# A Bayesian Approach for Simultaneously Radial Kernel Parameter Tuning in the Partition of Unity Method

Roberto Cavoretto, Sandro Lancellotti, and Federico Romaniello(✉)

Department of Mathematics "Giuseppe Peano", University of Torino, via Carlo Alberto 10, 10123 Torino, Italy
{roberto.cavoretto,sandro.lancellotti,federico.romaniello}@unito.it

**Abstract.** In this paper, Bayesian optimisation is used to simultaneously search the optimal values of the shape parameter and the radius in radial basis function partition of unity interpolation problem. It is a probabilistic iterative approach that models the error function with a step-by-step self-updated Gaussian process, whereas partition of unity leverages a mesh-free method that allows us to reduce cost-intensive computations when the number of scattered data is very large, as the entire domain is decomposed into several smaller subdomains of variable radius. Numerical experiments on the scattered data interpolation problem show that the combination of these two tools sharply reduces the search time with respect to other techniques such as the leave one out cross validation.

**Keywords:** Radial Basis Function · Kernel-based Interpolation · Shape Parameter · Bayesian Optimisation · Hyper-parameter Search

## 1 Introduction

Meshfree methods are popular tools for solving interpolation problems and one of the most used is the Partition of Unity Method (PUM). Here, it is implemented using Radial Basis Functions (RBFs) or radial kernels as local approximants, since PUM is a very efficient tool in scattered data interpolation [3,6,17]. The main disadvantage of radial kernel-based method is the computational cost associated with the solution of (usually) large linear systems; PUM are instead used to efficiently split the data into smaller subdomains (or balls) of radius $\delta$. Such a method is obtained by a weighted sum of some RBFs depending on a *shape parameter* $\varepsilon$ in each subdomain [2].

The aim of this work is to apply a well-known statistical technique, called *Bayesian Optimization* (BO) [16], to simultaneously search the optimal $(\varepsilon, \delta)$ values for each subdomain in RBF-PUM. This technique, developed in machine learning for optimisation of black-box or difficult-to-evaluate functions, can be used in hyperparameter tuning problems to avoid computation and evaluation


Y. D. Sergeyev et al. (Eds.): NUMTA 2023, LNCS 14477, pp. 215–222, 2025.
https://doi.org/10.1007/978-3-031-81244-6_18

of the interpolant for those parameters that are far from being optimal, leading to a significant reduction of the computational time. To complete this analysis, we compare BO results with those obtained by using the Leave One Out Cross Validation (LOOCV) scheme [15].

The paper is organised as follows. In Sect. 2, RBF-PUM interpolation and LOOCV method are briefly stated. In Sect. 3, BO Gaussian processes and acquisition functions are described. In Sect. 4, numerical experiments are analysed and discussed. Section 5 concludes the paper.

## 2 RBF-PUM Interpolation

In this section, we introduce the interpolation problem and the basic theory on RBF-PUM, highlighting the reasons that inspired this contribution.

### 2.1 The RBF Method

Given a set of distinct data $X = \{\boldsymbol{x}_i, i = 1, \ldots, n\}$ arbitrarily distributed on a domain $\Omega \subseteq \mathbb{R}^d$ with an associated set $F = \{f_i = f(\boldsymbol{x}_i), i = 1, \ldots, n\}$ of data values obtained by sampling some, possibly unknown, function $f : \Omega \to \mathbb{R}$ at the nodes $\boldsymbol{x}_i$, the *scattered data interpolation problem* consists in finding a function $P_f : \Omega \to \mathbb{R}$ such that it matches the measurements at the corresponding locations, i.e. $P_f(\boldsymbol{x}_i) = f_i$, $i = 1, \ldots, n$.

