CONSTRUCTIVE MATHEMATICAL ANALYSIS
7 (2024), No. SI, pp. 76-92
http://dergipark.org.tr/en/pub/cma
ISSN 2651 - 2939

Research Article

Adaptive residual subsampling algorithms for kernel interpolation based on cross validation techniques

Dedicated to Professor Paolo Emilio Ricci, on occasion of his 80th birthday, with respect and friendship.

ROBERTO CAVORETTO*, ADEEBA HAIDER, SANDRO LANCELLOTTI, DOMENICO MEZZANOTTE, AND AMIR NOORIZADEGAN

ABSTRACT. In this article, we present an adaptive residual subsampling scheme designed for kernel based interpolation. For an optimal choice of the kernel shape parameter we consider some cross validation (CV) criteria, using efficient algorithms of k-fold CV and leave-one-out CV (LOOCV) as a special case. In this framework, the selection of the shape parameter within the residual subsampling method is totally automatic, provides highly reliable and accurate results for any kind of kernel, and guarantees existence and uniqueness of the kernel based interpolant. Numerical results show the performance of this new adaptive scheme, also giving a comparison with other computational techniques.

Keywords: Adaptive interpolation, meshfree methods, radial basis function approximation, shape parameter optimization, cross validation schemes.

2020 Mathematics Subject Classification: 65D05, 65D12, 65D15.

1. INTRODUCTION

In [12] an adaptive residual subsampling method depending on radial basis function (RBF) interpolation is presented. The computational technique guarantees, on the one hand, the non-singularity of the interpolation matrix (and so existence and uniqueness of the interpolant) and, on the other, allows an optimal selection of the kernel shape parameter through application of a maximum profile likelihood estimation criterion. This twofold advantage is obtained by using strictly positive definite kernels that are radially symmetric [19]. Each of these positive features do not usually occur in the original method [18], as well as in its modified version [40]. Indeed, while in [18] the adaptive method is characterized by a kernel shape parameter that can vary at each node, thus breaking the above-mentioned symmetry and accordingly compromising the proof of matrix non-singularity, in [12] this issue is solved by an optimal choice of a *unique* shape parameter for all nodes. As a result, for practical purposes, any kind of radial kernel can be used and the adaptive scheme can be iteratively and automatically applied without any user's action. Other examples of adaptive interpolation algorithms can be found in literature, see e.g. [1, 7, 24, 38] and references therein.

In this work, we propose a change in the decision strategy regarding the choice of the optimal shape parameter associated with the kernel within the residual subsampling method [12]. In this context, we opt for use of some *cross validation* (CV) criteria. In fact, CV is a popular technique in statistics which, instead of the usually unknown solution, makes use of the given

Received: 19.07.2024; Accepted: 01.11.2024; Published Online: 16.12.2024
*Corresponding author: Roberto Cavoretto; roberto.cavoretto@unito.it
DOI: 10.33205/cma.1518603

data to predict optimal values of model parameters for data fitting. The main idea is to split the data into a training set and a validation one, then utilizing some form of error norm obtained by gauging the accuracy of the fit built from information on the training set at points of the validation set [21]. Here, we focus on some possible options of CV algorithms to be applied within the residual subsampling method. More precisely, we consider the k-fold CV formulated as an extended version of Rippa's scheme [30], which includes original Rippa's algorithm [34] as a particular case. The latter is an especially popular version of CV, known as leave-one-out cross validation (LOOCV), and corresponds to using a training set consisting of all of the data points except one, which in turn is the sole member of the validation set. In the setting of kernel or RBF methods the LOOCV scheme appears in several papers such as [7, 22, 26, 36, 37], to name a few. Since the LOOCV method is also efficiently implemented in the MATLAB `crossval` routine [31], it will also be used in this study as a term of comparison for our procedures.

The aim of this work is therefore to introduce a CV criterion for an optimal choice of the kernel shape parameter within the adaptive residual subsampling method. The use of k-fold CV or LOOCV strategies provides greater flexibility and sometimes efficiency than the maximum profile likelihood estimation criterion in [12]. Indeed, this study shows how the CV techniques are valid alternative within residual subsampling schemes, even if – due to several variables involved – is not possible to declare a clear and complete superiority of a specific CV scheme compared to other ones. However, the resulting algorithm allows a totally automatic computation of the shape parameter, i.e., any user's action is not required, either initially, and a single optimal shape parameter is found at any iteration and for each data point set. In this framework, the interpolation problem is well-posed and hence the kernel interpolant exists uniquely, obviously provided that the kernel matrix is positive definite (see e.g. [21]). Moreover, the use of CV techniques has some predictive role to control the ill-conditioning of the interpolation matrix, in particular when in the iterative/adaptive method the number of interpolation points grows. Finally, as our numerical results show, an application of CV criteria formulated in the framework of extended Rippa's scheme generally results in an adaptive interpolation scheme more efficient than commonly used MATLAB routines. The new adaptive method is tested in one and two dimensions and highlights good performance in term of both computational accuracy and efficiency.

The paper is organized as follows. In Section 2, we introduce some preliminaries on multivariate RBF/kernel based interpolation. Section 3 presents CV criteria to find optimal values of the kernel shape parameter in the interpolation method. In Section 4, we describe the adaptive residual subsampling scheme. In Section 5, we report some numerical results, showing accuracy and efficiency of the different CV schemes and providing a comparison with other algorithms. Section 6 concludes this article.

2. Preliminaries

RBF or kernel based methods are powerful and effective tools for multivariate data interpolation. In this section, we introduce some basic notations and a few theoretical results for kernel based interpolation. For further details, we refer the reader to [3, 21, 39].

2.1. **Multivariate interpolation and positive definite functions.**

Scattered data fitting is in general one of the fundamental problems in the field of approximation theory and its applications. In order to have a well-posed problem formulation, we need to recall the concepts of positive definite matrices, and strictly positive definite functions. Indeed, such functions provide a direct entry into meshfree approximation methods [19].

To give a precise definition of the scattered data interpolation problem, we assume to have a finite set $X = \{\boldsymbol{x}_i\}_{i=1}^N \subseteq \Omega$ of data points (or nodes) for some region Ω in $\mathbb{R}^d$, $d \geq 1$, and the corresponding scalar-valued data $f_i \in \mathbb{R}$. These values are often obtained by sampling some (unknown) function f at the data points, i.e. $y_i = f(\boldsymbol{x}_i)$, $i = 1, \ldots, N$. So we are now ready for a precise formulation of the multivariate interpolation problem.

Problem 2.1. *Given data $(\boldsymbol{x}_i, y_i)$, $i = 1, \ldots, N$, with $\boldsymbol{x}_i \in \mathbb{R}^d$, $d \geq 1$, and $y_i \in \mathbb{R}$, find a (continuous) function s such that*

$$s(\boldsymbol{x}_i) = y_i, \qquad i = 1, \ldots, N. \tag{2.1}$$

A suitable and common approach to solving this problem is to take the function s as a linear combination of certain *basis functions* B_j, i.e.,

$$s(\boldsymbol{x}) = \sum_{j=1}^{N} c_j B_j(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Omega. \tag{2.2}$$

Solving the interpolation problem under this assumption leads to a linear system of the form

$$\mathsf{A}\boldsymbol{c} = \boldsymbol{y},$$

where the entries of the *interpolation matrix* A are given by $\mathsf{A}_{ij} = B_j(\boldsymbol{x}_i)$, $i, j = 1, \ldots, N$, $\boldsymbol{c} = (c_1, c_2, \ldots, c_N)^T$, and $\boldsymbol{y} = (y_1, \ldots, y_N)^T$.

