fractal and fractional

Article

Second-Order Time Stepping Scheme Combined with a Mixed Element Method for a 2D Nonlinear Fourth-Order Fractional Integro-Differential Equations

Deng Wang, Yang Liu *, Hong Li * and Zhichao Fang

School of Mathematical Sciences, Inner Mongolia University, Hohhot 010021, China; mathwd@mail.imu.edu.cn (D.W.); zcfang@imu.edu.cn (Z.F.)

* Correspondence: mathliuyang@imu.edu.cn (Y.L.); smslh@imu.edu.cn (H.L.)

Abstract: In this article, we study a class of two-dimensional nonlinear fourth-order partial differential equation models with the Riemann–Liouville fractional integral term by using a mixed element method in space and the second-order backward difference formula (BDF2) with the weighted and shifted Grünwald integral (WSGI) formula in time. We introduce an auxiliary variable to transform the nonlinear fourth-order model into a low-order coupled system including two second-order equations and then discretize the resulting equations by the combined method between the BDF2 with the WSGI formula and the mixed finite element method. Further, we derive stability and error results for the fully discrete scheme. Finally, we develop two numerical examples to verify the theoretical results.

Keywords: nonlinear fourth-order fractional integro-differential equation; WSGI approximation; BDF2; mixed finite element method

Citation: Wang, D.; Liu, Y.; Li, H.; Fang, Z. Second-Order Time Stepping Scheme Combined with a Mixed Element Method for a 2D Nonlinear Fourth-Order Fractional Integro-Differential Equations. *Fractal Fract.* **2022**, *6*, 201. https://doi.org/10.3390/fractalfract6040201

Academic Editors: Maohua Ran, Haci Mehmet Baskonus, Xian-Ming Gu and Xiaoping Xie

Received: 1 March 2022
Accepted: 31 March 2022
Published: 2 April 2022

1. Introduction

In recent years, scholars in various fields of science and engineering have established a large number of mathematical models with fractional differential or integral operators and solved their solutions to explain practical problems. The main reason is that fractional calculus operators are non-local and have a memory effect. However, fractional differential or integral equation models have a complex structure, and thus their solutions are difficult to solve accurately with analytical methods, which has prompted scholars to look for their numerical solutions by designing efficient numerical methods.

Among many fractional calculus models, the fourth-order fractional calculus model has attracted much attention, which can describe many practical problems, such as traveling waves of reaction–diffusion systems, the pattern formation of bistable systems and the propagation of domain walls in liquid crystals. Naturally, increasingly efficient algorithms have been developed to solve these models, which include fourth-order fractional differential Equations (FDEs) (the fourth-order fractional diffusion Equation [1–5], fourth-order fractional wave model [6,7] and other fourth-order fractional models [8–12]) and fourth-order fractional integral Equations (FIEs) [13–16].

From these studies, we find that most scholars have studied numerical algorithms of fourth-order FDEs, and only a few scholars have paid attention to the research of fourth-order FIEs. At the same time, we also noticed that most of these studies on numerical methods for fourth-order FIEs are linear or one-dimensional. Based on these considerations, it is worthwhile to develop efficient numerical algorithms for high-dimensional nonlinear fourth-order FIEs.

Here, we propose an efficient numerical algorithm to solve the following initial and boundary value problem of the fourth-order FIE model

$$\begin{cases} u_t - \Delta_0\mathcal{I}_t^\alpha u + \Delta^2 u - \Delta f(u) = g(z,t), & (z,t) \in \Omega \times (0,T], \\ u(z,0) = u_0(z), & z \in \bar{\Omega}, \\ u(z,t) = 0, & (z,t) \in \partial\Omega \times [0,T], \end{cases} \tag{1}$$

where Ω is a two-dimensional convex polygon region, $(0,T]$ is the time interval with $T > 0$, $f(u)$ is a nonlinear term, and ${}_0\mathcal{I}_t^\alpha(\alpha \in (0,1))$ is the Riemann–Liouville integral operator defined by

$${}_0\mathcal{I}_t^\alpha u(z,t) = \frac{1}{\Gamma(\alpha)} \int_0^t (t-s)^{\alpha-1} u(z,s) \mathrm{d}s. \tag{2}$$

In this article, we design a second-order time stepping scheme based on a mixed element method for solving the fourth-order FIE model (1) with a nonlinear term, where the second-order time stepping scheme is generated by the combination between the BDF2 and the second-order WSGI formula. The WSGI formula is used to approximate the fractional integral term, which was developed in [17] based on the WSGD formula proposed by Tian et al. in [18] and applied in other references [17,19]. In [20], Cao et al. applied the Crank–Nicolson WSGI difference/finite element method to the linear time-fractional wave problem with a second-order space derivative.

However, the WSGI formula is seldom used for fourth-order FIE models. In particular, there has been little research on two-dimensional nonlinear models. For our model (1), in addition to the difficulty caused by the fractional integral term $-\Delta_0\mathcal{I}_t^\alpha u$, there exist the following technical difficulties: **(1)** due to the existence of the nonlinear term $-\Delta f(u)$, the general algorithm design is difficult; **(2)** the high-order space derivative term $\Delta^2 u$ in (1) will lead to the use of higher-order elements if the finite element algorithm is used directly; and **(3)** compared with the works for one-dimensional problems, the research of two-dimensional problems is complex and difficult.

In light of these reasons, we need to construct a fully discrete mixed element algorithm and to develop the theory analyses. First, we split the original problem by introducing a nonlinear auxiliary variable $\sigma = \Delta u - f(u)$ (that is different from [2]) into the following low-order coupled system

$$\begin{aligned} (a)\ \sigma &= \Delta u - f(u), \\ (b)\ u_t - \Delta_0\mathcal{I}_t^\alpha u + \Delta\sigma &= g(z,t). \end{aligned} \tag{3}$$

We discretize the resulting system (3) in time by using the BDF2 and the WSGI formula and then formulate a weak formulation and a fully discrete mixed finite element scheme. Here, our main research content and contributions are as follows:

(1). An efficient low-order mixed element system is proposed to solve the fourth-order FIE model, which can reduce the demand for higher-order elements.

(2). Stability and error analyses based on the proposed fully discrete mixed element system are conducted in detail.

(3). A detailed algorithm is provided to tell readers how to conduct the numerical calculations, and the numerical tests are implemented in two numerical examples to validate our method.

(4). The error data are calculated for our method and another numerical scheme to show the advantages of our method in computational accuracy.

The rest of the article is structured as follows: In Section 1, we formulate the weak formulation and the fully discrete mixed element scheme. In Section 2, we derive the stability. In Section 3, we provide the detailed error analysis. In Section 4, we show several numerical examples to verify the validity of the algorithm and the correctness of the results. Finally, we give some conclusions about our work.

2. Fully Discrete Scheme

First, for any given positive integer N, we divide the time interval $[0,T]$ into N equal parts with $N+1$ nodes, which satisfy $0 = t_0 < t_1 < \cdots < t_N = T$. We define $\tau := T/N$ and obtain $t_n = n\tau$. For convenience, we let $u^n = u(\cdot, t_n)$ and $v^n = v(\cdot, t_n)$.

Now, we need to introduce the following approximation formula for the Riemann–Liouville integral operator (for $\alpha \in (0,1)$) at time node t_n, which is called the WSGI approximation

$$ {}_0\mathcal{I}_t^{\alpha} u^n = \tau^{\alpha} \sum_{k=0}^{n} \lambda_k^{(\alpha)} u^{n-k} + \widetilde{E_1} \triangleq {}_0I_t^{\alpha} u^n + \widetilde{E_0^n}, \tag{4} $$

where the error is $\widetilde{E_0^n} = O(\tau^2)$, and

$$ \lambda_0^{(\alpha)} = (1-\frac{\alpha}{2})\omega_0^{(\alpha)} \ , \ \lambda_k^{(\alpha)} = (1-\frac{\alpha}{2})\omega_k^{(\alpha)} + \frac{\alpha}{2}\omega_{k-1}^{(\alpha)}, \ k \geq 1, \tag{5} $$

$$ \omega_0^{(\alpha)} = 1 \ , \ \omega_k^{(\alpha)} = (1 + \frac{\alpha - 1}{k})\omega_{k-1}^{(\alpha)}, \ k \geq 1. \tag{6} $$

By applying the WSGI approximation formula and taking the BDF2 in time when $n \geq 2$ and the backward Euler scheme when $n = 1$ in (3), we obtain the equivalent formulation as the following

$$ -\Delta u^n + \sigma^n + \Big[2f(u^{n-1}) - f(u^{n-2})\Big] = \widetilde{E_1^n}, \tag{7} $$

$$ \begin{cases} \frac{3u^n - 4u^{n-1} + u^{n-2}}{2\tau} - \Delta {}_0I_t^{\alpha} u^n + \Delta\sigma^n = g^n + \widetilde{E_2^n}, & n \geq 2, \\ \frac{u^1 - u^0}{\tau} - \Delta {}_0I_t^{\alpha} u^1 + \Delta\sigma^1 = g^1 + \widetilde{E_2^1}, & n = 1, \end{cases} \tag{8} $$

where $\widetilde{E_1^n} = O(\tau^2)$, $\widetilde{E_2^n} = O(\tau^2)$ and $\widetilde{E_2^1} = O(\tau)$, and $\bar{f}(u^n) \doteq 2f(u^{n-1}) - f(u^{n-2})$ is a time second-order approximation for the nonlinear term $f(u^n)$.

