International Journal of Applied Mathematics

Volume 38 No. 2 2025, 205–226
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)
doi: http://dx.doi.org/10.12732/ijam.v38i2.4

A RADIAL BASIS FUNCTION - FINITE DIFFERENCE APPROACH FOR TIME-DEPENDENT PARAMETER IDENTIFICATION OF THE HEAT EQUATION

Nadun Kulasekera Mudiyanselage [1,§],
Dulashini Karunarathna [2]

[1] Mount St Mary's University
Department of Mathematics and Computer Science
Emmitsburg, Maryland, USA
e-mail: n.l.mudiyanselage@msmary.edu ([§] corresponding author)
[2] University of Peradeniya
Postgraduate Institute of Science
Peradeniya, SRI LANKA
e-mail: dulashini@wyb.ac.lk

Received: 31 January 2025

.

Abstract

This manuscript focuses on solving inverse problems that approximate the time-dependent heat conductivity coefficients and the solutions of parabolic partial differential equations. Solving these problems poses significant challenges due to their ill-posed nature, as the solutions may not depend continuously on the input data, particularly in the presence of noise. As a result, it is crucial to design stable and reliable numerical methods. This paper introduces a new, straightforward, and effective framework based on the Radial Basis Function - Finite Difference (RBF-FD) method to address inverse time-dependent parameter identification problems in parabolic partial differential equations. The RBF-FD method offers a computationally efficient higher-order approach to tackle these problems, overcoming some common limitations of existing numerical methods, such as instability, high computational demands, and reduced accuracy.

Math. Subject Classification: 65M20, 65M32, 35F05

Key Words and Phrases: radial basis functions, inverse problems, augmented polynomials, time-dependent parameters

1. Introduction

Approximating the heat conductivity coefficient in parabolic partial differential equations has a wide variety of applications. In the literature, these problems are also known as a subclass of inverse problems. Applications are often found in the fields of engineering [1], biology [2], and geology [3]. However, solving these problems is challenging because they may be ill-posed. The problems often fail to meet the third Hadamard criteria, where the solutions may not continuously depend on input data due to noise [4]. Hence, developing stable, robust numerical methods is of the utmost importance. Past studies have proposed regularization techniques [4], finite difference methods [5], and finite element methods [6] to tackle these problems. However, the main drawbacks of some of these problems include issues with stability, high computational cost, and low accuracy [7]. Furthermore, the existing methods can be too complicated for a non-mathematical expert to implement easily in an application field. This manuscript proposes a simple, robust Radial Basis Function - Finite

Difference framework (RBF-FD) to solve inverse time-dependent parameter identification problems for parabolic partial differential equations. The RBF-FD method has been used extensively in geology [8, 9]. The method is computationally efficient, simple to code, and yields higher-order convergence [7]. A significant advantage of the RBF-FD method is that the programming complexity does not increase as the dimensions of the underlying problem increase. This manuscript only focuses on one-dimensional partial differential equations with time-dependent heat conductivity coefficients. The goal is to simultaneously approximate the heat conductivity coefficients (multiple) and the solution of the given partial differential equation subjected to typical Dirichlet or Neumann boundary conditions with additional overspecified boundary conditions. The organization of the rest of the paper is as follows. This section introduces the problem and provides the methodology to tackle the problem in the second section. The rest of the manuscript presents numerical results and a conclusion.

1.1. **Problem formulation.** Consider the one-dimensional initial value problem:

$$\begin{cases} c(t)\dfrac{\partial u}{\partial t}(x,t) & = k(t)\dfrac{\partial^2 u}{\partial x^2}(x,t) + f(x,t), \quad (x,t) \in \Omega, \\ u(x,0) & = g(x), \qquad\qquad\qquad\qquad x \in (0,\ell), \end{cases} \tag{1}$$

where $\Omega = [0,\ell] \times [0,T]$. Furthermore, equation (1) is subjected to boundary conditions:

$$\begin{cases} u(0,t) & = \phi_1(t), \quad t \in [0,T], \\ u(\ell,t) & = \phi_2(t), \quad t \in [0,T]. \end{cases} \tag{2}$$

Equation (1) also has one of the following set of overspecified conditions in the case of solving the inverse problem:

$$\{\ u(x^*,t) = q(x^*,t), \quad (x^*,t) \in \Omega, \tag{3}$$

$$\begin{cases} -k(t)u_x(0,t) & = \mu_1(t), \quad t \in [0,T], \\ -k(t)u_x(\ell,t) & = \mu_2(t), \quad t \in [0,T], \end{cases} \tag{4}$$

where x^* is any point in $(0,\ell)$. Notice that equation (3) is a Dirichelet-type, and equation (4) is a Neumann-type overspecified boundary condition. This manuscript aims to develop an RBF-FD methodology to approximate $k(t)$, $c(t)$, and $u(x,t)$ simultaneously. The next few subsections present an introduction to the RBF-FD method.

2. **RBF-FD framework for inverse problems**

Equation (1) can be discretized in space using the RBF-FD method, and for the time-dependent portion, the Forward Euler method was chosen for simplicity. However, the proposed methodology can be easily used with any explicit time integrator. This manuscript will only focus on using explicit time integrators. Assume D_{xx} as the RBF-FD discretized differential operator, defined as:

$$D_{xx} = k(t)\frac{\partial^2}{\partial x^2}.$$

The construction of the discretized differential operator is as follows. Given a pair of N node locations and function evaluations $(\underline{x}_i, f(\underline{x}_i))$, where $\underline{x}_i \in \mathbb{R}^n$, $f_i \in \mathbb{R}$, we can construct a linear combination as:

$$s\left(\vec{x}\right) = \sum_{j=1}^{N} \lambda_j \, \phi\left(\|\vec{x} - \vec{x}_j\|\right), \tag{5}$$

where ϕ, the basis function is a radially symmetric function of distance, and $\|\cdot\|$ is the Euclidean norm. ϕ is also called a radial basis function. This represents an interpolant and λ_j are unknowns that satisfy the interpolant condition $s\left(\vec{x}_j\right) = f\left(\vec{x}_j\right)$. The problem can be rewritten as a linear system $A\vec{\lambda} = \vec{f}$. In this manuscript, A will be referred to as the collocation matrix. The system of equations will be of the form:

$$\begin{bmatrix} \phi\left(\|\vec{x}_1 - \vec{x}_1\|\right) & \phi\left(\|\vec{x}_1 - \vec{x}_2\|\right) & \cdots & \phi\left(\|\vec{x}_1 - \vec{x}_N\|\right) \\ \phi\left(\|\vec{x}_2 - \vec{x}_1\|\right) & \phi\left(\|\vec{x}_2 - \vec{x}_2\|\right) & \cdots & \phi\left(\|\vec{x}_2 - \vec{x}_N\|\right) \\ \vdots & \vdots & \ddots & \vdots \\ \phi\left(\|\vec{x}_N - \vec{x}_1\|\right) & \phi\left(\|\vec{x}_N - \vec{x}_2\|\right) & \cdots & \phi\left(\|\vec{x}_N - \vec{x}_N\|\right) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix} = \begin{bmatrix} f\left(\vec{x}_1\right) \\ f\left(\vec{x}_2\right) \\ \vdots \\ f\left(\vec{x}_N\right) \end{bmatrix}.$$

Each row of the collocation matrix evaluates the radial basis function ϕ at a central node x_i for all N nodes in the dataset. If A is invertible, we can determine the weights as $\bar{\lambda} = A^{-1}\bar{f}$.

