_m}\dots a_{i_1 j}u(x_j).$$

It can write in matrix form: $\mathbf{U}^{(m)} = \mathbf{A}^m\mathbf{u}$, where $\mathbf{A}^m$ is called the m^{th} *order integration matrix*.

Next, we consider a two-dimensional domain $\Omega = [a, b]\times[c, d]$, where $a, b, c, d \in \mathbb{R}$ and we transform it into $\overline{\Omega} = [-1, 1] \times [-1, 1]$ which can be discretized by the zeros of the Chebyshev polynomial with the number of total nodes $M = N_1 \times N_2$, where N_1 and N_2 are the number of horizontal and vertical discretized nodes, respectively. For computational convenience, we index numbering of grid points along the x-direction by the global numbering system (Figure 1a) and grid points along y-direction by the local numbering system (Figure 1b).



(a) Global numbering system (b) Local numbering system

Figure 1: The indices of the grid points globally and locally

Let $-1 < x_1 < x_2 < x_3 < ... < x_{N_1} < 1$ and $-1 < y_1 < y_2 < y_3 < ... < y_{N_2} < 1$ be grid points that are generated by the zeros of Chebyshev polynomials. Let $U_x(x, y)$ and $U_y(x, y)$ be the single layer integrations with respect to variables x and y, respectively. Consider $U_x(x_k, y_s)$ in the global system when y_s is fixed.

Then, by the idea in FIM in one dimension, we have

$$U_x(x_k, y_s) = \int_{-1}^{x_k} u(\xi, y_s) d\xi = \sum_{i=1}^{N_1} a_{ki} u(x_i, y_s)$$

for $k \in \{1, 2, 3, ..., N_1\}$, then $\mathbf{U}_x(\cdot, y_s) = \mathbf{A}\mathbf{u}(\cdot, y_s)$, where
$\mathbf{u}(\cdot, y_s) = [u(x_1, y_s), u(x_2, y_s), ..., u(x_{N_1}, y_s)]^\top$ and
$\mathbf{U}_x(\cdot, y_s) = [U_x(x_1, y_s), U_x(x_2, y_s), ..., U_x(x_{N_1}, y_s)]^\top$.
For $s \in \{1, 2, 3, ..., N_2\}$, it can be expressed as

$$\begin{bmatrix} \mathbf{U}_x(\cdot, y_1) \\ \mathbf{U}_x(\cdot, y_2) \\ \vdots \\ \mathbf{U}_x(\cdot, y_{N_2}) \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A} & 0 & \dots & 0 \\ 0 & \mathbf{A} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \mathbf{A} \end{bmatrix}}_{N_2 \text{ blocks}} \begin{bmatrix} \mathbf{u}(\cdot, y_1) \\ \mathbf{u}(\cdot, y_2) \\ \vdots \\ \mathbf{u}(\cdot, y_{N_2}) \end{bmatrix}, \tag{2.3}$$

where $\mathbf{A} = \overline{\mathbf{T}}\mathbf{T}^{-1}$ is the $N_1 \times N_1$ matrix. We denote (2.3) by $\mathbf{U}_x = \mathbf{A}_x\mathbf{u}$. Next, we consider $U_y(x_k, y_s)$ in the local system when x_k is fixed. Then, by the idea in FIM in one dimension, we have

$$U_y(x_k, y_s) = \int_{-1}^{y_k} u(x_k, \eta) d\eta = \sum_{j=1}^{N_2} a_{sj} u(x_k, y_j)$$

for $s \in \{1, 2, 3, ..., N_2\}$, then $\widetilde{\mathbf{U}}_y(x_k, \cdot) = \mathbf{A}\widetilde{\mathbf{u}}(x_k, \cdot)$, where
$\widetilde{\mathbf{u}}(x_k, \cdot) = [u(x_k, y_1), u(x_k, y_2), ..., u(x_k, y_{N_2})]^\top$ and
$\widetilde{\mathbf{U}}_y(x_k, \cdot) = [U_y(x_k, y_1), U_y(x_k, y_2), ..., U_k(x_k, y_{N_2})]^\top$.
For $k \in \{1, 2, 3, ..., N_1\}$, it can be written as

$$\begin{bmatrix} \widetilde{\mathbf{U}}_y(x_1, \cdot) \\ \widetilde{\mathbf{U}}_y(x_2, \cdot) \\ \vdots \\ \widetilde{\mathbf{U}}_y(x_{N_1}, \cdot) \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A} & 0 & \dots & 0 \\ 0 & \mathbf{A} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \mathbf{A} \end{bmatrix}}_{N_1 \text{ blocks}} \begin{bmatrix} \widetilde{\mathbf{u}}(x_1, \cdot) \\ \widetilde{\mathbf{u}}(x_2, \cdot) \\ \vdots \\ \widetilde{\mathbf{u}}(x_{N_1}, \cdot) \end{bmatrix}, \tag{2.4}$$

where $\mathbf{A} = \overline{\mathbf{T}}\mathbf{T}^{-1}$ is the $N_2 \times N_2$ matrix and denote (2.4) by $\widetilde{\mathbf{U}}_y = \widetilde{\mathbf{A}}_y\widetilde{\mathbf{u}}$. The integration and integrand vectors in the local numbering system can be transformed to the global numbering system by using the transformation matrix $\mathbf{P}$, i.e., $\mathbf{U}_y = \mathbf{P}\widetilde{\mathbf{U}}_y$ and $\mathbf{u} = \mathbf{P}\widetilde{\mathbf{u}}$. The transformation matrix $\mathbf{P}$ is defined by

$$\mathbf{P}_{mn} = \begin{cases} 1 & ; \begin{cases} m = N_1 \times (j-1) + i, \\ n = N_2 \times (i-1) + j, \end{cases} \\ 0 & ; \text{ otherwise,} \end{cases}$$

for all $i \in \{1,2,3,...,N_1\}$ and $j \in \{1,2,3,...,N_2\}$. Therefore, we have the integration matrix with respect to y in the global numbering system as $\mathbf{U}_y = \mathbf{A}_y\mathbf{u}$, where $\mathbf{A}_y = \mathbf{P}\widetilde{\mathbf{A}}_y\mathbf{P}^{-1} = \mathbf{P}\widetilde{\mathbf{A}}_y\mathbf{P}^{\top}$.

Remark 2.5 ([4]). *For $m, n \in \mathbb{N}$, the multi-layer integrations in the global numbering system can be represented in the matrix form as follows:*

- *the m^{th} layer integration with respect to only x is $\mathbf{U}_x^{(m)} = \mathbf{A}_x^m\mathbf{u}$,*
- *the n^{th} layer integration with respect to only y is $\mathbf{U}_y^{(n)} = \mathbf{A}_y^n\mathbf{u}$,*
- *the multi-layer integration with respect to both x and y is $\mathbf{U}_{xy}^{(m,n)} = \mathbf{A}_x^m\mathbf{A}_y^n\mathbf{u} = \mathbf{A}_y^n\mathbf{A}_x^m\mathbf{u}$.*

2.4 Riemann-Liouville Definition of Fractional Order Derivative

The fractional order derivative is a derivative which has order as a fraction instead of an integer. Many researchers gave definitions for fractional derivatives. Each definition involves both local and global properties. For examples, the Riemann-Liouville and Caputo definitions involve the local property but the Grunwald-Letnikov definition involves the global property, see [8] and [9] for further details. In this work, we use the Riemann-Liouville definition of fractional derivative as follow.

Definition 2.6. ([8]) Let $\alpha \in (m-1, m)$ for $m \in \mathbb{Z}^+$ and $x \in [0, b]$ for $b \in \mathbb{R}^+$. The Riemann-Liouville fractional derivative of order α of a function u is

$$D^{\alpha}u(x) = \frac{1}{\Gamma(m-\alpha)}\frac{d^m}{dx^m}\int_0^x \frac{u(s)}{(x-s)^{\alpha-m+1}}ds, \tag{2.5}$$

where $\Gamma(\cdot)$ is the gamma function and $u \in L^1(0, b)$.

3 Numerical Solution of 2-D Linear Time-Dependent PDEs

In this section, we create a numerical algorithm for finding approximate solutions of linear time-dependent differential equation in two-dimensional space. First of all, we let a, b, c, d and T to be real numbers such that $a < b$, $c < d$ and $T > 0$. We consider the problem over $(a, b)\times(c, d)\times(0, T)$ with the time-dependent variables in two dimensions as follows.

$$\frac{\partial u}{\partial t} = \alpha_1\frac{\partial^2 u}{\partial x^2} + \alpha_2\frac{\partial^2 u}{\partial y^2} + \alpha_3\frac{\partial^2 u}{\partial x\partial y} + \alpha_4\frac{\partial u}{\partial x} + \alpha_5\frac{\partial u}{\partial y} + \alpha_6 u + f \tag{3.1}$$

subject to the initial condition $u(x, y, 0) = F(x, y)$ and the Dirichlet boundary conditions for $t \in [0, T]$,

$$u(a,y,t) = h_1(y,t), \quad u(b,y,t) = h_2(y,t), \quad y \in (c,d),$$
$$u(x,c,t) = h_3(x,t), \quad u(x,d,t) = h_4(x,t), \quad x \in (a,b),$$

where $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, f$ and u are the given functions of x, y and t. First, we approximate $\frac{\partial u}{\partial t}$ by using the forward difference quotient as

$$\frac{\partial u}{\partial t} = \frac{u^{j+1}(x,y) - u^j(x,y)}{\tau}, \tag{3.2}$$

where τ is a time step and $u^j = u^j(x,y) = u(x,y,t_j)$. Next, we let G be the right-hand-side of (3.1), that is $G(t,u) = \alpha_1 u_{xx} + \alpha_2 u_{yy} + \alpha_3 u_{xy} + \alpha_4 u_x + \alpha_5 u_y + \alpha_6 u + f$ and then we approximate $G(t,u)$ by using the Crank-Nicolson method [6], i.e.,

$$G(t,u) = \frac{1}{2}\Big[G(t_j, u^j) + G(t_{j+1}, u^{j+1})\Big]. \tag{3.3}$$

Thus, from (3.1), we see that (3.2) and (3.3) are equals, then we have

$$u^{j+1}(x,y) = u^j(x,y) + \frac{\tau}{2}\Big[G(t_j, u^j) + G(t_{j+1}, u^{j+1})\Big]. \tag{3.4}$$

For convenience, let $f^j = f(x,y,t_j)$ and $\alpha_i^j = \alpha_i(x,y,t_j)$ for $i \in \{1,2,3,4,5,6\}$, then we substitute the definition of G in (3.4) and group the terms of u at $(j+1)^{th}$ and j^{th} time steps together. Thus, we obtain

$$\begin{aligned} &-(\alpha_1^{j+1}u_{xx}^{j+1} + \alpha_2^{j+1}u_{yy}^{j+1} + \alpha_3^{j+1}u_{xy}^{j+1} + \alpha_4^{j+1}u_x^{j+1} + \alpha_5^{j+1}u_y^{j+1} + \alpha_6^{j+1}u^{j+1}) + \frac{2}{\tau}u^{j+1} \\ &= (\alpha_1^j u_{xx}^j + \alpha_2^j u_{yy}^j + \alpha_3^j u_{xy}^j + \alpha_4^j u_x^j + \alpha_5^j u_y^j + \alpha_6^j u^j) + \frac{2}{\tau}u^j + f^j + f^{j+1}. \end{aligned} \tag{3.5}$$

3.1 Numerical Algorithm for 2-D Linear Time-Dependent PDEs

We are now ready to apply the modified FIM using Chebyshev polynomial to devise an algorithm for calculating the approximate solution of (3.1).