Now, we first multiply (3)(a) by $v \in H_0^1$ and (3)(b) by $w \in H_0^1$, respectively, and further we integrate with respect to the spatial domain Ω to arrive at

$$ (\nabla u^n, \nabla v) + (\sigma^n, v) + (2f(u^{n-1}) - f(u^{n-2}), v) = (\widetilde{E_1^n}, v), \tag{9} $$

$$ \begin{cases} \left(\frac{3u^n - 4u^{n-1} + u^{n-2}}{2\tau}, w\right) + (\nabla {}_0I_t^{\alpha} u^n, \nabla w) - (\nabla\sigma^n, \nabla w) = (g^n, w) + (\widetilde{E_2^n}, w), & n \geq 2, \\ \left(\frac{u^1 - u^0}{\tau}, w\right) + (\nabla {}_0I_t^{\alpha} u^1, \nabla w) - (\nabla\sigma^1, \nabla w) = (g^1, w) + (\widetilde{E_2^1}, w), & n = 1. \end{cases} \tag{10} $$

Next, we take finite element space $V_h \subset H_0^1$ and obtain the fully discrete scheme as follows

$$ (\nabla u_h^n, \nabla v_h) + (\sigma_h^n, v_h) + (2f(u_h^{n-1}) - f(u_h^{n-2}), v_h) = 0, \tag{11} $$

$$ \begin{cases} \left(\frac{3u_h^n - 4u_h^{n-1} + u_h^{n-2}}{2\tau}, w_h\right) + (\nabla {}_0I_t^{\alpha} u_h^n, \nabla w_h) - (\nabla\sigma_h^n, \nabla w_h) = (g^n, w_h), & n \geq 2, \\ \left(\frac{u_h^1 - u_h^0}{\tau}, w_h\right) + (\nabla {}_0I_t^{\alpha} u_h^1, \nabla w_h) - (\nabla\sigma_h^1, \nabla w_h) = (g^1, w_h), & n = 1. \end{cases} \tag{12} $$

Remark 1. *If we introduce $\sigma = -{}_0\mathcal{I}_t^{\alpha} u + \Delta u - f(u)$, we reduce (1) into the following coupled system*

$$ \sigma = -{}_0\mathcal{I}_t^{\alpha} u + \Delta u - f(u), \tag{13} $$

$$ u_t + \Delta\sigma = g(z, t). \tag{14} $$

By a similar process to (11) and (12), we easily arrive at another fully discrete scheme

$$(\sigma_h^n, v_h) + ({}_0I_t^\alpha u_h^n, v_h) + (\nabla u_h^n, \nabla v_h) + (2f(u_h^{n-1}) - f(u_h^{n-2}), v_h) = 0, \tag{15}$$

$$\begin{cases} \left(\frac{3u_h^n - 4u_h^{n-1} + u_h^{n-2}}{2\tau}, w_h\right) - (\nabla \sigma_h^n, \nabla w_h) = (g^n, w_h), & n \geq 2, \\ \left(\frac{u_h^1 - u_h^0}{\tau}, w_h\right) - (\nabla \sigma_h^1, \nabla w_h) = (g^1, w_h), & n = 1. \end{cases} \tag{16}$$

In numerical tests, we make a comparison between our method in this article and the mixed element system (15) and (16) to illustrate the advantages of our method.

3. Stability Analysis

We analyze the stability of the numerical scheme above in this section. First, we need to introduce several lemmas to make preparations for it.

Lemma 1. *For series $\{\chi^n\}$, the following inequality holds*

$$\left(\frac{3\chi^n - 4\chi^{n-1} + \chi^{n-2}}{2\tau}, \chi^n\right) \geq \frac{1}{4\tau}\left[\mathbb{H}(\chi^n) - \mathbb{H}(\chi^{n-1})\right], \quad n \geq 2, \tag{17}$$

where

$$\mathbb{H}(\chi^n) = 3\|\chi^n\|^2 - \|\chi^{n-1}\|^2 + 2\|\chi^n - \chi^{n-1}\|^2 \geq \|\chi^n\|^2. \tag{18}$$

Theorem 1. *For $u_h^n, \sigma_h^n \in V_h$, we can obtain the stability for the fully discrete system (11)–(12)*

$$\|u_h^L\|^2 + K\tau \sum_{n=1}^{L} \|\sigma_h^n\|^2 \leq C(\|u_h^0\|^2 + \tau \sum_{n=1}^{L} \|g^n\|^2), \tag{19}$$

where $L = 1, 2, \cdots, N$.

Proof. For $n \geq 2$, we take $v_h = \sigma_h^n$ in (11) and $w_h = u_h^n$ in (12), we use Lemma 1 and apply the Hölder inequality and Young inequality to obtain

$$\begin{aligned} &\frac{1}{4\tau}\left[\mathbb{H}(u_h^n) - \mathbb{H}(u_h^{n-1})\right] + \tau^\alpha \sum_{k=0}^{n} \lambda_k^{(\alpha)} (\nabla u_h^{n-k}, \nabla u_h^n) \\ \leq &\left(\frac{3u_h^n - 4u_h^{n-1} + u_h^{n-2}}{2\tau}, u_h^n\right) + \tau^\alpha \sum_{k=0}^{n} \lambda_k^{(\alpha)} (\nabla u_h^{n-k}, \nabla u_h^n) \\ = &(g^n, u_h^n) + (\nabla \sigma_h^n, \nabla u_h^n) \\ = &(g^n, u_h^n) - \|\sigma_h^n\|^2 - (2f(u_h^{n-1}) - f(u_h^{n-2}), \sigma_h^n) \\ \leq &-\frac{1}{2}\|\sigma_h^n\|^2 + C(\|u_h^n\|^2 + \|u_h^{n-1}\|^2 + \|u_h^{n-2}\|^2) + C\|g^n\|^2. \end{aligned} \tag{20}$$

Sum (20) with respect to n from 2 to L and multiply both sides of the inequality by 4τ so that we can obtain

$$\begin{aligned} \mathbb{H}(u_h^L) + 2\tau \sum_{n=2}^{L} \|\sigma_h^n\|^2 + 4\tau^{1+\alpha} \sum_{n=2}^{L} \sum_{k=0}^{n} \lambda_k^{(\alpha)} (\nabla u_h^{n-k}, \nabla u_h^n) \\ \leq \mathbb{H}(u_h^1) + C\tau \sum_{n=2}^{L} \|g^n\|^2 + C\tau \sum_{n=0}^{L} \|u_h^n\|^2. \end{aligned} \tag{21}$$

For the case $n = 1$, we conduct a similar process to the above analyses to easily obtain

$$\|u_h^1\|^2 + 2\tau^{\alpha+1}\sum_{k=0}^{1}\lambda_k^{(\alpha)}(\nabla u_h^{1-k}, \nabla u_h^1) + \tau\|\sigma_h^1\|^2 \le \|u_h^0\|^2 + \tau\|g^1\|^2 + \tau\|u_h^1\|^2. \tag{22}$$

Combine (21) with (22) and use $\mathbb{H}(u_h^1) \le C(\|u_h^1\|^2 + \|u_h^0\|^2)$ to obtain

$$\begin{aligned}(1 - C\tau)\|u_h^L\|^2 + 4\tau^{\alpha+1}\sum_{n=0}^{L}\sum_{k=0}^{n}\lambda_k^{(\alpha)}(\nabla u_h^{n-k}, \nabla u_h^n) + \tau\sum_{n=1}^{L}\|\sigma_h^n\|^2 \\ \le C\|u_h^0\|^2 + C\tau\sum_{n=1}^{L}\|g^n\|^2 + C\tau\sum_{n=0}^{L-1}\|u_h^n\|^2.\end{aligned} \tag{23}$$

Due to the fact that $({}_0I_t^\alpha \nabla u_h^n, \nabla u_h^n) \ge 0$, we can remove it on the left-hand side of the inequality and use the Gronwall inequality to obtain the stability result. □

4. Error Analysis

Before we conduct the error analysis, the following Ritz-projection operator [5] needs to be introduced. We let the operator $\mathcal{R}_h : H_0^1(\Omega) \to V_h$ for any given $z \in H_0^1(\Omega)$ satisfy

$$(\nabla(z - \mathcal{R}_h z), \nabla z_h) = 0, \quad \forall z_h \in V_h, \tag{24}$$

with the following estimate inequality.

$$\|z - \mathcal{R}_h z\| + \|z_t - \mathcal{R}_h z_t\| + h\|z - \mathcal{R}_h z\|_1 \le Ch^{r+1}, \quad \forall z \in H^{r+1}(\Omega) \cap H_0^1(\Omega). \tag{25}$$

Theorem 2. *If (u, σ) is the solution of the mixed weak system (9) and (10) and (u_h, σ_h) is the solution of the fully discrete system (11) and (12), we would obtain the conclusion that there exists a constant C that makes the following inequality hold with the initial condition $\mathcal{R}_h u^0 = u_h^0$.*

$$\|u(t_L) - u_h^L\| + \left(\tau\sum_{n=1}^{L}\|\sigma(t_n) - \sigma_h^n\|\right)^{\frac{1}{2}} \le C(h^{r+1} + \tau^2), \quad L = 1, 2, \cdots, N, \tag{26}$$

where the constant C is independent of the spatial mesh parameter h and time grid step length τ.

