Element-free Galerkin (EFG) method for analysis of the time-fractional partial differential equations*

Ge Hong-Xia(葛红霞)[a], Liu Yong-Qing(刘永庆)[a], and Cheng Rong-Jun(程荣军)[b]†

[a] *Faculty of Science, Ningbo University, Ningbo* 315211, *China*

[b] *Ningbo Institute of Technology, Zhejiang University, Ningbo* 315100, *China*

(Received 21 July 2011; revised manuscript received 6 September 2011)

The present paper deals with the numerical solution of time-fractional partial differential equations using the element-free Galerkin (EFG) method, which is based on the moving least-square approximation. Compared with numerical methods based on meshes, the EFG method for time-fractional partial differential equations needs only scattered nodes instead of meshing the domain of the problem. It neither requires element connectivity nor suffers much degradation in accuracy when nodal arrangements are very irregular. In this method, the first-order time derivative is replaced by the Caputo fractional derivative of order α ($0 < \alpha \leq 1$). The Galerkin weak form is used to obtain the discrete equations, and the essential boundary conditions are enforced by the penalty method. Several numerical examples are presented and the results we obtained are in good agreement with the exact solutions.

Keywords: element-free Galerkin (EFG) method, meshless method, time fractional partial differential equations

PACS: 02.60.Lj, 03.65.Ge **DOI:** 10.1088/1674-1056/21/1/010206

1. Introduction

Fractional order differential equations are playing an increasingly important role in engineering, physics and other fields, such as the fractal theory and diffusion in porous media, electrolysis chemical and fractional capacitance theory, condensate physics, vibration control of viscoelastic systems, fractional biological neurons, statistical mechanics and so on. Therefore, fractional order partial differential equations of applied research have attracted wider attention. Recently, fractional order partial differential equations have been solved including the fractional KdV equations,[1] a finite difference method for fractional partial differential equations,[2,3] space–time-fractional partial differential equations[4−6] and fractional partial differential equations in fluid mechanics.[7]

The meshless method is a new, efficient numerical simulation technique that can solve many engineering problems. The element-free Galerkin (EFG) method is a common meshless method. In this method, the function over the solution domain needs only a set of nodes and does not require element connectivity. The variational method has been used to obtain the corresponding discrete equations, and the essential boundary conditions have been enforced by the penalty method. The EFG method has been greatly developed and is widely used in various fields, for example, to solve the linear hyperbolic equation and the KdV equations,[8,9] for simulating stationary shallow water flows in rivers,[10] analyzing thin plates,[11] and others.[12−27] However, we find that this method has not been used to solve fractional order differential equation. In the present paper, the EFG method is used to obtain the numerical solution of the time-fractional order differential equation.

We consider the following time-fractional partial differential equation in the present paper:

$$\frac{\partial^\alpha u}{\partial t^\alpha} + \beta\frac{\partial u}{\partial x} + \gamma\frac{\partial^2 u}{\partial x^2} = f(x,t), \tag{1}$$

where β, γ are real parameters and $t > 0$, $0 < \alpha \leq 1$, $a \leq x \leq b$. The Dirichlet boundary conditions and the initial condition are respectively

$$u(a,t) = \overline{u_1}(t), \quad u(b,t) = \overline{u_2}(t), \tag{2}$$

$$u(x,0) = u_0(x). \tag{3}$$

The structure of this paper is as follows: in Section 2, we describe the EFG method. In Section 3,

*Project supported by the National Natural Science Foundation of China (Grant No. 11072117), the Natural Science Foundation of Zhejiang Province, China (Grant Nos. Y6110007 and Y6110502), and the K.C.Wong Magna Fund in Ningbo University, China.

†Corresponding author. E-mail: chengrongjun76@126.com

© 2012 Chinese Physical Society and IOP Publishing Ltd

using the EFG method, the equation of the problem is discretized. To present a clear overview of the method, in Section 4, some examples are given and solved. Section 5 ends this article with our conclusion.

2. Moving least-square approximation

The EFG method, introduced by Belytschko *et al.*,[28] requires the moving least-square (MLS) approximation, which is made up of three components: a weight function of compact support associated with each node, a monomial basis and a set of coefficients that depend on node position.

