A Note on the Direct Approximation of Derivatives in Rational Radial Basis Functions Partition of Unity Method and its Application to the Convection-Diffusion Equations

Vahid Mohammadi[a], Stefano De Marchi[b,*]

[a]*Department of Mathematics, Faculty of Science, Shahid Rajaee Teacher Training University, Tehran, 16785-163, Iran*
[b]*Department of Mathematics "Tullio Levi-Civita", University of Padua, Italy*

Abstract

This paper proposes a Direct Rational Radial Basis Functions Partition of Unity (D-RRBF-PU) approach to compute derivatives of functions with steep gradients or discontinuities. The novelty of the method concerns how derivatives are approximated. More precisely, all derivatives of the partition of unity weight functions are eliminated while we compute the derivatives of the local rational approximants in each patch. As a result, approximate derivatives are obtained more easily and quickly than those obtained in the standard formulation. The corresponding error bounds are briefly discussed. Some numerical results are presented to show the technique's potential. As an application, we develop this meshfree approximation combined with an explicit fourth-order Runge-Kutta time discretization to find the numerical solution of the convection-diffusion equations in two dimensions.

Keywords: Rational RBF-PU method, Direct RRBF-PU approximation, Convection-diffusion equations.
2020 MSC: 32E30.

1. Introduction

The RBF method has been widely used to interpolate scattered data from high-dimensional spaces. The rational approximation is more effective and applicable when approximated with univariate functions with steep gradients or singularities. As detailed in [3, 4, 8], a possible choice is to use rational interpolation, where the numerator and the denominator are polynomials. Some approaches are available to find the unknown coefficients related to the polynomials. One is the Padé approximation, and the other is a least-squares technique [12]. Of course, it is more difficult to implement in high-dimensional spaces because it depends on mesh generation (cf. [4]). In [13], a simplified RRBFs-PU method has been proposed,

*Corresponding author
Email addresses: vahid.mohammadi@sru.ac.ir (Vahid Mohammadi), stefano.demarchi@unipd.it (Stefano De Marchi)

where the constant PU weight functions are considered to approximate the derivatives. The authors of [4] have extended and analyzed the RRBFs-PU method, which is mesh-independent. It depends only on the number and location of scattered data distributed in the computational domain. Accordingly, it could be executed in high-dimensional spaces. This technique is also modified to conditionally positive definite (CPD) functions [11]. Besides, the authors have proved the scaling property of the rational interpolation when the polyharmonic spline (PHS) radial kernels have been utilized. In our investigation, we rely on [4] and [10], but we apply another approach that allows us to describe a new formulation for the RRBF-PU method, the Direct RRBF-PU, shortly DRRBF-PU. To find the approximate value of a derivative in the RRBF-PU, we use a direct form (cf. [10])) instead of its standard version. This reduces the computational cost compared to that of the RRBF-PU method, and it will be observed that the error is at least of the same order as its worst local approximant. It is worth mentioning that the authors in [9] have recently found a compact RBF-PU technique to solve parabolic equations on surfaces. We also cite the paper [5] that has introduced a method for approximating discontinuous functions by variably scaled discontinuous kernels (VSDK).

The remainder of the paper is presented as follows. In Section 2, an RRBF-PU approach is given. The main idea, that is, the DRRBF-PU technique, is proposed in Section 3. Also, a brief discussion related to the error bounds is proposed in this section. In Section 4 we discuss examples of approximation of two-dimensional functions. Section 5 presents the application of the proposed meshfree method to solve the convection-diffusion equations, where the time variable is discretized via an explicit Runge-Kutta algorithm of order 4. Finally, we outline some conclusions in Section 6.

2. The RRBF-PU method

This section briefly reviews the RRBFs-PU method proposed in [4].

We assume that the computational domain, say Ω, is partitioned into the overlapping subdomains, that is, $\Omega \subseteq \bigcup_{l=1}^{N_c} \Omega_l$, with N_c the number of these subdomains. Also, assume that a family of nonnegative weight functions such as $\omega_\ell \in C^k(\mathbb{R}^d)$ forms a partition of unity (PU), where $supp(\omega_l) \subseteq \Omega_l$, which are open, bounded and regular (cf. [15, Definition 15.18] or [7, Chapter 29]). Moreover, we consider $\{\omega_\ell\}_{l=1}^{N_c}$ is *k-stable PU* w.r.t. $\{\Omega_\ell\}$. That is, for every multi-index $\alpha \in \mathbb{N}_0^d$, where $|\alpha| \leq k$, there exists a constant C_α such that

$$||D^\alpha \omega_\ell||_{L_\infty(\Omega)} \leq C_\alpha \rho_\ell^{-|\alpha|}, \tag{2.1}$$

in which $\rho_\ell := \frac{1}{2}\sup_{x,y\in\Omega_\ell}\|x-y\|_2$ [7, 10, 15].

To construct the approximation, a set of trial points $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\} \subset \Omega \subset \mathbb{R}^d$ is assumed. Also, the set of trial points falling in the patch Ω_ℓ is denoted by $X_\ell = X \cap \Omega_\ell$. Accordingly, the index set corresponding to Ω_ℓ for $1 \leq \ell \leq N_c$ is defined to be $J_\ell := \{j \in \{1, ..., N\} : \boldsymbol{x}_j \in X_\ell\}$ (cf. [10]).