We now suppose to have a univariate function $\varphi : [0, \infty) \to \mathbb{R}$, known as RBF, which depends on a shape parameter $\varepsilon > 0$ providing, for $\boldsymbol{x}, \boldsymbol{z} \in \Omega$, the real symmetric strictly positive definite kernel

$$\kappa_\varepsilon(\boldsymbol{x}, \boldsymbol{z}) = \varphi(\varepsilon||\boldsymbol{x} - \boldsymbol{z}||_2) := \varphi(\varepsilon r).$$

The kernel-based interpolant $P_f$ can be written as

$$P_f(\boldsymbol{x}) = \sum_{k=1}^{n} c_k \kappa_\varepsilon(\boldsymbol{x}, \boldsymbol{x}_k), \quad \boldsymbol{x} \in \Omega,$$

whose coefficients are the solution of the linear system

$$\mathsf{K}\boldsymbol{c} = \boldsymbol{f}, \tag{1}$$

where $\boldsymbol{c} = (c_1, \ldots, c_n)^\mathsf{T}$, $\boldsymbol{f} = (f_1, \ldots, f_n)^\mathsf{T}$, and $\mathsf{K}_{ik} = \kappa_\varepsilon(\boldsymbol{x}_i, \boldsymbol{x}_k)$, $i, k = 1, \ldots, n$. Since $\kappa_\varepsilon$ is a symmetric and strictly positive definite kernel, the system (1) has exactly one solution [8].

### 2.2 The PUM Scheme and the LOOCV Technique

It is well-known that inverting the kernel interpolation matrix in (1) might be computationally expensive when the amount of data highly increases. An effective method to overcome this problem is to partition the open and bounded

domain $\Omega$ into $m$ overlapping subdomains, i.e. $\Omega \subseteq \bigcup_{j=1}^{m} \Omega_j$, and hence to split the interpolation problem locally in each one of them.

The PU covering consists of overlapping balls of radius $\delta$ whose centres are the grid data $P = \{\tilde{\boldsymbol{x}}_k,\, k = 1, \ldots, m\}$.

In [7] it is shown that when the nodes are nearly uniformed distributed, $m$ is a suitable number of PU subdomains on $\Omega$ if $n/m \approx 2^d$. Then, the covering property is satisfied by taking the radius $\delta$ such that

$$\delta \geq \frac{1}{m^{1/d}}.$$

The PUM solves a local interpolation problem on each subdomain and constructs the global approximant by gluing together the local contributions using weights. To achieve that, we need those weights are a family of compactly supported, non-negative, continuous functions $w_j$, with $\operatorname{supp}(w_j) \subseteq \Omega_j$, such that

$$\sum_{j=1}^{m} w_j(\boldsymbol{x}) = 1, \quad \boldsymbol{x} \in \Omega.$$

Once we choose the partition of unity $\{w_j\}_{j=1}^{m}$, the global interpolant is formed by the weighted sum of $m$ local approximants $P_f^j$, i.e.

$$P_f(\boldsymbol{x}) = \sum_{j=1}^{m} P_f^j(\boldsymbol{x})\, w_j(\boldsymbol{x}) = \sum_{j=1}^{m} \left( \sum_{k=1}^{n_j} c_k^j \kappa_{\varepsilon,\delta}(\boldsymbol{x}, \boldsymbol{x}_k^j) \right) w_j(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega,$$

where $n_j = |\Omega_j|$ and $\boldsymbol{x}_k^j \in X_j = X \cap \Omega_j$, with $k = 1, \ldots, n_j$.

We remark that the accuracy of the fit strongly depends on the choices of the shape parameter and the radius, see e.g. [4,5,9,11,12].

