


Research Article

Fast Two-Grid Finite Element Algorithm for a Fractional Klein-Gordon Equation

Jingwei Jia^{1,2}, Nian Wang¹, Yang Liu^{1*} , Hong Li¹

¹School of Mathematical Sciences, Inner Mongolia University, Hohhot, China

²Department of Mathematics, Catholic University of Leuven, Leuven, Belgium

Email: mathliuyang@imu.edu.cn

Received: 4 December 2023; **Revised:** 20 March 2024; **Accepted:** 27 March 2024

Abstract: In this article, we propose a spatial two-grid finite element algorithm combined with a shifted convolution quadrature (SCQ) formula for solving the fractional Klein-Gordon equation. The time direction at $t_{n-\theta}$ is approximated utilizing a second-order SCQ formula, where θ is an arbitrary constant. The spatial discretization is performed using a two-grid finite element method involving three steps: calculating the numerical solution by solving a nonlinear system iteratively on the coarse grid, obtaining the interpolation solution based on the computed solutions in the first step, and solving a linear finite element system on the fine grid. We present a numerical algorithm, validate the two-grid finite element method's effectiveness, and demonstrate the computational efficiency for our method by the comparison of the computing results between the two-grid finite element method and the standard finite element method.

Keywords: Fractional Klein-Gordon equation, SCQ scheme, spatial two-grid finite element method

MSC: 65N30, 65M60

1. Introduction

Fractional Klein-Gordon equation which extends the classical Klein-Gordon equation [1-3], is renowned for its role in field theory and relativistic quantum mechanics. Initially it was designed to explain phenomena like dislocation propagation in crystals and the behavior of elementary particles [4]. At present, it has formed a versatile framework with broader applications. In quantum mechanics, it reveals insights for the fundamental particle behavior and contributes to cutting-edge areas such as soliton exploration and condensed matter physics [5-6]. Essentially, these equations serve as powerful tools, connecting theoretical constructs with tangible phenomena in particle physics, quantum mechanics, and condensed matter physics.

In this article, we explore the numerical solution of the time fractional Klein-Gordon equation (TFKGE)

$${}_0^R D_t^\alpha u = \Delta u - f(u) + g(z, t), (z, t) \in \Omega \times (0, T], \quad (1)$$

$$u(z, t) = \beta(z, t), z \in \partial\Omega, t \in (0, T], \quad (2)$$

$$u(z, 0) = 0, u_t(z, 0) = 0, z \in \Omega \cup \partial\Omega, \quad (3)$$

where $\alpha \in (1, 2]$. $f(u)$ denotes the nonlinear term and $g(z, t)$ stands as the source term. $\beta(z, t)$ signifies the value of $u(z, t)$ on the boundary. It is noteworthy that our calculations are conducted within a rectangular domain. This rectangular domain Ω could be defined as $(0, L) \times (0, L)$, where L denotes the side length of the rectangular domain. ${}^R_0 D_t^\alpha u$ is the time-fractional derivative defined by

$${}^R_0 D_t^\alpha u = \frac{1}{\Gamma(2-\alpha)} \frac{\partial^2}{\partial t^2} \int_0^t \frac{u(z, s)}{(t-s)^{\alpha-1}} ds, \quad \alpha \in (1, 2]. \quad (4)$$

The recent attention has been directed towards handling the non-linear term $f(u)$ of Klein-Gordon equation. In the pursuit of computational efficiency, researchers have put forth linearized schemes. Dehghan et al. [7] explored the implicit radial basis function meshless collocation method. Zhang and Jiang [8] devised a Crank-Nicolson Legendre spectral approach. These two methods both demonstrate the first-order convergence in time. Besides, Lyu and Vong [9] introduced an intriguing linearized finite difference scheme and proved that the fully discrete scheme converges with second-order accuracy in time. In [10], Zhang et al. developed the Galerkin finite element approach with a linearized high-order time discrete scheme for two-dimensional nonlinear TFKGEs. However, we rarely see relevant research on two-grid finite element methods for nonlinear time fractional Klein-Gordon equations.

Based on the existing research results [11-17], we notice that the two-grid finite element method (T-GFEM) has significant advantages in improving computational speed when solving nonlinear partial differential equation (PDE) models. Xu [11] first presented the T-GFEM for solving a large class of nonlinear elliptic boundary value problems and derived optimal L^p error estimates. Wu and Allen [12] used the T-GFEM solving two-dimensional nonlinear reaction-diffusion equations. In [13], Chen et al. designed T-G expanded mixed FEM to solve semilinear reaction-diffusion models. In [14], Shi and Yang proposed the T-GFEM for a semilinear parabolic equation and showed unconditional optimal error analyses. In [15-17], some scholars developed different T-GFEMs to quickly solve nonlinear time-fractional PDEs.

In this article, the T-GFEM combined with an SCQ formula [18-19] is developed to quickly solve the nonlinear TFKGE. The contributions of our work are as follows: The detailed computing process of our algorithm is provided, by which one can easily understand how to implement the computing; Compared to general approximation methods including Crank-Nicolson scheme and BDF2, the second-order time discrete scheme generated by the SCQ formula can be used at any time point $t = t_{n-\theta}$; Several numerical experiments have shown that compared to the standard FEM, our T-GFEM significantly improves computational efficiency while maintaining computational accuracy.

The upcoming sections of this article are organized as follows: In Section 2, we introduce an SCQ formula and subsequently present the time semi-discrete scheme. In Section 3, we present the main computing steps of the T-GFEM. In Section 4, we show the fully discrete scheme and the corresponding algorithm for solving the TFKGE. In Section 5, we involve three numerical examples to verify the efficiency of our approach. In the last section, we summarize the results obtained in this article and offer insights into future directions that warrant further exploration.

2. Time discretization

To derive the discrete scheme, we commence by subdividing the time interval into a uniform partition: $0 = t_0 < t_1 < t_2 < t_3 \dots < t_N = T$, where $t_n = n\Delta t$ and $\Delta t = \frac{T}{N}$. To simplify notation, we denote $u(t_n)$ as u^n . Subsequently, we introduce an SCQ formula [19] featuring a shifted parameter θ

$${}^R_0 D_t^\alpha u^{n-\theta} = \Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u^{n-k} + O(\Delta t^2), \quad t_{n-\theta} = (n-\theta)\Delta t, \quad n = 2, 3, \dots, N, \quad (5)$$

where the weights $w_k^{(\theta)}$ are coefficients of the generating function $w_k^{(\theta)}(\xi)$:

$$w_k^{(\theta)}(\xi) = \sum_{k=0}^{\infty} w_k^{(\theta)} \xi^k = \left[\frac{1-\xi}{\frac{1}{2}(1+\xi) + \frac{\theta}{\alpha}(1-\xi)} \right]^\alpha, \quad (6)$$

from which we can also obtain the recursive formula

$$w_k^{(\theta)} = \begin{cases} \left(\frac{2\alpha}{\alpha+2\theta}\right)^\alpha, & k=0, \\ -\alpha\left(\frac{2\alpha}{\alpha+2\theta}\right)^{\alpha+1}, & k=1, \\ \frac{2\alpha}{k(\alpha+2\theta)} \left\{ \left[\frac{2\theta}{\alpha}(k-1) - \alpha \right] w_{k-1}^{(\theta)} + \frac{\alpha-2\theta}{2\alpha}(k-2)w_{k-2}^{(\theta)} \right\}, & k \geq 2. \end{cases} \quad (7)$$

