Advances in Continuous and Discrete Models

RESEARCH Open Access

A time-splitting local meshfree approach for time-fractional anisotropic diffusion equation: application in image denoising

Jalil Mazloum[1*] and Behrang Hadian Siahkal-Mahalle[1]

*Correspondence:
jalil.mazloum@ssau.ac.ir
[1]Department of Electrical Engineering, Shahid Sattari Aeronautical University of Science and Technology, Tehran, Iran

Abstract

Image denoising approaches based on partial differential modeling have attracted a lot of attention in image processing due to their high performance. The nonlinear anisotropic diffusion equations, specially Perona–Malik model, are powerful tools that improve the quality of the image by removing noise while preserving details and edges. In this paper, we propose a powerful and accurate local meshless algorithm to solve the time-fractional Perona–Malik model which has an adjustable fractional derivative making the control of the diffusion process more convenient than the classical one. In order to overcome the complexities of the problem, a suitable combination of the compactly supported radial basis function method and operator splitting technique is proposed to convert a complex time-fractional partial differential equation into sparse linear algebraic systems that standard solvers can solve. The numerical results of classical and fractional models are explored in different metrics to demonstrate the proposed scheme's effectiveness. The numerical experiments confirm that the method is suitable to denoise digital images and show that the fractional derivative increases the model's ability to remove noise in images.

MSC: 65M30; 65M06; 65M55

Keywords: Meshless methods; Fractional calculus; Image denoising; Operator splitting; Perona–Malik

1 Introduction

Nowadays, images have tremendous applications in scientific studies and should be of good quality in order to be useful [1, 2]. So, image denoising has an important role in image processing and computer vision to prepare images with better resolutions. Partial differential equations (PDEs) are widely used in different parts of image processing, such as filtering, restoration, segmentation, edge enhancement, detection [3], and especially denoising. There are some PDE models that are applied for image and signal denoising, such as diffusion equation, geometric curvature equation, and TV flow [4, 5]. In the models based on the diffusion equation, the PDE model uses a nonlinear anisotropic diffusion to enhance the quality of an image by removing noise while preserving details and edges [6]. In this paper, we focus on the time-fractional Perona–Malik model (FPMM) with the

Caputo fractional derivative for temporal derivative denoted by ${}_0^C\mathcal{D}_t^\alpha$. The FPMM model is defined as follows:

$$\begin{aligned}
&{}_0^C\mathcal{D}_t^\alpha u(x,y,t) = \operatorname{div}\big(g\big(\big\|\nabla G_\sigma * u(x,y,t)\big\|\big)\nabla u(x,y,t)\big),\\
&\quad (x,y,t)\in\Omega\times I, 0<\alpha<1,\\
&\frac{\partial u}{\partial \mathbf{n}} = 0, \quad (x,y,t)\in\partial\Omega\times I,\\
&u(x,y,0) = u_0, \quad (x,y)\in\Omega,
\end{aligned} \tag{1.1}$$

where $\Omega\subseteq\mathbb{R}^2$, $I=(0,T)$ is the scaling time interval, α is the order of time fractional derivative, and $\mathbf{n}$ is the unit outward normal to the boundary of Ω. The Caputo fractional derivative of order α is defined as

$${}_0^C\mathcal{D}_t^\alpha u(x,y,t) = \frac{1}{\Gamma(1-\alpha)}\int_0^t \frac{1}{(t-\tau)^\alpha}\frac{\partial u(x,y,\tau)}{\partial\tau}\,d\tau, \tag{1.2}$$

in which $\Gamma(z)=\int_0^\infty e^{-s}s^{z-1}\,ds$ has the following property:

$$\Gamma(z+1) = z\Gamma(z-1).$$

The nonlinear anisotropic diffusion filtering has been widely used in image processing and performed significantly [7–9]. Koenderink found a relationship between the solution of a heat equation with an initial noisy image and its convolution with Gaussian function at each scale [10]. In fact, the denoising process can be described as the solution of the following linear diffusion equation:

$$\frac{\partial u(x,y,t)}{\partial t} = \Delta u(x,y,t), \qquad u(x,y,0) = u_0. \tag{1.3}$$

The solution of this equation can be obtained by the following integral convolution:

$$u(x,y,t) = G_t * f, \tag{1.4}$$

where G_t is the Gaussian function with standard deviation σ. In this model, $u(x,y,0)$ is the initial noisy image whose noise reduces over time. However, this model also fades the image on the edges which leads to losing some essential data of the image. So, Perona and Malik, in 1990, proposed a nonlinear diffusion model, called Perona–Malik model (PMM), to overcome the problems of previous models [11]. Actually, they used the gradient of the actual image $u(x,y,t)$ as feedback into a diffusion process and introduced the following anisotropic diffusion equation with a zero Neumann boundary condition:

$$\frac{\partial u(x,y,t)}{\partial t} = \operatorname{div}\big(g\big(\big\|\nabla u(x,y,t)\big\|\big)\nabla u(x,y,t)\big), \tag{1.5}$$

where $g(\cdot)$ is a smooth decreasing function in terms of $\|\nabla u(x,y,t)\|$, which controls the diffusion strength and is called the diffusion coefficient. This function is considered such

that becomes equal to 1 inside the region and tends to zero near the edges. A typically used diffusion coefficient is [12]

$$g\big(\|\nabla u(x,y,t)\|\big) = \frac{1}{1+(\frac{\|\nabla u(x,y,t)\|}{K})^2}, \tag{1.6}$$

where K is an edge detector threshold parameter that decides the amount of diffusion to take place.

However, PMM is an ill-posed problem as it is a degenerate and backward diffusion equation. In other words, if the signal-to-noise ratio (SNR) is very low, most of noises remain on the edges. Therefore, Catté et al. proposed the regularized PMM to triumph over this disadvantage [13]. This model is defined as follows:

$$\frac{\partial u(x,y,t)}{\partial t} = \operatorname{div}\big(g\big(\|\nabla G_\sigma * u(x,y,t)\|\big)\nabla u(x,y,t)\big), \tag{1.7}$$

where $G_\sigma \in C^\infty(\mathbb{R}^2)$ is Gauss kernel with variance σ, and we have

$$\|\nabla G_\sigma * u\| = \left(\left(\frac{\partial G_\sigma}{\partial x} * \tilde{u}\right)^2 + \left(\frac{\partial G_\sigma}{\partial y} * \tilde{u}\right)^2\right)^{\frac{1}{2}}, \tag{1.8}$$

in which $*$ represents the standard convolution and $\tilde{u}$ is an extension of u to $\mathbb{R}^2$ specified by a periodic reflection through the boundary of the problem.

PMM is a powerful model which has been utilized in nonlinear data filtration, image restoration, noise removal, and edge detection [11, 13, 14]. For instance, scientists used PMM in optical coherence tomography [15], radiography image processing [16], etc. Moreover, solving the PMM model with numerical algorithms is a challenging problem and some researchers have been trying to improve suitable approaches to simulate its solution. In [17], the domain decomposition approach combined with the finite difference method was proposed to solve nonlinear problems in image denoising. Kamranian and Dehghan developed a meshfree finite point method to solve PMM [12]. Gu, in 2020, proposed a finite element approach for two Peron–Malik and Yu–Kaveh models and compared the obtained results with the finite difference method [4]. Moreover, Hjouji et al. proposed a mixed finite element method for bivariate PMM [18]. Also, Sidi Ammi and Jamiai applied finite difference and Legendre spectral methods to a time-fractional diffusion–convection equation and its application for image processing [3].

Fractional integration and fractional differentiation are generalizations of integer–order ones which have various applications in modeling the phenomena in different fields of sciences [3, 19–22], and scientists have been trying to develop various proper numerical approaches for solving fractional differential equations [23–28]. Fractional calculus has a long history in mathematics [29, 30], and various definitions and operators are provided to express the fractional derivative, including Grunwald–Letnikow [31], Marchaud [31], Riemann–Liouville [31, 32], Caputo [33, 34], Riesz [35, 36], etc. [37–39]. Recently, fractional calculus has also been used for many image processing applications such as image restoration, segmentation, texture enhancement, edge detection, image encryption, and image denoising to improve their performances [19, 40–42].

FPMM is an extension of the classical one whose temporal derivatives are taken in the Caputo sense, which enhances model's applicability. It is worth mentioning that if $\alpha = 1$,

the classical model is obtained. The fractional model, in comparison with the classical one, has an extra parameter α which can enhance the model's flexibility and improve its performance.

In recent decades, meshless algorithms have been among the most powerful numerical techniques utilized to approximate solutions of complex differential equations. Unlike the other methods which need mesh generation, meshless methods only work with a scattered set of collocation points without any prior information on the domain of the problem. This property drastically reduces the computational cost compared to the other numerical methods which need mesh generation [43]. In addition, using radial basis functions (RBFs) in meshless methods has some advantages like spectral convergence for smooth functions in even complex geometries, ease of implementation, and is appropriate for high-dimension problems [44]. RBFs can be divided into two categories, namely global and local. Global RBFs (e.g., Gaussian, Multiquadric, etc.) have a severe drawback, in particular they have a dense and ill-conditioned coefficient matrix of the obtained linear system [45–47]. On the other hand, compactly-supported radial basis functions (CS–RBFs) such as Wendland's functions are defined on the local arbitrary subdomain [48–51] which makes them more stable. The CS–RBF method is more stable due to the sparsity of the obtained coefficient matrix, but it is not as accurate as the global RBFs. CS–RBFs have been introduced by Wu and Wendland for the first time for scattered data interpolation and have been used broadly in numerical simulations [52].

