Research Paper
Computational Methods for Differential Equations
http://cmde.tabrizu.ac.ir
Vol. *, No. *, *, pp. 1-14
DOI:10.22034/cmde.2025.65004.2967

RBF partition of unity methods for solving the Poisson equation on irregular domains

Fariba Fathi Dopolani and Mohammadreza Ahmadi Darani*
Department of Applied Mathematics, Faculty of Mathematical Sciences, Shahrekord University , P.O. Box. 115, Shahrekord, Iran.

Abstract

This paper presents a novel approach for solving the Poisson equation on arbitrary domains using a direct Radial Basis Function (DRBF) partition of unity technique. The method involves dividing the primary domain into overlapping subdomains, calculating local approximations within each subdomain, and then combining these approximations through discontinuous weight functions to form a global solution. We also use polyharmonic spline (PHS) kernels, with scaling properties. This strategy improves stability, lowers computational costs, and replaces a single ill-conditioned linear system with several smaller, well-conditioned linear systems. Numerical experiments are performed to confirm the efficacy of the proposed method.

Keywords. Radial basis function (RBF) Methods, Partition of unity (PU), Direct radial basis function partition of unity (D-RBF-PU)..
2010 Mathematics Subject Classification. 65L05, 34K06, 34K28.

1. Introduction

Methods based on radial basis functions (RBFs) are a rapidly growing area of research and have become a common tool for solving partial differential equations (PDEs) [**? ? ?**]. Several valuable properties–such as easy implementation, applicability to scattered data, flexibility with respect to geometry and dimension, and high-order convergence rates–make them particularly advantageous for higher-dimensional problems [7, 19]. However, despite their applicability, especially in high-dimensional spaces, one of the main disadvantages of RBFs is the ill-conditioned system generated by the interpolation conditions. Localized methods have been introduced to address this issue, providing the benefits of a sparse interpolation matrix and a well-conditioned final system, which results in greater stability and lower computational costs.

An interesting example of localized methods is the partition of unity (PU) method, a type of meshless approach that enables fast computation. This method was first introduced by Shepard in 1968 [15]. This method decomposes a large problem into many smaller problems. Combining popular numerical methods with the PU method yields various approaches that are widely used in computational research. Historically, Babuška and Melenk introduced the partition of unity finite element method (PUFEM) for solving PDEs [2, 10].

Wendland combined the PU method with the RBF method for scattered data problems [18]. Various RBF-PU methods exist [9], including a form based on collocation (C-RBF-PU) for PDE problems introduced by Larsson and Heryudono [14]. Another form, the compact radial basis function partition of unity (CRBF-PU) method, integrates standard Hermite interpolation, RBF-FD, and the PU method [1]. Mirzaei proposed a new direct RBF-PU (D-RBF-PU) scheme that omits PU weight derivatives [12], demonstrating its effectiveness in solving boundary value problems and showing it to be faster than traditional RBF-PU methods. The applications of RBF-PU methods for solving PDEs are extensively documented in references [3, 4, 6, 8, 11, 13].

In this paper, we want to obtain a numerical solution using the D-RBF-PU method for solving the Poisson equation with pure Dirichlet boundary conditions. The structure of this article is organized as follows. Section 2 defines the notations that will be used during the article. Following this, Section 3 offers a detailed description of the D-RBF-PU

Received: 14 December 2024 ; Accepted: 23 September 2025.
∗ Corresponding author. Email: ahmadi.darani@sku.ac.ir.

method, such as its formulation, implementation, and some theoretical subjects. In Section 4, the application of the D-RBF-PU method is demonstrated for the numerical solution of the Poisson equation, by a presentation of numerical investigations that shows the effectiveness and accuracy of the method. Finally, Section 5 includes the conclusions drawn from the study, summarizing the main findings and discussing the potential implications and future research in this area.

2. Notations

In order to describe the proposed method, we define some notations that we use them in whole of this article. For the given positive integer d, $\mathbb{R}^d$ denotes the d dimensional Euclidian space of real numbers and Ω is an open bounded domain in $\mathbb{R}^d$. In general the bold lower-case letters such as $\boldsymbol{x}, \boldsymbol{y}$, and the bold lower-case Greek letter such as $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{N}^d \cup \{0\}$ are used for vectors in $\mathbb{R}^d$ and multi-indices, respectively. For example

$$\boldsymbol{x} = [x_1, \cdots, x_d]^T, \quad \boldsymbol{x} \in \mathbb{R}^d,$$
$$\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_d]^T, \quad \boldsymbol{\alpha} \in \mathbb{N}^d.$$

. The inequality $\boldsymbol{\alpha} \leq \boldsymbol{\beta}$ means that $\alpha_j \leq \beta_j$, for $j = 1, \cdots d$. For an arbitrary set $X \subseteq \Omega$ with N distinct points $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_M$, we define two geometric quantities which express quality of point distribution.

- The fill distance quantity which is the radius of largest empty ball inside Ω and is defined by

  $$h = h_{X,\Omega} = \max_{\boldsymbol{x} \in \Omega} \min_{\boldsymbol{x}_i \in X} \|\boldsymbol{x} - \boldsymbol{x}_i\|.$$

- The separation of X which is defined by

  $$q_{X,\Omega} = \frac{1}{2} \min_{j \neq k} |\boldsymbol{x}_j - \boldsymbol{x}_k|_2.$$

For the multi-index $\boldsymbol{\alpha}$, and $\boldsymbol{x} \in \mathbb{R}^d$, we define $\boldsymbol{\alpha}! = \alpha_1! \times \cdots \times \alpha_d!$, $|\boldsymbol{\alpha}| = \alpha_1 + \cdots + \alpha_d$, and $\boldsymbol{x}^{\boldsymbol{\alpha}} = x_1^{\alpha_1}, \cdots, x_d^{\alpha_d}$. $\Pi_m(\mathbb{R}^d)$ is the Q dimensional space of multi-variable polynomials of degree less than or equal to m with the classical basis function $\{p_i(x)\}_1^Q$, where each element $q(\boldsymbol{x}) \in \{p_1(\boldsymbol{x}), \cdots, p_Q(\boldsymbol{x})\}$ is a monomial with the following form

$$q(\boldsymbol{x}) = \boldsymbol{x}^{\boldsymbol{\alpha}} = x_1^{\alpha_1} \cdots x_d^{\alpha_d}, \quad |\boldsymbol{\alpha}| = m.$$

We define the partial derivative operator $\mathcal{D}^{\boldsymbol{\alpha}}$ for multi-indices $\boldsymbol{\alpha}$ as

$$\mathcal{D}^{\boldsymbol{\alpha}} = \frac{\partial^{\alpha_1}}{\partial x_1^{\alpha_1}} \cdot \ldots \cdot \frac{\partial^{\alpha_d}}{\partial x_d^{\alpha_d}}.$$

3. Some preliminaries for D-RBF-PU method

D-RBF-PU method, which is proposed in [12] for the first time, is a localized RBF method based on PU and has used for solving boundary and initial-boundary value problems. This method benefits from a direct discretization approach and is more faster than standard PU method because it is avoiding all derivatives of PU weight functions.

3.1. The interpolation using Radial Basis Functions (RBFs).

In this section, we interpolate the function u at scattered data points using conditionally positive definite radial basis functions.

