On adaptive sampling techniques for metamodels based on NURBS entities from unstructured data

M. Zani, E. Panettieri, M. Montemurro *

Université de Bordeaux, Arts et Métiers Institute of Technology, CNRS, INRA, Bordeaux INP, I2M UMR 5295, Talence, F-33405, France

Keywords:
Metamodel
Surrogate model
NURBS
Optimisation
Adaptive sampling
Unstructured data
Additive manufacturing

A B S T R A C T

The paper investigates the influence of adaptive sampling strategies on the generation of a metamodel based on Non-Uniform Rational Basis Spline (NURBS) entities, obtained from unstructured data, with the purpose of improving accuracy while minimising computational resources. The metamodel is defined as solution of a constrained non-linear programming problem and it is solved through a three-step optimisation process based on a gradient-based algorithm. Moreover, this paper introduces a generalised formulation of the NURBS-based metamodel capable of handling unstructured sampling data, enabling simultaneous optimisation of control points and weights. Sensitivity analyses are performed to evaluate the influence of various adaptive sampling techniques, including cross-validation-based and geometry-based strategies, on the resulting metamodel, in terms of accuracy and computational costs. Analytical benchmarks functions and a complex real-world engineering problem (dealing with the non-linear thermomechanical analysis of a part produced with the fused deposition modelling technology) are used to prove the effectiveness of the NURBS-based metamodel coupled with adaptive sampling strategies in achieving high accuracy and efficiency.

1. Introduction

With increasing process complexity and the need to enhance manufacturing speed and understanding, fast and accurate Finite Element (FE) analyses have become essential in many industrial sectors. However, the high computational costs associated with complex physics-based simulations make these models unsuitable for optimisation, sensitivity analysis, and model calibration [1–3]. To address this challenge, metamodels (surrogate models) can be employed to substitute the costly simulation with an approximation derived from a suitable training dataset [4–6]. Among the broad variety of metamodeling strategies available in the literature, one can find reduced-order models [7,8], which retain the underlying physics but are limited to cases with accessible governing equations, making them unsuitable for complex models defined by commercial software. Common alternatives are response surface methods, such as Kriging [1,9–11], Artificial Neural Networks [12], Polynomial Chaos Expansions, Polynomial Chaos Expansion based kriging [13], kriging combined with partial least-squares and its gradient-enhanced version [14], and metamodels based on various basis of functions [4,14–17]. These approaches rely on a predefined training dataset and involve two main steps: (a) selecting an adequate Design of Experiments, and (b) choosing an approximation technique to model the desired response.

Consequently, research is ongoing on finding the most effective methods for creating accurate metamodels with a minimum amount of sample points and optimal approximation techniques. Despite progress, challenges remain, such as handling highly non-linear responses, limited training datasets, and the time required to generate them. In particular, in engineering applications, the

* Corresponding author.
E-mail addresses: marco.montemurro@ensam.eu, marco.montemurro@u-bordeaux.fr (M. Montemurro).

https://doi.org/10.1016/j.cma.2025.117781

Acronyms

AE	Accumulative Error
B-spline	Basis spline
CNLPP	Constrained Non-Linear Programming Problem
CP	Control Point
CV	Cross Validation
CVV	Cross Validation Voronoi
FDM	Fused Deposition Modelling
FE	Finite Element
GC-MMA	Global Convergent Method of Moving Asymptotes
LHD	Latin Hypercube Distribution
LOOCV	Leave-one-out cross validation
NRMSE	Normalised-root-mean-squared error
NURBS	Non-Uniform Rational Basis Spline
RBF	Radial Basis Function
TEAD	Taylor-Expansion Based Adaptive Design
TP	Target Point
WAE	Weighted Accumulative Error

greatest computational expense is associated with the evaluation of the high-fidelity responses of sample points, which are used to construct the surrogate model. Additionally, the position of sample points in the design domain can significantly affect the accuracy of the resulting models [2,18].

From the literature, sampling methods can be categorised into one-shot and adaptive sampling [19]. In one-shot sampling, the sample size and dataset are selected before model evaluation, without considering model error or accuracy. Common approaches include "space-filling" methods [20], like Latin Hypercube Distribution (LHD), orthogonal arrays, Hammersley, Sobol sequences, and uniform sampling [2,21]. However, determining the optimal sample size without prior knowledge of the target function is challenging. Adaptive sampling addresses this by iteratively updating the dataset based on the constructed metamodel, improving accuracy with fewer points [3,19,22]. The interested reader can find a detailed description of different types of sampling techniques in [2,21].

The strategy for selecting new points in adaptive surrogate models depends on its final objective. For regions with high error prediction or strong non-linearity, the focus is on local accuracy, using methods like Cross Validation (CV) errors [19] or gradient-based criteria [2,23]. However, pure local searches may cause samples clustering and poor global representation [21]. If the goal is high precision across the entire design space, a global search based on distance criteria may be more effective [19,21,22]. To improve overall accuracy, a combination of local and global search strategies is typically needed [19,21,22,24,25].

Lastly, it is noteworthy that for non-intrusive response surface methods a key challenge is the ability to approximate highly non-linear functions without overfitting, particularly with sparse data. Overfitted metamodels often fail to capture the underlying features, resulting in artificial, non-smooth responses and reduced ability to be generalised. To mitigate this, various techniques have been proposed, including early-stop methods, re-sampling, domain partitioning [13,26,27], and regularisation techniques [28,29]. Accordingly, different regularisation and domain partitioning techniques can be found in the literature in the context of kriging metamodels [30,31], polynomial chaos expansion based kriging metamodels [13], and metamodels based on basis of functions [32,33]. In general, in these works the different *smoothing* techniques allow obtaining a better global approximation of the output response.

This paper presents a metamodeling technique based on Non-Uniform Rational Basis Spline (NURBS) entities enriched with regularisation terms and integrated in an adaptive sampling strategy extending the works proposed by Vuillod, Zani et al. [4,5]. The metamodel is integrated in an adaptive sampling strategy, and following the work of Turner et al. [24], the most suitable adaptive sampling method for a NURBS-based metamodel are those based on k-fold CV, and those based on geometry features, such as distance and gradient criteria. Adaptive sampling strategies based on k-fold CV [34–36] allow to maximise the utilisation of training database saving computational resources, whilst the geometry-based strategies allow to fully leveraging the properties of NURBS entities, such as local support property and its analytical and deterministic natures. Regarding the construction of the NURBS-based metamodel, in [5], the metamodel is generated as a solution of a Constrained Non-Linear Programming Problem (CNLPP) whose resolution requires a three-step optimisation process of the continuous design variables of NURBS entities using the dimension separation formulation [37]. However, the strategy presented in [5] is applicable only for a structured domain which limits the possible gain in computational resources and limits the number of input variables that can be considered. The interested reader can find a detailed description of the previous formulation in [5,37]. To overcome this limitation, in this work the metamodel formulation is generalised to dataset obtained with unstructured and sparse domains. Moreover, the third step of the optimisation process [5] is improved by optimising concurrently both CPs coordinates and weights considering the correlation between these quantities due to the use of the adjoint method to derive the gradient of the model output responses. By allowing simultaneous

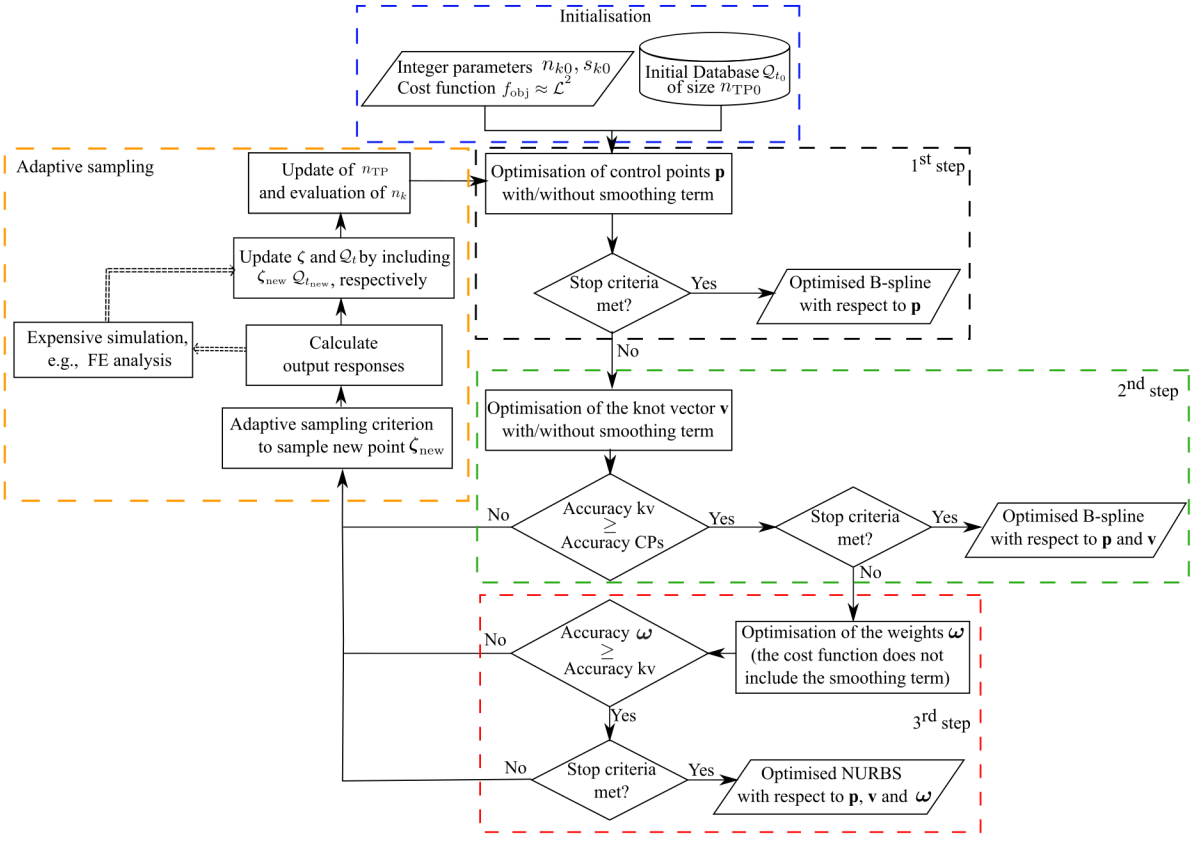


Fig. 1. Workflow of the general adaptive sampling process and of the three-step optimisation process used to generate the NURBS-based metamodel.

optimisation of weights and CPs coordinates, the metamodel can potentially capture highly non-linear trends, improving its fidelity in representing complex response hyper-surfaces with fewer computational resources. Finally, the regularisation term used to avoid overfitting, initially introduced in [26,38] to obtain smooth surface in the context of surface fitting problem, is here generalised by considering unstructured sampling techniques and the general formulation of the three-step optimisation process to generate the NURBS-based metamodel. To identify the most appropriate adaptive sampling method, a sensitivity analysis of the NURBS-based metamodel outcomes to one-shot strategies and sequential strategies is conducted. The aim of this sensitivity is to determine the sampling technique that concurrently enhances the accuracy of the metamodel and reduces the required number of samplings. Finally, the effectiveness of the proposed approach is evaluated through analytical benchmark functions and a complex real-world engineering application.

