Fast and stable rational RBF-based Partition of Unity interpolation

S. De Marchi, A. Martínez, E. Perracchione

Dipartimento di Matematica "Tullio Levi-Civita", Università di Padova, Italia

Abstract

We perform a local computation via the Partition of Unity (PU) method of rational Radial Basis Function (RBF) interpolants. We investigate the well-posedness of the problem and we provide pointwise error bounds. The resulting scheme, efficiently implemented by means of the Deflation Accelerated Conjugate Gradient (DACG), enables us to deal with huge data sets and, thanks to the use of Variably Scaled Kernels (VSKs), it turns out to be stable. For functions with steep gradients or discontinuities, which are truly common in applications, the results show that the new proposed method outperforms the classical and rescaled PU schemes.

Keywords: meshfree approximation, partition of unity method, rational RBF approximation.
2010 MSC: 65D05, 65D15, 65D17.

1. Introduction

It is well-known that, for univariate functions with steep gradients, the rational polynomial approximation is particularly suitable. Usually, this approach leads to overdetermined systems and consequently the interpolation conditions must be relaxed, leading to least square solutions. Moreover, its extension to high dimensions is a challenging problem (refer e.g. to [17, 19]) and, because of the dependence on meshes, it is not easy to implement for complex shapes of the domain. These are the main reasons for which we direct our research towards Radial Basis Function (RBF)-based schemes. Indeed, taking advantage of being meshfree, they are easy to implement in high dimension.

More precisely, we give a very general formulation of a rational RBF expansion and we investigate under which conditions it leads to problems that admit solutions. Then, since the general form of the rational approximant is not uniquely defined, following the idea presented in [18], we introduce additional constraints. The scheme, that reduces to a largest eigenvalue problem, is extended to work with the Partition of Unity (PU) method [27], allowing to overcome the usually hight complexity costs of global methods, mainly due to the solution of *large* linear systems. Indeed, this local approach, based on decomposing the original reconstruction domain into many *subdomains* or *patches*, leads to solving several systems of *small* sizes. Finally, we point out that the efficiency of the method which makes use of rational interpolants is improved by means of the so-called Deflation Accelerated Conjugate Gradient (DACG) algorithm (see e.g. [2]). It allows to efficiently compute the eigenvalues of positive definite matrices.

Furthermore, for the rational RBF expansion we investigate error bounds in terms of both *power function* and *fill distance*. These bounds show strong similarities with those of the standard RBF interpolation [13, 27]. In particular, the convergence order of the rational PU method depends on the one of the local approximants and it is consistent with the empirical convergence rates.

The proposed method, thanks to the local scheme and to the DACG approach, allows to consider large data sets but might suffer from instability due to the ill-conditioning of the local interpolation matrices; refer e.g. to [5, 8, 14]. To avoid this drawback, we develop a stable computation of the rational RBF local interpolants by means of the so-called Variably Scaled Kernels (VSKs) [4]. The VSKs, via a scale function, transform the original problem so that the usual ill-conditioning of the kernel matrix is sensibly reduced.

*Corresponding author: Stefano De Marchi
Email addresses: demarchi@math.unipd.it (S. De Marchi), acalomar@math.unipd.it (A. Martínez), emma.perracchione@math.unipd.it (E. Perracchione)

This investigation reveals that the new method, namely Rational VSK-PU (RVSK-PU), performs better than the classical PU [13] or the rescaled PU method [9], which can be viewed as a particular case of the method here proposed. Moreover, when Compactly Supported RBFs (CSRBFs) are used, it enables us to increase the sparsity of the kernel matrices and, at the same time, to maintain a good accuracy.

The paper is organized as follows. In Section 2, we briefly review the basic properties of RBF interpolation, while in Section 3 we introduce the problem of rational RBF expansions. Then, in Section 4, we describe how to perform a local computation of the rational RBF interpolants and we analyze stability and computational issues in Sections 5 and 6, respectively. Numerical results and applications are shown in Section 7. Finally, Section 8 is devoted to conclusions and future work.

2. RBF interpolation

Let $\Omega \subseteq \mathbb{R}^M$ be a bounded set, $\mathcal{X}_N = \{\boldsymbol{x}_i, i = 1, \dots, N\} \subseteq \Omega$ a set of distinct data points (also called data sites or nodes) and $\mathcal{F}_N = \{f_i = f(\boldsymbol{x}_i), i = 1, \dots, N\}$ a set of data values (or measurements or function values). The interpolation problem consists in finding a function $P : \Omega \longrightarrow \mathbb{R}$ such that $P(\boldsymbol{x}_i) = f_i$, $i = 1, \dots, N$. To this end, we consider $P \in H_\Phi(\mathcal{X}_N) = \text{span}\{\Phi(\cdot, \boldsymbol{x}_i), \boldsymbol{x}_i \in \mathcal{X}_N\}$, where $\Phi : \Omega \times \Omega \longrightarrow \mathbb{R}$ is a strictly positive definite and symmetric kernel. This assumption is not restrictive, indeed to every conditionally positive definite kernel there is an associate normalized positive definite one (see for example [24]). With this choice the interpolant assumes the form [13]

$$P(\boldsymbol{x}) = \sum_{k=1}^{N} \alpha_k \Phi(\boldsymbol{x}, \boldsymbol{x}_k), \quad \boldsymbol{x} \in \Omega. \tag{1}$$

The coefficients $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_N)^T$ in (1) are found out by solving the linear system $A\boldsymbol{\alpha} = \boldsymbol{f}$, where the entries of the matrix $A \in \mathbb{R}^{N \times N}$ are given by $(A)_{ik} = \Phi(\boldsymbol{x}_i, \boldsymbol{x}_k)$, $i, k = 1, \dots, N$, and $\boldsymbol{f} = (f_1, \dots, f_N)^T$. The uniqueness of the solution is ensured by the fact that the kernel Φ is strictly positive definite and symmetric. More specifically, we take RBFs and thus we suppose that there exist a function $\phi : [0, \infty) \to \mathbb{R}$ and a shape parameter $\varepsilon > 0$ such that for all $\boldsymbol{x}, \boldsymbol{y} \in \Omega$ we have $\Phi(\boldsymbol{x}, \boldsymbol{y}) = \phi_\varepsilon(\|\boldsymbol{x} - \boldsymbol{y}\|_2) := \phi(r)$.

For each positive definite and symmetric kernel Φ it is possible to define an associated real Hilbert space, the so-called *native space* $\mathcal{N}_\Phi(\Omega)$. To such scope, we first introduce $H_\Phi(\Omega) = \text{span}\{\Phi(\cdot, \boldsymbol{x}), \boldsymbol{x} \in \Omega\}$, equipped with a bilinear form $(\cdot, \cdot)_{H_\Phi(\Omega)}$ (see [13] for further details). Since $H_\Phi(\Omega)$ is a pre-Hilbert space with reproducing kernel Φ [26], i.e. need not be complete, we define the native space $\mathcal{N}_\Phi(\Omega)$ of Φ to be the completion of $H_\Phi(\Omega)$ with respect to the norm $\|\cdot\|_{H_\Phi(\Omega)}$ so that $\|f\|_{H_\Phi(\Omega)} = \|f\|_{\mathcal{N}_\Phi(\Omega)}$, for all $f \in H_\Phi(\Omega)$, see [13, 26].

We now report the following theorem on polynomial precision (cf. [26, Th. 3.14, p. 33]).

Theorem 2.1. *Suppose that $\Omega \subseteq \mathbb{R}^M$ is compact and satisfies an interior cone condition with angle $\theta = (0, \pi/2)$ and radius $\gamma > 0$. Fix $L \in \mathbb{N}$ and let Π^M_{L-1} be the set of polynomials of degree $L - 1$. Then, there exist h_0, C_1, $C_2 > 0$ constants depending only on L, θ and γ, such that for every $\mathcal{X}_N = \{\boldsymbol{x}_i, i = 1, \dots, N\} \subseteq \Omega$ with $h_{\mathcal{X}_N} \leq h_0$, where $h_{\mathcal{X}_N}$ is the so-called fill distance*

$$h_{\mathcal{X}_N} = \sup_{\boldsymbol{x} \in \Omega} \left(\min_{\boldsymbol{x}_k \in \mathcal{X}_N} \|\boldsymbol{x} - \boldsymbol{x}_k\|_2 \right),$$

and every $\boldsymbol{x} \in \Omega$, we can find real numbers $v_k(\boldsymbol{x})$, $k = 1, \dots, N$, such that $\sum_{k=1}^N v_k(\boldsymbol{x})\, p(\boldsymbol{x}_k) = p(\boldsymbol{x})$, for all $p \in \Pi^M_{L-1}$, $\sum_{k=1}^N |v_k(\boldsymbol{x})| \leq C_1$, and $v_k(\boldsymbol{x}) = 0$ provided that $\|\boldsymbol{x} - \boldsymbol{x}_k\|_2 \geq C_2 h_{\mathcal{X}_N}$.