Step 1: Transform the spatial domain $\Omega = [a,b] \times [c,d]$ into $\overline{\Omega} = [-1,1] \times [-1,1]$ by using the linear transformations $\bar{x} = \frac{2x-a-b}{b-a}$ and $\bar{y} = \frac{2y-c-d}{d-c}$. Then, (3.5) becomes

$$\begin{aligned} &-(p^2\bar{\alpha}_1^{j+1}\bar{u}_{\bar{x}\bar{x}}^{j+1} + q^2\bar{\alpha}_2^{j+1}\bar{u}_{\bar{y}\bar{y}}^{j+1} + pq\bar{\alpha}_3^{j+1}\bar{u}_{\bar{x}\bar{y}}^{j+1} + p\bar{\alpha}_4^{j+1}\bar{u}_{\bar{x}}^{j+1} + q\bar{\alpha}_5^{j+1}\bar{u}_{\bar{y}}^{j+1} \\ &+ \bar{\alpha}_6^{j+1}\bar{u}^{j+1}) + \frac{2}{\tau}\bar{u}^{j+1} \\ &= (p^2\bar{\alpha}_1^j\bar{u}_{\bar{x}\bar{x}}^j + q^2\bar{\alpha}_2^j\bar{u}_{\bar{y}\bar{y}}^j + pq\bar{\alpha}_3^j\bar{u}_{\bar{x}\bar{y}}^j + p\bar{\alpha}_4^j\bar{u}_{\bar{x}}^j + q\bar{\alpha}_5^j\bar{u}_{\bar{y}}^j + \bar{\alpha}_6^j\bar{u}^j) + \frac{2}{\tau}\bar{u}^j + \bar{f}^{j+1} + \bar{f}^j, \end{aligned}$$

where $p = \frac{2}{b-a}$, $q = \frac{2}{d-c}$, $\bar{f}^j(\bar{x},\bar{y}) = f(x,y,t_j)$, $\bar{u}^j(\bar{x},\bar{y}) = u(x,y,t_j)$ and $\bar{\alpha}_i^j(\bar{x},\bar{y}) = \alpha_i(x,y,t_j)$. Next, we let $Z^j = p^2\bar{\alpha}_1^j\bar{u}_{\bar{x}\bar{x}}^j + q^2\bar{\alpha}_2^j\bar{u}_{\bar{y}\bar{y}}^j + pq\bar{\alpha}_3^j\bar{u}_{\bar{x}\bar{y}}^j + p\bar{\alpha}_4^j\bar{u}_{\bar{x}}^j + q\bar{\alpha}_5^j\bar{u}_{\bar{y}}^j + \bar{\alpha}_6^j\bar{u}^j$, then the above equation becomes

$$-Z^{j+1} + \frac{2}{\tau}\bar{u}^{j+1} = Z^j + \frac{2}{\tau}\bar{u}^j + \bar{f}^{j+1} + \bar{f}^j. \tag{3.6}$$

Note that for x and y in the functions f, u and α_i are $x = \frac{1}{2}[(b-a)\bar{x} + a + b]$ and $y = \frac{1}{2}[(d-c)\bar{y} + c + d]$.

Step 2: We discretize the domain $\overline{\Omega} = [-1,1] \times [-1,1]$ into N_1 and N_2 nodes along x and y directions, which are generated by zeros of Chebyshev polynomials T_{N_1} and T_{N_2} as follows:

$$\bar{x}_k = \cos\Big(\frac{2k-1}{2N_1}\Big)\pi, \text{ for } k \in \{1,2,3,...,N_1\},$$
$$\bar{y}_s = \cos\Big(\frac{2s-1}{2N_2}\Big)\pi, \text{ for } s \in \{1,2,3,...,N_2\}.$$

Then, the number of total grid points in the global system is $M = N_1 \times N_2$.

Step 3: Eliminate the derivatives out of (3.6) by taking four-layer integration and use the technique of integration by parts. Thus, we have

$$\int_{-1}^{\bar{y}_s}\int_{-1}^{\eta_2}\int_{-1}^{\bar{x}_k}\int_{-1}^{\xi_2}\Big(-Z^{j+1} + \frac{2}{\tau}\bar{u}^{j+1}\Big)d\xi_1 d\xi_2 d\eta_1 d\eta_2$$
$$= \int_{-1}^{\bar{y}_s}\int_{-1}^{\eta_2}\int_{-1}^{\bar{x}_k}\int_{-1}^{\xi_2}\Big(Z^j + \frac{2}{\tau}\bar{u}^j + \bar{f}^{j+1} + \bar{f}^j\Big)d\xi_1 d\xi_2 d\eta_1 d\eta_2.$$

Now, we consider the integration of Z^j only. As a result, the integration of

Z^{j+1} in the same way. Then,

$$\begin{aligned}
&\int_{-1}^{\bar{y}_s}\int_{-1}^{\eta_2}\int_{-1}^{\bar{x}_k}\int_{-1}^{\xi_2} Z^j\, d\xi_1 d\xi_2 d\eta_1 d\eta_2 \\
&= \int_{-1}^{\bar{y}_s}\int_{-1}^{\eta_2} p^2\Big(\bar{\alpha}_1^j\bar{u}^j - 2\int_{-1}^{\bar{x}_k}\frac{\partial\bar{\alpha}_1^j}{\partial\xi_1}\bar{u}^j d\xi_2 + \int_{-1}^{\bar{x}_k}\int_{-1}^{\xi_2}\frac{\partial^2\bar{\alpha}_1^j}{\partial\xi_1^2}\bar{u}^j d\xi_1 d\xi_2\Big) d\eta_1 d\eta_2 \\
&+ \int_{-1}^{\bar{x}_k}\int_{-1}^{\xi_2} q^2\Big(\bar{\alpha}_2^j\bar{u}^j - 2\int_{-1}^{\bar{y}_s}\frac{\partial\bar{\alpha}_2^j}{\partial\eta_1}\bar{u}^j d\eta_2 + \int_{-1}^{\bar{y}_s}\int_{-1}^{\eta_2}\frac{\partial^2\bar{\alpha}_2^j}{\partial\eta_1^2}\bar{u}^j d\eta_1 d\eta_2\Big) d\xi_1 d\xi_2 \\
&+ \int_{-1}^{\bar{y}_s}\int_{-1}^{\bar{x}_k} pq\Big(\bar{\alpha}_3^j\bar{u}^j - \int_{-1}^{\xi_2}\frac{\partial\bar{\alpha}_3^j}{\partial\xi_1}\bar{u}^j d\xi_1 - \int_{-1}^{\eta_2}\frac{\partial\bar{\alpha}_3^j}{\partial\eta_1}\bar{u}^j d\eta_1 \\
&+ \int_{-1}^{\eta_2}\int_{-1}^{\xi_2}\frac{\partial^2\bar{\alpha}_3^j}{\partial\xi_1\partial\eta_1}\bar{u}^j d\xi_1 d\eta_1\Big) d\xi_2 d\eta_2 \\
&+ \int_{-1}^{\bar{y}_s}\int_{-1}^{\eta_2}\int_{-1}^{\bar{x}_k} p\Big(\bar{\alpha}_4^j\bar{u}^j - \int_{-1}^{\xi_2}\frac{\partial\bar{\alpha}_4^j}{\partial\xi_1}\bar{u}^j d\xi_1\Big) d\xi_2 d\eta_1\eta_2 \\
&+ \int_{-1}^{\bar{y}_s}\int_{-1}^{\bar{x}_k}\int_{-1}^{\xi_2} q\Big(\bar{\alpha}_5^j\bar{u}^j - \int_{-1}^{\eta_2}\frac{\partial\bar{\alpha}_5^j}{\partial\eta_1}\bar{u}^j d\eta_1\Big) d\xi_1 d\xi_2\eta_2 \\
&+ \int_{-1}^{\bar{y}_s}\int_{-1}^{\eta_2}\int_{-1}^{\bar{x}_k}\int_{-1}^{\xi_2}\bar{\alpha}_6^j\bar{u}^j d\xi_1 d\xi_2 d\eta_1 d\eta_2 + \bar{x}_k f_0(\bar{y}_s) + f_1(\bar{y}_s) + \bar{y}_s g_0(\bar{x}_k) + g_1(\bar{x}_k),
\end{aligned}$$

where $f_0(\bar{y}_s)$, $f_1(\bar{y}_s)$, $g_0(\bar{x}_k)$ and $g_1(\bar{x}_k)$ are arbitrary functions emerged from the process of integration that are approximated by Chebyshev interpolating polynomials,

$$g_l(\bar{x}_k) = \sum_{i=0}^{N_1-1} g_i^l T_i(\bar{x}_k) \quad \text{and} \quad f_l(\bar{y}_s) = \sum_{j=0}^{N_2-1} f_j^l T_j(\bar{y}_s) \tag{3.7}$$

for $l \in \{0,1\}$, where $g_0^l, g_1^l, g_2^l, ..., g_{N_1-1}^l$ and $f_0^l, f_1^l, f_2^l, ..., f_{N_2-1}^l$ are the unknown values of those interpolated points which are determined from the given boundary conditions and we define the vectors of these nodes in each coordinates x and y by giving

$$\begin{aligned}
\hat{\mathbf{x}} &= \underbrace{[\mathbf{x}_0, \mathbf{x}_0, \mathbf{x}_0, ..., \mathbf{x}_0]}_{N_2 \text{ blocks}}^{\top}, \quad \text{where} \quad \mathbf{x}_0 = [\bar{x}_1, \bar{x}_2, \bar{x}_3, ..., \bar{x}_{N_1}], \\
\hat{\mathbf{y}} &= [\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, ..., \mathbf{y}_{N_2}]^{\top}, \quad \text{where} \quad \mathbf{y}_s = \underbrace{[\bar{y}_s, \bar{y}_s, \bar{y}_s, ..., \bar{y}_s]}_{N_1 \text{ terms}},
\end{aligned}$$

with $s \in \{1, 2, 3, ..., N_2\}$. Hence, $\hat{\mathbf{x}}, \hat{\mathbf{y}} \in \mathbb{R}^M$ and let $\hat{x}_i$ and $\hat{y}_i$ be the i^{th} element in $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$, respectively.