Now, we are ready to provide the RRBF-PU approximation for a real function, $f \in C^k(\Omega)$, k being

the order of smoothness (note that the underlying function might not be also not continuous) [4]:

$$f(\boldsymbol{x}) \approx S_{f,X}(\boldsymbol{x}) = \sum_{l=1}^{N_c} \omega_l(\boldsymbol{x}) \mathcal{R}_l(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega, \tag{2.2}$$

in which $R_l(\boldsymbol{x})$ is the local approximation in patch l, and its representation is

$$R_l(\boldsymbol{x}) = \frac{\sum_{j \in J_l} \alpha_j^l \phi(||\boldsymbol{x} - \boldsymbol{x}_j||_2)}{\sum_{k \in J_l} \beta_k^k \phi(||\boldsymbol{x} - \boldsymbol{x}_k||_2)} \tag{2.3}$$

In addition, in (2.2), ω_l is considered to be a compactly supported, nonnegative and non-vanishing weight function on Ω_l. As the weight function, we may consider the Shepard function [7, 10, 15]:

$$\omega_l(\boldsymbol{x}) := \frac{\psi_l(\boldsymbol{x})}{\sum_{j=1}^{N_c} \psi_j(\boldsymbol{x})}, \tag{2.4}$$

in which ψ_l is a compactly supported RBF, and we selected one of the Wendland's family ([15, Chapter 9]).

Furthermore, as explained in [4], in each patch l, let $\mathbf{q}_l$ be the corresponding eigenvector to the smallest eigenvalue of the problem $\Lambda_l \mathbf{q}_l = \lambda_l \Theta_l \mathbf{q}_l$ and

$$\Lambda_l = \frac{1}{\|\mathbf{f}_l\|_2^2} \left(D_l^T A_l^{-1} D_l \right) + A_l^{-1}, \qquad \Theta_l = \frac{1}{\|\mathbf{f}_l\|_2^2} \left(D_l^T D_l \right) + I_l, \tag{2.5}$$

where the vector $\mathbf{f}_l$ is the function values at the trial points located in patch l. D_l is the diagonal matrix, in which the components of $\mathbf{f}_l$ are on its diagonal. Also, $A_l = [\phi(||\boldsymbol{x}_i - \boldsymbol{x}_j||_2)]_{1 \le i,j \le \sharp J_l}$ and I_l is an $\sharp J_l \times \sharp J_l$ identity. The Lagrange form of (2.3) can be written as follows:

$$R_l(\boldsymbol{x}) = \sum_{j \in J_l} \tilde{\alpha}_j^l \phi_R(||\boldsymbol{x} - \boldsymbol{x}_j||_2) = \Phi_R^T(\boldsymbol{x}) \tilde{\alpha}^{\mathbf{l}} = \Phi_R^T(\boldsymbol{x}) A_{\phi_R}^{-1} \mathbf{f}_{X_l}, = \Psi^T(\boldsymbol{x}) \mathbf{f}_{X_l}, = \sum_{j \in J_l} \Psi_j(\boldsymbol{x}) f_j, \tag{2.6}$$

in which $\Phi_R^T(\boldsymbol{x}) = [\phi_R(||\boldsymbol{x} - \boldsymbol{x}_j||_2)]_{1 \le j \le \sharp J_l}$. We also have used $A_{\phi_R} \tilde{\alpha}^{\mathbf{l}} = \mathbf{f}_{X_l}$, and defined $\Phi_R^T(\boldsymbol{x}) A_{\phi_R}^{-1} := \Psi^T(\boldsymbol{x})$. Note that if ϕ is a positive definite radial function, ϕ_R is also (cf. [4]). In our implementation here, we have applied the method of diagonal increments (MDI) to bypass the ill-conditioning of the matrix A_l (cf. [13]). We also refer the reader to [4, 8] for more details related to the RRBFs approach.

3. The DRRBF-PU method

To compute the approximate derivatives of real-valued function, say f at each point $\boldsymbol{x} \in \Omega$ via Eq. (2.2), the standard form takes as:

$$D^\alpha f(\boldsymbol{x}) \approx D^\alpha S_{f,X}(\boldsymbol{x}) = \sum_{l=1}^{N_c} D^\alpha(\omega_l(\boldsymbol{x})\mathcal{R}_l(\boldsymbol{x})), \quad \boldsymbol{x} \in \Omega, \tag{3.1}$$

which shows the derivative operator D^α should act on both the weight functions ω_l and the local approximation $\mathcal{R}_l$. Hence, the computational cost would be much more expensive. To avoid computing all derivatives of the PU weight functions, we apply the same approach proposed by Mirzaei in [10] for the non-rational case. Thanks to that approach, the complicated calculations of the RRBF-PU approach could be ignored when a direct scheme is applied. That idea also reduces the computational cost compared to the standard RBF-PU method (cf. [10]). Here, we use this approach for our purposes. So, the derivatives in (2.2) are approximated as

$$D^\alpha f(\boldsymbol{x}) \approx \widehat{D^\alpha S_{f,X}}(\boldsymbol{x}) = \sum_{l=1}^{N_c} \omega_l(\boldsymbol{x}) D^\alpha \mathcal{R}_l(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega, \tag{3.2}$$

where $D^\alpha S_{f,X}(\boldsymbol{x}) \neq \widehat{D^\alpha S_{f,X}}(\boldsymbol{x})$ for each $\boldsymbol{x} \in \Omega$, and $D^\alpha \mathcal{R}_l$ is the local rational approximation of $D^\alpha f$ in the l-th patch. This shows that we have directly approximated $D^\alpha f$ using the PU approach. To be more precise, we first find the derivatives of the local rational approximation via the RBF in each patch Ω_ℓ. After this, we blend these approximations using the PU technique through the weight functions ω_l to get a global approximation $\widehat{D^\alpha S_{f,X}}(\boldsymbol{x})$ to $D^\alpha f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \Omega$. From another point of view, we can only think of the action of the differential operator based on the local approximation. So, we can ignore the smooth assumption on the PU weights [10]. It is called a *direct approximation* since the method applies to RRBF in the PU setting.