The remainder of this paper is organised as follows: Section 2 introduces the NURBS-based metamodel formulation and the related numerical strategy. The different adaptive sampling strategies are introduced in Section 3, and compared on different test functions in Section 4. Section 5 shows the application of the adaptive NURBS-based algorithm to a real-world application. Lastly, Section 6 ends the paper with conclusions and perspectives.

Notation. Upper-case bold letters and symbols are used to indicate matrices, lower-case bold letters and symbols indicate column vectors.

2. Surrogate model based on non-uniform rational basis spline

The main purpose of this section is to present the theoretical fundamentals of the metamodel based on NURBS entities. The workflow of the general methodology is shown in Fig. 1. The adaptive technique is integrated into the three-step optimisation process [5] and new samples are added when the accuracy criteria are not satisfied. More precisely, the main steps are (i) metamodel initialisation; (ii) construction of the surrogate models using the three-step optimisation strategy, and (iii) adaptive sampling approach.

2.1. Fundamentals of non-uniform rational basis spline hyper-surfaces

A NURBS hyper-surface is a vector-valued function that maps a subset of an N-dimensional domain onto a subset of an M-dimensional co-domain, denoted as $\mathbf{h} : \mathcal{A} \subseteq \mathbb{R}^N \rightarrow \mathcal{B} \subseteq \mathbb{R}^M$ [5,37,39]. The explicit form of a NURBS hyper-surface reads:

$$\mathbf{h}(\zeta_1, \dots, \zeta_N) = \sum_{i_1=0}^{n_1} \cdots \sum_{i_N=0}^{n_N} R_{i_1 \dots i_N}(\zeta_1, \dots, \zeta_N) \mathbf{p}_{i_1 \dots i_N}, \tag{1}$$

where $\mathbf{h} \in \mathcal{B}$ represents the coordinates of the NURBS hyper-surface evaluated at the normalised parametric coordinates $(\zeta_1, \dots, \zeta_N)$. The term $R_{i_1,\dots,i_N}$ is the rational basis function defined as:

$$R_{i_1 \dots i_N}(\zeta_1, \dots, \zeta_N) := \frac{\omega_{i_1 \dots i_N} \prod_{k=1}^{N} N_{i_k,s_k}(\zeta_k)}{\sum_{j_1=0}^{n_1} \cdots \sum_{j_N=0}^{n_N} \omega_{j_1 \dots j_N} \prod_{k=1}^{N} N_{j_k,s_k}(\zeta_k)}. \tag{2}$$

In Eq. (1), $\mathbf{p}_{i_1 \dots i_N} \in \mathbb{R}^M$ denotes the coordinates of the generic Control Point (CP). The total number of CPs forming the *control hyper-net* of the NURBS entity is $n_{\text{CP}} := \prod_{k=1}^{N}(n_k + 1)$. Each CP has an associated weight $\omega_{i_1 \dots i_N} > 0$, and all weights are stored in the array $\boldsymbol{\omega} \in \mathbb{R}^{n_{\text{CP}}}$. The functions N_{i_k,s_k} are Bernstein polynomials of degree s_k, recursively computed using the Cox–De Boor formula [39], based on the i_kth knot vector component v_{i_k}. The knot vector $\mathbf{v}_k^{\mathrm{T}} \in \mathbb{R}^{m_k+1}$ is given by:

$$\begin{aligned} &\mathbf{v}_k^{\mathrm{T}} = \{\underbrace{0, \dots, 0}_{s_k+1}, \mathrm{v}_{s_k+1}, \dots, \mathrm{v}_{m_k-s_k-1}, \underbrace{1, \dots, 1}_{s_k+1}\}, \\ &\mathbf{v}_k \in \mathbb{R}^{m_k+1}, \\ &m_k = n_k + s_k + 1. \end{aligned} \tag{3}$$

The basis functions possess several key properties, such as the *local support property*, which allows for local modifications of the NURBS entity without affecting regions outside the local support [5,39]. This property is useful for modelling localised strong non-linear behaviour. Another important property is the partition of unity, meaning that when all weights are equal, the NURBS hyper-surface reduces to a B-spline hyper-surface [39].

Generally, NURBS entities are widely applied in various fields, including topology optimisation [40,41], curve and surface fitting problems [26,42], anisotropy optimisation for composite structures [43], and generating surrogate models [5,37]. For a deeper insight in NURBS entities, the interested reader is referred to [5,24,39].

2.2. Surrogate modelling: Generation of the metamodel as a solution of a three-step optimisation problem

As discussed in [5] and shown in Fig. 1, the NURBS-based metamodel is generated by solving a three-step optimisation process using information from a training dataset $\mathcal{Q}_t$, made of n_{TP} Target Points (TPs). As shown in Fig. 1, the number n_{TP} of TPs composing the training set is determined iteratively when considering an adaptive sampling technique.

Specifically, starting with an initial dataset $\mathcal{Q}_{t0}$ of size n_{TP0} and the normalised design of experiments $\boldsymbol{\zeta}_0 \in \mathbb{R}^N$, where normalisation is applied over the range $[0, 1]$ or through distance-based methods [44], the metamodel is initialised by setting the values of the integer parameters n_{k0}, s_{k0}, according to [37,39,42]. The number of CPs along each parametric dimension is calculated using the formula

$$n_k = \lfloor \tau \sqrt[N]{n_{\text{TP}}} \rfloor - 1, \tag{4}$$

where $\lfloor \cdot \rfloor$ is the floor function, and $\tau = \dfrac{n_{\text{CP}}}{n_{\text{TP}}}$ represents the ratio between the number of CPs and TPs in dataset $\mathcal{Q}_t$. This ratio is generally set as $\tau \geq 1/3$, following literature guidelines [5,37,42], as it provides a good balance between accuracy and manageable data size. Lastly, the degree of the basis functions, s_k, is typically set to 2 or 3, depending on the problems non-linearity. The objective function is usually formulated as an $\mathcal{L}^2$ norm, as in least-squares problems, but modifications are required when incorporating a smoothing term. Finally, the weights $\omega_{i_1 \dots i_N}$ associated with each CP can either be initialised to the same value (e.g., the unity), which leads to B-spline hyper-surfaces, or determined using empirical rules [45].

As mentioned above, the initialisation of the variables tuning the metamodel is highly dependent on the training database $\mathcal{Q}_{t0}$ that is provided as input to the problem at hand. For the applications proposed in this paper, the initial training database is generated using a space-filling sampling, such as Latin Hypercube Distribution (LHD), and a low-discrepancy one, such as Hammersley sequence [46]. Moreover, corner points have been added a posteriori to the initial sampling techniques to be more suitable for a NURBS-based metamodel.

Regarding the workflow implemented for the generation of the NURBS-based metamodel shown in Fig. 1, while the stopping criteria are not met:

- Generate the metamodel optimising the coordinates of CPs.
- If the accuracy metrics are not satisfied, the inner components of the knot vector are optimised. The inner knot vector components of $\mathbf{v}_k$ can be initialised uniformly or using De Boor's algorithm [13] and optimised using a gradient-based algorithm, as discussed in [5].

- If the accuracy of the optimised B-spline with respect to CPs coordinates and $\mathbf{v}_k$ is worst than the B-spline obtained only after optimisation of CPs coordinates, the training database is updated using the adaptive sampling technique. Otherwise, the stop criteria are checked and if they are not met, the weights associated to the CPs are included in the problem formulation and optimised using a gradient-based algorithm as discussed in [5].
- If the stop criteria are not satisfied, the training database is updated using the adaptive sampling technique, and the integer variable n_k is updated as a function of n_{TP}.

For each step of the three-stage optimisation process an appropriate problem formulation and a dedicated resolution strategy has been implemented based on the approach presented in [26,42]. Specifically, the optimal coordinates of CPs are computed analytically from TPs by solving a least-squares minimisation problem [5,26]. On the contrary, for both second and third optimisation phases, the Global Convergent Method of Moving Asymptotes (GC-MMA) algorithm [47,48] is used and the gradient of the cost function with respect to knot vectors components and weights is analytically derived, as presented in [5].

Regarding the stop criteria of each step of the optimisation process, the accuracy of the resulting metamodel, which is measured with respect to the control (or validation) database $\mathcal{Q}_c$, has to be greater than a threshold value and it is measured through the use of accuracy metrics. Moreover, if there is no increase in accuracy between two successive steps of the optimisation process, the training database is enriched as it does not contain sufficient sample points to build the hyper-surface response. In the following of this work the considered metrics are:

- Determination coefficient R^2_α along α output direction defined as
$$\mathrm{R}^2_\alpha = 1 - \sum_{i=1}^{n_{\mathrm{TP}}} \frac{\left(q_{\alpha,i} - h_\alpha\left(\zeta_{1,i}, \dots, \zeta_{N,i}\right)\right)^2}{\left(q_{\alpha,i} - \bar{q}_\alpha\right)^2}, \tag{5}$$
where h_α refers to the metamodel prediction assessed for the generic point of the validation database $\mathcal{Q}_c$, whereas $\bar{q}_\alpha$ is the arithmetic mean of the considered output values $q_{\alpha,i}$ associated to the TPs belonging to $\mathcal{Q}_c$.
- The mean relative error $\bar{\varepsilon}_\alpha$ along α output direction defined as
$$\bar{\varepsilon}_\alpha = \frac{1}{n_{\mathrm{TP}}} \sum_{i=1}^{n_{\mathrm{TP}}} \frac{\left|q_{\alpha,i} - h_\alpha\left(\zeta_{1,i}, \dots, \zeta_{N,i}\right)\right|}{|q_{\alpha,i}|}, \tag{6}$$
- The local, i.e., point-wise, error expressed as:
$$e_{M_{\alpha,i}} = |h_{\alpha,i} - q_{\alpha,i}|, \quad i = 1, \dots, n_{\mathrm{TP}}, \tag{7}$$
- The normalised-root-mean-squared error (NRMSE) along the general α output direction that can be calculated as follows:
$$\mathrm{NRMSE} = \frac{100}{\max(q_{\alpha,i}) - \min(q_{\alpha,i})} \sqrt{\frac{\sum_{i=1}^{n_{\mathrm{TP}}} \left(q_{\alpha,i} - h_\alpha(\zeta_{1,i}, \dots \zeta_{N,i})\right)^2}{n_{\mathrm{TP}}}}. \tag{8}$$
The NRMSE allows comparison of the metamodels error values with regard to different results.

Of course, not all considered problems require the resolution of the three steps of the optimisation process. As it will be shown in Section 4, very often a B-spline entity with only CPs and knot-vector components included in the vector of the design variables is enough to satisfy the above accuracy metrics.

2.2.1. *Optimisation process implementation*

This section provides a brief overview of the various optimisation steps depicted in Fig. 1 and described in detail in [5]. Nevertheless, the extension of the formulation presented in [5] is presented more comprehensively. It is noteworthy that this work fully generalises the formulation of the metamodel generation, making it suitable for use in an adaptive workflow and unstructured domain, in contrast to previous works [5,26,37], which addressed the generalisation of the metamodel to multiple-inputs multiple-outputs problems using the dimension separation formulation and defined the smoothing term specifically for surface fitting problems.

