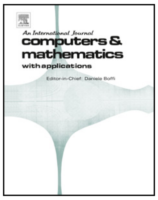
Computers and Mathematics with Applications 127 (2022) 1–11

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

www.elsevier.com/locate/camwa



A compact radial basis function partition of unity method

Sara Arefian [a], Davoud Mirzaei [a,b,*]

[a] Department of Applied Mathematics and Computer Science, Faculty of Mathematics and Statistics, University of Isfahan, 81746-73441 Isfahan, Iran
[b] School of Mathematics, Institute for Research in Fundamental Sciences (IPM), Tehran 19395-5746, Iran

ARTICLE INFO

Keywords:
Radial basis function (RBF)
Partial differential equations (PDEs)
Partition of unity (PU)
RBF-FD
Generalized interpolation
Hermit-Birkhoff interpolation
Compact formulas

ABSTRACT

In this work we develop the standard Hermite interpolation based RBF-generated finite difference (RBF-HFD) method into a new faster and more accurate technique based on partition of unity (PU) method. In the new approach, much fewer number of local linear systems needs to be solved for calculating the stencil weights. This reduces the computational cost of the method, remarkably. In addition, the method is flexible in using different types of PU weight functions, smooth or discontinuous, each results in a different scheme with additional nice properties. We also investigate the scaling property of polyharmonic spline (PHS) kernels to develop a simple and stable algorithm for computing local approximants in PU patches. Experimental results confirm the efficiency and applicability of the proposed method.

1. Introduction

Numerical methods based on *radial basis functions* (RBFs) for solving partial differential equations (PDEs) have received a lot of attention because of some attractive advantages. These methods are usually easy to implement, applicable on scattered point layouts, flexible with respect to geometry and dimension, and highly accurate for smooth solutions. However, global implementation of RBF methods may suffer from dense and ill-conditioned final linear systems which make them restricted for large scale problems. Beside, for N points on the domain, the cost of that method scales as $O(N^3)$ and due to the dense nature of the resulting differentiation matrices the cost of applying those matrices to solution vectors is $O(N^2)$. These are the main reasons for paying attention to localized RBF approaches to develop new techniques which are more stable and less costly to apply than the global RBF methods while retain the ability to use scattered nodes to approximate derivatives. The RBF-FD and RBF-PU methods have been developed in this direction.

RBF-FD borrows its idea from classical finite differences (FD) where the stencil weights are usually computed using univariate polynomial interpolation. These one dimensional formulas can be combined to create FD formulas in higher dimensions for partial derivatives. This strategy, however, requires that the nodes of the stencils are situated on some kind of structured grid which severely limits the geometric flexibility of the FD method. The multivariate FD formulas can be instead organized on stencils with scattered nodes, but this approach also raises the question of how the stencil weights should be computed for example in cases in which the polynomial interpolation is singular. These drawbacks can be bypassed if RBF interpolation is used instead to generate the weights in FD formulas. The RBF-FD method has originated in [1–5] and has been developed for various types of PDE problems during the last two decades (*e.g.* [6–8]). As in the classical finite difference (FD) methods, RBF-FD results in a sparse final matrix but with the added advantage that can naturally handle irregular geometries and scattered node layouts. Other applications and modifications can be found in [9–12].

In a more general form, a classical FD formula may consist of approximating some derivative of a function at a given point based on a linear combination of the function itself and its derivative values at some surrounding nodes. In this approach the symmetries can be exploited to increase the accuracy of the formula without increasing the stencil size. Such formulas were introduced by Collatz [13] and later developed into *compact* FD formulas by Lele [14] and Weinan and Liu [15]. In these schemes the improvement in accuracy is obtained by using additional information from the PDE itself, rather than increasing the stencil size as is the usual way to increase the accuracy in standard FD methods. An analogous compact RBF-FD formula was proposed by Wright and Fornberg in [16] by using the Hermite RBF interpolation method. This approach when applied on PDEs is called the RBF-HFD method. In [17] the RBF-HFD was developed for solving reaction-diffusion equations on surfaces.

The RBF-PU method for solving PDEs is essentially different from the RBF-FD although both approaches benefit from local RBF approxi-

[*] Corresponding author at: Division of Scientific Computing, Department of Information Technology, Uppsala University, Sweden.
E-mail addresses: d.mirzaei@sci.ui.ac.ir, davoud.mirzaei@it.uu.se (D. Mirzaei).

https://doi.org/10.1016/j.camwa.2022.09.029
Received 25 May 2022; Received in revised form 15 September 2022; Accepted 26 September 2022

mations in a same way. In RBF-PU a set of smooth PU weight functions are employed to blend the local approximants to obtain a smooth global solution. This can be mentioned as an advantage. However, the contribution of PU weights brings some difficulties to handle the derivatives, and results in a denser (but still sparse) final differentiation matrix compared to the RBF-FD method. Historically, a PU method was introduced by Shepard in 1968 [18] but its first combination with RBF interpolation goes back to [19] in 2002. The RBF-PU collocation method for solving transport equations on the unit sphere has been developed in [20]. A PU interpolation by product-type functions in two and three dimensions has been presented in [21], and an efficient computation through a block-based searching technique in [22]. A least squares RBF-PU method, an alternative to the standard collocation method, has been given in [23]. Authors of [24] have applied an optimized searching procedure to build stable bases in local PU patches for flat kernels. Positive constrained approximation via RBF-PU has been studied in [25], and optimal selection of local approximants in [26]. Adaptive algorithms based on RBF-PU for solving PDEs have been developed in [27–29]. Some applications in financial mathematics have been reported in [30–32]. The RBF-PU method has also been shown to be successful for a number of other PDE problems (*e.g.* [33–35]). Combination of the PU approximation with rational RBF interpolations has been presented in [36,37].

The *direct* RBF-PU (D-RBF-PU) method (see [38]) bypasses the calculation of PU weight derivatives and allows the use of some constant-generated weight functions to develop a faster algorithm than both RBF-FD and standard RBF-PU methods. In this paper, a Hermite interpolation based version of D-RBF-PU method is proposed for numerical solution of steady state PDEs. This method is connected to RBF-HFD but combination with the PU approach makes it faster in computing the stencil weights. In contrast to the RBF-HFD, in the new method a local approximant (a weight vector for a stencil) is not associated to just a single central point but is shared with a set of different test points in the local domain. This property speeds up the new algorithm. We employ the Hermite-Birkhoff RBF interpolation in PU patches and use two kinds of PU weights to join the local approximants to obtain a global solution. As we pointed out, the weight functions are not required to be differentiable, so we can implement a piecewise constant PU weight to obtain a sparse final differentiation matrix like as that of the standard RBF-HFD method. However, in the new method this matrix is set up using a much faster algorithm.

The paper is organized as follows. In sections 2 and 3 the RBF-FD and the RBF-HFD methods are reviewed. In section 4 the standard RBF-PU and D-RBF-PU methods are discussed. In section 5 the new compact D-RBF-PU method is proposed and its properties and advantages are explored. We also discuss how to use the polyharmonic spline (PHS) kernels in a stable way to obtain local approximants in PU patches. In section 7 we illustrate the effectiveness of the new method for solving some elliptic PDEs and discuss various implementation details. We conclude our paper with a summary and discussion of future research directions in section 8.

2. RBF-FD method

The RBF-FD method generalizes standard polynomial-based FD formulas for scattered nodes. Assume $\phi : \mathbb{R}^d \to \mathbb{R}$ is a conditionally positive definite function of order m, i.e., with respect to polynomials of degree $m-1$ in $\mathbb{R}^d$, denoted by $\mathbb{P}_{m-1}(\mathbb{R}^d)$. Let $\Omega \subset \mathbb{R}^d$ be an open and bounded region, and $\phi \in C^\ell(\Omega)$. Let $X_0 = \{x_1, x_2, \dots, x_n\} \subset \Omega$ be a localized set of scattered nodes around an evaluation point $x_0 \in \Omega$ that may or may not belong to X. For a multi-index α with $|\alpha| \leqslant \ell$, the value $D^\alpha u(x_0)$ can be approximated by a linear combination of u values at X_0:

$$Lu(x_0) \approx \sum_{j=1}^{n} c_j u(x_j) \tag{2.1}$$

where c_j are weight coefficients. In classical FD formulas the weight vector $\boldsymbol{c}$ is obtained by forcing (2.1) to be exact on polynomials up to a certain degree. There may be some degrees of freedom for looking for some norm-minimal weight vectors in some situations [39]. In RBF-FD formulas the weights are obtained by forcing the exactness on RBF space $\operatorname{span}\{\phi(\cdot - x_1), \dots, \phi(\cdot - x_n)\}$. To enrich the approximation we may consider the exactness on the space augmented with the polynomials of order m, i.e.,

$$\operatorname{span}\{\phi(\cdot - x_1), \dots, \phi(\cdot - x_n)\} + \mathbb{P}_{m-1}(\mathbb{R}^d).$$