Mainly, two types of radial basis functions exist in the literature: infinitely smooth and piecewise smooth radial basis functions. Gaussian RBF is an example of an infinitely smooth radial function that has the form $e^{-(\varepsilon r)^2}$. The r here is the distance, and ε is the shape parameter. All infinitely smooth radial functions contain a shape parameter, which dictates the shape of the function. Polyharmonic spline (PHS) is an example of a piece-wise smooth radial basis function and has the form r^{2m+1}. Notice that PHS does not have a shape parameter. This is a crucial difference between the two classes and plays a key role in selecting between global and local RBF approaches.

The same methodology we used with interpolation can also approximate linear operators. Since the second-order derivative is a linear operator, we can define $L = \dfrac{\partial^2}{\partial x^2}$. We can impose the operator on the interpolant as in equation (5):

$$Ls\left(\vec{x}_i\right) = L\sum_{j=1}^{N} \lambda_j \phi\left(\|\vec{x}_i - \vec{x}_j\|\right) = \sum_{j=1}^{N} \lambda_j L\phi\left(\|\vec{x}_i - \vec{x}_j\|\right), \; i = 1, \dots, N.$$

Notice that the weights are constants and do not change from equation (5). Hence we can express system of equations from above as $L\vec{s} = \tilde{A}\vec{\lambda}$, where

$$\tilde{A} = \begin{bmatrix} L\phi\left(\|\vec{x}_1 - \vec{x}_1\|\right) & L\phi\left(\|\vec{x}_1 - \vec{x}_2\|\right) & \cdots & L\phi\left(\|\vec{x}_1 - \vec{x}_N\|\right) \\ L\phi\left(\|\vec{x}_2 - \vec{x}_1\|\right) & L\phi\left(\|\vec{x}_2 - \vec{x}_2\|\right) & \cdots & L\phi\left(\|\vec{x}_2 - \vec{x}_N\|\right) \\ \vdots & \vdots & \ddots & \vdots \\ L\phi\left(\|\vec{x}_N - \vec{x}_1\|\right) & L\phi\left(\|\vec{x}_N - \vec{x}_2\|\right) & \cdots & L\phi\left(\|\vec{x}_N - \vec{x}_N\|\right) \end{bmatrix}.$$

Using $\bar{\lambda} = A^{-1}\bar{f}$, we obtain

$$L\vec{s} = \tilde{A}A^{-1}\vec{f} = D\,\vec{f} \approx L\vec{f}.$$

Here, D is called the Differentiation matrix (D_{xx} in our case), and it is the approximated second-order derivative operator L. In order to obtain D, one has to use every node in the domain (N), providing us with a global approach. This will produce a dense differentiation matrix. Therefore, using this approach is computationally expensive when N is large. However, the RBF-FD method takes a local approach, which helps to significantly reduce computational costs by producing sparser differentiation matrices. The next subsection focuses on introducing the RBF-FD method.

2.1. **Local RBF approach.** Derivative properties are inherently local, making it sufficient to consider only the nearest neighbors of a central node when calculating weights for the differentiation matrix. From our global data set of N nodes, we use n nearest neighboring nodes per each node to gather information ($n << N$) instead of using information from all the nodes in the domain. Hence, the approximations become local. This idea lays the foundation for the Radial Basis Function - Finite Difference Method (RBF-FD). The selection of the RBF is also important. One challenge with using an infinitely smooth RBF is selecting a suitable shape parameter. It is shown that when the shape parameter approaches zero, the error continues to decay, but the numerical ill-conditioning of the collocation matrix gets worse [7]. It has been shown that error reaches an optimal value when the shape parameter shrinks. Still, even if the problem is well posed, the conditioning of the numerical method causes the error to jump back up [10]. Hence, one must find the optimal shape parameter that balances out error and conditioning. However, no method exists to find the optimal shape parameter for a given problem. This issue can be avoided using piece-wise smooth RBFs. However, piecewise smooth RBFs present their own challenges, such as the potential for a singular collocation matrix [11]. This limitation can be addressed by including additional polynomial terms in the interpolant [7]. Therefore, we can redefine RBF interpolant for n nearest neighbors as:

$$s\left(\vec{x}\right) = \sum_{i=1}^{n} \lambda_j \left\|\vec{x} - \vec{x}_i\right\|^m + \sum_{j=1}^{k} \gamma_j P_j\left(\vec{x}_i\right), \;\; i = 1, \ldots, N. \tag{6}$$

where m is an odd number, P_j represents a term of a degree k polynomial. For instance, a first-degree augmented two-dimensional polynomial is of the form $\gamma_1 + \gamma_2 x + \gamma_3 y$. Then polynomials basis 1, x, and y are the P_js represented in equation (6).

Equation (6) is subject to the following conditions:

(1) $s\left(\vec{x}_i\right) = f\left(x_i\right)$;

(2) $\sum_{i=1}^{n} \lambda_i P_i\left(\vec{x}_i\right) = 0$.