The expression of the cost function in the most general case is:
$$f_{\mathrm{obj}} = \frac{1}{f_0} \sum_{\alpha=1}^{M} f_{\mathrm{obj},\alpha} = \frac{1}{f_0} \sum_{\alpha=1}^{M} \left[\sum_{i=1}^{n_{\mathrm{TP}}} \left(h_\alpha(\zeta_{1,i}, \dots, \zeta_{N,i}) - q_{\alpha,i}\right)^2 + \lambda J_\alpha \right], \tag{9}$$
where α is the generic output coordinate in the co-domain, $h_\alpha(\zeta_{1,i}, \dots, \zeta_{N,i})$ is the αth coordinate of the ith point of the hyper-surface evaluated at normalised parametric coordinates $\zeta_{k,i}$, and $q_{\alpha,i}$ is the corresponding TP. The term f_0 stands for the initial value of the objective function at the beginning of the optimisation. The term J_α is the smoothing term in the output direction α, and λ is a coefficient that aims at balancing the contributions of the first term, which is related to the distance between the NURBS entity and the TPs, and the second term, which is related to the local oscillations of the NURBS entity. The coefficient λ is assessed

according to the strategy discussed in [26,38] for surface fitting problems by generalising the theoretical framework to the case of hyper-surfaces. The general expression of the smoothing term J_α [49] is:

$$J_\alpha = \int_0^1 \cdots \int_0^1 \sum_{d_1+\cdots+d_N=d} \binom{d}{d_1,\ldots,d_N} \left(\frac{\partial^d h_\alpha(\zeta_1,\ldots,\zeta_N)}{\partial\zeta_1{}^{d_1}\cdots\partial\zeta_N{}^{d_N}} \right)^2 d\zeta_1\cdots\zeta_N, \tag{10}$$

where $d = \sum_{k=1}^{N} d_k$ is the maximum derivative order, which is set to $d = 2$ in the following of this paper. Quantities d_k $(k = 1,\ldots,N)$ in Eq. (10) are the partial derivative orders.

The following linear index is introduced to use a more compact matrix notation:

$$\kappa_{\mathrm{CP}} = 1 + \sum_{j=1}^{N} \left(i_j \prod_{k=1}^{j-1} (n_k + 1) \right). \tag{11}$$

By using the linear index of Eq. (11), the expression of the cost function in Eq. (9) can be written in a compact form as:

$$f_{\mathrm{obj}} = \frac{1}{f_0} \sum_{\alpha=1}^{M} \left[(\mathbf{R}\mathbf{p}_\alpha - \mathbf{q}_\alpha)^T (\mathbf{R}\mathbf{p}_\alpha - \mathbf{q}_\alpha) + \lambda J_\alpha \right], \tag{12}$$

where $\mathbf{h}_\alpha = \mathbf{R}\mathbf{p}_\alpha$ is the αth component of the NURBS entity computed at the parametric coordinates obtained from the input variables values used to generate the training dataset $\mathcal{Q}_t$, whilst $\mathbf{q}_\alpha \in \mathbb{R}^{n_{\mathrm{TP}}}$ is the vector that collects the αth component of the TPs in the training dataset obtained for various combinations of the input variables. The vector $\mathbf{p}_\alpha \in \mathbb{R}^{n_{\mathrm{CP}}}$ contains the αth coordinate of CPs, whereas matrix $\mathbf{R} \in \mathbb{R}^{n_{\mathrm{TP}} \times n_{\mathrm{CP}}}$ is the matrix of the basis functions whose components are defined as:

$$R_{\kappa_{\mathrm{TP}},\kappa_{\mathrm{CP}}} = R_{i_1\ldots i_N} \left(\zeta_{1,\kappa_{\mathrm{TP}}}, \ldots, \zeta_{N,\kappa_{\mathrm{TP}}} \right), \tag{13}$$

where the link between the indices i_j and κ_{CP} can be deduced from Eq. (11). Moreover, using the linear index, the smoothing term can be conveniently expressed in matrix form [26,38] as:

$$J_\alpha = \mathbf{p}_\alpha^{\mathrm{T}} \mathbf{E} \mathbf{p}_\alpha. \tag{14}$$

Matrix $\mathbf{E} \in \mathbb{R}^{n_{\mathrm{CP}} \times n_{\mathrm{CP}}}$ is built directly from the matrix of basis functions $\mathbf{R}$ and its components read:

$$E_{\gamma_1,\gamma_2} = \sum_{d_1+\cdots+d_N=d} \binom{d}{d_1,\ldots,d_N} \int_0^1 \cdots \int_0^1 \int_0^1 \cdots \int_0^1 \frac{\partial^d R_{i_1\ldots i_N}}{\partial\zeta_1^{d_1}\cdots\zeta_N^{d_N}} \frac{\partial^d R_{j_1\ldots j_N}}{\partial\zeta_1^{d_1}\cdots\zeta_N^{d_N}} d\zeta_1\ldots\zeta_N. \tag{15}$$

In the above formula, indices γ_1 and γ_2 vary like the linear index κ_{CP} of Eq. (11), whilst indices i_k, j_k vary like indices i_k of Eq. (1). Following the guidelines in [38], the constant λ, which controls the trade-off between smoothness and fit, is calculated as:

$$\lambda = \frac{\| \mathbf{R}^{\mathrm{T}}\mathbf{R} \|_F}{\| \mathbf{E} \|_F}, \tag{16}$$

where $\| \cdot \|_F$ stands for the Frobenius norm [38].

If the weights associated with CPs are all equal, Eqs. (12), (13), (15) and (16) can be simplified. Specifically, matrix $\mathbf{R}$ simplifies to matrix $\mathbf{N} \in \mathbb{R}^{n_{\mathrm{TP}} \times n_{\mathrm{CP}}}$ of Bernstein polynomials whose components read

$$N_{\kappa_{\mathrm{TP}},\kappa_{\mathrm{CP}}} = \prod_{j=1}^{N} N_{i_j,s_j} \left(\zeta_{j,\kappa_{\mathrm{TP}}} \right). \tag{17}$$

Consequently, the components of matrix $\mathbf{E} \in \mathbb{R}^{n_{\mathrm{CP}} \times n_{\mathrm{CP}}}$ become

$$\begin{aligned} E_{\gamma_1,\gamma_2} = & \sum_{d_1+\cdots+d_N=d} \binom{d}{d_1,\ldots,d_N} \left(\int_0^1 \frac{\partial^{d_1} N_{i_1,s_1}}{\partial\zeta_1^{d_1}} \int_0^1 \frac{\partial^{d_2} N_{i_2,s_2}}{\partial\zeta_2^{d_2}} \cdots \int_0^1 \frac{\partial^{d_N} N_{i_N,s_N}}{\partial\zeta_N^{d_N}} \right) \\ & \times \left(\int_0^1 \frac{\partial^{d_1} N_{j_1,s_1}}{\partial\zeta_1^{d_1}} \int_0^1 \frac{\partial^{d_2} N_{j_2,s_2}}{\partial\zeta_2^{d_2}} \cdots \int_0^1 \frac{\partial^{d_N} N_{j_N,s_N}}{\partial\zeta_N^{d_N}} \right) d\zeta_1\ldots\zeta_N. \end{aligned} \tag{18}$$

where the maximum value for the derivative $d_1 + \cdots + d_N = 2$ in the following of this paper.

Following the work of Bertolino et al. [26], the matrix $\mathbf{E}$, for a surface, can be defined as $\mathbf{E} = \mathbf{A} + 2\mathbf{B} + \mathbf{C}$ with $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathbb{R}^{n_{\mathrm{CP}} \times n_{\mathrm{CP}}}$. In the general case of a hyper-surface of dimension M with N inputs, the numerical implementation of the term $\mathbf{E}$ becomes very complicated. However, in the case of a B-spline, the matrix $\mathbf{E}$ can be expressed as

$$\mathbf{E} = \sum_{k=1}^{N} \left(\mathbf{A}_k + 2 \sum_{q>k}^{N} \mathbf{B}_{k,q} \right), \tag{19}$$

Table 1
Parameters of GC-MMA algorithm [47]. The variables n_{iter} depends on the problem at hand, whereas the variable n_{var} depends on the optimisation step.

Parameter	Value
move	0.1
alpha	0.1
Stop criterion	**Value**
Max evaluation of f_{obj}	$100 \times n_{\text{var}}$
Max number of iterations	n_{iter}
Threshold f_{obj}	10^{-8}
Threshold optimised variables	10^{-8}
Threshold on KKT norm	10^{-6}

where $\mathbf{A}_k, \mathbf{B}_{k,q} \in \mathbb{R}^{n_{\text{CP}} \times n_{\text{CP}}}$ have components defined as follows

$$A_{k,\gamma_1,\gamma_2} = \left(\prod_{l=1}^{k-1} \int_0^1 N_{i_l,s_l}(\zeta_l) d\zeta_l \int_0^1 \frac{\partial^2 N_{i_k,s_k}}{\partial \zeta_k^2} d\zeta_k \prod_{l=k+1}^{N} \int_0^1 N_{i_l,s_l}(\zeta_l) d\zeta_l \right) \times \left(\prod_{r=1}^{k-1} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \int_0^1 \frac{\partial^2 N_{j_k,s_k}}{\partial \zeta_k^2} d\zeta_k \prod_{r=k+1}^{N} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \right), \tag{20}$$

and

$$\begin{aligned} B_{k,q,\gamma_1,\gamma_2} &= \left(\prod_{l=1}^{k-1} \int_0^1 N_{i_l,s_l}(\zeta_l) d\zeta_l \int_0^1 \frac{\partial N_{i_k,s_k}}{\partial \zeta_l} d\zeta_k \prod_{l=k+1}^{q-1} \int_0^1 N_{i_l,s_l}(\zeta_l) d\zeta_l \int_0^1 \frac{\partial N_{i_q,s_q}}{\partial \zeta_q} d\zeta_q \right. \\ &\left. \prod_{l=q+1}^{N} \int_0^1 N_{i_l,s_l}(\zeta_l) d\zeta_l \right) \times \left(\prod_{r=1}^{k-1} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \int_0^1 \frac{\partial N_{j_k,s_k}}{\partial \zeta_k} d\zeta_k \right. \\ &\left. \prod_{r=k+1}^{q-1} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \int_0^1 \frac{\partial N_{j_q,s_q}}{\partial \zeta_q} d\zeta_q \prod_{r=q+1}^{N} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \right). \end{aligned} \tag{21}$$

In the first step of the optimisation process only B-spline entities are considered as the weights associated to CPs coordinates are set all equal to the same value. Following the approach presented in [5], the optimal values of the CPs coordinates are computed through the following formula:

$$\mathbf{p}_\alpha = \left(\mathbf{N}^{\text{T}} \mathbf{N} + \lambda \mathbf{E} \right)^{-1} \mathbf{N}^{\text{T}} \mathbf{q}_\alpha, \tag{22}$$

where matrix $\mathbf{N} \in \mathbb{R}^{n_{\text{TP}} \times n_{\text{CP}}}$ has components defined in Eq. (17). Eq. (22) results from the analytical resolution of a CNLPP problem of Eq. (12) with CPs as design variables.

