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, Italy

ARTICLE INFO

MSC:
65D05
65D15
65D17

Keywords:
Meshfree approximation
Partition of unity method
Rational RBF approximation

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 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.

1. Introduction

It is well-known that, for univariate functions with steep gradients, approximation by rational functions is particularly suitable due to their smoothing properties and less oscillatory behavior than polynomials. Let p_1 and p_2 be two univariate polynomials of degrees $m \geq 0$ and $n \geq 1$ respectively, a $[m/n]$ rational function is

$$t(x) = \frac{p_1(x)}{p_2(x)} = \frac{a_m x^m + \cdots + a_0}{b_n x^n + \cdots + 1}.$$

The particular choice of the polynomial p_2 allows to reduce the unknown coefficients to $m + n + 1$. These coefficients can be found by computing the *Padé approximant of order* $[m/n]$ [1] or by a least square approach. However, extension of polynomial approximations to high dimensions is a challenging problem (refer e.g. to [2,3]) and, because of their dependence on meshes, it is not easy to implement for complex shaped domains. These are the main reasons why we direct our interest to rational Radial Basis Function (RBF)-based schemes. In particular, taking advantage of being meshfree, they are easy to implement in high dimensions.

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 first presented in [4] and then developed also in [5,6], we introduce additional constraints. The scheme, that reduces to a largest eigenvalue problem, is extended to work with the Partition of Unity (PU) method [7], allowing to overcome the usually high 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

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

https://doi.org/10.1016/j.cam.2018.07.020

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, Italy

ARTICLE INFO

MSC:
65D05
65D15
65D17

Keywords:
Meshfree approximation
Partition of unity method
Rational RBF approximation

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 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.

1. Introduction

It is well-known that, for univariate functions with steep gradients, approximation by rational functions is particularly suitable due to their smoothing properties and less oscillatory behavior than polynomials. Let p_1 and p_2 be two univariate polynomials of degrees $m \geq 0$ and $n \geq 1$ respectively, a $[m/n]$ rational function is

$$t(x) = \frac{p_1(x)}{p_2(x)} = \frac{a_m x^m + \cdots + a_0}{b_n x^n + \cdots + 1}.$$

The particular choice of the polynomial p_2 allows to reduce the unknown coefficients to $m + n + 1$. These coefficients can be found by computing the *Padé approximant of order* $[m/n]$ [1] or by a least square approach. However, extension of polynomial approximations to high dimensions is a challenging problem (refer e.g. to [2,3]) and, because of their dependence on meshes, it is not easy to implement for complex shaped domains. These are the main reasons why we direct our interest to rational Radial Basis Function (RBF)-based schemes. In particular, taking advantage of being meshfree, they are easy to implement in high dimensions.

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 first presented in [4] and then developed also in [5,6], we introduce additional constraints. The scheme, that reduces to a largest eigenvalue problem, is extended to work with the Partition of Unity (PU) method [7], allowing to overcome the usually high 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

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

https://doi.org/10.1016/j.cam.2018.07.020

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. [8]). 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 [7,9]. 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 [10–12]. 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) [13]. The VSKs, via a scale function, transform the original problem so that the usual ill-conditioning of the kernel matrix is sensibly reduced.

This investigation reveals that the new method, namely Rational VSK-PU (RVSK-PU), performs better than the classical PU [9] or the rescaled PU method [14], 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 $R : \Omega \longrightarrow \mathbb{R}$ such that $R(\boldsymbol{x}_i) = f_i$, $i = 1, \dots, N$. To this end, we consider $R \in \text{span}\{\Phi(\cdot, \boldsymbol{x}_i), \boldsymbol{x}_i \in \mathcal{X}_N\}$, where $\Phi : \Omega \times \Omega \longrightarrow \mathbb{R}$ is a strictly positive definite and symmetric kernel. The interpolant assumes then the form

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

The coefficients $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_N)^T$ in (1) are found 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.

Definition 2.1. A function $\sigma : \mathbb{R}^M \to \mathbb{R}$ is called radial, if there exists a continuous function $\phi : [0, \infty) \to \mathbb{R}$ such that $\sigma(\boldsymbol{x}) = \phi(r)$ with $r = \|\boldsymbol{x}\|_2$.