To formulate error bounds, we also need to define the space $C^k_\nu(\mathbb{R}^M)$ of all functions $f \in C^k$ whose derivatives of order $|\boldsymbol{\mu}| = k$ satisfy $D^{\boldsymbol{\mu}} f(\boldsymbol{x}) = O\left(\|\boldsymbol{x}\|_2^\nu\right)$ for $\|\boldsymbol{x}\|_2 \longrightarrow 0$. We are now able to give the following generic estimate (cf. [26, Th. 11.11, p. 181]). Such statement will be given for strictly positive definite functions, but it also holds for the more general case of conditionally positive definite functions. However, in that case we would only have a native space semi-norm.

Theorem 2.2. *Suppose $\phi \in C^\nu_k(\mathbb{R}^M)$ is strictly positive definite and $f \in \mathcal{N}_\Phi(\Omega)$. Moreover, suppose that $\Omega \subseteq \mathbb{R}^M$ is bounded and satisfies an interior cone condition with constants θ and γ. For $\boldsymbol{\mu} \in \mathbb{N}^M_0$, with $|\boldsymbol{\mu}| \leq k/2$ and $\mathcal{X}_N = \{\boldsymbol{x}_i, i =$*

$1, \dots, N\} \subseteq \Omega$ *satisfying* $h_{\mathcal{X}_N} \leq h_0$*, there exists a constant* C *independent of* $h_{\mathcal{X}_N}$ *and depending on* M, k, θ *and* ϕ*, such that*

$$\|D^{\boldsymbol{\mu}} f - D^{\boldsymbol{\mu}} P\|_{L_\infty(\Omega)} \leq C h_{\mathcal{X}_N}^{(k+\nu)/2-|\boldsymbol{\mu}|} \|f\|_{N_\Phi(\Omega)},$$

where $h_0 = \gamma/C_2$*, with* C_2 *from Theorem 2.1.*

3. Rational RBF interpolation

In order to introduce the rational RBF interpolation, we need to remark the main features of univariate rational polynomial approximation. Let p_1 and p_2 two univariate polynomials of degrees m and n respectively, the rational approximant of degree $m + n$ is given by

$$t(x) = \frac{p_1(x)}{p_2(x)} = \frac{a_m x^m + \cdots + a_0 x^0}{x^n + \cdots + b_0}.$$

Note that the polynomial p_2 is monic and in this way, we reduce to $m + n + 1$ unknowns. We remark that, to solve such problem, one can use *Padé* [22] or least square approximation.

3.1. General framework

It is well-known that the rational approximation is more robust than the standard polynomial interpolation for functions characterized by steep gradients. However, it is a mesh-dependent approach and, as a consequence, extending polynomial approximation in higher dimensions is quite hard (refer e.g. to [17]). This is the main reason for which we focus on RBFs as basis functions. Indeed, they take advantage of being easy to implement in higher dimensions.

Let us fix the integers m,n, k and j such that $m, n \leq N$, $1 \leq k \leq N + m - 1$ and $1 \leq j \leq N + n - 1$. Letting $\mathcal{X}_m = \{\boldsymbol{x}_i, i = k, \dots, k + m - 1\}$ and $\mathcal{X}_n = \{\boldsymbol{x}_i, i = j, \dots, j + n - 1\}$ two non-empty subsets of $\mathcal{X}_N \subseteq \Omega$, with $\Omega \subseteq \mathbb{R}^M$, a natural extension of the RBF approximation to the rational case is

$$\mathscr{R}(\boldsymbol{x}) = \frac{P^{(1)}(\boldsymbol{x})}{P^{(2)}(\boldsymbol{x})} = \frac{\sum_{i_1=k}^{k+m-1} \alpha_{i_1} \Phi(\boldsymbol{x}, \boldsymbol{x}_{i_1})}{\sum_{i_2=j}^{j+n-1} \beta_{i_2} \Phi(\boldsymbol{x}, \boldsymbol{x}_{i_2})}, \tag{2}$$

provided $P^{(2)}(\boldsymbol{x}) \neq 0$, $\boldsymbol{x} \in \Omega$. Numerically, we observed that this is in practice not so restrictive, but from now on we will always assume that the denominator of (2) is different from zero. Moreover, note that taking sequentially the points on $\mathcal{X}_m$ and $\mathcal{X}_n$ is not limiting, in fact we can trivially re-order the nodes.

To solve the rational problem in this general framework, we need to determine $n + m$ coefficients. Of course if $m + n < N$, we can find a least square approximation, as for the polynomial case. However, we are interested in interpolation and so we suppose $m + n \geq N$. The reason why we also allow $m + n > N$ (as in [18]) will be clarified in the next subsection and will enable us to overcome the problem that the rational expansion defined above is obviously non-unique (see also [25]). To point out this fact we consider a very simple example which is also devoted to show the robustness of rational RBF in case of functions with discontinuities or steep gradients.

Let us take a set of $N = 26$ univariate scattered data on $\Omega = [0, 1]$ sampled from the constant function $f(x_1) = 1$. Let us fix $\mathcal{X}_m = \{\boldsymbol{x}_i, i = 1, \dots, m\} \equiv \mathcal{X}_n$, with $m = n = 13$. It is evident that imposing the interpolation conditions

$$\mathscr{R}(\boldsymbol{x}_i) = \frac{P^{(1)}(\boldsymbol{x}_i)}{P^{(2)}(\boldsymbol{x}_i)} = f_i, \quad i = 1, \dots, N, \tag{3}$$

leads to a squared and homogeneous system of equations $B\boldsymbol{\xi} = \mathbf{0}$, where $\boldsymbol{\xi} = (\boldsymbol{\alpha}, \boldsymbol{\beta})^T$, $\mathbf{0} \in \mathbb{R}^N$ is the null vector and $B \in \mathbb{R}^{N \times N}$ is from (3). Since the function f is constant, the matrix B is singular. Thus, we have infinite solutions. Among them, as usually done in literature, we look for the one satisfying the constraint $\|\boldsymbol{\xi}\|_2 = 1$. In this way, the unique solution is given by [15]

$$\min_{\boldsymbol{\xi} \in \mathbb{R}^N, \|\boldsymbol{\xi}\|_2 = 1} \|B\boldsymbol{\xi}\|_2. \tag{4}$$

In Figure 1 (left) we plot the function reconstructed via rational RBF by solving (4) and the one found out with the classical RBF interpolation. Moreover, note that the method described by equation (4) could be used to obtain

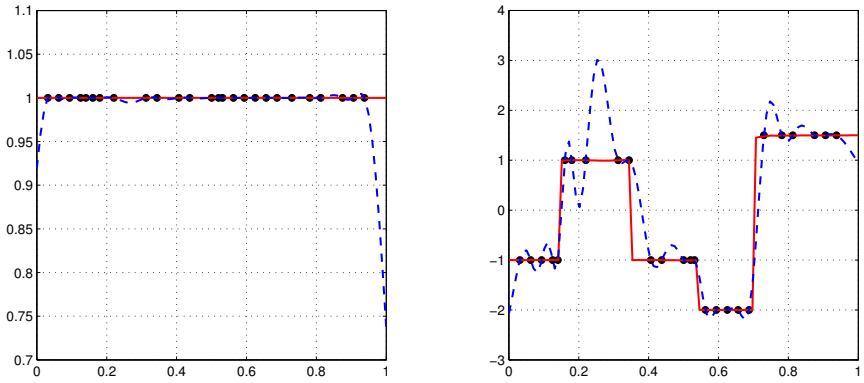

Figure 1: The black dots represent the set of scattered data, the red solid line and the blue dotted one are the curves reconstructed via the rational and classical RBF approximation, respectively.

non-trivial least square approximate solutions when the matrix is non-singular (which of course is not a solution). This is the case, for instance, of the piecewise constant function considered in Figure 1 (right) for which we take $\mathcal{X}_m = \{\boldsymbol{x}_i, i = 1, \dots, m\}$ and $\mathcal{X}_n = \{\boldsymbol{x}_i, i = m+1, \dots, N\}$, with $m = 13$.

From this first example, the fact that the rational RBF approximation enables us to better reconstruct functions with steep gradients or discontinuities turns out to be evident. This is also due to the use of VSKs, a tool that will be later introduced. However, it is also evident that the considered problem is not well-posed. Therefore, in the next subsection, following [18], we add further constraints to the interpolation conditions so that we can make $P^{(1)}$ and $P^{(2)}$ unique.