Since FPMM is applied to images, we usually face a large domain, making the numerical simulation difficult. In the current paper, with the aid of the operator splitting (OS) method, PMM is divided into two partial differential equations. Then, the CS–RBF approach is applied to obtained PDEs. Finally, the subproblems reduce to linear algebraic equations of small dimension that can be efficiently solved by a proper algorithm such as the LU factorization technique. Finally, we evaluate the proposed method by applying it to some examples for image denoising. To summarize, we propose an advanced and proper numerical method based on OS and CS–RBF schemes to remove noise by FPMM. This approach has some significant advantages. First, the model considered in this paper is FPMM which has a fractional order derivative that improves model's ability in noise removal compared to the classical PMM. Second, the proposed approach converts a complex nonlinear system to some simple linear ones and dramatically decreases computational costs. Finally, as we use a local RBF method, the obtained linear systems are sparse and well-posed in comparison with global approaches that can be solved by standard solvers. Moreover, we report the obtained numerical results with a proper value of α with different test cases and compare them with PMM in different metrics, which confirms the high accuracy and effectiveness of the proposed algorithm.

The organization of this paper is as follows: In Sect. 2, the basic properties of the algorithm presented in this work, such as the details about fractional derivative, OS, and CS–RBF collocation approaches, are briefly described. The numerical results obtained by the proposed method are presented and discussed in Sect. 3, and some concluding remarks are given in Sect. 4.

2 Proposed approach

In this section, the Trotter OS scheme [53] is applied to Eq. (1.1) and converts it to some simpler PDEs. Then, the time dimension is discretized by a backward Euler approach. Finally, the obtained equations are solved by the CS–RBF method.

2.1 Temporal discretization

In this part, we apply a divide-and-conquer approach which is known as the OS method, to divide Eq. (1.1) into two subproblems in separate directions. Then, the Caputo time derivative is discretized in each subproblem.

2.1.1 Trotter splitting scheme

First, in order to reduce the complexity of problem (1.1), a first-order OS method called Trotter splitting approach is employed on Eq. (1.1) to solve FPMM along separate directions. Using this scheme reduces the size of obtained matrices and thus substantially reduces the computational complexity. Consider the time interval I which is divided into $M+1$ equally spaced steps $t_0, t_2, \ldots, t_M$, $t_k = k\Delta t$, $k = 0, 1, \ldots, M$, and $\Delta t = \frac{T}{M}$. Equation (1.1) can be written as follows:

$$ {}_0^C\mathcal{D}_t^\alpha u = \mathcal{L}_x(u) + \mathcal{L}_y(u), \tag{2.1} $$

where $\mathcal{L}_x$ and $\mathcal{L}_y$ are suboperators on function u as follows:

$$ \mathcal{L}_x(u) = \frac{\partial}{\partial x} g\big(\|\nabla G_\sigma * u\|\big) \frac{\partial}{\partial x} u + g\big(\|\nabla G_\sigma * u\|\big) \frac{\partial^2}{\partial x^2} u, \tag{2.2} $$

$$ \mathcal{L}_y(u) = \frac{\partial}{\partial y} g\big(\|\nabla G_\sigma * u\|\big) \frac{\partial}{\partial y} u + g\big(\|\nabla G_\sigma * u\|\big) \frac{\partial^2}{\partial y^2} u. \tag{2.3} $$

By employing the Trotter splitting method, the approximate solution of $u(x, y, t_{k+1})$, which can simply be written as u^{k+1}, is obtained as $u^{k+1} = [\mathcal{S}_x^{\Delta t} \mathcal{S}_y^{\Delta t}] u^k$, in which $\mathcal{S}_x^{\Delta t}$ and $\mathcal{S}_y^{\Delta t}$ are ${}_0^C\mathcal{D}_t^\alpha u^* = \mathcal{L}_x(u^*)$ and ${}_0^C\mathcal{D}_t^\alpha u^{**} = \mathcal{L}_y(u^{**})$, respectively. Finally, we have $u^{k+1} = u^{**^{k+1}}$. The interested readers can see [43, 54–56] for more details about the OS methods.

Now, a proper scheme should be proposed to discretize the Caputo time derivative operator ${}_0^C\mathcal{D}_t^\alpha$ in the subproblems.

2.1.2 Discretization of Caputo derivative

At each time step t_k, $k = 0, 2, \ldots, M$, using the backward Euler approach, the time fractional derivative is approximated as follows:

$$ {}_0^C\mathcal{D}_t^\alpha u(x, y, t_{k+1}) = \frac{1}{\Gamma(1-\alpha)} \sum_{j=0}^{k} \int_{t_j}^{t_{j+1}} \frac{1}{(t_{k+1} - \tau)^\alpha} \frac{\partial}{\partial \tau} u(x, y, \tau)\, d\tau. \tag{2.4} $$

So, we have

$$ \begin{aligned} &{}_0^C\mathcal{D}_t^\alpha u(x, y, t_{k+1}) \\ &\quad = \frac{1}{\Gamma(1-\alpha)} \sum_{j=0}^{k} \frac{u(x, y, t_{j+1}) - u(x, y, t_j)}{\Delta t} \int_{t_j}^{t_{j+1}} \frac{d\tau}{(t_{k+1} - \tau)^\alpha} + R_{k+1}, \end{aligned} \tag{2.5} $$

in which R_{k+1} is the truncation error and we have

$$ R_{k+1} \leq \mathcal{C} \left[\frac{1}{\Gamma(1-\alpha)} \sum_{j=0}^{k} \int_{t_j}^{t_{j+1}} \frac{t_{j+1} + t_j - 2\tau}{(t_{k+1} - \tau)^\alpha} d\tau + \mathcal{O}\big(\Delta t^2\big) \right], \tag{2.6} $$

where $\mathcal{C}$ is a constant that depends only on u. Based on Lemma 3.1 in [57] and the fact that $\Gamma(2-\alpha) \geq 2$ for all $\alpha \in [0,1]$, it can be concluded that

$$\left| \frac{1}{\Gamma(1-\alpha)} \sum_{j=0}^{k} \int_{t_j}^{t_{j+1}} \frac{t_{j+1} + t_j - 2\tau}{(t_{k+1} - \tau)^{\alpha}} \, d\tau \right| \leq 2\Delta t^{2-\alpha}.$$

So

$$|R_{k+1}| \leq \mathcal{C}\Delta t^{2-\alpha}. \tag{2.7}$$

Moreover, from Eq. (2.4), the following equations can be computed:

$$\begin{aligned} &\frac{1}{\Gamma(1-\alpha)} \sum_{j=0}^{k} \int_{t_j}^{t_{j+1}} \frac{1}{(t_{k+1} - \tau)^{\alpha}} \frac{\partial}{\partial \tau} u(x,y,\tau) \, d\tau \\ &\quad = \frac{1}{\Gamma(1-\alpha)} \sum_{j=0}^{k} \frac{u(x,y,t_{j+1}) - u(x,y,t_j)}{\Delta t} \int_{t_{k-j}}^{t_{k+1-j}} \frac{dt}{t^{\alpha}} \\ &\quad = \frac{1}{\Gamma(1-\alpha)} \sum_{j=0}^{k} \frac{u(x,y,t_{k+1-j}) - u(x,y,t_{k-j})}{\Delta t} \int_{t_j}^{t_{j+1}} \frac{dt}{t^{\alpha}} \\ &\quad = \frac{1}{\Gamma(2-\alpha)} \sum_{j=0}^{k} \frac{u(x,y,t_{k+1-j}) - u(x,y,t_{k-j})}{\Delta t^{\alpha}} \big((j+1)^{1-\alpha} - j^{1-\alpha} \big). \end{aligned} \tag{2.8}$$

The discrete fractional differential operator D_t^{α} is as follows:

$$D_t^{\alpha} = \frac{1}{\Gamma(2-\alpha)} \sum_{j=0}^{k} \frac{u(x,y,t_{k+1-j}) - u(x,y,t_{k-j})}{\Delta t^{\alpha}} \big((j+1)^{1-\alpha} - j^{1-\alpha} \big). \tag{2.9}$$

Thus, we can write

$${}_0^C\mathcal{D}_t^{\alpha} u(x,y,t_{k+1}) = D_t^{\alpha} + R_{k+1}. \tag{2.10}$$

So D_t^{α} is an approximation to ${}_0^C\mathcal{D}_t$ and we have

$$\begin{aligned} &{}_0^C\mathcal{D}_t^{\alpha} u(x,y,t_{k+1}) \\ &\quad \approx \frac{1}{\Gamma(2-\alpha)(\Delta t)^{\alpha}} \sum_{j=0}^{k} \big((j+1)^{1-\alpha} - j^{1-\alpha} \big) \big(u(x,y,t_{k-j+1}) - u(x,y,t_{k-j}) \big). \end{aligned} \tag{2.11}$$

For the sake of simplicity, we define $\mathcal{C}_{\Delta t}^{\alpha} = \frac{1}{\Gamma(2-\alpha)(\Delta t)^{\alpha}}$, $\mathcal{A}_j^{\alpha} = (j+1)^{1-\alpha} - j^{1-\alpha}$, and $u(x,y,t_k) = u^k$.

Remark 2.1 From Theorems 3.1 and 3.2 in [57], it can be deduced that the proposed temporal discretization scheme is unconditionally stable for $\Delta t > 0$ and has globally $(2-\alpha)$-order accuracy for $0 < \alpha < 1$.