More precisely, we write

$$u(x) \approx s(x) = \sum_{j=1}^{n} a_j \phi(x - x_j) + \sum_{k=1}^{Q} b_k p_k(x) \tag{2.2}$$

where $\{p_1, \dots, p_Q\}$ is a basis for $\mathbb{P}_{m-1}(\mathbb{R}^d)$ and $Q = \frac{(m-1+d)!}{d!(m-1)!}$. Then, coefficient vectors $(a_1, \dots, a_n)^T =: \boldsymbol{a}$ and $(b_1, \dots, b_Q)^T =: \boldsymbol{b}$ are determined through interpolation conditions $u(x_j) = s(x_j)$, $j = 1, \dots, n$ and side conditions

$$\sum_{j=1}^{n} a_j p_k(x_j) = 0, \quad k = 1, \dots, Q.$$

In matrix form for a given vector $\boldsymbol{u}_e = (u(x_1), \dots, u(x_n))^T$ we have

$$\begin{bmatrix} A & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_e \\ 0 \end{bmatrix},$$

where $A_{jk} = \phi(x_k - x_j)$, $j, k = 1, \dots, n$ and $P_{jk} = p_k(x_j)$, $k = 1, \dots, Q$, $j = 1, \dots, n$. If ϕ is a conditionally positive definite function of order m and X_0 is a $\mathbb{P}_m(\mathbb{R}^d)$-unisolvent set then the above system is uniquely solvable [40]. Coming back to (2.2) we have

$$\begin{aligned} D^\alpha u(x_0) &\approx D^\alpha s(x_0) \\ &= [D^\alpha \boldsymbol{\phi}^T(x_0) \;\; D^\alpha \boldsymbol{p}^T(x_0)] \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{bmatrix} \\ &= [D^\alpha \boldsymbol{\phi}^T(x_0) \;\; D^\alpha \boldsymbol{p}^T(x_0)] \begin{bmatrix} A & P \\ P^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{u}_e \\ \boldsymbol{0} \end{bmatrix} \\ &=: [D^\alpha \boldsymbol{\psi}^T(x_0) \;\; D^\alpha \boldsymbol{v}^T(x_0)] \begin{bmatrix} \boldsymbol{u}_e \\ \boldsymbol{0} \end{bmatrix} \\ &= D^\alpha \boldsymbol{\psi}^T(x_0) \boldsymbol{u}_e = \sum_{j=1}^{n} D^\alpha \psi_j(x_0) u(x_j) \end{aligned}$$

where

$$D^\alpha \boldsymbol{\phi}(x_0) = (D^\alpha \phi(x_1 - x_0), \dots, D^\alpha \phi(x_n - x_0))^T,$$

$$D^\alpha \boldsymbol{p}(x_0) = (D^\alpha p(x_1), \dots, D^\alpha p(x_n))^T,$$

and $(D^\alpha \psi_1, \dots, D^\alpha \psi_n)^T = D^\alpha \boldsymbol{\psi}$ and $(D^\alpha v_1, \dots, D^\alpha v_Q)^T = D^\alpha \boldsymbol{v}$ are vectors of Lagrange functions at $x = x_0$ satisfying

$$\begin{bmatrix} A & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} D^\alpha \boldsymbol{\psi}(x_0) \\ D^\alpha \boldsymbol{v}(x_0) \end{bmatrix} = \begin{bmatrix} D^\alpha \boldsymbol{\phi}(x_0) \\ D^\alpha \boldsymbol{p}(x_0) \end{bmatrix}. \tag{2.3}$$

This shows that in (2.1)

$$c_j = D^\alpha \psi_j(x_0), \quad j = 1, 2, \dots, n.$$

Let us describe how can this approach be used for numerical solution of a PDE problem on a domain Ω. First, we consider two well distributed discrete sets $X = \{x_1, \dots, x_N\} \subset \Omega \cup \partial\Omega$ and $Y = \{y_1, \dots, y_M\} \subset \Omega \cup \partial\Omega$ as trial and test points, respectively. Then, x_0 is set to y_k for $k = 1, \dots, M$, local sets (stencils) $X_k \subset X$ in neighborhood of y_k are formed, and RBF-FD weight vectors $\boldsymbol{c}_k$ corresponding to test points y_k are obtained. Expanding $\boldsymbol{c}_k$ by adding zero elements associated to trial points outside the stencil X_k and putting them in rows of a global matrix C of size $M \times N$, we obtain the RBF-FD approximation

$$(Lu)|_Y \approx C\boldsymbol{u}_e.$$

Thus, the RBF-FD discretization of a PDE $Lu = f$ for a given function f reads as

$$C\widehat{\boldsymbol{u}} = \boldsymbol{f} \tag{2.4}$$

where $\boldsymbol{f} = (f(y_1), \dots, f(y_M))^T$ and $\widehat{\boldsymbol{u}}$ is an approximation for the exact solution vector $\boldsymbol{u}_e$ when '≈' is replaced by '='. If all stencils X_k are $\mathbb{P}_{m-1}(\mathbb{R}^d)$-unisolvent then all local systems (2.3) associated to points $x_0 = y_k$ are solvable. However, the solvability of the global nonsymmetric (and possibly non-square) matrix C is not guaranteed at all. Although deficiency occurs in some rare situations, one can select a sufficiently large test point set Y compared to the trial set X to obtain an overdetermined full rank system and admit a least-squares solution $\widetilde{\boldsymbol{u}}$ instead of the exact solution $\widehat{\boldsymbol{u}}$ for (2.4). This strategy is called *overtesting* or *oversampling* [41–43].

RBF-FD has been widely used for solving various types of PDE problems in engineering and science. Besides, special efforts have been made to overcome the instability of local RBF system (2.4). We refer the reader to [7] and the references therein.

3. RBF-HFD method

The RBF Hermite-based FD (RBF-HFD) method is based on the following generalized interpolation. Let H be a Hilbert space and H^* be its dual. We assume that $\Lambda = \{\lambda_1, \dots, \lambda_n\} \subseteq H^*$ is a set of linearly independent functionals on H and $u_1, \dots, u_n \in \mathbb{R}$ are certain given values. A generalized recovery problem means to find a function $s \in H$ such that $\lambda_k(s) = u_k$, $k = 1, \dots, n$. Such s is called *generalized interpolant*. The norm-minimal generalized interpolant is defined by

$$s^* = \arg\min\{\|s\|_H : s \in H,\, \lambda_k(s) = u_k,\, k = 1, \dots, n\}, \tag{3.1}$$

where $\|\cdot\|_H$ is the norm on H. One can prove that (see for example [44, Thm. 16.1]) the unique solution of (3.1) is given by

$$s^* = \sum_{j=1}^{n} a_j v_j$$

where v_j are Riesz' representors of λ_j and a_j are obtained from the interpolation conditions $\lambda_k(s^*) = u_k$, $k = 1, \dots, n$, i.e., $A\boldsymbol{a} = \boldsymbol{u}$ where $A = (\lambda_k(v_j))_{k,j=1}^n$. Since the functionals are assumed to be linearly independent on H the system is solvable.

The Hermite-Birkhoff interpolation is a special case of the above generalized interpolation. For a given integer ℓ, we consider the set $\alpha^{(1)}, \dots, \alpha^{(n)} \in \mathbb{N}_0^d$ of multi-indices with $|\alpha^{(j)}| \le \ell$. Then we define the functionals $\lambda_k := \delta_{x_k} \circ D^{\alpha^{(k)}}$ and assume for two different indices $k \neq j$ either $x_k \neq x_j$ or $\alpha^{(k)} \neq \alpha^{(j)}$. Let $\phi \in C^{\ell}(\Omega)$ be a positive definite function and $H = \mathcal{N}_\phi(\Omega)$ be its associated native space. This means that

$$u(x) = \langle u, \phi(\cdot - x)\rangle_H, \quad \forall u \in H,\, \forall x \in \Omega$$

and thus for $j = 1, \dots, n$ we have

$$D^{\alpha^{(j)}} u(x_j) = \langle u, D_2^{\alpha^{(j)}} \phi(\cdot - x_j)\rangle_H, \quad \forall u \in H,$$

showing that $v_j = D_2^{\alpha^{(j)}} \phi(\cdot - x_j)$. We have used the subscript 2 on differential operators to mean that it is applied to ϕ with respect to its second argument. Later we will similarly use the subscript 1 to mean that the operator is applied to ϕ with respect to its first argument. The Hermite-Birkhoff interpolant of $u \in H$ is formed as

$$s(x) = \sum_{j=1}^{n} a_j D_2^{\alpha^{(j)}} \phi(x - x_j) \tag{3.2}$$

with interpolation conditions

$$D^{\alpha^{(k)}} s(x_k) = u_k, \quad k = 1, \dots, n,$$

which lead to the system of equations

$$A_{\phi,\Lambda}\boldsymbol{a} = \boldsymbol{u} \tag{3.3}$$

where $A_{\phi,\Lambda} = D_1^{\alpha^{(k)}} D_2^{\alpha^{(j)}} \phi(x - y)|_{x = x_k, y = x_j} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{u} = (u_1, \dots, u_n)^T$. Note that u_k values are coming from function u via $u_k = D^{\alpha^{(k)}} u(x_k)$. In this formulation, points x_k are not required to be distinct but if two points coincide then their corresponding derivative operators should be different.

Analogously to (2.2) we can enrich the expansion (3.2) by adding polynomial terms to obtain

$$s(x) = \sum_{j=1}^{n} a_j D_2^{\alpha^{(j)}} \phi(x - x_j) + \sum_{i=1}^{Q} b_i p_i(x), \tag{3.4}$$

with interpolation conditions

$$D^{\alpha^{(k)}} s(x_k) = u_k, \quad k = 1, \dots, n,$$

together with side conditions

$$\sum_{j=1}^{n} a_j D^{\alpha^{(j)}} p_k(x_j) = 0, \quad k = 1, \dots, Q.$$

The final linear system then reads as

$$\begin{bmatrix} A_{\phi,\Lambda} & P_\Lambda \\ P_\Lambda^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u} \\ 0 \end{bmatrix}, \tag{3.5}$$

where $P_\Lambda = \lambda_j(p_\ell) \in \mathbb{R}^{n\times Q}$. We can prove that if ϕ is a conditionally positive definite function of order m and the functionals $\lambda_k = \delta_{x_k} \circ D^{\alpha^{(k)}}$ are linearly independent and that $\lambda_k(p) = 0$ for all $k = 1, \dots, n$ and $p \in \mathbb{P}_{m-1}(\mathbb{R}^d)$ implies that $p = 0$, then the system (3.5) is uniquely solvable [44, Chap. 16].

In a RBF-HFD formula we are interested in approximating a value $D^\alpha u(x_0)$ for a multi-index α with $|\alpha| \leqslant \ell$ in terms of nodal values $u_j = D^{\alpha^{(j)}} u(x_j)$ as

$$D^\alpha u(x_0) \approx \sum_{j=1}^{n} c_j u_j = \sum_{j=1}^{n} c_j\, D^{\alpha^{(j)}} u(x_j).$$

The weight vector $\boldsymbol{c}$ can be obtained by operating D^α on both sides of (3.2) or (3.4) and using (3.3) or (3.5) for coefficients. Doing the same as previous section, the weight vector $\boldsymbol{c}$ should satisfy

$$A_{\phi,\Lambda}\boldsymbol{c} = D_1^\alpha D_2^\alpha \boldsymbol{\phi}(x_0)$$

without polynomial terms, and

$$\begin{bmatrix} A_{\phi,\Lambda} & P_\Lambda \\ P_\Lambda^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix} = \begin{bmatrix} D_1^\alpha D_2^\alpha \boldsymbol{\phi}(x_0) \\ D^\alpha \boldsymbol{p}(x_0) \end{bmatrix}, \tag{3.6}$$

with appended polynomials, where

$$D_1^\alpha D_2^\alpha \boldsymbol{\phi}(x_0) = (D_1^\alpha D_2^{\alpha^{(1)}} \phi(x_0 - x_1), \dots, D_1^\alpha D_2^{\alpha^{(n)}} \phi(x_0 - x_n))^T.$$

If fact, $\boldsymbol{c} = D_1^\alpha D_2^\alpha \boldsymbol{\psi}(x_0)$ and $\boldsymbol{d} = D^\alpha \boldsymbol{v}(x_0)$.