3.2. Well-posed problems

To deal with well-posed problems, according to [18], we impose extra constraints. In doing so, we focus on the case for which $m = n = N$. In this case the rational interpolant can be rewritten as

$$\mathscr{R}(\boldsymbol{x}) = \frac{P^{(1)}(\boldsymbol{x})}{P^{(2)}(\boldsymbol{x})} = \frac{\sum_{i=1}^{N} \alpha_i \Phi(\boldsymbol{x}, \boldsymbol{x}_i)}{\sum_{k=1}^{N} \beta_k \Phi(\boldsymbol{x}, \boldsymbol{x}_k)}, \tag{5}$$

Imposing the interpolation conditions would lead to solve $(A \quad -DA)\boldsymbol{\xi} = \boldsymbol{0}$, where $D = \mathrm{diag}(f_1, \dots, f_N)$ and A is the standard kernel matrix. It is easy to see that the system is underdetermined and so we add extra conditions. In practice, to have a well-posed problem, one way is to find out a vector $\boldsymbol{q} = (P^{(2)}(\boldsymbol{x}_1), \dots, P^{(2)}(\boldsymbol{x}_N))^T$ so that we can construct its relative RBF interpolant $P^{(2)}$ in the standard way. Once we have $\boldsymbol{q}$, we then define $P^{(1)}$ such that it interpolates the function values $\boldsymbol{p} = D\boldsymbol{q}$. In other words, if we assume that $\boldsymbol{p}$ and $\boldsymbol{q}$ are given, to compute the rational interpolant, we need to solve

$$A\boldsymbol{\alpha} = \boldsymbol{p}, \quad \text{and} \quad A\boldsymbol{\beta} = \boldsymbol{q}. \tag{6}$$

The existence and uniqueness of the solutions of (6) trivially follows from the fact that the kernel we consider is strictly positive definite. Thus, the problem now turns to the one of determining the vectors $\boldsymbol{p}$ and $\boldsymbol{q}$. To this aim, first note that it is reasonable to select $P^{(1)}$ and $P^{(2)}$ such that the sum of their native space norms, relative to the size of their values, is as small as possible. Following [18] this leads to find

$$\min_{\substack{P^{(1)}, P^{(2)} \in \mathcal{N}_\Phi(\Omega), \\ 1/\|\boldsymbol{f}\|_2^2 \|\boldsymbol{p}\|_2^2 + \|\boldsymbol{q}\|_2^2 = 1, \\ P^{(1)}(\boldsymbol{x}_k) = f_k P^{(2)}(\boldsymbol{x}_k).}} \left(\frac{1}{\|\boldsymbol{f}\|_2^2} \|P^{(1)}\|_{\mathcal{N}_\Phi(\Omega)}^2 + \|P^{(2)}\|_{\mathcal{N}_\Phi(\Omega)}^2 \right). \tag{7}$$

Note that the so-constructed method is not able to handle the case where f is zero at some data sites. Indeed in this case $1/\|\boldsymbol{f}\|_2^2$ would be infinite. Nevertheless, one can trivially overcome such problem.

Then, taking into account (6) and the symmetry of the kernel matrix, the native space norms can be computed as

$$\|P^{(1)}\|^2_{\mathcal{N}_\Phi(\Omega)} = \boldsymbol{p}^T A^{-1}\boldsymbol{p}, \quad \text{and} \quad \|P^{(2)}\|^2_{\mathcal{N}_\Phi(\Omega)} = \boldsymbol{q}^T A^{-1}\boldsymbol{q}.$$

Therefore, the problem (7) reduces to solve

$$\min_{\substack{\boldsymbol{q}\in\mathbb{R}^N,\\ 1/\|\boldsymbol{f}\|_2^2\|D\boldsymbol{q}\|_2^2+\|\boldsymbol{q}\|_2^2=1.}} \left(\frac{1}{\|\boldsymbol{f}\|_2^2}\boldsymbol{q}^T D^T A^{-1} D\boldsymbol{q} + \boldsymbol{q}^T A^{-1}\boldsymbol{q}\right).$$

This is equivalent to find the eigenvector $\boldsymbol{q}$ associated to the smallest eigenvalue of the generalized eigenvalue problem $\Lambda\boldsymbol{q} = \lambda\Theta\boldsymbol{q}$, with [18, 23]

$$\Lambda = \frac{1}{\|\boldsymbol{f}\|_2^2} D^T A^{-1} D + A^{-1}, \quad \text{and} \quad \Theta = \frac{1}{\|\boldsymbol{f}\|_2^2} D^T D + I_N,$$

where I_N is the $N \times N$ identity matrix. Then, we simply define $P^{(1)}$ and $P^{(2)}$ as standard RBF interpolants of $\boldsymbol{p}$ and $\boldsymbol{q}$, respectively. In other words, taking into account the fact that $P^{(2)}(\boldsymbol{x}_i) = \boldsymbol{q}_i$, $i = 1, \ldots, N$, the rational approximant can be rewritten as

$$\begin{aligned}\mathscr{R}(\boldsymbol{x}) &= \textstyle\sum_{i=1}^N \alpha_i \dfrac{\Phi(\boldsymbol{x},\boldsymbol{x}_i)}{\sum_{k=1}^N \beta_k \Phi(\boldsymbol{x},\boldsymbol{x}_k)} = \sum_{i=1}^N \alpha_i \dfrac{\Phi(\boldsymbol{x},\boldsymbol{x}_i)\, q_i}{\sum_{k=1}^N \beta_k \Phi(\boldsymbol{x},\boldsymbol{x}_k) \sum_{k=1}^N \beta_k \Phi(\boldsymbol{x}_i,\boldsymbol{x}_k)},\\ &:= \textstyle\sum_{i=1}^N \tilde{\alpha}_i \dfrac{\Phi(\boldsymbol{x},\boldsymbol{x}_i)}{\sum_{k=1}^N \beta_k \Phi(\boldsymbol{x},\boldsymbol{x}_k) \sum_{k=1}^N \beta_k \Phi(\boldsymbol{x}_i,\boldsymbol{x}_k)} := \sum_{i=1}^N \tilde{\alpha}_i \Phi_R(\boldsymbol{x},\boldsymbol{x}_i),\end{aligned}$$

where Φ_R is strictly positive definite, provided so is Φ and $P^{(2)}(\boldsymbol{x}) \neq 0$, $\boldsymbol{x} \in \Omega$ (see [1, 9] for further details). The formulation given above is not practical for implementation purposes. Nevertheless, it points out that $\mathscr{R}$ shows strong similarities with the rescaled interpolant given in [9]. Indeed, as evident, $\mathscr{R}$ is rescaled. At the same time, the rescaled interpolant, based on constructing $P^{(2)}(\boldsymbol{x})$ such that it interpolates a constant function, can be seen as a particular case of the method here propose. Finally, note that in this paper we assume $P^{(2)}(\boldsymbol{x}) \neq 0$, $\boldsymbol{x} \in \Omega$. If it does not hold, it is sufficient to impose the following constraints for the minimization problem: $q_i > 0$, $i = 1, \ldots, N$. Once we have such function values, we can use the method proposed in [10] to obtain a positive approximant.

In practice, to construct the rational interpolant, we only need to solve (6) and evaluate (5). This implies that we construct a rational RBF interpolant by means of the standard RBF interpolation matrix A. This enables us to give error bounds. In particular, one could extend Theorem 2.2. This technique will be used in the next section to bound the error via the PU interpolant. Here instead we investigate an error bound expressed in terms of the so-called *power function* $P_{\Phi,\mathcal{X}_N}(\boldsymbol{x})$. To this aim we need to think of $\boldsymbol{p}$ and $\boldsymbol{q}$ as the values obtained by sampling some functions p and $q \in \mathcal{N}_\Phi(\Omega)$.

Proposition 3.1. *Let $\Omega \subseteq \mathbb{R}^M$, $\Phi \in C(\Omega \times \Omega)$ be a strictly positive definite kernel and suppose that the points $\mathcal{X}_N = \{\boldsymbol{x}_i, i = 1, \ldots, N\} \subseteq \Omega$ are distinct. Moreover, let us suppose that p and $q \in \mathcal{N}_\Phi(\Omega)$, then*

$$|f(\boldsymbol{x}) - \mathscr{R}(\boldsymbol{x})| \leq \frac{1}{|P^{(2)}(\boldsymbol{x})|}(|f(x)|\|q\|_{\mathcal{N}_\Phi(\Omega)} + \|p\|_{\mathcal{N}_\Phi(\Omega)})P_{\Phi,\mathcal{X}_N}(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega.$$

Proof 3.1. *Let us consider $\boldsymbol{x} \in \Omega$, then $|f(\boldsymbol{x}) - \mathscr{R}(\boldsymbol{x})| = |(P^{(2)}(\boldsymbol{x})f(\boldsymbol{x}) - q(\boldsymbol{x})f(\boldsymbol{x})) + (q(\boldsymbol{x})f(\boldsymbol{x}) - P^{(1)}(\boldsymbol{x}))|/|P^{(2)}(\boldsymbol{x})|$. For a function $g \in \mathcal{N}_\Phi(\Omega)$ we have $|g(\boldsymbol{x}) - P(\boldsymbol{x})| \leq P_{\Phi,\mathcal{X}_N}(\boldsymbol{x})\|g\|_{\mathcal{N}_\Phi(\Omega)}$, $\boldsymbol{x} \in \Omega$ (see e.g. [13, Th. 14.2, p.117]). Therefore,*

$$\left|q(\boldsymbol{x}) - P^{(2)}(\boldsymbol{x})\right| \leq P_{\Phi,\mathcal{X}_N}(\boldsymbol{x})\|q\|_{\mathcal{N}_\Phi(\Omega)}.$$

Moreover, taking into account how $\boldsymbol{p}$ and $\boldsymbol{q}$ are related, we obtain

$$|f(\boldsymbol{x}) - \mathscr{R}(\boldsymbol{x})| \leq \frac{1}{|P^{(2)}(\boldsymbol{x})|}\left(|f(\boldsymbol{x})|\|q\|_{\mathcal{N}_\Phi(\Omega)} + \|p\|_{\mathcal{N}_\Phi(\Omega)}\right) P_{\Phi,\mathcal{X}_N}(\boldsymbol{x}).$$

Remark 3.1. *Proposition 3.1 enables us to study pointwise error in terms of the power function. Thus, also for the rational RBF expansion we recover a bound split into two components. The former depends on the kernel Φ via the native space norm and the latter only depends on the data sites. Later, for the PU interpolant we will see that we can refine the error bound by expressing the influence of the data locations in terms of the fill distance and it should be clear that this can be used for the global case too.*

The main drawback of the method described in this section is the computational cost due to solving potentially large systems. For instance, when we deal with real applications, we often need to interpolate huge data sets and in those cases the computational cost is prohibitive. This is the main reason for which we direct our research towards the PU method.