Problem 2.1 is *well-posed*, i.e., a solution to a problem exists and is unique, if and only if the matrix A is non-singular.

In order to have basis functions B_j, $j = 1, \ldots, N$, that generate non-singular matrices A for any set of distinct nodes, we recall the special class of positive definite (PD) matrices.

Definition 2.1. *A real symmetric matrix A is called positive semi-definite if its associated quadratic form is non-negative, i.e.,*

$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j \mathsf{A}_{ij} \geq 0 \tag{2.3}$$

for $\boldsymbol{c} = (c_1, \ldots, c_N)^T \in \mathbb{R}^N$. If the quadratic form (2.3) is zero only for $\boldsymbol{c} \equiv \boldsymbol{0}$, then A is called positive definite.

An important property which involves all PD matrices is that, if A is a PD matrix, all its eigenvalues are positive and therefore A is non-singular (but not vice versa). In general, then, it is convenient to consider basis functions B_j of the form (2.2) which are the shifts of a certain function centred at $\boldsymbol{x}_j$, i.e. $B_j(\cdot) = \Phi(\cdot - \boldsymbol{x}_j)$, so that interpolation matrix is positive definite. For this reason, we introduce the concept of *strictly positive definite (SPD) function*.

Definition 2.2. *A complex-valued continuous function $\Phi : \mathbb{R}^d \to \mathbb{C}$ is called positive definite on $\mathbb{R}^d$ if*

$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_i \overline{c_j} \Phi\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) \geq 0 \tag{2.4}$$

for any N pairwise different data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^d$, and $\boldsymbol{c} = (c_1, \ldots, c_N)^T \in \mathbb{C}^N$. The function Φ is called strictly positive definite on $\mathbb{R}^d$ if the quadratic form (2.4) is zero only for $\boldsymbol{c} \equiv \boldsymbol{0}$.

It is also possible to characterize real-valued (S)PD functions using only real coefficients. In fact, Definition 2.2 implies that only functions whose quadratic form is real are candidates

for (S)PD functions. A characterization of such functions is given in the following theoretical result.

Theorem 2.1 ([19]). *A real-valued continuous function Φ is positive definite on $\mathbb{R}^d$ if and only if it is even and*

$$\sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j \Phi\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) \geq 0 \tag{2.5}$$

for any N pairwise different data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^d$, and $\boldsymbol{c} = (c_1, \ldots, c_N)^T \in \mathbb{R}^N$. The function Φ is called strictly positive definite on $\mathbb{R}^d$ if the quadratic form (2.5) is zero only for $\boldsymbol{c} \equiv \boldsymbol{0}$.

A celebrated result on PD functions is the integral characterization given by Bochner's theorem.

Theorem 2.2 (**Bochner**, [19]). *A (complex-valued) function $\Phi \in C(\mathbb{R}^d)$ is positive definite on $\mathbb{R}^d$ if and only if it is the Fourier transform of a finite non-negative Borel measure μ on $\mathbb{R}^d$, i.e.*

$$\Phi(\boldsymbol{x}) = \hat{\mu}(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^d}} \int_{\mathbb{R}^d} e^{-i\boldsymbol{x}\cdot\boldsymbol{y}} d\mu(\boldsymbol{y}), \qquad \boldsymbol{x} \in \mathbb{R}^d.$$

2.2. **Kernel based interpolation.** Given a compact domain $\Omega \subset \mathbb{R}^d$, we assume that the N distinct data points are defined by the set $X = \{\boldsymbol{x}_i\}_{i=1}^N \subseteq \Omega$, while the associated data values are given by $y_i = f(\boldsymbol{x}_i) \in \mathbb{R}$, $i = 1, \ldots, N$, the latter being obtained by sampling some function $f : \Omega \to \mathbb{R}$. For Problem 2.1 we want to determine a function $s : \Omega \to \mathbb{R}$ satisfying the interpolation conditions (2.1).

We can thus express the interpolant s as a linear combination of kernels $\kappa_\varepsilon : \Omega \times \Omega \to \mathbb{R}$ depending on the so-called shape parameter $\varepsilon > 0$, i.e.

$$s(\boldsymbol{x}) = \sum_{j=1}^{N} c_j \kappa_\varepsilon(\boldsymbol{x}, \boldsymbol{x}_j), \qquad \boldsymbol{x} \in \Omega. \tag{2.6}$$

The solution of this interpolation problem results in the symmetric linear system

$$\mathsf{A}_\varepsilon \boldsymbol{c} = \boldsymbol{y}, \tag{2.7}$$

where A_ε is the interpolation (or kernel) matrix with entries $(\mathsf{A}_\varepsilon)_{ij} = \kappa_\varepsilon(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $i, j = 1, \ldots, N$, while $\boldsymbol{c}$ and $\boldsymbol{y}$ are defined as above. Specifically, we remark that if the kernel κ_ε is symmetric and SPD, the matrix A_ε is PD for any data point set X, and the coefficients c_j in (2.6) can uniquely be found.

Starting from the kernel κ_ε in (2.6) we may define a SPD RBF $\phi : \mathbb{R}_0^+ \to \mathbb{R}$ such that

$$\kappa_\varepsilon(\boldsymbol{x}, \boldsymbol{x}_j) = \phi_\varepsilon(||\boldsymbol{x} - \boldsymbol{x}_j||_2) = \phi_\varepsilon(r) := \phi(\varepsilon r), \qquad \forall \boldsymbol{x}, \boldsymbol{x}_j \in \Omega,$$

where $||\cdot||_2$ denotes the Euclidean norm on $\mathbb{R}^d$. It is noteworthy to observe that the choice of a suitable shape parameter ε is usually relevant in radial kernel methods, even if it is known to be a big issue (see e.g. [13, 20, 25, 32], or [21, Chapter 14]). Some popular examples of SPD RBFs are listed below, together with their smoothness degrees and related abbreviations (see

[19]):

$$\phi_\varepsilon(r) = \begin{cases} \exp(-\varepsilon^2 r^2), & \text{Gaussian } C^\infty & \text{GA} \\ (1+\varepsilon^2 r^2)^{-1/2}, & \text{Inverse MultiQuadric } C^\infty & \text{IMQ} \\ \exp(-\varepsilon r)(\varepsilon^3 r^3 + 6\varepsilon^2 r^2 + 15\varepsilon r + 15), & \text{Matérn } C^6 & \text{M6} \\ \exp(-\varepsilon r)(\varepsilon^2 r^2 + 3\varepsilon r + 3), & \text{Matérn } C^4 & \text{M4} \\ \exp(-\varepsilon r)(\varepsilon r + 1), & \text{Matérn } C^2 & \text{M2}. \end{cases}$$

Additionally, the solution of the linear system (2.7) turns out often to be quite sensitive to changes in the data, and the choice of ε can greatly influence the numerical result. A way to measure the computational stability of this interpolation method consists in calculating the condition number of A_ε. As the kernel κ_ε is symmetric and SPD, the conditioning of the kernel matrix A_ε can simply be computed as the ratio between the largest and the smallest eigenvalue ($\lambda_{\max}$ and $\lambda_{\min}$, respectively) of A_ε as:

$$\text{cond}(\mathsf{A}_\varepsilon) = ||\mathsf{A}_\varepsilon||_2 ||\mathsf{A}_\varepsilon^{-1}||_2 = \frac{\lambda_{\max}}{\lambda_{\min}}. \tag{2.8}$$