Now, assume that we are given the PDE $Lu = f$ on a domain Ω for a linear with constant coefficient operator L and a given function f. Let $X = \{x_1, \dots x_N\} \subset \Omega$ be a discrete set of trial points in Ω. To apply the Hermite-Birkhoff interpolation for this PDE, we are allowed to use only specific operators I (identity) and L in our expansion. Without loss of generality, assume that $X_0 = \{x_1, \dots, x_n\}$ is a stencil for a test point x_0. Assume that $\tilde{J} \subseteq \{1, 2, \dots, n\} =: J$ is an index family of size $\#\tilde{J} =: \widetilde{n}$, and $\widetilde{X}_0 := \{x_j : j \in \tilde{J}\} \subseteq X_0$. The RBF-HFD expansion is written as

$$\begin{aligned} Lu(x_0) &\approx \sum_{j\in J} c_j u(x_j) + \sum_{j\in \tilde{J}} \widetilde{c}_j Lu(x_j) \\ &= \sum_{j\in J} L\psi_j(x_0) u(x_j) + \sum_{j\in \tilde{J}} LL\widetilde{\psi}_j(x_0) Lu(x_j), \end{aligned} \tag{3.7}$$

where ψ_j and $\widetilde{\psi}_j$ are Lagrange functions on sets X_0 and $\widetilde{X}_0$, respectively. The vector of Lagrange functions or the weights vector $\begin{bmatrix} \boldsymbol{c} \\ \widetilde{\boldsymbol{c}} \end{bmatrix}$ is computed by solving the linear system

$$\begin{bmatrix} A & A_L^1 & P \\ A_L^2 & A_{LL} & P_L \\ P^T & P_L^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{c} \\ \widetilde{\boldsymbol{c}} \\ \boldsymbol{d} \end{bmatrix} = \begin{bmatrix} L\boldsymbol{\phi} \\ LL\widetilde{\boldsymbol{\phi}} \\ L\boldsymbol{p} \end{bmatrix} \tag{3.8}$$

where $A_{\phi,\Lambda} = \begin{bmatrix} A & A_L^1 \\ A_L^2 & A_{LL} \end{bmatrix}$ and $P_\Lambda = \begin{bmatrix} P \\ P_L \end{bmatrix}$ where

$$\begin{aligned} A &= (\phi(x_k - x_j)), && k, j \in J \\ A_L^1 &= (L_1\phi(x_k - x_j)), && k \in J,\ j \in \tilde{J}, \\ A_L^2 &= (L_2\phi(x_k - x_j)), && k \in \tilde{J},\ j \in J, \\ A_{LL} &= (L_1L_2\phi(x_k - x_j)), && k, j \in \tilde{J}, \\ P &= (p_k(x_j)), && j \in J,\ k = 1, \dots, Q, \\ P_L &= (Lp_k(x_j)), && j \in J,\ k = 1, \dots, Q. \end{aligned}$$

The right hand side vectors are

$$\begin{aligned} L\boldsymbol{\phi} &= (L_1\phi(x_0 - x_1), \dots, L_1\phi(x_0 - x_n))^T, \\ LL\widetilde{\boldsymbol{\phi}} &= (L_1L_2\phi(x_0 - x_{\tilde{j}_1}), \dots, L_1L_2\phi(x_0 - x_{\tilde{j}_{\tilde{n}}}))^T, \\ L\boldsymbol{p} &= (Lp_1(x_0), \dots, Lp_Q(x_0))^T. \end{aligned}$$

To obtain a discrete analogue of Lu via the RBF-HFD method we assume another set $Y = \{y_1, \dots, y_M\} \subset \Omega$ of test points, possibly different from trial set X. For any test point y_k we form its corresponding stencil $X_k \subset X$ and give the role of x_0 in the above formulation to y_k to obtain the weight vectors $\boldsymbol{c}_k$ and $\widetilde{\boldsymbol{c}}_k$ associated to stencil X_k. Expanding both $\boldsymbol{c}_k$ and $\widetilde{\boldsymbol{c}}_k$ by adding zeros and putting them in rows of global matrices C and $\widetilde{C}$, we obtain the RBF-HFD approximation

$$(Lu)|_Y \approx C\boldsymbol{u}_e + \widetilde{C}L\boldsymbol{u}_e,$$

where $L\boldsymbol{u}_e = (Lu(x_1), \dots, Lu(x_N))^T = (f(x_1), \dots, f(x_N))^T =: \boldsymbol{f}_e$ and $\boldsymbol{u}_e$ is defined as before. Thus, the RBF-HFD discretization of PDE $Lu = f$ for a given function f reads as

$$C\hat{\boldsymbol{u}} = \boldsymbol{f} - \widetilde{C}\boldsymbol{f}_e \tag{3.9}$$

where $\boldsymbol{u}_e$ is replaced by $\hat{\boldsymbol{u}}$ when the approximate symbol is replaced by the equality symbol. Comparing with system (2.4) of RBF-FD, the right hand side of (3.9) is corrected by known vector $-\widetilde{C}\boldsymbol{f}_e$.

The RBF-HFD or compact RBF-FD method was first proposed in [16], but the idea of generalized RBF interpolation goes back to [45,46]. See also [47–49] for analysis of this approach for PDEs in a global form.

4. Partition of unity methods

Partition of unity (PU) proposes another possibility to develop a class of localized approximation methods. It is important in the RBF context because global RBF systems are usually dense and ill-conditioned. To construct a PU setting, the global domain Ω should be covered by a set of open, bounded and overlapping patches $\{\Omega_1, \Omega_2, \dots, \Omega_{N_c}\}$ such that

$$\Omega \subset \bigcup_{\ell=1}^{N_c} \Omega_\ell.$$

This inclusion implies that every point $x \in \Omega$ is covered by at least one patch from the covering. Assume that $I(x) = \{\ell : x \in \Omega_\ell\}$. The covering is called *regular* if there exists a global constant K such that $|I(x)| \leqslant K$ for all $x \in \Omega$, i.e., every $x \in \Omega$ is covered by at most K patches.

In continuation, local approximants s_ℓ are constructed on each subdomain (patch) Ω_ℓ. They should be blended to generate a global (and usually smooth) solution s. For this purpose, we can choose a family of nonnegative functions $\{w_1, w_2, \dots, w_{N_c}\}$ such that

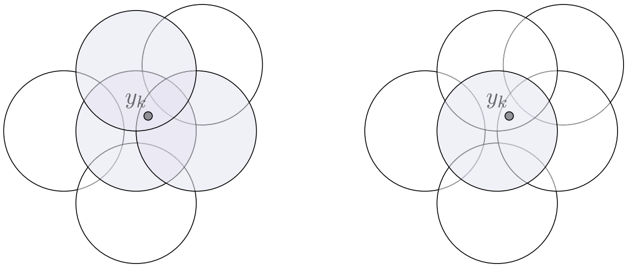


Fig. 1. Contributed patches at a test point y_k (shaded balls): for the smooth weight (4.2) and the discontinuous weight (4.3) (left), and for the discontinuous weight (4.4) (right).

(1) $\mathrm{supp}(w_\ell) \subseteq \Omega_\ell$,
(2) $\sum_{\ell=1}^{N_c} w_\ell(x) = 1, \quad \forall x \in \Omega.$

These functions are called PU weights. Having approximants s_ℓ and weights w_ℓ at hand, the global approximation s can be formed via

$$s(x) = \sum_{\ell=1}^{N_c} w_\ell(x)s_\ell(x), \quad x \in \Omega, \tag{4.1}$$

which is called a PU approximation.

Usually Shepard's functions are used as PU weights. For nonnegative, nonvanishing and compactly supported functions φ_ℓ on Ω_ℓ the Shepard weights are defined as

$$w_\ell(x) = \frac{\varphi_\ell(x)}{\sum_{j=1}^{N_c} \varphi_j(x)}, \quad 1 \le \ell \le N_c. \tag{4.2}$$

Clearly, these functions satisfy both required conditions for PU weights. In addition, w_ℓ are as smooth as generating functions φ_ℓ. Some discontinuous PU weights are also suggested in [38] that highly simplify the PU algorithms for solving partial differential equations. The first one is

$$w_\ell(x) = \begin{cases} \frac{1}{\#I(x)}, & x \in \Omega_\ell, \\ 0, & x \notin \Omega_\ell, \end{cases} \tag{4.3}$$

which is obtained from (4.2) via $\varphi_\ell(x) = \chi_{\Omega_\ell}(x)$, the characteristic function of set Ω_ℓ. Weight function (4.3) is obviously discontinuous and shares equal weights between patches Ω_ℓ for all $\ell \in I(x)$. Another discontinuous weight is defined by

$$w_\ell(x) = \begin{cases} 1, & \text{if } \Omega_\ell \text{ is the nearest patch to } x, \\ 0, & \text{otherwise}, \end{cases} \tag{4.4}$$

which gives the total weight 1 to the nearest patch and zero weights to the others. To identify the nearest patch we can measure and compare the distance between x and the center of all patches, i.e., if $\omega_1, \dots, \omega_{N_c}$ are patch centers then

$$\ell = \ell(k) = \mathrm{argmin}\|x_k - \omega_\ell\|_2$$

is the index of the nearest patch to test point $x = y_k$. See Fig. 1 for an illustration.