Subsequently, we construct our differentiation matrix in a manner akin to the previous method, yielding the linear system $A\vec{\lambda} = \vec{f}$, which is:

$$\left[\begin{array}{c|c} \Phi & \mathbf{P} \\ \hline \mathbf{P}^T & 0 \end{array}\right] \left[\begin{array}{c} \vec{\lambda} \\ \hline \vec{\gamma} \end{array}\right] = \left[\begin{array}{c} \vec{f} \\ \hline \vec{0} \end{array}\right],$$

where Φ is the collocation matrix, and it can be defined as

$$\begin{pmatrix} \phi\left(\left\|\vec{x}_1 - \vec{x}_1\right\|\right) & \phi\left(\left\|\vec{x}_1 - \vec{x}_2\right\|\right) & \cdots & \phi\left(\left\|\vec{x}_1 - \vec{x}_n\right\|\right) \\ \phi\left(\left\|\vec{x}_2 - \vec{x}_1\right\|\right) & \phi\left(\left\|\vec{x}_2 - \vec{x}_2\right\|\right) & \cdots & \phi\left(\left\|\vec{x}_2 - \vec{x}_n\right\|\right) \\ \vdots & \vdots & \ddots & \vdots \\ \phi\left(\left\|\vec{x}_n - \vec{x}_1\right\|\right) & \phi\left(\left\|\vec{x}_n - \vec{x}_2\right\|\right) & \cdots & \phi\left(\left\|\vec{x}_n - \vec{x}_n\right\|\right) \end{pmatrix},$$

and P is the transpose of Vandermonde matrix:

$$\begin{pmatrix} P_1(\vec{x}_1) & P_1(\vec{x}_2) & \cdots & P_1(\vec{x}_n) \\ P_2(\vec{x}_1) & P_2(\vec{x}_2) & \cdots & P_2(\vec{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ P_k(\vec{x}_1) & P_k(\vec{x}_2) & \cdots & P_k(\vec{x}_n) \end{pmatrix}.$$

We also can define A as:

$$A = \left[\begin{array}{c|c} \Phi & \mathbf{P} \\ \hline \mathbf{P}^T & 0 \end{array}\right].$$

Using the local approach, we can now apply the operator L to equation (6), which yields:

$$Ls(\vec{x}) = \sum_{i=1}^{n} -\lambda_j m \left\|\vec{x} - \vec{x}_i\right\|^{m-1} + \sum_{j=1}^{k} \gamma_j L P_j(\vec{x}_i), \ \ i = 1, \ldots, N. \tag{7}$$

Hence we obtain $L\vec{s} = \tilde{A}\vec{\lambda}$, where

$$\tilde{A} = \left[L\Phi \mid L\mathbf{P}\right].$$

Then we can obtain the discretized operator as:

$$L\vec{s} = \tilde{A}A^{-1}\vec{f} = D_{xx}\,\vec{f} \approx L\vec{f}.$$

Furthremore, the fully discretized equation (1) can be expressed as:

$$c^i \frac{u^{i+1} - u^i}{\Delta t} = D_{xx}u^i + f^i, \tag{8}$$

where i stands for indexing for time steps. Solving equation (8) for u^i yields:

$$u^{i+1} = u^i + \frac{\Delta t}{c^i}\,D_{xx}\,u^i + \frac{\Delta t}{c^i}\,f^i.$$

Furthermore,

$$u^{i+1} = u^i(I + \frac{\Delta t}{c^i}\,D_{xx}) + \frac{\Delta t}{c^i}\,f^i,$$

where I is the $N \times N$ identity matrix,

$$u^{i+1} = Du^i + \frac{\Delta t}{c^i}\,f^i, \tag{9}$$

where $D = (I + \frac{\Delta t}{c^i} D_{xx})$,

$$D = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ \cdots & (I + \frac{\Delta t}{c^i} D_{xx}) & \cdots & \cdots & \cdots \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

The reason for vectors $\begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}_{1\times N}$, and $\begin{bmatrix} 0 & 0 & \cdots & 1 \end{bmatrix}$ in D is to handle the Dirichlet boundary conditions.

The above methodology works only if we know $k(t)$, and $c(t)$, but in the context of this manuscript, $k(t)$, and $c(t)$ are also unknown. However, the methodology can be adjusted using the overspecified boundary conditions. Let us consider two different problems. One is only approximating $k(t)$, and $u(x,t)$ with assuming you know $c(t)$, and the other is approximating all three unknowns. First, let us only focus on approximating $k(t)$, and $u(x,t)$. In the case of the Dirichlet overspecified boundary condition:

$$u(x^*, t) = q(x^*, t). \tag{10}$$

We can use the overspecified boundary condition to find k (constant or time-dependent). We can rewrite equation (8) at $x = x^*$ as:

$$c^i \frac{u(x^*)^{i+1} - u(x^*)^i}{\Delta t} = k^i \left(D_{xx} u^i\right)_{x=x^*} + f(x^*)^i. \tag{11}$$

D_{xx} and D_x represent the discretized differentiation matrices of the second partial and first partial derivatives, respectively.

Using equation (10) in equation (11) we get:

$$c^i \frac{q(x^*)^{i+1} - q(x^*)^i}{\Delta t} = k^i \left(D_{xx} u^i\right)_{x=x^*} + f(x^*)^i. \tag{12}$$

Notice that the only unknown here is k^i. Equation (12) can be solved for k and then used to solve equation (9). Therefore, the algorithm is to first solve for k at the current time step and then use it to approximate u.

Now let us focus on Neumann-type overspecified boundary conditions:

$$\begin{cases} -k(t)u_x(0,t) &= \mu_1(t) \quad t \in [0,T], \\ -k(t)u_x(\ell,t) &= \mu_2(t) \quad t \in [0,T]. \end{cases} \tag{13}$$

In order to incorporate them, we can discretize the conditions as follows:

$$\begin{cases} -k(t)D_xu(0,t) &= \mu_1(t) \quad t \in [0,T], \\ -k(t)D_xu(\ell,t) &= \mu_2(t) \quad t \in [0,T]. \end{cases} \tag{14}$$

Then, one of the equations (14) is solved for k and, similar to the previous approach, is used to approximate $u(x,t)$. The above approach can be used when approximating $k(t)$, and $u(x,t)$. In the next section, we test this methodology with three test cases: test cases 1, 2, and 3.

However, we can extend the above method further if one wishes to approximate $c(t)$ in addition to $k(t)$, and $u(x,t)$.

When approximating two parameters (in addition to $u(x,t)$), we can still solve one of the equations (14) and find k^i where it is $k(t_i)$. Then we can rewrite equation (11) at the top or bottom boundary as:

$$c^i \frac{u(\ell)^{i+1} - u(\ell)^i}{\Delta t} = k^i \left(D_{xx}u^i\right)_{x=\ell} + f(\ell)^i. \tag{15}$$

Then equation (15) can be solved for c^i as we already approximated k^i. Then we can use them to approximate $u(x,t)$ using equation (9). Test cases 4, 5 and 6 focus on using this approach to simultaneously approximate $k(t)$, $c(t)$, and $u(x,t)$. The RBF–FD approach provides an estimated order of convergence of $\mathcal{O}(h^{k-l+1})$, where l is the order of the spatial operator and h is the average node spacing [7]. This notable convergence rate is facilitated by the inclusion of augmented polynomials, which effectively minimize stagnation error [13, 14, 15].