When dealing with the term $f(u^{n-\theta})$, we employ the following expression

$$f(u^{n-\theta}) = f^{n-\theta} + O(\Delta t^2), \quad (8)$$

where

$$f^{n-\theta} = (1-\theta)f(u^n) + \theta f(u^{n-1}). \quad (9)$$

For a function $u(t)$ that is sufficiently smooth, we have

$$u(t_{n-\theta}) = u^{n-\theta} + O(\Delta t^2), \quad (10)$$

where

$$u^{n-\theta} = (1-\theta)u^n + \theta u^{n-1}. \quad (11)$$

Following this, we proceed to implement our time discretization approach by incorporating the SCQ formula (5), (8) and (10), and find $u^n : [0, T] \mapsto H^1$ to satisfy

$$(\Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u^{n-k}, v) + (1-\theta)(\nabla u^n + f(u^n), v) + \theta(\nabla u^{n-1} + f(u^{n-1}), v) = g(z, t_{n-\theta}), \quad \forall v \in H_0^1. \quad (12)$$

3. Two-grid finite element method

To further provide the fully discrete two-grid finite element scheme, we define the following finite element space

$$\begin{aligned} U_h &:= \{u_h \in H^1(\Omega) \mid u_h|_e \in P_k(K_e), \forall K_e \in \mathcal{K}_h\}, \\ V_h &:= \{v_h \in H_0^1(\Omega) \mid v_h|_e \in P_k(K_e), \forall K_e \in \mathcal{K}_h\}, \end{aligned} \quad (13)$$

where $\mathcal{h} = h$ or H , U_H, V_H and U_h, V_h are subspaces composed of polynomial functions about space subdivision \mathcal{K}_H and \mathcal{K}_h , respectively, satisfying $U_H \subset U_h \subset H^1$, $V_H \subset V_h \subset H_0^1$, and P_k are polynomial functions with degree not exceeding k .

Then, we split the T-GFEM into two essential steps.

3.1 Step I: Nonlinear iteration on the coarse grid

Following the insights presented in [12], the nonlinear iteration on the coarse grid is a requisite step. Let u_H^n represent the numerical solution on the coarse grid at time layer n . Then, (1) is rewritten as

$$\Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_H^{n-k} = \Delta u_H^{n-\theta} - f_H^{n-\theta} + g^{n-\theta}. \quad (14)$$

Besides, we have

$$\begin{aligned} f(u_H^{n,q}) &= f(u_H^{n-1,Q}), \quad q = 1, \\ f(u_H^{n,q}) &= f(u_H^{n,q-1}), \quad q = 2, 3, \dots, Q, \end{aligned} \quad (15)$$

where Q represents the number of nonlinear iterations. The index q ranges from 1 to Q . When $n = 1$, $f(u_H^{n-1,Q})$ is initialized as $f(u_H^0)$, which is evidently a known quantity. Subsequently, substituting (15) into (14), we can obtain the following corrective scheme

$$\begin{aligned} \Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_H^{n-k} &= \Delta u_H^{n-\theta} - f(u_H^{n-1,Q}) + g^{n-\theta}, \quad q = 1, \\ \Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_H^{n-k} &= \Delta u_H^{n-\theta} - f(u_H^{n,q-1}) + g^{n-\theta}, \quad q = 2, 3, \dots, Q. \end{aligned} \quad (16)$$

With the above assignment, we transform (14), which is characterized by substantial computational demands, into (16).

Therefore, we obtain the fully discrete scheme on the coarse grid, which is to find $u_H^n : [0, T] \mapsto U_H$ that satisfies

$$\begin{aligned} &(\Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_H^{n-k}, v_H) + (\theta \nabla u_H^{n-1} + (1-\theta) \nabla u_H^n, \nabla v_H) \\ &= (g^{n-\theta} - f(u_H^{n-1,Q}), v_H), \quad q = 1, \quad \forall v_H \in V_H, \\ &(\Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_H^{n-k}, v_H) + (\theta \nabla u_H^{n-1} + (1-\theta) \nabla u_H^n, \nabla v_H) \\ &= (g^{n-\theta} - (1-\theta)f(u_H^{n,q-1}) - \theta f(u_H^{n-1,Q}), v_H), \quad q = 2, 3, \dots, Q, \quad \forall v_H \in V_H. \end{aligned} \quad (17)$$

By solving (17), we deduce the value of u_H on the coarse grid.

3.2 Step II: Correction on the fine grid

Prior to the correction process, we divide the space interval in the x -direction uniformly: $0 = x_0 < x_1 < x_2 < x_3 \dots < x_M = L$ with $x_m = mh$ where $h = \frac{L}{M}$, and for the y -direction, we implement the similar process. In (17), we compute u_H on the coarse grid, followed by solving on the fine grid, which needs interpolation on the coarse grid. For instance, Figure 1 illustrates the distinction between coarse and fine grids. On the coarse grid, represented by the red triangle, an element with $H = \frac{1}{2}$ is employed. Conversely, the fine grid employs an element with $h = \frac{1}{4}$ obtained through interpolation from the coarse grid, depicted by the blue triangle.

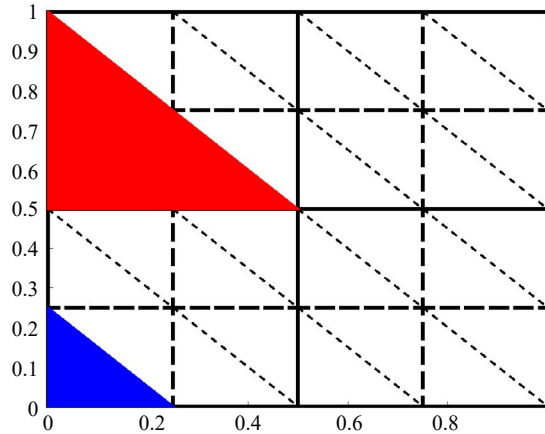


Figure 1. Coarse grid with $H = \frac{1}{2}$ and fine grid with $h = \frac{1}{4}$

In Figure 1, we begin by computing the numerical solutions at the three vertices of the red triangular elements on the coarse grid. Subsequently, we employ an interpolation technique to obtain the numerical solutions at the three vertices of the blue triangular elements on the fine grid. The interpolation formula is

$$u_{jH+\lambda H} = u_{jH} + \lambda(u_{(j+1)H} - u_{jH}), \quad (18)$$

where $u_{jH+\lambda H}$ is the numerical solution on the fine grid. Simultaneously, the imposition of the boundary conditions establishes known numerical solutions along the boundary. Following a process of linear interpolation, we subsequently adjust the solution along the boundary utilizing $\beta(x, t)$.

Now, we derive the approximate solution u_h on the fine grid through the interpolation process. However, it is essential to acknowledge that this interpolation process introduces certain errors. To enhance the accuracy of u_h on the fine grid, the correction is necessary, particularly concerning the nonlinear term. To achieve this, we find $u_h^n : [0, T] \mapsto U_h$ that satisfies

$$(\Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_h^{n-k}, v_h) + (\nabla u_h^{n-\theta}, \nabla v_h) = (g^{n-\theta}, v_h) - (f'(u_H^{n-\theta})(u_h^{n-\theta} - u_H^{n-\theta}), v_h), \quad \forall v_h \in V_h, \quad (19)$$

where u_h^n is the numerical solution on the fine grid at time layer n . Upon solving (19), we acquire the numerical solution on the fine grid.

4. Specific algorithm

4.1 Fully discrete scheme

We introduce the discrete methods and effective strategies for handling nonlinear terms, referring to (12), (17) and (19). Then, we can establish the fully discrete scheme for (1).