Step 4: Use the idea in Section 2.3 to transform the equation in Step 3 into the matrix form as

$$\Big(\frac{2}{\tau}\mathbf{A}_x^2\mathbf{A}_y^2 - \mathbf{Z}^{j+1}\Big)\mathbf{u}^{j+1} + \mathbf{X}\mathbf{\Phi}_y\mathbf{f}_0 + \mathbf{\Phi}_y\mathbf{f}_1 + \mathbf{Y}\mathbf{\Phi}_x\mathbf{g}_0 + \mathbf{\Phi}_x\mathbf{g}_1 = \Big(\frac{2}{\tau}\mathbf{A}_x^2\mathbf{A}_y^2 + \mathbf{Z}^{j}\Big)\mathbf{u}^{j} + \mathbf{A}_x^2\mathbf{A}_y^2(\mathbf{F}^{j+1} + \mathbf{F}^{j}), \tag{3.8}$$

where $\mathbf{X} = \mathrm{diag}(\hat{\mathbf{x}})$, $\mathbf{Y} = \mathrm{diag}(\hat{\mathbf{y}})$, $\mathbf{Z}^j = p^2\mathbf{A}_y^2(\bar{\boldsymbol{\alpha}}_1^j - 2\mathbf{A}_x\bar{\boldsymbol{\alpha}}_{1,\bar{x}}^j + \mathbf{A}_x^2\bar{\boldsymbol{\alpha}}_{1,\bar{x}\bar{x}}^j) + q^2\mathbf{A}_x^2(\bar{\boldsymbol{\alpha}}_2^j - 2\mathbf{A}_y\bar{\boldsymbol{\alpha}}_{2,\bar{y}}^j + \mathbf{A}_y^2\bar{\boldsymbol{\alpha}}_{2,\bar{y}\bar{y}}^j) + pq\mathbf{A}_x\mathbf{A}_y(\bar{\boldsymbol{\alpha}}_3^j - \mathbf{A}_x\bar{\boldsymbol{\alpha}}_{3,\bar{x}}^j - \mathbf{A}_y\bar{\boldsymbol{\alpha}}_{3,\bar{y}}^j + \mathbf{A}_x\mathbf{A}_y\bar{\boldsymbol{\alpha}}_{3,\bar{x}\bar{y}}^j) + p\mathbf{A}_x\mathbf{A}_y^2(\bar{\boldsymbol{\alpha}}_4^j - \mathbf{A}_x\bar{\boldsymbol{\alpha}}_{4,\bar{x}}^j) + q\mathbf{A}_x^2\mathbf{A}_y(\bar{\boldsymbol{\alpha}}_5^j - \mathbf{A}_y\bar{\boldsymbol{\alpha}}_{5,\bar{y}}^j\mathbf{u}^j) + \mathbf{A}_x^2\mathbf{A}_y^2\bar{\boldsymbol{\alpha}}_6^j$,
$\mathbf{u}^j = [\bar{u}^j(\hat{x}_1,\hat{y}_1), \bar{u}^j(\hat{x}_2,\hat{y}_2), ..., \bar{u}^j(\hat{x}_M,\hat{y}_M)]^\top$,
$\mathbf{F}^j = [\bar{f}^j(\hat{x}_1,\hat{y}_1), \bar{f}^j(\hat{x}_2,\hat{y}_2), ..., \bar{f}^j(\hat{x}_M,\hat{y}_M)]^\top$,
$\bar{\boldsymbol{\alpha}}_{i,\cdot}^j = \mathrm{diag}(\bar{\alpha}_{i,\cdot}^j(\hat{x}_1,\hat{y}_1), \bar{\alpha}_{i,\cdot}^j(\hat{x}_2,\hat{y}_2), ..., \bar{\alpha}_{i,\cdot}^j(\hat{x}_M,\hat{y}_M))$, $\mathbf{g}_l = [g_0^l, g_1^l, ..., g_{N_1-1}^l]^\top$ and $\mathbf{f}_l = [f_0^l, f_1^l, ..., f_{N_2-1}^l]^\top$. From (3.7), we obtain $\mathbf{\Phi}_x$ and $\mathbf{\Phi}_y$, which their elements can be found by (2.1), as follow:

$$\mathbf{\Phi}_x = \begin{bmatrix} T_0(\hat{x}_1) & T_1(\hat{x}_1) & \dots & T_{N_1-1}(\hat{x}_1) \\ T_0(\hat{x}_2) & T_1(\hat{x}_2) & \dots & T_{N_1-1}(\hat{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ T_0(\hat{x}_M) & T_1(\hat{x}_M) & \dots & T_{N_1-1}(\hat{x}_M) \end{bmatrix}$$

and

$$\mathbf{\Phi}_y = \begin{bmatrix} T_0(\hat{y}_1) & T_1(\hat{y}_1) & \dots & T_{N_2-1}(\hat{y}_1) \\ T_0(\hat{y}_2) & T_1(\hat{y}_2) & \dots & T_{N_2-1}(\hat{y}_2) \\ \vdots & \vdots & \ddots & \vdots \\ T_0(\hat{y}_M) & T_1(\hat{y}_M) & \dots & T_{N_2-1}(\hat{y}_M) \end{bmatrix}.$$

Step 5: We consider the given boundary conditions in four cases:

- Left boundary condition:
  $\bar{u}^{j+1}(-1,\bar{y}) = \sum_{n=0}^{N_1-1} c_n T_n(-1) := \mathbf{t}_l\mathbf{c} = \mathbf{t}_l\mathbf{T}_{N_1\times N_1}^{-1}\mathbf{u}^{j+1}(\cdot,\bar{y}) = \bar{h}_1^{j+1}(\bar{y})$
  for each $\bar{y} \in \{\bar{y}_1, \bar{y}_2, \bar{y}_3, ..., \bar{y}_{N_2}\}$, then we have

$$\underbrace{\begin{bmatrix} \mathbf{t}_l\mathbf{T}_{N_1\times N_1}^{-1} & 0 & \dots & 0 \\ 0 & \mathbf{t}_l\mathbf{T}_{N_1\times N_1}^{-1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \mathbf{t}_l\mathbf{T}_{N_1\times N_1}^{-1} \end{bmatrix}}_{N_2 \text{ blocks}} \begin{bmatrix} \mathbf{u}^{j+1}(\cdot,\bar{y}_1) \\ \mathbf{u}^{j+1}(\cdot,\bar{y}_2) \\ \vdots \\ \mathbf{u}^{j+1}(\cdot,\bar{y}_{N_2}) \end{bmatrix} = \begin{bmatrix} \bar{h}_1^{j+1}(\bar{y}_1) \\ \bar{h}_1^{j+1}(\bar{y}_2) \\ \vdots \\ \bar{h}_1^{j+1}(\bar{y}_{N_2}) \end{bmatrix},$$

  which we denote it by $\mathbf{T}_l\mathbf{u}^{j+1} = \mathbf{H}_1^{j+1}$, where $\mathbf{t}_l = [1, -1, 1, ..., (-1)^{N_1-1}]$.

- Right boundary condition:
  $\bar{u}^{j+1}(1,\bar{y}) = \sum_{n=0}^{N_1-1} c_n T_n(1) := \mathbf{t}_r\mathbf{c} = \mathbf{t}_r\mathbf{T}_{N_1\times N_1}^{-1}\mathbf{u}^{j+1}(\cdot,\bar{y}) = \bar{h}_2^{j+1}(\bar{y})$

for each $\bar{y} \in \{\bar{y}_1, \bar{y}_2, \bar{y}_3, ..., \bar{y}_{N_2}\}$, then we have

$$\underbrace{\begin{bmatrix} \mathbf{t}_r\mathbf{T}^{-1}_{N_1\times N_1} & 0 & \dots & 0 \\ 0 & \mathbf{t}_r\mathbf{T}^{-1}_{N_1\times N_1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \mathbf{t}_r\mathbf{T}^{-1}_{N_1\times N_1} \end{bmatrix}}_{N_2 \text{ blocks}} \begin{bmatrix} \mathbf{u}^{j+1}(\cdot, \bar{y}_1) \\ \mathbf{u}^{j+1}(\cdot, \bar{y}_2) \\ \vdots \\ \mathbf{u}^{j+1}(\cdot, \bar{y}_{N_2}) \end{bmatrix} = \begin{bmatrix} \bar{h}_2^{j+1}(\bar{y}_1) \\ \bar{h}_2^{j+1}(\bar{y}_2) \\ \vdots \\ \bar{h}_2^{j+1}(\bar{y}_{N_2}) \end{bmatrix},$$

which we denote it by $\mathbf{T}_r\mathbf{u}^{j+1} = \mathbf{H}_2^{j+1}$, where $\mathbf{t}_r = [1, 1, 1, ..., 1^{N_1-1}]$.

- Bottom boundary condition:
$\bar{u}^{j+1}(\bar{x}, -1) = \sum_{n=0}^{N_2-1} c_n T_n(-1) := \mathbf{t}_b\mathbf{c} = \mathbf{t}_b\mathbf{T}^{-1}_{N_2\times N_2}\widetilde{\mathbf{u}}^{j+1}(\bar{x}, \cdot) = \bar{h}_3^{j+1}(\bar{x})$
for each $\bar{x} \in \{\bar{x}_1, \bar{x}_2, \bar{x}_3, ..., \bar{x}_{N_1}\}$, then we have

$$\underbrace{\begin{bmatrix} \mathbf{t}_b\mathbf{T}^{-1}_{N_2\times N_2} & 0 & \dots & 0 \\ 0 & \mathbf{t}_b\mathbf{T}^{-1}_{N_2\times N_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \mathbf{t}_b\mathbf{T}^{-1}_{N_2\times N_2} \end{bmatrix}}_{N_1 \text{ blocks}} \begin{bmatrix} \widetilde{\mathbf{u}}^{j+1}(\bar{x}_1, \cdot) \\ \widetilde{\mathbf{u}}^{j+1}(\bar{x}_2, \cdot) \\ \vdots \\ \widetilde{\mathbf{u}}^{j+1}(\bar{x}_{N_1}, \cdot) \end{bmatrix} = \begin{bmatrix} \bar{h}_3^{j+1}(\bar{x}_1) \\ \bar{h}_3^{j+1}(\bar{x}_2) \\ \vdots \\ \bar{h}_3^{j+1}(\bar{x}_{N_1}) \end{bmatrix},$$

which we denote it by $\mathbf{T}_b\widetilde{\mathbf{u}}^{j+1} = \mathbf{H}_3^{j+1}$ or $\mathbf{T}_b\mathbf{P}^{-1}\mathbf{u}^{j+1} = \mathbf{H}_3^{j+1}$, where $\mathbf{t}_b = [1, -1, 1, ..., (-1)^{N_2-1}]$.