Now, by using Eq. (2.11), the subproblems are discretized along the temporal dimension as follows:

$$\begin{aligned} \mathcal{C}_{\Delta t}^{\alpha}\left[u^{*^{k+1}}-u^{k}+\sum_{j=1}^{k}\left(u^{k+1-j}-u^{k-j}\right)\mathcal{A}_{j}^{\alpha}\right] &= \frac{\partial}{\partial x}g\left(\left\|\nabla G_{\sigma}*u^{k}\right\|\right)\frac{\partial}{\partial x}u^{*^{k+1}} \\ &\quad + g\left(\left\|\nabla G_{\sigma}*u^{k}\right\|\right)\frac{\partial^{2}}{\partial x^{2}}u^{*k+1}, \end{aligned} \tag{2.12}$$

$$\begin{aligned} \mathcal{C}_{\Delta t}^{\alpha}\left[u^{**^{k+1}}-u^{*^{k}}+\sum_{j=1}^{k}\left(u^{*^{k+1-j}}-u^{*^{k-j}}\right)\mathcal{A}_{j}^{\alpha}\right] &= \frac{\partial}{\partial y}g\left(\left\|\nabla G_{\sigma}*u^{*^{k}}\right\|\right)\frac{\partial}{\partial y}u^{**^{k+1}} \\ &\quad + g\left(\left\|\nabla G_{\sigma}*u^{*^{k}}\right\|\right)\frac{\partial^{2}}{\partial y^{2}}u^{**k+1}. \end{aligned} \tag{2.13}$$

2.2 Spatial discretization

Before going to the space discretization of problems (2.12) and (2.13), we focus on the convolution term $G_{\sigma}*u^{k}(G_{\sigma}*u^{*^{k}})$. Since the kernel G_{σ} is smooth, the term $G_{\sigma}*u^{k}(G_{\sigma}*u^{*^{k}})$ can be replaced by a heat equation with initial condition $u^{k}(u^{*^{k}})$. These heat equations are solved by the OS method in conjunction with the Crank–Nicolson and CS–RBF approaches. The solutions of heat equations replaced in Eqs. (2.12) and (2.13) are represented by u^{c} and $u^{*^{c}}$, respectively.

By substituting of the solution of the heat equations, Eqs. (2.12) and (2.13) can be considered in the following forms:

$$\begin{aligned} &\frac{\partial}{\partial x}g\left(\left\|\nabla u^{c}\right\|\right)\frac{\partial}{\partial x}u^{*^{k+1}}+g\left(\left\|\nabla u^{c}\right\|\right)\frac{\partial^{2}}{\partial x^{2}}u^{*k+1}-\mathcal{C}_{\Delta t}^{\alpha}u^{*^{k+1}} \\ &\quad=\mathcal{C}_{\Delta t}^{\alpha}\sum_{j=1}^{k}\left(u^{k+1-j}-u^{k-j}\right)\mathcal{A}_{j}^{\alpha}-\mathcal{C}_{\Delta t}^{\alpha}u^{k}, \end{aligned} \tag{2.14}$$

$$\begin{aligned} &\frac{\partial}{\partial y}g\left(\left\|\nabla u^{*^{c}}\right\|\right)\frac{\partial}{\partial y}u^{**^{k+1}}+g\left(\left\|\nabla u^{*^{c}}\right\|\right)\frac{\partial^{2}}{\partial y^{2}}u^{**k+1}-\mathcal{C}_{\Delta t}^{\alpha}u^{**^{k+1}} \\ &\quad=\mathcal{C}_{\Delta t}^{\alpha}\sum_{j=1}^{k}\left(u^{*^{k+1-j}}-u^{*^{k-j}}\right)\mathcal{A}_{j}^{\alpha}-\mathcal{C}_{\Delta t}^{\alpha}u^{*^{k}}. \end{aligned} \tag{2.15}$$

By obtaining u^{c} and $u^{*^{c}}$, functions $g(\|\nabla u^{c}\|)$ and $g(\|\nabla u^{*^{c}}\|)$ are computed as follows:

$$g\left(\left\|\nabla u^{c}\right\|\right)=\frac{1}{1+\frac{(\frac{\partial u^{c}}{\partial x})^{2}+(\frac{\partial u^{c}}{\partial y})^{2}}{K^{2}}},\qquad g\left(\left\|\nabla u^{*^{c}}\right\|\right)=\frac{1}{1+\frac{(\frac{\partial u^{*^{c}}}{\partial x})^{2}+(\frac{\partial u^{*^{c}}}{\partial y})^{2}}{K^{2}}}.$$

Moreover, $\frac{\partial g(\|\nabla u^{c}\|)}{\partial x}$ and $\frac{\partial g(\|\nabla u^{*^{c}}\|)}{\partial y}$ can be obtained by the chain rule.

Now, CS–RBF method is applied on Eqs. (2.14) and (2.15) which has two main benefits. First, as there is no need for any mesh generation for constructing the shape functions, the CS–RBFs scheme is considered truly meshless, and the coefficient matrix of this approach is well-conditioned and can be solved by common methods easily. Moreover, the local RBFs like Wendland CS–RBFs do not have a shape parameter and are more stable than global ones.

At the first step of the space discretization, the nodal points over the domain Ω should be selected at each time step t_k. We consider pixels of the images, i.e., $X \times Y$ where $X = \{x_1, \ldots, x_{N_x}\}$ and $Y = \{y_1, \ldots, y_{N_y}\}$, as the mesh nodes on the boundary and inside of the domain. In order to approximate u^k for all $k \geq 0$, u^0 is computed by the interpolation, then u^k for $k > 0$ is obtained by solving Eq. (2.12) using the CS–RBFs scheme. In addition, CS–RBFs are symmetric with respect to their center points by the following definition:

Definition 2.2 ([45]) A function $\phi : \mathbb{R}^s \to \mathbb{R}$ is called radial provided there exists a univariate function $\varphi : [0, \infty) \to \mathbb{R}$ such that

$$\phi(\mathbf{x}) = \varphi(r), \quad \text{where } r = \|\mathbf{x}\| \tag{2.16}$$

and $\|\cdot\|$ is some norm on $\mathbb{R}^s$, usually the Euclidean norm.

Among different choices of existing radial basis functions, in this paper, Wendland's compactly supported radial basis functions (WCS–RBFs) with C^2 smoothness are selected because they do not include any shape parameters that make these functions easier to use. These functions are defined as follows:

$$\phi_i(x) = (1 - \delta r_i)_+^4 (1 + 4\delta r_i), \tag{2.17}$$

in which $r_i = \|x - x_i\|$ is the distance from node x_i to x, and δ is the size of support for the radial function $\phi_i(x)$. Moreover, function $(1 - \delta r_i)_+^4$ is defined such that for $0 \leq \delta r_i < 1$ it is equal to $(1 - \delta r_i)^4$, otherwise is zero.

To approximate functions u^k, $u^{*^{k+1}}$, and $u^{**^{k+1}}$ in Ω over center points (x_i, y_j), $i = 1, 2, \ldots, N_x$, $j = 1, 2, \ldots, N_y$, the CS–RBF interpolation approximations $\hat{u}^k$, $\hat{u}^{*^{k+1}}$, and $\hat{u}^{**^{k+1}}$ are defined as

$$\hat{u}^k(x, y_j) = \sum_{i=0}^{N_x} \theta_{ij}^k \phi_i(x), \quad j = 1, 2, \ldots, N_y, \tag{2.18}$$

$$\hat{u}^{*^{k+1}}(x, y_j) = \sum_{i=1}^{N_x} \lambda_{ij}^{k+1} \phi_i(x), \quad j = 1, 2, \ldots, N_y, \tag{2.19}$$

$$\hat{u}^{**^{k+1}}(x_i, y) = \sum_{j=1}^{N_y} \gamma_{ij}^{k+1} \phi_j(y), \quad i = 1, 2, \ldots, N_x, \tag{2.20}$$

in which ϕ_i and ϕ_j are RBFs based on centers $\{x_i\}_{i=0}^{N_x}$ and $\{y_j\}_{j=0}^{N_y}$. Additionally, λ_{ij} and γ_{ij} are unknown coefficients which should be specified.

Finally, substituting the approximation function of Eq. (2.19) into Eq. (2.14) associated to the x-direction, for all nodes in Ω, the matrix forms of their discrete equations are obtained as follows:

$$\mathbf{A}_j \Lambda_j^{k+1} = \mathbf{B}_j, \quad j = 1, 2, \ldots, N_y, \tag{2.21}$$

where $\Lambda_j^{k+1} = [\lambda_{1j}^{k+1}, \lambda_{2j}^{k+1}, \ldots, \lambda_{N_x j}^{k+1}]^T$, $\mathbf{A}_j$ and $\mathbf{B}_j$ are $N_x \times N_x$ matrix and $N_x \times 1$ vector, respectively, defined as

$$\mathbf{A}_j = \mathbf{D}\big(\mathbf{G}_x^j \Upsilon_x + \mathbf{G}^j \Psi_x - \mathcal{C}_{\Delta t}^{\alpha} \Phi_x\big) + \mathbf{H}_x, \tag{2.22}$$

$$\mathbf{B}_j = \mathbf{D}\left(\mathcal{C}_{\Delta t}^{\alpha} \sum_{p=1}^{k} \mathcal{A}_k^{\alpha}\big(\Phi_x \Theta_j^{p+1-k} - \Phi_x \Theta_j^{p-k}\big) - \mathcal{C}_{\Delta t}^{\alpha} \Phi_x \Theta_j^{k}\right), \tag{2.23}$$