In order to give some error estimates, we introduce the so-called native space associated with the kernel κ_ε, which is a reproducing kernel Hilbert space $\mathcal{N}_{\kappa_\varepsilon}(\Omega)$ with inner product $(\cdot,\cdot)_{\mathcal{N}_{\kappa_\varepsilon}(\Omega)}$, i.e., $f(\boldsymbol{x}) = (f, \kappa_\varepsilon(\cdot,\boldsymbol{x}))_{\mathcal{N}_{\kappa_\varepsilon}(\Omega)}$, for all $f \in \mathcal{N}_{\kappa_\varepsilon}(\Omega)$ and $\boldsymbol{x} \in \Omega$. Moreover, $H_{\kappa_\varepsilon}(\Omega) = \text{span}\{\kappa_\varepsilon(\cdot,\boldsymbol{x}), \boldsymbol{x} \in \Omega\}$ is a pre-Hilbert space with reproducing kernel κ_ε and equipped with the bilinear form $(\cdot,\cdot)_{\kappa_\varepsilon}$. The native space $\mathcal{N}_{\kappa_\varepsilon}(\Omega)$ of κ_ε is its completion w.r.t. the κ_ε-norm $||\cdot||_{\kappa_\varepsilon}$ so that $||f||_{\kappa_\varepsilon} = ||f||_{\mathcal{N}_{\kappa_\varepsilon}(\Omega)}$ for all $f \in H_{\kappa_\varepsilon}(\Omega)$ (see [19]). Now, we can thus provide a generic error bound in terms of the well-known *power function* $P_{\kappa_\varepsilon,X}$.

Theorem 2.3 ([19]). *Let $\Omega \subseteq \mathbb{R}^d$, $\kappa_\varepsilon \in C(\Omega \times \Omega)$ be strictly positive definite on $\mathbb{R}^d$, and suppose that $X = \{\boldsymbol{x}_i\}_{i=1}^N$ has distinct points. Then, for all $f \in \mathcal{N}_{\kappa_\varepsilon}(\Omega)$, we have*

$$|f(\boldsymbol{x}) - s(\boldsymbol{x})| \le P_{\kappa_\varepsilon,X}(\boldsymbol{x}) ||f||_{\mathcal{N}_{\kappa_\varepsilon}(\Omega)}, \quad \boldsymbol{x} \in \Omega.$$

The first error estimate of Theorem 2.3 can then be improved as shown in the following theorem.

Theorem 2.4 ([19]). *Let $\Omega \subseteq \mathbb{R}^d$ be bounded and satisfy an interior cone condition. Suppose that $\kappa_\varepsilon \in C^{2k}(\Omega \times \Omega)$ is symmetric and strictly positive definite. Then, for all $f \in \mathcal{N}_{\kappa_\varepsilon}(\Omega)$, there exist constants h_0, $C > 0$ (independent of $\boldsymbol{x}$, f and κ_ε) such that*

$$|f(\boldsymbol{x}) - s(\boldsymbol{x})| \le C h_{X,\Omega}^k \sqrt{C_{\kappa_\varepsilon}(\boldsymbol{x})}\, \|f\|_{\mathcal{N}_{\kappa_\varepsilon}(\Omega)},$$

provided $h_{X,\Omega} \le h_0$. Here

$$C_{\kappa_\varepsilon}(\boldsymbol{x}) = \max_{|\boldsymbol{\beta}|=2k,\, \boldsymbol{w},\boldsymbol{z} \in \Omega \cap B(\boldsymbol{x}, c_2 h_{X,\Omega})} \left| D_2^{\boldsymbol{\beta}} \kappa_\varepsilon(\boldsymbol{w},\boldsymbol{z}) \right|,$$

with $B(\boldsymbol{x}, c_2 h_{X,\Omega})$ denoting the ball of radius $c_2 h_{X,\Omega}$ centred at $\boldsymbol{x}$, and $h_{X,\Omega}$ being the fill distance

$$h_{X,\Omega} = \sup_{\boldsymbol{x} \in \Omega} \min_{\boldsymbol{x}_j \in X} ||\boldsymbol{x} - \boldsymbol{x}_j||_2.$$

Theorem 2.4 says that interpolation with a C^{2k} smooth kernel κ_ε has approximation order k. Accordingly, we can deduce that:

(a) for C^∞ SPD kernels, the approximation order k is arbitrarily high;
(b) for SPD kernels with limited smoothness, the approximation order is limited by the kernel smoothness.

For more refined error bounds, the reader may refer to [39].

3. Cross validation criteria for the shape parameter choice

In Section 2, we compute the interpolant s in (2.6) by solving the linear system (2.7), where the kernel matrix A_ε is symmetric and PD. By the uncertainty or trade-off principle [35, 33] we know that using a standard RBF one cannot have high accuracy and stability at the same time. In fact, when the best level of accuracy is typically achieved, i.e., in the flat limit $\varepsilon \to 0$, the interpolation matrix may be very ill-conditioned. A good compromise between numerical accuracy and computational stability needs to be found. It is therefore important to devise suitable techniques that allow us to make a reliable prediction of ε. In fact, the approximation quality is often strongly influenced by the shape parameter, and consequently several strategies have been proposed in the literature for its tuning, see e.g. [8, 15, 23, 36] and [21, Chapter 14].

In this work, we discuss three possible versions of CV algorithms, which we will apply in the residual subsampling method for kernel based interpolation. Hereinafter, firstly we consider the k-fold CV formulated as an extended version of Rippa's algorithm [30], including original Rippa's LOOCV scheme [34] as a particular case (i.e., when $k = N$); then, we refer to the LOOCV method that is implemented in the MATLAB `crossval` routine [31].

3.1. Extended Rippa's scheme.

Supposing to have N data points, in the k-fold CV the data set is divided into k (possibly equal-sized) disjoint subsets, $k \leq N$. Then, iteratively, $k \in \mathbb{N}$ different models are built upon $k-1$ training folds and their performance is evaluated on the respective remaining validation fold. An alternative CV scheme is the so-called leave-p-out cross validation (LpOCV) [16], $p \in \mathbb{N}$, $p < N$, where all possible combinations of p elements of the data set are taken into account as validation set. Since such a computation is very demanding in many situations, k-fold CV is usually preferred. In this work, with an abuse of notation, we refer to LpOCV meaning k-fold CV with $k \approx N/p$. A stochastic extension of extended Rippa's scheme can be found in [29].

To formulate extended Rippa's scheme in the k-fold CV setting [30], considering one of the k folds, we define a vector $\boldsymbol{p} = (p_1, \ldots, p_v)^T$ of distinct validation indices $p_j \in \{1, \ldots, N\}$, $v \in \mathbb{N}$, $v < N$. The data set is subdivided into a training data set $\mathcal{T}$ consisting of $N - v$ points $(\boldsymbol{x}_j, y_j)$ with $j \notin \boldsymbol{p}$, meaning the indices that are not elements of $\boldsymbol{p}$, and a validation data set $\mathcal{V}$ formed by the remaining v points $(\boldsymbol{x}_{p_j}, y_{p_j})$, $j = 1, \ldots, v$.