After evaluating the optimal values of the CPs coordinates using the analytical approach, the inner knots and weights are optimised in the second and third optimisation phases of the process shown in Fig. 1, respectively. The solution of the optimisation problem is found using the GC-MMA algorithm [47], of which the parameters and convergence criteria used in this work are listed in Table 1. In order to solve the CNLPP problem by means of the GC-MMA algorithm, the partial derivatives of the objective function f_{obj} of Eq. (9) with respect to inner knots ($\boldsymbol{\xi}_1$) and weights ($\boldsymbol{\xi}_2$) variables must be computed. Employing the linear index of Eq. (11), the partial derivatives can be expressed as follows:

$$\frac{\partial f_{\text{obj}}}{\partial \xi_{j,l}} = \frac{1}{f_0} \left(\sum_{\alpha=1}^{M} 2(\mathbf{R}\mathbf{p}_\alpha - \mathbf{q}_\alpha)^{\text{T}} \frac{\partial \mathbf{h}_\alpha}{\partial \xi_{j,l}} + \lambda \frac{\partial J_\alpha}{\partial \xi_{j,l}} \right), \tag{23}$$

with $l = 1, \dots, n_{\text{knot}}$ when $j = 1$ (second optimisation phase) and $l = 1, \dots, n_{\text{CP}}$ when $j = 2$ (third optimisation phase). For $j = 1$ (second optimisation phase) the matrix $\mathbf{R}$ simplifies to matrix $\mathbf{N}$ of Bernstein polynomials whose components are defined in Eq. (17).

Specifically, in [5], in the last step of the optimisation process the CPs coordinates where not updated at each iteration of the optimisation process. In this work, on the other hand, the user can choose to dynamically adjust the coordinates of CPs by replacing matrix $\mathbf{N}$ with matrix $\mathbf{R}$ in Eq. (22) as a function of the optimised values of the weights for each iteration of the optimisation process. This improvement allows to obtain better results at the end of the third optimisation step as the metamodel is able to capture non-linear trends, improving its accuracy in representing complex response surfaces. Moreover, the contribution associated to smoothing term is neglected by setting $\lambda = 0$. The analytical expression of the derivatives of quantities $\mathbf{h}_\alpha$ and J_α in Eq. (23) is provided in Appendix.

The effectiveness of the dynamic optimisation of CPs and weights is shown through a numerical application on a benchmark function. The problem studied corresponds to the Drop-Wave function already used in [5] to show the influence of the optimisation of the knots and weights on the metamodel accuracy.

Table 2
Parameters of the metamodel, training database size, input variables and bounds for the Drop-Wave function problem.

	Input variables	s_k	n_{TP}	n_k	τ	β_i	$\omega_{i_1 \dots i_N}$
M3	(x_1, x_2)	2	225	12	4/5	$]0, 1[$	$[1, 10]$

Table 3
Results in terms of f_{obj}, R^2 and $\bar{\varepsilon}$ for the Drop-wave function assessed on the corresponding training dataset $\mathcal{Q}_t$. The gain is evaluated comparing the value of the objective function at the end of the first phase of the optimisation process with that found at the end of the third phase.

	First step	Second step	Third step	Gain [%]	Formulation [5]
f_{obj}	1.000	0.201	0.058232	94.2	–
R^2	0.6	0.9357	0.983	–	0.948
$\bar{\varepsilon}$	0.256	0.0742	0.0224	91.25	0.0705

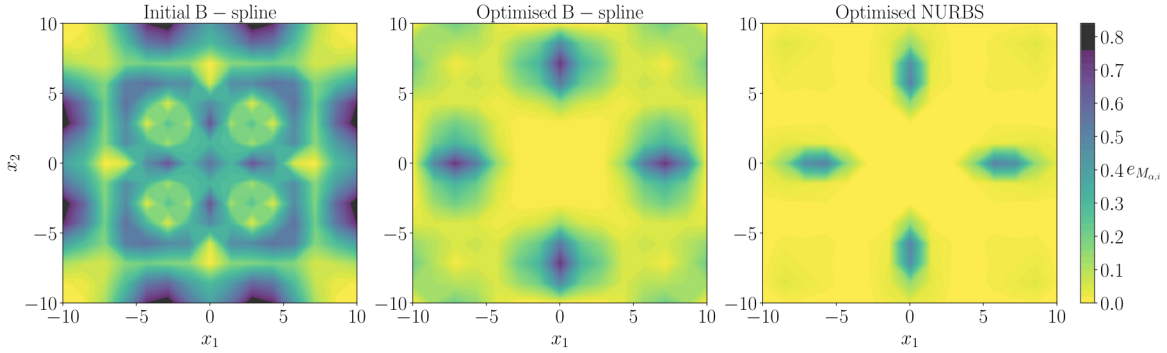


Fig. 2. Contour plot of the local error $e_{M_{\alpha,i}}$ with ratio $\tau = 4/5$ for the approximation of drop-wave surface using the general formulation for the weights optimisation.

The non-linear 2D function $f(x_1, x_2)$, with $N = 2$ and $M = 1$ reads:

$$f(x_1, x_2) = 1 - \cos\left(2\pi\sqrt{x_1^2 + x_2^2}\right) + 0.1\sqrt{x_1^2 + x_2^2}, \tag{24}$$

with $x_1, x_2 \in [-10,\ 10]$.

The parameters used to generate the metamodel along with the bounds of the inner knots and weights used during second and third optimisation phases, and the size of the training database are reported in Table 2. It is noteworthy that the TPs forming the training database have been obtained via uniform sampling technique. The whole optimisation process has been carried out without including the smoothing term in the formulation of the cost function.

Fig. 2 shows the local error for metamodel M3 for each optimisation phase between the points of the training set $\mathcal{Q}_t$ and the metamodel. The associated performances assessed on the corresponding training dataset, in terms of $\bar{\varepsilon}$ and R^2, are reported in Table 3 for both the problem formulation presented in [5] and that proposed in this paper.

By comparing the results listed in Table 3, one can notice that updating the coordinates of CPs during the third optimisation step has a considerable effect on the results.

Lastly, Fig. 3 shows the effects of CPs coordinates calculation during optimisation plotting the local error $e_{M_{\alpha,i}}$ between the points of the training set $\mathcal{Q}_t$ on metamodels obtained at the end of the third optimisation phase with the formulation presented in this paper and the dimension separation formulation presented in [5].

The associated performances in terms of $\bar{\varepsilon}$ and R^2 for the two metamodels are $\text{R}^2 = 0.983$ and $\bar{\varepsilon} = 0.0223$ for the general formulation, and $\text{R}^2 = 0.948$ and $\bar{\varepsilon} = 0.0705$ for the dimension separation formulation. These last results, in accordance with Fig. 3, show that updating the coordinates of CPs in the last step of the optimisation process remarkably improves the final results.

The use of the general formulation, however, highly increases the computational costs and memory usage making it impractical for high-dimensional problems or with a large number of variables. For this reason, different sampling strategies must be employed to train the metamodel on training set of reduced size to overcome this problem.

3. Adaptive sampling strategies for a NURBS-based metamodel

The main idea at the basis of an adaptive sampling method is that for a transfer function $\mathbf{h} : \mathcal{A} \subseteq \mathbb{R}^N \rightarrow \mathcal{B} \subseteq \mathbb{R}^M$, the initial training dataset $\mathcal{Q}_{t_0}$ is constituted of n_{TP0} TPs $\boldsymbol{\zeta}$. The new sample $\boldsymbol{\zeta}_{\text{new}}$ is iteratively updated by solving an optimisation problem

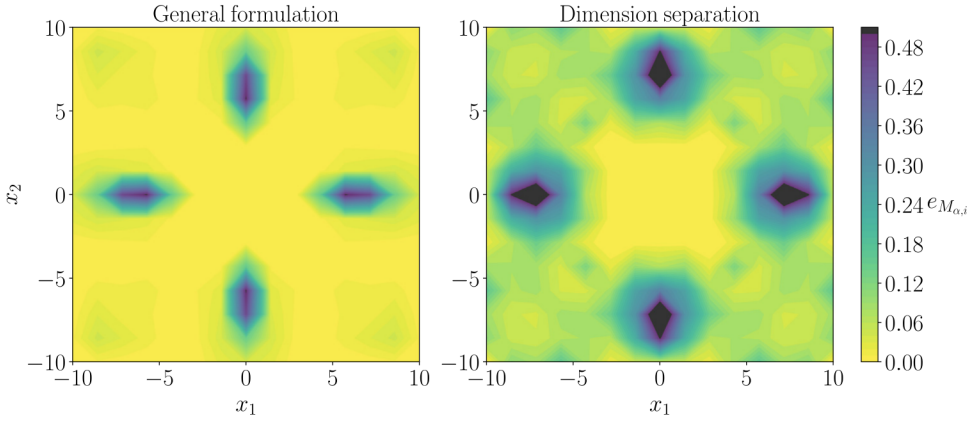


Fig. 3. Contour plot of the relative error $e_{M_{\alpha,i}}$ for general formulation and the dimension separation one [5] at the third step of the flowchart in Fig. 1 on reduced training database and ratio $\tau = 4/5$.

based on a global–local sampling criterion and defined as follows [19]:

$$\boldsymbol{\zeta}_{\text{new}} = \underset{\boldsymbol{\zeta}^\star \in \mathbb{R}^N}{\text{argmax}} \left(\omega_{\text{local}} \text{loc}(\boldsymbol{\zeta}) + \omega_{\text{global}} \text{glob}(\boldsymbol{\zeta}) \right), \tag{25}$$

where $\boldsymbol{\zeta}^\star$ is the feasible optimal solution, and ω_{local} and ω_{global} are the weights used for the local exploitation and global exploration criteria, indicated as loc and glob, respectively.

For a NURBS-based metamodel, as stated in [23], the most suitable local–global adaptive techniques can be classified with respect to two main categories:

- The Cross Validation (CV) based adaptive sampling methods that use the $k-$folds cross validation error as a local exploitation predictor, since it allows measuring the sensitivity of the metamodel to the lack of information, and a distance-based criterion as a global exploration predictor. The distance-based allows unveiling regions of the input domain with high prediction error adding distance constraints in the optimisation problem [19,21]. The distance criterion can be continuous and defined using a $\mathcal{L}^2$ norm operator, or discontinuous (e.g., Voronoi tessellation [50], Delaunay triangulation [21]). The CV-based approaches adapted for NURBS-based metamodels are detailed in Section 3.1.
- The geometry features based adaptive samplings techniques use a gradient-based search for the local exploitation predictor, and a distance-based criterion for the global exploration predictor. The geometry features based approaches employed for NURBS-based metamodels are detailed in Section 3.2.

The interested reader is referred to [19,21,23] for a survey on the adaptive sampling techniques available in the literature. Moreover, from a numerical point of view, the optimisation process is performed using GC-MMA algorithm [47] with parameters and convergence criteria listed in Table 1. The maximum number of iterations is set to $n_{\text{iter}} = 100$ to reduce computational costs.