where

$$\Phi_x = \begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & \ldots & \phi_{N_x-1}(x_1) & \phi_{N_x}(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \ldots & \phi_{N_x-1}(x_2) & \phi_{N_x}(x_2) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \phi_1(x_{N_x-1}) & \phi_2(x_{N_x-1}) & \ldots & \phi_{N_x-1}(x_{N_x-1}) & \phi_{N_x}(x_{N_x-1}) \\ \phi_1(x_{N_x}) & \phi_2(x_{N_x}) & \ldots & \phi_n(x_{N_x}) & \phi_{N_x}(x_{N_x}) \end{bmatrix},$$

$$\Upsilon_x = \begin{bmatrix} \phi_1'(x_1) & \phi_2'(x_1) & \ldots & \phi_{N_x-1}'(x_1) & \phi_{N_x}'(x_1) \\ \phi_1'(x_2) & \phi_2'(x_2) & \ldots & \phi_{N_x-1}'(x_2) & \phi_{N_x}'(x_2) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \phi_1'(x_{N_x-1}) & \phi_2'(x_{N_x-1}) & \ldots & \phi_{N_x-1}'(x_{N_x-1}) & \phi_{N_x}'(x_{N_x-1}) \\ \phi_1'(x_{N_x}) & \phi_2'(x_{N_x}) & \ldots & \phi_n'(x_{N_x}) & \phi_{N_x}'(x_{N_x}) \end{bmatrix},$$

$$\Psi_x = \begin{bmatrix} \phi_1''(x_1) & \phi_2''(x_1) & \ldots & \phi_{N_x-1}''(x_1) & \phi_{N_x}''(x_1) \\ \phi_1''(x_2) & \phi_2''(x_2) & \ldots & \phi_{N_x-1}''(x_2) & \phi_{N_x}''(x_2) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \phi_1''(x_{N_x-1}) & \phi_2''(x_{N_x-1}) & \ldots & \phi_{N_x-1}''(x_{N_x-1}) & \phi_{N_x}''(x_{N_x-1}) \\ \phi_1''(x_{N_x}) & \phi_2''(x_{N_x}) & \ldots & \phi_n''(x_{N_x}) & \phi_{N_x}''(x_{N_x}) \end{bmatrix},$$

$$\mathbf{D} = \operatorname{diag}(0, 1, 1, 1, \ldots, 1, 0),$$

$$\begin{aligned} \mathbf{G}^j = \operatorname{diag}\big(& g\big(\|\nabla u^c(x_1, y_j)\|\big), g\big(\|\nabla u^c(x_2, y_j)\|\big), \\ & \ldots, g\big(\|\nabla u^c(x_{N_x-1}, y_j)\|\big), g\big(\|\nabla u^c(x_{N_x}, y_j)\|\big)\big), \end{aligned}$$

$$\begin{aligned} \mathbf{G}_x^j = \operatorname{diag}\Big(& \frac{\partial}{\partial x} g\big(\|\nabla u^c(x_1, y_j)\|\big), \frac{\partial}{\partial x} g\big(\|\nabla u^c(x_2, y_j)\|\big), \\ & \ldots, \frac{\partial}{\partial x} g\big(\|\nabla u^c(x_{N_x-1}, y_j)\|\big), \frac{\partial}{\partial x} g\big(\|\nabla u^c(x_{N_x}, y_j)\|\big)\Big), \end{aligned}$$

$\mathbf{H}_x$ is an $N_x \times N_x$ matrix whose first and last rows are first and last rows of Υ_x and the other rows contain only zero elements. Moreover,

$$\Theta_j^m = \big(\theta_{1j}^m, \theta_{2j}^m, \ldots, \theta_{N_x j}^m\big)^T,$$

that are known coefficients computed in previous iterations. Indeed, there are N_y linear systems of N_x algebraic equations which are well-posed and can be solved by a standard approach such as LU factorization. By solving these systems, the coefficient vectors Λ_j^{k+1} are obtained and the approximation of $\hat{u}^{*^{k+1}}$ is computed as follows:

$$\mathbf{U}^{*^{k+1}} = \Phi\big[\Lambda_1^{k+1}, \Lambda_2^{k+1}, \ldots, \Lambda_{N_y}^{k+1}\big], \tag{2.24}$$

in which

$$\mathbf{U}^{*^{k+1}} = \begin{bmatrix} u^{*^{k+1}}(x_1,y_1) & u^{*^{k+1}}(x_1,y_2) & \dots & u^{*^{k+1}}(x_1,y_{N_y-1}) & u^{*^{k+1}}(x_1,y_{N_y}) \\ u^{*^{k+1}}(x_2,y_1) & u^{*^{k+1}}(x_2,y_2) & \dots & u^{*^{k+1}}(x_2,y_{N_y-1}) & u^{*^{k+1}}(x_2,y_{N_y}) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ u^{*^{k+1}}(x_{N_x-1},y_1) & u^{*^{k+1}}(x_{N_x-1},y_2) & \dots & u^{*^{k+1}}(x_{N_x-1},y_{N_y-1}) & u^{*^{k+1}}(x_{N_x-1},y_{N_y}) \\ u^{*^{k+1}}(x_{N_x},y_1) & u^{*^{k+1}}(x_{N_x},y_2) & \dots & u^{*^{k+1}}(x_{N_x},y_{N_y-1}) & u^{*^{k+1}}(x_{N_x},y_{N_y}) \end{bmatrix},$$

The same procedure is applied on Eq. (2.15); as a result, the following system is obtained:

$$\mathbf{A}_i\Gamma_i^{k+1} = \mathbf{B}_i, \quad i = 1,2,\dots,N_x, \tag{2.25}$$

where $\Gamma_i^{k+1} = [\gamma_{i1}^{k+1}, \gamma_{i2}^{k+1}, \dots, \gamma_{iN_y}^{k+1}]^T$, $\mathbf{A}_i$ and $\mathbf{B}_i$ are $N_y \times N_y$ matrix and $N_y \times 1$ vector, respectively, defined as follows:

$$\mathbf{A}_i = \mathbf{D}\big(\tilde{\mathbf{G}}_y^i\Upsilon_y + \tilde{\mathbf{G}}^i\Psi_y - \mathcal{C}_{\Delta t}^{\alpha}\Phi_y\big) + \mathbf{H}_y, \tag{2.26}$$

$$\mathbf{B}_i = \mathbf{D}\left(\mathcal{C}_{\Delta t}^{\alpha}\sum_{p=1}^{k}\mathcal{A}_k^{\alpha}\big(\Phi_y\overline{\Lambda}_i^{p+1-k} - \Phi_y\overline{\Lambda}_i^{p-k}\big) - \mathcal{C}_{\Delta t}^{\alpha}\Phi_y\overline{\Lambda}_i^{k}\right). \tag{2.27}$$

Vector $\overline{\Lambda}_i^m$ is calculated by the multiplying Φ^{-1} in the mth column of $(\mathbf{U}^{*^k})^T$; Φ_y, Υ_y, and Ψ_y are $N_y \times N_y$ matrices which are constructed same as Φ_x, Υ_x, and Ψ_x, respectively, with $\{y_1, y_2, \dots, y_{N_y}\}$. Moreover, $\mathbf{H}_y$ is an $N_y \times N_y$ matrix like $\mathbf{H}_x$ whose first and last rows are equal to the first and the last rows of matrix Υ_y; $\tilde{\mathbf{G}}^i$ and $\tilde{\mathbf{G}}_y^i$ have the following structures:

$$\begin{aligned} \tilde{\mathbf{G}}^i = {} & \operatorname{diag}\big(g\big(\|\nabla u^{*^c}(x_i,y_1)\|\big), g\big(\|\nabla u^{*^c}(x_i,y_2)\|\big), \\ & \dots, g\big(\|\nabla u^{*^c}(x_i,y_{N_y-1})\|\big), g\big(\|\nabla u^{*^c}(x_i,y_{N_y})\|\big)\big), \\ \tilde{\mathbf{G}}_y^i = {} & \operatorname{diag}\Big(\frac{\partial}{\partial y}g\big(\|\nabla u^{*^c}(x_i,y_1)\|\big), \frac{\partial}{\partial y}g\big(\|\nabla u^{*^c}(x_i,y_2)\|\big), \\ & \dots, \frac{\partial}{\partial y}g\big(\|\nabla u^{*^c}(x_i,y_{N_y-1})\|\big)g(x_i,y_{N_y-1}), \frac{\partial}{\partial y}g\big(\|\nabla u^{*^c}(x_i,y_{N_y})\|\big)\Big), \end{aligned}$$

By solving N_x systems (2.25), $\hat{u}^{**^{k+1}}$ is calculated as

$$\mathbf{U}^{**^{k+1}} = \Phi\big[\Gamma_1^{k+1}, \Gamma_2^{k+1}, \dots, \Gamma_{N_x}^{k+1}\big], \tag{2.28}$$

in which

$$\mathbf{U}^{**^{k+1}} = \begin{bmatrix} u^{**^{k+1}}(x_1,y_1) & u^{**^{k+1}}(x_1,y_2) & \dots & u^{**^{k+1}}(x_1,y_{N_y-1}) & u^{**^{k+1}}(x_1,y_{N_y}) \\ u^{**^{k+1}}(x_2,y_1) & u^{**^{k+1}}(x_2,y_2) & \dots & u^{**^{k+1}}(x_2,y_{N_y-1}) & u^{**^{k+1}}(x_2,y_{N_y}) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ u^{**^{k+1}}(x_{N_x-1},y_1) & u^{**^{k+1}}(x_{N_x-1},y_2) & \dots & u^{**^{k+1}}(x_{N_x-1},y_{N_y-1}) & u^{**^{k+1}}(x_{N_x-1},y_{N_y}) \\ u^{**^{k+1}}(x_{N_x},y_1) & u^{**^{k+1}}(x_{N_x},y_2) & \dots & u^{**^{k+1}}(x_{N_x},y_{N_y-1}) & u^{**^{k+1}}(x_{N_x},y_{N_y}) \end{bmatrix}.$$

Finally, $u^{k+1}(x_i,y_j)$, $i = 1,2,\dots,N_x$, $j = 1,2,\dots,N_y$ are obtained from $(\mathbf{U}^{**^{k+1}})^T$.