Using this definition, given a "basic" function ϕ and the (symmetric positive definite) kernel Φ, for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^M$ we have a radial kernel just by considering $\Phi(\boldsymbol{x}, \boldsymbol{y}) = \phi(\|\boldsymbol{x} - \boldsymbol{y}\|_2)$. In what follows, we can indifferently use Φ or ϕ by referring to Definition 2.1. An example of RBF is the well-known Gaussian function $\phi(r) = \mathrm{e}^{-\varepsilon^2 r^2}$, where ε is the shape parameter, which is indeed a scale parameter.

Moreover, to each positive definite and symmetric kernel Φ we associate a real Hilbert space $\mathcal{N}_\Phi(\Omega)$ called the *native space*, as follows. Firstly we consider $H_\Phi(\Omega) = \text{span}\{\Phi(\cdot, \boldsymbol{x}), \boldsymbol{x} \in \Omega\}$, equipped with the bilinear form $(\cdot, \cdot)_{H_\Phi(\Omega)}$ Since $H_\Phi(\Omega)$ is a pre-Hilbert space with reproducing kernel Φ, the native space $\mathcal{N}_\Phi(\Omega)$ of Φ is then its completion with respect to the norm $\|\cdot\|_{H_\Phi(\Omega)}$, that is $\|f\|_{H_\Phi(\Omega)} = \|f\|_{\mathcal{N}_\Phi(\Omega)}$, for all $f \in H_\Phi(\Omega)$ (cf. [9,15]).

For RBF interpolation, we report a first error bound in terms of the well-known *power function* $P_{\Phi, \mathcal{X}_N}$ (see e.g. [9, Th. 14.2, p.117]).

Theorem 2.1. *Let $\Phi \in C(\Omega \times \Omega)$ be a strictly positive definite kernel. Suppose that $\mathcal{X}_N = \{\boldsymbol{x}_i, i = 1, \dots, N\} \subseteq \Omega$ have distinct points. For all $f \in \mathcal{N}_\Phi(\Omega)$ we have*

$$|f(\boldsymbol{x}) - R(\boldsymbol{x})| \leq P_{\Phi, \mathcal{X}_N}(\boldsymbol{x}) \|f\|_{\mathcal{N}_\Phi(\Omega)}, \quad \boldsymbol{x} \in \Omega.$$

To give more generic error bounds we introduce common indicators of data regularity, the fill distance and the separation distances, denoted by $h_{\Omega, \mathcal{X}_N}$ and $q_{\mathcal{X}_N}$ respectively, and defined as

$$h_{\Omega, \mathcal{X}_N} := \sup_{\boldsymbol{x} \in \Omega} \left(\min_{\boldsymbol{x}_k \in \mathcal{X}_N} \|\boldsymbol{x} - \boldsymbol{x}_k\|_2 \right), \quad q_{\mathcal{X}_N} = \frac{1}{2} \min_{i \neq k} \|\boldsymbol{x}_i - \boldsymbol{x}_k\|_2 .$$

In what follows we use the simplified notation $h_{\mathcal{X}_N}$. The next theorem [15, Th. 3.14, p. 33] provides the polynomial precision of the interpolation process with RBF.

Theorem 2.2. *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 $h_{\mathcal{X}_N} \leq h_0$, 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 in terms of the fill distance, we need to consider the space $C^k_\nu(\mathbb{R}^M)$ of all functions $f \in C^k$ whose derivatives of order $|\boldsymbol{\mu}| = k$ satisfy $D^{\boldsymbol{\mu}} f(\boldsymbol{x}) = \mathcal{O}\left(\|\boldsymbol{x}\|_2^\nu\right)$ for $\|\boldsymbol{x}\|_2 \longrightarrow 0$.

The next statement is given for strictly positive definite functions but it holds also for conditionally positive definite functions. However, in that case we would only have a native space semi-norm. Notice that the result can be extended to the derivatives of the interpolant.

Theorem 2.3 (*[15, Th. 11.11, p. 181]*)**.** *Suppose $\phi \in C^k_\nu(\mathbb{R}^M)$ 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 $\mathcal{X}_N = \{\boldsymbol{x}_i, i = 1, \dots, N\} \subseteq \Omega$ satisfying $h_{\mathcal{X}_N} \leq h_0$, there exists a constant C such that*

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

where $h_0 = \gamma/C_2$, with C_2 from Theorem 2.2.

3. Rational RBF interpolation

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 [2]).

3.1. Generality

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 $\mathcal{X}_N \subseteq \Omega (\subseteq \mathbb{R}^M)$, a natural extension of the classical RBF approximation to the rational case is

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

provided $R^{(2)}(\boldsymbol{x}) \neq 0$, $\boldsymbol{x} \in \Omega$. 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, we can consider the points in $\mathcal{X}_m$ and $\mathcal{X}_n$ in a sequential way and trivially re-order them if needed.