Definition 3.1. The function $\psi : \mathbb{R}^d \longrightarrow \mathbb{R}$ is called conditionally positive definite of order m if

$$\sum_{i=1}^{N} \sum_{j=1}^{M} b_i b_j \psi(\boldsymbol{x}_i - \boldsymbol{x}_j) > 0,$$

for any pairwise distinct nodes $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N \in \mathbb{R}^d$ for all vectors $\boldsymbol{b} = (b_1, \cdots, b_N)^T \in \mathbb{R}^N$ satisfying

$$\sum_{j=1}^{N} b_j p_\ell(\boldsymbol{x}_j) = 0, \quad \ell = 1, \cdots, Q,$$

$$\begin{aligned} u(\boldsymbol{x}) \approx \tilde{u}(\boldsymbol{x}) &= \sum_{i=1}^{N} c_i \psi(\boldsymbol{x}-\boldsymbol{x}_i) + \sum_{\ell=1}^{Q} c_{N+\ell}\, p_\ell(\boldsymbol{x}) \\ &= \Psi(\boldsymbol{x})^T \boldsymbol{c}_1 + \boldsymbol{p}(\boldsymbol{x})^T \boldsymbol{c}_2, \end{aligned} \tag{3.1}$$

where $\boldsymbol{c}_1 = [c_1, \cdots, c_N]^T$, $\boldsymbol{c}_2 = [c_{N+1}, \cdots, c_{N+Q}]^T$, $\Psi(\boldsymbol{x}) = [\psi(\boldsymbol{x}-\boldsymbol{x}_1), \cdots, \psi(\boldsymbol{x}-\boldsymbol{x}_N)]^T \in \mathbb{R}^N$, and $\boldsymbol{p}(\boldsymbol{x}) = [p_1(\boldsymbol{x}), \cdots, p_Q(\boldsymbol{x})]^T$, which $p_1, \cdots, p_Q$ are the bases for the Q-dimensional space $\Pi_m(\mathbb{R}^d)$. The vectors $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ are obtained by interpolation conditions $\tilde{u}(\boldsymbol{x}_k) = u(\boldsymbol{x}_k), \quad k = 1, \cdots, N$, along with Q additional constraints

$$\sum_{k=1}^{N} c_k p_\ell(\boldsymbol{x}_k) = 0, \quad \ell = 1, \cdots, Q.$$

These computations lead to the following system of linear equations:

$$\begin{aligned} &\mathbf{R}\boldsymbol{c}_1 + \boldsymbol{p}\boldsymbol{c}_2 = \mathbf{U}|_X, \\ &\boldsymbol{p}^T \boldsymbol{c}_1 = 0, \end{aligned} \tag{3.2}$$

where $\mathbf{U}|_X = [u(\boldsymbol{x}_1), \cdots, u(\boldsymbol{x}_N)]^T$, $\boldsymbol{p} = \boldsymbol{p}^T(X) \in \mathbb{R}^{N\times Q}$ and $\mathbf{R} = \boldsymbol{\psi}^T(X) \in \mathbb{R}^{N\times N}$. If ψ is a positively definite conditional function of order m and the set X consists of unique solvers in $\Pi_m(\mathbb{R}^d)$, then the Equation (3.2) is uniquely solvable. Moving forward, we express the interpolation operator $\tilde{u}$ in Lagrangian form, which plays a pivotal role in implementing the $RBF - PU$ method. Let's assume $e^{(j)} \in \mathbb{R}^N$ denote the j -th unit vector. Assuming that the set $X \subseteq \Omega$ is $\Pi_m(\mathbb{R}^d)$ unisolvent, the following linear system is uniquely solvable:

$$\begin{bmatrix} \mathbf{R} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{O} \end{bmatrix} \begin{bmatrix} \boldsymbol{c}_1^{(j)} \\ \boldsymbol{c}_2^{(j)} \end{bmatrix} = \begin{bmatrix} e^{(j)} \\ 0 \end{bmatrix},$$

where $\mathbf{O}$ denotes the $Q \times Q$ zero matrix. Moreover, Lagrange functions ϑ_j^* exist and are given by:

$$\vartheta_j^* = \sum_{j=1}^{N} c_i^{(j)} \psi(\cdot - \boldsymbol{x}) + \sum_{\ell=1}^{Q} c_{N+\ell}^{(j)} p_\ell(\boldsymbol{x}),$$

Moreover, the Lagrange functions ϑ_j^* satisfy the property $\vartheta_j^*(\boldsymbol{x}_i) = \delta_{ij}$ and belong to the subspace:

$$v_j := \left\{ \sum_{j=1}^{N} a_j \psi(\boldsymbol{x}-\boldsymbol{x}_j), \quad \sum_{j=1}^{N} a_i p_l(\boldsymbol{x}_j), \quad l = 1, \cdots, Q, \quad p_l \in \Pi_m(\mathbb{R}^d) \right\} \oplus \Pi_m(\mathbb{R}^d).$$

Theorem 3.2. *[19] Let ψ be a positive definite kernel with respect to $\Pi_m(\mathbb{R}^d)$. If $X \subseteq \Omega$ be a $\Pi_m(\mathbb{R}^d)$ unisolvent, then functions $\vartheta_j^* \in v_j$ exist such that $\vartheta_j^*(\boldsymbol{x}_k) = \delta_{jk}$. Additionally, functions $\nu_j^*, \quad j = 1, 2, \cdots, Q$ exist such that*

$$\begin{bmatrix} \mathbf{R} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{O} \end{bmatrix} \begin{bmatrix} \vartheta^*(\boldsymbol{x}) \\ \nu^*(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} \Psi(\boldsymbol{x}) \\ \boldsymbol{p}(\boldsymbol{x}) \end{bmatrix}. \tag{3.3}$$

Then the function u can be interpolated as follows:

$$\tilde{u}(\boldsymbol{x}) = \sum_{j=1}^{N} \vartheta_j^*(\boldsymbol{x}) u(\boldsymbol{x}_j).$$

To approximate the function $\mathcal{D}^\alpha u$ using the kernel $\psi \in C^{2k}(\Omega \times \Omega)$, analogous to how we interpolated u, assume that $X \subseteq \Omega$, implying $u_{loc}(\boldsymbol{x}) \in C^k(\Omega)$, where $\tilde{u}(\boldsymbol{x})$ is the interpolation of u using ψ. Define the derivatives:

$$\begin{aligned} \mathcal{D}^\alpha \Psi(\boldsymbol{x}) &= [\mathcal{D}_1^\alpha \psi(\boldsymbol{x}), \cdots, \mathcal{D}_d^\alpha \psi(\boldsymbol{x})], \\ \mathcal{D}^\alpha \boldsymbol{p}(\boldsymbol{x}) &= [\mathcal{D}_1^\alpha p_1(\boldsymbol{x}), \cdots, \mathcal{D}_d^\alpha p_Q(\boldsymbol{x})], \end{aligned}$$

where $\mathcal{D}_i^\alpha$ denotes the derivative with respect to the ith component of $\boldsymbol{x}$. The approximation of $\mathcal{D}^\alpha u$ using Lagrange functions is expressed as:

$$\mathcal{D}^\alpha u(\boldsymbol{x}) \approx \sum_{j=1}^{N} \mathcal{D}^\alpha \vartheta_j^*(\boldsymbol{x}) u(\boldsymbol{x}_j),$$

where $\mathcal{D}^\alpha \vartheta_j^*(\boldsymbol{x})$ are Lagrange basis functions satisfying

$$\begin{bmatrix} \mathbf{R} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{O} \end{bmatrix} \begin{bmatrix} \mathcal{D}^\alpha \vartheta^*(\boldsymbol{x}) \\ \mathcal{D}^\alpha \nu^*(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} \mathcal{D}^\alpha \Psi(\boldsymbol{x}) \\ \mathcal{D}^\alpha \boldsymbol{p}(\boldsymbol{x}) \end{bmatrix}.$$

3.2. **Partition of Unity (PU).** The Partition of Unity method is a mesh-free approach that enables rapid computations by replacing the global approximation with specific number of local approximations. This method decomposes a large problem into numerous smaller problems, while extending the accuracy of local approximations to a global approximation. It solves several interpolation problems and then combines them into a global approximation.