4. Rational RBF-PU interpolation

The basic idea of the PU method consists in decomposing the original reconstruction domain into many subdomains. Then, for each patch we solve local interpolation problems of small sizes.

At first, we cover the domain Ω with d overlapping subdomains Ω_j. The overlap must be sufficient so that each point $\boldsymbol{x} \in \Omega$ is located in the interior of at least one subdomain Ω_j. To be more precise, we require a regular covering, i.e. $\{\Omega_j\}_{j=1}^{d}$ must fulfil the following properties [27]:

i. for each $\boldsymbol{x} \in \Omega$, the number of subdomains Ω_j, with $\boldsymbol{x} \in \Omega_j$, is bounded by a global constant d_0,

ii. the local fill distances $h_{\mathcal{X}_{N_j}}$ are uniformly bounded by the global fill distance $h_{\mathcal{X}_N}$, where $\mathcal{X}_{N_j} = \mathcal{X}_N \cap \Omega_j$.

iii. there exists $C_r > C_2$ such that each subdomain Ω_j satisfies an interior cone condition with angle $\tilde{\theta} \in (0, \pi/2)$ and radius $\tilde{\gamma} = C_r h_{\mathcal{X}_N}$.

Note that, the first assumption plays a crucial role also in the implementation of the PU method. In fact, such property leads to the requirement that the number of subdomains is proportional to the number of data [27].

Moreover, we need to select a family of d continuous and non-negative weight functions W_j which form a partition of unity and such that $\mathrm{supp}(W_j) \subseteq \Omega_j$. For instance, such conditions are fulfilled by the so-called Shepard's weights (see e.g. [13]). Then, the rational RBF-PU interpolant assumes the form

$$\mathcal{I}(\boldsymbol{x}) = \sum_{j=1}^{d} \mathscr{R}_j(\boldsymbol{x})\, W_j(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega,$$

where $\mathscr{R}_j$ denotes a rational RBF interpolant defined on a subdomain Ω_j. Therefore, it assumes the form

$$\mathscr{R}_j(\boldsymbol{x}) = \frac{P_j^{(1)}(\boldsymbol{x})}{P_j^{(2)}(\boldsymbol{x})} = \frac{\sum_{i=1}^{N_j} \alpha_i^j \Phi(\boldsymbol{x}, \boldsymbol{x}_i^j)}{\sum_{k=1}^{N_j} \beta_k^j \Phi(\boldsymbol{x}, \boldsymbol{x}_k^j)},$$

where N_j indicates the number of points on Ω_j and $\boldsymbol{x}_k^j \in \mathcal{X}_{N_j}$, with $k = 1, \ldots, N_j$.

Thus, for each patch Ω_j we define $\boldsymbol{q}_j$ as the eigenvector associated to the smallest eigenvalue of the problem $\Lambda_j \boldsymbol{q}_j = \lambda_j \Theta_j \boldsymbol{q}_j$, with

$$\Lambda_j = \frac{1}{\|\boldsymbol{f}_j\|_2^2} D_j^T A_j^{-1} D_j + A_j^{-1}, \quad \text{and} \quad \Theta_j = \frac{1}{\|\boldsymbol{f}_j\|_2^2} D_j^T D_j + I_{N_j},$$

where $\boldsymbol{f}_j = (f_1^j, \ldots, f_N^j)^T$ are the local function values, $D_j = \mathrm{diag}(f_1^j, \ldots, f_N^j)$ and A_j is the standard local kernel matrix. Once $\boldsymbol{p}_j$ and $\boldsymbol{q}_j$ are found out for each patch Ω_j, the coefficients $\boldsymbol{\alpha}_j$ and $\boldsymbol{\beta}_j$ are defined by solving the two linear systems $A_j \boldsymbol{\alpha}_j = \boldsymbol{p}^j$ and $A_j \boldsymbol{\beta}_j = \boldsymbol{q}^j$.

Remark 4.1. *The PU subdomains have some mild overlap among them and thus a point, let us say $\boldsymbol{x}_k$, could belong to more than one patch. Let us suppose that $\boldsymbol{x}_k$ belongs to Ω_l and Ω_m. On one hand, we might have $P_l^{(1)}(\boldsymbol{x}_k) \neq P_m^{(1)}(\boldsymbol{x}_k)$ and $P_l^{(2)}(\boldsymbol{x}_k) \neq P_m^{(2)}(\boldsymbol{x}_k)$, but on the other one we obtain that $\mathscr{R}_l(\boldsymbol{x}_k) = \mathscr{R}_m(\boldsymbol{x}_k)$, making the problem consistent.*

According to [27], several further requirements on the PU weights are needed. In particular, the functions W_j, $j = 1, \dots, d$, must form a k-stable partition of unity. Such assumption is essential to prove the convergence theorem for a general derivative of order $|\boldsymbol{\mu}| = k$ of the classical PU interpolant (see [26] Th. 15.19, p. 277 and [27]). For the rational approximant, we have the following error bound.

Proposition 4.1. *Suppose $\phi \in C_k^{\nu}(\mathbb{R}^M)$ is strictly positive definite. Let $\mathcal{X}_N = \{\boldsymbol{x}_i, i = 1, \dots, N\} \subseteq \Omega$. Let $\{\Omega_j\}_{j=1}^d$ be a regular covering for $(\Omega, \mathcal{X}_N)$ and W_j, $j = 1, \dots, d$, a family of continuous and non-negative functions which form a partition of unity and such that supp$(W_j) \subseteq \Omega_j$. Then, there exists an index $t \in \{1, \dots, d\}$ and a constant C^*, independent of $h_{\mathcal{X}_N}$, and depending on M, k, $\tilde{\theta}$ and ϕ such that, if p_t and $q_t \in \mathcal{N}_\Phi(\Omega_t)$, for all $\boldsymbol{x} \in \Omega$ we have:*

$$|f(\boldsymbol{x}) - \mathcal{I}(\boldsymbol{x})| \leq \frac{1}{\left|P_t^{(2)}(\boldsymbol{x})\right|} C^* h_{\mathcal{X}_N}^{(k+\nu)/2} (|f_{|\Omega_t}(x)| \|q_t\|_{\mathcal{N}_\Phi(\Omega_t)} + \|p_t\|_{\mathcal{N}_\Phi(\Omega_t)}),$$

provided that $\mathcal{X}_{N_t} = \{\boldsymbol{x}_i, i = 1, \dots, N_t\} \subseteq \Omega_t$ satisfies $h_{\mathcal{X}_{N_t}} \leq h_0$, with $h_0 = \tilde{\gamma}/C_2^t$ and C_2^t is from Theorem 2.1 applied to the local setting. We denote by $f_{|\Omega_t}$ the restriction of f on Ω_t.

Proof 4.1. *Since W_j $j = 1, \dots, d$, are a family of continuous and non-negative functions which form a partition of unity and such that supp$(W_j) \subseteq \Omega_j$, we obtain*

$$|f(\boldsymbol{x}) - \mathcal{I}(\boldsymbol{x})| \leq \sum_{j=1}^{d} W_j(\boldsymbol{x}) |f(\boldsymbol{x}) - \mathscr{R}_j(\boldsymbol{x})| \leq \max_{j=1,\dots,d} \|f(\boldsymbol{x}) - \mathscr{R}_j(\boldsymbol{x})\|_{L_\infty(\Omega_j)} := \|f_{|\Omega_t}(\boldsymbol{x}) - \mathscr{R}_t(\boldsymbol{x})\|_{L_\infty(\Omega_t)}.$$

Furthermore, we can apply Theorem 2.2 to the local setting. Indeed, since we require a regular covering, we know that each subdomain fulfills an interior cone condition. If $\mathcal{X}_{N_t} = \{\boldsymbol{x}_i, i = 1, \dots, N_t\} \subseteq \Omega_t$ satisfies $h_{\mathcal{X}_{N_t}} \leq h_0$, taking into account Theorem 2.2 and the argument used in the proof of Proposition 3.1, there exists a constant C^*, independent of $h_{\mathcal{X}_{N_t}}$, and depending on M, k, $\tilde{\theta}$ and ϕ such that

$$\|f_{|\Omega_t}(\boldsymbol{x}) - \mathscr{R}_t(\boldsymbol{x})\|_{L_\infty(\Omega_t)} \leq \frac{1}{\left|P_t^{(2)}(\boldsymbol{x})\right|} C^* h_{\mathcal{X}_N}^{(k+\nu)/2} (|f_{|\Omega_t}(x)| \|q_t\|_{\mathcal{N}_\Phi(\Omega_t)} + \|p_t\|_{\mathcal{N}_\Phi(\Omega_t)}).$$

Remark 4.2. *If we compare the result reported in Proposition 4.1 with the error estimate shown in Theorem 2.2, we can see that the rational PU interpolant preserves the local approximation order. This will be empirically verified in the next section where we estimate the convergence rates.*

Finally we point out that, as for the standard RBF interpolant, the rational one might suffer from instability problems, especially for small values of the shape parameter. Thus, we carry out a stable computation via VSKs.