To discuss the error bounds for the derivatives of the function approximated via the proposed DRRBF-PU method, we can consider the following. Assume that in each patch $\Omega \cap \Omega_l$, $D^\alpha f$ is approximated by $D^\alpha R_l$, where $\alpha \in \mathbb{N}_0^d$, $|\alpha| \leq k/2$ is a multi-index, so that

$$||D^\alpha f - D^\alpha R_l||_{L_\infty(\Omega \cap \Omega_l)} \leq \varepsilon_l.$$

Now, for any $\boldsymbol{x} \in \Omega$ and $\alpha \in \mathbb{N}_0^d$ with $|\alpha| \leq k/2$, we can write

$$\begin{aligned} |D^\alpha f(\boldsymbol{x}) - \widehat{D^\alpha S_{f,\boldsymbol{x}}}(\boldsymbol{x})| \leq \sum_{l=1}^{N_c} \omega_l(\boldsymbol{x}) \, |D^\alpha f(\boldsymbol{x}) - D^\alpha \mathcal{R}_l(\boldsymbol{x})| &\leq \sum_{l=1}^{N_c} \omega_l(\boldsymbol{x}) \, \|D^\alpha f - D^\alpha \mathcal{R}_l\|_{L_\infty(\Omega \cap \Omega_l)} \\ &\leq \max_{1 \leq l \leq N_c} \|D^\alpha f - D^\alpha \mathcal{R}_l\|_{L_\infty(\Omega \cap \Omega_l)} \leq \max_{1 \leq l \leq N_c} \varepsilon_l, \end{aligned} \tag{3.3}$$

which shows that the error bound of the derivatives of the DRRBF-PU technique is at least as good as its worst local interpolation. This bound is derived using only the PU property of the weight functions. So, discontinuous weights can also be utilized. It is also worth mentioning that the estimation for local errors ε_l is not available, and it needs a more in-depth study in future work. Of course, in the case $|\alpha| = 0$, these errors follow from [4, Proposition 3.2]. However, in the next section, we report the numerical convergence rates for the function and its first derivative.

4. Approximation of functions: tests

We consider the $\mathcal{C}^6$ Matérn kernel:

$$\phi(r) = e^{-cr}(15 + 15(cr) + 6(cr)^2 + (cr)^3), \tag{4.1}$$

whose native space is $\mathcal{N}_\phi(\mathbb{R}^d) = H^\nu(\mathbb{R}^d)$, with $\nu = \frac{d+7}{2}$ (cf. [7, §4.4, p. 41]). Here we choose $d = 2$ and thus $\mathcal{N}_\phi(\mathbb{R}^2) = H^{4.5}(\mathbb{R}^2)$. Moreover, we take $c = 35$ as a shape parameter (in an empirical way since it gives more stable and accurate solutions). In addition, as a weight function, we choose [7, 10, 15]:

$$\psi_\ell = \psi(||. - \boldsymbol{x}_\ell||/\rho_\ell), \tag{4.2}$$

with

$$\psi(r) = \begin{cases} (1-r)^6(35r^2 + 18r + 3), & 0 \le r \le 1, \\ 0, & r > 1, \end{cases} \tag{4.3}$$

the Wendland $\mathcal{C}^4$ function which is positive definite on $\mathbb{R}^d$ for $d \le 3$. Moreover, $\boldsymbol{x}_\ell$ and ρ_ℓ represent the center point of the patch Ω_ℓ and the radius of the patches, respectively.

As a first test, we consider the two-dimensional function with a steep gradient [6]:

$$f(x, y) = \tan^{-1}\left(125(\sqrt{(x-1.5)^2 + (y-0.25)^2} - 0.92)\right), \quad (x, y) \in [0, 1]^2. \tag{4.4}$$

The exact and approximate solutions of f and $\frac{\partial f}{\partial x}$ are plotted in Figure 1 using a uniform grid of $N = 16641 = 129^2$ nodes, with to be $N_c = 1024$ and $\rho_\ell = 1/\sqrt{N_c}$, respectively.

In addition, to measure the accuracy of the proposed technique, here we have used the following ℓ_2 relative error defined by:

$$||e_\beta||_2 := \frac{||\dfrac{\partial^\beta f}{\partial x^\beta} - \dfrac{\partial^\beta S_{f,X}}{\partial x^\beta}||_2}{||\dfrac{\partial^\beta f}{\partial x^\beta}||_2}, \quad \beta = 0, 1, \tag{4.5}$$

so that $||e_0||_2$ and $||e_1||_2$ will denote the ℓ_2 relative errors on the evaluation of function and its first derivative (in direction x), respectively. Note that the results for the first derivative in the y direction are

the same. So, we have not reported them here for brevity. The relative errors are computed by evaluating the numerator and the denominator on a uniform grid of 100×100 points of $[0,1]^2$.