Case on the coarse grid: find $u_H^n : [0, T] \mapsto U_H$ to satisfy

$$\begin{aligned} & (\Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_H^{n-k}, v_H) + ((1-\theta)\nabla u_H^n + \theta\nabla u_H^{n-1}, \nabla v_H) \\ & = (g^{n-\theta} - \theta f(u_H^{n-1}) - (1-\theta)f(u_H^{n-1, \theta}), v_H), \quad q=1, \quad \forall v_H \in V_H, \end{aligned} \quad (20)$$

$$\begin{aligned}
& (\Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_H^{n-k}, v_H) + ((1-\theta)\nabla u_H^n + \theta\nabla u_H^{n-1}, \nabla v_H) \\
& = (g^{n-\theta} - \theta f(u_H^{n-1}) - (1-\theta)f(u_H^{n,q-1}), v_H), \quad q = 2, 3, \dots, Q, \quad v_H \in V_H.
\end{aligned} \tag{21}$$

Case on the fine grid: find $u_h^n : [0, T] \mapsto U_h$ to satisfy

$$\begin{aligned}
& (\Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_h^{n-k}, v_h) + ((1-\theta)\nabla u_h^n + \theta\nabla u_h^{n-1}, \nabla v_h) \\
& = (g^{n-\theta}, v_h) - (((1-\theta)f'(u_H^n) + \theta f'(u_H^{n-1}))((1-\theta)\nabla u_h^n + \theta\nabla u_h^{n-1} - (1-\theta)u_H^n - \theta u_H^{n-1}), v_h), \quad \forall v_h \in V_h,
\end{aligned} \tag{22}$$

where Q represents a constant value, indicating a consistent number of nonlinear iterations maintained throughout the specified time horizon.

4.2 Algorithm

Contemplating the solution domain of the algorithm, we narrow our attention exclusively to the model built on triangular elements, as depicted in Figure 2.

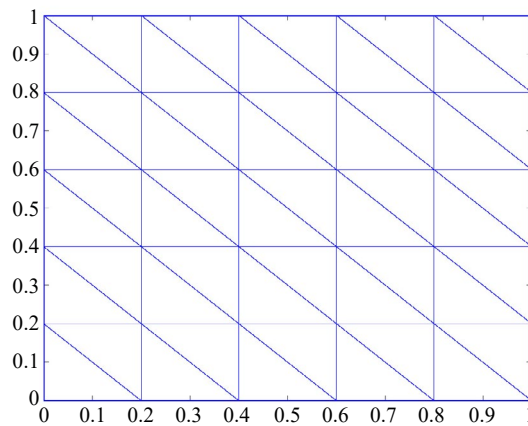


Figure 2. Triangular element

We employ a triangular element as a demonstrative example to expedite subsequent calculations. Assume that the coordinates of the three vertices, denoted as P_i, P_j, P_m , are $(x_i, y_i), (x_j, y_j), (x_m, y_m)$, and that the values at these points are represented as u_i, u_j, u_m .

For each triangular element, we formulate three basis functions

$$N_i(x, y) = \frac{1}{2\Delta_e} \left[\begin{vmatrix} y_j & 1 \\ y_m & 1 \end{vmatrix} u_i - \begin{vmatrix} x_j & 1 \\ x_m & 1 \end{vmatrix} u_j + \begin{vmatrix} x_j & y_j \\ x_m & y_m \end{vmatrix} u_m \right], \tag{23}$$

which could be abbreviated as:

$$\begin{cases} N_i(x, y) = \frac{1}{2\Delta_e} (a_i x + b_i y + c_i), \\ a_i = \begin{vmatrix} y_j & 1 \\ y_m & 1 \end{vmatrix}, \quad b_i = \begin{vmatrix} x_j & 1 \\ x_m & 1 \end{vmatrix}, \quad c_i = \begin{vmatrix} x_j & y_j \\ x_m & y_m \end{vmatrix}. \end{cases} \tag{24}$$

The remaining two basis functions denoted by $N_i(x, y)$, $N_m(x, y)$, have similar expressions to $N_j(x, y)$. The only distinction lies in the computation of their coefficients a_j, b_j, c_j and a_m, b_m, c_m . These coefficients can be derived from the expressions of a_i, b_i, c_i , it is imperative to carefully reorganize the order of indices i, j, m in this process, as exemplified in Figure 3.

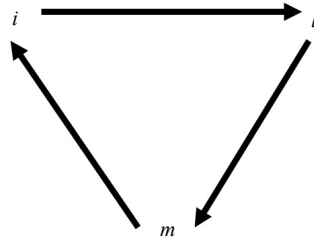


Figure 3. Assignment order

Afterward, we convert (23) and (24) into a matrix

$$\mathbf{B} = \frac{1}{2\Delta_e} \begin{bmatrix} a_i & a_j & a_m \\ b_i & b_j & b_m \end{bmatrix}, \quad (25)$$

and a three-dimensional column vector

$$\mathbf{N}(x, y) = [N_i(x, y) \quad N_j(x, y) \quad N_m(x, y)]^T. \quad (26)$$

Given the tripartite nature of this method, we shall construct the element stiffness matrices denoted as $\mathbf{A}^{(e)}$, as well as the element load vectors represented by $\mathbf{b}^{(e)}$, in accordance with each individual step.

Step 1 Case on the coarse grid where $q = 1$

$$\begin{aligned} \mathbf{A}^{(e)} &= \iint_{e_n} BB^T + w_0^{(\theta)} N(x, y)N(x, y)^T \, dx dy, \\ \mathbf{b}^{(e)} &= \iint_{e_n} \left\{ g^{n-\theta} - f(u_H^{n-1, \varrho}) - \Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_H^{n-k} \right\} N(x, y) \, dx dy. \end{aligned} \quad (27)$$

Step 2 Case on the coarse grid where $q = 2, 3, \dots, Q$:

$$\begin{aligned} \mathbf{A}^{(e)} &= \iint_{e_n} BB^T + w_0^{(\theta)} N(x, y)N(x, y)^T \, dx dy, \\ \mathbf{b}^{(e)} &= \iint_{e_n} \left\{ g^{n-\theta} - (1-\theta)f(u_H^{n, q-1}) - \theta f(u_H^{n-1, \varrho}) - \Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_H^{n-k} \right\} N(x, y) \, dx dy. \end{aligned} \quad (28)$$

Step 3 Case on the fine grid:

$$\begin{aligned} \mathbf{A}^{(e)} &= \iint_{e_n} BB^T + (w_0 + ((1-\theta)f'(u_H^n) + \theta f'(u_H^{n-1}))(1-\theta)) N(x, y)N(x, y)^T \, dx dy, \\ \mathbf{b}^{(e)} &= \iint_{e_n} \left\{ g^{n-\theta} + ((1-\theta)f'(u_H^n) + \theta f'(u_H^{n-1}))(\theta u_h^{n-1} - (1-\theta)u_H^n - \theta u_H^{n-1}) - \Delta t^{-\alpha} \sum_{k=0}^n w_k^{(\theta)} u_h^{n-k} \right\} N(x, y) \, dx dy. \end{aligned} \quad (29)$$

Once we acquire the method for constructing element stiffness matrix and element load vector, we have the capability to extend $\mathbf{A}^{(e)}$ into a high-order matrix

$$\mathbf{A}^{(e)} = \begin{bmatrix} \vdots & \vdots & \vdots & \vdots \\ \cdots a_{ii}^{(e)} \cdots a_{ij}^{(e)} \cdots a_{im}^{(e)} \cdots \\ \vdots & \vdots & \vdots & \vdots \\ \cdots a_{ji}^{(e)} \cdots a_{jj}^{(e)} \cdots a_{jm}^{(e)} \cdots \\ \vdots & \vdots & \vdots & \vdots \\ \cdots a_{mi}^{(e)} \cdots a_{mj}^{(e)} \cdots a_{mm}^{(e)} \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix},$$

and then we compute the global stiffness matrix based on $\mathbf{A}^{(e)}$ by

$$\mathbf{A} = \sum_e \mathbf{A}^{(e)}. \tag{30}$$

Similarly, we expand $\mathbf{b}^{(e)}$ to a high-dimensional vector

$$\mathbf{b}^{(e)} = [\cdots, b_i^{(e)}, \cdots, b_j^{(e)}, \cdots, b_m^{(e)}, \cdots]^T,$$

and the global load vector is derived from

$$\mathbf{b} = \sum_e \mathbf{b}^{(e)}. \tag{31}$$

Subsequently, we achieve the numerical solution by solving a system of linear equations

$$\mathbf{A}z = \mathbf{B}. \tag{32}$$

The specific execution steps of this algorithm are outlined in Table 1.