In order to obtain $u^h(x)$, which is the approximation function of $u(x)$, Lancaster and Salkauskas[29] defined a local approximation by

$$u^h(x,\overline{x}) = \sum_{k=1}^{m} p_k(x)a_k(\overline{x}) \equiv \boldsymbol{P}^{\mathrm{T}}(x)\boldsymbol{a}(\overline{x}), \tag{4}$$

where m is the number of terms in the basis, $p_k(x)$ are monomial basis functions and $a_k(\overline{x})$ are unknown coefficients, which are functions of the spatial coordinates x. The $\boldsymbol{P}^{\mathrm{T}}(x)$ is a vector composed of $p_k(x)$, and $\boldsymbol{a}(\overline{x})$ is also the vector form of $a_k(\overline{x})$.

To obtain the unknown coefficients $\boldsymbol{a}(\overline{x})$, we define the weighted discrete L_2 norm as

$$\begin{aligned}\boldsymbol{J} &= \sum_{i=1}^{N} w(x-x_i)[u^h(x_i,\overline{x}) - u_i]^2 \\ &= \sum_{i=1}^{N} w(x-x_i)[\boldsymbol{P}^{\mathrm{T}}(x_i)\boldsymbol{a}(\overline{x}) - u_i]^2, \end{aligned}\tag{5}$$

where $w(x-x_i)$ is a weight function, x_i is the node, N is the number of nodes in the compacted supported domain of x and u_i is the nodal parameter of u at $x = x_i$.

Then by minimizing $\boldsymbol{J}$ in Eq. (5), we have

$$\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{a}} = \boldsymbol{A}(x)\boldsymbol{a}(x) - \boldsymbol{B}(x)\boldsymbol{U} = \boldsymbol{0}, \tag{6}$$

where

$$\boldsymbol{A}(x) = \sum_{i=1}^{N} w(x-x_i)\boldsymbol{P}(x_i)\boldsymbol{P}^{\mathrm{T}}(x_i), \tag{7}$$

$$\begin{aligned}\boldsymbol{B}(x) = [&w(x-x_1)\boldsymbol{P}(x_1),\ w(x-x_2)\boldsymbol{P}(x_2),\ ..., \\ &w(x-x_N)\boldsymbol{P}(x_N)],\end{aligned} \tag{8}$$

$$\boldsymbol{U} = [u_1,\ u_2,\ ...,\ u_N]^{\mathrm{T}}. \tag{9}$$

Equation (6) is equal to

$$\boldsymbol{a}(x) = \boldsymbol{A}^{-1}(x)\boldsymbol{B}(x)\boldsymbol{U}. \tag{10}$$

According to Eqs. (4) and (10), the approximate function of $u(x)$ can be obtained as

$$\begin{aligned} u(x) &\approx u^h(x) \\ &= \sum_{k=1}^{m} p_k(x)a_k(x) \\ &= \sum_{i=1}^{N} \phi_i u_i = \boldsymbol{\Phi}\boldsymbol{U}, \end{aligned}\tag{11}$$

where $\boldsymbol{\Phi}$ is a matrix of shape function, defined as

$$\begin{aligned}\boldsymbol{\Phi} &= [\phi_1(x),\ \phi_2(x),\ ...,\ \phi_N(x)] \\ &= \boldsymbol{P}^{\mathrm{T}}(x)\boldsymbol{A}^{-1}(x)\boldsymbol{B}(x). \end{aligned}\tag{12}$$

The derivative of the unknown $u(x)$ is approximated by

$$\frac{\partial^q}{\partial x^q}u^h(x) = \sum_{i=1}^{N}\left[\frac{\partial^q}{\partial x^q}\phi_i(x)\right]u_i. \tag{13}$$