3. **Numerical results**

The numerical results presented in this section contain convergence analysis and performance analysis when noise is added to the overspecified boundary conditions. Overall, six test cases were analyzed using the proposed methodology. The first three cases show high-order convergence despite simultaneously approximating the time-dependent coefficient and the solution of the partial differential equations. Both Dirichlet and Neumann-type overspecified conditions were tested with the test cases. However, we only approximate $k(t)$, and $u(x,t)$ in the first three test cases. The number of nodes and polynomial degrees varied from 20 to 160 and 3 to 5, respectively. In the final three test cases, we simultaneously approximate $k(t)$, $c(t)$, and $u(x,t)$. Furthermore, different noise levels were added to the measured data to see how they impacted performance. Also, the results were compared with [4]. Test case 5 is selected because the partial differential equation in the test case may not have a unique solution [4]. The final test case contains nonsmooth parameters. For the first three test cases, relative L_∞ error is calculated for standard convergence analysis when $h \to 0$, where h is the fill distance. However, for the last three test cases, both relative L_∞ error and the root mean squared error were calculated to compare results with other work. This section has two main subsections: the first focuses on problems that approximate $k(t)$ and $u(x,t)$, and the second subsection focuses on approximating all three unknowns simultaneously.

3.1. **Approximating only** $k(t)$. **and** $u(x,t)$.

3.1.1. **Test case 1.**

The first test case contains a Dirichlet-type overspecified condition:

$$\begin{cases} \dfrac{\partial u}{\partial t}(x,t) & = k(t)\dfrac{\partial^2 u}{\partial x^2}(x,t), \quad (x,t) \in \Omega, \\ u(x,0) & = \sin(\pi x), \qquad\qquad x \in (0,1). \end{cases} \tag{16}$$

where $\Omega = [0,1] \times [0,1]$. Furthermore, the equation (16) is subjected to boundary conditions:

$$\begin{cases} u(0,t) & = 0, & t \in [0,T], \\ u(1,t) & = 0, & t \in [0,T], \\ u(x^*,t) & = \sin(\pi x^*)e^{\frac{-\pi^2 t^2}{2}}, & t \in [0,T], \; x^* \in (0,1). \end{cases}$$

The true solution is $u(x,t) = \sin(\pi x)e^{\frac{-\pi^2 t^2}{2}}$, and $k(t) = t$.

Figure 1 shows $p+1$ rate of convergence for approximated $u(x,t)$ where p is the degree of the augmented polynomial. The methodology achieves convergence rates for $u(x,t)$ between four and seven. The convergence rate for $k(t)$ is between one and two. Increasing the degree of the augmented polynomial has a minimal effect on approximating $k(t)$ but is certainly beneficial for $u(x,t)$.

3.1.2. **Test case 2.**

$$\begin{cases} \dfrac{\partial u}{\partial t}(x,t) & = k(t)\dfrac{\partial^2 u}{\partial x^2}(x,t), \quad (x,t) \in \Omega, \\ u(x,0) & = e\cos(x), \qquad\qquad x \in (0,1), \end{cases} \tag{17}$$

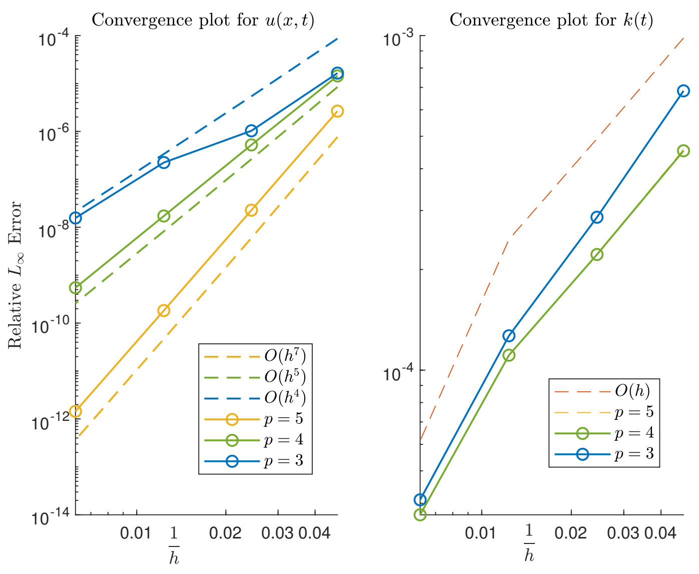

FIGURE 1. Test Case 1: Error decay for both approximated $u(x,t)$ (left), and $k(t)$ (right) as $h \to 0$, where h is the average distance between nodes. Overall, we observe $p+1$ rate of convergence or higher for approximated $u(x,t)$, where p is the degree of the augmented polynomial. The rate of convergence for $k(t)$ is between one and two.

where $\Omega = [0,1] \times [0,1]$. Furthermore, the equation (17) is subjected to boundary conditions:

$$\begin{cases} u(0,t) &= e^{\cos^3(t)-\frac{t}{2}}, & t \in [0,T], \\ u(1,t) &= \cos(1)e^{\cos^3(t)-\frac{t}{2}}, & t \in [0,T], \\ u(x^*,t) &= \cos(x^*)e^{\cos^3(t)-\frac{t}{2}}, & t \in [0,T], \;\; x^* \in (0,1). \end{cases}$$

The true solution is:

$$u(x,t) = \cos(x)e^{\cos^3(t)-\frac{t}{2}}, \text{ and } k(t) = 3\cos^2(t)\sin(t) + 0.5.$$

Figure 2 shows $p+1$ rate of convergence for approximated $u(x,t)$, where p is the degree of the augmented polynomial. The convergence rates are as low as four and as high as six. The convergence rate for $k(t)$ is between two and four.

In both test cases 1 and 2, when approximating $u(x,t)$, we observe higher order convergence consistent with the literature [7]. The higher the

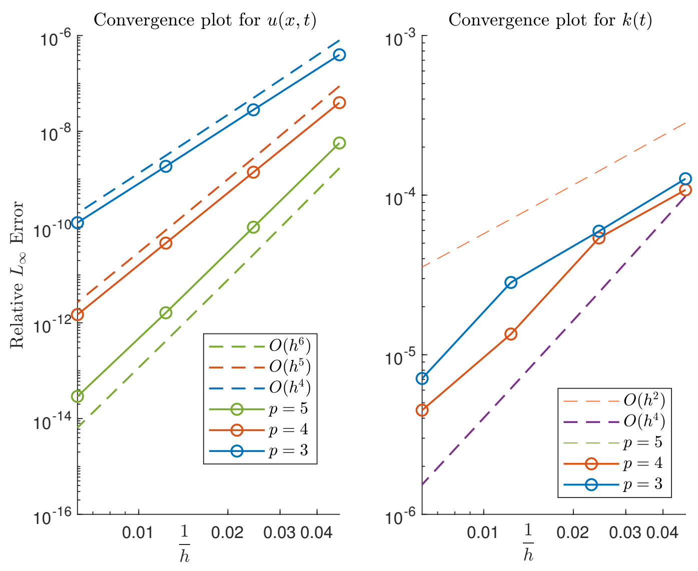

FIGURE 2. Test Case 2: Error decay for both approximated $u(x,t)$ (left) and $k(t)$ (right) as $h \to 0$, where h is the average distance between nodes. Overall, we observe $p+1$ rate of convergence for approximated $u(x,t)$, where p is the degree of the augmented polynomial. The rate of convergence for $k(t)$ is between two and four.

degree of the augmented polynomial, the higher the order of convergence. However, as noticed in both test cases, the degree of the polynomial does not affect approximating $k(t)$, which is reasonable given that the heat conductivity coefficient only depends on time.