For a fixed ε, we define the partial RBF interpolant constructed upon $\mathcal{T}$ as

$$s^{[\boldsymbol{p}]}(\boldsymbol{x}) = \sum_{j=1,\ j\notin\boldsymbol{p}}^{N} c_j^{[\boldsymbol{p}]} \kappa_\varepsilon(\boldsymbol{x}, \boldsymbol{x}_j).$$

The column vector $\boldsymbol{c}^{[\boldsymbol{p}]} = (c_j^{[\boldsymbol{p}]})_{j\notin\boldsymbol{p}}$ is found by solving the system of linear equations

$$\mathsf{A}_\varepsilon^{\boldsymbol{p},\boldsymbol{p}} \boldsymbol{c}^{[\boldsymbol{p}]} = \boldsymbol{y}^{\boldsymbol{p}}, \tag{3.9}$$

where $\mathsf{A}_\varepsilon^{\boldsymbol{p},\boldsymbol{p}} = (\mathsf{A}_\varepsilon)_{i,j}$, with $i, j \notin \boldsymbol{p}$, and $\boldsymbol{y}^{\boldsymbol{p}} = (y_j)_{j\notin\boldsymbol{p}}$. Notice that by writing $\boldsymbol{c}^{\boldsymbol{p}} = (c_i)_{j\notin\boldsymbol{p}}$ and $\boldsymbol{c}_{\boldsymbol{p}} = (c_j)_{j\in\boldsymbol{p}}$ we are, in practice, considering some subvectors of $\boldsymbol{c}$, while $\boldsymbol{c}^{[\boldsymbol{p}]}$ represents the solution of the linear system (3.9). Thus, in general, we have that $\boldsymbol{c}^{[\boldsymbol{p}]} \neq \boldsymbol{c}^{\boldsymbol{p}}$.

To extend Rippa's algorithm, our aim is to compute the validation errors at $\mathcal{V}$, i.e.,

$$\boldsymbol{e}_{\boldsymbol{p}} := \boldsymbol{e}_{\boldsymbol{p}}(\varepsilon) = \boldsymbol{y}_{\boldsymbol{p}} - s^{[\boldsymbol{p}]}(\boldsymbol{x}_{\boldsymbol{p}}) = (y_{p_1} - s^{[\boldsymbol{p}]}(\boldsymbol{x}_{p_1}), \ldots, y_{p_v} - s^{[\boldsymbol{p}]}(\boldsymbol{x}_{p_v}))^T \tag{3.10}$$

by means of (2.7) and without solving (3.9). Thus, from [30, Theorem 1], if A_ε and $\boldsymbol{c}$ are as in (2.7), the vector of ε-dependent errors $\boldsymbol{e}_{\boldsymbol{p}} = \boldsymbol{e}_{\boldsymbol{p}}(\varepsilon)$ in (3.10) related to the points $\boldsymbol{x}_{p_1}, \ldots, \boldsymbol{x}_{p_v}$ is the unique solution of the linear system

$$(\mathsf{A}_\varepsilon^{-1})_{\boldsymbol{p},\boldsymbol{p}} \boldsymbol{e}_{\boldsymbol{p}} = \boldsymbol{c}_{\boldsymbol{p}}, \qquad \boldsymbol{e}_{\boldsymbol{p}} \in \mathbb{R}^v, \tag{3.11}$$

where $(\mathsf{A}_\varepsilon^{-1})_{\boldsymbol{p},\boldsymbol{p}} = (\mathsf{A}_\varepsilon^{-1})_{i,j}$, with $i,j \in \boldsymbol{p}$, and $\boldsymbol{c_p} = (c_i)_{i\in\boldsymbol{p}}$, and the vectorized index $\boldsymbol{p}$ extracts the i-th rows and j-th columns subsystem with $i,j \in \boldsymbol{p}$ of the original matrix.

Concatenating all k validation error vectors $\boldsymbol{e}(\varepsilon) = (\boldsymbol{e}_{\boldsymbol{p}_1}^T, \ldots, \boldsymbol{e}_{\boldsymbol{p}_k}^T)^T(\varepsilon)$ of all k folds yields the vector of errors, and we define the LpOCV optimal value by

$$\varepsilon_* = \arg\min_{\varepsilon} ||\boldsymbol{e}(\varepsilon)||, \tag{3.12}$$

where $||\cdot||$ is any norm used in the minimization problem, for instance, the ∞-norm. So from (3.11) and (3.12) the LpOCV cost function to be minimized is given by

$$\text{L}p\text{OCV}(\varepsilon) = ||\boldsymbol{e}(\varepsilon)||_\infty. \tag{3.13}$$

In particular, by setting $\boldsymbol{p} = p \in \{1,\ldots,N\}$ in (3.10) and (3.11), we get to original Rippa's scheme [34], i.e.

$$e_p(\varepsilon) = y_p - s^{[p]}(\boldsymbol{x}_p) = \frac{c_p}{(\mathsf{A}_\varepsilon^{-1})_{p,p}}.$$

Indeed, when we set $k = N$ in the k-fold CV, this choice is equivalent to consider the LpOCV with $p = 1$, thus defining the LOOCV, because each validation fold consists of a single point. The resulting LOOCV scheme computes an exact N-fold CV, which has been widely employed by the scientific community and also generalized to other contexts e.g. in [5, 10, 11, 14, 22].

3.2. **Other MATLAB CV techniques.** CV is a model assessment technique that is commonly used to evaluate the performance of machine learning algorithms in making predictions on new data sets that it has not been trained on. This is carried out by creating a partition of the known data set in two subsets: a subset is used to train the algorithm, while the remaining one is applied for model validation. More precisely, each CV phase involves randomly partitioning the original data set into a training set and a validation set. The former is then used to train a supervised learning algorithm, whereas the latter is considered to evaluate its performance. This process is repeated several times and the average CV error is used as a performance indicator [31].

Among the most usual techniques of CV, already implemented in MATLAB, we here mention for our purposes only two as follows:

a) **k-fold CV**: It partitions data into k randomly chosen subsets (or folds) of roughly equal size. One subset is employed for validation of the model trained using the remaining subsets. This process is repeated k times such that each subset is used exactly once for validation. Across all k partitions the average error is computed. This approach is one of the most popular CV techniques even if it can be quite expensive from the computational point of view since the model needs to be trained repeatedly.

b) **LOOCV**: It partitions data using the k-fold approach, k being equal to the total number N of data. This data is used once as a validation set.

MATLAB software enables to use both k-fold CV and LOOCV algorithms through suitable application of `crossval` and `cvpartition` routines.

4. Adaptive residual subsampling scheme

In this section, we present our adaptive residual subsampling procedure, which allows us to refine and coarsen the node distribution by applying a kernel based interpolation process. More precisely, this scheme is used to approximate an unknown target function on uniformly distributed points, and then the residual error is evaluated at midpoints. The latter are added to the point set when the residual is over a prescribed refinement threshold, whereas they are

removed from that set when they are under a predefined coarsening threshold. Hereinafter, we give a more detailed explanation of this adaptive scheme.

We firstly introduce a finite sequence of data point sets, which in the iterative procedure are denoted by

$$X^{(0)}, X^{(1)}, \cdots, X^{(kmax)},$$

so that the $(k+1)$-th set $X^{(k+1)}$ is obtained from the k-th one, i.e. $X^{(k)} = \{\boldsymbol{x}_i^{(k)}\}_{i=1}^{N^{(k)}}$, after applying some refinement and/or coarsening procedures till a maximum number $kmax$ of iterations. This adaptive process thus brings to an update of the node distribution based on the computation of residual errors evaluated on some suitable test points. The iterative scheme we are constructing follows the common paradigm to solve, estimate, refine and/or coarsen till stop criteria are satisfied, or $kmax$ is reached.