The weight function used in the MLS plays a very important role in the EFG method, so the selection of an appropriate weight function is essential. In Eq. (6), $w(x-x_i)$ is the weight function, which should be greater than zero within only a small neighborhood of x_i, called the domain of influence of node x_i. In the rest of the solution domain, the weight function should be equal to zero. The cubic spline weight function used in the present analysis can be written as a function of normalized radius r as follows:

$$\begin{aligned} &w(x-x_i) \\ &= w(r) \\ &= \begin{cases} \dfrac{2}{3} - 4r^2 + 4r^3, & \left(r \le \dfrac{1}{2}\right), \\ \dfrac{4}{3} - 4r + 4r^2 - \dfrac{4}{3}r^3, & \left(\dfrac{1}{2} < r \le 1\right), \\ 0, & (r > 1), \end{cases} \end{aligned}\tag{14}$$

with

$$r = \frac{d_i}{d_{m_i}},\quad d_i = \|x - x_i\|,\quad d_{m_i} = d_{\mathrm{scale}}c_i, \tag{15}$$

where d_{scale} is a scaling parameter, the distance c_i is chosen so that the matrix $\boldsymbol{A}(x)$ is not singular.

3. Numerical scheme

3.1. Time-fractional derivative approximation

To establish the numerical approximation scheme, we consider N points: $x_i = (i-1)\Delta x, i = 1, 2, 3, \cdots, N$, where Δx is the spatial step size and $\Delta x = (b-a)/(N-1)$. $x_1 = a$, $x_N = b$ are the boundary points. The grid points in the time interval $[0, T]$ are labeled as $t_j = n\Delta t, n = 0, 1, 2, 3, \cdots$, where Δt is the time interval.

In Eq. (1), the approximation of $\partial^\alpha u(x,t)/(\partial t^\alpha)$ $(0 < \alpha < 1)$ can be obtained by a simple quadrature formula[30]

$$\begin{aligned}
\frac{\partial^\alpha u(x,t_{n+1})}{\partial t^\alpha} &= \frac{1}{\Gamma(1-\alpha)}\int_0^{t_{n+1}} \frac{\partial u(x,s)}{\partial s}\frac{\mathrm{d}s}{(t_{n+1}-s)^\alpha} \\
&= \frac{1}{\Gamma(1-\alpha)}\sum_{j=0}^{n}\left(\frac{u(x,t_{j+1})-u(x,t_j)}{\Delta t}+O(\Delta t)\right)\int_{t_j}^{t_{j+1}}\frac{\mathrm{d}s}{(t_{n+1}-s)^\alpha} \\
&= \frac{1}{\Gamma(1-\alpha)}\sum_{j=0}^{n}\left(\frac{u(x,t_{j+1})-u(x,t_j)}{\Delta t}+O(\Delta t)\right)\int_{t_{n-j}}^{t_{n-j+1}}\frac{\mathrm{d}t}{(t)^\alpha} \\
&= \frac{1}{\Gamma(1-\alpha)}\sum_{j=0}^{n}\left(\frac{u(x,t_{n-j+1})-u(x,t_{n-j})}{\Delta t}+O(\Delta t)\right)\int_{t_j}^{t_{j+1}}\frac{\mathrm{d}t}{(t)^\alpha} \\
&= \frac{1}{\Gamma(2-\alpha)}\sum_{j=0}^{n}\frac{u(x,t_{n-j+1})-u(x,t_{n-j})}{(\Delta t)^\alpha}[(j+1)^{1-\alpha}-j^{1-\alpha}]+O((\Delta t)^{2-\alpha}) \\
&= Aa[u(x,t_{n+1})-u(x,t_n)]+Aa\sum_{j=1}^{n}Ba(j)[u(x,t_{n-j+1})-u(x,t_{n-j})]+O((\Delta t)^{2-\alpha}), \quad (16)
\end{aligned}$$

where

$$Aa = \frac{1}{\Gamma(2-\alpha)(\Delta t)^\alpha},\ \ Ba(j) = (j+1)^{1-\alpha} - j^{1-\alpha},$$

where $\Gamma(.)$ is the gamma function.

3.2. EFG method scheme

Now, the EFG method will be implemented to obtaining the discrete equation. To obtain the discrete equation, we use a weak form of the equation and employ the penalty function method to deal with the boundary conditions.