Algorithm 1 The developed numerical algorithm to solve FPMM

Input:Noisy image (u^0), Δt, M, K, and α
Compute Wendland's functions and their derivatives on the pixels
for $i = 1$ **to** M
$u_1^c \leftarrow$ Solve $\frac{\partial u^c}{\partial t} = \frac{\partial^2 u^c}{\partial x^2}$ with initial condition u^{i-1}
$u^c \leftarrow$ Solve $\frac{\partial u^c}{\partial t} = \frac{\partial^2 u^c}{\partial y^2}$ with initial condition u_1^c
Compute $g(\|\nabla u^c\|)$ and $\frac{\partial g(\|\nabla u^c\|)}{\partial x}$
$u^{*^i} \leftarrow$ Solve Eq. (2.14) by the proposed method
$u_1^{*^c} \leftarrow$ Solve $\frac{\partial u^{*^c}}{\partial t} = \frac{\partial^2 u^{*^c}}{\partial x^2}$ with initial condition u^{*^i}
$u^{*^c} \leftarrow$ Solve $\frac{\partial u^{*^c}}{\partial t} = \frac{\partial^2 u^{*^c}}{\partial y^2}$ with initial condition $u_1^{*^c}$
Compute $g(\|\nabla u^{*^c}\|)$ and $\frac{\partial g(\|\nabla u^{*^c}\|)}{\partial y}$
$u^{**^i} \leftarrow$ Solve Eq. (2.15) by the proposed method
$u^i \leftarrow$ Transpose of u^{**^i}
end for
Output:Denoised image (u^M)

We summarize the proposed approach in Algorithm 1.

3 Numerical experiments

This section is devoted to evaluating the performance of the proposed algorithm by applying it to some examples. This section includes a 512×512 image (Lena) and two 256×256 images (Cameraman and Racoon). We consider Signal-to-Noise Ration (SNR), Peak Signal-to-Noise Ration (PSNR), Structure Similarity Index Measure (SSIM), and Mean Squared Error as four metrics to measure the performance of denoising with FPMM and PMM which are defined as follows:

$$\mathrm{SNR} = 10\log\left(\frac{\sum_{i=1}^{N}\sum_{j=1}^{M}\hat{u}^2(x_i,y_j)}{\sum_{i=1}^{N}\sum_{j=1}^{M}(\hat{u}(x_i,y_j)-u(x_i,y_j))^2}\right), \tag{3.1}$$

$$\mathrm{PSNR} = 10\log\left(\frac{MN\kappa^2}{\sum_{i=1}^{N}\sum_{j=1}^{M}(\hat{u}(x_i,y_j)-u(x_i,y_j))^2}\right), \tag{3.2}$$

$$\mathrm{MSE} = \frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M}\big(\hat{u}(x_i,y_j)-u(x_i,y_j)\big)^2, \tag{3.3}$$

$$\mathrm{SSIM} = \frac{(2\mu_u\mu_{\hat{u}}+c_1)(2\sigma_{u\hat{u}}+c_2)}{(\mu_u^2+\mu_{\hat{u}}^2+c_1)+(\sigma_u^2+\sigma_{\hat{u}}^2+c_2)}, \tag{3.4}$$

in which u is the uncorrupted image, κ is its maximum intensity value, $\hat{u}$ is the reconstructed image, and $M \times N$ is the size of the image. Moreover, μ_u and $\mu_{\hat{u}}$ are the averaging over all the pixel values of the images, $\sigma_{\hat{u}}^2$, σ_u^2 are the variance of all the pixel values of the images and $\sigma_{u\hat{u}}$ is the covariance of u and $\hat{u}$. The c_1, and c_2 coefficients are also defined as follows:

$$c_1 = (k_1 L)^2, \qquad c_2 = (k_2 L)^2,$$

where $k_1, k_2 \ll 1$ are constant coefficients and L is the dynamic range of pixel values.

In order to illuminate the validity of the proposed method, different kinds of noise, i.e., Gaussian, Poisson, and speckle, are added to images, and the denoising process results of FPMM and PMM are demonstrated for each example. Moreover, the appropriate value of α in FPMM is selected according to SNR and PSNR indicators. Also, the stopping criterion for the denoising process is the value of PSNR that should be maximized. All the implementations are done with MATLAB 2017a, on a computer with the following hardware configuration: Intel Core i7, 16 GB RAM.

3.1 Test case 1

As the first example, we consider the image of Lena of size 512×512 pixels. We set $\Delta t = 10^{-3}$, $\delta = 5$. Based on the value of δ, the obtained coefficient matrices are well-posed, and their condition numbers are almost 10^7. According to Fig. 1, $\alpha = 0.9$ can be a good choice for this example. Figure 2 shows the results of FPMM with $\alpha = 0.9$ and PMM for different input noises. This figure implies that FPMM overall is better than the classic one and is applicable for all types of noise. Table 1 reports this example's result for different Gaussian noise variances. It seems that by increasing the noise variance, FPMM performs better than PMM.

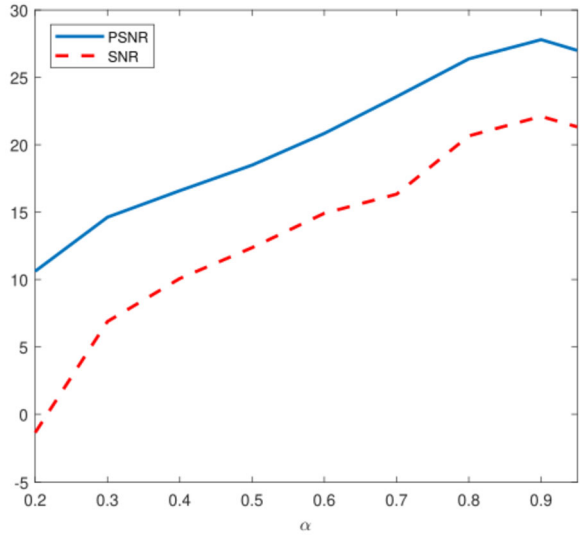

Figure 1 PSNR and SNR for different values of α in test case 1

Table 1 Quantitative analysis of obtained results of FPMM and PMM for denoising the test case 1 with the Gaussian noise

	SNR	PSNR	SSIM	MSE
Noisy image (variance 0.01)	14.5313	20.0380	0.2617	644.5915
FPMM ($\alpha = 0.9$, $K = 100$, iteration = 20)	23.4089	29.1101	0.7721	89.6835
PMM ($K = 4$, iteration = 20)	22.5107	28.1970	0.7148	100.4876
Noisy image (variance 0.05)	13.2799	17.5327	0.2599	1.1476e+03
FPMM ($\alpha = 0.9$, $K = 100$, iteration = 32)	20.9961	26.7036	0.5994	189.9814
PMM ($K = 100$, iteration = 32)	20.5716	25.8288	0.7525	169.9014
Noisy image (variance 0.09)	8.5878	13.6391	0.0980	2.81e+03
FPMM ($\alpha = 0.9$, $K = 105$, iteration = 50)	19.4164	25.1185	0.5634	231.1342
PMM ($K = 100$, iteration = 50)	19.1496	24.18522	0.5023	245.7101