Proof. For the convenience of expression, we write the errors as $u(t_n) - u_h^n = (u(t_n) - \mathcal{R}_h u^n) + (\mathcal{R}_h u^n - u_h^n) \triangleq \eta_u^n + \theta_u^n$, $\sigma(t_n) - \sigma_h^n = (\sigma(t_n) - \mathcal{R}_h \sigma^n) + (\mathcal{R}_h \sigma^n - \sigma_h^n) \triangleq \phi_\sigma^n + \xi_\sigma^n$. We still consider the case of $n \ge 2$ first. Subtract (11) from (9), subtract (12) from (10), apply the formula (24), take $v_h = \xi_\sigma^n$, $w_h = \theta_u^n$ and use the Hölder inequality as well as Young inequality to arrive at

$$\begin{aligned}&\left(\frac{3\theta_u^n - 4\theta_u^{n-1} + \theta_u^{n-2}}{2\tau}, \theta_u^n\right) + \tau^\alpha\sum_{k=0}^{n}\lambda_k^{(\alpha)}(\nabla\theta_u^{n-k}, \nabla\theta_u^n) \\ =&(\nabla\xi_\sigma^n, \nabla\theta_u^n) - \left(\frac{3\eta_u^n - 4\eta_u^{n-1} + \eta_u^{n-2}}{2\tau}, \theta_u^n\right) + (\widetilde{E_2}, \theta_u^n) \\ =&-\|\xi_\sigma^n\|^2 - \left(\frac{3\eta_u^n - 4\eta_u^{n-1} + \eta_u^{n-2}}{2\tau}, \theta_u^n\right) - (\bar{f}(u^n) - \bar{f}(u_h^n), \xi_\sigma^n) \\ &+ (\widetilde{E_1^n}, \xi_\sigma^n) - (\phi_\sigma^n, \xi_\sigma^n) + (\widetilde{E_2^n}, \theta_u^n) \\ =&-\frac{1}{2}\|\xi_\sigma^n\|^2 + \left\|\frac{3\eta_u^n - 4\eta_u^{n-1} + \eta_u^{n-2}}{2\tau}\right\|^2 + \|\phi_\sigma^n\|^2 + \|\widetilde{E_1^n}\|^2 + \|\widetilde{E_2^n}\|^2 + C\|\theta_u^n\|^2 \\ &+ C\|f'(\bar{u}^{n-1})\|_\infty^2(\|\eta_u^{n-1}\|^2 + \|\theta_u^{n-1}\|^2) + C\|f'(\bar{u}^{n-2})\|_\infty^2(\|\eta_u^{n-2}\|^2 + \|\theta_u^{n-2}\|^2),\end{aligned} \tag{27}$$

where $\bar{f}(z^n) = 2f(z^{n-1}) - f(z^{n-2})$, $z = u$ or u_h, $\bar{u}^i$ $(i = n-1$ or $n-2)$ is the value between u^i and u_h^i.

Multiply (27) by 4τ and sum the resulting inequality from $n = 2$ to L to arrive at

$$\begin{aligned}
&\mathbb{H}(\theta_u^L) + 4\tau^{1+\alpha} \sum_{n=2}^{L} \sum_{k=0}^{n} \lambda_k^{(\alpha)} (\nabla\theta_u^{n-k}, \nabla\theta_u^n) + 2\tau \sum_{n=2}^{L} \|\xi_\sigma^n\|^2 \\
\leq &\mathbb{H}(\theta_u^1) + C\tau \sum_{n=2}^{L} \left(\left\| \frac{3\eta_u^n - 4\eta_u^{n-1} + \eta_u^{n-2}}{2\tau} \right\|^2 + \|\phi_\sigma^n\|^2 + \|\widetilde{E_1^n}\|^2 + \|\widetilde{E_2^n}\|^2 \right) \\
&+ C\tau \sum_{n=0}^{L} \|\theta_u^n\|^2 + C\tau \sum_{n=0}^{L-1} \|\eta_u^n\|^2.
\end{aligned} \tag{28}$$

Secondly, we consider the case $n = 1$. By a similar process as the case $n \geq 2$, we take $v_h = \xi_\sigma^1$ and $w_h = \theta_u^1$ to easily derive

$$\begin{aligned}
&\|\theta_u^1\|^2 + 2\tau^{\alpha+1} \sum_{k=0}^{1} \lambda_k^{(\alpha)} (\nabla\theta_u^{1-k}, \nabla\theta_u^1) + 2\tau\|\xi_\sigma^1\|^2 \\
\leq &C\tau^2 \left\| \frac{\eta_u^1 - \eta_u^0}{\tau} \right\|^2 + \frac{1}{2}\|\theta_u^1\|^2 + \tau\|\xi_\sigma^1\|^2 + C\tau\|\phi_\sigma^1\|^2 \\
&+ C\tau(\|C(u^0, u_h^0)\|_\infty + 1)(\|\eta_u^0\|^2 + \|\theta_u^0\|^2) + C\tau^4.
\end{aligned} \tag{29}$$

Now, combine (28) and (29) with (25) and use the Gronwall lemma to obtain

$$\|\theta_u^L\|^2 + \tau^{\alpha+1} \sum_{n=0}^{L} \sum_{k=0}^{n} \lambda_k^{(\alpha)} (\nabla\theta_u^{n-k}, \nabla\theta_u^n) + \tau \sum_{n=1}^{L} \|\xi_\sigma^n\|^2 \leq C(h^{2r+2} + \tau^4). \tag{30}$$

Finally, combine (30) with (25) and use the triangle inequality to obtain the conclusion. □

5. Numerical Tests

5.1. Two-Dimensional Example Based on the Triangular Meshes

Here, a specific algorithm is given to illustrate how to implement the calculation process, and numerical results are given to verify our theoretical results.

5.1.1. Numerical Algorithm

We show the numerical algorithm with two processes, including the preliminary knowledge of algorithm and the algorithm based on our scheme.

Process I: Preliminary knowledge of the algorithm

We give the numerical algorithm based on the space–time mesh parameters M and N, where N is the number of time cells and M is the number of spatial triangular units.

In order to use linear interpolation in each triangular unit $I_p, (1 \leq p \leq M)$, we take the three vertexes of the unit as interpolation points. If we define the coordinates of three vertexes as (x_i, y_i), (x_j, y_j), (x_m, y_m), the corresponding triangular unit's area Δ_e can be expressed as a third-order determinant consisting of the coordinates above.

Setting the values of the linear interpolation function u_h at the three nodes u_i, u_j, u_m, in triangular unit I_p, we can easily work out the three unknown coefficients $\beta_1, \beta_2, \beta_3$ of u_h, which are completely determined by (x_k, y_k) and u_k, $(k = i, j, m)$.

Further, substituting $\{\beta_k, k = 1, 2, 3\}$, into the general form of the linear interpolation function u_h, we can obtain the expression for u_h in I_p, which is

$$u_h = N_i u_i + N_j u_j + N_m u_m, \tag{31}$$

where

$$N_k(x,y) = \frac{1}{2\Delta_e}(a_k x + b_k y + c_k), \quad k = i,j,m, \tag{32}$$

specifically, Δ_e is given as follows, and a_k, b_k, c_k are defined by the latter equation

$$\Delta_e = \frac{1}{2}\begin{vmatrix} x_i & y_i & 1 \\ x_j & y_j & 1 \\ x_m & y_m & 1 \end{vmatrix}, h_l = \begin{vmatrix} s_{l+1} & r_{l+1} \\ s_{l+2} & r_{l+2} \end{vmatrix}, \tag{33}$$

where $h = a,b,c$, and the values of s and r depending on h and the indexes $l+1$ are the following

$$s = \begin{cases} y, & h = a, \\ -x, & h = b, \\ x, & h = c, \end{cases}, r = \begin{cases} 1, & h = a,b \\ y, & h = c, \end{cases}, l+1 = \begin{cases} j, & l = i, \\ m, & l = j, \\ i, & l = m, \end{cases} \tag{34}$$

Likewise, the indexes $l+2$ can be obtained by the indexes $l+1$.