In Table 1, we report the relative errors ℓ_2 of computing the approximate values of f, the values of the partial derivatives along x, and the numerical convergence rates.

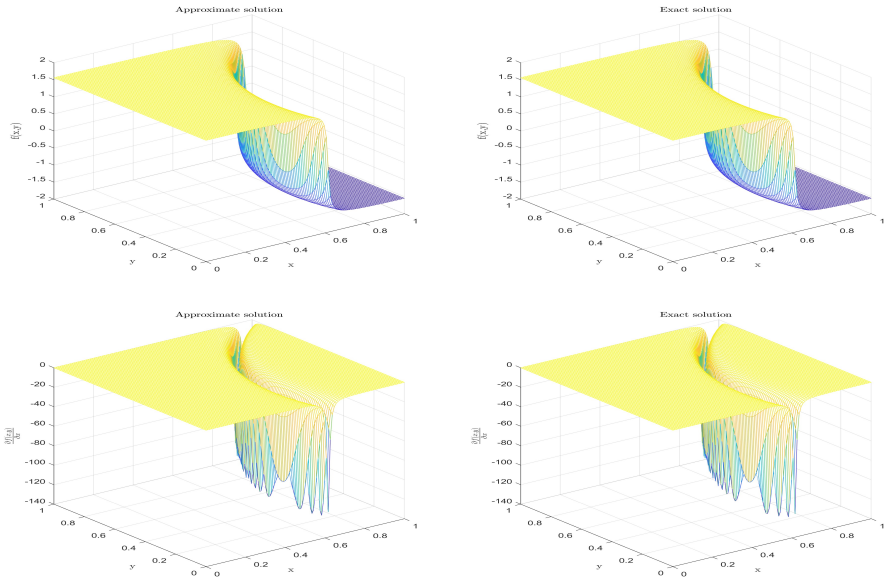

Figure 1: The approximate (left) and exact (right) solutions to function 4.4 (first row), and the approximate (left) and exact derivatives (second row).

N	$\|\|e_0\|\|_2$	$Orders$	N	$\|\|e_1\|\|_2$	$Orders$
1089	$1.56e-2$	–	1089	$2.74e-1$	–
4225	$3.50e-3$	2.10	4225	$8.66e-2$	1.66
16641	$1.72e-4$	4.28	16641	$2.17e-2$	2.00
65536	$8.94e-6$	4.27	65536	$1.61e-3$	3.75

Table 1: For different N, ℓ_2 relative errors on f (left), ℓ_2 relative errors on $\frac{\partial f}{\partial x}$ (right) and the corresponding order of convergence.

For the second test, we have considered the function also analyzed in [4]:

$$f(x,y) = \frac{\tan\Big(9(y-x)+1)\Big)}{\tan(9)+1}, \quad (x,y) \in [0,1]^2. \tag{4.6}$$

This function has singularities across six lines $y = x + \frac{(k-9/2)\pi-1}{9}$, $k = 1, \ldots, 6$ (cf. [6]). In Figure 2, the exact and approximate solutions to f and $\frac{\partial f}{\partial x}$ are pictured using the same N, c, N_c, and ρ_ℓ as in the previous example. Table 2 collects the corresponding ℓ_2 errors.

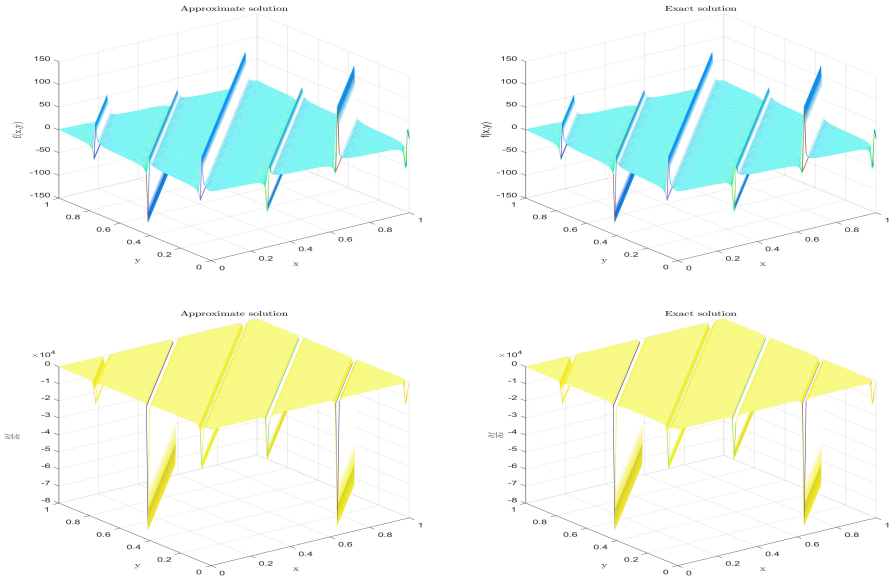

Figure 2: The approximate (left) and exact (right) solutions to function 4.6 (first row), and the approximate (left) and exact derivatives (second row).

N	$\|\|e_0\|\|_2$	$Orders$	N	$\|\|e_1\|\|_2$	$Orders$
1089	$5.02e-2$	–	1089	$9.95e-2$	–
4225	$4.43e-3$	3.52	4225	$9.08e-3$	3.45
16641	$2.71e-4$	4.03	16641	$5.12e-4$	4.15
65536	$1.43e-5$	4.24	65536	$7.24e-5$	2.82

Table 2: For different N, ℓ_2 relative errors on f (left), ℓ_2 relative errors on $\frac{\partial f}{\partial x}$ (right) and the corresponding order of convergence.