Definition 3.3. Let $\Omega \subseteq \mathbb{R}^d$ be an open and bounded region, and let $\{\omega_j\}_{j=1}^J$ constitute an open and bounded cover for Ω (meaning all Ω_j are open, bounded, and $\Omega \subseteq \cup_{j=1}^J \Omega_j$). We define:

$$\delta_j = \operatorname{diam}(\Omega_j) = \sup_{\boldsymbol{x},\boldsymbol{y}\in\Omega_j} |\boldsymbol{x}-\boldsymbol{y}|_2.$$

Definition 3.4. A family of non-negative functions $\{\omega_j\}_{j=1}^J$, where $\omega_j \in C^k(\mathbb{R}^d)$, constitutes a Partition of Unity with respect to the cover $\{\omega_j\}_{j=1}^J$ if it satisfies the following conditions:

(1) $\operatorname{supp}(\omega_j) \subseteq \Omega_j$
(2) On Ω, we have: $\sum_{j=1}^J \omega_j(\boldsymbol{x}) = 1, \quad for\ all\ \boldsymbol{x} \in \Omega,$
(3) For every multi-index $\boldsymbol{\alpha} \in \mathbb{N}_0^d$ such that $|\boldsymbol{\alpha}| \leq k$, , there exists a positive constant $C_\alpha > 0$ such that for each $j = 1, \cdots, J$, we have:

$$|\mathcal{D}^\alpha \omega_j|_{L_\infty(\Omega_j)} \leq \frac{C_\alpha}{\delta_j^{|\alpha|}}.$$

In [7], Shepard weights are used to construct Partition of Unity weight functions. If $\{\psi_j(\boldsymbol{x})\}_{j=1}^J$ are non-negative, non-zero, and compactly supported functions, then the Partition of Unity weight functions are defined as follows:

$$\omega_j(\boldsymbol{x}) = \frac{\psi_j(\boldsymbol{x})}{\sum_{j=1}^J \psi_j(\boldsymbol{x})}, \quad j = 1, \cdots, J. \tag{3.4}$$

In Equation (3.4), we can interchange the summation indices with the following set:

$$I(\boldsymbol{x}) = \{\ell \in \{1, 2, \cdots, J\} : \quad \boldsymbol{x} \in \Omega_\ell\}.$$

The function u within each subdomain is approximated using the local approximation $\tilde{u}_j \in v_j$. So, these approximations are combined using the weight functions ω_j forming a global approximation across the entire space Ω. This means

$$u_{\text{pu}}(\boldsymbol{x}) = \sum_{\ell \in I(\boldsymbol{x})} \tilde{u}_\ell(\boldsymbol{x}) \omega_\ell(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega. \tag{3.5}$$

Theorem 3.5. *[19] Let $\Omega \subseteq \mathbb{R}^d$ be a bounded and open region, and let $\{\Omega_j\}_{j=1}^J$ be an open and bounded cover of Ω. Suppose $\{\Omega_j\}_{j=1}^J$ constitutes a k-stable partition of unity. If $u \in C^k(\Omega)$ is a function approximated by $\tilde{u}_j \in v_j \subseteq C^k(\Omega_j)$, a local approximation space, such that u is approximated by u_{pu} within each region $\Omega_j \cap \Omega$, where*

$$|\mathcal{D}^\alpha u - \mathcal{D}^\alpha \tilde{u}_j|_{L_\infty(\Omega\cap\Omega_j)} \leq \varepsilon(\alpha),$$

Then, in relation to (3.5), the function $u_{\text{pu}} \in C^k(\Omega)$ satisfies the following for all multi-indices $|\alpha| \leq k$:

$$\left|(\mathcal{D}^\alpha u - \mathcal{D}^\alpha u_{\text{pu}})(\boldsymbol{x})\right| \leq \sum_{\ell \in I(\boldsymbol{x})} \sum_{\beta \leq \alpha} \binom{\alpha}{\beta} C_{\alpha-\beta} \delta_\ell^{|\beta|-|\alpha|} \varepsilon_\ell(\alpha), \quad \boldsymbol{x} \in \Omega. \tag{3.6}$$

Our objective is to utilize this method and its outcomes for the radial basis function (RBF) interpolation. Let $X_j = X \cap \Omega_j$, and let ψ be a positive definite function of order m. Then, the local approximation space is defined as follows:

$$v_j := \text{span}\{\psi(\boldsymbol{x} - \boldsymbol{x}_k), \quad \boldsymbol{x} \in X_j\} \oplus \Pi_m(\mathbb{R}^d).$$

Now, we apply this method to approximate the solution of the following partial differential equation (PDE):

$$\begin{aligned} &\mathcal{L}u = f, \quad \boldsymbol{x} \in \Omega, \\ &\mathcal{B}u = g, \quad \boldsymbol{x} \in \partial\Omega, \end{aligned} \tag{3.7}$$

where $\mathcal{L}$ denotes an arbitrary differential operator and $\mathcal{B}$ represents the boundary operators such as Dirichlet or Neumann boundary conditions. To obtain the numerical solution of the PDE, we apply these operators to the approximation u_{pu} as defined in (3.5). Therefore, we have:

$$\begin{aligned} \mathcal{L}u_{\text{pu}} &\approx \sum_{j=1}^{J} \mathcal{L}(\omega_j \tilde{u}_j), \\ \mathcal{B}u_{\text{pu}} &\approx \sum_{j=1}^{J} \mathcal{B}(\omega_j \tilde{u}_j), \end{aligned}$$

where $\tilde{u}_j$ and u_{pu} represent the local and global approximations on sub domains Ω_j and domain Ω, respectively, as introduced earlier. The operators $\mathcal{L}$ and $\mathcal{B}$ involve partial derivatives $\mathcal{D}^\alpha$, where $\alpha \in \mathbb{N}_0^d$ denotes multi-indices. For a sufficiently smooth $\tilde{u}_j$, we can apply the Leibniz rule, which states:

$$\mathcal{D}^\alpha u_{\text{pu}} = \sum_{j=1}^{J} \sum_{\beta \leq \alpha} \binom{\alpha}{\beta} \mathcal{D}^\beta \omega_j \mathcal{D}^{\alpha-\beta} \tilde{u}_j.$$

For example, the Laplace operator is in the form:

$$L = \Delta = \mathcal{D}^{(2,0,\cdots,0)} + \mathcal{D}^{(0,2,\cdots,0)} + \ldots + \mathcal{D}^{(0,0,\cdots,2)},$$

and we can express Δu_{pu} as:

$$\Delta u_{\text{pu}} = \sum_{j=1}^{J} (\tilde{u}_j \Delta\omega_j + 2\nabla\omega_j . \nabla\tilde{u}_j + \omega_j \Delta\tilde{u}_j).$$

Here, computing the derivatives ω_j from Equation (3.4) is complex and has computational cost. To address these issues, an alternative method was used to avoid these challenging computations, reducing both computational cost and algorithmic complexity.