To solve the rational problem, we need to determine $n + m$ coefficients and if $m + n < N$, we can find a least square solution, 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 [4]) 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 [16]). Indeed, consider a simple example which is also devoted to show the robustness of rational RBF in case of functions with discontinuities or steep gradients. Suppose $m + n = N$. Imposing the interpolation conditions

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

leads to a squared and homogeneous system of equations $B\boldsymbol{\xi} = \boldsymbol{0}$, where $\boldsymbol{\xi} = (\boldsymbol{\alpha}, \boldsymbol{\beta})^T, \boldsymbol{0} \in \mathbb{R}^N$ is the null vector and the entries of $B = (B_1 \quad B_2) \in \mathbb{R}^{N \times N}$ are given by $(B_1)_{ir} = \Phi(\boldsymbol{x}_i, \boldsymbol{x}_r)$ and $(B_2)_{is} = -f_i \Phi(\boldsymbol{x}_i, \boldsymbol{x}_r)$, with $i = 1, \dots, N$, $r = k, \dots, k + m - 1$, and $s = j, \dots, j + n - 1$. Depending on $\mathcal{X}_m$, $\mathcal{X}_n$ and on the function values, we might have no cancellations and therefore the matrix can be non-singular. For instance, 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 $m = n = 13$. 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 [17]

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

In Fig. 1 (left) we plot the function reconstructed via rational RBF by solving (4) and the one found with the classical RBF interpolation.

As second test case, we take the piecewise constant function considered in Fig. 1 (right) for which we fix $\mathcal{X}_m = \{\boldsymbol{x}_i, i = 1, \dots, m\}$ and $\mathcal{X}_n = \{\boldsymbol{x}_i, i = m + 1, \dots, N\}$, with $m = 13$. Again, the solution in a least square sense is given by (4).

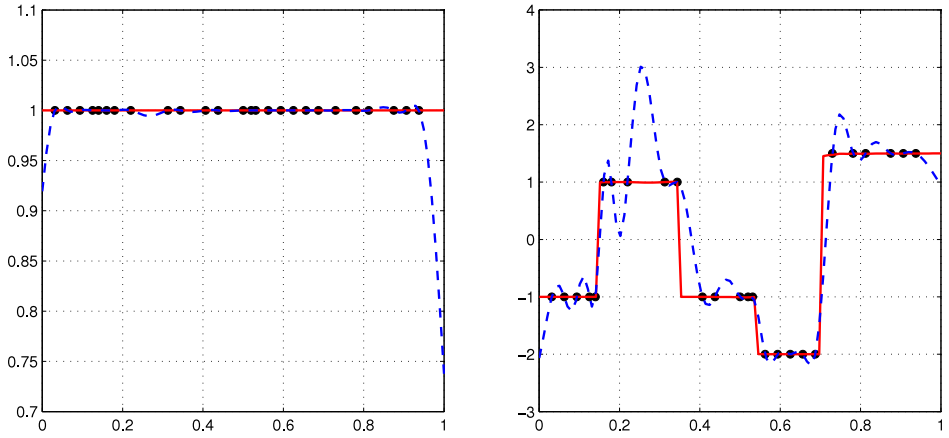

Fig. 1. The black dots represent the set of scattered data, the red solid line and the blue dotted one are the curves reconstructed via the rational and classical RBF approximation, respectively.

These examples aim to point out that the rational RBF problem is not well-posed and that the method that we are going to describe enables us to better reconstruct functions with steep gradients. As a remark, we point out that the latter fact is also due to the use of VSKs, which, beside providing stable solutions, are effective for functions with discontinuities; refer e.g. to [18].

In the next subsection, following [4], we show that by adding further constraints to the interpolation conditions, we can uniquely determine $R^{(1)}$ and $R^{(2)}$.

3.2. Well-posedness

To deal with well-posed problems, according to [4], we can impose extra constraints. In doing so, we focus on the case $m = n = N$. In this case the rational interpolant can be rewritten as

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

Imposing the interpolation conditions requires to solve the homogeneous system $(A - DA)\boldsymbol{\xi} = \mathbf{0}$, where $D = \mathrm{diag}(f_1, \ldots, f_N)$ and A is the standard kernel matrix. The system is underdetermined and so we add extra conditions. To get a well-posed problem a possible way is to find a vector $\boldsymbol{q} = (R^{(2)}(\boldsymbol{x}_1), \ldots, R^{(2)}(\boldsymbol{x}_N))^T$ so that we can construct its corresponding RBF interpolant $R^{(2)}$ in the standard way. Once we have $\boldsymbol{q}$, we then define $R^{(1)}$ so that it interpolates the function values $\boldsymbol{p} = D\boldsymbol{q}$. This means that if $\boldsymbol{p}$ and $\boldsymbol{q}$ are given, then determining the rational interpolant reduces to solving

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

The existence and uniqueness of the solutions of (6) easily follows from the fact that the kernel we consider is strictly positive definite.