3.1. CV-based adaptive sampling methods

The adaptive sampling techniques based on the CV approach allow for a maximum utilisation of the training database. Generally, among the $k-$folds approaches, the most used one is the leave-one-out cross validation (LOOCV) error in which k is equal to the size of $\mathcal{Q}_t$. In fact, the entire dataset $\mathcal{Q}_t$ of size n_{TP} is divided in k mutually exclusive subsets $\mathcal{Q}_{t_i}$ and, $\forall j \in [1, n_{\text{TP}}]$, an auxiliary metamodel $\mathbf{h}_{-j}$ is trained on $n_{\text{TP}} - 1$ samples. The LOOCV error $e_{\text{LOOCV}}(\boldsymbol{\zeta}_i)$ is then evaluated at point $\boldsymbol{\zeta}_i$ as

$$e_{\text{LOOCV},j}(\boldsymbol{\zeta}_i) = |\mathbf{h}(\boldsymbol{\zeta}_i) - \mathbf{h}_{-j}(\boldsymbol{\zeta}_i)|, \quad \forall j \in [1, n_{\text{TP}}]. \tag{26}$$

This local measure is repeated for all the samples and the total LOOCV error is evaluated as:

$$e^{\text{tot}}_{\text{LOOCV}} = \sqrt{\frac{1}{n_{\text{TP}}} \sum_{i=1}^{n_{\text{TP}}} \left(\mathbf{h}(\boldsymbol{\zeta}_i) - \mathbf{h}_{-j}(\boldsymbol{\zeta}_i) \right)^2}. \tag{27}$$

However, to make the value of LOOCV error continuously available over the entire domain, a degree of influence criterion is introduced to estimate the error of any unobserved points as a function of $e_{\text{LOOCV},j}(\boldsymbol{\zeta}_i)$ [21,34,35]. Lastly, a distance-based criterion is used for the exploration part to avoid clustering phenomenon (points too close to each other). In general, the new sample point is defined as a solution of the constrained optimisation problem

$$\begin{aligned} &\boldsymbol{\zeta}_{\text{new}} = \max_{\boldsymbol{\zeta}^\star \in \mathbb{R}^N} \; e_{\text{LOOCV},j}(\boldsymbol{\zeta}_1), \ldots, e_{\text{LOOCV},j}(\boldsymbol{\zeta}_{n_{\text{TP}}}), \\ &\text{subject to: } \min \mathcal{L}^2(\boldsymbol{\zeta}) \geq S, \end{aligned} \tag{28}$$

with S a space-filling metric value that is defined as a distance threshold.

Among the existing strategies [21], in this paper the Accumulative Error (AE) [35], Weighted Accumulative Error (WAE) [34], and the Cross Validation Voronoi (CVV) [50] adaptive sampling techniques, initially implemented for kriging metamodels, are adapted to NURBS-based metamodel. For every strategy, the CV criterion is used for exploitation, and a distance-based criterion is employed for exploration of the design domain, and the new sample point is obtained by solving a constrained optimisation problem. The numerical implementation of these techniques can be found in [21,34,35,50]. When applied to a metamodel based on NURBS entities the gradient of the optimisation problem is obtained analytically. Moreover, for a metamodel based on NURBS entities, this method only works correctly by keeping the samples on the boundaries during calculation. It is noteworthy that, when employing the Cross Validation Voronoi (CVV) technique, for high dimensional problems, the discontinuous distance criterion is more complex than in 2D or 3D due to the nature of N-D manifold. In fact, instead of a polygon (2D) or polyhedron (3D) the search has to be done in a polytope with a method based on the concept of half spaces [51].

3.2. Geometry-based adaptive sampling methods

Adaptive sampling methods based on exploitation of geometric features are the most suitable for a metamodel based on B-spline or NURBS entities as they allow the exploitation of their analytic nature and the local support property [23]. Among the different approaches available in the literature [21], in this work only the Taylor-Expansion Based Adaptive Design (TEAD) is used and discussed.

The TEAD technique, initially introduced by Mo et al. [25] for kriging, uses a gradient-based criterion for exploitation and a distance-based criterion for exploration with non-fixed balance between the two contributions. The new sample point is obtained solving a discontinuous optimisation problem defined as follows

$$\boldsymbol{\zeta}_{\text{new}} = \max_{\boldsymbol{\zeta}^\star \in C_{\text{cand}}} \left(\frac{D_{\min}(\boldsymbol{\zeta}^\star)}{\max\limits_{\boldsymbol{\zeta} \in C_{\text{cand}}} D_{\min}(\boldsymbol{\zeta})} + w_{\text{TEAD}}(\boldsymbol{\zeta}^\star) \frac{R(\boldsymbol{\zeta}^\star)}{\max\limits_{\boldsymbol{\zeta} \in C_{\text{cand}}} R(\boldsymbol{\zeta})} \right). \tag{29}$$

The term $\dfrac{D_{\min}(\boldsymbol{\zeta}^\star)}{\max_{\boldsymbol{\zeta} \in C_{\text{cand}}} D_{\min}(\boldsymbol{\zeta})}$ is the normalised distance metric defined as the minimum distance between sample points in the existing design of experiments and a set of candidate points $\boldsymbol{\zeta} \in C_{\text{cand}}$, uniformly distributed in the domain (by considering the LHD sampling technique in the following of this work). In general, the number of candidate points n_{cand} is a multiple of TPs, for which the output responses are not evaluated. Accordingly, they are points only added in the input domain that do not have a calculated image in the co-domain.

The gradient-based exploitation, defined as a Taylor-expansion based scheme, is used to obtain the local information through the metamodel gradient. Thanks to the nature of NURBS entities, the gradient is evaluated analytically. The term $R(\boldsymbol{\zeta})$ represents the local residual and it can be formulated as

$$R(\boldsymbol{\zeta}) = |\mathbf{h}(\boldsymbol{\zeta}) - t(\boldsymbol{\zeta})|, \tag{30}$$

where $t(\boldsymbol{\zeta})$ is the first-order Taylor expansion of $\mathbf{h}(\boldsymbol{\zeta})$ defined as

$$t(\boldsymbol{\zeta}, \mathbf{a}) = \mathbf{h}(\mathbf{a}) + \nabla \mathbf{h}(\mathbf{a})(\boldsymbol{\zeta} - \mathbf{a}), \tag{31}$$

where $\mathbf{a}$ is the sample having the smallest distance to $\boldsymbol{\zeta}$, and $\nabla \mathbf{h}(\mathbf{a})$ is the gradient of $\mathbf{h}(\boldsymbol{\zeta})$ in $\mathbf{a}$ [25]. It is noteworthy that the exploitation term is weighted using the weight $w_{\text{TEAD}}(\boldsymbol{\zeta})$ given by

$$w_{\text{TEAD}}(\boldsymbol{\zeta}) = 1 - \frac{D_{\min}(\boldsymbol{\zeta})}{L_{\max}}, \tag{32}$$

with $L_{\max}$ the maximum distance between two points in the domain. The weight allows for an optimal compromise between exploration and exploitation criteria.

4. Application of adaptive samplings techniques to analytical functions

The aforementioned adaptive sampling methods are adapted to the NURBS-based metamodel and tested on several analytical functions taken from the literature. The test functions are shown in Fig. 4 and their analytical form is reported in the following.

- Droplet function [20]

$$D(\zeta_1, \zeta_2) = -4e^{-\frac{25}{8}(\zeta_1^2 + \zeta_2^2)} + 7e^{-\frac{125}{4}(\zeta_1^2 + \zeta_2^2)}, \tag{33}$$

  with $(\zeta_1, \zeta_2) \in [-1, 1] \times [-1, 1]$ and represented in Fig. 4(a).

- Franke's function [20]

$$F(\zeta_1, \zeta_2) = \frac{3}{4}\left[e^{-\frac{1}{4}((9\zeta_1 - 2)^2 + (9\zeta_2 - 2)^2)} + e^{-\frac{1}{49}(9\zeta_1 + 1)^2 - \frac{1}{10}(9\zeta_2 + 1)^2} \right] + \frac{1}{2} e^{-\frac{1}{4}((9\zeta_1 - 7)^2 + (9\zeta_2 - 3)^2)} - \frac{1}{5} e^{-(9\zeta_1 - 4)^2 - (9\zeta_2 - 7)^2}, \tag{34}$$

  with $(\zeta_1, \zeta_2) \in [0, 1] \times [0, 1]$ and represented in Fig. 4(b).

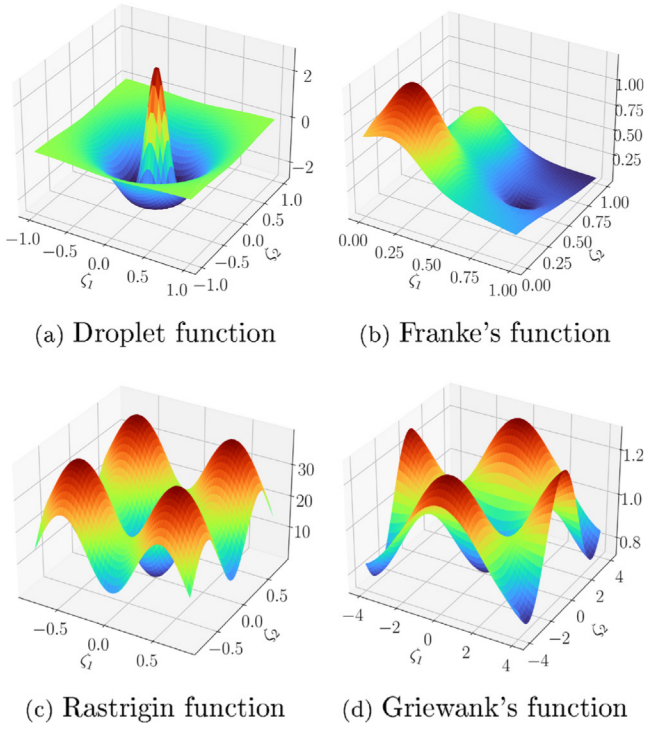


Fig. 4. Test functions for comparing sampling methods on the generation of a NURBS-based metamodel.

- Rastrigin function [2]

$$R(\zeta_1, \zeta_2) = 20 + \zeta_1^2 + \zeta_2^2 - 10\left(\cos\left(2\pi\zeta_1\right) + \cos\left(2\pi\zeta_2\right)\right), \tag{35}$$

  with $(\zeta_1, \zeta_2) \in [-0.8, 0.8] \times [-0.8, 0.8]$, represented in Fig. 4(c).
- the Griewank's function with $N = 4$ [35]

$$G(\boldsymbol{\zeta}) = \sum_{i=1}^{N} \frac{\zeta_i^2}{4000} - \prod_{i=1}^{N} \cos\left(\frac{\zeta_i}{\sqrt{i}}\right) + 1, \tag{36}$$

  with $\zeta_i \in [-4,\ 4]$ and represented in Fig. 4(d), for $N = 2$.

The test functions are chosen to represent non-linear multi-modal problems. The initial training database $\mathcal{Q}_{t_0}$ is defined using Hammersley low-discrepancy distribution and the optimised LHD distribution. The stop criterion in terms of maximum number of samples is set as $n_{\mathrm{TP}}^{\max} = 150$ for the 2D cases, and $n_{\mathrm{TP}}^{\max} = 200$ for the 4D one, to limit the computational costs. Moreover, for the TEAD adaptive sampling technique, a number of candidate samples equal to $n_{\mathrm{cand}} = 200 n_{\mathrm{TP}}$ is considered for the different test functions of this section. The accuracy of final results is evaluated on a validation database $\mathcal{Q}_c$ obtained from fine grid of 50×50 evaluation points for 2D cases, and 50^4 points for the 4D one, in terms of $\overline{\varepsilon}$ and NRMSE of Eqs. (6), and (8), respectively. The metamodel is considered accurate enough when $\overline{\varepsilon} < 5\%$ and NRMSE$< 2\%$ for 2D cases, and when $\overline{\varepsilon} < 10\%$ and NRMSE$< 5\%$ for the 4D case.