Similarly, we set that function v takes v_i, v_j, v_m at mesh nodes $\{i,j,m\}$ in each triangular unit I_p. From (31)–(34), we know that

$$\begin{aligned} u_h(x,y) &= \sum_{k=i,j,m} N_k(x,y)u_k \quad , \quad v(x,y) = \sum_{k=i,j,m} N_k(x,y)v_k, \\ \frac{\partial u_h}{\partial x} &= \frac{1}{2\Delta_e}\sum_{k=i,j,m} a_k u_k \quad , \quad \frac{\partial u_h}{\partial y} = \frac{1}{2\Delta_e}\sum_{k=i,j,m} b_k u_k. \end{aligned} \tag{35}$$

To simplify this expression, let us introduce the matrix B and three dimensional column vector $\boldsymbol{u}^{(e)}, \boldsymbol{v}^{(e)}, \boldsymbol{w}^{(e)}, \boldsymbol{N}(x,y)$ as follows

$$\begin{aligned} B =& \frac{1}{2\Delta_e}\begin{bmatrix} a_i & a_j & a_m \\ b_i & b_j & b_m \end{bmatrix}, \boldsymbol{u}^{(e)} = \begin{bmatrix} u_i & u_j & u_m \end{bmatrix}^T, \boldsymbol{v}^{(e)} = \begin{bmatrix} v_i & v_j & v_m \end{bmatrix}^T, \\ \boldsymbol{w}^{(e)} =& \begin{bmatrix} w_i & w_j & w_m \end{bmatrix}^T, \boldsymbol{N}(x,y) = \begin{bmatrix} N_i & N_j & N_m \end{bmatrix}^T. \end{aligned} \tag{36}$$

From (36), the gradients of $u_h(x,y), v(x,y)$ can be expressed as

$$\nabla u_h = B\boldsymbol{u}^{(e)}, \nabla v = B\boldsymbol{v}^{(e)}. \tag{37}$$

Therefore,

$$u_h(x,y) = \boldsymbol{N}(x,y)^T\boldsymbol{u}^{(e)}, v(x,y) = \boldsymbol{N}(x,y)^T\boldsymbol{v}^{(e)}. \tag{38}$$

Process II: The algorithm based on our scheme

According to the formulas above, we can express (10) and (11) (the case of $n \geq 2$) as

$$(\left[\boldsymbol{u}^n_{(e)}\right]^T B^T B, \boldsymbol{v}(e)) + (\left[\boldsymbol{\sigma}^n_{(e)}\right]^T NN^T, \boldsymbol{v}(e)) = -((2f(\boldsymbol{u}^{n-1}) - f(\boldsymbol{u}^{n-2}))N, \boldsymbol{v}_{(e)}), \tag{39}$$

$$\begin{aligned} &(\frac{3}{2\tau}\left[\boldsymbol{u}^n_{(e)}\right]^T NN^T, \boldsymbol{w}_{(e)}) + (\tau^\alpha \lambda_0^{(\alpha)}\left[\boldsymbol{u}^n_{(e)}\right]^T B^T B, \boldsymbol{w}_{(e)}) - (\left[\boldsymbol{\sigma}^n_{(e)}\right]^T B^T B, \boldsymbol{w}_{(e)}) \\ =&(-\tau^\alpha \sum_{k=1}^{n} \lambda_k^{(\alpha)}\left[\boldsymbol{u}^{n-k}_{(e)}\right]^T B^T B, \boldsymbol{w}_{(e)}) + (\frac{2}{\tau}\left[\boldsymbol{u}^{n-1}_{(e)}\right]^T NN^T, \boldsymbol{w}_{(e)}) \\ &- (\frac{1}{2\tau}\left[\boldsymbol{u}^{n-2}_{(e)}\right]^T NN^T, \boldsymbol{w}_{(e)}) + (g^n, \boldsymbol{w}_{(e)}). \end{aligned} \tag{40}$$

For the case $n = 1$ in (11), we can deal with it by a similar method and do not repeat it here. Now, we write (39) and (40) as the following

$$\sum_e \left(\left[\boldsymbol{u}^n_{(e)}\right]^T A^1_{(e)} \boldsymbol{v}_{(e)} + \left[\boldsymbol{\sigma}^n_{(e)}\right]^T B^1_{(e)} \boldsymbol{v}_{(e)} \right) = \sum_e \left[b^1_{(e)}\right]^T \boldsymbol{v}_{(e)}, \tag{41}$$

$$\sum_e \left(\left[\boldsymbol{u}^n_{(e)}\right]^T A^2_{(e)} \boldsymbol{w}_{(e)} + \left[\boldsymbol{\sigma}^n_{(e)}\right]^T B^2_{(e)} \boldsymbol{w}_{(e)} \right) = \sum_e \left[b^2_{(e)}\right]^T \boldsymbol{w}_{(e)}, \tag{42}$$

where

$$\begin{aligned}
A^1_{(e)} &= \iint_e (B^T B)\mathrm{d}x\mathrm{d}y,\ B^1_{(e)} = \iint_e (NN^T)\mathrm{d}x\mathrm{d}y, \\
\boldsymbol{b}^1_{(e)} &= -\iint_e (2f(\boldsymbol{u}^{n-1}) - f(\boldsymbol{u}^{n-2}))N\mathrm{d}x\mathrm{d}y, \\
A^2_{(e)} &= \iint_e (\frac{3}{2\tau}NN^T + \tau^\alpha \lambda_0^{(\alpha)} B^T B)\mathrm{d}x\mathrm{d}y,\ B^2_{(e)} = -\iint_e (B^T B)\mathrm{d}x\mathrm{d}y, \\
\boldsymbol{b}^2_{(e)} &= \iint_e (-\tau^\alpha \sum_{k=1}^{n} \lambda_k^{(\alpha)} B^T B \boldsymbol{u}^{n-k}_{(e)} + \frac{2}{\tau} NN^T u^{n-1}_{(e)} - \frac{1}{2\tau} NN^T u^{n-2}_{(e)} + g^n N)\mathrm{d}x\mathrm{d}y.
\end{aligned} \tag{43}$$

Clearly, (41) and (42) are equivalent to

$$\begin{bmatrix} A^1_{(e)} & B^1_{(e)} \\ A^2_{(e)} & B^2_{(e)} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^n_{(e)} \\ \boldsymbol{\sigma}^n_{(e)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}^1_{(e)} \\ \boldsymbol{b}^2_{(e)} \end{bmatrix}. \tag{44}$$

In other words,

$$K\vec{U}^n = \vec{G}^n,\ 1 \le n \le N, \tag{45}$$

where

$$K = \begin{bmatrix} A_1 & B_1 \\ A_2 & B_2 \end{bmatrix}, \vec{U}^n = \begin{bmatrix} \boldsymbol{u}^n \\ \boldsymbol{\sigma}^n \end{bmatrix}, \vec{G}^n = \begin{bmatrix} \boldsymbol{b}^n_1 \\ \boldsymbol{b}^n_2 \end{bmatrix}, \tag{46}$$

$$A_1 = \sum_e A^1_{(e)}, A_2 = \sum_e A^2_{(e)}, B_1 = \sum_e B^1_{(e)}, B_2 = \sum_e B^2_{(e)}, \boldsymbol{b}^n_1 = \sum_e \boldsymbol{b}^1_{(e)}, \boldsymbol{b}^n_2 = \sum_e \boldsymbol{b}^2_{(e)}. \tag{47}$$