3.3. **Classical RBF-PU method.** Consider the conditionally positive definite function $\psi : \mathbb{R}^d \longrightarrow \mathbb{R}$ of order m, and X as the set of N distinct points in Ω as mentioned earlier. Let $X_\ell = X \cap \Omega_\ell, \quad \ell = 1, \cdots, J$. Consider the following set:

$$J_\ell = \{j \in \{1, \cdots, J\} : \quad \boldsymbol{x}_j \in X_\ell\}.$$

If the local approximation is comes from the following approximation space:

$$\text{span}\{\psi(\boldsymbol{x} - \boldsymbol{x}_j) : \quad j \in J_\ell\} \oplus \Pi_m(\mathbb{R}^d), \quad \ell = 1, \cdots, J,$$

then the local approximation $\tilde{u}_\ell$ is an interpolation of the function u on X_ℓ as follows:

$$\tilde{u}_\ell(\boldsymbol{x}) = \sum_{j\in J_l} c_j \psi(\boldsymbol{x}-\boldsymbol{x}_j) + \sum_{k=1}^{Q} b_k p_k(\boldsymbol{x}), \quad \boldsymbol{x}\in\Omega_\ell\cap\Omega,$$

It can be written in the following Lagrangian form:

$$\tilde{u}_\ell(\boldsymbol{x}) = \sum_{j\in J_\ell} \vartheta_j^*(\ell;\boldsymbol{x}) u(\boldsymbol{x}_j),$$

so that the Lagrangian functions $\vartheta_j^*(l;\boldsymbol{x})$ satisfying in the following linear system of equations

$$\begin{bmatrix} \mathbf{R} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{O} \end{bmatrix} \begin{bmatrix} \vartheta^*(\boldsymbol{x}) \\ \nu(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} \Psi(\boldsymbol{x}) \\ \boldsymbol{p}(\boldsymbol{x}) \end{bmatrix},$$

where

$$\begin{aligned} &\mathbf{R}_{ij} = \psi(\boldsymbol{x}_i-\boldsymbol{x}_j), \quad i,j\in J_l,\\ &\mathbf{P}_{jk} = p_k(\boldsymbol{x}_j), \quad j\in J_l, \quad k=1,2,\cdots,Q,\\ &\Psi(\boldsymbol{x}) = \Psi(\boldsymbol{x}_l), \quad l\in J_l,\\ &\boldsymbol{p}(\boldsymbol{x}) = [p_1(\boldsymbol{x}),\cdots,p_Q(\boldsymbol{x})]^T. \end{aligned}$$

Therefore, we have the following global approximation:

$$u_{\mathrm{pu}}(\boldsymbol{x}) = \sum_{\ell=1}^{J}\sum_{j\in J_\ell} (\omega_\ell(\boldsymbol{x})\psi_j(\ell;\boldsymbol{x}))\tilde{u}(\boldsymbol{x}_j), \quad \boldsymbol{x}\in\Omega. \tag{3.8}$$

Now we want to perform the collocation method RBF-PU for the problem (3.7). To do this, suppose we have the following point set:

$$Y = \{\boldsymbol{y}_1,\cdots,\boldsymbol{y}_M\} \quad M\geq N.$$

Separate Y into two sets, Y_b and Y_i, each of which lies on the boundary or inside Ω, respectively. Therefore $Y = Y_b\cup Y_i$. We use the collocation method to solve the problem (3.7) at points Y

$$\begin{aligned} &(\mathcal{L}u)(\boldsymbol{y}_k) = f(\boldsymbol{y}_k), \quad \boldsymbol{y}_k\in Y_i,\\ &(\mathcal{B}u)(\boldsymbol{y}_k) = g(\boldsymbol{y}_k), \quad \boldsymbol{y}_k\in Y_b. \end{aligned} \tag{3.9}$$

According to what was stated at the beginning of this section and in the first section $\mathcal{L}u$ and $\mathcal{B}u$ are approximated as follows:

$$\begin{aligned} \mathcal{L}u \approx \mathcal{L}\tilde{u} &= \sum_{\ell=1}^{J}\sum_{j\in J_\ell} \mathcal{L}(\omega_\ell\vartheta_j(\ell;.))u(\boldsymbol{x}_j),\\ \mathcal{B}u \approx \mathcal{B}\tilde{u} &= \sum_{\ell=1}^{J}\sum_{j\in J_\ell} \mathcal{B}(\omega_\ell\vartheta_j(\ell;.))u(\boldsymbol{x}_j). \end{aligned}$$

We insert these equations into Equation (3.9), resulting in the following linear system of equations:

$$\begin{bmatrix} L \\ B \end{bmatrix} \mathbf{U} = \begin{bmatrix} f|_{Y_i} \\ g|_{Y_b} \end{bmatrix},$$

That approximate solution of u in points X denoted as $\mathbf{U} = [u_1, \cdots, u_N]^T$ and the matrix components L and B are as follows:

$$L(k,j) = \sum_{\ell \in I_{\boldsymbol{y}_k}} \left(\mathcal{L}(\omega_\ell \vartheta_j^*(\ell;.))\right)(\boldsymbol{y}_k), \quad \boldsymbol{y}_k \in Y_i,$$

$$B(k,j) = \sum_{\ell \in I_{\boldsymbol{y}_k}} \left(\mathcal{B}(\omega_\ell \vartheta_j^*(\ell;.))\right)(\boldsymbol{y}_k), \quad \boldsymbol{y}_k \in Y_b,$$

As mentioned earlier, this device results in challenging and algorithm complexity.