For the 2D test cases, the initial number of samples is set as $n_{\mathrm{TP0}} = 29$ (25 samples + 4 samples at design space corners), while for the 4D case it is considered an initial number of samples of $n_{\mathrm{TP0}} = 100$. For computational reasons, the adaptive techniques on the Griewank's function are compared only for Hammersley distribution.

Lastly, the training is performed by only considering the formulation of the problem without the smoothing term. The parameters used to generate the metamodel along with the bounds of the inner knots and weights used during second and third optimisation phases are reported in Table 4. The number of CPs along each input direction is evaluated using Eq. (4) for each iteration of the training process.

The results obtained after constructing metamodels through four different adaptive sampling methods are listed in Table 5, together with the gain (in brackets) in accuracy compared to the beginning of the training. For the 2D cases, it can be seen that TEAD and CVV techniques bring the best performances after convergence in spite of a slight higher number of samples, as listed in Table 5. Moreover, all the methods have relative improvement of $\overline{\varepsilon}$ from the initial metamodel based on 29 samples. Specifically, for the Hammersley sampling the relative improvement ranges from 7% for AE technique to 97% in the case of TEAD technique, whereas for the LHD the minimum improvement is of 33.7% for TEAD strategy and the maximum one is 96% for WAE technique.

Table 4
Input variables and bounds for the benchmark function problems.

	s_k	τ	β_i	$\omega_{i_1 \dots i_N}$
Droplet function Franke function Rastrigin function Griewank function	2	4/5	]0, 1[	[1, 10]

Table 5
Results in terms of n_{TP}, $\bar{\varepsilon}$, and NRMSE at convergence for the benchmark functions. The gain (in brackets) is evaluated comparing the value of the metric at the beginning and at the end of training. The worst and best results for each considered case are highlighted using red and green colours, respectively. (For interpretation of the references to colour in this table, the reader is referred to the web version of this article.)

Test function		LHD			Hammersley		
		n_{TP}	$\bar{\varepsilon}$ %	NRMSE%	n_{TP}	$\bar{\varepsilon}$ %	NRMSE %
Droplet	ACE	69	4.71 (96%)	1 (96%)	69	4.23 (97.1%)	0.8 (93.9%)
	WAE	**100**	4.06 (96.6%)	1.73 (93%)	88	3.897 (97.3%)	0.76 (94.2%)
	CVVor	76	4.617 (96.15%)	1.61 (96.9%)	**89**	4.03 (97%)	1.491 (88.6%)
	TEAD	71	4.768 (95%)	0.58 (97.7%)	77	3.72 (97.4%)	0.56 (95.7%)
Franke	ACE	49	4.71 (82%)	1.81 (69.2%)	**98**	3.67 (85.1%)	1.508 (76.44%)
	WAE	107	3.42 (86.9%)	0.54 (90.8%)	77	2.85 (88.4%)	2.17 (66%)
	CVVor	86	2.29 (91.3%)	0.43 (92.6%)	72	4.74 (80.73%)	1.697 (66.1%)
	TEAD	**128**	3.29 (87.4%)	0.63 (89.3%)	78	4.92 (80%)	1.01 (84.2%)
Rastrigin	ACE	33	4.14 (41.2%)	1.9 (51.1%)	33	4.35 (7.2%)	1.999 (10.4%)
	WAE	33	4.96 (29.6%)	1.85 (52.4%)	38	4.19 (10.7%)	1.87 (16.1%)
	CVVor	**36**	4.545 (35.5%)	1.97 (49.3%)	**41**	4.14 (11.73%)	1.98 (11.2%)
	TEAD	33	4.669 (33.7%)	1.84 (52.6%)	36	3.795 (19.1%)	1.71 (23.3%)

Fig. 5 shows the relative improvement in terms of $\bar{\varepsilon}$, and NRMSE compared to the metamodel generated using the one-shot sampling strategy. Nevertheless, as the metamodel is run until achievement of convergence criteria in terms of $\bar{\varepsilon}$ and NRMSE, the results of each technique shown in Fig. 5 are not obtained under the same conditions, neither for the number of CPs nor for the step of the adaptive optimisation process at convergence. Accordingly, the improvement shown in Fig. 5 is obtained generating the one-shot metamodel with the same number of n_{TP} and n_{CP} and optimisation step after convergence for each adaptive techniques.

From Fig. 5, it can be observed that only TEAD technique presents an improvement in terms of performances if compared to the one-shot strategy for all the presented 2D test cases. Lastly, it can be noticed that the stronger the non-linear behaviour of the function, the more samples are needed to satisfy the convergence criteria.

The results reported in Table 5 and shown in Fig. 5 for the LHD initial sampling cannot represent the advantages and disadvantages of different sampling methods adequately because the samples are generated randomly when considering the LHD technique. Accordingly, every metamodel is constructed 10 times with the same sampling method with a stop criterion for the maximum number of samples at $n_{TP}^{max} = 50$ for computational reasons. The results are shown in statistical graphics of Fig. 6, in terms of NRMSE, to illustrate the deviations of the accuracy of each metamodel. Boxplots allow splitting numerical data in quartiles. The 50th percentile value (median) is indicated by the centre line of the boxplot, while the 25th and 75th percentiles are shown by the bottom and top lines of the box, respectively. Small boxes represents results with low standard deviations, exhibiting, hence, low uncertainty in the predictions. The points outside the horizontal lines indicate erroneous/unexpected results. From the results shown in Fig. 6, one can notice that the median values of NRMSE for TEAD technique is smaller compared to the other methods for the Franke's function (Fig. 6(b)), and Rastrigin function (Fig. 6(c)), whilst for the Droplet function, the lowest median value of NRMSE is found for the CVV method. Nonetheless, the size of the box for this method is important suggesting high variability of the results. Accordingly, also for this test case, the TEAD predictions have a higher level of confidence and global accuracy. Lastly, outliers occur because, with a limited sample size and the randomness of initial sampling, not all 10 runs meet the accuracy criteria. As a result, the algorithm reaches the maximum number of samples allowed before achieving the desired accuracy. Nonetheless certain techniques offer better approximation capabilities with reduced variability and among the strategies proposed, the TEAD method consistently exhibits lower discrepancy (smaller box) and a reduced overall median compared to the other adaptive sampling methods for a NURBS-based metamodel. Moreover, the erroneous points for this method are very close to the average value.

For the 4D benchmark function, on the other hand, it is noteworthy that only TEAD sampling method allows respecting the imposed convergence criteria, in terms of $\bar{\varepsilon}$ and NRMSE, for a number of samples $n_{TP} < 200$, as listed in Table 6. However, all the adaptive sampling strategies have relative improvement from the initial metamodel based on 100 samples.

Lastly, Fig. 7 shows the improvement in performances if compared to the one-shot strategy with the number of samples after convergence or after meeting the stop criterion (maximum number of samples) for the 4D Griewank's function. For this case, the results are obtained after the third step of the optimisation process of flowchart of Fig. 1. As for the 2D cases, the TEAD technique is that ensuring the best improvement compared to the one-shot technique with same initial conditions.

In conclusion, the TEAD method guarantees better final results compared to the other proposed strategies and it is used in the following of this paper as adaptive sampling method for a NURBS-based metamodel. In fact, on the one hand, this strategy allows

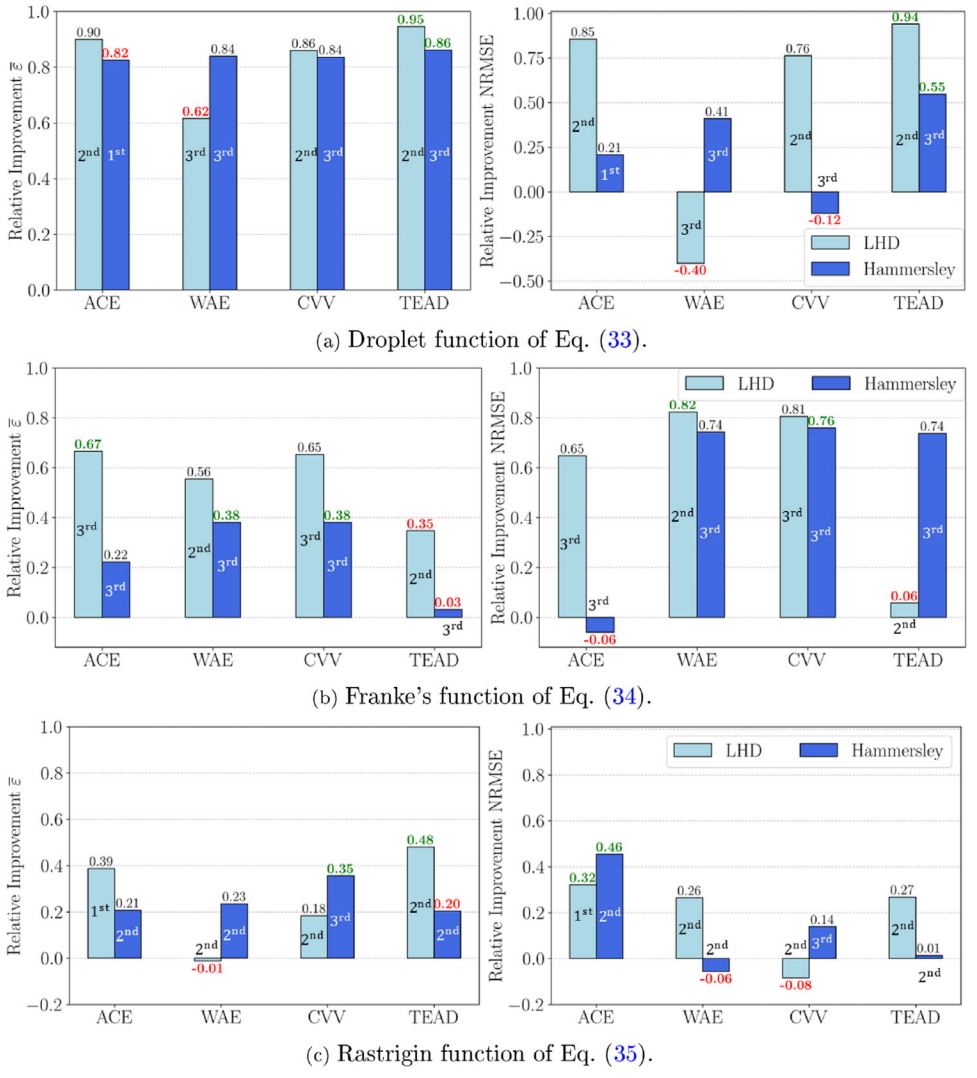


Fig. 5. Performance evolution in terms of $\bar{\varepsilon}$, and NRMSE from the one-shot strategy to the metamodel generated using adaptive sampling using the same number of samples and at the same step of flowchart of Fig. 1. The worst and best results for each displayed case are highlighted using red and green colours, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

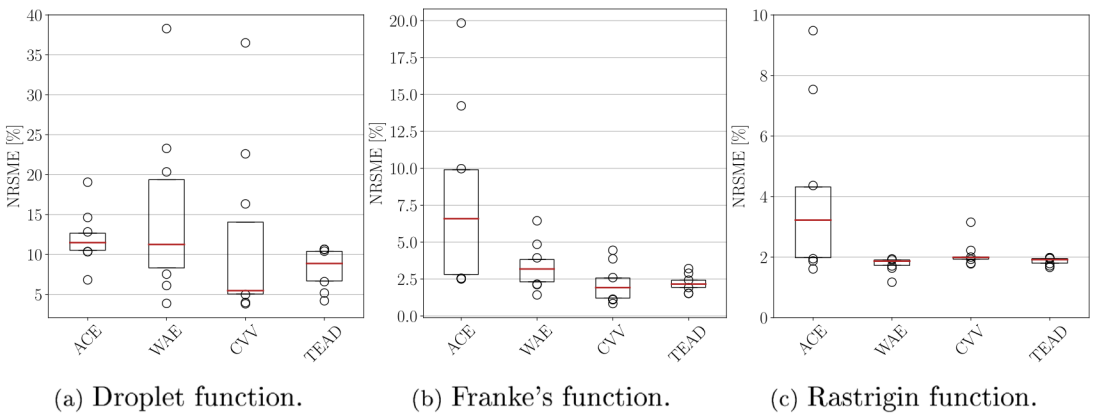


Fig. 6. Assessment of metamodels: mean NRMSE between different adaptive samplings designs.