According to (41) and (43), we know that

$$A^1_{(e)} = \begin{bmatrix} & \vdots & & \vdots & & \vdots & \\ \cdots & a^{(e)}_{ii} & \cdots & a^{(e)}_{ij} & \cdots & a^{(e)}_{im} & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & a^{(e)}_{ji} & \cdots & a^{(e)}_{jj} & \cdots & a^{(e)}_{jm} & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & a^{(e)}_{mi} & \cdots & a^{(e)}_{jm} & \cdots & a^{(e)}_{mm} & \cdots \\ & \vdots & & \vdots & & \vdots & \end{bmatrix}, B^1_{(e)} = \begin{bmatrix} & \vdots & & \vdots & & \vdots & \\ \cdots & b^{(e)}_{ii} & \cdots & b^{(e)}_{ij} & \cdots & b^{(e)}_{im} & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & b^{(e)}_{ji} & \cdots & b^{(e)}_{jj} & \cdots & b^{(e)}_{jm} & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & b^{(e)}_{mi} & \cdots & b^{(e)}_{jm} & \cdots & b^{(e)}_{mm} & \cdots \\ & \vdots & & \vdots & & \vdots & \end{bmatrix},$$

$$A^2_{(e)} = \begin{bmatrix} & \vdots & & \vdots & & \vdots & \\ \cdots & c^{(e)}_{ii} & \cdots & c^{(e)}_{ij} & \cdots & c^{(e)}_{im} & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & c^{(e)}_{ji} & \cdots & c^{(e)}_{jj} & \cdots & c^{(e)}_{jm} & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & c^{(e)}_{mi} & \cdots & c^{(e)}_{jm} & \cdots & c^{(e)}_{mm} & \cdots \\ & \vdots & & \vdots & & \vdots & \end{bmatrix}, B^2_{(e)} = \begin{bmatrix} & \vdots & & \vdots & & \vdots & \\ \cdots & d^{(e)}_{ii} & \cdots & d^{(e)}_{ij} & \cdots & d^{(e)}_{im} & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & d^{(e)}_{ji} & \cdots & d^{(e)}_{jj} & \cdots & d^{(e)}_{jm} & \cdots \\ & \vdots & & \vdots & & \vdots & \\ \cdots & b^{(e)}_{mi} & \cdots & d^{(e)}_{jm} & \cdots & d^{(e)}_{mm} & \cdots \\ & \vdots & & \vdots & & \vdots & \end{bmatrix},$$

$$\boldsymbol{b}^1_{(e)} = \begin{bmatrix} \cdots & b^{(e)}_{i1} & \cdots & b^{(e)}_{j1} & \cdots & b^{(e)}_{m1} & \cdots \end{bmatrix}^T, \boldsymbol{b}^2_{(e)} = \begin{bmatrix} \cdots & b^{(e)}_{i2} & \cdots & b^{(e)}_{j2} & \cdots & b^{(e)}_{m2} & \cdots \end{bmatrix}^T,$$

and

$$\begin{aligned}
a^{(e)}_{kl} &= \iint_e \frac{1}{4\Delta_e^2}(a_k a_l + b_k b_l)\mathrm{d}x\mathrm{d}y,\ b^{(e)}_{kl} = \iint_e N_k N_l \mathrm{d}x\mathrm{d}y,\ (k, l = i, j, m),\\
c^{(e)}_{kl} &= \iint_e (\frac{3}{2\tau} N_k N_l + \tau^\alpha \lambda_0 \frac{1}{4\Delta_e^2}(a_k a_l + b_k b_l))\mathrm{d}x\mathrm{d}y,\ d^{(e)}_{kl} = \iint_e -\frac{1}{4\Delta_e^2}(a_k a_l + b_k b_l)\mathrm{d}x\mathrm{d}y,\\
b^{(e)}_{1i} &= \iint_e -(2f(u_e^{n-1}) - f(u_e^{n-2}))N_i \mathrm{d}x\mathrm{d}y,\\
b^{(e)}_{1j} &= \iint_e -(2f(u_e^{n-1}) - f(u_e^{n-2}))N_j \mathrm{d}x\mathrm{d}y,\\
b^{(e)}_{1m} &= \iint_e -(2f(u_e^{n-1}) - f(u_e^{n-2}))N_m \mathrm{d}x\mathrm{d}y,\\
b^{(e)}_{2i} &= \iint_e (-\tau^\alpha \frac{1}{4\Delta_e^2} \sum_{k=1}^{n} \lambda_k^\alpha ((a_i^2 + b_i^2)u_{e,i}^{n-k} + (a_i a_j + b_i b_j)u_{e,j}^{n-k} + (a_i a_m + b_i b_m)u_{e,m}^{n-k})\\
&\quad + \frac{2}{\tau}(N_i N_i u_{e,i}^{n-1} + N_i N_j u_{e,j}^{n-1} + N_i N_m u_{e,m}^{n-1})\\
&\quad - \frac{1}{2\tau}(N_i N_i u_{e,i}^{n-2} + N_i N_j u_{e,j}^{n-2} + N_i N_m u_{e,m}^{n-2}) + g^n N_i)\mathrm{d}x\mathrm{d}y,\\
b^{(e)}_{2j} &= \iint_e (-\tau^\alpha \frac{1}{4\Delta_e^2} \sum_{k=1}^{n} \lambda_k^\alpha ((a_i a_j + b_i b_j)u_{e,i}^{n-k} + (a_j^2 + b_j^2)u_{e,j}^{n-k} + (a_j a_m + b_j b_m)u_{e,m}^{n-k})\\
&\quad + \frac{2}{\tau}(N_i N_j u_{e,i}^{n-1} + N_j N_j u_{e,j}^{n-1} + N_j N_m u_{e,m}^{n-1})\\
&\quad - \frac{1}{2\tau}(N_i N_j u_{e,i}^{n-2} + N_j N_j u_{e,j}^{n-2} + N_j N_m u_{e,m}^{n-2}) + g^n N_j)\mathrm{d}x\mathrm{d}y,\\
b^{(e)}_{2m} &= \iint_e (-\tau^\alpha \frac{1}{4\Delta_e^2} \sum_{k=1}^{n} \lambda_k^\alpha ((a_i a_m + b_i b_m)u_{e,i}^{n-k} + (a_j a_m + b_j b_m)u_{e,j}^{n-k} + (a_m^2 + b_m^2)u_{e,m}^{n-k})\\
&\quad + \frac{2}{\tau}(N_i N_m u_{e,i}^{n-1} + N_j N_m u_{e,j}^{n-1} + N_m N_m u_{e,m}^{n-1})\\
&\quad - \frac{1}{2\tau}(N_i N_m u_{e,i}^{n-2} + N_j N_m u_{e,j}^{n-2} + N_m N_m u_{e,m}^{n-2}) + g^n N_m)\mathrm{d}x\mathrm{d}y.
\end{aligned} \tag{48}$$

Based on system (45), we can obtain the unique numerical solution of u and σ.

5.1.2. Numerical Calculations

For reflecting the effectiveness of the considered numerical method in the current article, two numerical examples with initial and boundary conditions are provided. In these tests, we take $z = (x, y) \in \Omega = (0, 1) \times (0, 1)$ and $T = 1$.

Example 1. *We take the exact solution of (1) as* $u(x,y,t) = t^3(\sin \pi x)y^3(1-y)^3$ *and then substitute it into (2) to find*

$$ {}_0\mathcal{I}_t^{\alpha} u = \frac{6(\sin \pi x)y^3(1-y)^3}{\Gamma(\alpha+4)} t^{\alpha+3}. \tag{49} $$

Now, we take the nonlinear term $f(u) = \sin u$ and obtain the corresponding source term $g(x,y,t)$. Here, we show the effectiveness by the calculated data in Tables 1 and 2.

For the fractional parameters $\alpha = 0.1, 0.3, 0.5, 0.7, 0.9$, we fix the time step length parameter $\tau = 1/200$ and change the spatial grid parameters $h = 1/4, 1/8, 1/16, 1/32$ to arrive at the spatial convergence results for both u and σ in Table 1. Further, by taking $(\tau, h) = (\frac{1}{10}, \frac{1}{10}), (\frac{1}{20}, \frac{1}{20}), (\frac{1}{30}, \frac{1}{30}), (\frac{1}{40}, \frac{1}{40})$, we calculate the space–time convergence results shown in Table 2. The computed data illustrate that we can arrive at the approximating second-order convergence rate, which is in agreement with our theory results. In Figures 1–4, we show the approximate process between u_h and u by taking the fractional parameter $\alpha = 0.1$ and space–time step length parameters $\tau = h = 1/20, 1/30$ and $1/40$. We also show the approximation behavior between σ_h and σ in Figures 5–8.

Table 1. The spatial convergence results for u and σ with $\tau = \frac{1}{200}$.