Assume that we are given a linear PDE problem $Lu = f$ on a domain $\Omega \subset \mathbb{R}^d$. The standard PU approach approximates Lu by Ls where s has representation (4.1), i.e.,

$$Lu \approx Ls = \sum_{\ell=1}^{N_c} L(w_\ell s_\ell). \tag{4.5}$$

In this approach L should operate on products $w_\ell s_\ell$. Moreover, only smooth weight functions are applicable. As an example, the Laplacian of u is approximated by

$$\Delta u \approx \Delta s = \sum_{\ell=1}^{N_c} (w_\ell \Delta s_\ell + 2\nabla w_\ell \cdot \nabla s_\ell + s_\ell \Delta w_\ell).$$

In this form the gradient and the Laplacian of both w_ℓ and s_ℓ need to be computed, where the computation of derivatives of w_ℓ in the Shepard form (4.2) sounds complicated. A simpler approach was suggested in [38] which employs PU weights to approximate Lu *directly* from its local approximants. More precisely,

$$Lu = \sum_{\ell=1}^{N_c} w_\ell s_\ell^L =: s^L, \tag{4.6}$$

where s_ℓ^L are local approximants of Lu in patches Ω_ℓ. A good candidate for s_ℓ^L is obviously Ls_ℓ. In the new approach derivatives of w_ℓ and lower derivatives of s_ℓ are not required at all. For example, the Laplacian of u is approximated by

$$\Delta u \approx s^\Delta = \sum_{\ell=1}^{N_c} w_\ell \Delta s_\ell.$$

Comparing with the standard approach, the second and third terms in front of the summation symbol are not present and the only first term is doing the whole job.

4.1. RBF-PU method

Let $X = \{x_1, x_2, \dots, x_N\}$ be a discrete set of trial centers in $\Omega \subset \mathbb{R}^d$ and let $X_\ell = X \cap \Omega_\ell$, $1 \le \ell \le N_c$. Assume further that J_ℓ is the collection of trial point indices in patch Ω_ℓ, i.e., $J_\ell := \{j \in \{1, \dots, N\} : x_j \in X_\ell\}$. If the local approximants s_ℓ are constructed from the approximation spaces

$$\mathrm{span}\{\phi(\cdot - x_j) : j \in J_\ell\} \oplus \mathbb{P}_{m-1}(\mathbb{R}^d), \quad 1 \le \ell \le N_c,$$

then the resulting PU method is called the RBF-PU method. If s_ℓ are interpolants of a function u on centers X_ℓ then

$$s_\ell(x) = \sum_{j \in J_\ell} \psi_j(\ell; x) u(x_j) \tag{4.7}$$

with Lagrange functions $\psi_j(\ell; \cdot)$, associated to centers X_ℓ, satisfying

$$\begin{bmatrix} A & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\psi}(\ell; x) \\ \boldsymbol{v}(\ell; x) \end{bmatrix} = \begin{bmatrix} \boldsymbol{\phi}(x) \\ \boldsymbol{p}(x) \end{bmatrix}.$$

The global interpolant s then reads as

$$s(x) = \sum_{\ell=1}^{N_c} \sum_{j \in J_\ell} \Big(w_\ell(x) \psi_j(\ell; x) \Big) u(x_j), \quad x \in \Omega. \tag{4.8}$$

Instead of looping over all $\ell \in \{1, \dots, N_c\}$ in the first summation we may loop over $\ell \in I(x)$ only, because $w_\ell(x) = 0$ if $\ell \notin I(x)$.

The RBF-PU discretization of $Lu = f$ on a discrete test set $Y = \{y_1, \dots, y_M\} \subset \Omega$ is obtained by approximating Lu by Ls at each point y_k based on standard approach (4.5). If s is the global RBF-PU interpolant of u then we have

$$Lu(y_k) \approx Ls(y_k) = \sum_{\ell \in I(y_k)} \sum_{j \in J_\ell} L\Big(w_\ell(x) \psi_j(\ell; x) \Big)_{x = y_k} u(x_j).$$

This leads to a linear system of equations of the form $C\hat{\boldsymbol{u}} = \boldsymbol{f}$ with

$$C_{kj} = \sum_{\ell \in I(y_k)} \Big(L(w_\ell \psi_j(\ell; \cdot)) \Big)(y_k), \quad y_k \in Y.$$

Again we remind that the PDE operator L should act on products $w_\ell \psi_j(\ell, \cdot)$ leading to some complicated calculations and algorithmic complexity especially when L contains some high order partial derivatives.

4.2. D-RBF-PU method

The D-RBF-PU method is based on the *direct* approximation (4.6) instead of the standard form (4.5). Local approximants $s_\ell^L = Ls_\ell$ are obtained from local RBF interpolations in patches Ω_ℓ as

$$s_\ell^L(x) = \sum_{j \in J_\ell} L\psi_j(\ell; x) u(x_j) \quad x \in \Omega_\ell \cap \Omega,$$

and the global approximation is derived as

$$s^L(x) = \sum_{\ell=1}^{N_c} \sum_{j \in J_\ell} \Big(w_\ell(x) L\psi_j(\ell; x) \Big) u(x_j) \tag{4.9}$$

for $x \in \Omega$. Therefore, the PDE $Lu = f$ can be discretized on test set Y via $C\hat{\boldsymbol{u}} = \boldsymbol{f}$ where

$$C_{kj} = \sum_{\ell \in I(y_k)} w_\ell(y_k) L\psi_j(\ell; y_k), \quad y_k \in Y.$$

Comparing with the standard RBF-PU, the PDE operator L acts only on Lagrange functions $\psi_\ell(\ell, \cdot)$ and not on PU weights w_ℓ. This will reduce the complexity of the algorithm but retain the consistency error, and allows using some discontinuous weights such as (4.3) and (4.4). The second discontinuous weight leads to a sparser final linear system. More details are given in [38].

5. Compact D-RBF-PU method

As we discussed, in RBF-HFD the accuracy of approximation of Lu is increased without expanding the stencil by adding new nodes. This approach borrows the idea from the standard compact finite differences [14,15], but now the stencil weights are not known a priori and should be obtained by solving lots of local linear systems of the form (3.6). This is a cost one should pay for working on scattered points and arbitrary geometries instead of working on a grid point set on a rectangular domain. In this section we propose a new implementation via D-RBF-PU method to relax this complexity, remarkably.

5.1. Compact D-RBF-PU formulation

For a trial set $X = \{x_1, \dots, x_N\}$ in the global domain Ω let X_ℓ and J_ℓ be defined as before. In addition, assume $\widetilde{X}_\ell \subseteq X_\ell$ is a subset of trial points in patches Ω_ℓ for $\ell = 1, \dots, N_c$, and $\widetilde{J}_\ell = \{j \in J_\ell : x_j \in \widetilde{X}_\ell\}$. Inspired from (3.7), the local approximants in our new D-RBF-PU method are obtained as

$$s_\ell^L(x) = \sum_{j \in J_\ell} L\psi_j(\ell; x) u(x_j) + \sum_{j \in \widetilde{J}_\ell} LL\widetilde{\psi}_j(\ell; x) Lu(x_j).$$

In contrast to RBF-HFD, we will apply this approximation not only for a single point $x = x_0$ but also for all evaluation points $x \in \Omega_\ell \cap \Omega$. The Lagrange functions are the solutions of linear systems

$$\begin{bmatrix} A & A_L & P \\ A_L & A_{LL} & P_L \\ P^T & P_L^T & 0 \end{bmatrix} \begin{bmatrix} L\boldsymbol{\psi}(\ell; x) \\ LL\widetilde{\boldsymbol{\psi}}(\ell; x) \\ L\boldsymbol{v}(x) \end{bmatrix} = \begin{bmatrix} L\boldsymbol{\phi}(\ell; x) \\ LL\widetilde{\boldsymbol{\phi}}(\ell; x) \\ L\boldsymbol{p}(x) \end{bmatrix}, \tag{5.1}$$

for $\ell = 1, 2, \dots, N_c$. Like as the right-hand side, the block matrix in (5.1) also depends on ℓ because the submatrices are based on sets X_ℓ and $\widetilde{X}_\ell$. The global approximation then is written as

$$\begin{aligned} s^L(x) &= \sum_{\ell=1}^{N_c} w_\ell(x) s_\ell^L(x) \\ &= \sum_{\ell=1}^{N_c} \sum_{j \in J_\ell} \Big(w_\ell(x) L\psi_j(\ell; x) \Big) u(x_j) \\ &\quad + \sum_{\ell=1}^{N_c} \sum_{j \in \widetilde{J}_\ell} \Big(w_\ell(x) LL\widetilde{\psi}_j(\ell; x) \Big) Lu(x_j). \end{aligned} \tag{5.2}$$

Thus, a discrete version of Lu on test points $Y=\{y_1,\dots,y_M\}$ is obtained as $(Lu)|_Y \approx s^L|_Y = C\boldsymbol{u}_e + \widetilde{C}L\boldsymbol{u}_e$ where

$$\begin{aligned} C_{kj} &= \sum_{\ell=1}^{N_c} w_\ell(y_k)L\psi_j(\ell;y_k),\\ \widetilde{C}_{kj} &= \sum_{\ell=1}^{N_c} w_\ell(y_k)LL\widetilde{\psi}_j(\ell;y_k), \end{aligned} \tag{5.3}$$

for $k=1,\dots,M$ and $j=1,\dots,N$. For PDE problem $Lu=f$ we have $C\boldsymbol{u}_e \approx \boldsymbol{f}-\widetilde{C}L\boldsymbol{u}_e = \boldsymbol{f}-\widetilde{C}\boldsymbol{f}_e$. If we replace '≈' by '=' and $\boldsymbol{u}_e$ by an approximate vector $\hat{\boldsymbol{u}}$, we end with

$$C\hat{\boldsymbol{u}} = \boldsymbol{f}-\widetilde{C}\boldsymbol{f}_e,$$

which looks the same as system (3.9) of the RBF-HFD method but with different matrices C and $\widetilde{C}$. We highlight that in the new D-RBF-PU method the number of local linear systems to be solved for setting up the final matrices C and $\widetilde{C}$ is highly reduced form M (number of test points) to N_c (number of PU patches) which leads to a considerably reduction in the computational cost of the method for setting up the final linear system.