Rippa in [15] proposed the LOOCV for the search of the optimal value of the RBF shape parameter $\varepsilon$. This technique can be extended to simultaneously search the optimal values for $\delta$ and $\varepsilon$ and applied in each PU subdomain $\Omega_j$. It consists in evaluating in every subdomain $\Omega_j$, for each $(\varepsilon, \delta)$ and $k \in \{1, \ldots, n_j\}$, the interpolation error $e_{j,k}(\varepsilon, \delta)$ at the point $x_k$ of the RBF interpolant $P_f^{j,k}$ fitted on the data set $X_k = X \setminus \{x_k\}$ and the data values $F_k = F \setminus \{f_k\}$. To avoid this computation, the error can be computed by the rule:

$$e_{j,k}(\varepsilon, \delta) = \frac{c_k^j}{(\mathsf{K}_{kk}^{-1})_j}.$$

Hence, the optimal value $(\varepsilon, \delta)_j^*$ is the one that minimises the error function $Er(\varepsilon, \delta)_j$ defined as follows:

$$Er(\varepsilon, \delta)_j = \max_{k=1,\ldots,n_j} \left| \frac{c_k^j}{(\mathsf{K}_{kk}^{-1})_j} \right|. \tag{2}$$

The LOOCV technique is then formalised by imposing to evaluate the error function also for that value in the discrete set that does not lead to a good

result. A drawback of this scheme is the impossibility, in general, to attain the global minimum due to the discrete research. In this work we propose a faster procedure that conducts a continuous research in the parameters space in order to obtain a better approximation of $(\varepsilon, \delta)_j^*$.

## 3 Bayesian Optimisation

When it comes to find a global maximiser for an unknown or difficult-to-evaluate function $f$ on some bounded set $X$, the Bayesian Optimisation (BO) [13] is an approach to carry out the research. Very popular in machine learning, BO is an iterative technique that is based on exploiting all the available resources. It consists in building a probabilistic model of $f$, called *surrogate model*, and using it to help directly the sampling point in $X$, by means of an acquisition function, where the target function will be evaluated. Once an iteration is made, the distribution is updated and then used in the next iteration. Though there is a computation for the selection of the next point to evaluate, when evaluations of $f$ are expensive, the computation of a better point is motivated by reaching the maximum in a few iterations, as in the case of the error function of some expensive training machine learning algorithms like multi-layer neural networks. Hereinafter, we briefly review the BO technique [1].

A Gaussian Process (GP) is a collection of random variables such that any subsets of these have a joint Gaussian distribution. Then GPs are completely specified by a mean function $m : \mathcal{X} \to \mathbb{R}$ and a positive definite covariance function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (see [14]). Moreover, they are the most common choice for the surrogate model for BO due to the low evaluation cost and to the ability of incorporating prior beliefs about the objective function. When modeling the target function with a GP as $f(\mathbf{x}) \sim \mathcal{GP}\big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\big)$, we impose that

$$\mathbb{E}\big[f(\mathbf{x})\big] = m(\mathbf{x}), \qquad \mathbb{E}\big[\big(f(\mathbf{x}) - m(\mathbf{x})\big)\big(f(\mathbf{x}') - m(\mathbf{x}')\big)\big] = k(\mathbf{x}, \mathbf{x}').$$

In the matter of making a prediction given by some observations, the assumption of joint Gaussianity allows retrieving the prediction using the standard formula for mean and variance of a conditional normal distribution. Hence, suppose to have $s$ observation $\mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_s))^\intercal$ on $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_s)^\intercal$ and a new point $\bar{\mathbf{x}}$ on which we are interested in having a prediction of $\bar{f} = f(\bar{\mathbf{x}})$. The previous observations $\mathbf{f}$ and the predicted value $f(\bar{\mathbf{x}})$ are jointly normally distributed:

$$Pr\left(\begin{bmatrix}\mathbf{f} \\ f(\bar{\mathbf{x}})\end{bmatrix}\right) = \mathcal{N}\left[\begin{bmatrix}\mu(\mathbf{X}) \\ \mu(\bar{\mathbf{x}})\end{bmatrix}, \begin{bmatrix}K(\mathbf{X}, \mathbf{X}) & K(\mathbf{X}, \bar{\mathbf{x}}) \\ K(\mathbf{X}, \bar{\mathbf{x}})^\intercal & k(\bar{\mathbf{x}}, \bar{\mathbf{x}})\end{bmatrix}\right],$$