α	h	$E_u(\tau,h)$	Rate	$E_\sigma(\tau,h)$	Rate
0.1	$1/4$	1.2160×10^{-3}	—	7.3073×10^{-2}	—
	$1/8$	3.0300×10^{-4}	2.0047	1.8151×10^{-2}	2.0093
	$1/16$	7.5223×10^{-5}	2.0101	4.7932×10^{-3}	1.9210
	$1/32$	1.8735×10^{-5}	2.0054	1.1330×10^{-3}	2.0809
0.3	$1/4$	1.2189×10^{-3}	—	7.3130×10^{-2}	—
	$1/8$	3.0367×10^{-4}	2.0050	1.8170×10^{-2}	2.0089
	$1/16$	7.5369×10^{-5}	2.0104	4.7981×10^{-3}	1.9210
	$1/32$	1.8771×10^{-5}	2.0055	1.1342×10^{-3}	2.0807
0.5	$1/4$	1.2212×10^{-3}	—	7.3175×10^{-2}	—
	$1/8$	3.0420×10^{-4}	2.0052	1.8185×10^{-2}	2.0086
	$1/16$	7.5486×10^{-5}	2.0107	4.8020×10^{-3}	1.9210
	$1/32$	1.8799×10^{-5}	2.0055	1.1353×10^{-3}	2.0806
0.7	$1/4$	1.2230×10^{-3}	—	7.3210×10^{-2}	—
	$1/8$	3.0463×10^{-4}	2.0054	1.8197×10^{-2}	2.0084
	$1/16$	7.5579×10^{-5}	2.0110	4.8052×10^{-3}	1.9210
	$1/32$	1.8821×10^{-5}	2.0056	1.1361×10^{-3}	2.0806
0.9	$1/4$	1.2245×10^{-3}	—	7.3238×10^{-2}	—
	$1/8$	3.0496×10^{-4}	2.0055	1.8206×10^{-2}	2.0082
	$1/16$	7.5651×10^{-5}	2.0112	4.8076×10^{-3}	1.9210
	$1/32$	1.8839×10^{-5}	2.0057	1.1367×10^{-3}	2.0805

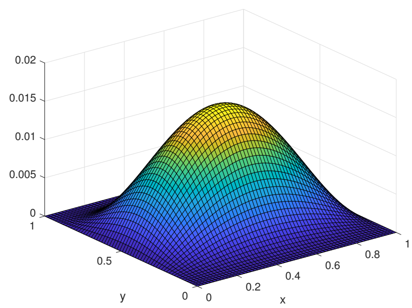

Figure 1. Exact solution u at $t = 1$.

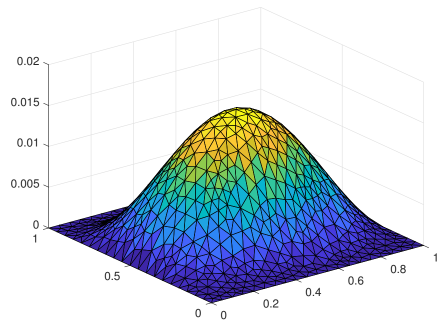

Figure 2. u_h with $\tau = h = \frac{1}{20}$.

Table 2. The space–time convergence results for u and σ.

α	(τ, h)	$E_u(\tau, h)$	Rate	$E_\sigma(\tau, h)$	Rate
0.1	(1/10,1/10)	2.0896×10^{-4}	—	1.3569×10^{-2}	—
	(1/20,1/20)	4.6046×10^{-5}	2.1821	3.0399×10^{-3}	2.1582
	(1/30,1/30)	2.0672×10^{-5}	1.9751	1.3515×10^{-3}	1.9993
	(1/40,1/40)	1.1502×10^{-5}	2.0381	7.4700×10^{-4}	2.0609
0.3	(1/10,1/10)	2.0955×10^{-4}	—	1.3585×10^{-2}	—
	(1/20,1/20)	4.6178×10^{-5}	2.1820	3.0437×10^{-3}	2.1582
	(1/30,1/30)	2.0731×10^{-5}	1.9753	1.3532×10^{-3}	1.9992
	(1/40,1/40)	1.1535×10^{-5}	2.0377	7.4796×10^{-4}	2.0609
0.5	(1/10,1/10)	2.0995×10^{-4}	—	1.3597×10^{-2}	—
	(1/20,1/20)	4.6263×10^{-5}	2.1821	3.0465×10^{-3}	2.1581
	(1/30,1/30)	2.0768×10^{-5}	1.9754	1.3545×10^{-3}	1.9991
	(1/40,1/40)	1.1557×10^{-5}	2.0374	7.4866×10^{-4}	2.0609
0.7	(1/10,1/10)	2.1018×10^{-4}	—	1.3606×10^{-2}	—
	(1/20,1/20)	4.6309×10^{-5}	2.1822	3.0484×10^{-3}	2.1581
	(1/30,1/30)	2.0788×10^{-5}	1.9754	1.3554×10^{-3}	1.9990
	(1/40,1/40)	1.1568×10^{-5}	2.0373	7.4916×10^{-4}	2.0609
0.9	(1/10,1/10)	2.1028×10^{-4}	—	1.3612×10^{-2}	—
	(1/20,1/20)	4.6327×10^{-5}	2.1824	3.0498×10^{-3}	2.1581
	(1/30,1/30)	2.0795×10^{-5}	1.9755	1.3560×10^{-3}	1.9990
	(1/40,1/40)	1.1573×10^{-5}	2.0373	7.4950×10^{-4}	2.0609

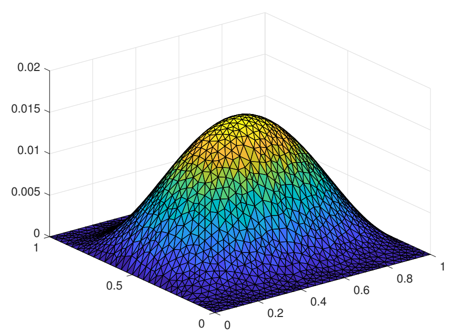

Figure 3. u_h with $\tau = h = \frac{1}{30}$.

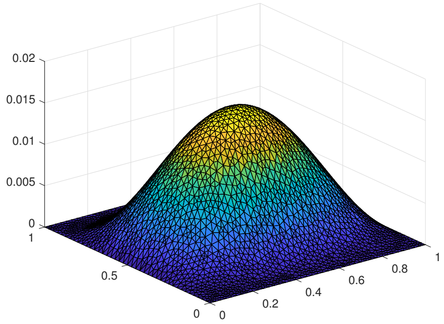

Figure 4. u_h with $\tau = h = \frac{1}{40}$.

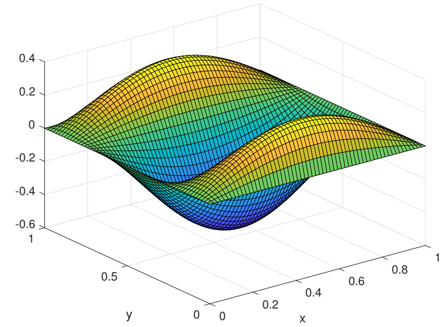

Figure 5. Exact solution σ at $t = 1$.

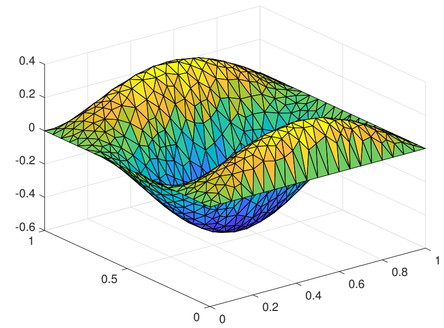

Figure 6. σ_h with $\tau = h = \frac{1}{20}$.

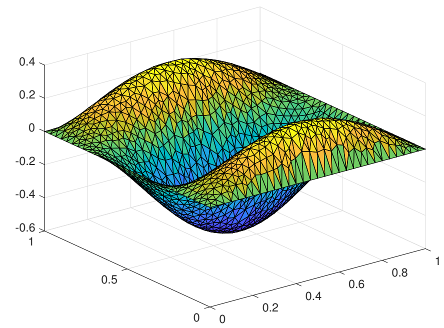

Figure 7. σ_h with $\tau = h = \frac{1}{30}$.

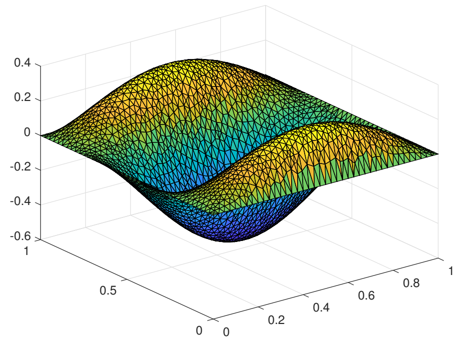

Figure 8. σ_h with $\tau = h = \frac{1}{40}$.

Example 2. *Now, we consider another numerical example to further validate the convergence for our method. We take the exact solution* $u(x, y, t) = t^{2.5}x^3(1-x)^3y^3(1-y)^3$ *and then obtain*

$$ {}_0\mathcal{I}_t^{\alpha} u = \frac{\Gamma(3.5)}{\Gamma(\alpha+3.5)} x^3(1-x)^3 y^3(1-y)^3 t^{\alpha+2.5}. \tag{50} $$

Next, we choose the nonlinear term $f(u) = u^3 - u$ to obtain the corresponding source term $g(x, y, t)$. In this case, we continue to do our tests.

Here, we list the computing data including errors and convergence orders in Tables 3 and 4 and the approximation behaviors between the numerical solution and the exact solution in Figures 9–12 with the chosen same fractional parameter α and space–time step length sizes to the ones used in Example 1. From Tables 3 and 4 and Figures 13–16, it is easy to see that the approximation effect is consistent with the theoretical results.