By taking variation, the weak form of Eq. (1) is

$$\begin{aligned}
&\int \delta u\left(\frac{\partial^\alpha u}{\partial t^\alpha}+\beta\frac{\partial u}{\partial x}+\gamma\frac{\partial^2 u}{\partial x^2}-f(x,\ t)\right)\mathrm{d}\Gamma \\
&+\varepsilon\delta u(u-\overline{u})|\Gamma_u = 0, \quad (17)
\end{aligned}$$

where Γ refers to the domain of x, Γ_u refers to the boundaries, $\overline{u}$ are the values of u in the boundaries, and ε is a penalty parameter and usually equals 10^3–10^8.

Equation (17) also can be written as

$$\begin{aligned}
&\int \delta u\frac{\partial^\alpha u}{\partial t^\alpha}\mathrm{d}\Gamma - \int\gamma\frac{\partial(\delta u)}{\partial x}\frac{\partial u}{\partial x}\mathrm{d}\Gamma + \int\beta\delta u\frac{\partial u}{\partial x}\mathrm{d}\Gamma \\
&-\int\delta u f(x,\ t)\mathrm{d}\Gamma+\varepsilon\delta u(u-\overline{u})|\Gamma_u = 0. \quad (18)
\end{aligned}$$

According to Eqs. (11) and (13), we can obtain the approximations of u^h, $\partial u^h/(\partial x)$ and $\partial^\alpha u^h/(\partial t^\alpha)$ as

$$u^h = \sum_{i=1}^{N}\phi_i(x)u_i(t) = \boldsymbol{\Phi}(x)\boldsymbol{U}(t), \quad (19)$$

$$\frac{\partial u^h}{\partial x} = \sum_{i=1}^{N}\frac{\partial\phi_i(x)}{\partial x}u_i(t) = \boldsymbol{\Phi}_x(x)\boldsymbol{U}(t), \quad (20)$$

$$\begin{aligned}
&\frac{\partial^\alpha u^h(x,t_{n+1})}{\partial t^\alpha} \\
&= \boldsymbol{\Phi}(x)[Aa(\boldsymbol{U}(t_{n+1})-\boldsymbol{U}(t_n)) \\
&\quad + Aa\sum_{j=1}^{n}Ba(j)(\boldsymbol{U}(t_{n-j+1})-\boldsymbol{U}(t_{n-j}))], \quad (21)
\end{aligned}$$

where

$$\boldsymbol{U}(t) = [u_1(t),\ u_2(t),\ u_3(t),\ \cdots, u_N(t)]^{\mathrm{T}}, \quad (22)$$

$$\boldsymbol{\Phi}(x) = [\phi_1(x),\ \phi_2(x),\ \phi_3(x), \cdots, \phi_N(x)], \quad (23)$$

$$\begin{aligned}
&\boldsymbol{\Phi}_x(x) \\
&= \frac{\partial\boldsymbol{\Phi}(x)}{\partial x} \\
&= \left[\frac{\partial\phi_1(x)}{\partial x}, \frac{\partial\phi_2(x)}{\partial x}, \frac{\partial\phi_3(x)}{\partial x}, \cdots, \frac{\partial\phi_N(x)}{\partial x}\right] \\
&= [\phi_{1,x}(x),\ \phi_{2,x}(x),\ \phi_{3,x}(x), \cdots, \phi_{N,x}(x)], \quad (24)
\end{aligned}$$

and $\boldsymbol{U}(t_i)$ is the parameters when $t = i\Delta t$.

Substituting Eqs. (19)–(24) into the weak form (18) yields

$$\delta \boldsymbol{U}^{\mathrm{T}} \boldsymbol{C}[Aa(\boldsymbol{U}_{n+1} - \boldsymbol{U}_n) + \boldsymbol{L}U] + \delta \boldsymbol{U}^{\mathrm{T}} \boldsymbol{K}\boldsymbol{U} - \delta \boldsymbol{U}^{\mathrm{T}} \boldsymbol{F} + \delta \boldsymbol{U}^{\mathrm{T}} \boldsymbol{H}\boldsymbol{U} - \delta \boldsymbol{U}^{\mathrm{T}} \boldsymbol{Q} = \boldsymbol{0}, \quad (25)$$

where $\boldsymbol{C}$, $\boldsymbol{K}$ and $\boldsymbol{H}$ are $N \times N$ matrices, whose values of the ith row, jth column are

$$C_{ij} = \int \phi_i(x)\phi_j(x)\mathrm{d}\Gamma, \quad (26)$$

$$K_{ij} = \int \beta\phi_i(x)\phi_{j,x}(x)\mathrm{d}\Gamma - \int \gamma\phi_{i,x}\phi_{j,x}\mathrm{d}\Gamma, \quad (27)$$

$$H_{ij} = \varepsilon[\phi_i(x)\phi_j(x)|_{x=a} + \phi_i(x)\phi_j(x)|_{x=b}], \quad (28)$$