Then, after defining a set of test points $T^{(k)} = \{\boldsymbol{t}_i^{(k)}\}_{i=1}^{N_{T^{(k)}}} \subset \Omega$, for $k \geq 0$, we compute the residual absolute error

$$\xi(\boldsymbol{t}_i^{(k)}) = \left| s(\boldsymbol{t}_i^{(k)}) - f(\boldsymbol{t}_i^{(k)}) \right|, \qquad \boldsymbol{t}_i^{(k)} \in T^{(k)}, \tag{4.14}$$

the interpolating function s being here constructed on the set $X^{(k)}$, and $N_{T^{(k)}}$ defining the number of points belonging to $T^{(k)}$.

The residual in (4.14) provides a measure of the error between the approximate solution and the function value computed at the test set $T^{(k)}$. In particular, the absolute error $\xi(\boldsymbol{t}_i^{(k)})$ is expected to be small when the test point $\boldsymbol{t}_i^{(k)}$ is on or close to a smooth region, while it is expected to be large when $\boldsymbol{t}_i^{(k)}$ lies in a part of the domain characterized by a low regularity or close to a discontinuous region.

Remark 4.1. *At the earliest stage (i.e. for $k = 0$), the test set $T^{(0)}$ is defined by starting from $X^{(0)} \equiv X$, whereas in the next iterations (i.e. for $k \geq 1$), the test set $T^{(k)}$ depends on both the sets $X^{(k)}$ and $X^{(k-1)}$.*

The residual (4.14) is used as an error indicator to define two different sets of our adaptive scheme, namely a refinement set and a coarsening one, called $X^{(k)}_{\text{refine}}$ and $X^{(k)}_{\text{coarse}}$, respectively. Therefore, introducing two positive thresholds θ_{refine} and θ_{coarse}, with $\theta_{\text{coarse}} < \theta_{\text{refine}}$, we can act as follows:

(1) If the error $\xi(\boldsymbol{t}_i^{(k)})$ in (4.14) is larger than θ_{refine}, the test point $\boldsymbol{t}_i^{(k)}$ is added in the set $X^{(k)}_{\text{refine}}$, and so $X^{(k)}$ is replaced by $X^{(k)} \cup X^{(k)}_{\text{refine}}$.

(2) If the error $\xi(\boldsymbol{t}_i^{(k)})$ in (4.14) is smaller than θ_{coarse}, the test point is moved from the set $X^{(k)}$ to the set $X^{(k)}_{\text{coarse}}$, and so $X^{(k)}$ is then updated with $X^{(k)} \backslash X^{(k)}_{\text{coarse}}$.

Accordingly, the set $X^{(k+1)}$ is adaptively obtained by adding the set $X^{(k)}_{\text{refine}}$ to the set $X^{(k)}$ and deleting the set $X^{(k)}_{\text{coarse}}$, i.e.,

$$X^{(k+1)} = \left\{ X^{(k)} \cup X^{(k)}_{\text{refine}} \right\} \backslash X^{(k)}_{\text{coarse}}.$$

The iterative method concludes once the process of addition/removal was completed, returning the final set $X^{(k^*)}$, k^* denoting the last iteration.

Analysing the proposed method, we note that it turns out to be dependent on the error (4.14). Indeed, at each iteration k, the adaptive procedure generates a kernel based interpolant of the form (2.6) defined on the set $X^{(k)}$, thus requiring to make some extra evaluations of the function f at the test points $\boldsymbol{t}_i^{(k)}$, $\forall i$. Since the function evaluation might be costly (or,

for instance, in real world applications, even not available at all), one could think of using an alternative strategy creating a local approximation around $\boldsymbol{t}_i^{(k)}$ and using the latter (instead of function value) in (4.14). A pseudo-code of this adaptive scheme is outlined in Algorithm 1. Similar computational techniques have already been studied in e.g. [6, 40].

Algorithm 1: Adaptive residual subsampling scheme

STEP 1 Assume $X^{(0)} \equiv X$

STEP 2 Fix $\theta_{\text{refine}} > \theta_{\text{coarse}} > 0$

STEP 3 While $k \leq kmax$ & $X^{(k)}_{\text{refine}} \cup X^{(k)}_{\text{coarse}} \neq \emptyset$

3.1: Compute $\varepsilon_*^{(k)}$ minimizing (3.13)

3.2: Solve the system (2.7) on $X^{(k)}$ and get the interpolant (2.6)

3.3: Define $T^{(k)}$

3.4: Evaluate the residual $\xi(\boldsymbol{t}_i^{(k)})$ in (4.14)

3.5: Define

$$X^{(k)}_{\text{refine}} = \{\boldsymbol{t}_i^{(k)} \in T^{(k)} \,:\, \xi(\boldsymbol{t}_i^{(k)}) > \theta_{\text{refine}},\, i = 1, \ldots, N_{T^{(k)}}\}$$
$$X^{(k)}_{\text{coarse}} = \{\boldsymbol{x}_i^{(k)} \in X^{(k)} \,:\, \xi(\boldsymbol{t}_i^{(k)}) < \theta_{\text{coarse}},\, i = 1, \ldots, N_{T^{(k)}}\}$$

3.6: Construct the set

$$X^{(k+1)} = \left\{X^{(k)} \cup X^{(k)}_{\text{refine}}\right\} \setminus X^{(k)}_{\text{coarse}}$$

5. NUMERICAL EXPERIMENTS

In this section, we analyze computational accuracy and efficiency of the residual subsampling scheme, which is implemented in MATLAB for adaptive 1D and 2D interpolation. All programs are run on a laptop with an Intel(R) Core(TM) i7-1065G7 CPU 1.50 GHz processor with 16 GB RAM.

In these tests we highlight the performance of k-fold CV algorithms including LOOCV as a special case, firstly focusing on approximation error and computational time and then emphasizing on iteration number (# iter), final number of points required to achieve convergence (N_{fin}), and conditioning of the kernel matrix (cond(A_ε)). Furthermore, we also compare numerical results obtained by considering a benchmark MATLAB implementation of LOOCV via `crossval` routine and those deriving from the adaptive method in [12]. To show how the adaptive CV based methods work, we consider various types of radial kernels involving both infinity and finite regularity like GA, IMQ, M6, M4 and M2. In such a case, we select the shape parameter ε as discussed in Section 3. In particular, the optimal values of the shape parameter are found by minimizing a cost function via the MATLAB `fminbnd` routine with a default tolerance of 10^{-4} and searching ε in the range $[0.2, 20]$.

To analyze the precision of the adaptive scheme, we compute the root mean square error (RMSE), i.e.,

$$\text{RMSE} = \frac{1}{\sqrt{N_e}}||f - s||_2 = \sqrt{\frac{1}{N_e}\sum_{i=1}^{N_e}[f(\boldsymbol{\xi}_i) - s(\boldsymbol{\xi}_i)]^2},$$

where the $\boldsymbol{\xi}_i$ is a uniform (equally-spaced or gridded) data set consisting of N_e evaluation points.

The matrix conditioning in (2.8) is estimated by using the MATLAB `cond` command, whereas the execution or CPU time of the adaptive algorithm is computed in seconds. We remark that the CPU time reported in this article is the result of an average obtained by sequentially running the code 100 times.