To further show the advantages of our method, we need to make a comparison with other numerical schemes. Now, we compute the Example 2 by using another method presented in Remark 1 and obtain the numerical results shown in Table 5. By the comparison of errors $E_u(\tau, h)$ and $E_\sigma(\tau, h)$ between Tables 3 and 5, one easily finds that our method can obtain the better calculation accuracy.

Table 3. The spatial convergence results for u and σ with $\tau = \frac{1}{200}$.

α	h	$E_u(\tau, h)$	Rate	$E_\sigma(\tau, h)$	Rate
0.1	1/4	1.9466×10^{-5}	—	1.4787×10^{-3}	—
	1/8	4.8028×10^{-6}	2.0190	3.2835×10^{-4}	2.1711
	1/16	1.2213×10^{-6}	1.9755	8.8243×10^{-5}	1.8957
	1/32	3.1213×10^{-7}	1.9682	2.1371×10^{-5}	2.0458
0.3	1/4	1.9495×10^{-5}	—	1.4795×10^{-3}	—
	1/8	4.8101×10^{-6}	2.0190	3.2864×10^{-4}	2.1705
	1/16	1.2229×10^{-6}	1.9758	8.8322×10^{-5}	1.8957
	1/32	3.1254×10^{-7}	1.9682	2.1392×10^{-5}	2.0457
0.5	1/4	1.9520×10^{-5}	—	1.4801×10^{-3}	—
	1/8	4.8163×10^{-6}	2.0190	3.2889×10^{-4}	2.1701
	1/16	1.2242×10^{-6}	1.9760	8.8388×10^{-5}	1.8957
	1/32	3.1289×10^{-7}	1.9681	2.1409×10^{-5}	2.0456
0.7	1/4	1.9541×10^{-5}	—	1.4806×10^{-3}	—
	1/8	4.8213×10^{-6}	2.0190	3.2909×10^{-4}	2.1697
	1/16	1.2253×10^{-6}	1.9763	8.8442×10^{-5}	1.8957
	1/32	3.1317×10^{-7}	1.9681	2.1423×10^{-5}	2.0456
0.9	1/4	1.9557×10^{-5}	—	1.4811×10^{-3}	—
	1/8	4.8254×10^{-6}	2.0190	3.2925×10^{-4}	2.1694
	1/16	1.2262×10^{-6}	1.9764	8.8485×10^{-5}	1.8957
	1/32	3.1340×10^{-7}	1.9681	2.1434×10^{-5}	2.0455

Table 4. The space–time convergence results for u and σ.

α	(τ, h)	$E_u(\tau, h)$	Rate	$E_\sigma(\tau, h)$	Rate
0.1	(1/10,1/10)	3.6207×10^{-6}	—	2.5835×10^{-4}	—
	(1/20,1/20)	8.4923×10^{-7}	2.0920	5.9985×10^{-5}	2.1066
	(1/30,1/30)	3.7741×10^{-7}	2.0002	2.6440×10^{-5}	2.0204
	(1/40,1/40)	2.0890×10^{-7}	2.0561	1.4528×10^{-5}	2.0816
0.3	(1/10,1/10)	3.6281×10^{-6}	—	2.5859×10^{-4}	—
	(1/20,1/20)	8.5093×10^{-7}	2.0921	6.0046×10^{-5}	2.1065
	(1/30,1/30)	3.7812×10^{-7}	2.0005	2.6468×10^{-5}	2.0203
	(1/40,1/40)	2.0931×10^{-7}	2.0556	1.4543×10^{-5}	2.0816
0.5	(1/10,1/10)	3.6338×10^{-6}	—	2.5879×10^{-4}	—
	(1/20,1/20)	8.5222×10^{-7}	2.0922	6.0095×10^{-5}	2.1065
	(1/30,1/30)	3.7865×10^{-7}	2.0008	2.6490×10^{-5}	2.0203
	(1/40,1/40)	2.0963×10^{-7}	2.0553	1.4555×10^{-5}	2.0816
0.7	(1/10,1/10)	3.6379×10^{-6}	—	2.5894×10^{-4}	—
	(1/20,1/20)	8.5315×10^{-7}	2.0923	6.0133×10^{-5}	2.1064
	(1/30,1/30)	3.7903×10^{-7}	2.0010	2.6508×10^{-5}	2.0202
	(1/40,1/40)	2.0986×10^{-7}	2.0550	1.4565×10^{-5}	2.0816
0.9	(1/10,1/10)	3.6407×10^{-6}	—	2.5905×10^{-4}	—
	(1/20,1/20)	8.5375×10^{-7}	2.0923	6.0162×10^{-5}	2.1063
	(1/30,1/30)	3.7927×10^{-7}	2.0011	2.6521×10^{-5}	2.0202
	(1/40,1/40)	2.1000×10^{-7}	2.0549	1.4572×10^{-5}	2.0816

Table 5. The spatial convergence rate for u and σ with $\tau = \frac{1}{200}$.

α	h	$E_u(\tau, h)$	Rate	$E_\sigma(\tau, h)$	Rate
0.1	1/4	2.5596×10^{-5}	—	1.8764×10^{-3}	—
	1/8	5.6306×10^{-6}	2.1845	4.2919×10^{-4}	2.1283
	1/16	1.4004×10^{-6}	2.0075	1.0788×10^{-4}	1.9922
	1/32	3.5652×10^{-7}	1.9738	2.7133×10^{-5}	1.9913
0.3	1/4	2.5633×10^{-5}	—	1.8737×10^{-3}	—
	1/8	5.6380×10^{-6}	2.1847	4.2868×10^{-4}	2.1279
	1/16	1.4020×10^{-6}	2.0077	1.0776×10^{-4}	1.9921
	1/32	3.5693×10^{-7}	1.9738	2.7104×10^{-5}	1.9913
0.5	1/4	2.5664×10^{-5}	—	1.8714×10^{-3}	—
	1/8	5.6443×10^{-6}	2.1849	4.2826×10^{-4}	2.1275
	1/16	1.4033×10^{-6}	2.0079	1.0766×10^{-4}	1.9920
	1/32	3.5727×10^{-7}	1.9738	2.7079×10^{-5}	1.9913
0.7	1/4	2.5689×10^{-5}	—	1.8695×10^{-3}	—
	1/8	5.6494×10^{-6}	2.1850	4.2792×10^{-4}	2.1272
	1/16	1.4044×10^{-6}	2.0081	1.0758×10^{-4}	1.9919
	1/32	3.5755×10^{-7}	1.9738	2.7059×10^{-5}	1.9912
0.9	1/4	2.5710×10^{-5}	—	1.8680×10^{-3}	—
	1/8	5.6535×10^{-6}	2.1851	4.2765×10^{-4}	2.1270
	1/16	1.4053×10^{-6}	2.0083	1.0752×10^{-4}	1.9919
	1/32	3.5777×10^{-7}	1.9738	2.7043×10^{-5}	1.9912

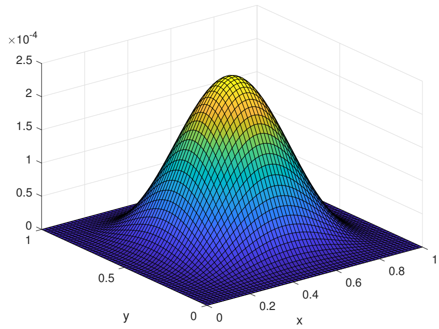

Figure 9. Exact solution u at $t = 1$.

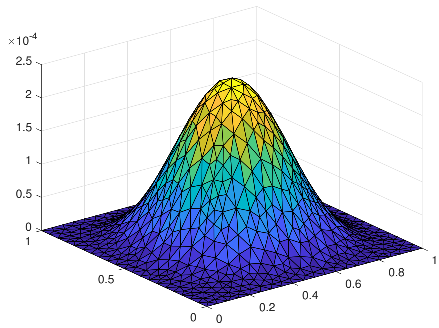

Figure 10. u_h with $\tau = h = \frac{1}{20}$.

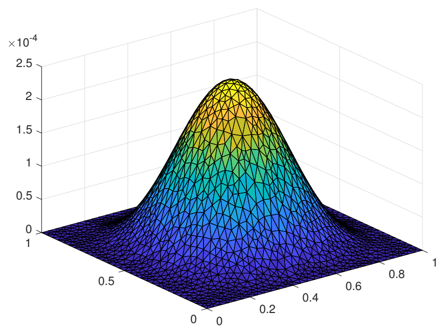

Figure 11. u_h with $\tau = h = \frac{1}{30}$.