where $K(\mathbf{X}, \mathbf{X})$ is the $s \times s$ matrix with $(i, j)$-element $k(\mathbf{x}_i, \mathbf{x}_j)$, and $K(\mathbf{X}, \bar{\mathbf{x}})$ is a $s \times 1$ vector whose element $i$ is given by $k(\mathbf{x}_i, \bar{\mathbf{x}})$. Since $Pr(f(\bar{\mathbf{x}})|\mathbf{f})$ must also be normal, it is also possible to estimate the distribution, the mean and the covariance, for any point in the domain. When data points and data values retrieved by the evaluation of the target function are fed to the model, they induce a posterior distribution over functions which is used for the next iteration

as a prior. We point out that if a function is modelled by a GP, when we observe a value, we are observing the random variable associated to the point.

*An acquisition function* $a : \mathcal{X} \to \mathbb{R}$ is a function used to determine the next evaluation point for the objective function. This chosen point maximises the acquisition function, and its evaluation by the objective function is used to update the surrogate model. An acquisition function is defined such that high acquisition corresponds to potentially high values of the objective function. There exists a trade-off between exploration and exploitation in the selection of an acquisition function: exploration means selecting points where the uncertainty is high, that is, far from the already evaluated points; exploitation, on the contrary, means selecting those points close to those already evaluated by the objective function. The most common acquisition functions are:

- **Probability of Improvement**, which maximises the probability of improvement over the best current value;
- **Expected Improvement**, which maximises the expected improvement over the current best;
- **GP Upper Confidence Bound**, which minimises the cumulative regret.

The acquisition function used in this work is the "Expected Improvement" [10] as it considers not only the probability of improvement of the candidate point w.r.t. the previous maximum, but also the magnitude of this improvement.

Suppose that after a number of iterations the current maximum of the objective function is $f(\hat{\mathbf{x}})$. Given a new point $\mathbf{x}$, the Expected Improvement acquisition function computes the expectation of improvement $f(\mathbf{x}) - f(\hat{\mathbf{x}})$ over the part of the normal distribution that is above the current maximum:

$$EI(\mathbf{x}) = \int_{f(\hat{\mathbf{x}})}^{\infty} \big(f^*(\mathbf{x}) - f(\hat{\mathbf{x}})\big) \frac{1}{\sqrt{2\pi}\sigma(\mathbf{x})} e^{-\frac{1}{2}[(f^*(\mathbf{x}) - \mu(\mathbf{x}))/\sigma(\mathbf{x})]^2} df^*(\mathbf{x}), \quad (3)$$

where $f^*(\mathbf{x})$, $\mu(\mathbf{x})$ and $\sigma(\mathbf{x})$ represent the predicted value by the surrogate model, the expected value and the variance of $\mathbf{x}$, respectively. Solving integral (3) leads to the following closed form for the evaluation of the Expected Improvement:

$$EI(x) = \begin{cases} (\mu(\mathbf{x}) - f(\hat{\mathbf{x}}))\Phi(Z) + \sigma(\mathbf{x})\phi(Z), & \text{if } \sigma(\mathbf{x}) > 0, \\ 0, & \text{if } \sigma(\mathbf{x}) = 0, \end{cases}$$

where $Z = \frac{\mu(\mathbf{x}) - f(\hat{\mathbf{x}})}{\sigma(\mathbf{x})}$, while $\phi$ and $\Phi$ are the Probability Density Function and Cumulative Distribution Function of the standard normal distribution $\mathcal{N}(0, 1)$.