$\boldsymbol{Q}$ and $\boldsymbol{F}$ are $N \times 1$ matrices, whose values of the jth row are

$$Q_j = \varepsilon(\phi_j \overline{u}|_{x=a} + \phi_j \overline{u}|_{x=b}), \quad (29)$$

$$F_j = \int \phi_j(x) f(x,\ t)\mathrm{d}\Gamma, \quad (30)$$

and

$$\boldsymbol{L}\boldsymbol{U} = Aa \sum_{j=1}^{n} Ba(j)(\boldsymbol{U}_{n-j+1} - \boldsymbol{U}_{n-j}). \quad (31)$$

According to the arbitrariness of $\delta \boldsymbol{U}^{\mathrm{T}}(t)$, Eq. (25) becomes

$$\boldsymbol{C}[Aa(\boldsymbol{U}_{n+1} - \boldsymbol{U}_n) + \boldsymbol{L}U] + (\boldsymbol{K} + \boldsymbol{H})\boldsymbol{U} - \boldsymbol{F} - \boldsymbol{Q} = \boldsymbol{0}. \quad (32)$$

Making the time discretion of Eq. (32) by using the time central difference method, then we have

$$\boldsymbol{C}[Aa(\boldsymbol{U}_{n+1} - \boldsymbol{U}_n) + \boldsymbol{L}\boldsymbol{U}] + (\boldsymbol{K} + \boldsymbol{H})\frac{\boldsymbol{U}_{n+1} + \boldsymbol{U}_n}{2} - \frac{\boldsymbol{F}_{n+1} + \boldsymbol{F}_n}{2} - \frac{\boldsymbol{Q}_{n+1} + \boldsymbol{Q}_n}{2} = \boldsymbol{O}. \quad (33)$$

Now we can obtain the discrete equations about $\boldsymbol{U}_{n+1}$ as

$$\hbar \mathbf{U}_{n+1} = \ell \boldsymbol{U}_n + \boldsymbol{\Theta}, \quad (34)$$

where

$$\hbar = \boldsymbol{C}Aa + \frac{\boldsymbol{K} + \boldsymbol{H}}{2}, \quad (35)$$

$$\ell = \boldsymbol{C}Aa - \frac{\boldsymbol{K} + \boldsymbol{H}}{2}, \quad (36)$$

$$\boldsymbol{\Theta} = \frac{\boldsymbol{F}_{n+1} + \boldsymbol{F}_n + \boldsymbol{Q}_{n+1} + \boldsymbol{Q}_n}{2} - \boldsymbol{C}\boldsymbol{L}\boldsymbol{U}. \quad (37)$$

Equation (34) is the discrete expression of the time-fractional partial differential equations. Through Eq. (34), we can obtain any moment parameters $\boldsymbol{U}_{n+1}$. Then by substituting $\boldsymbol{U}_{n+1}$ into Eq. (11), the approximation of the exact solution of time-fractional partial differential equations is known.

4. Numerical examples

In this section, the application of the EFG method to the time-fractional partial differential equation is demonstrated by some examples whose exact solutions are known. In our examples, a regular arrangement of nodes is used and a cell structure is to set the quadrature cells to be equal to the intervals between the nodes. In each integration cell, a 4-Gauss quadrature scheme is used to evaluate the stiffness matrix.

To illustrate the accuracy of the method and compare our results with the exact solutions, we compute the error norms L_2 and L_∞ as

$$L_2 = \|u - u^h\|_2 = \sqrt{\Delta x \sum_{j=1}^{N} (u - u^h)^2}, \quad (38)$$

$$L_\infty = \|u - u^h\|_\infty = \max|u - u^h|, \quad (39)$$

where Δx is the spatial step size.