4. Implementation of D-RBF-PU for Poisson Equation

The new D-RBF-PU method was introduced for the first time in [12] to reduce the computational cost, by avoiding differentiation with respect to PU weight functions. This important property, allows using some types of discontinuous weight functions. Comparing with the RBF-FD method, this method is much faster and has better accuracy. Indeed in the D-RBF-PU method, the differential operators just act on the local approximations. In this case we have

$$\mathcal{L}u_{\text{pu}}(\boldsymbol{x}) = \sum_{\ell=1}^{J} \sum_{j \in J_\ell} (\omega_\ell(\boldsymbol{x}) \mathcal{L}\psi_j(\ell;\boldsymbol{x}))\tilde{u}(\boldsymbol{x}_j), \quad \boldsymbol{x} \in \Omega. \tag{4.1}$$

In this section, we apply the D-RBF-PU Method for numerical solution of Poisson Equation. Now, we consider Poisson equation

$$\Delta u(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega, \tag{4.2}$$

with boundary conditions:

$$u(\boldsymbol{x}) = g(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial\Omega,$$

where f and g are given functions. The pure Dirichlet boundary condition is imposed on whole of Ω. Considering $\mathcal{L} = \Delta$, in (4.1) and replacing in (4.2), we get

$$\Delta u_{\text{pu}}(\boldsymbol{x}) = \sum_{\ell=1}^{J} \sum_{j \in J_\ell} (\omega_\ell(\boldsymbol{x}) \Delta\psi_j(\ell;\boldsymbol{x}))\tilde{u}(\boldsymbol{x}_j), \quad \boldsymbol{x} \in \Omega. \tag{4.3}$$

$$u_{\text{pu}}(\boldsymbol{x}) = \sum_{\ell=1}^{J} \sum_{j \in J_\ell} (\omega_\ell(\boldsymbol{x}) \psi_j(\ell;\boldsymbol{x}))\tilde{u}(\boldsymbol{x}_j), \quad \boldsymbol{x} \in \partial\Omega. \tag{4.4}$$

Eventually, collocating the main Equation (4.3) and the boundary condition (4.4) generates the final system, and the approximate solution is obtained by solving it.

5. Test examples

We compute our all computations in MATLAB $R2014a$ on a computer with Intel(R) Core(TM) i5-3230M CPU @ 2.60 GHz; memory (RAM): 4.00 GB; and system type: 32-bit Operating System.

There are three domains Ω_1, Ω_2, Ω_3 that the boundary of them are defined by using polar coordinate as follow:

$$\begin{aligned}
\Omega_1: \quad & r_1(\theta) = 0.25\left(2 + \sin(2\theta) - 0.01\cos(5\theta - \frac{\pi}{2}) + 0.63\sin(6\theta - 0.1)\right), \\
\Omega_2: \quad & r_2(\theta) = 0.5 + 0.25\left(\sin^2(4\theta) + \sin(5\theta)\right) + 0.2\left(\sin^3(\theta)\right), \\
\Omega_3: \quad & r_3(\theta) = 0.7 + 0.12\left(\sin(6\theta) + \sin(3\theta)\right), ,
\end{aligned}$$

where r is a radial coordinate and $\theta \in [0, 2\pi]$ is the angle. These domains are shown in figures, Fig.1, Fig.2 and Fig.3, respectively. The polyharmonic spline (PHS) kernel, defined as

$$\psi_m(r) = \begin{cases} r^{2m}\log r & m \text{ is even}, \\ r^{2m-1} & m \text{ is odd}, \end{cases}$$

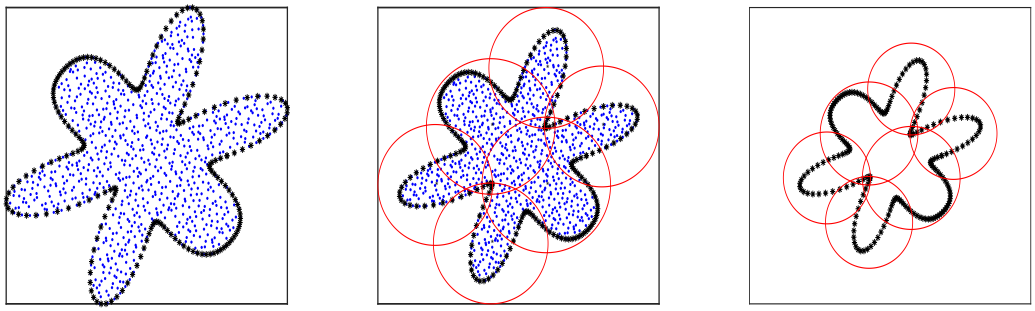

FIGURE 1. Schematic diagrams of the domain Ω_1 with boundary points and internal points and partitioning of domain with circular patches.

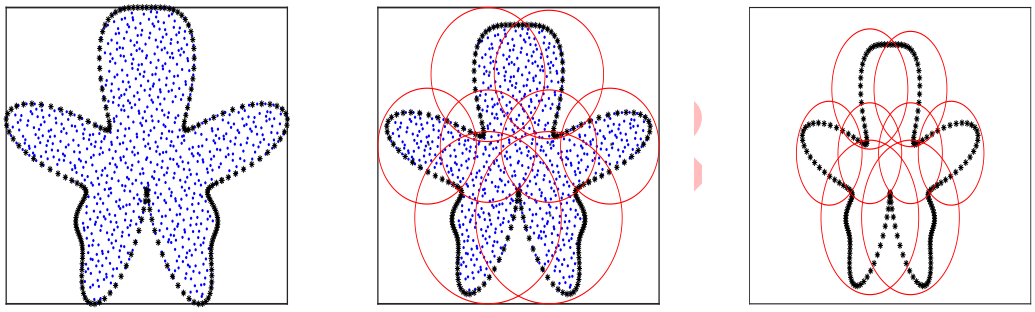

FIGURE 2. Schematic diagrams of the domain Ω_2 with boundary points and internal points and partitioning of domain with circular patches.

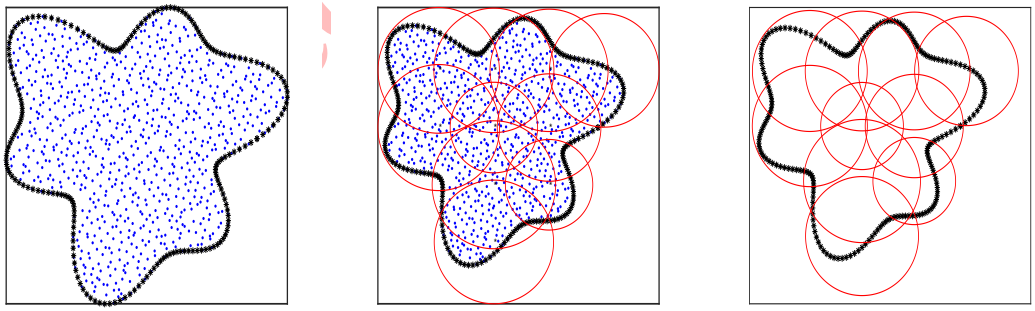

FIGURE 3. Schematic diagrams of the domain Ω_3 with boundary points and internal points and partitioning of domain with circular patches.

is employed as the RBF basis function, which is conditionally positive definite of order m. Some different values for m are chosen, and results are depicted in the figures.

Example 5.1. Consider the function u_1 as exact solution of (4.2), as follows:

$$u_1(\boldsymbol{x}) = \frac{3}{4}e^{\frac{-1}{4}[(9x-2)^2+(9y-2)^2]} + \frac{3}{4}e^{[\frac{-1}{49}(9x+1)^2+\frac{1}{10}(9y+1)^2]} + \frac{1}{2}e^{\frac{-1}{4}[(9x-7)^2+(9y-3)^2]} - \frac{1}{5}e^{-[(9x-4)^2+(9y-7)^2]},$$

where u_1 is the Franke function, and $\boldsymbol{x} = [x, y]^T$.

There is a comparison between the convergence rates and the absolute values of error ($\mathcal{C}_{rate}$) for C-RBF-PU and D-RBF-PU, as reported in Tables 1, 2, 3, and 4, for different values of h (where $h \approx \frac{1}{\sqrt{N}}$) and various PHS kernels with different choices of m. Additionally, the absolute values of error are shown in Figures 4, 5, and 6.