Table 1. T-GFEM algorithm

T-GFEM algorithm	
Step 1	Time partition and space partition on the coarse grid
Step 2	For $n = 1:N$
Step 3	Solve (20)
Step 4	For $q = 2:Q$
Step 5	Solve (21)
Step 6	End the loop and get u_H on the coarse grid at time layer n
Step 7	End the loop and get u_H on the coarse grid at all time layer
Step 8	Do linear interpolation
Step 9	For $n = 1:N$
Step 10	Solve the function on the fine grid at time layer n
Step 12	End the loop and get u_h on the fine grid at time layer n
Output	End the loop and get u_h on the fine grid at all time layer

4.3 Non-zero boundary

Now we consider a particular case in which the boundary conditions are non-zero. In this situation, we employ row and column transformations to recast (32) into the following form:

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \quad (33)$$

where Z_2 represents the known boundary $\beta(x, t)$, and Z_1 denotes the numerical solution. We employ block matrix multiplication to formulate a new equation that serves as a suitable framework for addressing non-zero boundary conditions

$$\begin{aligned} A_{11}Z_1 + A_{12}Z_2 &= b_1, \\ Z_1 &= A_{11}^{-1}(b_1 - A_{12}Z_2), \end{aligned} \quad (34)$$

where Z_1 represents the numerical solution subsequent to boundary treatment.

5. Numerical tests

In this section, we employ some numerical examples to validate the efficiency of our algorithm in comparison to the finite element method when solving the fractional Klein-Gordon model. Additionally, we utilize the following formula to demonstrate that our algorithm exhibits a convergence order of the same magnitude as that of the FEM. The convergence order is denoted as the following:

$$\text{Convergence order} = \log \frac{\max_n \|u^n - u_{h_1, \Delta t_1}^n\|}{\max_n \|u^n - u_{h_2, \Delta t_2}^n\|},$$

where $(h_1, \Delta t_1)$ and $(h_2, \Delta t_2)$ are the combinations of different partitions of space and time.

5.1 Example 1

We consider the following equation with the non-zero boundary

$$\begin{aligned} {}_0^R D_t^\alpha u &= \Delta u - u^2 + g(z, t), \quad (z, t) \in \Omega \times (0, T], \\ u(z, t) &= t^4 (\sin \pi x + \sin \pi y), \quad z \in \partial\Omega, \quad t \in (0, T], \\ u(z, 0) &= 0, \quad u_t(z, 0) = 0, \quad z \in \Omega \cup \partial\Omega, \end{aligned} \quad (35)$$

where $\Omega = (0, 1) \times (0, 1)$, $T = 1$, $z = (x, y)$ and $g(z, t) = \frac{\Gamma(5)}{\Gamma(5-\alpha)} t^{4-\alpha} (\sin \pi x + \sin \pi y) + t^4 \pi^2 (\sin \pi x + \sin \pi y) + t^8 (\sin \pi x + \sin \pi y)^2$. The exact solution of the above equation is

$$u(z, t) = t^4 (\sin \pi x + \sin \pi y), \quad (z, t) \in \Omega \times (0, 1], \quad (36)$$

and for the purpose of illustrating that the algorithm is universal for the parameter α , we conduct identical tests using various α values. In Tables 2-3, fixing $\Delta t = 1/1,000$, and changing $h = H^2 = \frac{1}{4}, \frac{1}{16}$ and $h = \frac{H}{2} = \frac{1}{4}, \frac{1}{8}, \frac{1}{16}$, respectively, we obtain five sets of numerical solutions computed with the $\alpha = 1.1, 1.3, 1.5, 1.7, 1.9$.

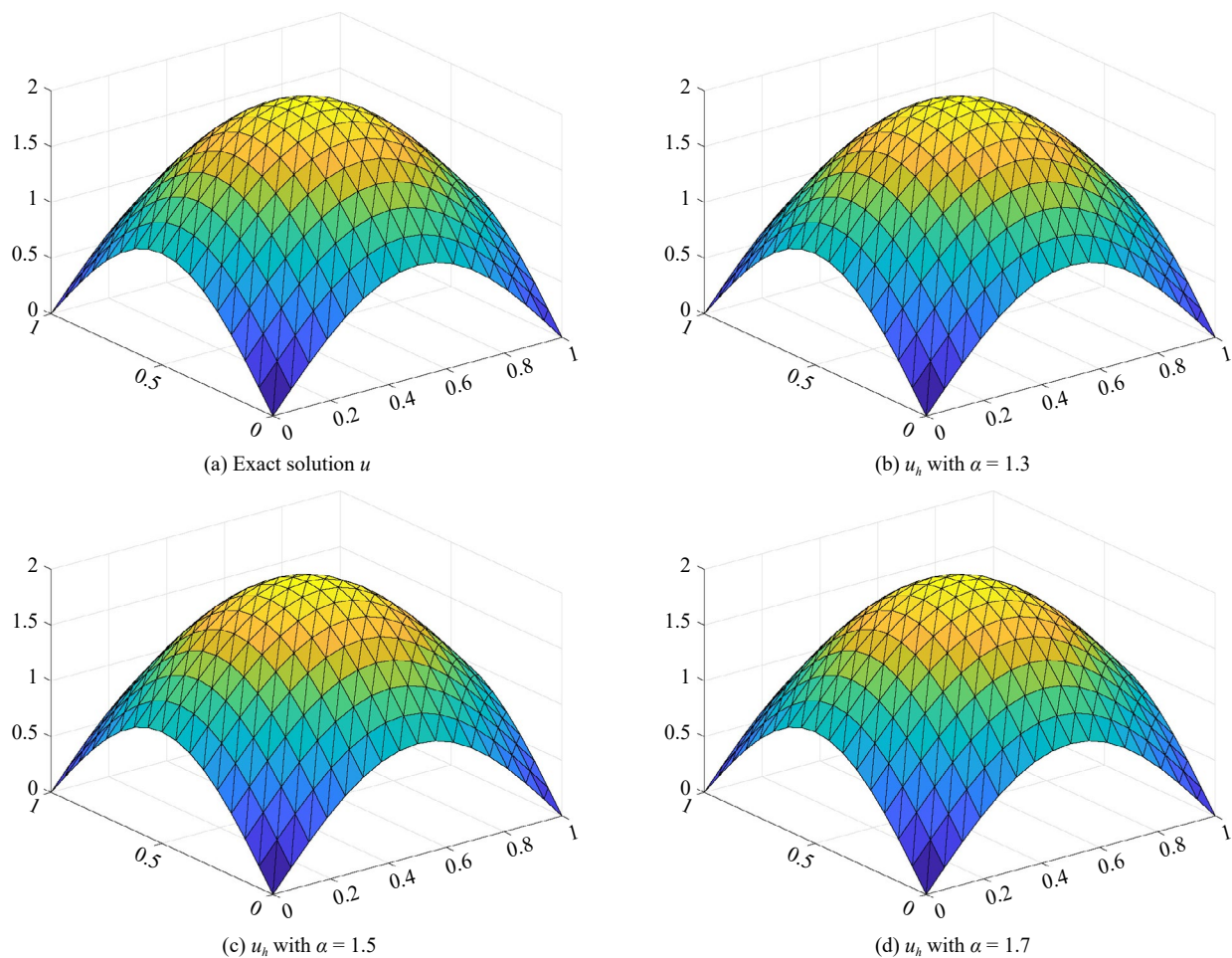


Figure 4. Comparison between exact solution u and numerical solution u_h for the T-GFEM with $h = H^2 = \frac{1}{16}$, $\Delta t = \frac{1}{1,000}$ and different α