## 4 Numerical Experiments

A sketch of the proposed method is shortly described in Algorithm 1. Numerical experiments are presented below to find the best $(\varepsilon, \delta)$ that minimise the error function for each subdomain. We compare the results obtained by applying BO

**Algorithm 1** BO-PUM

**Input:** Data points $X$, data values $F$.

Construct a partition of unity with centers generated as an equispaced grid;
In each subdomain, find the value for the radius that ensures the minimum density;
Apply BO in each subdomain to find $(\varepsilon, \delta)$;
For each subdomain construct a local RBF approximant;
Create a global RBF-PUM approximant from the local ones.

**Output:** RBF-PUM approximation.

and LOOCV in the RBF-PUM. To apply the BO for the search of the optimal parameters $(\varepsilon, \delta)$, we suppose that the objective function to maximise is the Maximum Absolute Error (MAE) of the RBF interpolant, changed of sign because the BO is a maximisation process as described in Sect. 3. We use a variable number of points for the training set and a fixed one of 1000 points for the test set. During the BO, for each subdomain, after determining the points belonging to it, we further divide them in a sub-training and sub-validation set to allow the evaluation of training error. After identifying the best parameter pairs for each subdomain and each optimiser, a PUM interpolant is trained on the training set for each optimiser with the parameters found. For each subdomain, the search space is $(0, 20] \times [\delta_{min}, 2\delta_{min}]$, where $\delta_{min}$ is the radius value that ensures a minimum density in the subdomain. The LOOCV tries each possible combination between 500 equally spaced points in $(0, 20]$ and 30 equally spaced points in $[\delta_{min}, 2\delta_{min}]$. On the other hand, BO performs 5 random steps plus at most 25 Bayesian steps in the search space. The iterative process stops if the desired tolerance $\tau$ is reached. We perform the experiments on 3 different sizes of random data in the interval $[0, 1]^2$ using the RBFs

$$\varphi_1(\varepsilon r) = e^{-\varepsilon^2 r^2} \qquad \text{(Gaussian } C^\infty\text{)},$$
$$\varphi_2(\varepsilon r) = e^{-\varepsilon r}(1 + 3\varepsilon r + \varepsilon^2 r^2) \qquad \text{(Matérn } C^4\text{)},$$

and the test function

$$f(x_1, x_2) = 2\cos(10x_1)\sin(10x_2) + \sin(10x_1 x_2).$$

Results are shown in Table 1. As it can be seen in this table, the errors obtained using the different techniques are similar in all cases, while the runtime of BO is always lower than LOOCV. It should also be noted that, for the Gaussian kernel, as the number of points increases, the execution time of the BO decreases. This fact is due to the high density of the space when a greater number of points is considered. In particular, when this happens, there are denser subdomains and thus better accuracy and fewer BO iterations are needed to satisfy the tolerance $\tau$.

**Table 1.** Computational time and MAE using LOOCV and BO optimiser for Gaussian and Matérn kernels and different number $N$ of random points in $[0, 1]^2$. Two tolerances $\tau$ for the BO training error are used.

| | | | Gaussian kernel | | Matérn kernel | |
|---|---|---|---|---|---|---|
| $N$ | optimizer | $\tau$ | time (s) | MAE | time (s) | MAE |
| 8000 | LOOCV | | 3.95e+03 | 1.87e-04 | 4.06e+03 | 1.68e-03 |
| | BO | 1e-04 | 1.18e+01 | 7.83e-05 | 3.36e+02 | 4.56e-04 |
| | | 1e-05 | 4.19e+01 | 8.84e-06 | 8.37e+02 | 4.96e-04 |
| 4000 | LOOCV | | 1.96e+03 | 1.34e-05 | 2.01e+03 | 6.26e-03 |
| | BO | 1e-04 | 1.70e+01 | 6.91e-05 | 4.19e+02 | 1.08e-03 |
| | | 1e-05 | 1.07e+02 | 4.88e-05 | 4.69e+02 | 3.04e-03 |
| 2000 | LOOCV | | 9.73e+02 | 9.07e-03 | 1.01e+03 | 1.87e-02 |
| | BO | 1e-04 | 4.01e+01 | 8.67e-05 | 2.81e+02 | 1.24e-02 |
| | | 1e-05 | 2.04e+02 | 1.61e-04 | 2.71e+02 | 8.89e-03 |