Example 5.2. We use function u_2 as exact solution of (4.2) as follow:

$$u_2(x, y) = \sum_{j=0}^{5} e^{-\sqrt{2^j}} \left(\cos(2^j x) + \cos(2^j y)\right).$$

Similar to previous example, Tables 5, 6, 7, and 8 lists the detailed results of the computation and Figures 8, 9, and 10 plot the computational errors. Already, this equation has been solved by the least squres RBF-FD method in [17].

To evaluate accuracy of this method, we use following infinity relative error norm in this study:

$$\mathcal{E}_\infty = \frac{\| \mathcal{S}_{u,X}(\boldsymbol{x}) - \mathcal{U}(\boldsymbol{x}) \|_\infty}{\| \mathcal{S}_{u,X}(\boldsymbol{x}) \|_\infty}.$$

We compute the convergence rate of the proposed method as follows:

$$\mathcal{C}_{rate} = \frac{\log(\frac{\mathcal{E}_1}{\mathcal{E}_2})}{\log(\frac{h_1}{h_2})},$$

where $\mathcal{E}_1$ and $\mathcal{E}_2$ are errors that corresponding to h_1 and h_2, respectively.

We set polyharmonic spline (PHS) as kernel in this study. This kernel is defined by following form:

$$\psi(r) = \begin{cases} r^b \log r & b \quad \text{even} \\ r^b & b \quad \text{odd}, \end{cases}$$

in which b is a positive real number, $r = \| \boldsymbol{x} \|_2$ and $\psi(r)$ is conditionally positive definite of order $n = \lfloor \frac{b}{2} \rfloor$. We use $\psi(r) = r^5$ and $\psi(r) = r^6 \log r$ as PHSs.

Here, in the PU method, we set covering $\Omega_j = B(\varpi, r_c)$ for $j = 1, \cdots, J$ where points ϖ_j are covering centers. $r_c = 4h$ is covering radius where h is fill distance.

We employed the Wendlands function $\omega_{3,2}$ with $\varepsilon = r_c$ as

$$\omega_{3,2}(r) = \left(1 - (\frac{r}{\varepsilon})\right)_+^6 \left(35(\frac{r}{\varepsilon})^2 + 18(\frac{r}{\varepsilon}) + 3\right).$$

The function $\omega_{3,2}(r)$ is C^4 compactly supported.

TABLE 1. The obtained error, convergence rate $\mathcal{C}_{rate}$, the condition number and the computational time using the C-RBF-PU method with different values of h with the kernel r^5 on Ω_3 for Example 5.1.

h	*Error*	$\mathcal{C}_{rate}$	*Cond*	*time*
0.10000	$2.8682e-03$	-	$8.2826e+03$	0.4052
0.05773	$1.4033e-03$	1.3011	$3.8943e+04$	0.6239
0.03333	$5.2201e-04$	1.8002	$2.0718e+05$	3.6726
0.01924	$2.2340e-04$	1.5446	$8.4742e+05$	58.2016

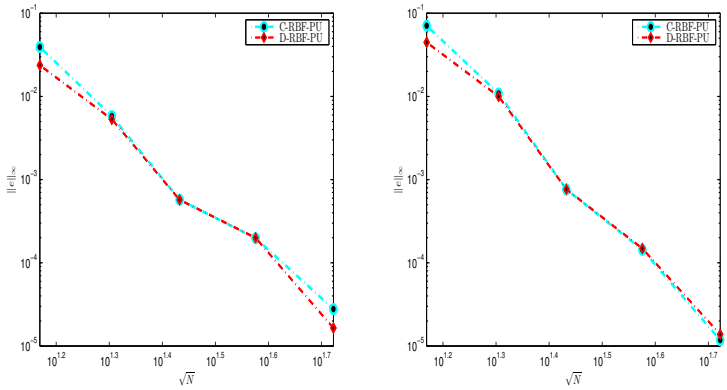

FIGURE 4. The error obtained using the C-RBF-PU method and the D-RBF-PU method with PHS kernels r^7 (left) and $r^8 \log r$ (right) on domain Ω_1 for Example 5.1.

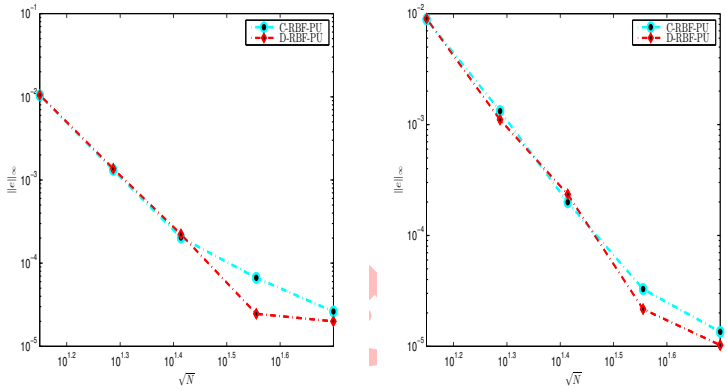

FIGURE 5. The absolute error obtained by using the C-RBF-PU method and the D-RBF-PU method with kernels r^7 (left) and $r^8 \log r$ (right) on Ω_2 for Example 5.1.

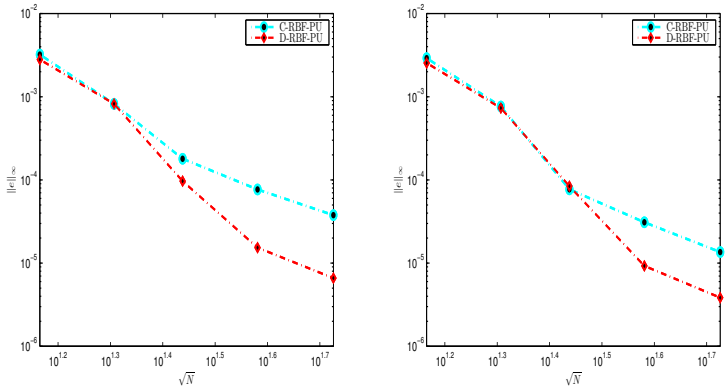

FIGURE 6. The absolute error obtained by using the C-RBF-PU method and the D-RBF-PU method with kernels r^7 (left) and $r^8 \log r$ (right) on Ω_3 for Example 5.1.