Table 6
Results in terms of n_{TP}, $\bar{\varepsilon}$, and NRMSE at convergence for the 4D benchmark function. The gain (in brackets) is evaluated comparing the value of the metric at the beginning and end of training. Red and green colours represents the worst and best results. (For interpretation of the references to colour in this table, the reader is referred to the web version of this article.)

		n_{TP}	$\bar{\varepsilon}$ %	NRMSE %
Griewank	AE	**200**	**9.58** (70.6%)	**6.88** (73.54%)
	WAE	**200**	**8.2447** (74.2%)	**5.12** (80.3%)
	CVV	**200**	**8.56** (73.2%)	**6.97** (80.4%)
	TEAD	168	7.638 (76.5%)	4.958 (80.9%)

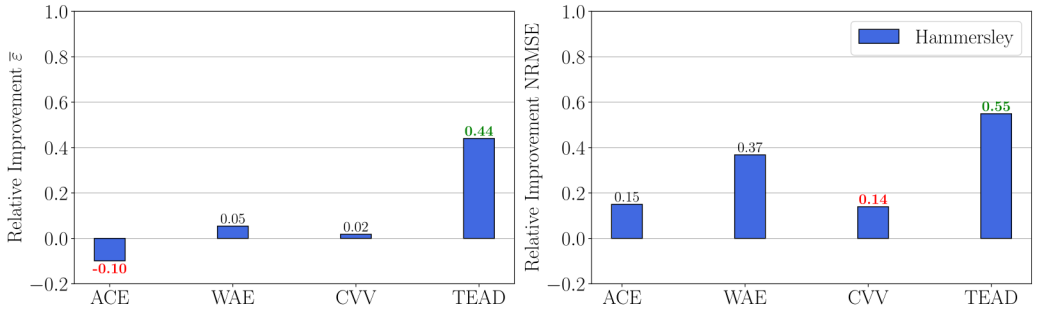


Fig. 7. Performance evolution from the one-shot strategy to the metamodel generated using adaptive sampling using the same number of samples for the 4D Griewank's function. Red and green colours represents the worst and best results. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 7
Design variables for the training of the considered metamodeling strategies. All the other parameters are equal to their default values [14].

Kriging	RBF
$\theta = 10^{-2}$	$d_0 = 4$

obtaining results with less variability and a positive improvement for all the presented examples by considering, on the other hand, a slightly higher number of added samples. Moreover, it uses the gradient information thereby making it particularly suitable for a NURBS-based metamodel.

4.1. Comparison with classical approaches

The effectiveness of TEAD adaptive sampling method coupled with the NURBS-based metamodel is further assessed by comparing it with classical surrogate modelling techniques, specifically Kriging and RBF, which are, of course, coupled with the same adaptive sampling method. Two benchmark functions are used for this analysis: the 2D analytical function of Eq. (34) and the 4D analytical function of Eq. (36).

The classical approaches were implemented using the Python surrogate modelling toolbox [14], with the primary hyperparameters detailed in Table 7. The mathematical formulations and implementation details of Kriging and RBF are provided in [14] to which the reader is addressed for a deeper insight in the matter.

The results, summarised in Table 8, demonstrate that NURBS-based metamodel has better performances compared to classical surrogate models in terms of both accuracy and computational efficiency. For the 2D case, the NURBS-based metamodel with the LHD initial training database achieves the lowest NRMSE of 0.63% (89.3% improvement) and an error reduction of 3.29% (87.4% improvement). Using the Hammersley initial training database, Kriging exhibits more accurate solutions with fewer needed samples. RBF shows lower gains and fails to meet the imposed stop and convergence criteria satisfactorily.

For the 4D case, the NURBS-based metamodel shows clear advantages when coupled with TEAD adaptive sampling, achieving an NRMSE of 4.96% compared to 9.68% and 22% for Kriging and RBF, respectively.

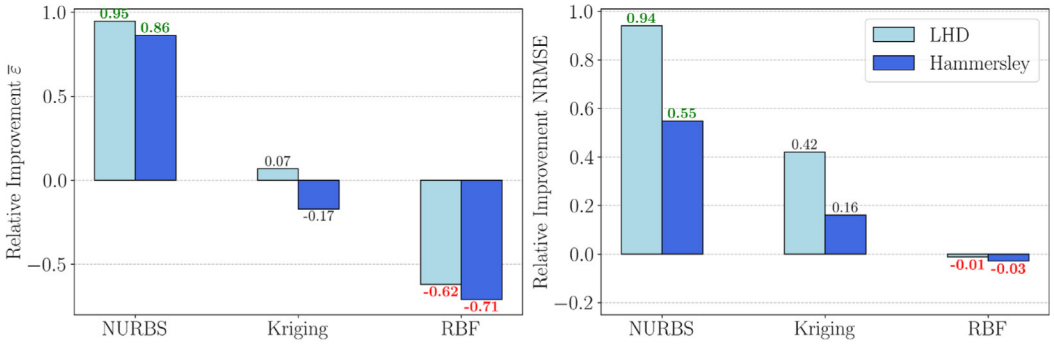
Nonetheless, all the proposed metamodeling strategies have relative improvement of $\bar{\varepsilon}$ and NRSME from the initial metamodel based on 29 and 100 samples for the 2D and 4D case, respectively.

The plots in Fig. 8 show the relative improvement in terms of $\bar{\varepsilon}$, and NRMSE for each surrogate model against the one-shot training database using the same number of samples after convergence.

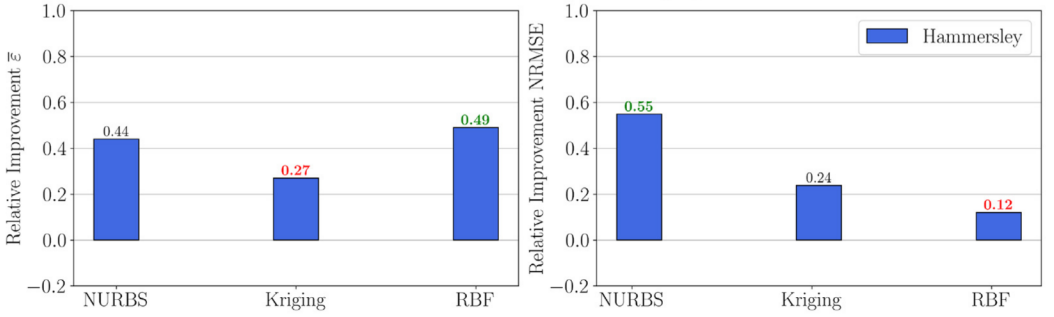
As shown in Fig. 8(a) for the 2D case, only the NURBS-based metamodel demonstrates consistent improvement over one-shot strategies. This can be attributed to the analytical evaluation of gradients in the proposed approach, which provides a significant advantage over numerical gradient evaluations used in Kriging and RBF.

Table 8
Comparison of different metamodels. Results in terms of n_{TP}, $\bar{\varepsilon}$, and NRMSE at convergence for benchmark functions of Eqs. (34), (36). The gain (in brackets) is evaluated comparing the value of the metric at the beginning and at the end of training.

Type of metamodel		n_{TP}	$\bar{\varepsilon}$ %	NRSME %
2D case (Eq. (34))				
NURBS	Hammersley	78	4.92 (80%)	1.01 (84.2%)
	LHD	128	3.29 (87.4%)	0.63(89.3%)
Kriging	Hammersley	67	4.34 (63.8%)	0.84 (70%)
	LHD	60	4.77 (74.8%)	0.69 (73%)
RBF	Hammersley	150	16.8 (39.3)	2.52 (59.02%)
	LHD	150	16.7 (44.8%)	2.45 (47.5%)
4D case (Eq. (36))				
NURBS		168	7.638 (76.5%)	4.958 (80.9%)
Kriging	Hammersley	143	14.75 (26%)	9.68 (76.7%)
RBF		200	32.57 (38.7%)	22 (47.6%)



(a) Franke's function of Eq. (34).



(b) Griewank's function of Eq. (36).

Fig. 8. Comparison of different metamodels. Performance evolution from the one-shot strategy to the metamodels generated using adaptive sampling using the same number of samples. Red and green colours represents the worst and best results. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

For the 4D case, Fig. 8(b) shows that an adaptive sampling training improves the accuracy of all proposed metamodels compared to the one-shot training database. However, the NURBS-based metamodel achieves the best balance of accuracy and number of samples.

In conclusion, the NURBS-based metamodel, when coupled with an adaptive sampling strategy, consistently demonstrates higher accuracy and efficiency, particularly in multi-dimensional applications. In comparison, classical approaches, such as Kriging, provide satisfactory results in lower dimensions, but their precision and scalability are considerably reduced in higher dimensions. In addition, the RBF approach does not meet convergence criteria, thus highlighting the limitations of this method for the complex databases.

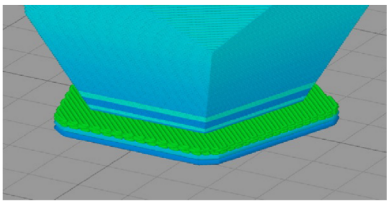


Fig. 9. Example of a 3D printed part built on a raft.

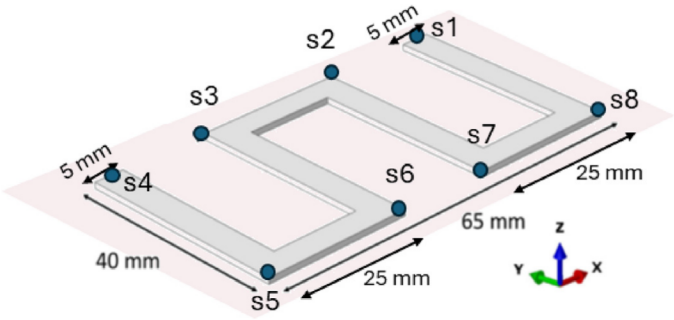


Fig. 10. Geometry of the planar spring in ABS material. The points S-i ($i = 1, \ldots, 8$) represent the nodes where the output is extracted at the end of the simulation. The part thickness is 0.95 mm.