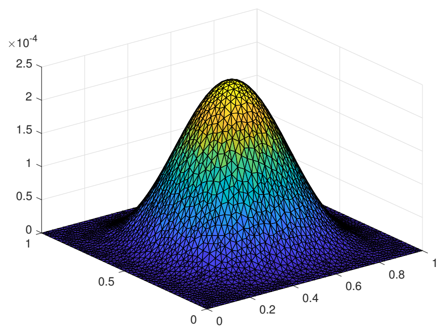

Figure 12. u_h with $\tau = h = \frac{1}{40}$.

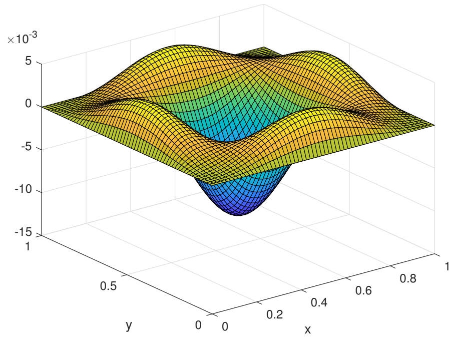

Figure 13. Exact solution σ at $t = 1$.

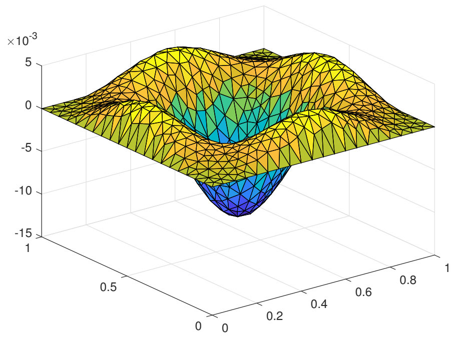

Figure 14. σ_h with $\tau = h = \frac{1}{20}$.

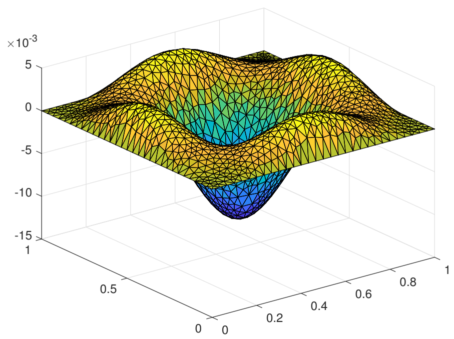

Figure 15. σ_h with $\tau = h = \frac{1}{30}$.

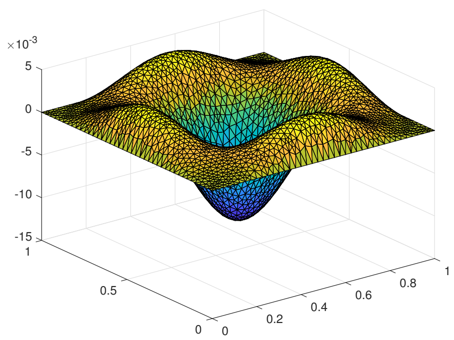

Figure 16. σ_h with $\tau = h = \frac{1}{40}$.

6. Conclusions

In this article, we developed a fully discrete mixed element system with a second-order time stepping scheme to numerically solve 2D nonlinear fourth-order fractional integral equations. By computing numerical data, including errors and convergence orders, we found that the proposed fully discrete mixed element system was feasible.

Author Contributions: Conceptualization, D.W.; methodology, Y.L. and H.L.; software, D.W.; validation, Y.L., H.L. and Z.F.; formal analysis, D.W. and Y.L.; writing—original draft preparation, D.W.; writing—review and editing, Y.L., H.L. and Z.F.; funding acquisition, D.W., Y.L. and H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by Natural Science Foundation of Inner Mongolia (2020MS01003, 2021MS01018), Young Innovative Talents Project of Grassland Talents Project, Program for Innovative Research Team in Universities of Inner Mongolia Autonomous Region (NMGIRT2207) and National Innovation Project (202010126022).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All the data were computed using our algorithm.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Abbaszadeh, M.; Dehghan, M. Direct meshless local Petrov-Galerkin (DMLPG) method for time-fractional fourth-order reaction-diffusion problem on complex domains. *Comput. Math. Appl.* **2020**, *79*, 876–888. [CrossRef]
2. Liu, Y.; Du, Y.W.; Li, H.; Li, J.C.; He, S. A two-grid mixed finite element method for a nonlinear fourth-order reaction-diffusion problem with time-fractional derivative. *Comput. Math. Appl.* **2015**, *70*, 2474–2492. [CrossRef]
3. Nikan, O.; Machado, J.A.T.; Golbabai, A. Numerical solution of time-fractional fourth-order reaction-diffusion model arising in composite environments. *Appl. Math. Model.* **2021**, *89*, 819–836. [CrossRef]
4. Ji, C.C.; Sun, Z.Z.; Hao, Z.P. Numerical algorithms with high spatial accuracy for the fourth-order fractional sub-diffusion equations with the first Dirichlet boundary conditions. *J. Sci. Comput.* **2016**, *66*, 1148–1174. [CrossRef]
5. Haghi, M.; Ilati, M.; Dehghan, M. A fourth-order compact difference method for the nonlinear time-fractional fourth-order reaction-diffusion equation. *Eng. Comput.* **2021** , 1–12. [CrossRef]
6. Wang, J.R.; Liu, Y.; Wen, C.; Li, H. Efficient numerical algorithm with the second-order time accuracy for a two-dimensional nonlinear fourth-order fractional wave equation. *Results Appl. Math.* **2022**.
7. Li, X.H.; Wong, P.J.Y. A non-polynomial numerical scheme for fourth-order fractional diffusion-wave model. *Appl. Math. Comput.* **2018**, *331*, 80–95. [CrossRef]
8. Akram, G.; Abbas, M.; Tariq, H.; Sadaf, M.; Abdeljawad, T.; Alqudah, M.A. Numerical approximations for the solutions of fourth order time fractional evolution problems using a novel spline technique. *Fractal Fract.* **2022**, *6*, 170. [CrossRef]
9. Fakhar-Izadi, F.; Shabgard, N. Time-space spectral Galerkin method for time-fractional fourth-order partial differential equations. *J. Appl. Math. Comput.* **2022** , 1–20. [CrossRef]
10. Arshad, S.; Wali, M.; Huang, J.F.; Khalid, S.; Akbar, N. Numerical framework for the Caputo time-fractional diffusion equation with fourth order derivative in space. *J. Appl. Math. Comput.* **2021** , 1–22. [CrossRef]
11. Nandal, S.; Pandey, D.N. Numerical technique for fractional variable-order differential equation of fourth-order with delay. *Appl. Numer. Math.* **2021**, *161*, 391–407. [CrossRef]
12. Ran, M.H.; Zhang, C.J. New compact difference scheme for solving the fourth-order time fractional sub-diffusion equation of the distributed order. *Appl. Numer. Math.* **2018**, *129*, 58–70. [CrossRef]
13. Nawaz, Y. Variational iteration method and homotopy perturbation method for fourth-order fractional integro-differential equations. *Comput. Math. Appl.* **2011**, *61*, 2330–2341. [CrossRef]
14. Yang, X.H.; Xu, D.; Zhang, H.X. Crank-Nicolson/quasi-wavelets method for solving fourth order partial integro-differential equation with a weakly singular kernel. *J. Comput. Phys.* **2013**, *234*, 317–329. [CrossRef]
15. Liu, Y.L.; Yang, X.H.; Zhang, H.X.; Liu, Y. Analysis of BDF2 finite difference method for fourth-order integro-differential equation. *Comput. Appl. Math.* **2021**, *40*, 57. [CrossRef]
16. Qiu, W.L.; Xu, D.; Guo, J. Numerical solution of the fourth-order partial integro-differential equation with multi-term kernels by the Sinc-collocation method based on the double exponential transformation. *Appl. Math. Comput.* **2021**, *392*, 125693. [CrossRef]
17. Wang, Z.B.; Vong, S. Compact difference schemes for the modified anomalous fractional sub-diffusion equation and the fractional diffusion-wave equation. *J. Comput. Phys.* **2014**, *277*, 1–15. [CrossRef]

18. Tian, W.Y.; Zhou, H.; Deng, W.H. A class of second order difference approximations for solving space fractional diffusion equations. *Math. Comput.* **2015**, *84*, 1703–1727. [CrossRef]
19. Feng, R.H.; Liu, Y.; Hou, Y.X.; Li, H.; Fang, Z.C. Mixed element algorithm based on a second-order time approximation scheme for a two-dimensional nonlinear time fractional coupled sub-diffusion model. *Eng. Comput.* **2022**, *38*, 51–68. [CrossRef]
20. Cao, Y.; Yin, B.L.; Liu, Y.; Li, H. Crank-Nicolson WSGI difference scheme with finite element method for multi-dimensional time-fractional wave problem. *Comput. Appl. Math.* **2018**, *37*, 5126–5145. [CrossRef]