Table 2. Space convergence results in L^2 -norm for FEM and T-GFEM

α	$h = H^2$	Δt	Error (FEM)	Order	CPU	Error (T-GFEM)	Order	CPU
1.1	1/4	1/1,000	2.3739E-02	-	53.1	2.4432E-02	-	32.6
1.1	1/16	1/1,000	1.5822E-03	1.9536	1,121.9	1.6469E-03	1.9455	445.5
1.3	1/4	1/1,000	3.0558E-02	-	52.6	2.7305E-02	-	32.8
1.3	1/16	1/1,000	1.7728E-03	2.0537	1,209.7	1.8241E-03	1.9520	460.0
1.5	1/4	1/1,000	3.0557E-02	-	52.2	3.1020E-02	-	34.1
1.5	1/16	1/1,000	2.0224E-03	1.9587	1,130.2	2.0591E-03	1.9566	455.6
1.7	1/4	1/1,000	3.5874E-02	-	54.6	3.6211E-02	-	33.9
1.7	1/16	1/1,000	2.3704E-03	1.9599	1,139.8	2.3932E-03	1.9597	453.6
1.9	1/4	1/1,000	4.3457E-02	-	53.6	4.3678E-02	-	33.7
1.9	1/16	1/1,000	2.8477E-03	1.9659	1,180.4	2.8602E-03	1.9664	455.6

In Figure 4, it becomes apparent that the results generated by our algorithm demonstrate a remarkable alignment with the exact solution. Subsequently, upon comparing the outcomes presented in Table 2, we can see that when $h = H^2 = \frac{1}{16}$, the computational time required for the FEM ranges between 1,100 s and 1,300 s, while the T-GFEM requires a time between 400 s and 500 s, achieving an efficiency improvement of more than twice the original. Therefore, our algorithm not only exhibits spatial second-order convergence akin to the standard finite element method but also significantly decreases computational time, thereby enhancing computational efficiency.

Table 2 presents the outcomes derived from the experiment with the number of subdivisions growing by a square relationship. Analogously, we conduct the other sets of experiments with the number of subdivisions growing by a factor of two, and the result lies in Table 3, where we can draw the conclusion that our algorithm is time efficient on the basis of satisfying convergence. Figure 5 compares the L^2 -norm between FEM and T-GFEM, verifying the T-GFEM could arrive at the same precision as FEM.

Table 3. Space convergence results in L^2 -norm for FEM and T-GFEM

α	$h = H/2$	Δt	Error (FEM)	Order	CPU	Error (T-GFEM)	Order	CPU
1.1	1/4	1/1,000	2.3739E-02	-	53.1	2.4432E-02	-	32.6
1.1	1/8	1/1,000	6.2602E-03	1.9230	222.3	6.3036E-03	1.9545	135.5
1.1	1/16	1/1,000	1.5822E-03	1.9843	1,121.9	1.5846E-03	1.9921	631.6
1.3	1/4	1/1,000	3.0558E-02	-	52.6	2.7305E-02	-	31.8
1.3	1/8	1/1,000	7.0199E-03	1.9854	52.6	7.0546E-03	1.9525	135.3
1.3	1/16	1/1,000	1.7728E-03	1.9859	223.1	1.7747E-03	1.9910	629.2
1.5	1/4	1/1,000	3.0557E-02	-	52.2	3.1020E-02	-	31.1
1.5	1/8	1/1,000	8.0115E-03	1.9314	221.5	8.0367E-03	1.9486	136.7
1.5	1/16	1/1,000	2.0224E-03	1.9589	1,130.2	2.0239E-03	1.9895	621.4
1.7	1/4	1/1,000	3.5874E-02	-	54.6	3.6211E-02	-	31.9
1.7	1/8	1/1,000	9.3935E-03	1.9332	223.1	9.4096E-03	1.9442	135.5
1.7	1/16	1/1,000	2.3704E-03	1.9865	1,139.8	2.3714E-03	1.9884	625.5
1.9	1/4	1/1,000	4.3457E-02	-	53.6	4.3679E-02	-	32.7
1.9	1/8	1/1,000	1.1306E-02	1.9425	222.9	1.1315E-02	1.9487	135.3
1.9	1/16	1/1,000	2.8477E-03	1.9892	1,180.4	2.8483E-03	1.9901	628.6

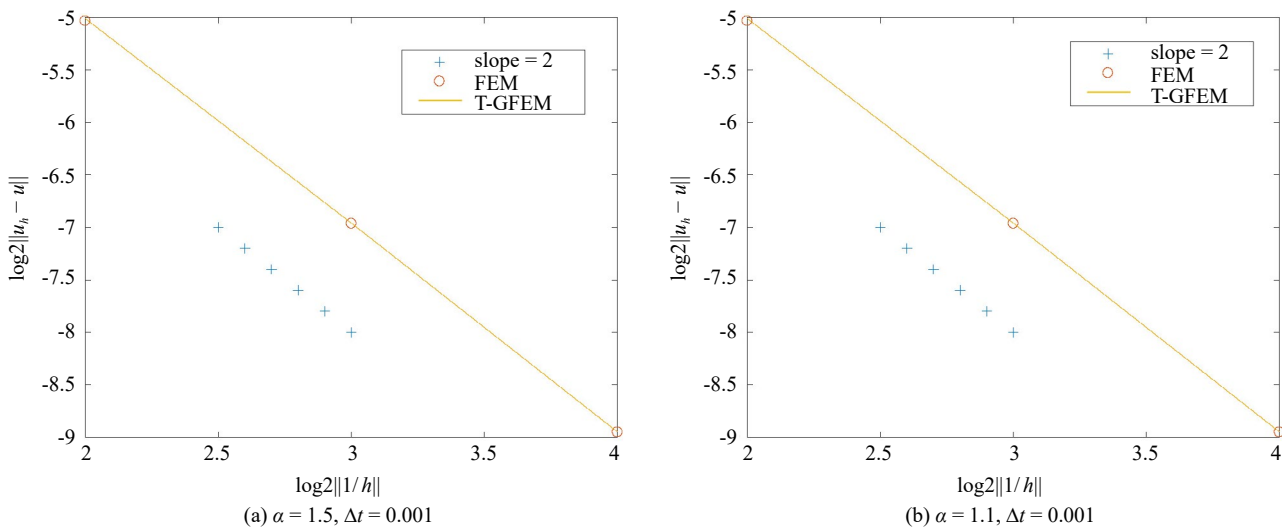


Figure 5. L^2 -norm with different mesh size and fixed temporal step

5.2 Example 2

Then, we consider the zero boundary case

$$\begin{aligned} {}^R_0D_t^\alpha u &= \Delta u - u^2 + g(z, t), \quad (z, t) \in \Omega \times (0, T], \\ u(z, t) &= 0, \quad z \in \partial\Omega, \quad t \in (0, T], \\ u(z, 0) &= 0, \quad u_t(z, 0) = 0, \quad z \in \Omega \cup \partial\Omega, \end{aligned} \quad (37)$$

where $\Omega = (0, 1) \times (0, 1)$, $T = 1$, and $g(z, t) = \frac{\Gamma(5)}{\Gamma(5-\alpha)} t^{4-\alpha} \sin \pi x \sin \pi y + 2t^4 \pi^2 \sin \pi x \sin \pi y + t^8 (\sin \pi x \sin \pi y)^2$. In order to make the solution have an object of reference to compare, we give an exact solution to this model:

$$u(z, t) = t^4 \sin \pi x \sin \pi y, \quad (z, t) \in \Omega \times (0, 1]. \quad (38)$$