3.1.3. **Test case 3.** Test case 3 [16] has a Neumann-type overspecified boundary condition:

$$\begin{cases} \dfrac{\partial u}{\partial t}(x,t) &= k(t)b(x)\dfrac{\partial^2 u}{\partial x^2}(x,t) + f(x,t), \quad (x,t) \in \Omega, \\ u(x,0) &= x + \sin(x), \qquad\qquad\qquad\qquad x \in (0,1). \end{cases} \tag{18}$$

where $b(x) = 2 - x^2$, and $\Omega = [0,1] \times [0,1]$. Furthermore, the equation (18) is subjected to boundary conditions:

$$\begin{cases} u(0,t) &= 8t, & t \in [0,T], \\ u(1,t) &= 1 + \sin(1) + 8t, & t \in [0,T], \\ -k(t)u_x(0,t) &= -2 - 2t, & t \in [0,T]. \end{cases}$$

The true solution is $u(x,t) = x + \sin(x) + 8t$, and $k(t) = 1 + t$ [16].

Similar to previous test cases, Figure 3 shows $p+1$ rate of convergence for approximated $u(x,t)$, where p is the degree of the augmented polynomial. The methodology achieves convergence rates as low as four and as high as six. The convergence rate for $k(t)$ is between two and four. Overall, the above three test cases show that the proposed RBF-FD methodology can obtain higher order convergence for approximating $u(x,t)$, and $k(t)$ in both Dirichlet and Neumann-type overdetermined boundary conditions.

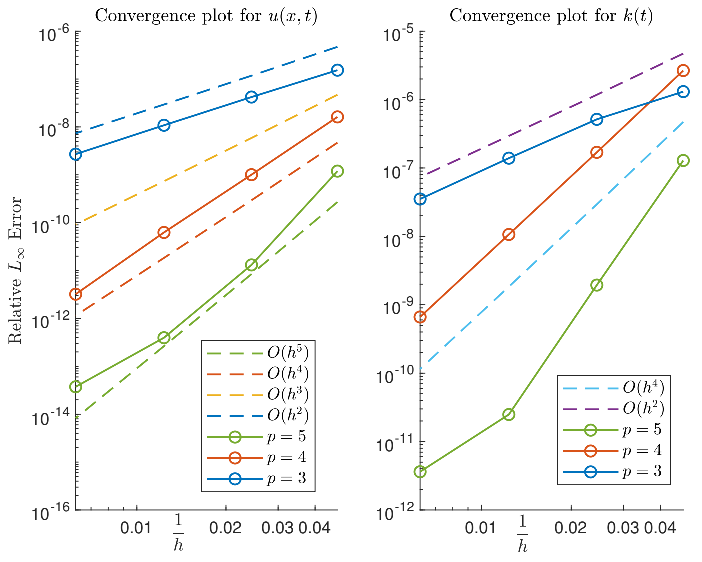

FIGURE 3. Test Case 3: Error decay for both approximated $u(x,t)$, and $k(t)$ as $h \to 0$, where h is the average distance between nodes. Overall, we observe $p+1$ rate of convergence for approximated $u(x,t)$, where p is the degree of the augmented polynomial. The rate of convergence for $k(t)$ is between two and four.

3.2. **Approximating two time-dependent parameters.**

In this subsection we approximate $k(t)$, $c(t)$, and $u(x,t)$. Furthermore, we look at how adding noise to data impacts the methodology for the next three cases.

In order to add random noise to the data, we used the same methodology outlined in [4]. That is, noise-added data is defined as:

$$\mu_1^{\eta_1}(t) = \mu_1(t) + \epsilon_1,$$

ϵ_1 is random Gaussian noise with percentage of noise n_p, mean zero and standard deviation σ defined as:

$$\sigma = n_p \max_{t\in[0,T]} |\mu_1(t)|.$$

Furthermore, for test cases 4, 5, and 6, we used root mean square error (rrmse) to analyze the error of $k(t)$ for a consistent comparison with [4].

3.2.1. **Test case 4.** This test case [4] looks at another partial differential equation with Neumann-type overspecified boundary condition. However, we look at how the error behaves when noise is added to the measured data (20):

$$\begin{cases} c(t)\dfrac{\partial u}{\partial t}(x,t) &= k(t)\dfrac{\partial^2 u}{\partial x^2}(x,t), \quad (x,t)\in\Omega, \\ u(x,0) &= (1+x)^2, \quad x\in(0,\ell), \end{cases} \tag{19}$$

where $\Omega = [0,1]\times[0,1]$. Furthermore, the equation (19) is subjected to boundary conditions:

$$\begin{cases} u(0,t) &= t^2+t, \quad t\in[0,T], \\ u(1,t) &= t^2+t+4, \quad t\in[0,T], \end{cases}$$

$$\begin{cases} -k(t)u_x(0,t) &= -(1+t)(1+2t), \quad t\in[0,T], \\ -k(t)u_x(1,t) &= 2(1+t)(1+2t), \quad t\in[0,T]. \end{cases} \tag{20}$$

The true solution is $u(x,t) = (1+x)^2 + t^2 + t$, $k(t) = (1+t)(t+0.5)$, and $c(t) = (1+t)$ [4]. Since the exact solution is degree 2 in space, we used only degree 3 augmented polynomials in RBF-FD approximation.

Table 1 contains relative L_∞ and rrmse errors of $u(x,t)$, $k(t)$, and $c(t)$ respectively. The rows of Table 1 present different noise levels starting from noise to 10% noise.

It is evident from the table that there is a rapid loss of accuracy of $k(t)$ and $c(t)$ when noise is introduced to data. However, $u(x,t)$ only increases the error minimally compared to $k(t)$. Figures 4 and 5 show the difference between actual and approximated $k(t)$, and $c(t)$ solutions at different noise levels. The rrmse for $k(t)$, and $c(t)$ is better by a couple of orders of magnitudes compared to [4].

TABLE 1. Test Case 4: Errors of approximated $u(x,t)$,$k(t)$, and $c(t)$ at different noise levels.