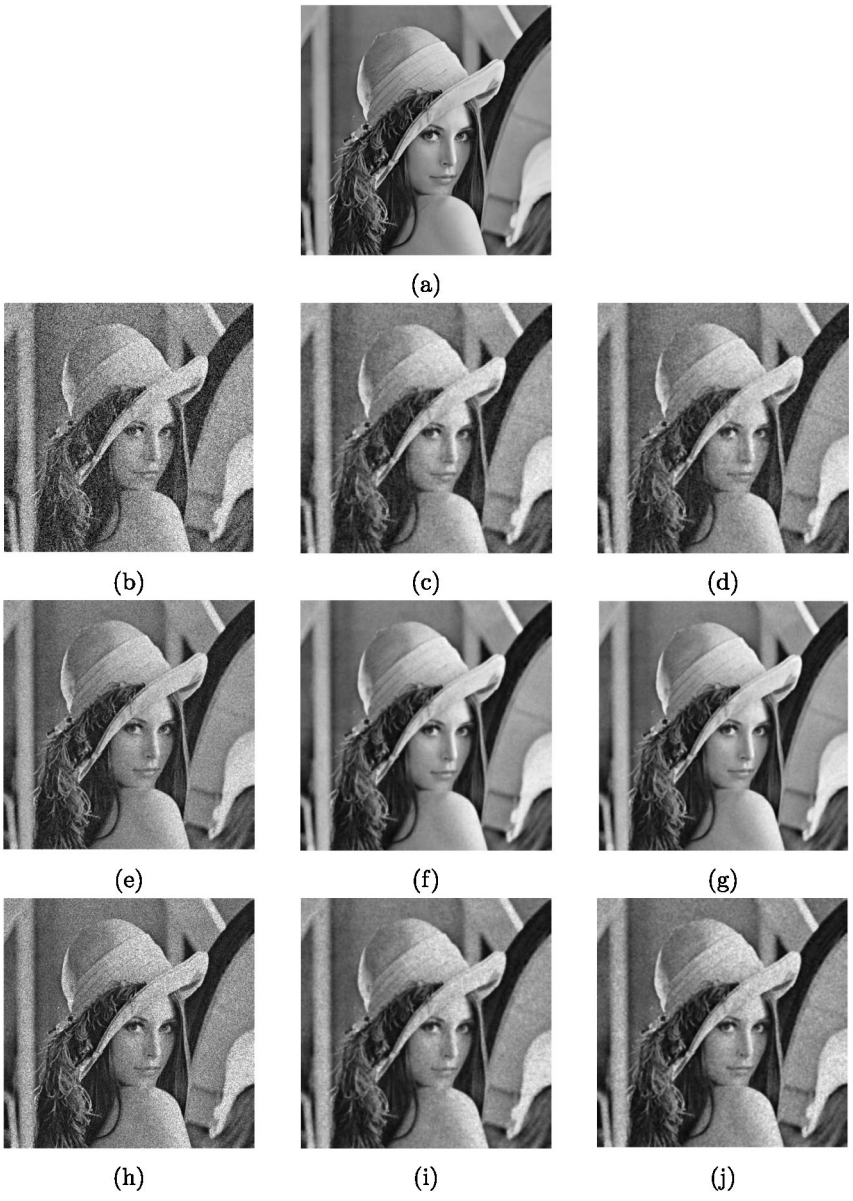

Figure 2 Test case 1: (**a**) the original image Lena, (**b**) noisy image with Gaussian noise (variance 0.02), (**c**) denoised image of the Gaussian noise by FPMM ($\alpha = 0.9$), (**d**) denoised image of the Gaussian noise by PMM, (**e**) noisy image with Poisson noise, (**f**) denoised image of the Poisson noise by FPMM ($\alpha = 0.9$), (**g**) denoised image of the Poisson noise by PMM, (**h**) noisy image with speckle noise, (**i**) denoised image of the speckle noise by FPMM ($\alpha = 0.9$), and (**j**) denoised image of the speckle noise by PMM

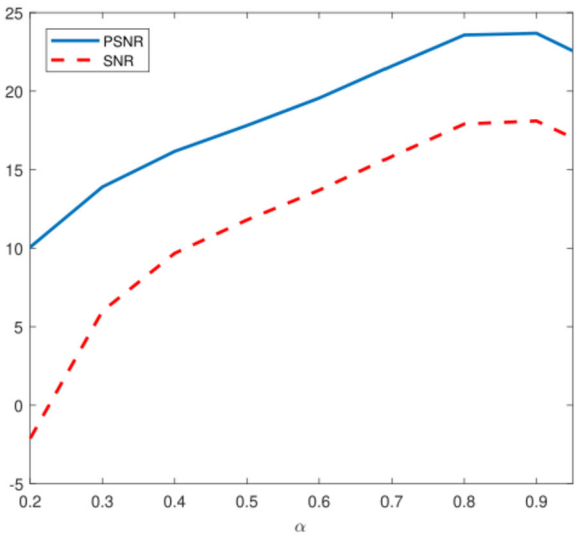

Figure 3 PSNR and SNR for different values of α in test case 2

Table 2 Quantitative analysis of obtained results of FPMM and PMM for denoising the test case 2 with the Gaussian noise

	SNR	PSNR	SSIM	MSE
Noisy image (variance 0.01)	15.0030	20.4392	0.3417	587.7050
FPMM ($\alpha = 0.9$, $K = 90$, iteration = 20)	20.1193	25.7335	0.6025	203.2346
PMM ($K = 100$, iteration = 20)	17.8155	23.3781	0.5845	190.7179
Noisy image (variance 0.03)	10.6852	15.8952	0.2126	1.67e+03
FPMM ($\alpha = 0.9$, $K = 110$, iteration = 42)	16.8226	22.4020	0.4141	419.4402
PMM ($K = 100$, iteration = 42)	16.3136	21.8572	0.3919	431.1948
Noisy image (variance 0.05)	8.8399	13.8739	0.1647	2.66e+03
FPMM ($\alpha = 0.9$, $K = 110$, iteration = 70)	14.8705	20.4323	0.3384	654.0687
PMM ($K = 110$, iteration = 70)	14.4288	19.9599	0.3143	668.8500

3.2 Test case 2

In this case, we regard the Cameraman image of size 256×256. The parameters in this example are $\Delta t = 10^{-3}$, $\delta = 5$, and $\alpha = 0.9$. Figure 3 represents the effect of α on the accuracy of the denoising process. From this figure, $\alpha = 0.9$ is the suitable choice for the order of fractional derivative. Figure 4 depicts the result of denoising by FPMM ($\alpha = 0.9$) and PMM for the Gaussian (variance 0.02), Poisson, and speckle noises. In addition, Table 2 contains the values of performance measurement metrics for FPMM and PMM against different Gaussian noise variances. It can be concluded that FPMM is stronger than PMM, and fractional derivative lets us denoise images more effectively.

3.3 Test case 3

In the last example, the Racoon image of size 256×256 is considered as the test case. The numerical approach is applied with $\Delta t = 10^{-2}$, $\alpha = 0.8$, and $\delta = 5$. Like the previous examples, $\alpha = 0.8$ is selected based on the values of PSNR and SNR in Fig. 5. Additionally, Table 3 analyzes the results of FPMM and PMM. Both models can eliminate the noise; however, FPMM is much better for a higher level of noise variance. For instance, for variance 0.01, the MSE of PMM is less than that of FPMM, but as the noise intensity increases,

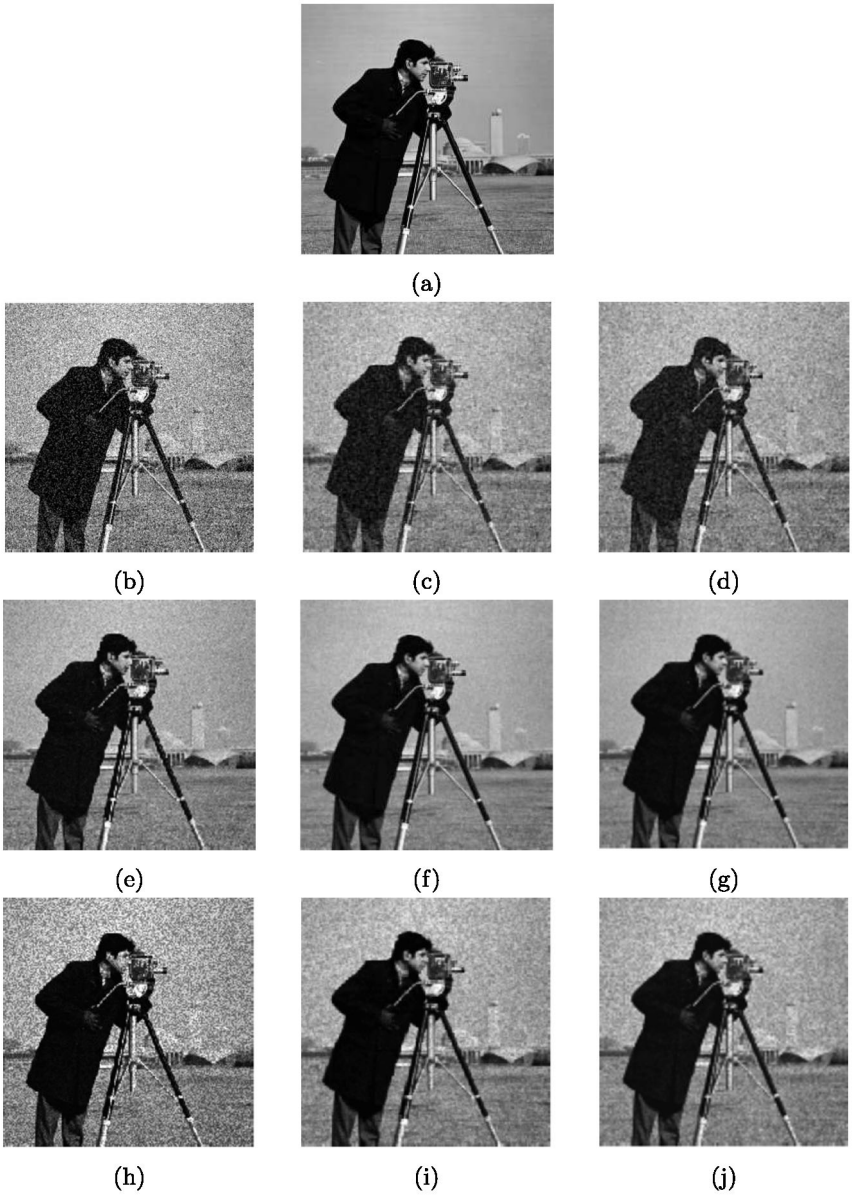

Figure 4 Test case 2: (**a**) the original image of a cameraman, (**b**) noisy image with Gaussian noise (variance 0.02), (**c**) denoised image of the Gaussian noise by FPMM ($\alpha = 0.9$), (**d**) denoised image of the Gaussian noise by PMM, (**e**) noisy image with Poisson noise, (**f**) denoised image of the Poisson noise by FPMM ($\alpha = 0.9$), (**g**) denoised image of the Poisson noise by PMM, (**h**) noisy image with speckle noise, (**i**) denoised image of the Speckle noise by FPMM ($\alpha = 0.9$), and (**j**) denoised image of the speckle noise by PMM