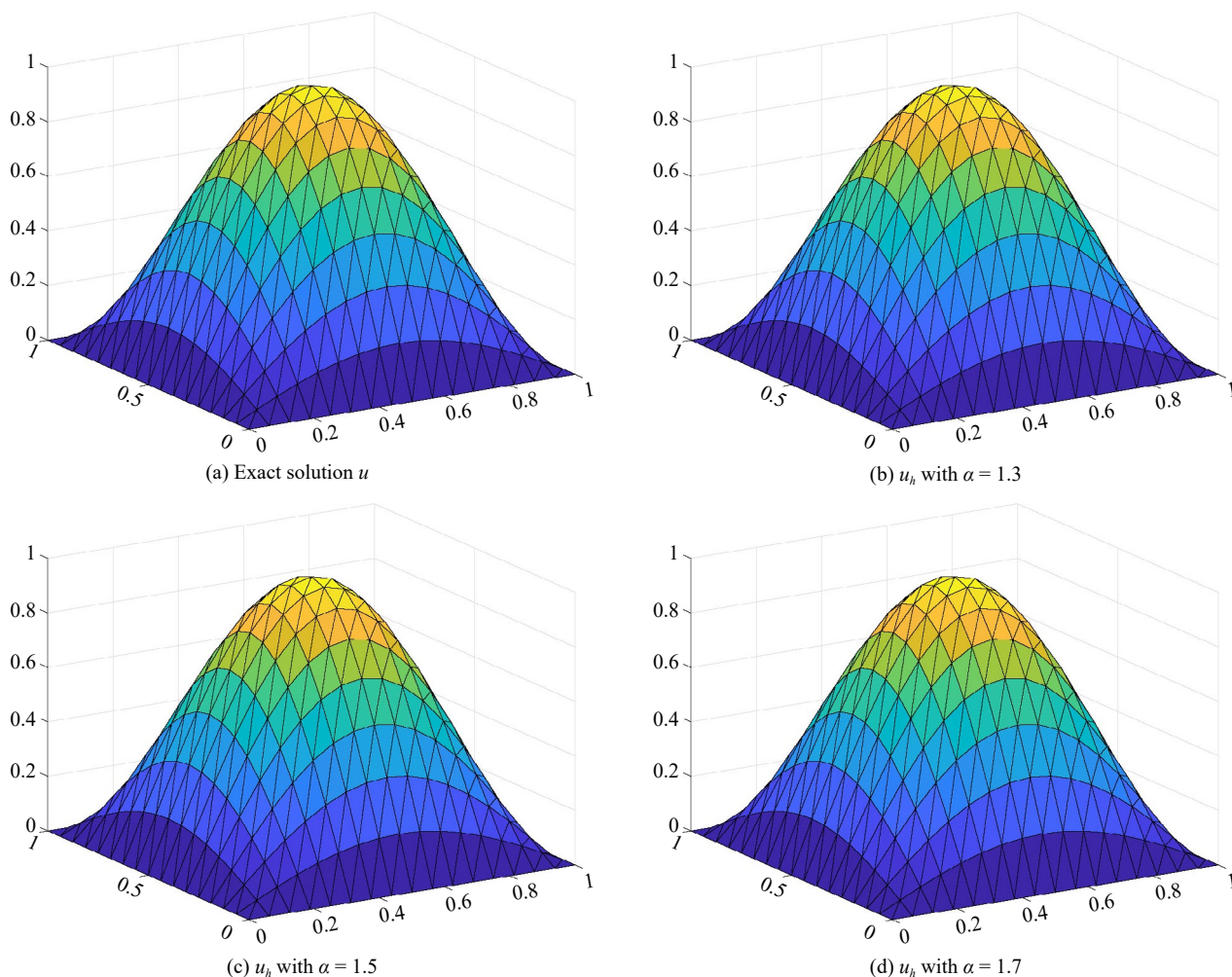


Figure 6. Comparison between exact solution u and numerical solution u_h for the T-GFEM with $h = \frac{H}{2} = \frac{1}{16}$, $\Delta t = \frac{1}{1,000}$ and different α

We conduct experiments with varying values of the α , specifically $\alpha = 1.1, 1.3, 1.5, 1.7, 1.9$, resulting in five sets of numerical results computed with $\Delta t = 1/1,000$. From Figure 6, it is apparent that the numerical solution yielded

by our algorithm matches the exact solution. Furthermore, as shown in Table 4, our algorithm exhibits second-order convergence in space, which is a benchmark for the standard finite element method. Our algorithm also outperforms this method by significantly reducing computation time. When $h = \frac{H}{2} = \frac{1}{16}$, the FEM exhibits a computational time ranging between 900 s and 1,000 s, while the T-GFEM requires a time range of 450 s to 500 s, resulting in an efficiency improvement of over double the original. Therefore, our algorithm not only delivers accurate numerical solution but also enhances computational efficiency, making it a superior solution compared to the standard finite element method.

Table 4. Space convergence results in L^2 -norm for FEM and T-GFEM

α	$h = H/2$	Δt	Error (FEM)	Order	CPU	Error (T-GFEM)	Order	CPU
1.1	1/8	1/1,000	3.1988E-03	-	160.5	3.1371E-03	-	125.5
1.1	1/16	1/1,000	7.6012E-04	2.0732	978.6	7.5654E-04	2.0519	450.1
1.3	1/8	1/1,000	2.1612E-03	-	160.3	2.1165E-03	-	125.3
1.3	1/16	1/1,000	4.9740E-04	2.1194	975.8	4.9492E-04	2.0964	455.4
1.5	1/8	1/1,000	1.0578E-03	-	160.3	1.0413E-03	-	126.7
1.5	1/16	1/1,000	2.3815E-04	2.1512	980.0	2.3758E-04	2.1319	451.9
1.7	1/8	1/1,000	1.7766E-03	-	162.6	1.7814E-03	-	125.5
1.7	1/16	1/1,000	4.8347E-04	1.8776	978.6	4.8377E-04	1.8806	460.0
1.9	1/8	1/1,000	3.7745E-03	-	161.1	3.7868E-03	-	125.3
1.9	1/16	1/1,000	1.0151E-03	1.8950	978.6	1.0157E-03	1.8985	455.5

5.3 Example 3

Finally, we discuss the case in which significant fluctuations occur within the solution domain.

$$\begin{aligned}
 {}_0^R D_t^\alpha u &= \Delta u - 2u^2 + g(z, t), \quad (z, t) \in \Omega \times (0, T], \\
 u(z, t) &= t^4 (\sin 2\pi x^2 + \sin 2\pi y^2), \quad z \in \partial\Omega, \quad t \in (0, T], \\
 u(z, 0) &= 0, \quad u_t(z, 0) = 0, \quad z \in \Omega \cup \partial\Omega,
 \end{aligned} \tag{39}$$

where $\Omega = (0, 1) \times (0, 1)$, $T = 1$, and $g(z, t) = \frac{\Gamma(5)}{\Gamma(5-\alpha)} t^{4-\alpha} (\sin 2\pi x^2 + \sin 2\pi y^2) + t^4 (16\pi^2 (x^2 \sin 2\pi x^2 + y^2 \sin 2\pi y^2) - 4\pi (\cos 2\pi x^2 + \cos 2\pi y^2)) + 2t^8 (\sin \pi x + \sin \pi y)^2$. In order to make the solution have an object of reference to compare, we give an exact solution to this model:

$$u(z, t) = t^4 (\sin 2\pi x^2 + \sin 2\pi y^2), \quad (z, t) \in \Omega \times (0, 1]. \tag{40}$$

Initially, the accuracy of our algorithm can be observed from the close agreement between the numerical solution and the exact solution, as depicted in Figures 7-8. In Table 5, changing $h = \frac{H}{2} = \Delta t = \frac{1}{8}, \frac{1}{16}$, we get error results and computational time of two methods with $\alpha = 1.1, 1.3, 1.5, 1.7, 1.9$. In addition, the comparison of Table 5 indicates that the second-order convergence for our algorithm in space is comparable to the traditional finite element method. On the other hand, our algorithm offers significant improvements in computational efficiency by reducing the computation time. We can see that when $h = \frac{H}{2} = \frac{1}{16}$ the FEM requires a computational time in the range of 4 s to 5 s, whereas the T-GFEM operates within a time frame of 2 s to 2.5 s, leading to an efficiency enhancement exceeding twice the original performance, making it a better alternative to the standard finite element method.