5. Stability issues

Also for rational interpolation the selection of the shape parameter ε is a tedious computational issue. Indeed, wrong choices of the shape parameter might lead to serious problems of ill-conditioning, especially for smooth RBFs.

Therefore, the choice of the shape parameter is crucial and can be selected such that it is *optimal* (for a general overview see [13]). However, selecting its optimal value is usually expensive and moreover, all the techniques based on *a priori* error estimates only give an approximated optimal value; meaning that such value is *close* to the one that can be found via trials provided that the solution is known.

Alternatively, one can use stable approximations of the solution. For the Gaussian kernel there are well-established tools, such as RBF-QR and Hilbert-Schmidt Singular Value Decomposition (HS-SVD) methods, that allow to overcome the instability issues. Refer to [12, 14] for further details on these techniques. Finally, we remark that there are also two other classes of stable methods, namely the Weighted SVD (WSVD) [5, 8] and the VSKs [4] that properly work with any RBF.

Here, we propose a stable method for the solution of the rational RBF problem via VSKs. The idea is from [4] and will be reviewed in what follows. The VSKs allow to vary the scales in a continuous way by letting the scale parameter be an additional coordinate. According to this, we give the following (cf. [4, Def. 2.1]).

Definition 5.1. *Let $\psi : \mathbb{R}^M \to (0,\infty)$ be a given scale function. A Variably Scaled Kernel (VSK) K_ψ on $\mathbb{R}^M$ is*

$$K_\psi(\boldsymbol{x},\boldsymbol{y}) := \mathcal{K}((\boldsymbol{x},\psi(\boldsymbol{x})),(\boldsymbol{y},\psi(\boldsymbol{y}))), \quad \forall \boldsymbol{x},\boldsymbol{y} \in \mathbb{R}^M.$$

where $\mathcal{K}$ is a kernel on $\mathbb{R}^{M+1}$.

In particular the rational PU interpolant that makes use of VSKs (RVSK-PU) can be defined as follows.

Definition 5.2. *Given the set $\mathcal{X}_N = \{\boldsymbol{x}_i,\ i = 1,\dots N\}$ on Ω and the scale functions $\psi_j : \mathbb{R}^M \to (0,\infty)$, $j = 1,\dots,d$, the RVSK-PU interpolant is*

$$\mathcal{I}_\psi(\boldsymbol{x}) = \sum_{j=1}^{d} \mathscr{R}_{\psi_j}(\boldsymbol{x})\, W_j(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega, \tag{8}$$

with

$$\mathscr{R}_{\psi_j}(\boldsymbol{x}) = \frac{\sum_{i=1}^{N_j} \alpha_i^{\psi_j} \mathcal{K}((\boldsymbol{x},\psi_j(\boldsymbol{x})),(\boldsymbol{x}_i^j,\psi_j(\boldsymbol{x}_i^j)))}{\sum_{k=1}^{N_j} \beta_k^{\psi_j} \mathcal{K}((\boldsymbol{x},\psi_j(\boldsymbol{x})),(\boldsymbol{x}_k^j,\psi_j(\boldsymbol{x}_k^j)))}, \quad \boldsymbol{x} \in \Omega_j, \tag{9}$$

In Definition 5.2, the coefficients $\boldsymbol{\alpha}_{\psi_j}$ and $\boldsymbol{\beta}_{\psi_j}$ are found out at first by solving the eigenvalue problem

$$\Lambda_{\psi_j} \boldsymbol{q}_{\psi_j} = \lambda_{\psi_j} \Theta_j \boldsymbol{q}_{\psi_j}, \tag{10}$$

where

$$\Lambda_{\psi_j} = \frac{1}{\|\boldsymbol{f}_j\|_2^2} D_j^T A_{\psi_j}^{-1} D_j + A_{\psi_j}^{-1}, \tag{11}$$

and A_{ψ_j} is constructed via (9). Then, we simply need to solve the two linear systems

$$A_{\psi_j} \boldsymbol{\alpha}_{\psi_j} = \boldsymbol{p}_{\psi_j}, \quad \text{and} \quad A_{\psi_j} \boldsymbol{\beta}_{\psi_j} = \boldsymbol{q}_{\psi_j}. \tag{12}$$

Furthermore, from [4] we know that if the VSK $\mathcal{K}$ is radial then the entries of the associated kernel matrix A_{ψ_j} are given by

$$(A_{\psi_j})_{ik} = \phi\left(\left(\|\boldsymbol{x}_i^j - \boldsymbol{x}_k^j\|_2^2 + (\psi_j(\boldsymbol{x}_i^j) - \psi_j(\boldsymbol{x}_k^j))^2\right)^{1/2}\right), \quad i,k = 1,\dots,N_j. \tag{13}$$

Remark 5.1. *From* (13)*, the fact that the well-known separation distance (see e.g. [13]) never decreases easily follows. This allows to partially overcome problems due to instability. Indeed, the ill-conditioning grows primarily due to the decrease of the separation distance and not necessarily due to the increase of the number of data points. However, also the fill distance, which from Proposition 2.2 we know that it is a measure of the error, grows. In fact, the rational VSK interpolation by the kernel $\mathcal{K}$ takes place on $\mathbb{R}^{M+1}$. So the analysis of error and stability of the variably-scale problem on $\mathbb{R}^M$ coincides with the analysis of a fixed-scale problem on a submanifold in $\mathbb{R}^{M+1}$.*

6. Computational issues

This section is devoted to summarize the RVSK-PU algorithm and to point out the computational aspects.

`Step 1: Construction of the PU framework`

In the PU setting, at first we need to define the set of evaluation points $\mathcal{E}_s = \{\bar{\boldsymbol{x}}_i, i = 1,\dots,s\} \subseteq \Omega$ and the set of PU centres $C_d = \{\tilde{\boldsymbol{x}}_i, j = 1,\dots,d\} \subseteq \Omega$. Both these sets are constructed as grids of points on Ω. Furthermore, in Section 4, we require that the subdomains Ω_j, $j = 1,\dots,d$, form an open, bounded and regular covering for Ω. In what follows, as subdomains we take balls of a certain radius δ. In particular, we know that the number of patches d should be proportional to N and a suitable choice is given by $N/d \approx 2^M$ [13].

Also the PU radius δ must be carefully chosen. In fact, the patches must be a covering of the domain Ω and must satisfy the overlap condition. A possible choice that fulfills the above requirements is given by: $\delta \geq \delta_0$, with $\delta_0 = (1/d)^{1/M}$.

Once the PU subdomains are generated, the whole problem reduces to solving for each patch a local rational interpolation problem. Specifically, for the j-th local interpolation problem, only those data sites and evaluation points belonging to Ω_j are involved. Consequently, a partitioning data structure and a related searching procedure must be employed to efficiently find for each subdomain the set of nodes $\mathcal{X}_{N_j}$ and the one of evaluation points $\mathcal{E}_{s_j}$ lying on Ω_j. To this end, we perform the so called integer-based data structure outlined in [7].

Step 2: Construction of the VSKs local distance matrices

After organizing both data sites and evaluation points, on a given Ω_j we define a scale function ψ_j and compute the local matrices Λ_{ψ_j} and Θ_j (see (10)–(11)), where A_{ψ_j} is obtained by means of VSKs. In this way we are able to overcome the instability problems. Then, to compute local rational fits, i.e. rational interpolants on each subdomain, we need to solve the eigenvalue problem outlined in (10).

Step 3: Construction of the local fit

To solve the eigenvalue problem, a number of iterative procedures have been developed to compute a few eigenpairs of a large and sparse symmetric positive definite matrix, or to solve the partial generalized symmetric eigenvalue problem on Ω_j defined as in (10). One of the most widely used algorithms is the Implicitly Restarted Lanczos method (IRLM). Here instead, we consider the DACG [3]. All these methods are iterative, in the sense that they compute one or more eigenpairs by constructing a sequence of vectors which approximate the exact solution. However, the DACG has been shown to be highly competitive with the Lanczos method [20] when a small number of eigenpairs are being sought. As numerical evidence will confirm, in this case, it leads to a meaningful saving of computational time. The DACG sequentially computes on Ω_j the eigenpairs by minimizing the *Rayleigh quotient* $Q(z_j) = z_j^T \Lambda_{\psi_j} z_j / z_j^T \Theta_j z_j$, $z_j \in \mathbb{R}^{N_j}$, over the subspace orthogonal to the eigenvectors previously computed.