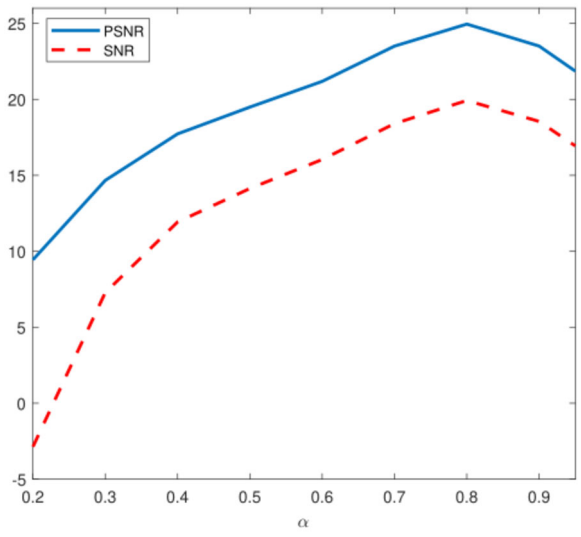

Figure 5 PSNR and SNR for different values of α in test case 3

Table 3 Quantitative analysis of obtained results of FPMM and PMM for denoising the test case 3 with the Gaussian noise

	SNR	PSNR	SSIM	MSE
Noisy image (variance 0.01)	15.2463	20.950	0.4278	636.1865
FPMM ($\alpha = 0.8$, $K = 90$, iteration = 21)	20.7950	25.8404	0.7078	204.7152
PMM ($K = 90$, iteration = 21)	20.4959	25.4701	0.6501	195.3838
Noisy image (variance 0.03)	10.8789	15.5071	0.2445	1.80e+03
FPMM ($\alpha = 0.8$, $K = 95$, iteration = 37)	19.0699	24.1365	0.5789	303.7352
PMM ($K = 95$, iteration = 37)	18.7141	23.6894	0.5689	317.1837
Noisy image (variance 0.05)	9.0813	13.5449	0.1748	2.87e+03
FPMM ($\alpha = 0.8$, $K = 110$, iteration = 82)	17.7842	22.8075	0.4986	408.3009
PMM ($K = 110$, iteration = 82)	14.2662	19.2006	0.3647	734.4369

the MSE model will be better. In Fig. 6, we show the performance of FPMM (with $\alpha = 0.8$) and PMM for eliminating various kinds of noise for the Racoon example. In summary, the superiority of FPMM with $\alpha = 0.8$ is obvious in this test case.

4 Conclusion

In this study, we proposed a practical and accurate numerical approach based on the combination of Trotter splitting and compactly supported radial basis function methods to investigate the time-fractional Perona–Malik model (FPMM) which is a strong model for image denoising. The operator splitting approach allowed us to divide the main problem with a large domain into simpler subproblems, significantly decreasing computational costs. By employing the proposed scheme, the model was reduced to sparse and well-posed linear algebraic systems; thus, these systems could be solved by classical algorithms like the LU approach. Various test cases were examined in different factors such as SNR, PSNR, MSE, and SSIM to demonstrate the efficiency and accuracy of the proposed method. We showed that by choosing an appropriate value for fractional derivative order α, FPMM can outperform the classical one. In this paper, the proper value of α was selected such that the values of PSNR and SNR were maximized.

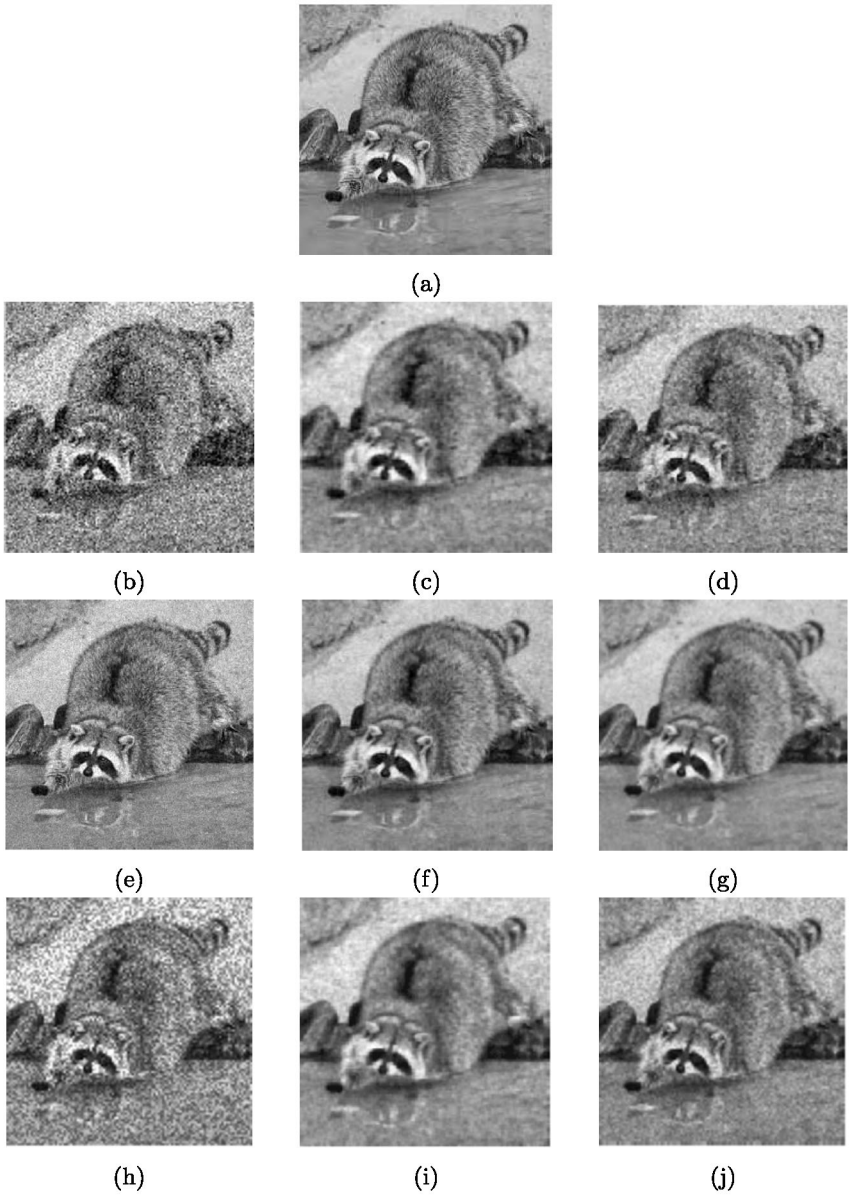

Figure 6 Test case 3: (**a**) the orginal image Racoon, (**b**) noisy image with Gaussian noise (variance 0.02), (**c**) denoised image of the Gaussian noise by FPMM ($\alpha = 0.8$), (**d**) denoised image of the Gaussian noise by PMM, (**e**) noisy image with Poisson noise, (**f**) denoised image of the Poisson noise by FPMM ($\alpha = 0.8$), (**g**) denoised image of the Poisson noise by PMM, (**h**) noisy image with speckle noise, (**i**) denoised image of the speckle noise by FPMM ($\alpha = 0.8$), and (**j**) denoised image of the speckle noise by PMM

Acknowledgements
The authors are very grateful to the reviewer for carefully reading this paper and for their comments and suggestions, which have improved the quality of the paper.

Funding
Not applicable.

Availability of data and materials
Data sharing not applicable to this article as no especial datasets were generated during the current study.

Declarations

Competing interests
The authors declare that they have no competing interests.

Author contribution
JM: Software, Validation, Formal analysis, Conceptualization, Methodology, Writing—Original Draft, Supervision. BHS-M: Investigation, Writing—Review Editing. All authors read and approved the final manuscript.

Publisher's Note

Received: 22 March 2022 Accepted: 18 August 2022 Published online: 01 September 2022