Example 1 We consider Eq. (1) with the following conditions: $\beta = 1$, $\gamma = 0$, and $f(x,t) = t^{1-\alpha}/[\Gamma(2-\alpha)\sin x + t\cos x]$. The boundary conditions are $u(-3,t) = t\sin(-3), u(3,t) = t\sin(3)$, the initial condition is $u(x,0) = 0$ and the exact solution is[7] $u(x,t) = t\sin x$.

We solve the problem by using the present method for the penalty factor $\varepsilon = 1 \times 10^7$, scaling parameter $d_{\mathrm{scale}} = 3$, time step length $\Delta t = 0.01$, spatial interval $[-3, 3]$ and number of nodes $N = 121$. The weight function is chosen to be a cubic spline function and the bases are chosen to be linear. The results are displayed in Table 1 and Figs. 1 and 2.

Table 1. Error norms of Example 1, when $\alpha = 0.6$, $\Delta t = 0.01$, $N = 121$.

Time	0.1	0.5	1.0	1.5	2.0
$L_\infty/10^{-3}$	5.3	5.2	4.6	3.8	3.2
$L_2/10^{-5}$	3.8229	3.1131	2.2555	1.5727	1.0726

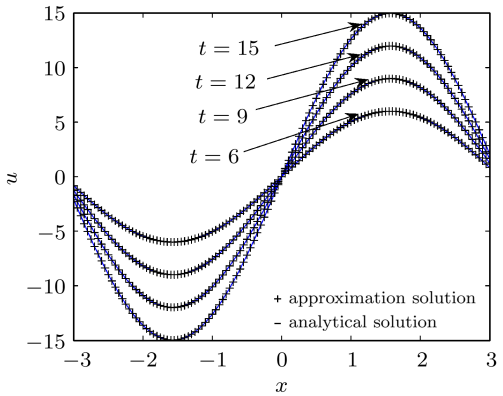


Fig. 1. Numerical simulations of Example 1, when $\alpha = 0.6$, $\Delta t = 0.01$, $N = 121$.

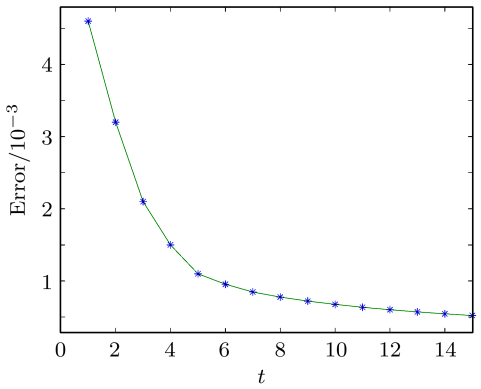


Fig. 2. The maximum errors of Example 1 at $t = 1$–15, when $\alpha = 0.6$, $\Delta t = 0.01$, $N = 121$.

Example 2 For the choice of parameters $\beta = 0, \gamma = -1$, and $f(x,t) = 2t^{2-\alpha}/[\Gamma(3-\alpha)\sin 2\pi x + 4\pi^2 t^2 \sin 2\pi x]$, the initial condition and the boundary conditions respectively are $u(x,0) = 0$, $u(0,t) = u(3,t) = 0$. The exact solution of the above problem is given by $u(x,t) = t^2 \sin 2\pi x$. Computation was carried out with parameters $\alpha = 0.25$, scaling parameter $d_{\text{scale}} = 3$, time step length $\Delta t = 0.01$, spatial interval $[0,3]$ and number of nodes $N = 121$. In Figs. 3 and 4, the exact solutions of these problems are plotted in comparison with the numerical results when $t = 0.5$, 1, 1.5. Some results are also given in Table 2.