Given a positive definite matrix S, it calculates the eigenvalues sequentially, starting from the smallest one [2, 3], by a nonlinear Conjugate Gradient minimization of the Rayleigh quotient over a subspace orthogonal to the previously computed eigenvectors. Like Preconditioned Conjugate Gradient (PGD) for linear systems, also DACG takes advantage of preconditioning. The best results in terms of convergence are obtained when the preconditioner is S^{-1}. In such case the number of DACG iterations are found to be inversely proportional to the square root of the relative separation between consecutive eigenvalues. Similar convergence rates (and a more cost-effective algorithm) are obtained using the inverse of the Cholesky factorization as preconditioner. In any case, when two eigenvalues are relatively very close, DACG convergence may be very slow.

The extension of DACG to the generalized eigenproblem defined in (10) is straighforward. In this case, being both matrices strictly positive definite, the algorithm returns a set of Θ_j-orthonormal vectors approximating the leftmost eigenvectors of the matrix pencil $(\Lambda_{\psi_j}, \Theta_j)$.

Step 4: Construction of the RVSK-PU

Finally, when $\boldsymbol{q}_{\psi_j}$ and $\boldsymbol{p}_{\psi_j}$ are computed, we need to solve the two linear systems (12). Then, by means of the PU weights, the local fits are accumulated into the global rational interpolant $\mathcal{I}_\psi$ (8).

7. Numerical experiments

The experiments have been carried out with MATLAB on a Intel(R) Core(TM) i7 CPU 4712MQ 2.13 GHz processor. The codes are available at `http://www.math.unipd.it/~demarchi/RBF/CAARBF.html`.

In this section we fix $M = 2$ and $\Omega = [0, 1]^2$. However, we point out that any extension to non-trivial domains is possible and straightforward. In the numerical results we use two kinds of data sets: Halton [16] and Non-Conformal points [11]. The former are quasi-uniform, while the latter are clustered in certain regions and therefore the interpolation problem is particularly challenging.

To test the accuracy of the RVSK-PU method, we take the following oscillating functions with steep gradients

$$f_1(x_1, x_2) = \frac{\tan[9(x_2 - x_1) + 1]}{\tan 9 + 1}, \quad f_2(x_1, x_2) = (x_1 + x_2 - 1)^7,$$

and

$$f_3(x_1, x_2) = \sin(3x_1^2 + 6x_2^2) - \sin(2x_1^2 + 4x_2 - 0.5).$$

The RBFs considered in the examples are the Gaussian, the Inverse MultiQuadric (IMQ), the Wendland's C^2 and the Matérn C^2 functions, whose formulae respectively are

$$\phi_1(r) = e^{-\varepsilon^2 r^2}, \quad \phi_2(r) = \frac{1}{\sqrt{1+(\varepsilon r)^2}}, \quad \phi_3(r) = (1-\varepsilon r)_+^4(4\varepsilon r+1), \quad \text{and} \quad \phi_4(r) = \mathrm{e}^{-\varepsilon r}(\varepsilon r+1).$$

where r is the Euclidean distance, ε is the shape parameter and $(\cdot)_+$ denotes the truncated power function. The interpolants are evaluated on a grid of $s = 40 \times 40$ points $\mathcal{E}_s = \{\bar{\boldsymbol{x}}_i, i = 1, \ldots, s\}$. To point out the accuracy of the RVSK-PU, we refer to the Root Mean Square Error (RMSE) or to the Relative RMSE (RRMSE):

$$\text{RMSE} = \sqrt{\frac{1}{s}\sum_{i=1}^{s} |f(\bar{\boldsymbol{x}}_i) - \mathcal{I}_\psi(\bar{\boldsymbol{x}}_i)|^2}, \quad \text{RRMSE} = \sqrt{\frac{1}{s}\sum_{i=1}^{s} \frac{|f(\bar{\boldsymbol{x}}_i) - \mathcal{I}_\psi(\bar{\boldsymbol{x}}_i)|^2}{f(\bar{\boldsymbol{x}}_i)^2}}.$$

We also estimate the empirical convergence rate via the formula:

$$\lambda_k = \frac{\log(\text{RMSE}_{k-1}/\text{RMSE}_k)}{\log(h_{k-1}/h_k)}, \quad k = 2, 3, \ldots$$

where RMSE_k is the error for the k-th numerical experiment and h_k is the fill distance of the k-th computational mesh. Furthermore, we evaluate the Maximum Condition Number (MCN) of the local matrices.

Remark 7.1. *Using VSKs can lead to small perturbation on the fill distances. Indeed, the original points are essentially mapped throughout the scale function. However, since we compare the RVSK-PU with other methods, it is reasonable to approximate the convergence rates with the fill distances of the original data sets.*

The RVSK-PU is applied with circular patches defined in Section 6. Furthermore, for a given Ω_j, we consider the following scale function

$$\psi_j^{(1)}(x_1, x_2) = 0.5 + \sqrt{r^2 - [(x_1 - \tilde{x}_{1j})^2 + (x_2 - \tilde{x}_{2j})^2]}, \tag{14}$$

with $\boldsymbol{x} = (x_1, x_2) \in \Omega_j$. This choice is quite natural; indeed given circular patches, we map them on a semisphere of radius r. In what follows we fix $r = 3$. In general, since the aim consists in increasing the separation distance to improve the stability, *small* values for r are not recommended. Furthermore, note that for CSRBFs, the scale function enables us to control the sparsity of the interpolation matrix. For instance, let us consider the following scale function on Ω_j

$$\psi_j^{(2)}(x_1, x_2) = u\sqrt{x_1^2 + x_2^2}, \tag{15}$$

where u is the scale factor and $\boldsymbol{x} = (x_1, x_2) \in \Omega_j$. If we take $u = 1$, then the sparsity of the interpolation matrix will coincide with that of the standard one found out with the support of the CSRBF equal to 1. Thus, the sparsity of the interpolation matrix grows with u. In the numerical experiments that follow we take $u = 9$. Moreover, we point out that for a standard CSRBF interpolant, as the support of the function becomes smaller, the sparsity of the interpolation matrix increases, but the accuracy of the fit usually gets worse. On the opposite, the scale via VSKs enables us to construct moderately sparse matrices and to maintain a good accuracy.

The generalized eigenvalue problem (10) is solved by means of the DACG method. To stop the DACG iteration on a given subdomain Ω_j, we use the following exit test: $w_k/w_0 \leq \tau_{DACG}$, with $w_k = \|\Lambda_{\psi_j} \boldsymbol{z}_j^k - Q(\boldsymbol{z}_j^k)\Theta_j \boldsymbol{z}_j^k\|/\|\boldsymbol{z}_j^k\|_{\Theta_j}$, where k identifies the iteration number of DACG. It allows the iteration to proceed until the initial eigenresidual norm has been reduced by a factor τ_{DACG}. All the results included in this section have been obtained using $\tau_{DACG} = 10^{-2}$. Decreasing the value of parameter τ_{DACG} does not produce a reduction of the RMSE. The DACG method has been preconditioned by matrix A_{ψ_j}, which can be thought as a good approximation of $\Lambda_{\psi_j}^{-1}$ (see (11)). For the same reason, we compute a suitable starting vector for the DACG method by performing a few (very cheap) iterations of the power method to approximate the largest eigenvalue of A_{ψ_j}. Roughly speaking, we will show that with the DACG algorithm we are able to obtain approximations that are *really close* to the ones that can be found with the IRLM routine, but with a meaningful saving in term of computational time. We remark that the IRLM is implemented in MATLAB by the function `eigs` (see also ARPACK [21]).

7.1. Tests with globally defined RBFs

As first example we take several sets of Halton data and the test function f_1. We compare the standard PU method, which in what follows is denoted by SRBF-PU, with the RVSK-PU interpolant. For both methods we fix the radius of patches $\delta = \delta_0$. The RMSEs, MCNs and empirical convergence rates, obtained with the Gaussian kernel and the scale function (14), are reported in Table 1. The shape parameter for the standard PU has been selected so that the MCN is comparable to the one that is found out with the RVSK-PU scheme. As expected, the RVSK-PU gives more accurate approximations when the conditioning grows, enhancing the stability of the method. Finally, the surfaces reconstructed with the SRBF-PU and RVSK-PU are plotted in Figure 2.

The accuracy of the RVSK-PU method is also confirmed by the estimated convergence rates. However, it is difficult to deal with the spectral convergence predicted for infinitely smooth kernels, which is usually visible for a limited number of experiments [13]. We can see that the convergence rates of the two methods are comparable, but the RVSK-PU saturates faster. In other words, we observe in some sense a kind of trade-off; precisely, the rational approach enables us to improve the accuracy, especially for functions with steep gradients and also for data sets of relatively small size, but saturates for data sets that are only moderately large.