Hence, the problem now consists in determining the vectors $\boldsymbol{p}$ and $\boldsymbol{q}$. To this aim, first note that it is reasonable to select $R^{(1)}$ and $R^{(2)}$ such that the sum of their native space norms at the data sites is as small as possible. Following [4] this leads to finding

$$\min_{\substack{R^{(1)}, R^{(2)} \in \mathcal{N}_\Phi(\Omega), \\ c_3\|\boldsymbol{p}\|_2^2 + c_4\|\boldsymbol{q}\|_2^2 = 1, \\ R^{(1)}(\boldsymbol{x}_k) = f_k R^{(2)}(\boldsymbol{x}_k).}} \left(c_1 \|R^{(1)}\|^2_{\mathcal{N}_\Phi(\Omega)} + c_2 \|R^{(2)}\|^2_{\mathcal{N}_\Phi(\Omega)} \right), \tag{7}$$

where a discussion on the selection of the parameters c_i, $i = 1, \ldots, 4$, is provided too. Here, in analogy with (4), we fix $c_1 = c_3 = 1/\|\boldsymbol{f}\|_2^2$ and $c_2 = c_4 = 1$.

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

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

Therefore, the problem (7) reduces in solving

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

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 [4,6]

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

where I_N is the $N \times N$ identity matrix.

Taking into account that $R^{(2)}(\boldsymbol{x}_i) = \boldsymbol{q}_i$, $i = 1, \ldots, N$, the rational RBF interpolant can be rewritten as

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

where the kernel Φ_R is strictly positive definite, provided Φ is and $R^{(2)}(\boldsymbol{x}) \neq 0$, $\boldsymbol{x} \in \Omega$ (see [14,19] 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 [14] and $\mathscr{R}$ is indeed a rescaled interpolant. Vice versa, the rescaled interpolant in [14], based on constructing $R^{(2)}(\boldsymbol{x})$ so that it interpolates the constant function $g(x) \equiv 1$, can be seen as a particular case of the method here proposed. Finally, note that in this paper we assume $R^{(2)}(\boldsymbol{x}) \neq 0$, $\boldsymbol{x} \in \Omega$. If it does not hold, it is sufficient to set the following constraints to the minimization problem: $q_i > 0$, $i = 1, \ldots, N$. Moreover, once we have such function values, we can use the method proposed in [20] to obtain a positive approximant.

In summary, to construct and evaluate the rational RBF interpolant, we need to solve the systems (6) and use formula (5) for its evaluation.

This also says that the rational RBF interpolant is constructed by means of the standard RBF interpolation matrix A. This observation enables us to give error bounds. 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*

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

Proof. For any $\boldsymbol{x} \in \Omega$ we have

$$|f(\boldsymbol{x}) - \mathscr{R}(\boldsymbol{x})| = |(R^{(2)}(\boldsymbol{x}) f(\boldsymbol{x}) - q(\boldsymbol{x}) f(\boldsymbol{x})) + (q(\boldsymbol{x}) f(\boldsymbol{x}) - R^{(1)}(\boldsymbol{x}))| / |R^{(2)}(\boldsymbol{x})|.$$

By Theorem 2.1, $\left|R^{(2)}(\boldsymbol{x}) - q(\boldsymbol{x})\right| \leq P_{\Phi, \mathcal{X}_N}(\boldsymbol{x}) \|q\|_{\mathcal{N}_\Phi(\Omega)}$ and $\left|R^{(1)}(\boldsymbol{x}) - q(\boldsymbol{x}) f(\boldsymbol{x})\right| \leq P_{\Phi, \mathcal{X}_N}(\boldsymbol{x}) \|p\|_{\mathcal{N}_\Phi(\Omega)}$, we obtain

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

Remark 3.1. Proposition 3.1 bounds the pointwise error in terms of both the power function and data values, as for the classical RBF interpolant. Furthermore, we have the following proposition.