Noise level	Errors		
	$u(x,t)$ using L_∞	$k(t)$ using rrmse	$c(t)$ using rrmse
0%	5.66E-12	7.67E-22	3.33E-09
1%	5.72E-12	7.61E-04	7.61E-04
10%	6.84E-12	7.76E-02	7.75E-02

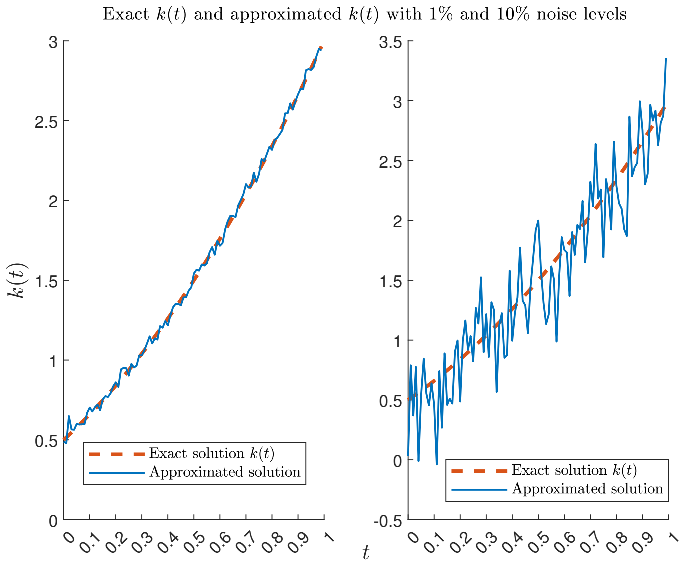

FIGURE 4. Test Case 4: Exact $k(t)$, and approximated $k(t)$ at different noise levels. The left plot has 1% noise, and the right plot has 10% noise.

3.2.2. **Test case 5.** Test case 5 is also from [4], but the solutions of equation (21) may not be unique:

$$\begin{cases} c(t)\dfrac{\partial u}{\partial t}(x,t) &= k(t)\dfrac{\partial^2 u}{\partial x^2}(x,t), \quad (x,t)\in\Omega, \\ u(x,0) &= \dfrac{x^4}{12}+2x-4, \quad x\in(0,\ell), \end{cases} \tag{21}$$

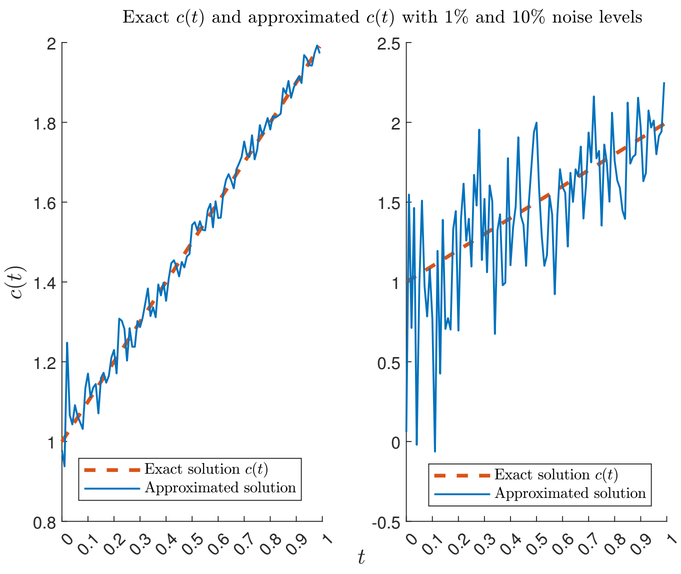

FIGURE 5. Test Case 4: Exact $c(t)$, and approximated $c(t)$ at different noise levels. The left plot has 1% noise, and the right plot has 10% noise.

where $\Omega = [0,1] \times [0,1]$. Furthermore, equation (21) is subjected to boundary conditions:

$$\begin{cases} u(0,t) &= t^4 + 2t^3 + t^2 - 4, & t \in [0,T], \\ u(1,t) &= t^4 + 2t^3 + 2t^2 + t - \frac{23}{12}, & t \in [0,T]. \end{cases}$$

$$\begin{cases} -k(t)u_x(0,t) &= -2t - 2, & t \in [0,T], \\ -k(t)u_x(1,t) &= (t+1)\left(2t^2 + 2t + \frac{7}{3}\right), & t \in [0,T]. \end{cases}$$

The true solution is $u(x,t) = t^4 + 2t^3 + t^2\left(x^2+1\right) + tx^2 + \dfrac{x^4}{12} + 2x - 4$, $k(t) = (1+t)$, and $c(t) = \dfrac{1+t}{1+2t}$.

Table 2 contains a similar analysis to test problem 04 but with degree 4 augmented polynomials in RBF-FD approximation. The rapid loss of accuracy for $k(t)$, and $c(t)$ is again visible in Table 2, but the errors for $u(x,t)$ do not change significantly (there is a relative difference as small as six digits). Figures 6 and 7 show the difference between true and approximated $k(t)$, and $c(t)$ solutions at different noise levels. When compared with results in [4] for 1% noise level, there are about three orders of magnitude improvements in the proposed RBF-FD methodology.

TABLE 2. Test Case 5: Errors of approximated $u(x,t)$,$k(t)$, and $c(t)$ at different noise levels.

Noise level	Errors		
	$u(x,t)$ using L_∞	$k(t)$ using rrmse	$c(t)$ using rrmse
0%	3.71E-06	4.15E-10	1.33E-09
1%	3.71E-06	1.98E-04	1.98E-04
10%	3.71E-12	2.01E-02	2.01E-02

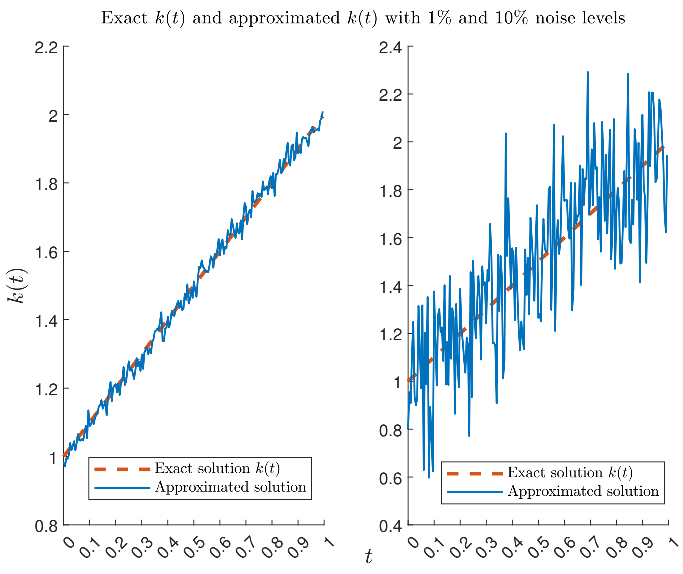

FIGURE 6. Test Case 5: Exact $k(t)$, and approximated $k(t)$ at different noise levels. The left plot has 1% noise, and the right plot has 10% noise.