5.1. **Results in 1D interpolation.** In this subsection, we focus on adaptive 1D interpolation. All these tests are carried by starting from an initial point set $X^{(0)} \equiv X$, which consists of $N^{(0)} = 13$ equally-spaced points in the interval $\Omega = [-1, 1]$. Then, to connect the sets $X^{(k)}$ and $T^{(k)}$ within the adaptive method, for $k \geq 0$, we define the set $T^{(k)}$ of test points that are the midpoints taken from (sorted) interpolation nodes, i.e.

$$T^{(k)} = \{t_i^{(k)} = 0.5(x_i^{(k)} + x_{i+1}^{(k)}),\ i = 1, \ldots, N^{(k)} - 1\}.$$

The threshold values are assumed to be equal to $\theta_{\text{refine}} = 10^{-6}$ and $\theta_{\text{coarse}} = 10^{-8}$. However, in the comparison among the different residual subsampling algorithms, the refinement threshold θ_{refine} varies, while the coarsening one is kept fixed at the value $\theta_{\text{coarse}} = 10^{-8}$.

In order to validate in depth our adaptive algorithms, we consider the following benchmark target (or test) functions:

$$f_1(x) = \frac{1}{1 + 25x^2}, \qquad f_2(x) = 2\sin(5)\cos\left[\frac{10(x+1)}{2}\right] + \sin\left[\frac{5(x+1)}{2}\right],$$

where f_1 is the Runge function, and f_2 represents a trigonometric function (see [18, 40]).

In Tables 1, 2, 3 and 4, we present the results obtained by applying the adaptive residual subsampling method and using 10-fold CV and LOOCV schemes for the shape parameter choice. From these tables we get some useful information regarding the algorithm execution, i.e. the number of iterations and the final number of points required to achieve convergence. Specifically, we remark that the average approximation error (RMSE) is significantly smaller than the prescribed value θ_{refine}. Moreover, the automatic shape parameter choice also allows us to control the conditioning of the interpolation matrix that is always smaller than 10^{+17}. At the same time, we observe a high level of precision of the numerical method (roughly around the order of 10^{-7} or 10^{-8}). Indeed, taking into consideration the various situations, condition number and execution time assume quite similar values.

kernel	# iter	N_{fin}	RMSE	cond(A_ε)	time
GA	4	53	9.6e-8	2.3e+9	0.5
IMQ	4	51	1.8e-7	1.6e+8	0.4
M6	24	37	2.9e-7	2.3e+14	1.7
M4	5	81	6.1e-8	2.7e+10	0.5
M2	12	107	1.9e-7	6.7e+11	1.1

TABLE 1. Results with 10-fold CV for f_1.

In Tables 5 and 6, we compare our adaptive CV based methods, specifically the 10-fold CV and LOOCV, deriving from extended Rippa's scheme in Subsection 3.1, with the method [12]

kernel	# iter	N_{fin}	RMSE	cond(A_ε)	time
GA	4	51	2.3e-7	1.5e+10	0.4
IMQ	4	51	9.4e-8	8.2e+13	0.4
M6	7	99	1.8e-7	1.3e+8	0.6
M4	11	55	1.6e-7	2.0e+9	0.8
M2	7	99	1.8e-7	1.3e+8	0.6

TABLE 2. Results with LOOCV for f_1.

kernel	# iter	N_{fin}	RMSE	cond(A_ε)	time
GA	2	25	1.6e-7	3.9e+17	0.3
IMQ	3	49	2.1e-8	8.8e+17	0.4
M6	9	52	1.1e-7	4.1e+15	0.7
M4	7	66	2.5e-7	7.2e+15	0.7
M2	12	118	2.5e-7	6.8e+13	1.0

TABLE 3. Results with 10-fold CV for f_2.

kernel	# iter	N_{fin}	RMSE	cond(A_ε)	time
GA	3	20	7.8e-8	2.1e+14	0.3
IMQ	2	25	2.1e-7	1.2e+16	0.3
M6	11	116	2.6e-7	6.1e+13	0.9
M4	10	53	1.8e-7	7.3e+13	0.7
M2	11	116	2.6e-7	6.1e+13	0.9

TABLE 4. Results with LOOCV for f_2.

and another one characterized by the implementation of LOOCV, called LOOCV*, through the MATLAB `crossval` routine, as described in Subsection 3.2. In this experimentation, all tests have been run by M4 kernel. From the variation of the threshold θ_{refine}, we can observe that, on average, 10-fold CV and LOOCV turn out to be comparable to method [12] in terms of both number of nodes and execution time (necessary to achieve convergence). Though deducing the superiority of one approach in regard to another is not easy, we can state that the LOOCV is more efficient than LOOCV*. This fact is clearly evident when the value of θ_{refine} becomes smaller and smaller and so the threshold request is more demanding.

5.2. **Results in 2D interpolation.** In this subsection, we consider adaptive 2D interpolation. These experiments are run by taking an initial node set $X^{(0)} \equiv X$, containing $N^{(0)} = 320$

θ_{refine}	method [12]		10-fold CV		LOOCV		LOOCV*	
	N_{fin}	time	N_{fin}	time	N_{fin}	time	N_{fin}	time
1e-04	33	0.3	33	0.3	33	0.3	31	0.3
1e-05	39	0.4	39	0.3	39	0.3	39	0.4
1e-06	54	1.4	81	0.4	55	0.7	47	1.7

TABLE 5. Comparison among residual subsampling methods using M4 for f_1. Note that * refers to the use of the MATLAB `crossval` routine [31].

θ_{refine}	method [12]		10-fold CV		LOOCV		LOOCV*	
	N_{fin}	time	N_{fin}	time	N_{fin}	time	N_{fin}	time
1e-04	35	0.3	44	0.4	33	0.3	32	0.7
1e-05	44	0.4	48	0.5	45	0.4	45	2.4
1e-06	58	0.7	66	0.6	53	0.6	68	11.8

TABLE 6. Comparison among residual subsampling methods using M4 for f_2. Note that * refers to the use of the MATLAB `crossval` routine [31].

uniformly distributed points on $\Omega = [-1, 1]^2$. Then, we update the node set $X^{(k)}$ by applying the adaptive subsampling procedure described in Section 4. To correlate the interpolation node set $X^{(k)}$ with the corresponding test point set $T^{(k)}$, for $k \geq 0$, we compute the halfway points of $T^{(k)}$, as in [12]. As refinement threshold we set $\theta_{\text{refine}} = 10^{-4}$, while the coarsening threshold is $\theta_{\text{coarse}} = 10^{-8}$.

In our tests we analyze the performance of our algorithms taking the data values by three test functions. The first is known as a Franke-type function [40], i.e.,

$$f_3(x,y) = \exp\left[-0.1\left(x^2+y^2\right)\right] + \exp\left[-5\left((x-0.5)^2+(y-0.5)^2\right)\right] + \exp\left[-15((x+0.2)^2+(y+0.4)^2)\right] + \exp\left[-9\left((x+0.8)^2+(y-0.8)^2\right)\right].$$

The second is a hyperbolic tan function

$$f_4(x,y) = \frac{1}{9}\tanh\left[\frac{9}{2}(y-x)\right] + 1,$$

while the third is the exponential function [40] given by

$$f_5(x,y) = \exp\left[-60((x-0.35)^2+(y-0.25)^2)\right] + 0.2.$$

In Tables 7, 8, 9 and 10, we provide a numerical analysis to show how the adaptive residual subsampling methods based on CV techniques work when they are applied to solve some unknown functions characterized by quick variations in the domain Ω. As already done in the one dimensional case, in the above-mentioned tables we provide a detailed summary regarding the performance of the bivariate algorithm. Specifically, it collects the number of iterations needed to get convergence, the corresponding final number of nodes, the approximation error, the condition number of the kernel matrix and the total CPU time. From this study we point

out that both methods, involving 10-fold CV and LOOCV, enable the algorithm to converge in a relatively small number of iterations (namely, between 1 and 5). Moreover, the adaptive procedure is able to avoid an excessive additions of points because – in all considered examples – the final number of nodes is always less than 1000. This fact has a twofold importance: on the one hand the conditioning is kept under control, on the other the executing time takes a few seconds only. As regards the CPU time, we point the reader out that the type of radial kernel and so its smoothness can influence the convergence speed of the numerical scheme, which is subjected to an automatic addition or removal of points.