5. Application to the convection-diffusion equation

The proposed DRRBF-PU method is implemented, combined with an explicit fourth-order Runge-Kutta (RK4) time integration to the following convection-diffusion equation (cf. [1])

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} + \frac{\partial g(u)}{\partial y} = \nu \Delta u, \tag{5.1}$$

where $u =: u(\boldsymbol{x}, t)$, $(\boldsymbol{x}, t) \in \Omega \times (0, T]$ with $\Omega \subset \mathbb{R}^2$ is a nonempty and bounded set, ν is the diffusion coefficient. The nonlinear fluxes $f(u)$ and $g(u)$ are differentiable. Moreover, the Dirichlet or periodic boundary conditions are considered for the unknown variable u. This model has been utilized to describe important phenomena in different sciences and engineering branches, for example, gas dynamics, fluid flows in porous media, and transport procedures (see [1] for more details). As it is known, when the convection is dominated, most standard numerical techniques, such as mesh-dependent and meshfree, do not give a stable and accurate solution. Indeed, steep fronts, sharp discontinuities, boundary layers, and shocks appear in the solution. Hence, some improvements such as the hyperviscosity formulation as a stabilizer term could be added to the original problem to avoid any spurious growth in the numerical solution (among many research works, see, e.g., [14]). However, other numerical methods such as the RRBFs interpolant or RRBFs-PU approximation are successfully designed to solve these problems (cf. [13]).

Now, to discretize (5.1) in space, we first consider a set of scattered trial points $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\} \subset \Omega$ in $\mathbb{R}^2$, which contains both the interior and the boundary points, i.e.,

$$X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_{N_I}\} \cup \{\boldsymbol{x}_{N_I+1}, ..., \boldsymbol{x}_{N_I+N_B}\},$$

and $N = N_I + N_B$. Also, we consider N_c patches so that $\{\Omega_\ell\}_{\ell=1}^{N_C}$ is a covering of Ω. The set $X_\ell = X \cap \Omega_\ell$ is the set of trial points in the ℓ-th patch, and J_ℓ is previously defined.

Hence, for $t \geq 0$, an approximation to u can be written as follows:

$$u(\boldsymbol{x}, t) \approx \widehat{u}(\boldsymbol{x}, t) = \sum_{l=1}^{N_c} \sum_{j \in J_l} \omega_l(\boldsymbol{x}) \Psi_j(\boldsymbol{x}) u_j(t), \quad \boldsymbol{x} \in \Omega, \tag{5.2}$$

in which $\{\Psi_j(\boldsymbol{x})\}_{j \in J_\ell}$ are the trial functions at point $\boldsymbol{x}$ (see relation (2.6)). Inserting the approximation

(5.2) in (5.1) at points X_I we get

$$\sum_{l=1}^{N_c}\sum_{j\in J_l}\omega_l(\boldsymbol{x}_i)\Psi_j(\boldsymbol{x}_i)\frac{du_j(t)}{dt}+f'(\widehat{u}(\boldsymbol{x}_i,t))\Big(\sum_{l=1}^{N_c}\sum_{j\in J_l}\omega_l(\boldsymbol{x}_i)\frac{\partial\Psi_j(\boldsymbol{x}_i)}{\partial x}u_j(t)\Big)+$$
$$g'(\widehat{u}(\boldsymbol{x}_i,t))\Big(\sum_{l=1}^{N_c}\sum_{j\in J_l}\omega_l(\boldsymbol{x}_i)\frac{\partial\Psi_j(\boldsymbol{x}_i)}{\partial y}u_j(t)\Big)=\nu\sum_{l=1}^{N_c}\sum_{j\in J_l}\omega_l(\boldsymbol{x}_i)\Delta\Psi_j(\boldsymbol{x}_i)u_j(t),\quad 1\le i\le N_I. \tag{5.3}$$

The boundary conditions give

$$\sum_{l=1}^{N_c}\sum_{j\in J_l}\omega_l(\boldsymbol{x}_i)\mathcal{B}\Psi_j(\boldsymbol{x}_i)u_j(t)=g_{\mathcal{B}}(\boldsymbol{x}_i,t),\quad N_I+1\le i\le N_I+N_B, \tag{5.4}$$

where $\mathcal{B}$ is a boundary operator defined on Ω, and $g_{\mathcal{B}}$ is a known function on the boundary.

Equations (5.3) and (5.4) can be written as a system of ODEs in time, that is

$$\begin{aligned}&\frac{d\mathbf{u}_I(t)}{dt}=-f'(\mathbf{u}_I(t)).*(A_x^I\mathbf{u}(t))-g'(\mathbf{u}_I(t)).*(A_y^I\mathbf{u}(t))+\nu(L^I\mathbf{u}(t))\\&A_B\mathbf{u}(t)=\mathbf{g}_B(t),\end{aligned} \tag{5.5}$$

where, thanks to the DRRBF-PU method, A_x^I, A_y^I, L^I are sparse differentiation matrices related to the first derivatives and Laplace operator computed at the interior points. A_B is a sparse matrix related to the boundary operator computed at the boundary points. The notation ".$*$" denotes the point-wise product between each element of two vectors. Note that, due to the method of constructing differentiation matrices via the DRRBF-PU method (see Section 2), we need the values of u at points X per time step. Hence, these differentiation matrices should be constructed in time, which increases the computational cost significantly (see, e.g., [13]). Here, we have built these matrices via the initial values of u at points X, which are available. This type of implementation permits us to have time-independent matrices, which could reduce the computational cost of solving such a time-dependent nonlinear PDE problem using the proposed meshfree method.