Proposition 3.2. *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 $\mathcal{X}_N = \{\boldsymbol{x}_i, i = 1, \ldots, N\} \subseteq \Omega$ satisfying $h_{\mathcal{X}_N} \leq h_0$, there exists a constant C such that*

$$\|f - \mathscr{R}\|_{L_\infty(\Omega)} \leq \frac{C}{\|R^{(2)}\|_{L_\infty(\Omega)}} (\|f\|_{L_\infty(\Omega)} \|q\|_{\mathcal{N}_\Phi(\Omega)} + \|p\|_{\mathcal{N}_\Phi(\Omega)}) h_{\mathcal{X}_N}^{(k+\nu)/2},$$

where $h_0 = \gamma / C_2$, with C_2 from Theorem 2.2.

Proof. We have that $\|f - \mathscr{R}\|_{L_\infty(\Omega)} \leq 1/\|R^{(2)}\|_{L_\infty(\Omega)} \left(\|f\|_{L_\infty(\Omega)} \|R^{(2)} - q\|_{L_\infty(\Omega)} + \|R^{(1)} - qf\|_{L_\infty(\Omega)}\right)$. Then, by using Theorem 2.3, the thesis follows.

The main drawback of the method described in this section is the computational cost needed to solve potentially large linear systems. For instance, when we deal with real applications, we often have to interpolate huge data sets and in those cases the computational cost is prohibitive. This is the main reason that suggests to use a Partition of Unity 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 fulfill the following properties [7]:

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 [7].

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 $\text{supp}(W_j) \subseteq \Omega_j$. For instance, such conditions are fulfilled by the so-called Shepard's weights (see e.g. [9]). Then, the rational RBF-PU interpolant assumes the form

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

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

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

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

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 $\boldsymbol{f}_j = (f_1^j, \dots, f_N^j)^T$ are the local function values, $D_j = \text{diag}(f_1^j, \dots, f_N^j)$ and A_j is the standard local kernel matrix. Once $\boldsymbol{p}_j$ and $\boldsymbol{q}_j$ are found 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 $\mathcal{R}_l(\boldsymbol{x}_k) = \mathcal{R}_m(\boldsymbol{x}_k)$, making the problem consistent.

For the rational RBF-PU approximant, we can state following error bound.

Proposition 4.1. *Suppose $\phi \in C_\nu^k(\mathbb{R}^M)$ 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, such that, if p_t and $q_t \in \mathcal{N}_\Phi(\Omega_t)$, for all $\boldsymbol{x} \in \Omega$ we have:*

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

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 2.2 applied to the local setting. We denote by $f_{|\Omega_t}$ the restriction of f on Ω_t.

Proof. 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

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

Furthermore, we can apply Proposition 3.2 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 $h_{\mathcal{X}_{N_t}} \leq h_0$, then there exists a constant C such that

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

Remark 4.2. If we compare the result reported in Proposition 4.1 with the error estimate shown in Theorem 2.3, 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, which are linked to the decrease of the separation distance and not necessarily to the increase of the number of points. Thus, we carry out a stable computation via VSKs, which, as recently proven, are also extremely performing with functions showing singularities [18].

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 infinitely 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 [9]). 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 [12,21] for further details on these techniques. Finally, we remark that there are also two other classes of stable methods, namely the Weighted SVD (WSVD) [10,11] and the VSKs [13] that properly work with any RBF.

Here, we propose a stable method for the solution of the rational RBF problem via VSKs. The idea, which will be reviewed in what follows, is from [13] and further analyzed for functions with discontinuities [18]. 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. [13, 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 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 [13] 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 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 Theorem 2.3 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 underlying some computational aspects.

`Step 1: Construction of the PU framework`

In the PU setting, at first we need to define the set of evaluation points $\mathcal{E}_s = \{\tilde{\boldsymbol{x}}_i, i = 1, \ldots, s\} \subseteq \Omega$ and the set of PU centers $\mathcal{C}_d = \{\tilde{\boldsymbol{x}}_i, j = 1, \ldots, d\} \subseteq \Omega$. Both these sets are constructed as grids of points on Ω. Furthermore, in Section 4, we require that the subdomains $\Omega_j, j = 1, \ldots, d$, form an open, bounded and regular covering for Ω. As subdomains we consider balls of a certain radius δ. In particular, we know that the number of patches d should be proportional to N and following [9] a suitable choice is given by $N/d \approx 2^M$.