N	h	Method	ε	RMSE	MCN	λ
1089	3.93E − 02	SRBF-PU	2	4.03E − 02	6.58E + 19	–
		RVSK-PU	–	1.39E − 04	1.04E + 20	–
4225	2.91E − 02	SRBF-PU	3	5.48E − 04	2.41E + 21	14.3
		RVSK-PU	–	6.04E − 06	7.61E + 21	10.4
16641	1.03E − 02	SRBF-PU	4	3.46E − 05	2.75E + 23	2.65
		RVSK-PU	–	6.97E − 07	4.67E + 21	2.07
66049	4.93E − 03	SRBF-PU	5	1.22E − 06	5.60E + 22	4.53
		RVSK-PU	–	2.76E − 07	3.09E + 22	1.25

Table 1: Fill distances, RMSEs, MCNs and convergence rates computed on Halton points. Results are obtained using f_1 as test function and the Gaussian C^∞ kernel as local approximant.

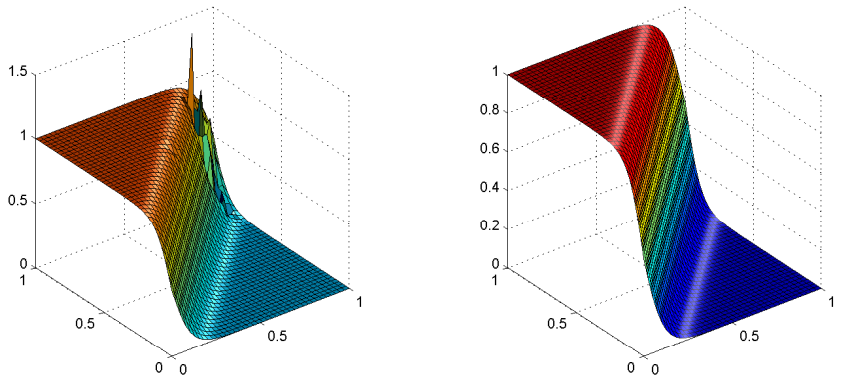

Figure 2: The approximate surface f_1 obtained with the SRBF-PU (left) and RVSK-PU (right) methods with $N = 1089$ Halton data. Results are computed using the Gaussian C^∞ kernel as local approximant.

For this example, we also compare the RMSEs and CPU times of the RVSK-PU computed without the DACG algorithm, i.e. the RVSK-PU implemented with the IRLM scheme. This test is devoted to show that by means of

the DACG algorithm, RVSK-PU performs with about the same CPU times of the standard PU, while the RVSK-PU implemented with the IRLM method is slower; see Figure 3 (left).

DACG revealed superior to the IRLM scheme on all test cases. This is mainly due to the fact that, differently from IRLM, DACG does not require any linear system solution, but only matrix vector products with the matrices Λ_{ψ_j} and Θ_j. In this way these matrices are never formed explicitly inside the MATLAB code, and its product times a vector is done by means of suitable function handles. Besides, the IRLM is penalized by the need of constructing a projection subspace even in the case the number of eigenpairs sought is equal to one. Further, we also compare the RMSE of the two different implementation of the RVSK-PU, to point out that the two different approaches (DACG and IRLM) are comparable from the point of view of the accuracy, refer to Figure 3 (right). In view of these results, in what follows we omit the comparison between the two different implementation of the RVSK-PU. Indeed, similar results also hold for the following tests and, because of the saving in terms of computational time, we will always use the DACG method.

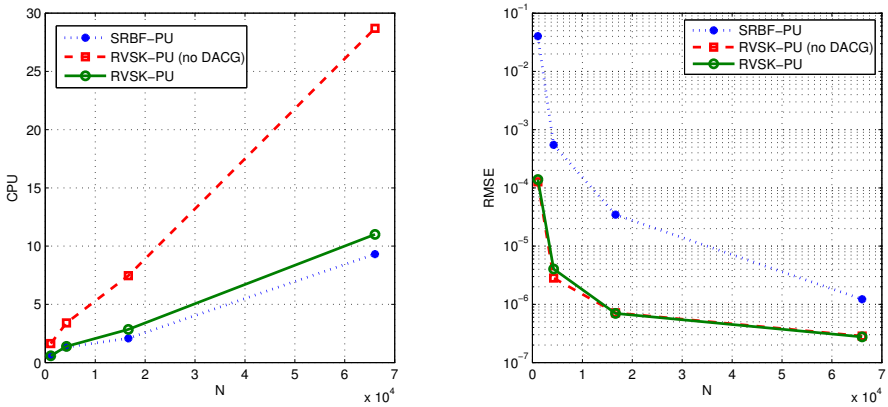

Figure 3: Left: the number of points N versus the CPU times for the SRBF-PU, RVSK-PU and RVSK-PU (no DACG), i.e. the RVSK-PU computed with the IRLM scheme. Right: the number of points N versus the logarithmic scale of the RMSE for the SRBF-PU, RVSK-PU and RVSK-PU (no DACG).

For the second test we take Non-Conformal points and the test function f_2. Again, we compare the classical PU method with the RVSK-PU. For the latter we select the optimal PU radius as in [7], while for the classical PU the radius is selected so that the Average of Points (AP) per patch is about the same than the one of the RVSK-PU approach. The results obtained with the IMQ kernel and the scale function (14) are reported in Table 2. We also point out that, from numerical experiments here omitted, the choice of the shape parameter for the SRBF-PU does not improve the accuracy of the approximation; meaning that for the considered points and test function the approximation obtained with this method is not performing. On the opposite, the RVSK-PU gives truly accurate approximations. This is also evident from Figure 4, where we can see that it is more robust in approximating the function where the gradient is more steep.

Concerning the convergence rates, for the RVSK-PU we recover the same pattern of the previous example. On the contrary, with this kind of data, the SRBF-PU method leads to negative convergence rates, meaning that it completely fails.

7.2. Tests with compactly supported RBFs

For the last example we consider noisy Halton data (0.01 noise) and the test function f_3. For both the SRBF-PU and RVSK-PU methods we select $d = 16$ subdomains (fixed for all the data sets) and the radius of patches equal to δ_0. This choice follows from the fact that we want to take advantage of using compactly supported functions, allowing the construction of sparse interpolation matrices. We will refer to the Average Density (AD) of the matrices, which represents an average of the number of non-zero entries of the local matrices. In this experiment we take the

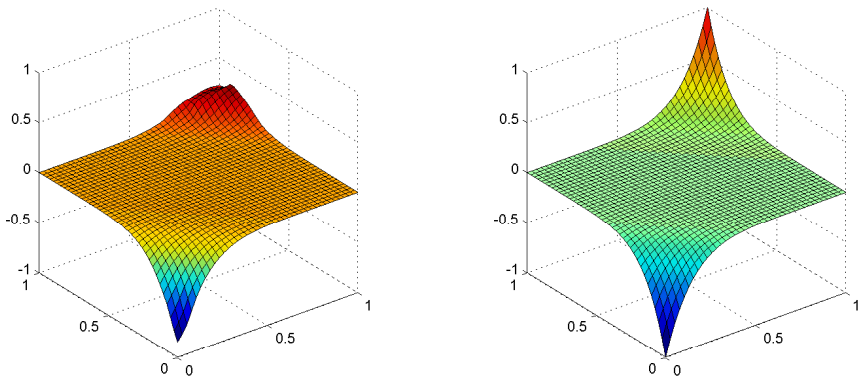

Figure 4: The approximate surface f_2 obtained with the SRBF-PU (left) and RVSK-PU (right) methods with $N = 1089$ Non-Conformal points. Results are computed using the IMQ C^∞ kernel as local approximant.

Wendland's C^2 kernel. Therefore, we carry out a comparison with the rescaled method proposed in [9], which is well-known to be truly performing when CSRBFs are used. Such method will be denoted by RSRBF-PU.

The results obtained with the scale function (15) are reported in Table 3. The shape parameter for the RSRBF-PU method is selected such that the related RMSEs are comparable with the ones of the RVSK-PU. In this case, about with the same accuracy, the RVSK-PU increases the sparsity of the matrices. If we instead choose the shape parameter for the RSRBF-PU so that the average densities of the considered approaches are comparable, then the approximations via the RSRBF-PU method are not satisfying.

7.3. Tests with real data

In this subsection we focus on an application to Earth's topography. The data set we consider is called *Montebelluna data*. The points describe the homonymous municipality in Province of Treviso and surrounding areas, such as Valdobbiadene. They can be downloaded at `http://idt.regione.veneto.it/app/metacatalog/`, where the geographic data of Veneto Region are available. The data set consists of 62677 grid data and among them we take 2726 points to test the accuracy via cross-validation, only taking the remaining 59951 for the interpolation. For a 3D view refer to Figure 5 (left).