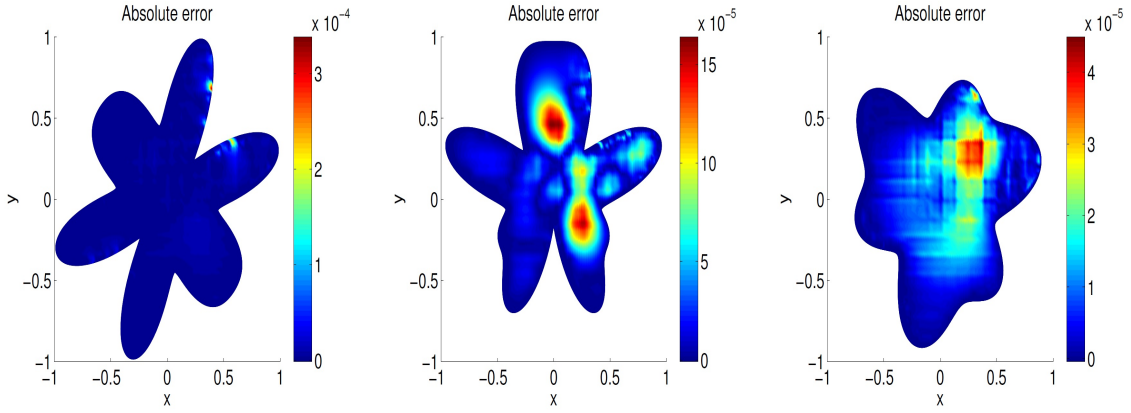

FIGURE 7. Density plot of the error on Ω_1 (left), Ω_2 (middle) and Ω_3 (right) for $h = .1$, with the kernel $r^6 \log$ for Example 5.1.

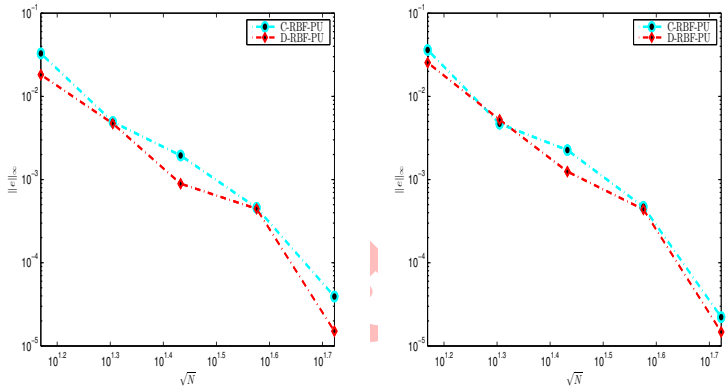

FIGURE 8. The error obtained using C-RBF-PU method and D-RBF-PU method with PHS kernels r^7 (left) and $r^8 \log r$ (right) on domain Ω_1 for Example 5.2.

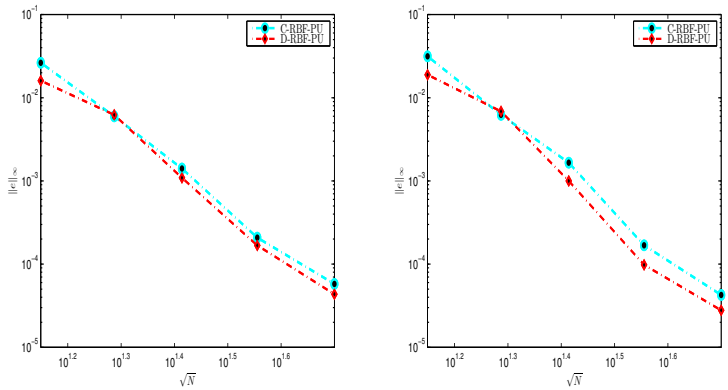

FIGURE 9. The error obtained using C-RBF-PU method and D-RBF-PU method with PHS kernels r^7 (left) and $r^8 \log r$ (right) on on domain Ω_2 for Example 5.2.

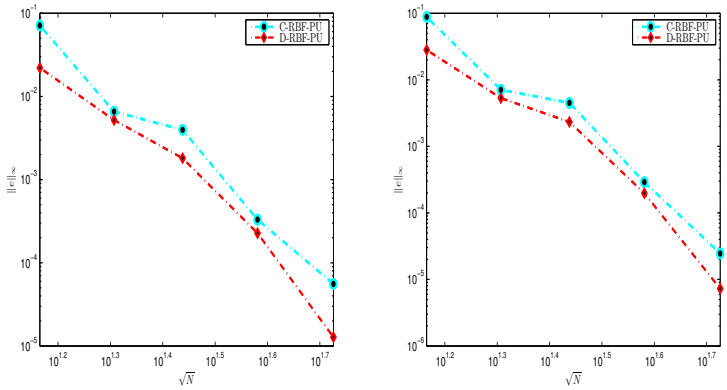

FIGURE 10. The error obtained using C-RBF-PU method and D-RBF-PU method with PHS kernels r^7 (left) and $r^8 \log r$ (right) on domain Ω_3 for Example 5.2.

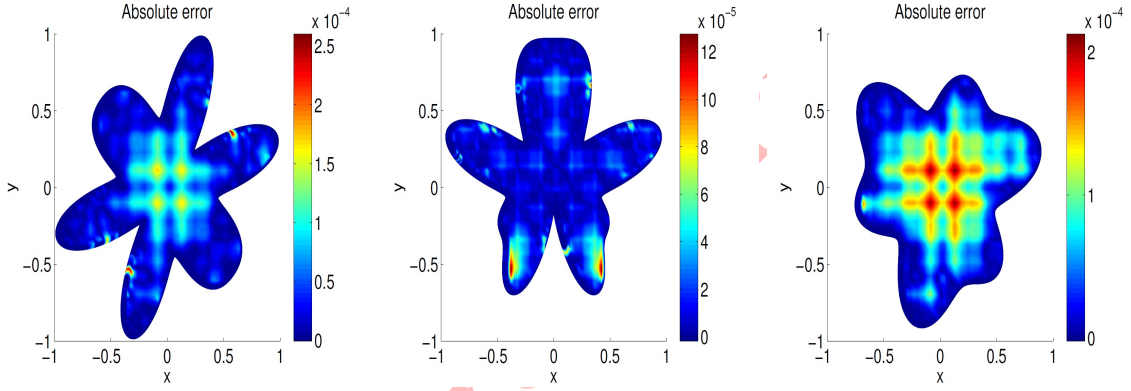

FIGURE 11. Density plot of error on the Ω_1 (left), on the Ω_2 (middle) and on the Ω_3 (right) for $h = .1$ with PHS kernels $r^6 \log$ for Example 5.2.

TABLE 2. The obtained error, $\mathcal{C}_{rate}$, the condition number and the computational time using D-RBF-PU method with different values of h and the PHS kernels r^5 on Ω_3 for Example 5.1.

h	*Error*	$\mathcal{C}_{rate}$	*Cond*	*time*
0.10000	$1.4322e-03$	-	$7.8028e+03$	0.2664
0.05773	$4.1323e-04$	2.2624	$3.9859e+04$	.3415
0.03333	$1.2465e-04$	2.1818	$2.1306e+05$	3.2557
0.01924	$5.4932e-05$	1.4913	$5.4932e+05$	55.0363

6. CONCLUSION

In this study, we use the direct radial basis function partition of unity (D-RBF-PU) method to obtain the numerical solution of the two-dimensional Poisson equation. In addition, different irregular 2D domains, two exact solutions, and various types of polyharmonic spline kernels for this equation are presented in this paper. Finally, through different examples, a comparative study of the efficiency, convergence rate, and speed demonstrates that this method is less expensive, more accurate, and faster than the classical radial basis function partition of unity method.