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 $\delta_0 = (1/d)^{1/M}$. The choice of the radius affects both accuracy and efficiency of the interpolation process. For large radii we might need to invert huge local matrices and thus we partially lose the efficiency coming from the PU method. On the other hand, taking a radius of small size can lead to empty patches containing very few points and therefore to inaccurate solutions. For more details the reader can refer to [22]. Few tests by varying the patch size have been carried out.

Once the PU subdomains are generated, the whole problem reduces to solving for each patch a local rational interpolation problem. Specifically, for the jth 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 [22].

`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 [23]. 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 [24] 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* $\mathcal{Q}(\boldsymbol{z}_j) = \boldsymbol{z}_j^T \Lambda_{\psi_j} \boldsymbol{z}_j / \boldsymbol{z}_j^T \Theta_j \boldsymbol{z}_j$, $\boldsymbol{z}_j \in \mathbb{R}^{N_j}$, 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 [8,23], 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 straightforward. 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 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 [25] and Non-Conformal points [26]. 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):

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

We also estimate the empirical convergence rate via the formula:

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

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

Remark 7.1. Using VSKs can lead to small perturbations 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

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

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 v. In what follows we fix $v = 3$. In general, since the aim consists in increasing the separation distance to improve the stability, *small* values for v 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 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 $w_k = \|\Lambda_{\psi_j}\boldsymbol{z}_j^k - \mathcal{Q}(\boldsymbol{z}_j^k)\Theta_j\boldsymbol{z}_j^k\|/\|\boldsymbol{z}_j^k\|_{\Theta_j}$, 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 terms of computational time. We remark that the IRLM is implemented in MATLAB by the function `eigs` (see also ARPACK [27]).

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

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.

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

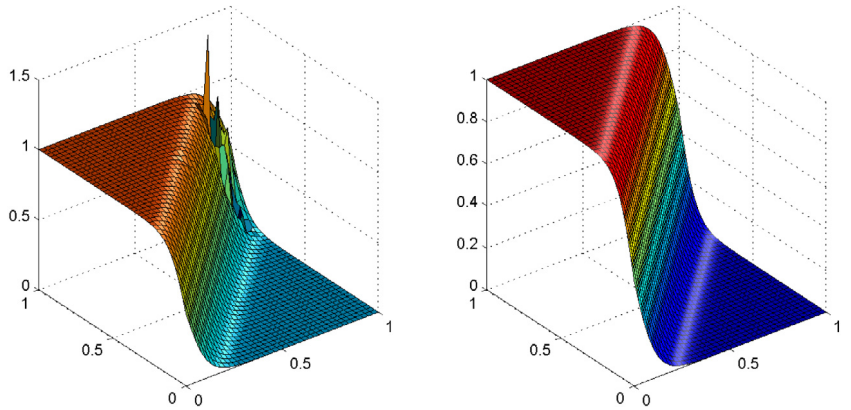

Fig. 2. The approximate surface f_1 obtained with the SRBF-PU (left) and RVSK-PU (right) methods with $N = 1089$ Halton data. Results are computed using the Gaussian C^∞ kernel as local approximant.

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 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 Fig. 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 [9]. 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.

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 Fig. 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 Fig. 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.

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 [22], 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

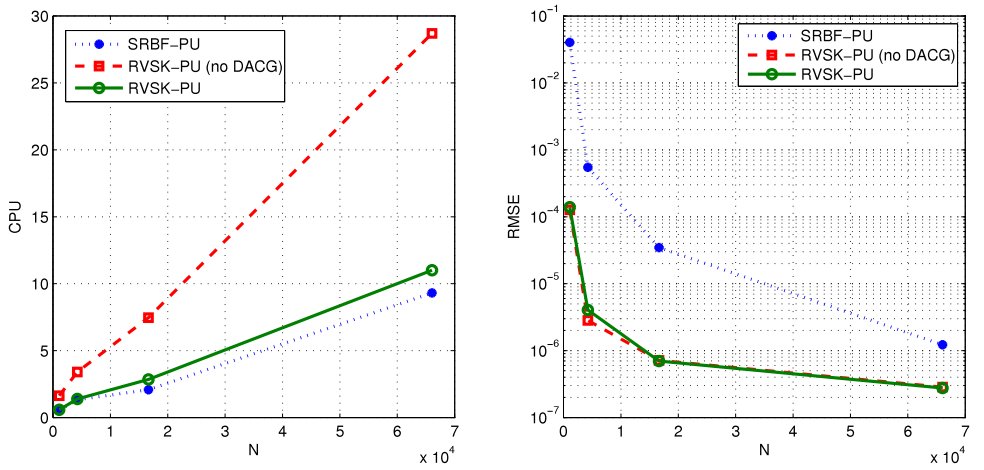

Fig. 3. Left: the number of points N versus the CPU times for the SRBF-PU, RVSK-PU and RVSK-PU (no DACG), i.e. the RVSK-PU computed with the IRLM scheme. Right: the number of points N versus the logarithmic scale of the RMSE for the SRBF-PU, RVSK-PU and RVSK-PU (no DACG).