kernel	# iter	N_{fin}	RMSE	cond(A_ε)	time
GA	2	386	9.6e-6	5.1e+17	1.3
IMQ	2	321	5.0e-6	7.4e+15	1.0
M6	2	681	6.3e-6	3.2e+13	3.3
M4	3	496	1.5e-5	1.2e+13	2.5
M2	5	852	2.3e-5	7.9e+8	9.9

TABLE 7. Results with 10-fold CV for f_3.

kernel	# iter	N_{fin}	RMSE	cond(A_ε)	time
GA	2	550	3.1e-7	1.2e+19	2.3
IMQ	1	315	7.4e-6	9.7e+12	0.5
M6	3	369	2.0e-5	5.9e+14	2.1
M4	5	508	2.1e-5	7.7e+9	4.3
M2	4	860	2.3e-5	7.9e+8	5.4

TABLE 8. Results with LOOCV for f_3.

kernel	# iter	N_{fin}	RMSE	cond(A_ε)	time
GA	3	806	1.8e-5	2.3e+13	5.1
IMQ	2	654	9.3e-6	3.7e+11	2.2
M6	2	623	1.1e-5	2.6e+8	2.8
M4	2	622	6.3e-6	8.1e+8	2.4
M2	3	690	1.3e-5	1.5e+7	4.7

TABLE 9. Results with 10-fold CV for f_4.

kernel	# iter	N_{fin}	RMSE	cond(A_ε)	time
GA	3	938	1.2e-5	7.0e+11	5.9
IMQ	1	320	2.4e-5	7.4e+11	0.6
M6	4	387	1.4e-5	5.1e+9	2.4
M4	3	428	1.0e-5	1.2e+9	2.0
M2	4	593	1.4e-5	4.9e+7	4.8

TABLE 10. Results with LOOCV for f_4.

Finally, in order to highlight the benefit coming from the use of the new adaptive scheme, we conclude this section by making a comparison among different k-fold CV methods. In Table 11 we report the numerical results obtained by considering some specific values of k in the k-fold CV. Indeed, in this study we assume $k = 5, 25, 50, 100$ and $k = N$, where the latter results in the particular case of LOOCV. From this table, we can observe that all the CV techniques produce good results. The main differences in the algorithm performance seems to be dependent on the kind of radial kernel used. Conversely, for a fixed kernel, as evident focusing on each row of Table 11, the final number of interpolation points (required to satisfy the thresholds θ_{refine} and θ_{coarse}) and the CPU time are enough similar. In conclusion, these numerical experiments for 2D adaptive interpolation show that the use of CV techniques are efficient and effective for the selection of the kernel shape parameter but, at the same time, it is not possible to declare a clear and complete superiority of a CV scheme compared to other ones.

kernel	5-fold CV		25-fold CV		50-fold CV		100-fold CV		LOOCV	
	N_{fin}	time	N_{fin}	time	N_{fin}	time	N_{fin}	time	N_{fin}	time
GA	1243	13.1	1420	33.4	1330	19.6	1327	9.7	1231	9.7
IMQ	824	3.5	881	4.7	806	4.0	800	4.0	1053	7.4
M6	780	4.7	782	5.1	783	5.1	783	5.1	784	5.1
M4	683	7.2	683	7.6	684	7.0	689	7.7	682	7.0
M2	846	11.7	818	11.6	816	10.7	818	11.8	828	11.8

TABLE 11. Comparison among different k-fold CV methods for f_5.

6. CONCLUSIONS

In this work, we proposed the use of various CV criteria for selecting optimal kernel shape parameters within the adaptive residual subsampling method. More precisely, we focused on extended Rippa's scheme, considering k-fold CV and LOOCV as a special case. The application of such strategies in an interpolation framework showed that CV based techniques are valid alternative compared to other computational methods such as maximum likelihood estimation approaches. Indeed, the resulting CV based schemes revealed good level of flexibility and accuracy, also turning out to be more efficient than commonly used MATLAB routines.

As future work we expect to extend the application area of the adaptive CV based schemes to variably scaled kernels and discontinuous functions (see e.g. [2, 17, 27, 28]). Moreover, we also consider the chance to implement efficient residual subsampling algorithms for partition of unity methods (see e.g. [4, 7, 9]).

Acknowledgments

The authors sincerely thank the reviewers for their constructive and valuable comments. This work has been supported by the INdAM Research group GNCS as part of the GNCS-INdAM 2024 project "Metodi kernel e polinomiali per l'approssimazione e l'integrazione: teoria e software applicativo". The work of R.C. and D.M. has been supported by the Spoke 1 "FutureHPC & BigData"of ICSC - Centro Nazionale di Ricerca in High-Performance Computing, Big Data and Quantum Computing, funded by European Union - NextGenerationEU. Moreover, the work has been supported by the Fondazione CRT, project 2022 "Modelli matematici e algoritmi predittivi di intelligenza artificiale per la mobilità sostenibile". This research has been accomplished within the RITA "Research ITalian network on Approximation", the UMI Group TAA "Approximation Theory and Applications", and the SIMAI Activity Group ANA&A "Numerical and Analytical Approximation of Data and Functions with Applications". The work of A.N. has financially been supported by the National Science and Technology Council of Taiwan under grant numbers 112-2811-E-002-020-MY3.

References

[1] M. Bozzini, L. Lenarduzzi and R. Schaback: *Adaptive interpolation by scaled multiquadrics*, Adv. Comput. Math., **16** (2002), 375–387.
[2] M. Bozzini, L. Lenarduzzi, M. Rossini and R. Schaback: *Interpolation with variably scaled kernels*, IMA J. Numer. Anal., **35** (2015), 199–219.
[3] M. D. Buhmann: *Radial Basis Functions: Theory and Implementation*, Cambridge Monogr. Appl. Comput. Math., vol. 12, Cambridge Univ. Press, Cambridge (2003).
[4] R. Cavoretto, A. De Rossi, E. Perracchione and E. Venturino: *Reliable approximation of separatrix manifolds in competition models with safety niches*, Int. J. Comput. Math., **92** (2015), 1826–1837.
[5] R. Cavoretto, A. De Rossi: *A two-stage adaptive scheme based on RBF collocation for solving elliptic PDEs*, Comput. Math. Appl., **79** (2020), 3206–3222.
[6] R. Cavoretto, A. De Rossi: *Adaptive refinement procedures for meshless RBF unsymmetric and symmetric collocation methods*, Appl. Math. Comput., **382** (2020), 125354.
[7] R. Cavoretto: *Adaptive radial basis function partition of unity interpolation: A bivariate algorithm for unstructured data*, J. Sci. Comput., **87** (2021), Article ID: 41.
[8] R. Cavoretto, A. De Rossi, M. S. Mukhametzhanov and Y. D. Sergeyev: *On the search of the shape parameter in radial basis functions using univariate global optimization methods*, J. Global Optim., **79** (2021), 305–327.
[9] R. Cavoretto, A. De Rossi and W. Erb: *Partition of unity methods for signal processing on graphs*, J. Fourier Anal. Appl., **27** (2021), Article ID: 66.
[10] R. Cavoretto: *Adaptive LOOCV-based kernel methods for solving time-dependent BVPs*, Appl. Math. Comput., **429** (2022), Article ID: 127228.
[11] R. Cavoretto, A. De Rossi, A. Sommariva and M. Vianello: *RBFCUB: A numerical package for near-optimal meshless cubature on general polygons*, Appl. Math. Lett., **125** (2022), Article ID: 107704.
[12] R. Cavoretto, A. De Rossi: *An adaptive residual sub-sampling algorithm for kernel interpolation based on maximum likelihood estimations*, J. Comput. Appl. Math., **418** (2023), Article ID: 114658.
[13] R. Cavoretto, A. De Rossi and S. Lancellotti: *Bayesian approach for radial kernel parameter tuning*, J. Comput. Appl. Math., **441** (2024), Article ID: 115716.
[14] R. Cavoretto, A. De Rossi, F. Dell'Accio, F. Di Tommaso, N. Siar, A. Sommariva and M. Vianello: *Numerical cubature on scattered data by adaptive interpolation*, J. Comput. Appl. Math., **444** (2024), Article ID: 115793.
[15] R. Cavoretto, A. De Rossi, A. Haider and S. Lancellotti: *Comparing deterministic and statistical optimization techniques for the shape parameter selection in RBF interpolation*, Dolomites Res. Notes Approx., **17** (2024), 48–55.
[16] A. Celisse, S. Robin: *Nonparametric density estimation by exact leave-p-out cross-validation*, CSDA, **52** (2008), 2350–2368.