Remark 5.1 (1) In the numerical tests, the numerical examples with exact solution are considered. When the exact solution is not available, one can confirm the efficiency of the proposed algorithm using the method as shown in [20].

(2) From a computational perspective, our numerical algorithm is efficient and fast. However, in view of the characteristics of the developed fully discrete scheme (17) and (19), it is hard for us to derive the error result, which will be left to scholars as an open problem.

Table 5. Space-time convergence results in L^2 -norm for FEM and T-GFEM

α	$h = H/2$	Δt	Error (FEM)	Order	CPU	Error (T-GFEM)	Order	CPU
1.1	1/8	1/8	1.8146E-02	-	0.2	1.8653E-02	-	0.1
1.1	1/16	1/16	4.3962E-03	2.0453	4.5	4.6541E-03	2.0028	2.1
1.3	1/8	1/8	1.6656E-02	-	0.2	1.7277E-02	-	0.1
1.3	1/16	1/16	3.9842E-03	2.0637	4.5	4.2686E-03	2.0111	2.1
1.5	1/8	1/8	1.4902E-02	-	0.2	1.5555E-02	-	0.1
1.5	1/16	1/16	3.4976E-03	2.0910	4.5	3.8137E-03	2.0281	2.2
1.7	1/8	1/8	1.2896E-03	-	0.2	1.3461E-02	-	0.1
1.7	1/16	1/16	2.9379E-03	2.1341	4.5	3.2429E-03	2.0536	2.1
1.9	1/8	1/8	1.0729E-02	-	0.3	1.0873E-02	-	0.1
1.9	1/16	1/16	2.3468E-03	2.1928	4.7	2.5576E-03	2.0878	2.1

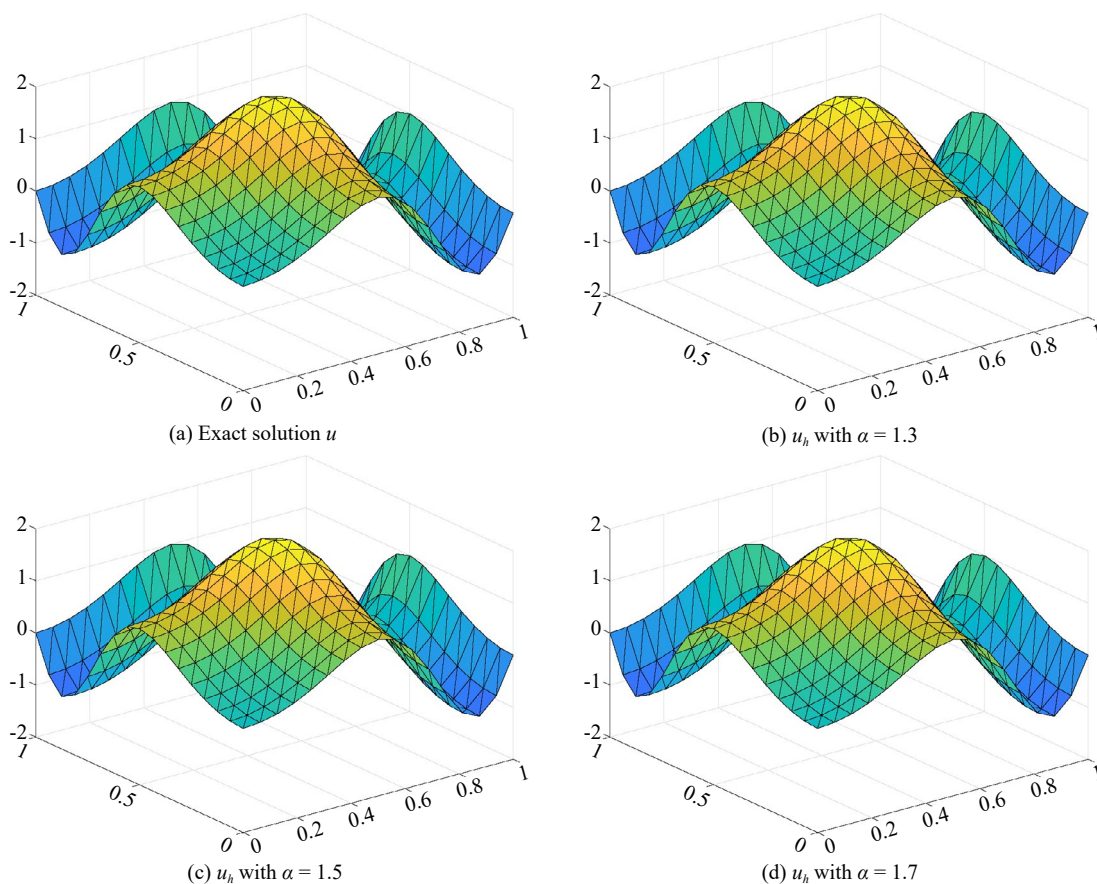


Figure 7. Comparison between exact solution u and numerical solution u_h for the T-GFEM with $h = \frac{H}{2} = \frac{1}{16}$, $\Delta t = \frac{1}{16}$ and different α