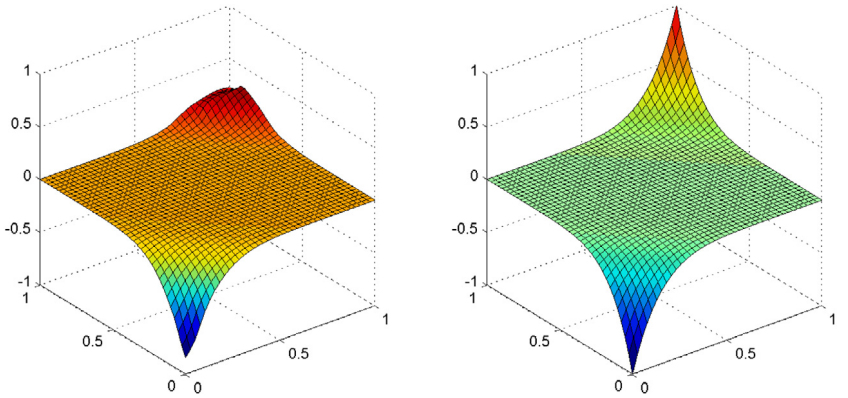

Fig. 4. The approximate surface f_2 obtained with the SRBF-PU (left) and RVSK-PU (right) methods with $N = 1089$ Non-Conformal points. Results are computed using the IMQ C^∞ kernel as local approximant.

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.

N	h	Method	ε	RMSE	AP	λ
1 089	1.27E−01	SRBF-PU	4	3.94E−02	58	–
		RVSK-PU	–	1.32E−04	59	–
4 225	7.63E−02	SRBF-PU	4	3.32E−02	63	0.33
		RVSK-PU	–	2.76E−06	63	7.59
16 641	5.93E−02	SRBF-PU	4	5.15E−02	68	−1.74
		RVSK-PU	–	1.65E−06	69	2.26
66 049	3.62E−02	SRBF-PU	4	5.70E−02	62	−0.20
		RVSK-PU	–	2.93E−07	62	3.38

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 Fig. 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.

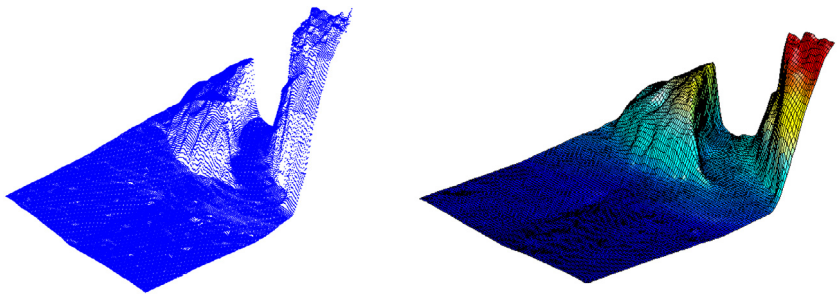

Fig. 5. The 3D view of the Montebelluna (left) data. The surface reconstructed via the RVSK-PU (right) method.

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.

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	–
1 089	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
4 225	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
16 641	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

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 Wendland's C^2 kernel. Therefore, we carry out a comparison with the rescaled method proposed in [14], 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.[1] The data set consists of 62 677 grid data and among them we take 2726 points to test the accuracy via cross-validation, only taking the remaining 59 951 for the interpolation. For a 3D view refer to Fig. 5 (left).

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

[1] At this link the user needs to perform a query search for the digital data of Montebelluna.

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 [28] and also in finding suitable locations for the PU centers when considering clustered data. Moreover, the collocation via rational interpolation might be useful to approximate the solution of partial differential equations.

Acknowledgments

The authors sincerely thank the reviewers for their insightful comments. This research has been accomplished within Rete ITaliana di Approssimazione (RITA) and partially supported by the GNCS-INdAM funds 2017. The first author has been 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 has been partially supported by the GNCS project *Numerical methods for constrained optimization problems*. The third author has been supported by the research projects *Radial basis functions approximations: stability issues and applications*, No. BIRD167404 and *Data assimilation methods for machine learning with applications to environmental data*, founded by UE ERA-PLANET GA n. 689443.