- Upper boundary condition:
$\bar{u}^{j+1}(\bar{x}, 1) = \sum_{n=0}^{N_2-1} c_n T_n(1) := \mathbf{t}_u\mathbf{c} = \mathbf{t}_u\mathbf{T}^{-1}_{N_2\times N_2}\widetilde{\mathbf{u}}^{j+1}(\bar{x}, \cdot) = \bar{h}_4^{j+1}(\bar{x})$
for each $\bar{x} \in \{\bar{x}_1, \bar{x}_2, \bar{x}_3, ..., \bar{x}_{N_1}\}$, then we have

$$\underbrace{\begin{bmatrix} \mathbf{t}_u\mathbf{T}^{-1}_{N_2\times N_2} & 0 & \dots & 0 \\ 0 & \mathbf{t}_u\mathbf{T}^{-1}_{N_2\times N_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \mathbf{t}_u\mathbf{T}^{-1}_{N_2\times N_2} \end{bmatrix}}_{N_1 \text{ blocks}} \begin{bmatrix} \widetilde{\mathbf{u}}^{j+1}(\bar{x}_1, \cdot) \\ \widetilde{\mathbf{u}}^{j+1}(\bar{x}_2, \cdot) \\ \vdots \\ \widetilde{\mathbf{u}}^{j+1}(\bar{x}_{N_1}, \cdot) \end{bmatrix} = \begin{bmatrix} \bar{h}_4^{j+1}(\bar{x}_1) \\ \bar{h}_4^{j+1}(\bar{x}_2) \\ \vdots \\ \bar{h}_4^{j+1}(\bar{x}_{N_1}) \end{bmatrix},$$

which we denote it by $\mathbf{T}_u\widetilde{\mathbf{u}}^{j+1} = \mathbf{H}_4^{j+1}$ or $\mathbf{T}_u\mathbf{P}^{-1}\mathbf{u}^{j+1} = \mathbf{H}_4^{j+1}$, where $\mathbf{t}_b = [1, 1, 1, ..., 1^{N_2-1}]$.

Thus, all boundary conditions can be represented in matrix forms as

$$\mathbf{T}_l\mathbf{u}^{j+1} = \mathbf{H}_1^{j+1}, \quad \mathbf{T}_r\mathbf{u}^{j+1} = \mathbf{H}_2^{j+1}, \quad \mathbf{T}_b\mathbf{P}^{-1}\mathbf{u}^{j+1} = \mathbf{H}_3^{j+1}, \quad \mathbf{T}_u\mathbf{P}^{-1}\mathbf{u}^{j+1} = \mathbf{H}_4^{j+1}. \tag{3.9}$$

Step 6: Construct the system of linear equations from (3.8) and (3.9). Then, we obtain

$$\left[\begin{array}{c|cccc} \frac{2}{\tau}\mathbf{A}_x^2\mathbf{A}_y^2 - \mathbf{Z}^{j+1} & \mathbf{X}\mathbf{\Phi}_y & \mathbf{\Phi}_y & \mathbf{Y}\mathbf{\Phi}_x & \mathbf{\Phi}_x \\ \hline \mathbf{T}_l & 0 & 0 & \dots & 0 \\ \mathbf{T}_r & 0 & 0 & \dots & 0 \\ \mathbf{T}_b\mathbf{P}^{-1} & \vdots & \vdots & \ddots & \vdots \\ \mathbf{T}_u\mathbf{P}^{-1} & 0 & 0 & \dots & 0 \end{array}\right] \left[\begin{array}{c} \mathbf{u}^{j+1} \\ \hline \mathbf{f}_0 \\ \mathbf{f}_1 \\ \mathbf{g}_0 \\ \mathbf{g}_1 \end{array}\right] = \left[\begin{array}{c} \mathbf{B} \\ \hline \mathbf{H}_1^{j+1} \\ \mathbf{H}_2^{j+1} \\ \mathbf{H}_3^{j+1} \\ \mathbf{H}_4^{j+1} \end{array}\right], \quad (3.10)$$

where $\mathbf{B} = (\frac{2}{\tau}\mathbf{A}_x^2\mathbf{A}_y^2 + \mathbf{Z}^j)\mathbf{u}^j + \mathbf{A}_x^2\mathbf{A}_y^2(\mathbf{F}^{j+1} + \mathbf{F}^j)$. Finally, we can find the approximate solutions $\bar{u}^{j+1}(\bar{x}, \bar{y})$ by solving above linear system (3.10). Note that the numerical solution $u^{j+1}(x, y)$ for $(x, y) \in [a, b] \times [c, d]$ is equivalent to $\bar{u}^{j+1}(\bar{x}, \bar{y})$ for $(\bar{x}, \bar{y}) \in [-1, 1] \times [-1, 1]$.

3.2 Numerical Examples for 2-D Linear Time-Dependent PDEs

In this subsection, we use our proposed numerical algorithm to find the approximate solutions of some linear time-dependent PDEs in two-dimensional space. To demonstrate the efficiency of this procedure by comparing an error of their solutions. In this work, we consume the average relative error defined by ARE $= \frac{1}{M}\sum_{i=1}^{M}\left|\frac{u_i^* - u_i}{u_i^*}\right|$, where u^* and u are the analytical and numerical solutions, respectively. Moreover, we demonstrate the computational cost in terms of CPU times(s). All calculations are implemented by MatLab R2016 and run on the Intel(R) Core(TM) i7-6700 CPU @ 3.40 GHz. Finally, we also depict the surface plot and the grid points of our solutions throughout the following examples.

Example 3.1. Consider a time-dependent linear PDE in which the coefficients do not depend on time.

$$\begin{aligned} \frac{\partial u}{\partial t} =& (x^2 + y^2 + 1)\frac{\partial^2 u}{\partial x^2} + (x^2 + y^2 + 1)\frac{\partial^2 u}{\partial y^2} \\ &+ 2x\frac{\partial u}{\partial x} + 2y\frac{\partial u}{\partial y} - 2u + [1 - 2(x^2 + y^2 + x + y)]e^{x+y+t} \end{aligned}$$

over $(x, y) \in (0, 1) \times (0, 1)$ and $t \in (0, 1)$, with the initial and boundary conditions:

$$\begin{aligned} &u(x, y, 0) = e^{x+y}, \quad (x, y) \in [0, 1] \times [0, 1], \\ &u(x, 0, t) = e^{x+t}, \quad u(x, 1, t) = e^{x+1+t}, \quad x \in [0, 1], \quad t \in [0, 1], \\ &u(0, y, t) = e^{y+t}, \quad u(1, y, t) = e^{1+y+t}, \quad y \in [0, 1], \quad t \in [0, 1]. \end{aligned}$$

The exact solution for this problem is $u^*(x, y, t) = e^{x+y+t}$. We first transform our domain $\Omega = [0, 1] \times [0, 1]$ by using the transformations $\bar{x} = 2x - 1$ and $\bar{y} = 2y - 1$. By our numerical algorithm, we choose the numbers of nodes along x and y directions that are equal, i.e., $N_1 = N_2 = N$ for $N \in \{6, 8, 10, 12\}$. Thus, we can solve this

problem by using the linear system (3.10) and we compare the average relative errors obtained by our proposed method (CBS) with the original FIMs, consisting the trapezoidal and Simpson's rules (TPZ and SIM), at different nodal points N for $\tau \in \{0.1, 0.01\}$ as shown in Table 1. The graphical solutions of this problem at final time $t = 1$ are illustrated in Figure 2 both the three-dimensional surface of our numerical solution and the two-dimensional graph that horizontal axis is the orders of nodes in global numbering system versus vertical axis is the exact and approximate solutions.

N	FIM-TPZ	FIM-SIM	Our proposed method	
	$\tau = 0.1$	$\tau = 0.1$	$\tau = 0.1$	$\tau = 0.01$
6	2.9684×10^{-2}	2.6237×10^{-2}	1.1486×10^{-5}	2.3278×10^{-6}
8	3.1323×10^{-2}	2.9074×10^{-2}	9.9216×10^{-6}	1.0197×10^{-7}
10	3.2534×10^{-2}	3.0984×10^{-2}	9.9158×10^{-6}	9.9978×10^{-8}
12	3.3434×10^{-2}	3.2308×10^{-2}	9.9158×10^{-6}	9.9981×10^{-8}

Table 1: The average relative errors of FIMs with TPZ, SIM and CBS in Example 3.1

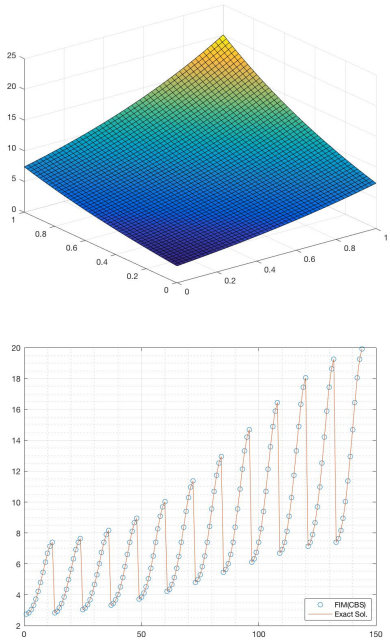


Figure 2: The surface and the grid points of the solutions at $t = 1$ in Example 3.1

Example 3.2. Consider a time-dependent linear PDE in which coefficients are the given functions in terms of x, y and t over domain $(x, y) \in (0, 2) \times (0, 2)$ and $t \in (0, 1)$.

$$\frac{\partial u}{\partial t} = x^2 e^t \frac{\partial^2 u}{\partial x^2} + y^2 e^t \frac{\partial^2 u}{\partial y^2} + x e^t \frac{\partial u}{\partial x} + y e^t \frac{\partial u}{\partial y} + u - (4x^2 e^{2t} + 4y^2 e^{2t})$$

with the initial and boundary conditions:

$$\begin{aligned}
&u(x, y, 0) = x^2 + y^2 + 1, \quad (x, y) \in [0, 2] \times [0, 2], \\
&u(x, 0, t) = e^t(x^2 + 1), \quad u(x, 1, t) = e^t(x^2 + 5), \quad x \in [0, 2], \quad t \in [0, 1], \\
&u(0, y, t) = e^t(y^2 + 1), \quad u(1, y, t) = e^t(y^2 + 5), \quad y \in [0, 2], \quad t \in [0, 1].
\end{aligned}$$

The analytical solution for this problem is $u^*(x, y, t) = e^t(x^2 + y^2 + 1)$. First, we transform our domain $\Omega = [0, 2] \times [0, 2]$ into $\overline{\Omega} = [-1, 1] \times [-1, 1]$ by using the transformations $\bar{x} = x - 1$ and $\bar{y} = y - 1$. From our numerical algorithm, we select the numbers of nodal points with respect to x- and y-axes which are the same numbers, i.e., $N_1 = N_2 = N$ for $N \in \{6, 8, 10, 12\}$. Thus, we can solve this problem by employing the linear system (3.10). Table 2 shows the average relative errors when $\tau \in \{0.1, 0.01\}$ comparing between our method and the traditional FIMs via SIM and SIM. We can see that the proposed method give higher accuracy than other method under the same $\tau = 0.1$. Finally, we also plot the graphs of numerical solutions at $t = 1$ in a surface and grid points forms as shown in Figure 3.