In addition, in the discrete scheme (5.5), $\mathbf{u}_I(t)$ and $\mathbf{u}(t)$ are the vectors containing the approximate values of u at points X_I and X, respectively.

To deal with the fully discrete scheme for the model (5.1), we have used an explicit RK4 time integration for (5.5). We note that the right-hand side of (5.5) is $\mathbf{F}(\mathbf{u}(t),A_x^I,A_y^I,L^I)$.

The time interval $[0,T]$ is divided into M sub-intervals such that $[0,T]=\bigcup_{n=1}^{M}[t_{n-1},t_n]$, where $t_n=n\Delta t$

(Δt is the time step so that $t_0 = 0$ and $t_M = T$). Hence, for $n = 0, ..., M-1$,

$$\begin{aligned}
&\mathbf{U}_I^{n+1} = \mathbf{U}_I^n + \frac{\Delta t}{6}\Big(\mathbf{K}_1 + 2(\mathbf{K}_2 + \mathbf{K}_3) + \mathbf{K}_4\Big),\\
&\mathbf{K}_1 = \mathbf{F}(\mathbf{U}^n, A_x^I, A_y^I, L^I),\\
&\mathbf{K}_2 = \mathbf{F}(\mathbf{U}^n + \frac{\Delta t\mathbf{K}_1}{2}, A_x^I, A_y^I, L^I),\\
&\mathbf{K}_3 = \mathbf{F}(\mathbf{U}^n + \frac{\Delta t\mathbf{K}_2}{2}, A_x^I, A_y^I, L^I),\\
&\mathbf{K}_4 = \mathbf{F}(\mathbf{U}^n + \Delta t\mathbf{K}_3, A_x^I, A_y^I, L^I),
\end{aligned} \tag{5.6}$$

and we must solve the following small linear system to find the approximate values of u at boundary points

$$A_B^B \mathbf{U}_B^{n+1} = \mathbf{g}_B^{n+1} - A_B^I \mathbf{U}_I^{n+1}, \tag{5.7}$$

A_B^B is a sparse matrix obtained by eliminating the columns related to the interior points from the matrix A_B. Also, A_B^I is the sparse matrix, where the columns related to the boundary points are eliminated from the matrix A_B. Note that if the Dirichlet boundary conditions are implemented, the linear system (5.7) switches to

$$\mathbf{U}_B^{n+1} = \mathbf{g}_B^{n+1}. \tag{5.8}$$

5.1. A rotating Gaussian pulse

The governing equation is given by

$$\frac{\partial u}{\partial t} - 4y\frac{\partial u}{\partial x} + 4x\frac{\partial u}{\partial y} = \nu\Delta u, \tag{5.9}$$

where $(x, y) \in [0, 1]^2$. The initial and boundary conditions are taken so that they satisfy the exact solution, which is

$$u(x, y, t) = \frac{\sigma^2}{\sigma^2 + 4\nu t}\exp(-\frac{(\tilde{x} - x_0)^2 + (\tilde{y} - y_0)^2}{\sigma^2 + 4\nu t}), \quad (x, y) \in [0, 1]^2, \quad t \geq 0, \tag{5.10}$$

with $\tilde{x} = x\cos(4t) + y\sin(4t)$, $\tilde{y} = -x\sin(4t) + y\cos(4t)$, $x_0 = 0.5$, $y_0 = 0.75$ and $\sigma^2 = 0.002$.

In this test, we have considered two different values of ν: $\nu_1 = 0.001$ and $\nu_2 = 0.0005$, respectively. The relative errors (4.5) are computed in Table 3 after 1 revolution (i.e., $T = \frac{\pi}{2}$) via $\Delta t = \frac{\pi}{8000}$ and different values of N, where the uniform points are utilized to construct the trial approximation (5.2). Note that, to get a more stable and accurate solution, the radius of the patch, i.e., ρ_ℓ, is chosen to be $6h$, where h is

the mesh size.

In Figure 3, the numerical solutions of u are displayed for ν_1 and ν_2 after 1 revolution. These results are obtained by employing $N = 65536$ uniform nodes and $\Delta t = \frac{\pi}{8000}$.

	ν_1			ν_2	
N	$\|\|e_0\|\|_2$	$Orders$	N	$\|\|e_0\|\|_2$	$Orders$
4225	$1.71e-3$	–	4225	$3.36e-3$	–
16641	$1.26e-4$	3.76	16641	$2.38e-4$	3.82
65536	$4.18e-6$	4.91	65536	$9.82e-6$	4.60

Table 3: For different N, ℓ_2 relative errors of the approximate solution of rotating Gaussian pulse problem after 1 revolution for ν_1 (left) and ν_2 (right) and the corresponding order of convergence.