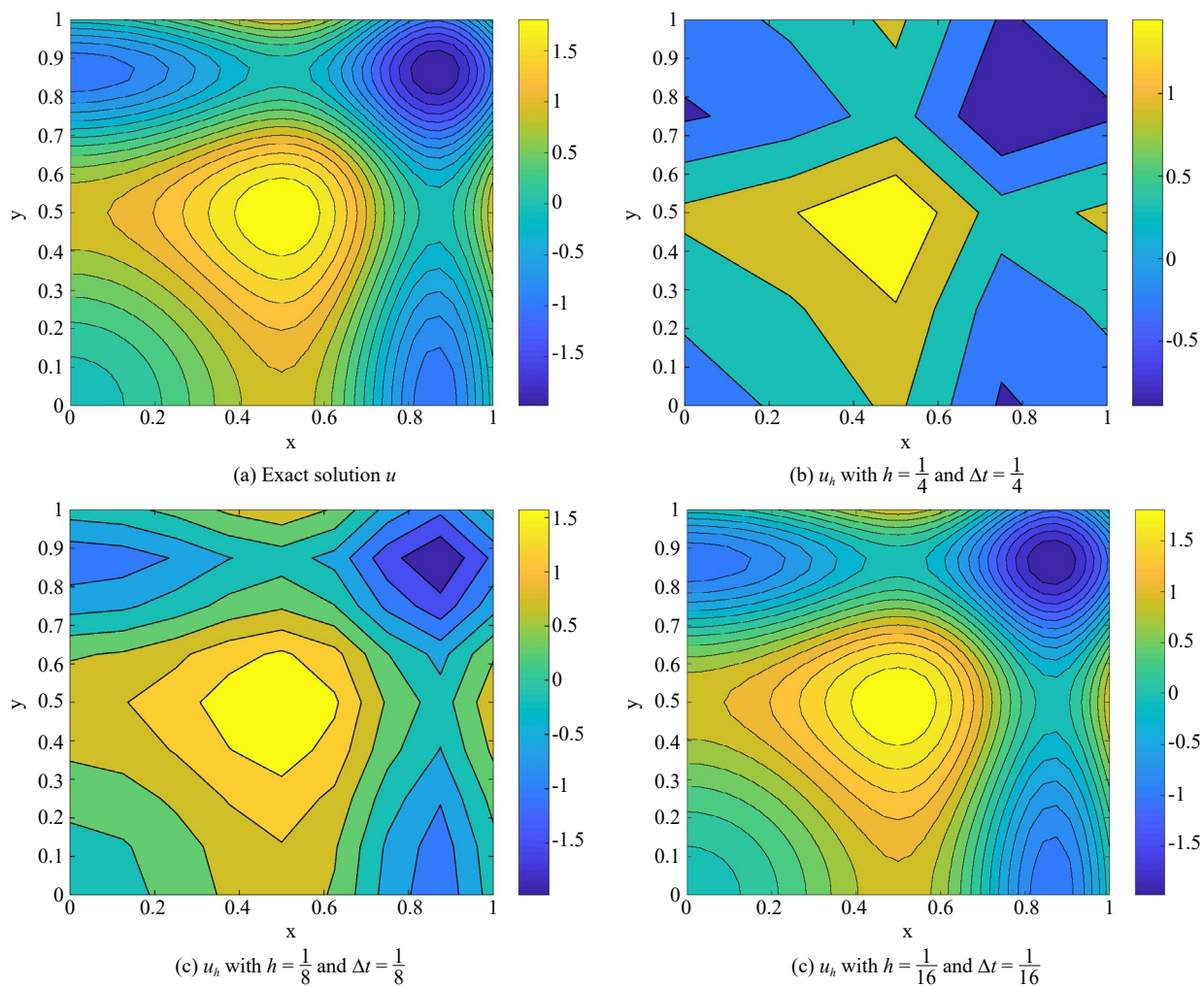


Figure 8. Comparison between exact solution u and numerical solution u_h for the T-GFEM with $\alpha = 1.5$ and different h and Δt

6. Conclusions and future advancements

In this work, a two-grid finite element algorithm has been introduced for solving the two-dimensional nonlinear time fractional Klein-Gordon equation. The explicit treatment of the quadratic nonlinear term provides the advantage of avoiding a significant amount of nonlinear iterations. Through interpolation technique, the two-grid algorithm reduces computational cost while ensuring second-order convergence. Our results have been confirmed through numerical experiments, revealing that, under the same temporal and spatial discretization, our algorithm saves approximately half of computational time compared to standard finite element method. In future work, we will continue to develop the two-grid finite element method for other nonlinear PDEs, and also combine the two-grid method with finite difference methods [21-22] to solve the time fractional Klein-Gordon equation.

Acknowledgments

The authors would like to thank the editor and all the anonymous referees for their valuable suggestions and carefully reading, which greatly improved the presentation of this article. This work is supported by the National Natural Science Foundation of China (12061053, 12161063), Young Innovative Talents Project of Grassland Talents Project and Program for Innovative Research Team in Universities of Inner Mongolia Autonomous Region (NMGIRT2413,

Conflict of interest

The authors declare no competing financial interest.

References

- [1] Verma A, Jiware R, Kumar S. A numerical scheme based on differential quadrature method for numerical simulation of nonlinear Klein-Gordon equation. *International Journal of Numerical Methods for Heat and Fluid Flow*. 2014; 24(7): 1390-1404.
- [2] Jiware R, Pandit S, Mittal RC. Numerical simulation of two-dimensional sine-Gordon solitons by differential quadrature method. *Computer Physics Communications*. 2012; 183(3): 600-616.
- [3] Dehghan M, Ghesmati A. Application of the dual reciprocity boundary integral equation technique to solve the nonlinear Klein-Gordon equation. *Computer Physics Communications*. 2010; 181: 1410-1418.
- [4] Li XL. Meshless numerical analysis of a class of nonlinear generalized Klein-Gordon equations with a well-posed moving least squares approximation. *Applied Mathematical Modelling*. 2017; 48: 153-182.
- [5] Mainardi F. The fundamental solutions for the fractional diffusion-wave equation. *Applied Mathematics Letters*. 1996; 9(6): 23-28.
- [6] Schneider WR, Wyss W. Fractional diffusion and wave equations. *Journal of Mathematical Physics*. 1989; 30(1): 134-144.
- [7] Dehghan M, Abbaszadeh M, Mohebbi A. An implicit RBF meshless approach for solving the time fractional nonlinear Sine-Gordon and Klein-Gordon equations. *Engineering Analysis with Boundary Elements*. 2015; 50: 412-434.
- [8] Zhang H, Jiang X. Unconditionally convergent numerical method for the two-dimensional nonlinear time fractional diffusion-wave equation. *Applied Numerical Mathematics*. 2019; 146: 1-12.
- [9] Lyu P, Vong S. A linearized second-order scheme for nonlinear time fractional Klein-Gordon type equations. *Numerical Algorithms*. 2018; 78(2): 485-511.
- [10] Zhang GY, Huang CM, Fei MF, Wang N. A linearized high-order Galerkin finite element approach for two-dimensional nonlinear time fractional Klein-Gordon equations. *Numerical Algorithms*. 2020; 78(2): 485-511.
- [11] Xu JC. Two-grid discretization techniques for linear and nonlinear PDEs. *SIAM Journal on Numerical Analysis*. 1996; 33: 1759-1777.
- [12] Wu L, Allen MB. A two-grid method for mixed finite-element solution of reaction-diffusion equations. *Numerical Methods for Partial Differential Equations*. 1999; 15(3): 317-332.
- [13] Chen YP, Huang YQ, Yu DH. A two-grid method for expanded mixed finite-element solution of semilinear reaction-diffusion equations. *International Journal for Numerical Methods in Engineering*. 2003; 57(2): 193-209.
- [14] Shi DY, Yang HJ. Unconditional optimal error estimates of a two-grid method for semilinear parabolic equation. *Applied Mathematics and Computation*. 2017; 310: 40-47.
- [15] Liu Y, Du YW, Li H, Wang JF. A two-grid finite element approximation for a nonlinear time-fractional Cable equation. *Nonlinear Dynamics*. 2016; 85(4): 2535-2548.
- [16] Chen CJ, Liu H, Zheng HC, Wang H. A two-grid MMOC finite element method for nonlinear variable-order time-fractional mobile/immobile advection-diffusion equations. *Computers & Mathematics with Applications*. 2019; 79: 2771-2783.
- [17] Tan Z, Zeng Y. Temporal second-order fully discrete two-grid methods for nonlinear time-fractional variable coefficient diffusion-wave equations. *Applied Mathematics and Computation*. 2024; 466: 128457.
- [18] Liu Y, Yin BL, Li H, Zhang ZM. The unified theory of shifted convolution quadrature for fractional calculus. *Journal of Scientific Computing*. 2021; 89: 18.
- [19] Yin BL, Liu Y, Li H. Necessity of introducing non-integer shifted parameters by constructing high accuracy finite difference algorithms for a two-sided space-fractional advection-diffusion model. *Applied Mathematics Letters*. 2020; 105: 106347.
- [20] Liu Y, Yu ZD, Li H, Liu FW, Wang JF. Time two-mesh algorithm combined with finite element method for time fractional water wave model. *International Journal of Heat and Mass Transfer*. 2018; 120: 1132-1145.

- [21] Huang J, Tang Y, Vazquez L, Yang J. Two finite difference schemes for time fractional diffusion wave equation. *Numerical Algorithms*. 2013; 64(4): 707-720.
- [22] Zhang YN, Sun ZZ, Zhao X. Compact alternating direction implicit scheme for the two dimensional fractional diffusion-wave equation. *SIAM Journal on Numerical Analysis*. 2012; 50(3): 1535-1555.