N	FIM-TPZ	FIM-SIM	Our proposed method	
	$\tau = 0.1$	$\tau = 0.1$	$\tau = 0.1$	$\tau = 0.01$
6	5.7281×10^{-2}	2.5104×10^{-2}	7.8415×10^{-5}	7.8566×10^{-7}
8	2.6256×10^{-2}	7.6268×10^{-2}	9.8155×10^{-5}	9.8348×10^{-7}
10	8.4489×10^{-3}	1.3237×10^{-2}	1.1148×10^{-4}	1.1170×10^{-6}
12	5.7623×10^{-3}	2.0794×10^{-2}	1.2101×10^{-4}	1.2122×10^{-6}

Table 2: The average relative errors of FIMs with TPZ, SIM and CBS in Example 3.2

Example 3.3. Consider a time-dependent differential equation in which coefficients are functions in terms of x, y and t. The forcing term involves trigonometric functions.

$$\begin{aligned}
\frac{\partial u}{\partial t} &= (x^2 + t^2)\frac{\partial^2 u}{\partial x^2} + (y^2 + t^2)\frac{\partial^2 u}{\partial y^2} + (2x + t)\frac{\partial u}{\partial x} + (2y + t)\frac{\partial u}{\partial y} \\
&\quad - (2x + 2y + 2t)t^2 \cos(x + y + 1) + [(x^2 + y^2 + 2t^2)t^2 + 2t] \sin(x + y + 1)
\end{aligned}$$

over the domain $(x, y) \in (0, 1) \times (0, 1)$ and $t \in (0, 1)$, with the initial and boundary

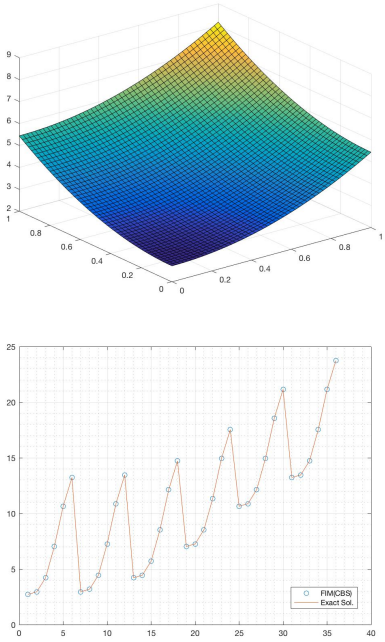


Figure 3: The surface and the grid points of the solutions at $t = 1$ in Example 3.2

conditions:

$$u(x, y, 0) = 0, \quad x \in [0, 1] \times [0, 1],$$
$$u(x, 0, t) = t^2 \sin(x + 1), \quad u(x, 1, t) = t^2 \sin(x + 2), \quad x \in [0, 1], \quad t \in [0, 1],$$
$$u(0, y, t) = t^2 \sin(y + 1), \quad u(1, y, t) = t^2 \sin(y + 2), \quad y \in [0, 1], \quad t \in [0, 1].$$

The analytical solution for this problem is $u^*(x, y, t) = t^2 \sin(x + y + 1)$. We first transform our domain $\Omega = [0, 1] \times [0, 1]$ by using the transformations $\bar{x} = 2x - 1$ and $\bar{y} = 2y - 1$. By our numerical algorithm, we choose the nodal numbers along x and y directions to be the same, that is $N_1 = N_2 = N$ for $N \in \{6, 8, 10, 12\}$. Hence, we can find the approximate solution at time $t = 1$ by solving the linear system (3.10). Moreover, we present the average relative errors of the approximate solution obtained by our modified method and the traditional FIMs with TPZ and SIM for $\tau \in \{0.1, 0.01\}$ in Table 3, together with display the graphically numerical solution at time $t = 1$ in Figure 4.

N	FIM-TPZ	FIM-SIM	Our proposed method	
	$\tau = 0.1$	$\tau = 0.1$	$\tau = 0.1$	$\tau = 0.01$
6	1.1343×10^{-2}	9.6989×10^{-3}	1.3795×10^{-6}	1.3795×10^{-6}
8	1.1191×10^{-2}	1.0837×10^{-2}	2.1107×10^{-9}	2.1092×10^{-9}
10	1.2355×10^{-2}	1.1652×10^{-2}	1.9190×10^{-12}	8.6832×10^{-12}
12	1.2684×10^{-2}	1.3024×10^{-2}	2.7685×10^{-12}	6.0370×10^{-12}

Table 3: The average relative errors of FIMs with TPZ, SIM and CBS in Example 3.3

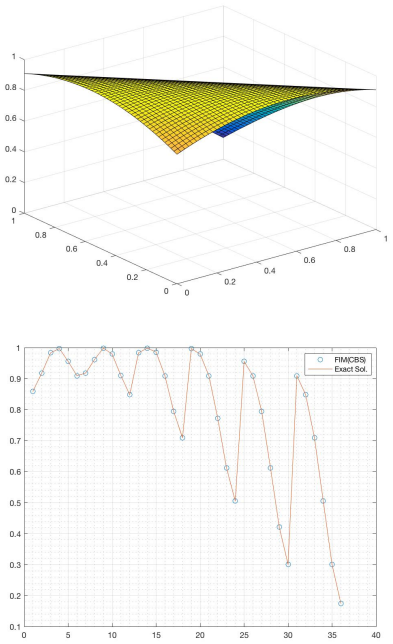


Figure 4: The surface and the grid points of the solutions at $t = 1$ in Example 3.3

From Examples 3.1-3.3, we can see that our proposed algorithm gives a lot better accurate results comparing to other traditional FIMs when using the same number of nodes. In addition, if we increase the time step, then we still obtain the better approximate solutions. We demonstrate the computational cost in terms of the CPU times(s) in Table 4. Also, we consider the use of memory storages for the implemented algorithm on MatLab software via determining the same accuracy, we can see that our proposed method uses smaller calculated matrix dimension than other traditional FIMs as shown in Table 5.

Example	$N=8$		$N=10$		$N=12$	
	$\tau=0.1$	$\tau=0.01$	$\tau=0.1$	$\tau=0.01$	$\tau=0.1$	$\tau=0.01$
3.1	0.056208	0.39353	0.086397	0.74078	0.15016	1.29235
3.2	0.057747	0.41505	0.091488	0.80091	0.14925	1.28501
3.3	0.050478	0.39359	0.078153	0.70575	0.13508	1.28016

Table 4: The CPU times(s) of our proposed method for Examples 3.1-3.3

Example	ARE	$\tau=0.05$			$N=0.025$		
		FIM-TPZ	FIM-SIM	FIM-CBS	FIM-TPZ	FIM-SIM	FIM-CBS
3.1	1.0×10^{-3}	9	6	4	9	6	4
3.2	5.0×10^{-3}	22	24	4	25	11	4
3.3	1.4×10^{-3}	15	8	4	8	8	4

Table 5: Dimension of matrix involved when considering the same accuracy

4 Numerical Solution of Linear Space-Fractional PDEs

Recently, the fractional differential equations can be found in various problems such as time delay problem, tautochrone problem, viscoelastic materials, fluid flow, diffusive transport, etc, see [10] and [9] for further details. In this section, we construct an algorithm based on the modified FIM applying shifted Chebyshev polynomials for finding approximate solutions of linear FDEs with Riemann-Liouville definition (2.5) for the fractional order derivative. To simplify our construction, let us consider the following linear FDE in one-dimensional space as

$$D^{\alpha}u(x)+a_2(x)u''(x)+a_1(x)u'(x)+a_0(x)u(x)=f(x) \quad \text{for} \quad x\in(0,L), \tag{4.1}$$

where $m\in\{1,2\}$ and $\alpha\in(m-1,m)$ with boundary conditions $u(0)=0$ and $u(L)=b\in\mathbb{R}$. Throughout this section, let us assume that the solution of (4.1) exists and unique. Moreover, for applying the FIM, we have to assume further that $\lim_{x\to0^+}u'(x)(-x)^{m-\alpha}$ exists.

First of all, we let $\beta=\alpha-m+1$ for $m\in\{1,2\}$, $\omega\in[0,1]$ and $F(x):=\int_0^x\frac{u(s)}{(x-s)^{\beta}}ds$ for $x\in(0,1)$. Next, we approximate the function $F(x)$ at each computational node $x_k\in[0,1]$ for $k\in\{1,2,3,...,M\}$, where we let the initial point

$x_0 = 0$. Then, we have

$$\begin{aligned} F(x_k) &= \int_0^{x_k} (x_k - s)^{-\beta} u(s)\, ds = \sum_{i=0}^{k-1} \int_{x_i}^{x_{i+1}} (x_k - s)^{-\beta} u(s)\, ds \\ &\approx \sum_{i=0}^{k-1} \int_{x_i}^{x_{i+1}} (x_k - s)^{-\beta} (\omega u(x_i) + (1-\omega) u(x_{i+1}))\, ds \\ &= \sum_{i=0}^{k-1} (\omega u(x_i) + (1-\omega) u(x_{i+1})) \left[\frac{(x_k - x_i)^{1-\beta}}{1-\beta} - \frac{(x_k - x_{i+1})^{1-\beta}}{1-\beta} \right]. \end{aligned}$$

By letting $q_{i,i+1}(x_k) = (x_k - x_i)^{1-\beta} - (x_k - x_{i+1})^{1-\beta}$, we have from the fact that $1 - \beta = m - \alpha$. Thus,

$$F(x_k) = \frac{1}{m-\alpha} \sum_{i=0}^{k-1} (\omega u(x_i) + (1-\omega) u(x_{i+1})) q_{i,i+1}(x_k). \tag{4.2}$$

We note that to obtain (4.2), we have used the approximation

$$\sum_{i=0}^{k-1} \int_{x_i}^{x_{i+1}} (x_k - s)^{-\beta} u(s)\, ds \approx \sum_{i=0}^{k-1} \int_{x_i}^{x_{i+1}} (x_k - s)^{-\beta} (\omega u(x_i) + (1-\omega) u(x_{i+1}))\, ds.$$

This approximation is suggested from the integration over $[x_i, x_{i+1}]$ using trapezoidal rule and the term should be $\frac{1}{2}u(x_i) + \frac{1}{2}u(x_{i+1})$. However, it will be shown later in Section 4.2 via numerical experiments as our conjecture that there are an optimal weights of $u(x_i)$ and $u(x_{i+1})$ rather than $\frac{1}{2}$, that our numerical algorithm obtain more accuracy.