References

1. Lotfi, Y., Parand, K.: Anti-aliasing of gray-scale/color/outline images: looking through the lens of numerical approaches for PDE-based models. Comput. Math. Appl. **113**(1), 130–147 (2020)
2. Singh, A., Agarwal, P., Chand, M.: Image encryption and analysis using dynamic AES. In: 2019 5th International Conference on Optimization and Applications (ICOA) (2019)
3. Sidi ammi, M.R., Jamiai, I.: Finite difference and Legendre spectral method for a time-fractional diffusion–convection equation for image restoration. Discrete Contin. Dyn. Syst., Ser. A **11**(1), 103–117 (2020)
4. Gu, Y.: Finite element numerical approximation for two image denoising models. Circuits Syst. Signal Process. **39**, 2042–2062 (2020)
5. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Phys. D: Nonlinear Phenom. **60**(1–4), 259–268 (1992)
6. Deng, L., Zhu, H., Yang, Z., Li, Y.: Hessian matrix-based fourth-order anisotropic diffusion filter for image denoising. Opt. Laser Technol. **110**, 184–190 (2019)
7. Abdallah, M.B., Malek, J., Azar, A.T., Belmabrouk, H., Monreal, J.E., Krissian, K.: Adaptive noise-reducing anisotropic diffusion filter. Neural Comput. Appl. **27**(5), 1273–1320 (2016)
8. Li, Y., Ding, Y., Li, T.: Nonlinear diffusion filtering for peak-preserving smoothing of a spectrum signal. Chemom. Intell. Lab. Syst. **159**, 157–165 (2016)
9. Barbu, T.: Robust anisotropic diffusion scheme for image noise removal. Proc. Comput. Sci. **35**, 522–530 (2014)
10. Koenderink, J.J., Ding, Y., Li, T.: The structure of images. Biol. Cybern. **50**, 363–370 (1984)
11. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. IEEE Trans. Pattern Anal. Mach. Intell. **12**, 629–639 (1990)
12. Kamranian, M., Dehghan, M., Tatari, M.: An image denoising approach based on a meshfree method and the domain decomposition technique. Eng. Anal. Bound. Elem. **39**, 101–110 (2014)
13. Catté, F., Lions, P.-L., Morel, J.-M., Coll, T.: Image selective smoothing and edge detection by nonlinear diffusion. SIAM J. Numer. Anal. **29**, 182–193 (1992)
14. Handlovicova, A., Mikula, K., Sgallari, F.: Variational numerical methods for solving nonlinear diffusion equations arising in image processing. J. Vis. Commun. Image Represent. **13**(1–2), 217–273 (2002)
15. Salinas, H.M., Fernandez, D.C.: Comparison of PDE-based nonlinear diffusion approaches for image enhancement and denoising in optical coherence tomography. IEEE Trans. Med. Imaging **26**(6), 761–771 (2007)
16. Shangguan, H., Zhang, X., Cui, X., Liu, Y., Zhang, Q., Gui, Z.: Sinogram restoration for low-dose X-ray computed tomography using regularized Perona–Malik equation with intuitionistic fuzzy entropy. Signal Image Video Process. **13**, 1511–1519 (2019)
17. Firsov, D., Lui, S.: Domain decomposition methods in image denoising using Gaussian curvature. J. Comput. Appl. Math. **193**(2), 460–473 (2006)
18. Hjouji, A., Jourhmane, M., EL-Mekkaoui, J., Es-sabry, M.: Mixed finite element approximation for bivariate Perona–Malik model arising in 2D and 3D image denoising. 3D Res. **9**(36), 460–473 (2018)
19. Jun, Z., Wei, Z., Xiao, L.: Adaptive fractional-order multi-scale method for image denoising. J. Math. Imaging Vis. **43**, 39–49 (2012)
20. Momani, S.: An algorithm for solving the fractional convection–diffusion equation with nonlinear source term. J. Comput. Phys. **12**(7), 1283–1290 (2007)
21. Agarwal, P., Baleanu, D., Chen, Y.Q., Momani, S., Machado, J.A.T.: Fractional Calculus, ICFDA 2018, Amman, Jordan. Springer Singapore **12**(7), 1283–1290 (2019)
22. Agarwal, P., Ramadan, M.A., Regeh, A.A.M., Hadhoud, A.R.: A fractional-order mathematical model for analyzing the pandemic trend of COVID-19. Math. Methods Appl. Sci. **45**(8), 4625–4642 (2022)
23. Jafari, H., Momani, S.: Solving fractional diffusion and wave equations by modified homotopy perturbation method. Phys. Lett. A **370**(5–6), 388–396 (2007)
24. Chu, Y.-M., Shah, N.A., Agrawal, P., Chung, J.D.: Analysis of fractional multi-dimensional Navier–Stokes equation. Adv. Differ. Equ. **2021**, 91 (2021)
25. Sunarto, A., Agrawal, P., Sulaiman, J., Chew, J.V.L.: Computational approach via half-sweep and preconditioned AOR for fractional diffusion. Intell. Autom. Soft Comput. **31**(2), 1173–1184 (2022)
26. Sunarto, A., Agrawal, P., Sulaiman, J., Chew, J.V.L., Aruchunan, E.: Iterative method for solving one-dimensional fractional mathematical physics model via quarter-sweep and PAOR. Adv. Differ. Equ. **2021**, 147 (2021)
27. Al-Smadi, M., Momani, S., Djeddi, N., El-Ajou, A., Al-Zhour, Z.: Adaptation of reproducing kernel method in solving Atangana–Baleanu fractional Bratu model. Int. J. Dyn. Control (2022). https://doi.org/10.1007/s40435-022-00961-1
28. Hasan, S., Al-Smadi, M., Dutta, H., Momani, S., Hadid, S.: Multi-step reproducing kernel algorithm for solving Caputo–Fabrizio fractional stiff models arising in electric circuits. Soft Comput. **26**, 3713–3727 (2022)

29. Miller, K.S., Ross, B.: An Introduction to the Fractional Calculus and Fractional Differential Equations, 1st edn. Wiley, New York (1993)
30. Ross, B.: A brief history and exposition of the fundamental theory of fractional calculus. In: Fractional Calculus and Its Applications. Springer, Berlin (1975)
31. Hilfer, R.: Applications of Fractional Calculus in Physics. World Scientific, Singapore (2000)
32. Blank, L.: Numerical Treatment of Differential Equations of Fractional Order, Manchester Center for Computational Mathematics. University of Manchester, Manchester (1996)
33. Caputo, M.: Linear model of dissipation whose q is almost frequency independent. Geophys. J. R. Astron. Soc. **13**, 529–539 (1967)
34. Caputo, M.: Elasticita e Dissipazione. Zanichelli, Bologna (1969)
35. Podlubny, I.: Fractional Differential Equations: An Introduction to Fractional Derivatives, Fractional Differential Equations, to Methods of Their Solution and Some of Their Applications, vol. 198. Elsevier, Amsterdam (1998)
36. Bu, W., Tang, Y., Wu, Y., Yang, J.: Crank–Nicolson ADI Galerkin finite element method for two-dimensional fractional Fitzhugh–Nagumo monodomain model. Appl. Math. Comput. **257**, 355–364 (2015)
37. Atangana, A., Baleanu, D.: New fractional derivatives with nonlocal and non-singular kernel: theory and application to heat transfer model. Therm. Sci. **20**, 763–769 (2016)
38. Karaagac, B.: Analysis of the cable equation with non-local and non-singular kernel fractional derivative. Eur. Phys. J. Plus **133**(2), 54 (2018)
39. Atangana, A., Gómez-Aguilar, J.F.: A new derivative with normal distribution kernel: theory, methods and applications. Phys. A, Stat. Mech. Appl. **476**, 1–14 (2017)
40. Pedram, G., Micael, S.C., Jon, A.B., Nuno, M.F.F.: An efficient method for segmentation of images based on fractional calculus and natural selection. Expert Syst. Appl. **39**, 12407–12417 (2012)
41. Yin, X., Zhou, S., Jon, A.B., Siddique, M.A.: Fractional nonlinear anisotropic diffusion with p-Laplace variation method for image restoration. Multimed. Tools Appl. **75**, 4505–4526 (2016)
42. Yi-Fei, P., Ji-Liu, Z., Xiao, Y.: Fractional differential mask: a fractional differential-based approach for multiscale texture enhancement. IEEE Trans. Image Process. **19**, 491–511 (2010)
43. Hemami, M., Rad, J.A., Parand, K.: The use of space-splitting RBF-FD technique to simulate the controlled synchronization of neural networks arising from brain activity modeling in epileptic seizures. J. Comput. Sci. **42**, 101090 (2020)
44. Safdari-Vaighani, A., Larsson, E., Heryudono, A.: Radial basis function methods for the Rosenau equation and other higher order PDEs. J. Sci. Comput. **75**, 84–93 (2018)
45. Fasshauer, G.E.: Meshfree Approximation Methods with MATLAB, vol. 6. World Scientific, Singapore (2007)
46. Rad, J.A., Kazem, S., Parand, K.: A numerical solution of the nonlinear controlled Duffing oscillator by radial basis functions. Comput. Math. Appl. **64**, 2049–2065 (2012)
47. Rad, J.A., Höök, L.J., Larsson, E., von Sydow, L.: Forward deterministic pricing of options using Gaussian radial basis functions. J. Comput. Sci. **24**, 209–217 (2018)
48. Wendland, H.: Scattered Data Approximation. Cambridge University Press, New York (2005)
49. Rad, J.A., Parand, K., Abbasbandy, S.: Local weak form meshless techniques based on the radial point interpolation (RPI) method and local boundary integral equation (LBIE) method to evaluate European and American options. Commun. Nonlinear Sci. Numer. Simul. **22**(1–3), 1178–1200 (2015)
50. Hemami, M., Parand, K., Rad, J.A.: Numerical simulation of reaction–diffusion neural dynamics models and their synchronization/desynchronization: application to epileptic seizures. Comput. Math. Appl. **78**(11), 3644–3677 (2019)
51. Hemami, M., Rad, J.A., Parand, K.: Phase distribution control of neural oscillator populations using local radial basis function meshfree technique with application in epileptic seizures: a numerical simulation approach. Commun. Nonlinear Sci. Numer. Simul. **103**, 105961 (2021)
52. Wendland, H.: Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. Adv. Comput. Math. **78**(1), 389–396 (1995)
53. Moayeri, M.M., Rad, J.A., Parand, K.: Dynamical behavior of reaction–diffusion neural networks and their synchronization arising in modeling epileptic seizure: a numerical simulation study. Comput. Math. Appl. **80**(8), 1887–1927 (2020)
54. Holden, H., Karlsen, K.H., Lie, K.A., Risebro, N.H.: Splitting Methods for Partial Differential Equations with Rough Solutions: Analysis and Matlab Programs. Eur. Math. Soc., Zurich (2010)
55. Hellander, P., Lawson, P.J., Drawert, B., Petzold, L.: Local error estimates for adaptive simulation of the reaction–diffusion master equation via operator splitting. J. Comput. Phys. **266**, 89–100 (2014)
56. Alonso-Mallo, I., Cano, B., Reguera, N.: Avoiding order reduction when integrating reaction–diffusion boundary value problems with exponential splitting methods. J. Comput. Appl. Math. **357**, 228–250 (2019)
57. Lin, Y., Xu, C.: Finite difference/spectral approximations for the time-fractional diffusion equation. J. Comput. Phys. **255**(2), 1533–1552 (2007)