3.2.3. **Test case 6.** A partial differential equation with non-smooth input data and parameters is chosen as the final test case. This test case is also chosen from [4]:

$$\begin{cases} c(t)\dfrac{\partial u}{\partial t}(x,t) & = k(t)\dfrac{\partial^2 u}{\partial x^2}(x,t), \quad (x,t) \in \Omega, \\ u(x,0) & = \dfrac{x^2+x}{2} - \dfrac{1}{4}, \qquad x \in (0,\ell), \end{cases} \tag{22}$$

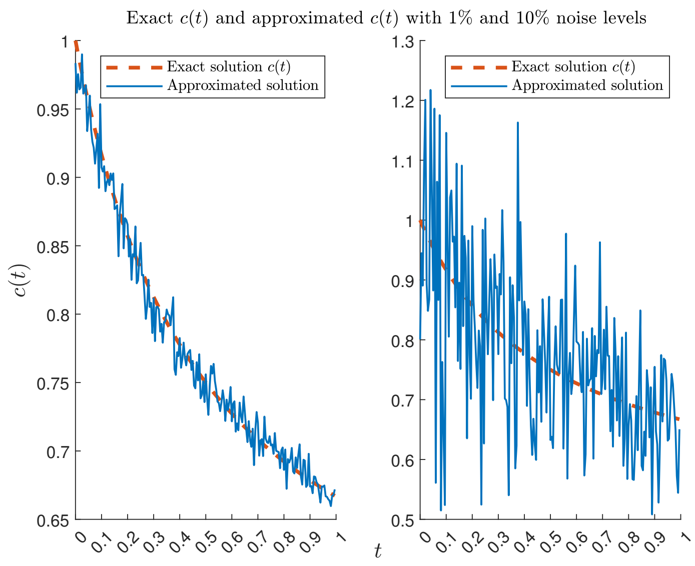

FIGURE 7. Test Case 5: Exact $c(t)$, and approximated $c(t)$ at different noise levels. The left plot has 1% noise, and the right plot has 10% noise.

where $\Omega = [0,1] \times [0,1]$. Furthermore, the equation (22) is subjected to boundary conditions:

$$\begin{cases} u(0,t) &= \begin{cases} \frac{3t-t^2}{2} - \frac{1}{4}, & \text{if } t \in \left[0, \frac{1}{2}\right], \\ \frac{t+t^2}{2}, & \text{if } t \in \left[\frac{1}{2}, 1\right], \end{cases} \quad t \in [0,T], \\ u(1,t) &= \begin{cases} \frac{3t-t^2}{2} + \frac{3}{4}, & \text{if } t \in \left[0, \frac{1}{2}\right], \\ \frac{t+t^2}{2} + 1, & \text{if } t \in \left[\frac{1}{2}, 1\right], \end{cases} \quad t \in [0,T]. \end{cases}$$

$$\begin{cases} -k(t)u_x(0,t) &= -\frac{1}{2}\left(1 + \left|t - \frac{1}{2}\right|\right), \quad t \in [0,T], \\ -k(t)u_x(1,t) &= \frac{3}{2}\left(1 + \left|t - \frac{1}{2}\right|\right), \quad t \in [0,T]. \end{cases}$$

The true solutions are:

$$u(x,t) = \frac{x+x^2}{2} + \begin{cases} \frac{3t-t^2}{2} - \frac{1}{4}, & \text{if } t \in \left[0, \frac{1}{2}\right], \\ \frac{t+t^2}{2}, & \text{if } t \in \left[\frac{1}{2}, 1\right], \end{cases}$$

and

$$k(t) = 1 + \left|t - \frac{1}{2}\right|.$$

Here $c(t) = 1$. Table 3 shows similar behavior to the previous test cases. Despite the rapid error increase in $k(t)$ and $c(t)$, the relative L_∞ error of $u(x,t)$ is not impacted by a considerable margin. The results are excellent for the given noise levels. When compared with work in [4] again, the RBF-FD methodology displays superior performance. Figures 8 and 9 illustrate the difference between true and approximated $k(t)$ solutions at different noise levels.

Table 3. Test Case 6 - Errors of approximated $u(x,t)$,$k(t)$, and $c(t)$ at different noise levels.

Noise level	Errors		
	$u(x,t)$ using L_∞	$k(t)$ using rrmse	$c(t)$ using rrmse
0%	2.17E-08	4.91E-14	4.03E-10
1%	2.17E-08	1.49E-04	1.49E-04
10%	2.17E-08	1.50E-02	1.50E-02

Overall, we notice similar behavior in all three test cases with added noise. Even though root mean squared error doubled when noise was added, there was only an order of magnitude increase in relative L_∞ error in approximated $u(x,t)$.

The future work for this study includes topics such as modifying the method to work with implicit time solvers and parallel time integrators. One drawback of the RBF-FD method is the differentiation matrix produces spurious eigenvalues that may lead to numerical instability or smaller time steps [7, 18, 19]. As an alternative, parallel time integrators such as the Parareal method have been shown to work well with the RBF-FD method by mitigating the effects of spurious eigenvalues and speeding up computations significantly [17]. Therefore, it is interesting to incorporate Parareal-RBF-FD into this methodology in the future and tackle two-dimensional problems.

4. Conclusion

In this manuscript, we proposed a new RBF-FD methodology to approximate both the solutions and the time-dependent heat coefficients for parabolic partial differential equations. The new methodology was tested with six partial differential equations, and higher-order convergence was obtained when approximating the solutions of the equations. Convergence rates as high as seven were observed. Furthermore, the