4.1 Numerical Algorithm for Linear Space-Fractional PDEs

We are now ready to apply the FIM using the shifted Chebyshev polynomials to devise an algorithm for computing the approximate solution of (4.1) as the following procedures:

Step 1. Transform $x \in [0, L]$ into $\bar{x} \in [0, 1]$ by the transformation $\bar{x} = \frac{x}{L}$ and $\eta = \frac{s}{L}$, then (4.1) becomes

$$\frac{p^\alpha}{\Gamma(m-\alpha)} \frac{d^m}{d\bar{x}^m} \int_0^{\bar{x}} \frac{\bar{u}(\eta)}{(\bar{x} - \eta)^\beta} d\eta + p^2 \bar{a}_2(\bar{x}) \bar{u}''(\bar{x}) + p \bar{a}_1(\bar{x}) \bar{u}'(\bar{x}) + \bar{a}_0(\bar{x}) \bar{u}(\bar{x}) = \bar{f}(\bar{x}), \tag{4.3}$$

where $p = \frac{1}{L}$, $\bar{f}(\bar{x}) = f(L\bar{x})$, $\bar{a}_i(\bar{x}) = a_i(L\bar{x})$ for $i \in \{0, 1, 2\}$, $\bar{u}(\bar{x}) = u(L\bar{x})$ and $\bar{u}(0) = 0$ by assumption.

Step 2. Discretize the domain $[0,1]$ into M nodes by using zeros of $T_M^*(\bar{x})$ as defined in (2.2), i.e.,

$$\bar{x}_k = \frac{1}{2}\Big[\cos\Big(\frac{2k-1}{2M}\Big)\pi + 1\Big], \text{ where } k \in \{1,2,3,...,M\}.$$

Step 3. Eliminate all derivatives with respect to $\bar{x}$ by taking the double layer integration on both sides of (4.3) and using the integration by parts with integer order terms. Then, (4.3) becomes

$$\begin{aligned}
&\frac{p^\alpha}{\Gamma(m-\alpha)}\int_0^{\bar{x}_k}\int_0^{\xi 2}\frac{d^m}{d\xi_1^m}\int_0^{\xi_1}\frac{\bar{u}(\eta)}{(\xi_1-\eta)^\beta}\,d\eta\,d\xi_1 d\xi_2\\
&+p^2\bar{a}_2(\bar{x}_k)\bar{u}(\bar{x}_k)-2p^2\int_0^{\bar{x}_k}\bar{a}_2'(\xi_2)\bar{u}(\xi_2)\,d\xi_2+p^2\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{a}_2''(\xi_1)\bar{u}(\xi_1)\,d\xi_1 d\xi_2\\
&+p\int_0^{\bar{x}_k}\bar{a}_1(\xi_2)\bar{u}(\xi_2)\,d\xi_2-p\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{a}_1'(\xi_1)\bar{u}(\xi_1)\,d\xi_1 d\xi_2\\
&+\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{a}_0(\xi_2)\bar{u}(\xi_2)\,d\xi_1 d\xi_2+c_1\bar{x}_k+c_0=\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{f}(\xi_2)\,d\xi_1 d\xi_2, \qquad (4.4)
\end{aligned}$$

where c_0 and c_1 are arbitrary constants of integration. Next, we can divide (4.4) into 2 cases as follows:

Case 1: For $m = 1$, we can get rid of the first order derivative of the first term of (4.4) by integrating once with respect to ξ_1 from 0 to ξ_2. By using the assumption that $\lim_{x\to 0^+} u'(x)(-x)^{m-\alpha}$ exists, then we have $\lim_{\eta\to 0^+}\bar{u}'(\eta)(-\eta)^{1-\beta}$ exists and also substituting (4.2) into (4.4). Then, we get

$$\begin{aligned}
&\frac{p^\alpha}{\Gamma(1-\alpha)}\int_0^{\bar{x}_k}\frac{1}{1-\alpha}\sum_{i=0}^{k-1}(\omega\bar{u}(\bar{x}_i)+(1-\omega)\bar{u}(\bar{x}_{i+1}))q_{i,i+1}(\xi_2)\,d\xi_2\\
&+p^2\bar{a}_2(\bar{x}_k)\bar{u}(\bar{x}_k)-2p^2\int_0^{\bar{x}_k}\bar{a}_2'(\xi_2)\bar{u}(\xi_2)\,d\xi_2+p^2\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{a}_2''(\xi_1)\bar{u}(\xi_1)\,d\xi_1 d\xi_2\\
&+p\int_0^{\bar{x}_k}\bar{a}_1(\xi_2)\bar{u}(\xi_2)\,d\xi_2-p\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{a}_1'(\xi_1)\bar{u}(\xi_1)\,d\xi_1 d\xi_2\\
&+\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{a}_0(\xi_2)\bar{u}(\xi_2)\,d\xi_1 d\xi_2+c_1\bar{x}_k+c_0=\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{f}(\xi_2)\,d\xi_1 d\xi_2. \qquad (4.5)
\end{aligned}$$

Case 2: For $m = 2$, we can get rid of the second order derivative of the first term of (4.4) by integrating twice with respect to ξ_1 and ξ_2 from 0 to ξ_2 and 0 to $\bar{x}_k$, respectively. Then, we also substitute (4.2) into (4.4). Thus,

we obtain

$$\frac{p^\alpha}{\Gamma(2-\alpha)}\frac{1}{2-\alpha}\sum_{i=0}^{k-1}(\omega\bar{u}(\bar{x}_i)+(1-\omega)\bar{u}(\bar{x}_{i+1}))q_{i,i+1}(\bar{x}_k)$$
$$+p^2\bar{a}_2(\bar{x}_k)\bar{u}(\bar{x}_k)-2p^2\int_0^{\bar{x}_k}\bar{a}_2'(\xi_2)\bar{u}(\xi_2)\,d\xi_2+p^2\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{a}_2''(\xi_1)\bar{u}(\xi_1)\,d\xi_1d\xi_2$$
$$+p\int_0^{\bar{x}_k}\bar{a}_1(\xi_2)\bar{u}(\xi_2)\,d\xi_2-p\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{a}_1'(\xi_1)\bar{u}(\xi_1)\,d\xi_1d\xi_2$$
$$+\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{a}_0(\xi_2)\bar{u}(\xi_2)\,d\xi_1d\xi_2+c_1\bar{x}_k+c_0=\int_0^{\bar{x}_k}\int_0^{\xi_2}\bar{f}(\xi_2)\,d\xi_1d\xi_2. \quad (4.6)$$

Step 4. Transform (4.5) or (4.6) into the matrix form by using the idea in Section 2.4. First, we consider the summation term $\sum_{i=0}^{k-1}(\omega\bar{u}(\bar{x}_i)+(1-\omega)\bar{u}(\bar{x}_{i+1}))q_{i,i+1}(\bar{x}_k)$. For $k\in\{1,2,3,...,M\}$, we can transform them into the matrix form as $\mathbf{Qu}$, where $\mathbf{Q}=[Q_{ij}]_{M\times M}$ with

$$Q_{ij}=\begin{cases}(1-\omega)q_{i-1,i}(\bar{x}_i) & \text{if } i=j,\\ (1-\omega)q_{i-1,j}(\bar{x}_i)+\omega q_{j,i}(\bar{x}_i) & \text{if } i>j,\\ 0 & \text{if } i<j,\end{cases} \quad (4.7)$$

and $\mathbf{u}=[\bar{u}(\bar{x}_1),\bar{u}(\bar{x}_2),\bar{u}(\bar{x}_3),...,\bar{u}(\bar{x}_M)]^\top$. By the first and second order integration matrices using the shifted Chebyshev polynomial described in Section 2.4 and (4.7), then (4.5) or (4.6) can be expressed as

$$\frac{p^\alpha}{\Gamma(m+1-\alpha)}(\mathbf{A}^*)^{2-m}\mathbf{Qu}+p^2\mathbf{B}_2^{(0)}\mathbf{u}-2p^2\mathbf{A}^*\mathbf{B}_2^{(1)}\mathbf{u}+p^2(\mathbf{A}^*)^2\mathbf{B}_2^{(2)}\mathbf{u}$$
$$+p\mathbf{A}^*\mathbf{B}_1^{(0)}\mathbf{u}-p(\mathbf{A}^*)^2\mathbf{B}_1^{(1)}\mathbf{u}+(\mathbf{A}^*)^2\mathbf{B}_0^{(0)}\mathbf{u}+c_1\mathbf{x}+c_0\mathbf{e}=(\mathbf{A}^*)^2\mathbf{f}.$$

where $\mathbf{A}^*$ is the integration matrix which uses the zeros of T_M^*, $\mathbf{x}=[\bar{x}_1,\bar{x}_2,\bar{x}_3,...,\bar{x}_M]^\top$, $\mathbf{e}=[1,1,1,..,1]^\top$, $\mathbf{f}=[\bar{f}(\bar{x}_1),\bar{f}(\bar{x}_2),...,\bar{f}(\bar{x}_M)]^\top$ and $\mathbf{B}_i^{(j)}=\mathrm{diag}(\bar{a}_i^{(j)}(\bar{x}_1),\bar{a}_i^{(j)}(\bar{x}_2),...,\bar{a}_i^{(j)}(\bar{x}_M))$ for $i,j\in\{0,1,2\}$. Next, let $\mathbf{K}=\frac{p^\alpha(\mathbf{A}^*)^{2-m}\mathbf{Q}}{\Gamma(m+1-\alpha)}+p^2\mathbf{B}_2^{(0)}-2p^2\mathbf{A}^*\mathbf{B}_2^{(1)}+p^2(\mathbf{A}^*)^2\mathbf{B}_2^{(2)}+p\mathbf{A}^*\mathbf{B}_1^{(0)}-p(\mathbf{A}^*)^2\mathbf{B}_1^{(1)}+(\mathbf{A}^*)^2\mathbf{B}_0^{(0)}$, then above equation of matrix for $m\in\{1,2\}$ and $\alpha\in(m-1,m)$ can be simplified in the form:

$$\mathbf{Ku}+c_1\mathbf{x}+c_0\mathbf{e}=(\mathbf{A}^*)^2\mathbf{f}, \quad (4.8)$$

Step 5. Consider the given boundary conditions which are $\bar{u}(0)=0$ and $\bar{u}(1)=b$. Then, we have

$$\bar{u}(0)=\sum_{n=0}^{M-1}c_nT_n^*(0):=\mathbf{t}_l\mathbf{c}=\mathbf{t}_l(\mathbf{T}^*)^{-1}\mathbf{u}, \quad (4.9)$$

$$\bar{u}(1)=\sum_{n=0}^{M-1}c_nT_n^*(1):=\mathbf{t}_r\mathbf{c}=\mathbf{t}_r(\mathbf{T}^*)^{-1}\mathbf{u}, \quad (4.10)$$

where $\mathbf{t}_l = [1, -1, 1, ..., (-1)^{M-1}]$ and $\mathbf{t}_r = [1, 1, 1, ..., 1]$.