If the standard Shepard weights (4.2) are used the stencil of a test point y_k is the union of all X_ℓ for $\ell\in I(y_k)$ which is larger than the single stencil X_k of the standard RBF-HFD method. Thus, the final matrix of D-RBF-PU is denser than that of the RBF-HFD. The situation can be modified if the constant-generated weight function (4.4) is applied. In this case every test point y_k is subjected to a single stencil (the closet stencil) which makes this version of D-RBF-PU as sparse as RBF-HFD. Let us describe it in a more detail. According to the PU procedure, each point y_k is subjected to a local set $X_\ell=\Omega_\ell\cap X$ for an index $\ell=\ell(k)\in\{1,\dots,N_c\}$ in which $\Omega_{\ell(k)}$ is the closest patch to y_k. From (4.4) and (5.2) we have

$$\begin{aligned} s^L(y_k) &= \sum_{\ell=1}^{N_c}\sum_{j\in J_\ell} w_\ell(y_k)L\psi_j(\ell;y_k)u(x_j)\\ &\quad+\sum_{\ell=1}^{N_c}\sum_{j\in \tilde{J}_\ell} w_\ell(y_k)LL\widetilde{\psi}_j(\ell;y_k)Lu(x_j)\\ &=\sum_{j\in J_{\ell(k)}} L\psi_j(\ell(k);y_k)u(x_j)+\sum_{j\in\tilde{J}_{\ell(k)}} LL\widetilde{\psi}_j(\ell(k);y_k)Lu(x_j), \end{aligned}$$

which leads to

$$\begin{aligned} C_{kj} &= L\psi_j(\ell(k);y_k),\\ \widetilde{C}_{kj} &= LL\widetilde{\psi}_j(\ell(k);y_k), \end{aligned} \tag{5.4}$$

for $k=1,\dots,M$ and $j=1,\dots,N$. The difference between this approach and the RBF-HFD method is that in the new method a set (stencil) X_ℓ is shared with many points y_k while in the RBF-HFD each stencil X_k is associated to a unique test point y_k.

In summary, the compact D-RBF-PU method with either smooth or constant-generated weights is much faster than the RBF-HFD method for setting up the final differentiation matrices. With the smooth weight function the final matrix of D-RBF-PU is denser but the majority of the total cost is subjected to the setting up phase because the final systems can be effectively inverted using iterative linear algebra solvers in few seconds, even for large values of N. See the experimental results of section 7.

5.2. How to avoid the instability of local systems?

The standard implementation of all above localized techniques suffers from the ill-conditioning of local linear systems. One of sources of instability is the shape parameter, where keeping it small creates an ill-conditioned kernel matrix. Several attempts have been done to resolve this problem. Some attentions have been paid to find an "optimal" shape parameter instead of choosing it by trial and error. The earliest work goes back to Hardy [50] and then to Franke [51]. A variation of the leave-one-out cross validation (LOOCV) forms the basis of an algorithm proposed in [52]. The method of maximum likelihood estimation (MLE) was applied as an alternative algorithm for choosing the shape parameter in [53]. The list of papers for choosing an optimal shape parameter in RBF approximation is long. As some examples, we refer to [54–57] to see some extensions of the Rippa's algorithm.

A different approach was suggested by Wright in his Ph.D thesis [5] which computes the solution at different values of shape parameter on a safe path in the complex plane and then approximates the solution at the desired positive shape parameter using the Padè approximation. The approach is called the RBF Contour Padè (RBF-CP) method. In [58] the RBF-CP was improved to a more efficient technique RBF-RA where the rational interpolation is used instead of the Padè approximation. The RBF-HFD has equipped with RBF-CP in [16] and with RBF-RA in [58] where in both cases the stability at near flat cases is obtained at the price of a more computational cost.

For standard interpolation the RBF-QR [59,8] and the RBF-GA [60] methods suggest an alternative way to suppress the instability of Gaussian kernel at small values of shape parameter. Unfortunately, these approaches as well as other stabilization techniques (*e.g.* see [61,62]) are not easily adaptable to Hermite interpolation in the form introduced here.

The spectral convergence of infinitely smooth kernels such as Gaussians and (inverse) multiquadrics for a global RBF interpolation will reduce to an algebraic rate in a localized RBF form. This may encourage users to get use of finitely smooth and shape parameter free basis functions such as *polyharmonic spline* (PHS) kernels. These kernels are defined as

$$\varphi_\beta(r) := \begin{cases} r^\beta\log r, & \beta \text{ even}\\ r^\beta, & \text{otherwise} \end{cases} \tag{5.5}$$

for a positive real number β. The function $\phi=\varphi_\beta(\|\cdot\|_2)$ is conditionally positive definite of order $m=\lfloor\beta/2\rfloor+1$. However, the instability has another source other than working in a small shape parameter regime. When the spacing distance between the nodes decreases the condition number of the RBF system increases algebraically for finitely smooth or exponentially for infinitely smooth basis functions [44]. This kind of instability could be fairly overcome for PHS kernels because the approximation process with a PHS kernel is *scalable* meaning that the Lagrange functions $\psi_j^h(x^h)$ on a localized set X^h with spacing distance h are identical with the Lagrange functions $\psi_j(x)$ on scaled (blown up) set $X=X^h/h$ with fill distance 1 for $x=x^h/h$ [63]. More generally, for an order α derivative we have

$$D^\alpha\psi_j^h(x^h)=h^{-|\alpha|}D^\alpha\psi_j(x). \tag{5.6}$$

This means that we can always compute the stencil weights on the blown up set and then rescale them to the original situation using the scaling rule (5.6). Note that, the conditioning of the PHS system in the blown up set behaves as $\mathcal{O}(1)$. For some applications in numerical solution of PDEs see [64,65,38].

Now, we show that the scaling works for local systems (5.1) for the new compact D-RBF-PU method as well. Assume that L is a linear functional with scaling order (homogeneity) σ, i.e., $Lu(\cdot h)=h^\sigma Lu$ for all $h>0$. For example D^α, ∇ and Δ have scaling orders $\sigma=|\alpha|$, $\sigma=1$ and $\sigma=2$, respectively. Then we can prove that if the system (5.1) is formed via PHS kernels (5.5) on a data set X^h with fill-distance h and monomial basis functions $\{x^\alpha\}_{|\alpha|<m}$ for $\mathbb{P}_{m-1}(\mathbb{R}^d)$ then

$$\begin{aligned} h^\sigma L\boldsymbol{\psi}^h(\ell;x^h) &= L\boldsymbol{\psi}(\ell;x),\\ LL\widetilde{\boldsymbol{\psi}}^h(\ell;x^h) &= LL\widetilde{\boldsymbol{\psi}}(\ell;x),\\ h^{\sigma-\beta}HL\boldsymbol{v}^h(x^h) &= L\boldsymbol{v}(\ell;x) \end{aligned} \tag{5.7}$$

where the right hand side terms are solutions of (5.1) on the blown up set $X = X^h/h$ at evaluation point $x = x^h/h$. In the third line H is the following diagonal matrix

$$H = \operatorname{diag}\Big\{1,\ \underbrace{h,\dots,h}_{\binom{d}{d-1}\text{ times}}\ ,\underbrace{h^2,\dots,h^2}_{\binom{d+1}{d-1}\text{ times}},\dots,\underbrace{h^{m-1},\dots,h^{m-1}}_{\binom{d+m-2}{d-1}\text{ times}}\Big\}\in\mathbb{R}^{Q\times Q}.$$

To form H we used the fact that $\#\{x^\alpha : |\alpha| = k\} = \binom{d+k-1}{d-1}$ for $k = 0,1,\dots,m-1$. The scaling rules (5.7) can be simply proved. For example, in the case of power kernel $\varphi_\beta(r) = r^\beta$ the relation between blocks of the Hermite matrix in (5.1) in original and blown up cases are

$$A^h = h^\beta A, \quad A^h_L = h^{\beta-\sigma} A_L, \quad A^h_{LL} = h^{\beta-2\sigma} A_{LL},$$

$$P^h = PH, \quad P^h_L = h^{-\sigma} P_L H,$$

and the right hand is scaled via

$$L\boldsymbol{\phi}^h(\ell; x^h) = h^{\beta-\sigma} L\boldsymbol{\phi}(\ell; x),\ \ LL\widetilde{\boldsymbol{\phi}}^h(\ell; x^h) = h^{\beta-2\sigma} LL\widetilde{\boldsymbol{\phi}}(\ell; x),$$

$$L\boldsymbol{p}^h(x^h) = h^{-\sigma} H L\boldsymbol{p}(x).$$

These simply proves (5.7). The proof for the case of TPS function $\varphi_\beta(r) = r^\beta \log r$ follows a same direction but needs the vanishing property of the quadratic forms of polynomials of certain degrees which is true for appended polynomials to the TPS kernel.

This scaling property allows to form the local linear system (5.1) on a scaled data set with spacing distance 1 to avoid any serious instability. Then the scaling rules (5.7) are used to compute the Lagrange functions of the original configuration. In fact, we take out the h from data, solve the problem stably, and finally bring it back to the solution via (5.7).

Unless the other stabilization techniques, it is clear that this approach does not impose any additional cost to the algorithm. We also note that the same scaling rule is applicable to obtain the stencil weights from system (3.8) for the standard RBF-HFD method.

6. Implementation details

In this section we provide implementation details to make the method easier to follow and more convenient to extent. Assume that a PDE problem

$$Lu = f, \text{ in } \Omega$$

$$Bu = g, \text{ on } \partial\Omega$$

is given where L and B are linear domain and boundary differential operator, respectively. Let X be a set of N trial points in $\Omega \cup \partial\Omega$ and $X = X_I \cup X_B$ where X_I is the set of interior points and X_B is the set of boundary points. Assume that $\boldsymbol{f} = f|_{X_I}$ and $\boldsymbol{g} = g|_{X_B}$. Then the main algorithm can be written as follows.

```
Algorithm 1: The main PDE solver
Data: Trial set $X$, interior set $X_I$, boundary set $X_B$, patches $\{\Omega_\ell\}$, PDE operators
      $L$ and $B$, and PDE data $\boldsymbol{f}$ and $\boldsymbol{g}$.
Result: Approximate solution $\boldsymbol{u}$ at points $X$.
1: Call Algorithm 2 for operator $L$ to compute matrices $C_L$ and $\widetilde{C}_L$ from data $X$,
   $X_I$ and $\{\Omega_\ell\}$;
2: Call Algorithm 2 for operator $B$ to compute matrices $C_B$ and $\widetilde{C}_B$ from data $X$,
   $X_B$ and $\{\Omega_\ell\}$;
3: Set $C = \begin{bmatrix} C_L \\ C_B \end{bmatrix}$ and $\boldsymbol{b} = \begin{bmatrix} (I-\widetilde{C}_L)\boldsymbol{f} \\ (I-\widetilde{C}_B)\boldsymbol{g} \end{bmatrix}$;
4: Solve $C\boldsymbol{u} = \boldsymbol{b}$;
5: Return $\boldsymbol{u}$;
```