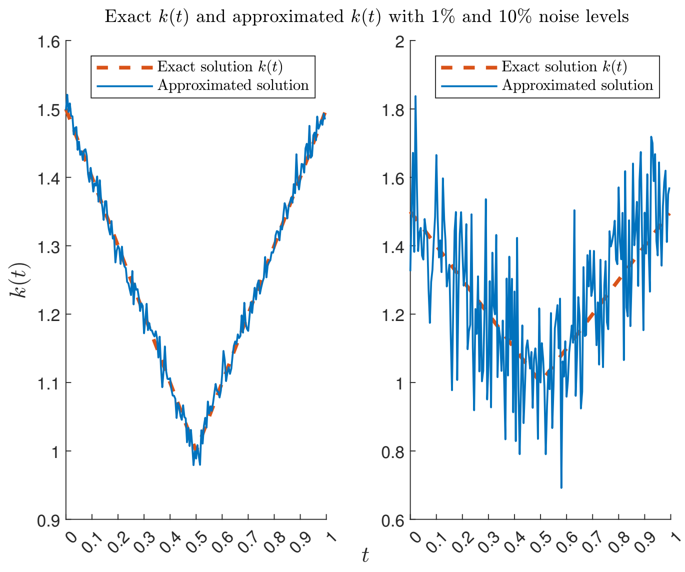

Figure 8. Test Case 6: Exact $k(t)$, and approximated $k(t)$ at different noise levels. The left plot has 1% noise, and the right plot has 10% noise.

convergence rates for approximating the time-dependent parameter were limited between two and four. Finally, three different test cases were used with 1% and 10% noise in overspecified boundary conditions. The root mean squared error for approximating the time-dependent parameter displayed roughly doubled error. However, despite the noise levels, the methodology produced excellent results for approximating the solution with only negligible differences in error.

References

[1] D. Lesnic, *Inverse Problems with Applications in Science and Engineering* (1st ed.), Chapman and Hall/CRC (2021).

[2] M.H. Ding, H. Liu, G.H. Zheng, On inverse problems for several coupled PDE systems arising in mathematical biology, *J. Math. Biol.* **87** (2023), # 86.

[3] J. Gottlieb, P. DuChateau, *Parameter Identification and Inverse Problems in Hydrology, Geology and Ecology*, Water Science and Technology Library, Springer, Dordrecht (2011).

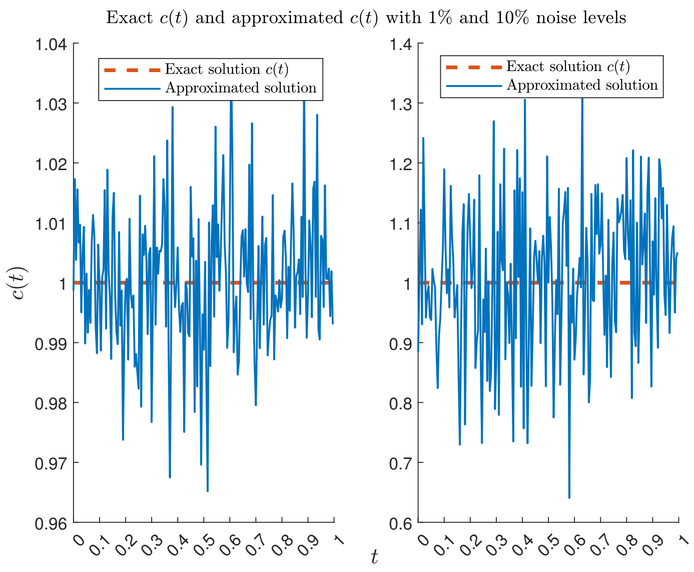

FIGURE 9. Test Case 6: Exact $c(t)$, and approximated $c(t)$ at different noise levels. The left plot has 1% noise, and the right plot has 10% noise.

[4] M.S. Hussein, D. Lesnic, M.I. Ivanchov, Simultaneous determination of time-dependent coefficients in the heat equation, *Computers & Mathematics with Applications*, **67**, No 5 (2014), 1065-1091.

[5] Fu-le Li, Zi-ku Wu, Chao-rong Ye, A finite difference solution to a two-dimensional parabolic inverse problem, *Applied Mathematical Modelling*, **36**, Issue 5 (2012), 2303-2313.

[6] B. Harrach, An introduction to finite element methods for inverse coefficient problems in elliptic PDEs, *Jahresber. Dtsch. Math. Ver.* **123** (2021), 183-210.

[7] B. Fornberg, N. Flyer, *A Primer on Radial Basis Functions with Applications to the Geosciences*, Society for Industrial and Applied Mathematics, Philadelphia, PA (2015).

[8] C. Piret, N. Dissanayake, J. Gierke, and B. Fornberg, The radial basis functions method for improved numerical approximations of geological processes in heterogeneous systems, *Mathematical Geo-Sciences*, **52**‘(2019), #08.

[9] V. Bayona, N. Flyer, G.M. Lucas, and A.J.G. Baumgaertner, A 3-D RBF-FD solver for modeling the atmospheric global electric circuit

with topography (GEC-RBFFD v1.0), *Geoscientific Model Development*, **8**, No 10 (2015), 3007-3020.

[10] T.A. Driscoll and B. Fornberg, Interpolation in the limit of increasingly flat radial basis functions, *Computers & Mathematics with Applications*, **43** (2002), 413-422.

[11] J.C. Mairhuber, On Haar's theorem concerning chebychev approximation problems having unique solutions, *Proc. of the American Mathematical Society*, **7** (1956), 609-615).

[12] N. Flyer, B. Fornberg, V. Bayona, G. Barnett, On the role of polynomials in RBF-FD approximations: I, Interpolation and accuracy, *Journal of Computational Physics*, **321** (2016), 21-38.

[13] V. Bayona, An insight into RBF-FD approximations augmented with polynomials, *Computers and Methematics with Applications*, **77** (2019), 2337-2353.

[14] V. Bayona, N. Flyer, B. Fornberg, G. Barnett, On the role of polynomials in RBF-FD approximations: II, Numerical solution of elliptic PDEs, *Journal of Computational Physics*, **322** (2017), 257-273.

[15] V. Bayona, N. Flyer, B. Fornberg, On the role of polynomials in RBF-FD approximations: III, Behavior near domain boundaries, *Journal of Computational Physics*, **380** (2019), 378-399.

[16] M.S. Hussein, D. Lesnic, Determination of the time-dependent thermal conductivity in the heat equation with spacewise dependent heat capacity, *Finite Difference Methods, Theory and Applications, FDM 2014*, Lecture Notes in Computer Science, **9045** (2015).

[17] N.D.K. Mudiyanselage, J. Blazejewski, B. Ong, C. Piret, A Radial Basis Function - Finite Difference and Parareal Framework for Solving Time Dependent Partial Differential Equations, *Dolomites Research Notes on Approximation*,**15**, No 5 (2022), 8-23.

[18] R.B. Platte and T.A. Driscoll, Eigenvalue stability of radial basis function discretizations for time-dependent problems, *Computers & Mathematics with Application*, **51** (2006), 1251-1268.

[19] V. Shankar, G. Wright, and A. Narayan, A robust hyperviscosity formulation for stable RBF-FD discretizations of advection-diffusion-reaction equations on manifolds, *SIAM Journal on Scientific Computing*, **42**, No 4 (2020), A2371-A2401.