To reconstruct the surface described by these data, we consider the RVSK-PU method with the Matérn C^2 function. In Figure 5 (right), we show the so-reconstructed surface (RRMSE = 2.11E − 02). In this case we do not report the

N	h	Method	ε	RMSE	AP	λ
1089	1.27E − 01	SRBF-PU	4	3.94E − 02	58	–
		RVSK-PU	–	1.32E − 04	59	–
4225	7.63E − 02	SRBF-PU	4	3.32E − 02	63	0.33
		RVSK-PU	–	2.76E − 06	63	7.59
16641	5.93E − 02	SRBF-PU	4	5.15E − 02	68	−1.74
		RVSK-PU	–	1.65E − 06	69	2.26
66049	3.62E − 02	SRBF-PU	4	5.70E − 02	62	−0.20
		RVSK-PU	–	2.93E − 07	62	3.38

Table 2: Fill distances, RMSEs, APs and convergence rates computed on Halton points. Results are obtained using f_2 as test function and the IMQ C^∞ kernel as local approximant.

errors obtained via the classical PU method since it fails.

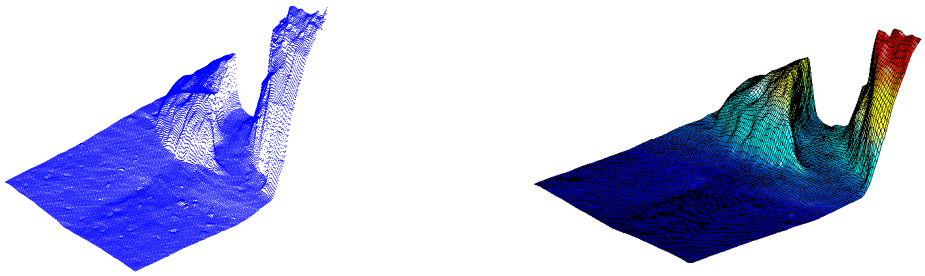

Figure 5: The 3D view of the Montebelluna (left) data. The surface reconstructed via the RVSK-PU (right) method.

8. Conclusions

In this paper we have presented a fast algorithm for the PU method which makes use of rational RBF expansions as local approximants. Both theoretical investigations and numerical evidence show that it is truly effective compared to the classical PU scheme. Furthermore, when CSRBFs are used it performs better than the rescaled PU method.

Future work consists in investigating the behavior of the RVSK-PU for interpolating nodes on the sphere [6]. Moreover, the collocation via rational interpolation might be useful to approximate the solution of partial differential equation, especially for functions with steep gradients and discontinuities.

N	h	Method	ε	RMSE	AD	λ
289	8.17E − 02	RSRBF-PU	2	1.83E − 02	1.00E + 00	–
		RSRBF-PU	6	4.60E − 02	5.32E − 01	–
		RVSK-PU	–	2.57E − 02	5.73E − 01	–
1089	4.43E − 02	RSRBF-PU	2	3.79E − 03	1.00E + 00	2.32
		RSRBF-PU	6	9.41E − 03	5.32E − 01	2.34
		RVSK-PU	–	3.75E − 03	5.04E − 01	2.84
4225	2.93E − 02	RSRBF-PU	2	1.87E − 03	1.00E + 00	1.70
		RSRBF-PU	6	4.03E − 03	5.33E − 01	2.05
		RVSK-PU	–	6.60E − 04	5.76E − 01	4.19
16641	1.30E − 02	RSRBF-PU	2	5.35E − 04	1.00E + 00	1.54
		RSRBF-PU	6	1.21E − 03	5.35E − 01	1.35
		RVSK-PU	–	3.00E − 04	5.76E − 01	0.97

Table 3: Fill distances, RMSEs, ADs and convergence rates computed on noisy Halton points. Results are obtained using f_3 as test function and the Wendland's C^2 kernel to construct local approximant.

9. Acknowledgments

This research has been accomplished within Rete ITaliana di Approssimazione (RITA) and partially supported by the GNCS-INdAM funds 2017. The first author is partially supported by the research project *Approximation by radial basis functions and polynomials: applications to CT, MPI and PDEs on manifolds*, No. DOR1695473. The second author is partially supported by the GNCS project *Numerical methods for constrained optimization problems*. The third author is supported by the research project *Radial basis functions approximations: stability issues and applications*, No. BIRD167404.

References

[1] N. Aronszajn, *Theory of Reproducing Kernels*, Trans. Amer. Math. Soc. **68** (1950), pp. 337–404.
[2] L. Bergamaschi, G. Gambolati, G. Pini, *Asymptotic convergence of conjugate gradient methods for the partial symmetric eigenproblem*, Numer. Linear Algebra Appl. **4** (1997), pp. 69–84.
[3] L. Bergamaschi, M. Putti, *Numerical comparison of iterative eigensolvers for large sparse symmetric matrices*, Comp. Methods App. Mech. Engrg. **191** (2002), pp. 5233–5247.
[4] M. Bozzini, L. Lenarduzzi, M. Rossini, R. Schaback, *Interpolation with variably scaled kernels*, IMA J. Numer. Anal. **35** (2015), pp. 199–219.
[5] R. Cavoretto, S. De Marchi, A. De Rossi, E. Perracchione, G. Santin, *Partition of unity interpolation using stable kernel-based techniques*, Appl. Numer. Math. **116** (2017), pp. 95–107.
[6] R. Cavoretto, A. De Rossi, *Fast and accurate interpolation of large scattered data sets on the sphere*, J. Comput. Appl. Math. **234** (2010), pp. 1505–1521.
[7] R. Cavoretto, A. De Rossi, E. Perracchione, *Optimal selection of local approximants in RBF-PU interpolation*, to appear on J. Sci. Comput. (2017).
[8] S. De Marchi, G. Santin, *Fast computation of orthonormal basis for RBF spaces through Krylov space methods*, BIT **55** (2015), pp. 949–966.
[9] S. De Marchi, A. Idda, G. Santin, *A rescaled method for RBF approximation*, Approximation theory XV: San Antonio 2016, pp. 3959, Springer Proc. Math. Stat., 201, 2017.
[10] A. De Rossi, E. Perracchione, *Positive constrained approximation via RBF-based partition of unity method*, J. Comput. Appl. Math. **319** (2017), pp. 338–351.
[11] T.A. Driscoll, *Algorithm 843: Improvements to the Schwarz-Christoffel toolbox for* Matlab, ACM Trans. Math. Softw. **31** (2005), pp. 239–251.
[12] G.E. Fasshauer, M.J. McCourt, *Kernel-based Approximation Methods Using* Matlab, World Scientific, Singapore, 2015.
[13] G.E. Fasshauer, *Meshfree Approximations Methods with* Matlab, World Scientific, Singapore, 2007.
[14] B. Fornberg, E. Larsson, N. Flyer, *Stable computations with Gaussian radial basis functions*, SIAM J. Sci. Comput. **33** (2011), pp. 869–892.
[15] G.H. Golub, C. Reinsch, *Singular value decomposition and least squares solutions*, Numer. Math. **14** (1970), pp. 403–420.
[16] J.H. Halton, *On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals*, Numer. Math. **2** (1960), pp. 84–90.
[17] X.G. Hu, T.S. Ho, H. Rabitz, *Rational approximation with multidimensional scattered data*, Phys. Rev. **65** (2002), pp. 035701-1–035701-4.
[18] S. Jakobsson, B. Andersson, F. Edelvik, *Rational radial basis function interpolation with applications to antenna design*, J. Comput. Appl. Math. **233** (2009), pp. 889–904.
[19] R. Lehmensiek, P. Meyer, *Creating accurate multivariate rational interpolation models of microwave circuits by using efficient adaptive sampling to minimize the number of computational electromagnetic analyses*, IEEE Trans. Microw. Theory Tech. **49** (2001), pp. 1419–1430.
[20] R.B. Lehoucq, D.C. Sorensen, *Deflation techniques for an implicitly restarted Arnoldi iteration,* SIAM J. Matrix Anal. Appl. **17** (1996), pp. 789–821.
[21] R.B. Lehoucq, D.C. Sorensen, C.Yang, *ARPACK Users Guide: Solution of Large Scale Eigenvalue Problems by Implicitly Restarted Arnoldi Methods*, (1997).
[22] H. Padé, *Sur la répresentation approchée d'une fonction par des fractions rationelles*, Thesis, Ann. École Nor. **9** (1892), pp. 1–93.
[23] S.A. Sarra, Y. Bay, *A rational radial basis function method for accurately resolving discontinuities and steep gradients*, preprint, 2017.
[24] R. Schaback, *Native Hilbert spaces for radial basis functions I*, in: M. D. Buhmann et al. (Eds.), *New Developments in Approximation Theory*, Int. Series of Numerical Mathematics 132. Birkh$^{\epsilon}$auser Verlag 199, pp. 255–282.
[25] J. Stoer, *Einführung in die Numerische Mathematik I*, Springer-Verlag, 1972.
[26] H. Wendland, *Scattered Data Approximation*, Cambridge Monogr. Appl. Comput. Math., vol. 17, Cambridge Univ. Press, Cambridge, 2005.
[27] H. Wendland, *Fast evaluation of radial basis functions: Methods based on partition of unity*, in: C.K. Chui et al. (Eds.), Approximation Theory X: Wavelets, Splines, and Applications, Vanderbilt Univ. Press, Nashville, 2002, pp. 473–483.