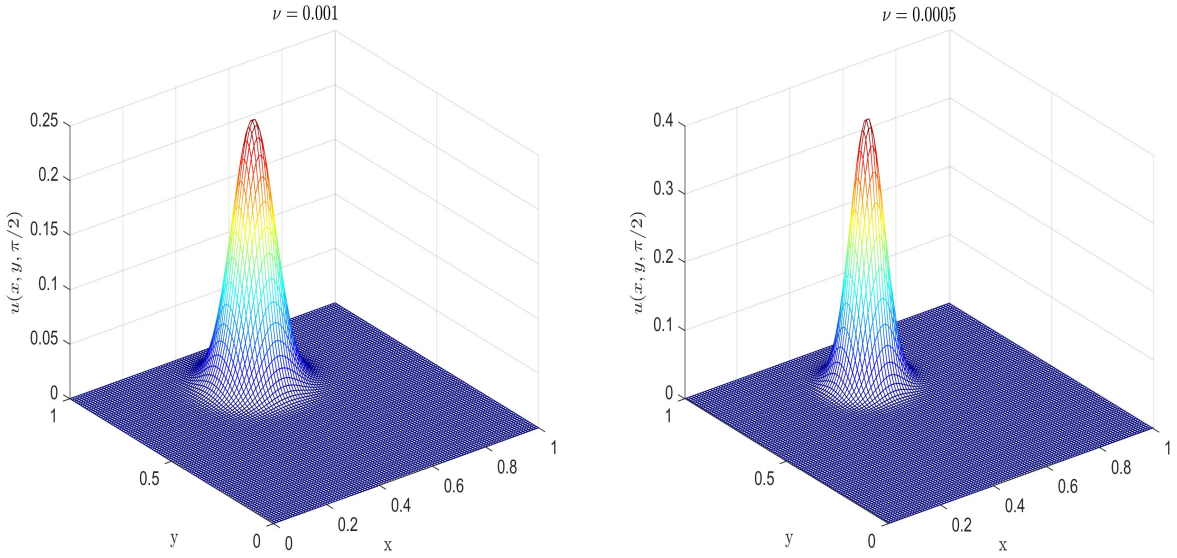

Figure 3: The numerical solution of advection-diffusion equation (5.9) for ν_1 (left) and ν_2 (right) after 1 revolution.

5.2. The inviscid Burgers equation

We want to solve numerically the inviscid Burgers equation (cf. [2]):

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + u\frac{\partial u}{\partial y} = \nu \Delta u, \tag{5.11}$$

where $(x, y) \in [0, 1]^2$ and $\nu = 0.001$. Periodic boundary conditions are considered while the initial condition is

$$u(x, y, 0) = \sin^2(\pi x) \sin^2(\pi y), \quad (x, y) \in [0, 1]^2. \tag{5.12}$$

To compute the numerical solution, we have utilized $N = 65536$ uniform nodes and $\Delta t = 10^{-4}$. Also, the radius of each patch is chosen to be $\rho_\ell = 1/\sqrt{N_c}$. The simulation results at $t = 1,\ 2$ and 3 are presented in Figure 4, which shows a shock and its propagation along the main diagonal by increasing time.

6. Conclusion

In this paper, we have presented the DRRBF-PU technique for computing the derivatives of functions with a steep gradient or some singularities. After outlining the numerical formulation of the method, a brief discussion on the error bounds is provided. This showed that the ℓ_2 relative errors are of the same order as the worst local approximate constructed via the positive definite kernels. However, error estimations for approximating the derivatives regarding the fill distance are left for future work. The two examples presented show the potential of the numerical method. Besides, to show the potency of the proposed meshfree approximation in solving the PDE problems, its implementation combined with an explicit Runge-Kutta algorithm of order 4 has been designed for the convection-diffusion equation in two dimensions. Accordingly, two famous examples, i.e., the rotating Gaussian pulse and the inviscid Burgers problems, have been simulated. Finally, for reproducibility of numerical tests in `MATLAB`, the codes are available at the link `https://github.com/VM-2020-MATH/DRRBFs-PU-Method`.

Acknowledgments. The second author thanks for collaborating with the *Approximation Theory and Applications* topical group of the Italian Mathematical Union, RITA the Italian Network on Approximation, and the INdAM-GNCS group.

References

[1] F. Benkhaldoun, S. Sari, M. Seaid, A family of finite volume Eulerian-Lagrangian methods for two-dimensional conservation laws, J. Comput. Appl. Math. 285 (2015) 181-202.

[2] F. Benkhaldoun, A. Halassi, D. Ouazar, M. Seaid, A. Taik, Slope limiters for radial basis functions applied to conservation laws with discontinuous flux function, Eng. Anal. bound. Elem. 66 (2016) 49-65.

[3] M. Buhmann, S. De Marchi and E. Perracchione, Analysis of a new class of rational RBF expansions IMA J. Numer. Analysis 40(3) (2020), pp.1972–1993.