References

[1] H. Padé, Sur la Répresentation Approchée D'une Fonction par des Fractions Rationelles, Vol. 9 (thesis), Ann. École Nor., 1892, pp. 1–93.
[2] X.G. Hu, T.S. Ho, H. Rabitz, Rational approximation with multidimensional scattered data, Phys. Rev. 65 (2002) 035701–1–035701–4.
[3] 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) 1419–1430.
[4] S. Jakobsson, B. Andersson, F. Edelvik, Rational radial basis function interpolation with applications to antenna design, J. Comput. Appl. Math. 233 (2009) 889–904.
[5] E. Perracchione, Rational RBF-based partition of unity method for efficiently and accurately approximating 3D objects, J. Comput. Appl. Math. (2018) 1–16.
[6] S.A. Sarra, Y. Bay, A rational radial basis function method for accurately resolving discontinuities and steep gradients, Appl. Numer. Math. 130 (2018) 131–142.
[7] 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.
[8] L. Bergamaschi, G. Gambolati, G. Pini, Asymptotic convergence of conjugate gradient methods for the partial symmetric eigenproblem, Numer. Linear Algebra Appl. 4 (1997) 69–84.
[9] G.E. Fasshauer, Meshfree Approximations Methods with Matlab, World Scientific, Singapore, 2007.
[10] 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) 95–107.
[11] S. De Marchi, G. Santin, Fast computation of orthonormal basis for RBF spaces through Krylov space methods, BIT 55 (2015) 949–966.
[12] B. Fornberg, E. Larsson, N. Flyer, Stable computations with Gaussian radial basis functions, SIAM J. Sci. Comput. 33 (2011) 869–892.
[13] M. Bozzini, L. Lenarduzzi, M. Rossini, R. Schaback, Interpolation with variably scaled kernels, IMA J. Numer. Anal. 35 (2015) 199–219.
[14] S. De Marchi, A. Idda, G. Santin, A rescaled method for RBF approximation, in: Approximation Theory XV: San Antonio 2016, in: Springer Proc. Math. Stat., vol. 201, 2017, pp. 39–59.
[15] H. Wendland, Scattered Data Approximation, in: Cambridge Monogr. Appl. Comput. Math., vol. 17, Cambridge Univ. Press, Cambridge, 2005.
[16] J. Stoer, Einführung in Die Numerische Mathematik I, Springer-Verlag, 1972.
[17] G.H. Golub, C. Reinsch, Singular value decomposition and least squares solutions, Numer. Math. 14 (1970) 403–420.
[18] M. Rossini, Interpolating functions with gradient discontinuities via variably scaled kernels, Dolom. Res. Notes Approx. 11 (2018) 3–14.
[19] N. Aronszajn, Theory of reproducing kernels, Trans. Amer. Math. Soc. 68 (1950) 337–404.
[20] A. De Rossi, E. Perracchione, Positive constrained approximation via RBF-based partition of unity method, J. Comput. Appl. Math. 319 (2017) 338–351.
[21] G.E. Fasshauer, M.J. McCourt, Kernel-based Approximation Methods using Matlab, World Scientific, Singapore, 2015.
[22] R. Cavoretto, A. De Rossi, E. Perracchione, Optimal selection of local approximants in RBF-PU interpolation, J. Sci. Comput. 74 (2018) 1–22.
[23] L. Bergamaschi, M. Putti, Numerical comparison of iterative eigensolvers for large sparse symmetric matrices, Comput. Methods Appl. Mech. Eng. 191 (2002) 5233–5247.
[24] R.B. Lehoucq, D.C. Sorensen, Deflation techniques for an implicitly restarted Arnoldi iteration, SIAM J. Matrix Anal. Appl. 17 (1996) 789–821.
[25] J.H. Halton, On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals, Numer. Math. 2 (1960) 84–90.
[26] T.A. Driscoll, Algorithm 843: Improvements to the Schwarz-Christoffel toolbox for Matlab, ACM Trans. Math. Software 31 (2005) 239–251.
[27] R.B. Lehoucq, D.C. Sorensen, C. Yang, ARPACK Users Guide: Solution of Large Scale Eigenvalue Problems by Implicitly Restarted Arnoldi Methods, 1997.
[28] R. Cavoretto, A. De Rossi, Fast and accurate interpolation of large scattered data sets on the sphere, J. Comput. Appl. Math. 234 (2010) 1505–1521.