In Algorithm 1, the compact D-RBF-PU algorithm (Algorithm 2) is called twice (for interior operator L and boundary operator B) to setup the final matrix C and the right-hand side vector $\boldsymbol{b}$. This subroutine reads as follows.

```
Algorithm 2: The compact D-RBF-PU subroutine
Data: Trial set $X$ of size $m\times d$, test set $Y$ of size $n\times d$, patches $\{\Omega_\ell\}$, and
      operator $L$.
Result: Matrix $C$ of size $m\times n$ and matrix $\widetilde{C}$ of size $n\times n$.
1: Initialize: $C = 0$ and $\widetilde{C} = 0$;
2: Collect indices of points in $X$ belonging to each patch $\Omega_\ell$ using the $kd$-tree
   algorithm;
3: Collect indices of points in $Y$ belonging to each patch $\Omega_\ell$ using the $kd$-tree
   algorithm;
4: for $\ell = 1$ : Number of patches do
    4-1: Find trial set $X_\ell$ in patch $\Omega_\ell$ using the corresponding collection of line
      2;
    4-2: Find test set $Y_\ell$ in patch $\Omega_\ell$ using the corresponding collection of line 3;
    4-3: Find Hermite set $\widetilde{X}_\ell$ out of $Y_\ell$;
    if weight is constant-generated then
        4-4: Find points $\widehat{Y}_\ell$ out of $Y_\ell$ for which $\Omega_\ell$ is their closet patch;
        4-5: Call Algorithm 3 to compute Lagrange functions based on $X_\ell$ and
          $\widetilde{X}_\ell$ at $\widehat{Y}_\ell$;
        4-6: Update $C$ and $\widetilde{C}$ via (5.4).
    else
        4-4: Compute Shepard weight vector $\boldsymbol{w}_\ell$ at $Y_\ell$ via (4.2);
        4-5: Call Algorithm 3 to compute Lagrange functions based on $X_\ell$ and
          $\widetilde{X}_\ell$ at $Y_\ell$;
        4-6: Update $C$ and $\widetilde{C}$ via (5.3).
5: Return $C$ and $\widetilde{C}$.
```

```
Algorithm 3: PHS Lagrange functions
Data: Point sets $X$ of size $m\times d$, $\widetilde{X}$ of size $\tilde{n}\times d$, $Y$ of size $n\times d$, the center $\omega_\ell$
      and radius $\rho$ of patch $\Omega_\ell$, and operator $L$.
Result: Matrix $\Psi$ of size $n\times m$ and matrix $\widetilde{\Psi}$ of size $n\times\tilde{n}$.
1: Shift $X$, $\widetilde{X}$ and $Y$ by $\omega_\ell$ and divide by $\rho$ (shift and scale);
2: Compute matrices $A$, $A_L$, $A_{LL}$, $P$, $P_L$, $L\boldsymbol{\phi}(\ell;\cdot)$, $LL\widetilde{\boldsymbol{\phi}}(\ell;\cdot)$, $L\boldsymbol{p}(x)$ and form
   (5.1);
3: Solve (5.1) for $\Psi$ and $\widetilde{\Psi}$;
4: Determine $s$ the scaling order of $L$;
5: Re-scale $\Psi$ and $\widetilde{\Psi}$ using the scaling rule (5.7);
6: Return the new $\Psi$ and $\widetilde{\Psi}$;
```

Finally, we note that the MATLAB implementation is freely available at GitHub via https://github.com/ddmirzaei/C_D_RBF_PU to facilitate the reproduction of the examples presented in the next section.

7. Numerical experiments

In this section, numerical results of the compact D-RBF-PU method and comparisons with the RBF-HFD method are given. We consider the Poisson equation on the unit square $\Omega = [0,1]^2$ with Dirichlet and Neumann boundary conditions in two dimension. The well-known Franke's function is used as an exact solution [51]. Halton sets $X = \{x_1,\dots,x_N\}$ for different number N with fill distance $h = h_{X,\Omega}$ are used for trial points. The fill distance is approximated by $1/\sqrt{N}$. We use a set of circular patches $\Omega_\ell = B(\omega_\ell, \rho)$ for PU approximation. In all examples, $\rho = 4h$, being h the fill distance, and $N_c = \lceil N/16 \rceil$ grid points in the domain are used for patch centers. We assume $X_\ell = \{x_j \in X : \|x_j - \omega_\ell\|_2 \leqslant \rho\}$ and $\widetilde{X}_\ell = \{x_j \in X_\ell : c_H\rho < \|x_j - \omega_\ell\|_2 \leqslant \rho\}$ where $0 \leqslant c_H \leqslant 1$ controls the width of the annulus. Values of c_H close to 1 put the points in sub-stencil $\widetilde{X}_\ell$ near the boundary of patch Ω_ℓ. Experiments show more satisfactory results for values of c_H bigger than 0.5. In this study we set $c_H = 0.75$ for internal points and $c_H = 1$ for boundary test points. As a smooth PU weight, the C^2 Wendland's function

$$\varphi_\ell = \varphi(\|\cdot - \omega_\ell\|_2/\rho), \quad \varphi(r) = (1-r)_+^4(4r+1),$$

is used in (4.2) [40]. The constant-generated weight function (4.4) will be applied as well. PHS kernel (5.5) with $\beta = 4,5,6,7,8,9$ where we refer to as PHSβ are employed. Polynomials of degree $\lfloor \beta/2 \rfloor = 2,2,3,3,4,4$ are augmented, respectively, to guarantee the solvability of local systems. The scaling rule (5.7) is applied in all patches for computing local RBF approximations. The results will be compared with the RBF-HFD

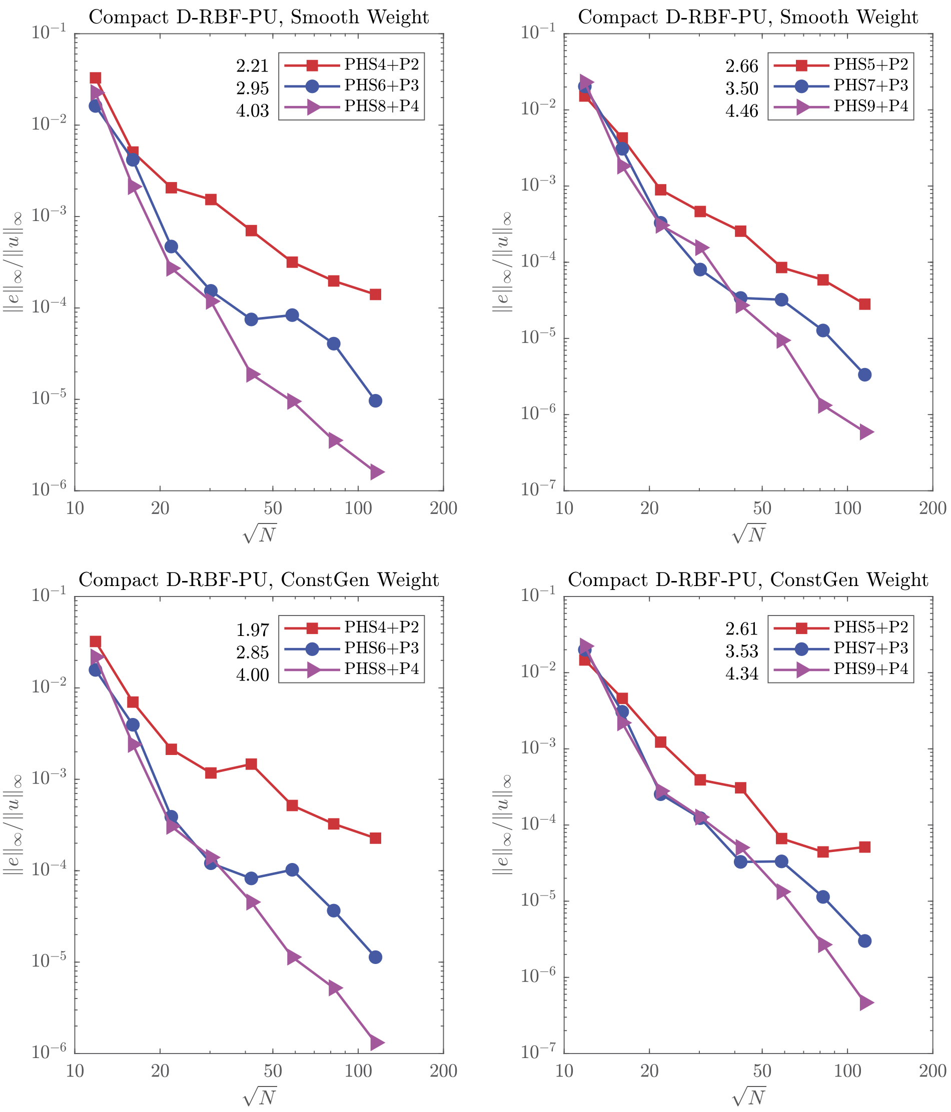


Fig. 2. Errors and convergence orders of compact D-RBF-PU method with different PHS kernels and polynomial degrees. Computational convergence orders are appeared along side the legends. The smooth PU weight is used for the plots of the first row and the constant-generate weight for those on the second row. These results are obtained for a pure Dirichlet BVP.

method. For a fair comparison the RBF-HFD stencils are chosen to be the intersection of trial points with balls of radius ρ centered at test points. Also, the same sub-stencils are used for Hermite part of the approximation in each stencil. The scaling rule is used for computing the RBF-HFD weights as well. In all plots computational convergence orders are obtained by the linear least squares fitting to error values and are written alongside the figure legends.

In Fig. 2, the norm infinity of errors in terms of $\sqrt{N}$ (the square of number of trial points) and numerical convergence orders are shown. We observe the convergence order $\beta/2$ for each kernel PHSβ. For results of Fig. 2, the Dirichlet boundary condition is applied on all sides of the domain's boundary. Fig. 3 presents the errors and orders when the Neumann boundary condition is imposed on top and bottom sides of the boundary. Comparing with the previous figure, we observe a reduction in accuracy which is natural in appearance of Neumann boundary conditions.