[17] S. De Marchi: *Padua points and fake nodes for polynomial approximation: old, new and open problems*, Constr. Math. Anal., **5** (2022), 14–36.
[18] T. A. Driscoll, A. R. H. Heryudono, *Adaptive residual subsampling methods for radial basis function interpolation and collocation problems*, Comput. Math. Appl., **53** (2007), 927–939.
[19] G. E. Fasshauer: *Meshfree Approximation Methods with MATLAB*, Interdisciplinary Mathematical Sciences, vol. 6, World Scientific Publishing Co., Singapore (2007).
[20] G. E. Fasshauer: *Positive definite kernels: Past, present and future*, Dolomites Res. Notes Approx., **4** (2011), 21–63.
[21] G. E. Fasshauer, M. J. McCourt: *Kernel-based Approximation Methods using MATLAB*, Interdisciplinary Mathematical Sciences, Vol. 19, World Scientific Publishing Co., Singapore (2015).
[22] G. E. Fasshauer, J. G. Zhang: *On choosing "optimal" shape parameters for RBF approximation*, Numer. Algorithms, **45** (2007), 345–368.
[23] B. Fornberg, J. Zuev: *The Runge phenomenon and spatially variable shape parameters in RBF interpolation*, Comput. Math. Appl., **54** (2007), 379–398.
[24] K. Gao, G. Mei, S. Cuomo, F. Piccialli and N. Xu: *ARBF: adaptive radial basis function interpolation algorithm for irregularly scattered point sets*, Soft Computing, **24** (2020), 17693–17704.
[25] A. Golbabai, E. Mohebianfar and H. Rabiei: *On the new variable shape parameter strategies for radial basis functions*, Comput. Appl. Math., **34** (2015), 691–704.
[26] F. J. Hickernell, Y. C. Hon: *Radial basis function approximations as smoothing splines*, Appl. Math. Comput., **102** (1999), 1–24.
[27] M. Karimnejad Esfahani, S. De Marchi and F. Marchetti: *Moving least squares approximation using variably scaled discontinuous weight function*, Constr. Math. Anal., **6** (2023), 38–54.
[28] L. Lenarduzzi, R. Schaback: *Kernel-based adaptive approximation of functions with discontinuities*, Appl. Math. Comput., **307** (2017), 113–123.
[29] L. Ling, F. Marchetti: *A stochastic extended rippa's algorithm for LpOCV*, Appl. Math. Lett., **129** (2022), Article ID: 107955.
[30] F. Marchetti: *The extension of Rippa's algorithm beyond LOOCV*, Appl. Math. Lett., **120** (2021), Article ID: 107262.
[31] MATLAB version: *9.13.0.2553342 (R2022b) Update 9*, Natick, Massachusetts, The MathWorks Inc. (2022).
[32] A. Noorizadegan, C.-S. Chen, R. Cavoretto and A. De Rossi: *Efficient truncated randomized SVD for mesh-free kernel methods*, Comput. Math. Appl., **164** (2024), 12–20.
[33] A. Noorizadegan, R. Schaback: *Introducing the evaluation condition number: A novel assessment of conditioning in radial basis function methods*, Eng. Anal. Bound. Elem., **166** (2024), Article ID: 105827.
[34] S. Rippa: *An algorithm for selecting a good value for the parameter c in radial basis function interpolation*, Adv. Comput. Math., **11** (1999), 193–210.
[35] R. Schaback: *Error estimates and condition numbers for radial basis function interpolation*, Adv. Comput. Math., **3** (1995), 251–264.
[36] M. Scheuerer: *An alternative procedure for selecting a good value for the parameter c in RBF-interpolation*, Adv. Comput. Math., **34** (2011), 105–126.
[37] M. Scheuerer, R. Schaback and M. Schlather: *Interpolation of spatial data – A stochastic or a deterministic problem?*, European J. Appl. Math., **24** (2013), 601–629.
[38] G. K. Veni, C. Satyanarayana and M. C. Krishnareddy: *Residual error based adaptive method with an optimal variable scaling parameter for RBF interpolation*, International Journal of Applied Mechanics and Engineering, **28** (2023), 37–46.
[39] H. Wendland: *Scattered Data Approximation*, Cambridge Monogr. Appl. Comput. Math., vol. 17, Cambridge Univ. Press, Cambridge (2005).
[40] Q. Zhang, Y. Zhao and J. Levesley: *Adaptive radial basis function interpolation using an error indicator*, Numer. Algorithms, **76** (2017), 441–471.

ROBERTO CAVORETTO
UNIVERSITY OF TURIN
DEPARTMENT OF MATHEMATICS "GIUSEPPE PEANO "
VIA CARLO ALBERTO 10, 10123, TURIN, ITALY
ORCID: 0000-0001-6076-4115
Email address: roberto.cavoretto@unito.it

ADEEBA HAIDER
UNIVERSITY OF TURIN
DEPARTMENT OF MATHEMATICS "GIUSEPPE PEANO "
VIA CARLO ALBERTO 10, 10123, TURIN, ITALY
ORCID: 0009-0001-4198-1808
Email address: adeeba.haider@unito.it

SANDRO LANCELLOTTI
UNIVERSITY OF TURIN
DEPARTMENT OF MATHEMATICS "GIUSEPPE PEANO "
VIA CARLO ALBERTO 10, 10123, TURIN, ITALY
ORCID: 0000-0003-4253-3561
Email address: sandro.lancellotti@unito.it

DOMENICO MEZZANOTTE
UNIVERSITY OF TURIN
DEPARTMENT OF MATHEMATICS "GIUSEPPE PEANO "
VIA CARLO ALBERTO 10, 10123, TURIN, ITALY
ORCID: 0000-0001-5154-6538
Email address: domenico.mezzanotte@unito.it

AMIR NOORIZADEGAN
NATIONAL TAIWAN UNIVERSITY
DEPARTMENT OF CIVIL ENGINEERING
10617, TAIPEI, TAIWAN
ORCID: 0000-0003-3191-0990
Email address: amirnoori23@ntu.edu.tw