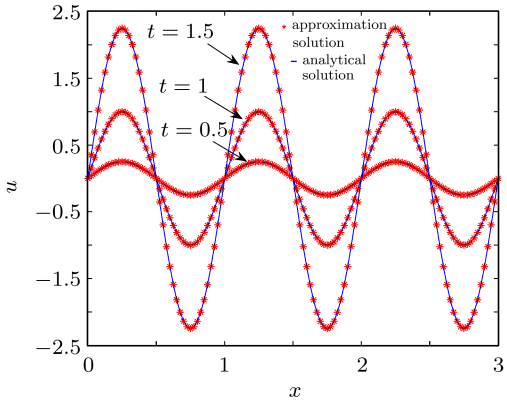


Fig. 3. (colour online) Numerical simulations of Example 2, when $\alpha = 0.25$, $\Delta t = 0.01$, $N = 121$.

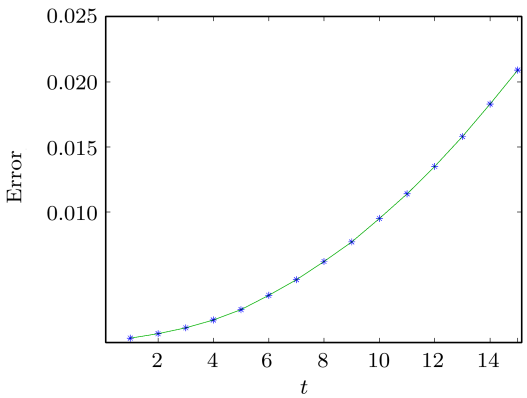


Fig. 4. The maximum errors of Example 2 at $t = 1 - 15$, when $\alpha = 0.25$, $\Delta t = 0.01$, $N = 121$.

Table 2. Error norms of Example 2, when $\alpha = 0.25$, $\Delta t = 0.01$, $N = 121$.

Time	0.1	0.5	1.0	1.5	2.0
$L_\infty/10^{-4}$	0.4545	1.6612	3.1045	4.6840	6.4619
$L_2/10^{-7}$	0.0153	0.1896	0.5745	1.1514	1.9817

Example 3 For the last problem, $\beta = 1, \gamma = -1$ and $f(x,t) = 2t^{2-\alpha}/[\Gamma(3-\alpha)+2x-2]$, which is a one-dimensional linear inhomogeneous fractional Burgers equation. The initial condition is $u(x,0) = x^2$ and the boundary conditions are $u(0,t) = t^2$, $u(1,t) = 1 + t^2$. The exact solutions[7] of this case is $u(x,t) = t^2 + x^2$. In this example, some parameters we used are the same as Example 1 except the number of nodes $N = 51$ and spatial interval $[0,1]$. The problem is solved by the proposed method at $t = 1$–15. The results are shown in Figs. 5 and 6 and Table 3, which show that the numerical solutions agree with the exact solution.

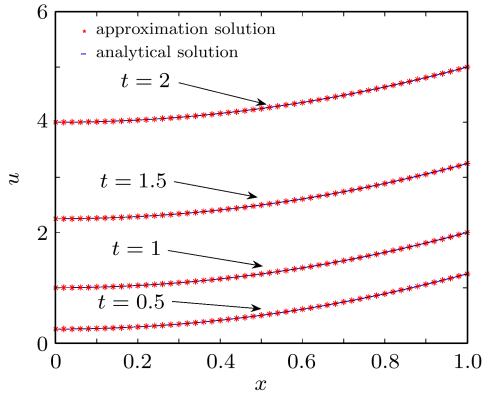


Fig. 5. (colour online) Numerical simulations of Example 3, when $\alpha = 0.5$, $\Delta t = 0.01$, $N = 51$.

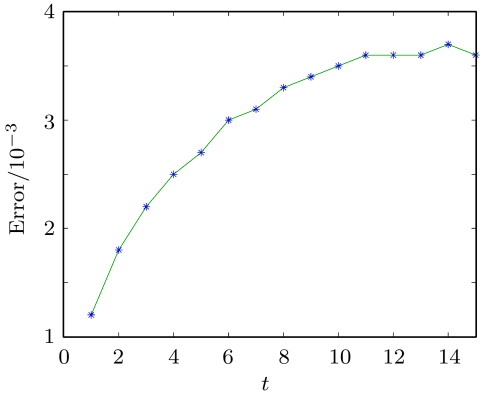


Fig. 6. The maximum errors of Example 3 at $t = 1$–15, when α=0.5, Δt=0.01, $N = 51$.