In Fig. 4 the sparsity of final matrices of compact D-RBF-PU and RBF-HFD methods are compared for two different patch sizes. As N increases the percentage of nonzeros goes down to fall under 1% for $N \gtrsim 10\text{e}+4$. The final matrix of D-RBF-PU with the constant-generated weight is approximately as sparse as that of the RBF-HFD method. The smooth weight results in an approximately 3 times denser matrix. This is the cost one should pay to obtain a smooth solution for the PDE.

In Fig. 5, a comparison between the new method and the RBF-HFD method is given for both types of boundary conditions. For the Dirichlet problem both methods provide approximately the same accuracy and order of convergence but for the Neumann problem the new method outperforms the RBF-HFD method. However, the main advantage of the compact D-RBF-PU over the RBF-HFD is reported in Fig. 6 where the computational costs are compared. Although the final matrix of D-RBF-PU with smooth weight is denser than two others, the left panel shows that the CPU time needed for solving the final system in each cases is a fraction of second for values of N up to 15000 in this experiment. We observe a remarkable difference in the right panel where the setting up times are compared for both methods. The results show an average speedup of 10x for the new method. This speedup is caused by the costs for solving local linear systems. If we ask for a more accurate solution using a wider stencil (or patch) then a higher speedup with the new method is observed.

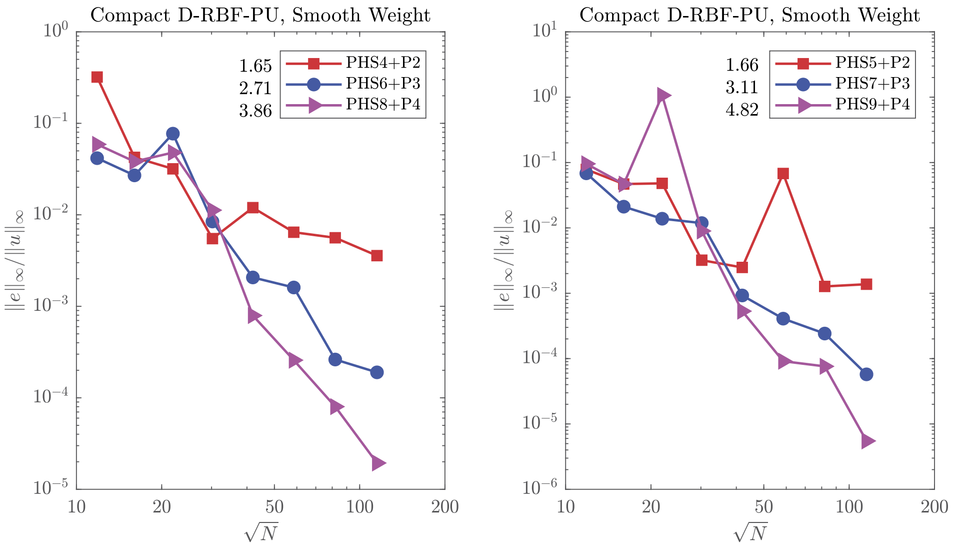


Fig. 3. Errors and convergence orders of compact D-RBF-PU method with different PHS kernels and polynomial degrees for Neumann BVP. Computational convergence orders are appeared along side the legends. The smooth PU weight is used in this case.

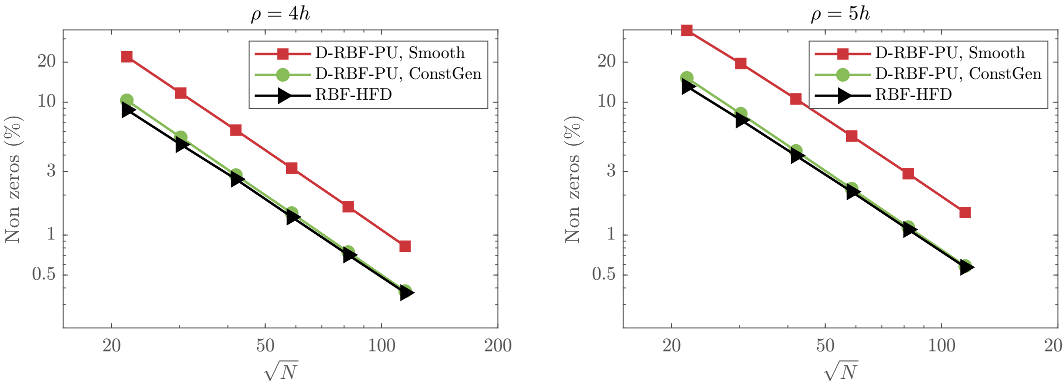


Fig. 4. Amount of sparsity: a comparison between the new method with smooth and discontinuous weights and the RBF-HFD method. The left panel for $\rho = 4h$ and the right panel for $\rho = 5h$.

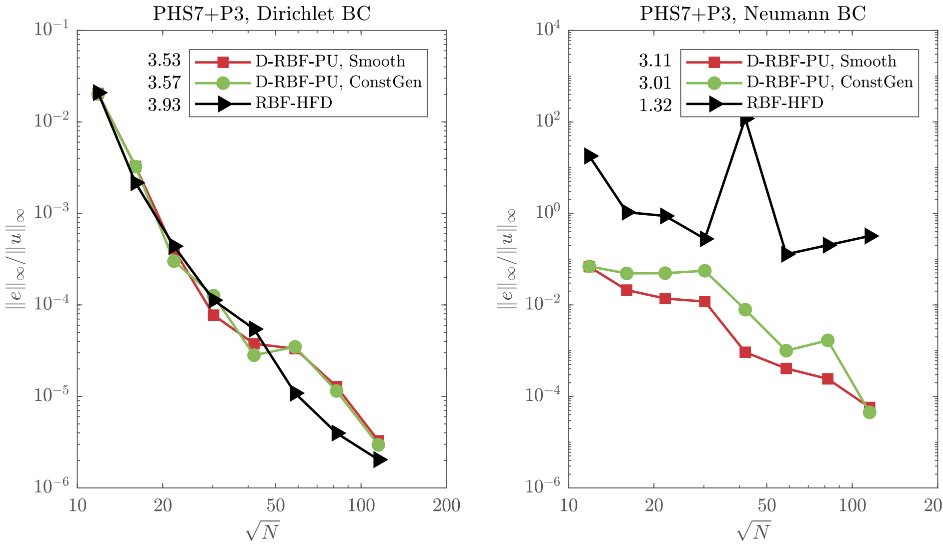


Fig. 5. Comparison of accuracy and convergence orders: Compact D-RBF-PU method v.s RBF-HFD method with Dirichlet (left) and Neumann (right) boundary conditions.

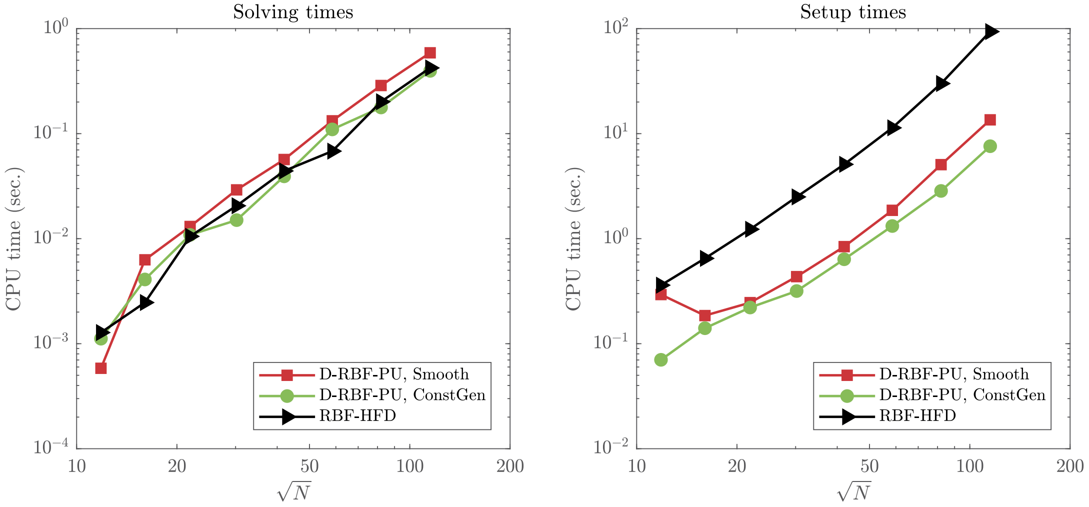


Fig. 6. The CPU times used for solving (left) and setting up (right) the final linear systems at different number of points. The rates are the same but D-RBF-PU is approximately 10 times faster.

8. Conclusion

We developed the standard RBF-HFD method into a new technique based on a direct RBF approximation and the partition of unity method. The new method, called the compact direct RBF partition of unity (D-RBF-PU) method, possesses some advantages over the standard method. This approach leads to a new RBF-HFD formulation in a partition of unity setting, still gives rise to sparse differentiation matrices. We investigated an scalability property of polyharmonic spline kernels to develop a simple and stable algorithm for solving local linear systems in each PU patches. The scaling is also applicable for obtaining the stencil weights of the RBF-HFD method. We also illustrated the efficiency and applicability of the new method through some numerical examples. Compared with the standard RBF-HFD method, the compact D-RBF-PU method is much faster and results in more accurate solutions in appearance of Neumann boundary conditions. In our experiments with a certain selection of method parameters the speedup of 10x was observed.

Finally, the use of the new compact D-RBF-PU method with the method of line (MOL) to extend it to time-dependent PDEs such as reaction-diffusion and advection-diffusion problems is left for a future study.

Data availability

No data was used for the research described in the article.

Acknowledgements

The second author was in part supported by a grant from IPM, No. 1401650417 and by a grant form INSF, No. 98012657. We greatly acknowledge the helpful comments of anonymous reviewers.