Step 6. Construct the system of linear equation from (4.8), (4.9) and (4.10) as follows

$$\left[\begin{array}{c|cc} \mathbf{K} & \mathbf{x} & \mathbf{e} \\ \hline \mathbf{t}_l(\mathbf{T}^*)^{-1} & 0 & 0 \\ \mathbf{t}_r(\mathbf{T}^*)^{-1} & 0 & 0 \end{array}\right] \left[\begin{array}{c} \mathbf{u} \\ \hline c_1 \\ c_0 \end{array}\right] \left[\begin{array}{c} (\mathbf{A}^*)^2\mathbf{f} \\ \hline 0 \\ b \end{array}\right]. \tag{4.11}$$

Then, the approximate solution can be found by solving the linear system (4.11). To obtain the numerical solution $u(x)$ for $x \in [0, b]$, we use transformation $x = b\bar{x}$.

4.2 Numerical Examples for Linear Space-Fractional PDEs

We use our proposed method to find the approximate solutions of some linear FDEs. In each example, we use different errors based on the results from each corresponding paper that we would like to compare with. Furthermore, we employ the error $E = \frac{1}{M}\sum_{i=1}^{M} \left|\frac{u_i - u_i^*}{u_{\max}^*}\right|$, where u^* and u are the analytical and numerical solutions, respectively. Also, we show the consuming of CPU times(s) in each example.

Example 4.1. Consider a linear FDE with $\alpha \in (0, 1)$.

$$D^{\alpha}(u) + \frac{d^2u}{dx^2} + u = \frac{6x^{3-\alpha}}{\Gamma(4-\alpha)} + x^3 + 6x, \quad x \in (0, 1),$$

with the boundary conditions $u(0) = 0$ and $u(1) = 1$. The analytical solution is $u^*(x) = x^3$. By using our numerical algorithm, this problem can be written in the matrix form as $\mathbf{Ku} + c_1\mathbf{x} + c_0\mathbf{e} = (\mathbf{A}^*)^2\mathbf{f}$, where $\mathbf{K} = \frac{1}{\Gamma(2-\alpha)}\mathbf{A}^*\mathbf{Q} + \mathbf{I} + (\mathbf{A}^*)^2$. For the boundary conditions, we have $\mathbf{t}_l(\mathbf{T}^*)^{-1}\mathbf{u} = 0$ and $\mathbf{t}_r(\mathbf{T}^*)^{-1}\mathbf{u} = 1$. Hence, we can solve this problem by solving the linear system (4.11). Finally, we obtain the approximate solutions $u(x)$. Table 6 shows the errors E and the CPU times(s) for our modified FIM using the shifted Chebyshev polynomials when $M = 10$ with several values of α. For each values of α, we give the optimal ω that achieve the highest accuracy. Moreover, we also plot the graphs of the analytical and numerical solutions as shown in Figure 5a. Note that Table 6 demonstrates the optimal ω that give the best accuracy for each value of α.

Example 4.2. Consider a linear FDE with $\alpha \in (0, 1)$.

$$\begin{aligned} D^{\alpha}(u) + \frac{d^2u}{dx^2} + \frac{6}{5}\frac{du}{dx} + \frac{1}{5}u = &\frac{3.5x^{\frac{5}{2}-\alpha}}{\Gamma(3.5-\alpha)} + \frac{2x^{2-\alpha}}{\Gamma(3-\alpha)} + \frac{1}{5}x^{\frac{5}{2}} + \frac{1}{5}x^2 \\ &+ 3x^{\frac{3}{2}} + \frac{15}{4}x^{\frac{1}{2}} + \frac{12}{5}x + 2, \quad x \in (0, 1), \end{aligned}$$

α	The optimal ω	The error E	CPU time(s)
0.001	0.54	3.4276×10^{-5}	0.016188
0.100	0.53	3.3185×10^{-5}	0.018178
0.300	0.49	3.4051×10^{-5}	0.017099
0.500	0.43	1.3316×10^{-4}	0.016407
0.700	0.32	2.1657×10^{-4}	0.017680
0.900	0.14	1.8333×10^{-4}	0.016136
0.990	0.02	7.4989×10^{-5}	0.017969
0.999	0.00	3.4180×10^{-5}	0.017662

Table 6: The errors E and CPU times of our method for $M = 10$ in Example 4.1

with the boundary conditions $u(0) = 0$ and $u(1) = 2$. The exact solution is $u^*(x) = x^{\frac{5}{2}} + x^2$. By using our numerical algorithm, this problem can be written in the matrix form as $\mathbf{Ku} + c_1\mathbf{x} + c_0\mathbf{e} = (\mathbf{A}^*)^2\mathbf{f}$, where $\mathbf{K} = \frac{1}{\Gamma(2-\alpha)}\mathbf{A}^*\mathbf{Q} + \mathbf{I} + \frac{6}{5}\mathbf{A}^* + \frac{1}{5}(\mathbf{A}^*)^2$. For the boundary conditions, we have $\mathbf{t}_l(\mathbf{T}^*)^{-1}\mathbf{u} = 0$ and $\mathbf{t}_r(\mathbf{T}^*)^{-1}\mathbf{u} = 2$. Consequently, we can solve this problem by solving the linear system (4.11). Finally, we obtain the approximate solutions $u(x)$. Table 7 shows the errors E and the computational time(s) for our presented FIM using the shifted Chebyshev polynomials for $M = 10$ with the different values of α and ω. The graphs of the analytical and approximate solutions are shown in Figure 5b. We note here that Table 7 shows the optimal ω that give the best accuracy for each values of α.

α	The optimal ω	The error E	CPU time(s)
0.001	0.53	3.5582×10^{-5}	0.019852
0.100	0.52	4.0456×10^{-5}	0.017335
0.300	0.48	8.0875×10^{-5}	0.017734
0.500	0.43	1.6888×10^{-4}	0.019479
0.700	0.33	2.9007×10^{-4}	0.016535
0.900	0.15	2.4695×10^{-4}	0.019813
0.990	0.02	5.0977×10^{-5}	0.016576
0.999	0.00	2.3218×10^{-5}	0.019966

Table 7: The errors E and CPU times of our method for $M = 10$ in Example 4.2

Example 4.3. Consider a linear FDE with $\alpha \in (1, 2)$.

$$D^{\alpha}(u) + \frac{d^2u}{dx^2} + u = \frac{2x^{2-\alpha}}{\Gamma(3-\alpha)} + x^2 + 2, \quad x \in (0, 1),$$

with the boundary conditions $u(0) = 0$ and $u(1) = 1$. The analytical solution is $u^*(x) = x^2$. By using our numerical algorithm, this problem can be written in

the matrix form as $\mathbf{Ku} + c_1\mathbf{x} + c_0\mathbf{e} = (\mathbf{A}^*)^2\mathbf{f}$, where $\mathbf{K} = \frac{1}{\Gamma(3-\alpha)}\mathbf{Q} + \mathbf{I} + (\mathbf{A}^*)^2$. For the given boundary conditions, we have $\mathbf{t}_l(\mathbf{T}^*)^{-1}\mathbf{u} = 0$ and $\mathbf{t}_r(\mathbf{T}^*)^{-1}\mathbf{u} = 1$. Therefore, we can solve this problem by solving the linear system (4.11). Finally, we obtain the approximate solutions $u(x)$. For each α, we give the optimal ω that achieves the highest accuracy. Table 8 demonstrates the error E and the time consuming for our FIM using the shifted Chebyshev polynomials when $M = 10$ with different α and its optimal ω. Also, the graph of analytical and approximate solutions are shown in Figure 5c.

α	The optimal ω	The error E	CPU time(s)
1.001	0.50	1.5150×10^{-4}	0.018116
1.100	0.51	1.2287×10^{-4}	0.016660
1.300	0.52	1.3863×10^{-3}	0.016202
1.500	0.25	1.8318×10^{-3}	0.017471
1.700	0.14	1.5271×10^{-3}	0.017206
1.900	0.05	6.5496×10^{-4}	0.017990
1.990	0.00	1.4920×10^{-4}	0.018570
1.999	0.00	2.2748×10^{-4}	0.017631

Table 8: The errors E and CPU times of our method for $M = 10$ in Example 4.3

Example 4.4. Consider a linear FDE with $\alpha \in (1, 2)$.

$$D^\alpha(u) + \frac{d^2u}{dx^2} + 2\frac{du}{dx} - u = \frac{6x^{3-\alpha}}{\Gamma(4-\alpha)} + \frac{2x^{2-\alpha}}{\Gamma(3-\alpha)} - x^3 + 5x^2 + 10x + 2, \quad x \in (0, 1),$$

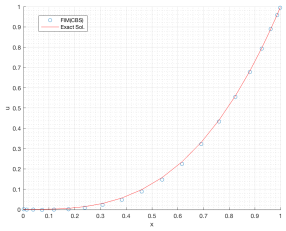
with the boundary conditions $u(0) = 0$ and $u(1) = 2$. The exact solution is $u^*(x) = x^3 + x^2$. By using our numerical algorithm, this problem can be written in the matrix form as $\mathbf{Ku} + c_1\mathbf{x} + c_0\mathbf{e} = (\mathbf{A}^*)^2\mathbf{f}$, where $\mathbf{K} = \frac{1}{\Gamma(3-\alpha)}(\mathbf{A}^*)\mathbf{Q} + \mathbf{I} + 2(\mathbf{A}^*) - (\mathbf{A}^*)^2$. For the boundary conditions, we have $\mathbf{t}_l(\mathbf{T}^*)^{-1}\mathbf{u} = 0$ and $\mathbf{t}_r(\mathbf{T}^*)^{-1}\mathbf{u} = 2$. Therefore, we can solve this problem by solving the linear system (4.11). Table 9 shows the errors E and the calculating times(s) for our FIM using the shifted Chebyshev polynomials when $M = 10$ with different α and ω. Also, the graph of analytical and approximate solutions are shown in Figure 5d. Note that Table 9 are the optimal ω that give the best accuracy.