Table 3. Error norms of Example 3, when $\alpha = 0.5$, $\Delta t = 0.01$, $N = 51$.

Time	0.1	0.5	1.0	1.5	2.0
$L_\infty/10^{-3}$	0.2838	0.8075	1.2000	1.5000	1.8000
$L_2/10^{-7}$	0.2151	1.7359	3.8535	6.0045	8.1303

5. Conclusions

In this work, we have applied the element-free Galerkin (EFG) method for the solution of a class of time-fractional order partial differential equations. The meshless property of the EFG method is the most important advantage of this scheme over traditional mesh-dependent techniques such as the finite difference method. The numerical results, which are in good agreement with the analytical solutions, display that the technique is accurate and efficient. The problem presented in this paper suggests that the mess-free approximation method should be considered as one of the possible ways to solve a class of time-fractional order partial differential equations.

References

[1] Debnath L and Bhatta D 2004 *Frac.Calc.appl. Anal.* **7** 21

[2] Sweilam N H, Khader M M and Nagy A M 2011 *J. Comput. Appl. Math.* **235** 2832

[3] Zhang Y 2009 *Appl. Math. Comput.* **215** 524

[4] Chen C M, Liu F and Burrage K 2008 *Appl. Math. Comput.* **198** 754

[5] Liu F, Zhuang P, Anh V, Turner I and Burrage K 2007 *Appl. Math. Comput.* **191** 12

[6] Zheng Y Y, Li C P and Zhao Z G 2010 *Comput. Math. Appl.* **59** 1718

[7] Odibat Z and Momani S 2009 *Comput. Math. Appl.* **58** 2199

[8] Wang J F, Sun F X and Cheng R J 2010 *Chin. Phys.* B **19** 060201

[9] Cheng R J and Ge H X 2009 *Chin. Phys.* B **18** 4059

[10] Du C 2000 *Comput. Meth. Appl. Mesh. Engng.* **182** 89

[11] Kryl P and Belytschko T 1995 *Comput. Mech.* **17** 26

[12] Cheng Y M and Peng M J 2005 *Sci. China Ser. G* **48** 641

[13] Cheng Y M and Li J H 2005 *Acta Phys. Sin.* **54** 4463 (in Chinese)

[14] Qin Y X and Cheng Y M 2006 *Acta Phys. Sin.* **55** 3215 (in Chinese)

[15] Cheng R J and Cheng Y M 2007 *Acta Phys. Sin.* **56** 5569 (in Chinese)

[16] Dai B D and Cheng Y M 2007 *Acta Phys. Sin.* **56** 597 (in Chinese)

[17] Cheng R J and Cheng Y M 2008 *Acta Phys. Sin.* **57** 6037 (in Chinese)

[18] Cheng R J and Cheng Y M 2008 *Appl. Numer. Math.* **58** 884

[19] Chen L and Cheng Y M 2008 *Acta Phys. Sin.* **57** 1 (in Chinese)

[20] Chen L and Cheng Y M 2008 *Acta Phys. Sin.* **57** 5069 (in Chinese)

[21] Ren H P and Zhang W 2009 *Chin. Phys.* B **18** 4065

[22] Chen L and Cheng Y M 2010 *Chin. Phys.* B **19** 090202

[23] Cheng R J and Ge H X 2010 *Chin. Phys.* B **19** 090201

[24] Wang J F and Cheng Y M 2011 *Chin. Phys.* B **20** 030206

[25] Cheng R J and Cheng Y M 2011 *Chin. Phys.* B **20** 070206

[26] Cheng R J and Liew K M 2009 *Comput. Mech.* **45** 1

[27] Cheng R J and Cheng Y M 2011 *Acta Phys. Sin.* **60** 070206 (in Chinese)

[28] Belytschko T, Krongauz Y and Organ D 1994 *Comput. Meth. Appl. Mech. Engng.* **139** 3

[29] Belytschko T, Lu Y Y and Gu L 1994 *Int. J. Numer. Meth. Engng.* **37** 229

[30] Diego A M 2008 *Comput. Math. Appl.* **56** 1138