TABLE 3. The obtained error, $\mathcal{C}_{rate}$, the condition number and the computational time using the C-RBF-PU method with different values of h and the PHS kernels $r^6 \log r$ on Ω_3 for Example 5.1.

h	$Error$	$\mathcal{C}_{rate}$	$Cond$	$time$
0.10000	$1.2391e-03$	-	$9.8904e+03$	0.2802
0.05773	$5.6124e-04$	1.4416	$4.2038e+04$	0.5786
0.03333	$2.0892e-04$	1.7989	$1.6004e+05$	4.0098
0.01924	$8.7740e-05$	1.5789	$9.9442e+05$	57.1031

TABLE 4. The obtained error, $\mathcal{C}_{rate}$, the condition number and the computational time using the D-RBF-PU method with different values of h with the PHS kernels $r^6 \log r$ on Ω_3 for Example 5.1.

h	$Error$	$\mathcal{C}_{rate}$	$Cond$	$time$
0.10000	$8.5341e-04$	-	$8.5131e+03$	0.2040
0.05773	$2.4980e-04$	2.2362	$4.3688e+04$	0.3797
0.03333	$5.7844e-05$	2.6631	$1.8393e+05$	3.1993
0.01924	$2.3860e-05$	1.6116	$9.8714e+05$	55.4523

TABLE 5. The obtained error, $\mathcal{C}_{rate}$, condition number and time using C-RBF-PU method with different values of h with PHS kernels r^5 on domain Ω_2 for Example 5.2.

h	$Error$	$\mathcal{C}_{rate}$	$Cond$	$time$
0.10000	$1.1276e-02$	-	$7.9291e+03$	0.6622
0.05773	$2.6155e-03$	2.6597	$6.5195e+04$	0.5583
0.03333	$9.1794e-04$	1.9061	$4.6443e+05$	2.6694
0.01924	$2.0675e-04$	2.7129	$4.5973e+06$	40.1659

TABLE 6. The obtained error, $\mathcal{C}_{rate}$, condition number and time using D-RBF-PU method with different values of h with PHS kernels r^5 on domain Ω_2 for Example 5.2.

h	$Error$	$\mathcal{C}_{rate}$	$Cond$	$time$
0.10000	$1.3591e-02$	-	$8.0555e+03$	0.2348
0.05773	$2.6771e-03$	2.9572	$6.3298e+04$	.3121
0.03333	$3.3828e-04$	3.7658	$3.2810e+05$	2.3675
0.01924	$4.8249e-05$	3.5443	$3.5417e+06$	38.7791

TABLE 7. The obtained error, $\mathcal{C}_{rate}$, condition number and time using C-RBF-PU method with different values of h with PHS kernels $r^6 \log r$ on domain Ω_2 for Example 5.2.

h	$Error$	$\mathcal{C}_{rate}$	$Cond$	$time$
0.10000	$1.9025e-02$	-	$8.7762e+03$	0.2541
0.05773	$2.6337e-03$	3.5992	$1.4972e+05$	0.5281
0.03333	$4.5905e-04$	3.1803	$8.2592e+05$	3.0229
0.01924	$8.7150e-05$	3.0239	$9.7761e+06$	40.5526

TABLE 8. The obtained error, $\mathcal{C}_{rate}$, condition number and time using D-RBF-PU method with different values of h with PHS kernels $r^6 \log r$ on domain Ω_2 for Example 5.2.

h	$Error$	$\mathcal{C}_{rate}$	$Cond$	$time$
0.10000	$1.4764e-02$	-	$9.0659e+03$	0.1497
0.05773	$2.6844e-03$	3.1029	$1.3527e+05$	0.3139
0.03333	$2.6439e-04$	4.2194	$4.5221e+05$	2.3552
0.01924	$2.2325e-05$	4.4984	$7.7704e+06$	39.2626

REFERENCES

[1] S. Arefian and D. Mirzaei, *A compact radial basis function partition of unity method*, Computers and Mathematics with Applications, *127* (2022), 1–11.

[2] I. Babuska and J. M. Melenk, *The partition of unity method*, The International Journal for Numerical Methods in Engineering, *40*(4) (1997), 727–758.

[3] E. Ben-Ahmed, M. Sadik and M. Wakrim, *Radial basis function partition of unity method for modelling water flow in porous media*, Computers and Mathematics with Applications, *75*(8) (2018), 2925–2941.

[4] F. Bernal, A. Safdari-Vaighani, and E. Larsson, *A radial basis function partition of unity method for steady flow simulations*, Journal of Computational Physics, *503* (2024), 112842.

[5] R. Covoretto and A. De Rossi, *Error indicators and refinement strategies for solving Poisson problems through a RBF partition of unity collocation scheme*, Applied Mathematics and Computation, *369* (2020), 124824.

[6] A. Ebrahimijahan, M. Dehghan, and M. Abbaszadeh, *A reduced-order model based on integrated radial basis functions with partition of unity method for option pricing under jump-diffusion models*, Engineering Analysis With Boundary Elements, *155* (2023), 48–61.

[7] G. E. Fasshauer, *Meshfree Approximation Methods with Matlab*, Word Scientific, (2007).

[8] F. Gholampour, E. Hesamedini, and A. Taleei, *A stable RBF partition of unity local method for elliptic interface problems in two dimensions*, Engineering Analysis with Boundary Elements, *123* (2021), 220–232.

[9] E. Larsson, V. Shcherbakov, and A. Heryudono, *A least squres radial basis function partition of unity method for solving PDEs*, SIAM Journal on Scientific Computing, *39* (2017), A2538–A2563.

[10] J. M. Melenk and I. Babuska, *The partition of unity finite element method: Basic theory and applications*, Computer Methods in Applied Mechanics and Engineering, *139*(1–4) (1996), 289–314.

[11] R. Mir and D. Mirzaei, *The D-RBF-PU method for solving surface PDEs*, Journal of Computational Physics, *479* (2023), 112001.

[12] D. Mirzaei, *The direct radial basis function partiton of unity (D-RBF-PU) method for solving PDEs* SIAM Journal on Scientific Computing, *43* (2021), A54–A83,.

[13] N. Narimani and M. Dehghan, *Predicting the effect of a combination drug therapy on the prostate tumor growth via an improvement of a direct radial basis function partition of unity technique for a diffuse-interface model*, Computers in Biology and Medicine, *157* (2023), 106708.

[14] A. Safdari-Vaighani, A. Heryudono, and E. Larsson, *A radial basis function partition of unity collocation method for convection-diffusion equations arising in financial applications*, Journal of Computational, *64* (2015), 341–367.

[15] D. Shepard, *A two-dimensional interpolation function for irregularly-spaced data*, In Proceedings of the 23th National Conference ACM, (1968), 517–523.

[16] A. E. Tolstykh, *On using RBF-based differencing formulas for unstructured and mixed structured-unstructured grid calculations*, In Proceedings of the 16th IMACS World Congress 228, Lausanne, (2000), 4606–4624.

[17] I. Tomine and Eva Breznik, *An unfitted RBF-FD method in a least-squres setting for elliptic PDEs on complex geometries*, Journal of Computational Physics, *436* (2021), 110283.

[18] H. Wendland, *Fast evalution of radial basis functions: methods based on partition of unity In approximation Theory*, X: Wavelets, Splines, and Applications, Nashville, TN., Vanderbilt University Press, (2002), 473–483.

[19] H. Wendland, *Scattered Data Approximation*, Cambridge University Press, 2005.