## 5 Conclusions

As can be seen from the numerical experiments in Sect. 4, when comparing the parameters search in the case of interpolation with LOOCV and BO, it becomes evident that the error values generally hover around the same magnitude. However, the key distinguishing factor between the two methods is the computational time, wherein the BO approach significantly reduces the time required, often by an order of magnitude.

**Acknowledgments.** The authors sincerely thank the reviewers for the careful reading and valuable comments on the paper. This research has been accomplished within the RITA "Research ITalian network on Approximation" and the UMI Group TAA "Approximation Theory and Applications". This work has been supported by the INdAM–GNCS 2022 Project "Computational methods for kernel-based approximation and its applications", code CUP_E55F22000270001, and by the Spoke 1 "FutureHPC & BigData" of the ICSC–National Research Center 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".

## References

1. Brochu, E., Cora, V.M., De Freitas, N.: A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning (2010). arXiv:1012.2599
2. Cavoretto, R.: Adaptive radial basis function partition of unity interpolation: a bivariate algorithm for unstructured data. J. Sci. Comput. **87**, 41 (2021)

3. Cavoretto, R., De Rossi, A., Lancellotti, S., Perracchione, E.: Software implementation of the partition of unity method. Dolomites Res. Notes Approx. **15**, 35–46 (2022)
4. Cavoretto, R., De Rossi, A., Mukhametzhanov, M.S., Sergeyev, Y.D.: On the search of the shape parameter in radial basis functions using univariate global optimization methods. J. Global Optim. **79**, 305–327 (2021)
5. Cavoretto, R., De Rossi, A., Sommariva, A., Vianello, M.: RBFCUB: a numerical package for near-optimal meshless cubature on general polygons. Appl. Math. Lett. **125**, 107704 (2022)
6. Chen, C., Noorizadegan, A., Young, D.L., Chen, C.S.: On the selection of a better radial basis function and its shape parameter in interpolation problems. Appl. Math. Comput. **442**, 127713 (2023)
7. Fasshauer, G.E.: Meshfree Approximation Methods with MATLAB. World Scientific, Singapore (2007)
8. Fasshauer, G.E., McCourt, M.J.: Kernel-based Approximation Methods Using MATLAB. World Scientific, Singapore (2015)
9. Fornberg, B., Wright, G.: Stable computation of multiquadrics interpolants for all values of the shape parameter. Comput. Math. Appl. **47**, 497–523 (2004)
10. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. Global Optim. **13**, 455–492 (1998)
11. Larsson, E., Fornberg, B.: Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions. Comput. Math. Appl. **49**, 103–130 (2005)
12. Ling, L., Marchetti, F.: A stochastic extended Rippa's algorithm for LpOCV. Appl. Math. Letters **129**, 107955 (2022)
13. Mockus, J., Tiesis, V., Zilinskas, A.: The application of Bayesian methods for seeking the extremum. Towards Global Optim. **2**, 117–129 (1978)
14. Rasmussen, C.E., Williams, C.: Gaussian Processes for Machine Learning, MIT Press (2006)
15. Rippa, S.: An algorithm for selecting a good value for the parameter $c$ in radial basis function interpolation. Adv. Comput. Math. **11**, 193–210 (1999)
16. Snoek, J., Larochelle, H., Adams, R.P.: Practical bayesian optimization of machine learning algorithms. Adv. Neural. Inf. Process. Syst. **25**, 2960–2968 (2012)
17. Wendland, H.: Scattered Data Approximation. Cambridge Monogr. Appl. Comput. Math., vol. 17, Cambridge Univ. Press, Cambridge (2005)