References

[1] T. Cecil, J. Qian, S. Osher, Numerical methods for high dimensional Hamilton-Jacobi equations using radial basis functions, J. Comput. Phys. 196 (2004) 327–347.
[2] C. Shu, H. Ding, K.S. Yeo, Local radial basis function-based differential quadrature method and its application to solve two-dimensional incompressible Navier-Stokes equations, Comput. Methods Appl. Mech. Eng. 192 (2003) 941–954.
[3] 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, pp. 4606–4624.
[4] A.E. Tolstykh, On using radial basis functions in a "finite difference mode" with applications to elasticity problems, Comput. Mech. 33 (2003) 68–79.
[5] G.B. Wright, Radial basis function interpolation: Numerical and analytical developments, Ph.D. thesis, University of Colorado, Boulder, 2003.
[6] B. Fornberg, C. Piret, A stable algorithm for flat radial basis functions on a sphere, SIAM J. Sci. Comput. 30 (2007) 60–80.
[7] B. Fornberg, N. Flyer, Solving PDEs with radial basis functions, Acta Numer. (2015) 215–258.
[8] E. Larsson, E. Lehto, A. Heryudono, B. Fornberg, Stable computation of differentiation matrices and scattered node stencils based on Gaussian radial basis functions, SIAM J. Sci. Comput. 35 (2013) A2096–A2119.
[9] M. Jabalameli, D. Mirzaei, A weak-form RBF-generated finite difference method, Comput. Math. Appl. 79 (2020) 2624–2643.
[10] V. Shankar, The overlapped radial basis function-finite difference (RBF-FD) method: a generalization of RBF-FD, J. Comput. Phys. 342 (2017) 211–228.
[11] V. Shankar, A. Narayan, R.M. Kirby, RBF-LOI: augmenting radial basis functions (RBFs) with least orthogonal interpolation (LOI) for solving PDEs on surfaces, J. Comput. Phys. 373 (2018) 722–735.
[12] V. Shankar, G.B. Wright, A. Narayan, A robust hyperviscosity formulation for stable RBF-FD discretizations of advection-diffusion-reaction equations on manifolds, SIAM J. Sci. Comput. 42 (2020) A2371–A2401.
[13] L. Collatz, The Numerical Treatment of Differential Equation, Springer Verlag, 1960.
[14] S.K. Lele, Compact finite difference schemes with spectral-like resolution, J. Comput. Phys. 103 (1992) 16–42.
[15] E. Weinan, J.G. Liu, Essentially compact schemes for unsteady viscous incompressible flows, J. Comput. Phys. 126 (1996) 122–138.
[16] G.B. Wright, B. Fornberg, Scattered node compact finite difference-type formulas generated from radial basis functions, J. Comput. Phys. 212 (2006) 99–123.
[17] E. Lehto, V. Schankar, G.B. Wright, A radial basis function (RBF) compact finite difference (FD) scheme for reaction-diffusion equations on surfaces, SIAM J. Sci. Comput. 39 (2017) A2129–A2151.
[18] D. Shepard, A two-dimensional interpolation function for irregularly-spaced data, in: Proceedings of the 23th National Conference ACM, 1968, pp. 517–523.
[19] H. Wendland, Fast evaluation of radial basis functions: methods based on partition of unity, in: Approximation Theory, X: Wavelets, Splines, and Applications, Vanderbilt University Press, Nashville, TN, 2002, pp. 473–483.
[20] K. Aiton, A radial basis function partition of unity method for transport on the sphere, Master's thesis, Boise State University, Boise, Idaho, 2014.
[21] R. Cavoretto, Two and three dimensional partition of unity interpolation by product-type functions, Appl. Math. Inf. Sci. 9 (2015) 1–8.
[22] R. Cavoretto, A. De Rossi, E. Perracchione, Efficient computation of partition of unity interpolants through a block-based searching technique, Comput. Math. Appl. 71 (2016) 2568–2584.
[23] E. Larsson, V. Shcherbakov, A. Heryudono, A least squares radial basis function partition of unity method for solving PDEs, SIAM J. Sci. Comput. 39 (2017) A2538–A2563.
[24] R. Cavoretto, A. De Marchi, S. De Rossi, G. Santin, Partition of unity interpolation using stable kernel-based techniques, Appl. Numer. Math. 116 (2017) 95–107.
[25] A. De Rossi, E. Perracchione, Positive constrained approximation via RBF-based partition of unity method, J. Comput. Appl. Math. 319 (2017) 338–351.
[26] R. Cavoretto, A. De Rossi, E. Perracchione, Optimal selection of local approximants in RBF-PU interpolation, J. Sci. Comput. 74 (2018) 1–22.
[27] R. Cavoretto, A. De Rossi, Adaptive meshless refinement schemes for RBF-PUM collocation, Appl. Math. Lett. 90 (2019) 131–138.

[28] R. Cavoretto, A. De Rossi, Error indicators and refinement strategies for solving Poisson problems through a RBF partition of unity collocation scheme, Appl. Math. Comput. 369 (2020) 124824.

[29] S. De Marchi, A. Martínez, E. Perracchione, M. Rossini, RBF-based partition of unity methods for elliptic PDEs: adaptivity and stability issues via variably scaled kernels, J. Sci. Comput. 79 (1) (2019) 321–344.

[30] A. Safdari-Vaighani, A. Heryudono, E. Larsson, A radial basis function partition of unity collocation method for convection—diffusion equations arising in financial applications, J. Sci. Comput. 64 (2) (2015) 341–367.

[31] V. Shcherbakov, Radial basis function partition of unity operator splitting method for pricing multi-asset American options, BIT Numer. Math. 56 (2016) 1401–1423.

[32] V. Shcherbakov, E. Larsson, Radial basis function partition of unity methods for pricing Vanilla basket options, Comput. Math. Appl. 71 (1) (2016) 185–200.

[33] M.R.A. Darani, The RBF partition of unity method for solving the Klein-Gordon equation, Eng. Comput. 37 (2021) 795.

[34] G. Garmanjani, R. Cavoretto, M. Esmaeilbeigi, A RBF partition of unity collocation method based on finite difference for initial-boundary value problems, Comput. Math. Appl. 75 (2018) 4066–4090.

[35] R. Mollapourasl, A. Fereshtian, M. Vanmaele, Radial basis functions with partition of unity method for American options with stochastic volatility, Comput. Econ. 53 (2019) 259–287.

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

[37] E. Farazandeh, D. Mirzaei, A rational rbf interpolation with conditionally positive kernels, Adv. Comput. Math. 47 (2021) 74.

[38] 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.

[39] O. Davydov, R. Schaback, Error bounds for kernel based numerical differentiation, Numer. Math. 132 (2016) 243–269.

[40] H. Wendland, Piecewise polynomial, positive definite and compactly supported radial basis functions of minimal degree, Adv. Comput. Math. 4 (1995) 389–396.

[41] R. Schaback, Convergence of unsymmetric kernel-based meshless collocation methods, SIAM J. Numer. Anal. 45 (2007) 333–351.

[42] R. Schaback, Unsymmetric meshless methods for operator equations, Numer. Math. 114 (2010) 629–651.

[43] R. Schaback, Error analysis of nodal meshless methods, in: Meshfree Methods for Partial Differential Equations VIII, in: Lecture Notes in Computational Science and Engineering, vol. 115, Springer, 2017, Bonn, 2015, pp. 117–143.

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

[45] F.J. Narcowich, J.D. Ward, Generalized Hermite interpolation via matrix-valued conditionally positive definite functions, Math. Comput. 63 (1994) 661–687.

[46] Z. Wu, Hermite Birkhoff interpolation of scattered data by radial basis functions, Approx. Theory Appl. 8 (1992) 1–10.

[47] C. Franke, R. Schaback, Convergence order estimates of meshless collocation methods using radial basis functions, Adv. Comput. Math. 8 (1998) 381–399.

[48] C. Franke, R. Schaback, Solving partial differential equations by collocation using radial basis functions, Appl. Math. Comput. 93 (1998) 73–82.

[49] P. Giesl, H. Wendland, Meshless collocation: error estimates with application to dynamical systems, SIAM J. Numer. Anal. 45 (2007) 1723–1741.

[50] R.L. Hardy, Multiquadric equations of topography and other irregular surfaces, J. Geophys. Res. 76 (1971) 1905–1915.

[51] R. Franke, Scattered data interpolation: tests of some methods, Math. Comput. 48 (1982) 181–200.

[52] S. Rippa, An algorithm for selecting a good value for the parameter c in radial basis function interpolation, Adv. Comput. Math. 11 (1999) 193–210.

[53] M. Scheuerer, An alternative procedure for selecting a good value for the parameter c in RBF-interpolation, Adv. Comput. Math. 34 (2011) 105–126.

[54] G.E. Fasshauer, M. McCourt, On choosing “optimal” shape parameters for RBF approximation, Numer. Algorithms 45 (2007) 345–368.

[55] L. Ling, F. Marchetti, A stochastic extended Rippa’s algorithm for LpOCV, Appl. Math. Lett. 129 (2022) 107955.

[56] F. Marchetti, The extension of Rippa’s algorithm beyond LOOCV, Appl. Math. Lett. 120 (2021) 107262.

[57] H. Ramezannezhad Azarboni, M. Keyanpour, M. Yaghouti, Leave-two-out cross validation to optimal shape parameter in radial basis functions, Eng. Anal. Bound. Elem. 100 (2019) 204–210.

[58] G.B. Wright, B. Fornberg, Stable computations with flat radial basis functions using vector-valued rational approximations, J. Comput. Phys. 331 (2017) 137–156.

[59] B. Fornberg, E. Larsson, N. Flyer, Stable computations with Gaussian radial basis functions, SIAM J. Sci. Comput. 33 (2011) 869–892.

[60] B. Fornberg, E. Lehto, C. Powell, Stable calculation of Gaussian-based RBF-FD stencils, Comput. Math. Appl. 65 (2013) 627–635.

[61] S. De Marchi, G. Santin, A new stable basis for radial basis function interpolation, J. Comput. Appl. Math. 253 (2013) 1–13.

[62] G.E. Fasshauer, M. McCourt, Stable evaluation of Gaussian RBF interpolants, SIAM J. Sci. Comput. 34 (2012) A737–A762.

[63] A. Iske, On the Approximation Order and Numerical Stability of Local Lagrange Interpolation by Polyharmonic Splines, International Series of Numerical Mathematics, vol. 145, Birkhäuser Verlag, Basel, 2003, pp. 153–165.

[64] O. Davydov, R. Schaback, Optimal stencils in Sobolev spaces, IMA J. Numer. Anal. 39 (2019) 398–422.

[65] A. Iske, On the construction of kernel-based adaptive particle methods in numerical flow simulation, in: Notes on Numerical Fluid Mechanics and Multidisciplinary Design (NNFM), Springer, Berlin, 2013, pp. 197–221.