4.3 Further Observation

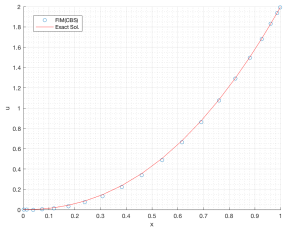
From Examples 4.1 and 4.2, for $0 < \alpha < 1$, we can see a relationship between α and the optimal ω that gives the best accuracy as shown in Figure 6a. Similarly, for $1 < \alpha < 2$, Figure 6b shows the relationship between α and the optimal ω

α	The optimal ω	The error E	CPU time(s)
1.001	0.51	1.5010×10^{-4}	0.020451
1.100	0.52	9.5674×10^{-5}	0.017865
1.300	0.54	1.2643×10^{-3}	0.018648
1.500	0.25	2.1052×10^{-3}	0.018468
1.700	0.14	1.9663×10^{-3}	0.016513
1.900	0.04	9.3191×10^{-3}	0.018311
1.990	0.00	1.4630×10^{-4}	0.019721
1.999	0.00	1.4758×10^{-5}	0.017167

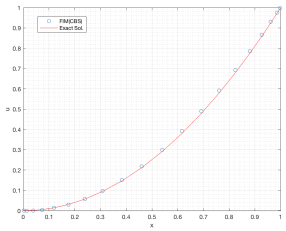
Table 9: The errors E and CPU times of our method for $M = 10$ in Example 4.4



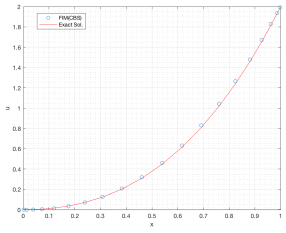
(a) Example 4.1 with $\alpha = 0.5$



(b) Example 4.2 with $\alpha = 0.5$



(c) Example 4.3 with $\alpha = 1.5$



(d) Example 4.4 with $\alpha = 1.5$

Figure 5: The graphs of the exact and numerical solutions for Examples 4.1-4.4

for Examples 4.3 and 4.4. Therefore, we automatically choose ω according to the polynomial interpolation that interpolates the average of ω from each to examples with respect to the values of α. Thus, we add Step 0 for choosing the value ω in our numerical algorithm from Section 4.1 as follow:

Step 0. Choose the value ω which divides into 2 cases for $\alpha \in (0, 1)$ or $\alpha \in (1, 2)$ as the following.

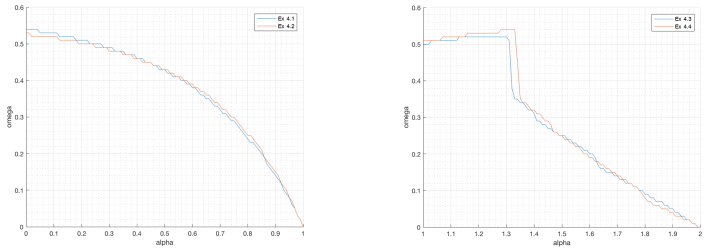
- For $\alpha \in (0,1)$,

$$\omega = -0.6238\alpha^3 + 0.3120\alpha^2 - 0.2175\alpha + 0.5397.$$

- For $\alpha \in (1,2)$,

$$\omega = \begin{cases} 0.0775\alpha + 0.4298 & \text{if} \quad 1 \leq \alpha \leq 1.31, \\ -0.5643\alpha + 1.1080 & \text{if} \quad 1.31 < \alpha \leq 2. \end{cases}$$

However, if the value ω obtained from these formulas is less than 0, then we take $\omega = 0$. The next two examples verify our suggestion about the optimal ω.



(a) Examples 4.1 and 4.2 (b) Examples 4.3 and 4.4

Figure 6: The optimal ω versus α for $M = 10$ in Examples 4.1-4.4

Example 4.5. Consider a linear FDE with $\alpha \in (0,1)$.

$$D^{\alpha}(u) + \frac{d^2u}{dx^2} + \frac{du}{dx} - u = \frac{24x^{4-\alpha}}{\Gamma(5-\alpha)} + \frac{2x^{2-\alpha}}{\Gamma(3-\alpha)} + 4x^3 + 11x^2 + 2x - x^4 + 2, \quad x \in (0,1),$$

with boundary conditions $u(0) = 0$ and $u(1) = 2$. The exact solution is $u^*(x) = x^4 + x^2$. By using our numerical algorithm with Step 0, we can see that our suggested ω is a good approximation for the optimal ω of this problem as shown in Table 10 and Figure 7a.

α	The optimal ω		The suggested ω	
	ω	E	ω	E
0.001	0.54	3.4169×10^{-5}	0.5395	3.0160×10^{-5}
0.010	0.53	3.8895×10^{-5}	0.5376	2.4006×10^{-5}
0.100	0.52	3.9675×10^{-5}	0.5204	3.3041×10^{-5}
0.300	0.49	4.2996×10^{-5}	0.4857	5.6484×10^{-5}
0.500	0.43	1.2478×10^{-4}	0.4310	1.2458×10^{-4}
0.700	0.33	2.1812×10^{-4}	0.3264	2.1634×10^{-4}
0.900	0.14	1.9063×10^{-4}	0.1419	1.8140×10^{-4}
0.990	0.02	5.1359×10^{-5}	0.0249	1.1259×10^{-4}
0.999	0.00	2.6051×10^{-5}	0.0119	1.4497×10^{-4}

Table 10: The errors E of our proposed method for $M = 10$ in Example 4.5

Example 4.6. Consider a linear FDE with $\alpha \in (1,2)$.

$$D^{\alpha}(u) + \frac{d^2u}{dx^2} + 2u = \frac{6x^{3-\alpha}}{\Gamma(4-\alpha)} + \frac{2x^{2-\alpha}}{\Gamma(3-\alpha)} + \frac{x^{1-\alpha}}{\Gamma(2-\alpha)} + 2x^3 + 2x^2 + 8x + 2, \quad x \in (0,1),$$

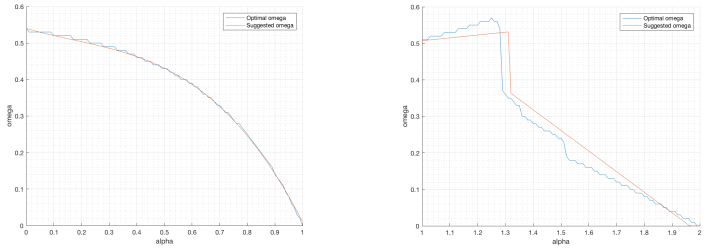
with boundary conditions $u(0) = 0$ and $u(1) = 3$. The exact solution is $u^*(x) = x^3 + x^2 + x$. By using our numerical algorithm with Step 0, we can see that our suggested ω is a good approximation for the optimal ω of this problem as shown in Table 11 and Figure 7b.

5 Conclusion and Discussion

In this paper, we propose an numerical algorithm for solving two-dimensional linear time-dependent differential equations based on the FIM using Chebyshev polynomials. As we know that all traditional FIMs using trapezoidal and Simpson's rules gave a better result than those solved by the FDM. Hence, each example

α	The optimal ω		The suggested ω	
	ω	E	ω	E
1.001	0.51	9.6872×10^{-5}	0.50738	1.0872×10^{-5}
1.010	0.51	8.6439×10^{-5}	0.50808	9.5303×10^{-5}
1.100	0.53	1.6900×10^{-4}	0.48727	4.6312×10^{-4}
1.300	0.26	1.5695×10^{-3}	0.37441	1.5738×10^{-3}
1.500	0.24	1.6789×10^{-3}	0.26155	1.7960×10^{-3}
1.700	0.12	1.1829×10^{-3}	0.14869	1.2926×10^{-3}
1.900	0.04	4.2573×10^{-4}	0.03538	4.4215×10^{-4}
1.990	0.00	1.1761×10^{-4}	0.00000	1.1761×10^{-4}
1.999	0.00	1.1957×10^{-5}	0.00000	1.1957×10^{-5}

Table 11: The errors E of our proposed method for $M = 10$ in Example 4.6



(a) Example 4.5 (b) Example 4.6

Figure 7: The optimal ω and suggested ω for $M = 10$ in Examples 4.5 and 4.6

in this paper, we compare our result with the traditional FIMs only. It turns out that our proposed method give a better accuracy for a large time step and give a lot better accuracy for a small time step. We can see that to obtain the same accuracy, the traditional FIMs need to calculate on a larger matrix dimension, thus, consuming more computational times than our proposed algorithm. However, the same idea can be extended to the n-dimensional linear time-dependent PDEs of higher order as well.

For the linear space-fractional differential equations, we obtain a preliminary result in terms of having an algorithm to find the numerical solutions for second order linear FDEs. The results demonstrate that for $\alpha \in (0, 1)$ and $\alpha \in (1, 2)$, then our method works very well. We also notice that in terms of the procedure, our numerical algorithm can be extended to solve FDEs of higher order.

For our future research, we try to prove that there is the optimal ω that makes good accuracy when we find approximate solutions of second order linear FDEs in the algorithm of Section 4.1. We also hope to propose algorithms based on our modified FIM using Chebyshev polynomial expansion to overcome nonlinear differential equations and the time-space fractional multi-order derivatives of dif-

ferential equations in Riemann-Liouville sense.

Acknowledgement : The authors would like to thank to the anonymous referees for their valuable comments and suggestions on this paper.

References

[1] P.H. Wen, Y.C. Hon, M. Li, T. Korakianitis, Finite integration method for partial differential equations, Appl. Math. Model. 37 (2013) 10092–10106.

[2] M. Li, C.S. Chen, Y.C. Hon, P.H. Wen, Finite integration method for solving multi-dimensional partial differential equations, Appl. Math. Model. 39 (2015) 4979–4994.

[3] M. Li, Z.L. Tian, Y.C. Hon, C.S. Chen, P.H. Wen, Improved finite integration method for partial differential equations, Eng. Anal. Boundary Elem. 64 (2016) 230–236.

[4] R. Boonklurb, A. Duangpan, T. Treeyaprasert, Modified Finite Integration Method Using Chebyshev Polynomial for Solving Linear Differential Equations, J. Numer. Ind. Appl. Math. 12 (3-4) (2018) 1–19.

[5] A. Saengsiritongchai, R. Boonklurb, Finite integration method using Chebyshev polynomials for solving linear time-dependent differential equations, Proceedings of the 23^{rd} Annual Meeting in Mathematics (2018) 232–238.

[6] W. Cheney, D. Kincaid, Numerical Mathematics and Computing, Cengage Learning, 2013.

[7] J.P. Boyd, Chebyshev and Fourier Spectral Methods, (2nd ed.), DOVER Publishing, 2000.

[8] R. Khalil, M. All Horani, A. Yousef, M. Sababheh, A new definition of fractional derivative, J. Comput. Appl. Math. 264 (2014) 65–70.

[9] M. Rahimy, Applications of Fractional Differential Equations, Appl. Math. Sci. 50 (2010) 2453–2461.

[10] R.M. Evans, U.N. Katugampola, D.A. Edwards, Applications of fractional calculus in solving Abel-type integral equations: Surface–volume reaction problem, Comput. Math. Appl. 73 (6) (2017) 1346–1362.

(Received 18 May 2019)
(Accepted 24 December 2019)