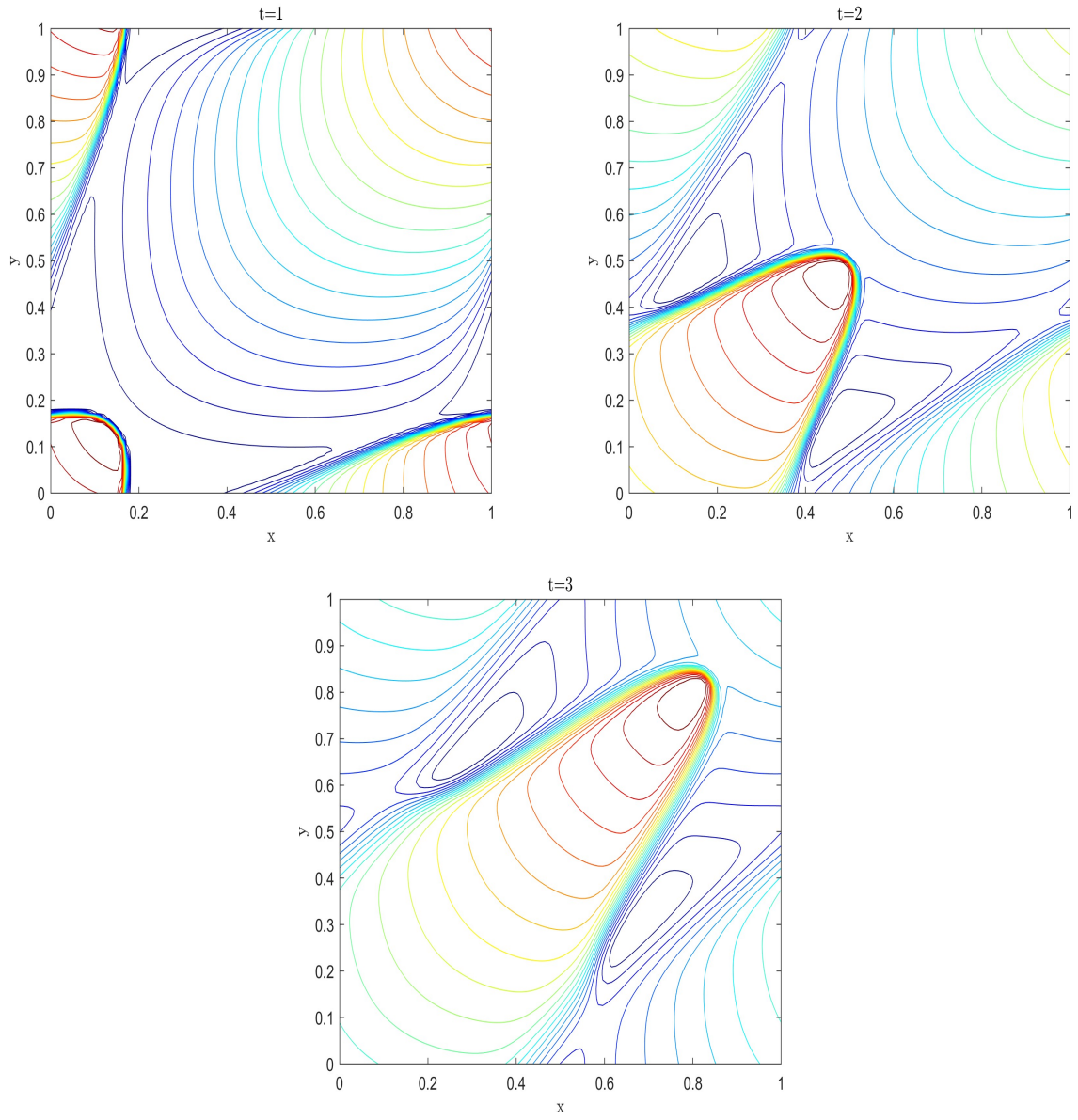

Figure 4: The numerical solution of inviscid Burgers equation (5.11) for $t = 1, 2, 3$.

[4] S. De Marchi, A. Martinez, E. Perracchione, Fast and stable rational RBF-based partition of unity interpolation, J. Comput. Appl. Math. 349 (2019) 331-343.

[5] S. De Marchi, F. Marchetti and E. Perracchione, Jumping with variably scaled discontinuous kernels (VSDK), BIT Numerical Mathematics, 60 (2020) 441-463,

[6] E. Farrazandeh, D. Mirzaei, A rational RBF interpolation with conditionally positive definite kernels, Adv. Comput. Math. 47 (2021) https://doi.org/10.1007/s10444-021-09900-8.

[7] G. E. Fasshauer, Meshfree Approximation Methods with MATLAB, Vol. 6, World Scientific, 2007.

[8] S. Jakobsson, B. Andersson, F. Edelvik, Rational radial basis function interpolation with applications to antenna design, J. Comput. Appl. Math. 233 (2009) 889-904.

[9] Y. Liu, Y. Qiao, X. Feng, A compact direct radial basis function partition of unity method for parabolic equations on surfaces, Int. Commun. Heat Mass Transf. 161 (2025) 108422.

[10] D. Mirzaei, The direct radial basis function partition of unity (D-RBF-PU) method for solving PDEs, SIAM J. Sci. Comput. 43 (2021) A54-A83.

[11] R. Mir, D. Mirzaei, The D-RBF-PU method for solving surface PDEs, J. Comput. Phys. 479 (2023) 112001.

[12] H. Padé, Sur la Répresentation Approchée D'une Fonction par des Fractions Rationelles, Vol. 9 (thesis), Ann. École Nor., 1892, pp. 1–93.

[13] S. A. Sarra, Y. Bai, A rational radial basis function method for accurately resolving discontinuities and steep gradients, Appl. Numer. Math. 130 (2018) 131-142.

[14] V. Shankar, A. L. Fogelson, Hyperviscosity-based stabilization for radial basis function-finite difference (RBF-FD) discretizations of advection-diffusion equations, J. Comput. Phys. 372 (2018) 616-639.

[15] H. Wendland, Scattered Data Approximation, Cambridge University Press, New York, 2004.