5. Evaluation of the warpage of a 3D-printed planar spring

In this section, the effectiveness of TEAD adaptive sampling method for a NURBS-based metamodel is tested on a complex numerical problem involving $N = 3$ design variables. The problem aims to evaluate the warpage of the part after cooling as a function of the selected Fused Deposition Modelling (FDM) process parameters [52]. FDM is a cost-effective and versatile additive manufacturing technology widely used in aerospace, automotive, and medical sectors [53] due to its easy integration into the design processes [54], ability to print various materials, and low cost [55]. FDM creates 3D structures through the layer-by-layer melt-extrusion of plastic filament.

Several process parameters and printing strategies may significantly affect the quality of FDM-printed parts, necessitating an investigation of the design space to enhance accuracy, mechanical properties, and efficiency, in particular for high performing filaments such as PEEK, ABS and other composite reinforced filaments. The studied numerical problem focuses on the geometrical accuracy related to the thermomechanical phenomena involved in the printing process. Specifically, the combination of temperature gradients and printing strategies leads to part distortion, known as warpage, which arises from thermal shrinkage and stresses during manufacturing. Factors influencing warpage include part geometry, material properties, and deposition parameters, such as layer thickness, chamber temperature, deposition speed, strategy, and building plate temperature [52,55,56]. Experimentally, to prevent detachment and help with bed adhesion, support structures known as *rafts* and *skirts* are typically used. Specifically, a raft is a horizontal latticework of filaments on which the part is printed, as shown in Fig. 9.

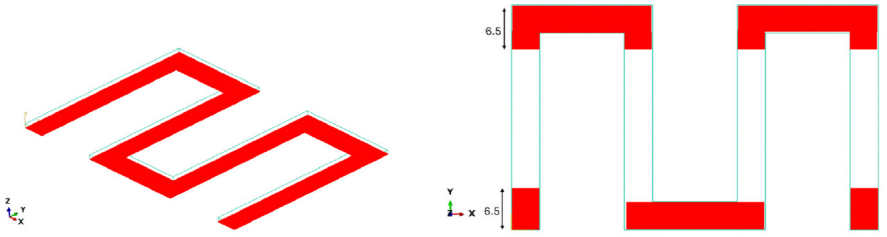
In this section, the thermomechanical simulation of FDM process is presented and applied to a simple 3D part manufactured by using an Acrylonitrile Butadiene Styrene (ABS) filament. The purpose is to simulate the post-cooling residual distortion of the part as a function of the selected process parameters. The commercial software ABAQUS is used to perform the analyses and simulate the printing process through the progressive element activation technique [4,57]. The FE thermomechanical model is inspired of that presented in [54]. Finally, the simulation results are compared and validated against data available in literature [54].

5.1. Thermomechanical model and validation

The thermomechanical FE model presented in this section simulates the manufacturing of a thin planar spring as shown in Fig. 10. Furthermore, no bed adhesion structures, such as raft or skirt, have been considered in the proposed model.

The printing strategy is extracted from the G-code generated from the 3D printer slicer RepetierHost[1] and converted, through Python scripts, into the definition of the printing strategy used to generate the FE model. The analysis involves a weak coupled thermomechanical simulation where nodal temperatures obtained in the thermal analysis are transformed in equivalent nodal forces for the mechanical analysis. The model outputs are the out-of-plane displacements at nodes s-i ($i = 1, \ldots, 8$) shown in Fig. 10. Furthermore, to simulate the bed adhesion, the bottom surface of the part is constrained throughout deposition and cooling, as shown in Fig. 11(a). After cooling, constraints are released allowing for part deformation, as shown in Fig. 11(b).

[1] www.repetier.com.



(a) Bottom surface of the planar spring fully constrained during printing and cooling phases.

(b) Sets of nodes deactivated after cooling to simulate the distortion of the part [54].

Fig. 11. Employed mechanical boundary conditions for the planar spring warpage simulation.

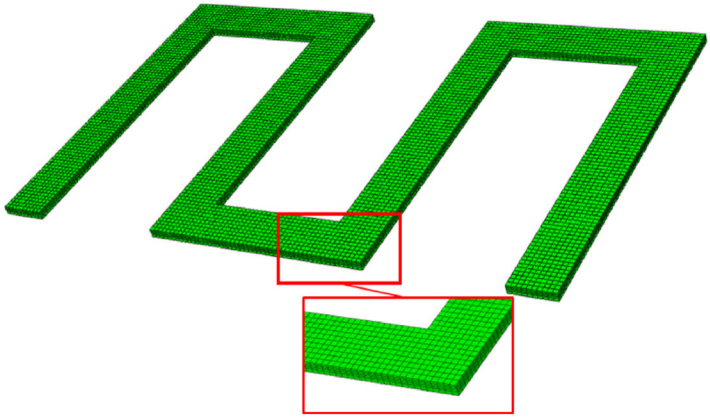


Fig. 12. Structured mesh of the planar spring model.

The thermomechanical simulation is performed with a transient non linear thermal analysis followed by a steady state structural analysis with plasticity. The equations governing the thermal and structural analyses are outlined in [4,54]. The ABS filament material properties are specified in [54] and they are temperature-dependent with elasto-plastic behaviour. During the printing process, the filament is extruded in a semi-molten state and deposited onto a substrate through a nozzle at a deposition speed DS. The substrate (bed plate) is maintained at a temperature BT, within a chamber set to a temperature TC. The deposition path follows a 100% infill strategy. The extruded filament is modelled with a rectangular section of dimension 0.5 mm × 0.2 mm, and material deposition is simulated using the progressive elements activation technique that correlates numerical elements with the nozzle movement. More details on this technique can be found in [4,57,58].

For the simulations, a structured mesh of thermal brick elements (DC3D8) is used for thermal analysis, while solid brick elements (C3D8) are employed for mechanical analysis [59]. An element size of 0.5 mm × 0.5 mm × 0.2 mm is chosen to accurately model the process, with each layer represented by a single element through the thickness. The time increment is set at $\Delta t = 0.5$ s as a trade-off between accuracy and computational time. Elements are deposited at a fixed temperature of $T_f = 240$ °C on a building plate at a temperature BT = 90 °C and in a chamber at a temperature TC = 45 °C, with a deposition speed DS = 40 mm/s. During deposition, radiation and convection boundary conditions are applied to the part free surfaces, with a convective coefficient $h = 15$ W/(m^2 K) and a radiation coefficient $\epsilon = 0.925$. The model mesh is represented in Fig. 12.

In order to check the effectiveness of the proposed model, a verification is carried out by comparing the vertical displacement at points s-*i* shown in Fig. 10 with the experimental results taken from Cattenone et al. [54]. As listed in Table 9 and shown in Fig. 13 there is a good agreement between experimental and numerical results, as far as the vertical displacement u_z is concerned. The error evaluation is performed using the metrics of Eq. (6). In fact, the average error between the numerical and experimental results is lower than 10% for almost all selected points and it is comparable with the results reported in [54]. Possible sources of such discrepancies are related to the detached nodes positions that differ from the experimental ones.

5.2. Design of experiments and generation of NURBS-based metamodel

The warpage phenomenon after cooling is approximated as a function of three deposition parameters: the deposition speed ζ_1 = DS, the building plate temperature ζ_2 = BT and the chamber temperature ζ_3 = TC. The bounds of the input variables are

Table 9
Numerical and experimental [54] u_z displacement on the planar spring S–i points.

	u_z [mm]	u_z^{exp} [mm]	$\bar{\varepsilon}$ [%]
S-1	0.532	0.544	2.2
S-2	0.313	0.253	23.7
S-3	0.311	0.316	1.6
S-4	0.308	0.296	4.05
S-5	0.311	0.324	4.01
S-6	0.313	0.282	10.9
S-7	0.344	0.74	8.7
S-8	0.687	0.351	5.04

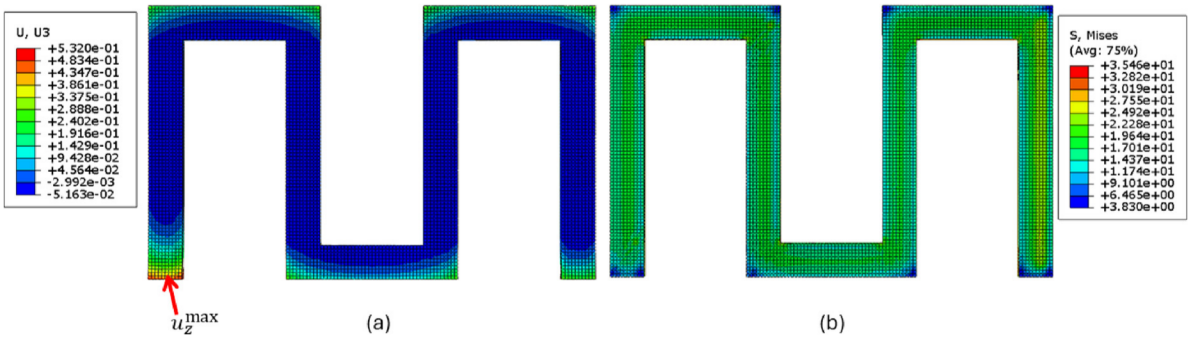


Fig. 13. (a) Z-axis displacement and (b) Von Mises stress after part detachment.

Table 10
Characteristics of the metamodel for the FDM problem case.

Variable	Range	s_k	τ	β_i	$\omega_{i_1 \dots i_N}$
ζ_1 = DS	[10, 60] mm/s				
ζ_2 = BT	[80, 120] °C	2	9/10	]0, 1[	[0.1, 10]
ζ_3 = TC	[27, 75] °C				

dictated by the 3D printer 3NTR A4v3 employed by Cattenone et al. [54] to manufacture their benchmark parts. The metamodel is then characterised by $N = 3$ inputs and $M = 1$ output corresponding to the maximum z-axis displacement after detachment.

The initial training database $\mathcal{Q}_{t0}$ is built with a fractional factorial method of weighted symmetric Leja knots distribution [60] and improved with the adaptive sampling design. The validation dataset $\mathcal{Q}_c$ is constructed from an optimised LHD and the number of points is obtained after convergence analysis. The initial training database has a size of $n_{\text{TP0}} = 69$ samples, and the samples distribution is shown in Fig. 14 as pairwise plot. The parameters tuning the behaviour of the metamodel are given in Table 10, together with the ranges of variation of the input variables and the optimisation variables, i.e., the coefficients β_i and weights $\omega_{i_1}, \dots, \omega_{i_N}$. The number of CPs is derived from the size of the database $\mathcal{Q}_t$ with $\tau = 9/10$ for this specific problem, and the value $s_k = 2$ is considered to limit numerical noise on the results approximation. Moreover, the number of CPs along each parametric direction n_k evolves with n_{TP} and they are evaluated at every iteration of the adaptive algorithm using Eq. (4), as shown in Fig. 1. Unlike the analytical test functions of Section 4, in this example the values of n_k are different in every input direction, and greater importance is assigned to the most influential variable on the result. Accordingly, a qualitative sensitivity analysis has been previously carried out and allowed us to conclude that BT is the variable that most influence the distortion, followed by the velocity DS. Lastly, the formulation of NURBS-based metamodel with smoothing term of Eq. (14) is used to limit the overfitting phenomenon. For this application, the metamodel is considered accurate enough when $\bar{\varepsilon} < 5\%$ and NRMSE$< 10\%$.

To assess the minimum size of the validation database, a convergence study is performed by gradually increasing the samples until the difference between two consecutive values of the considered metrics does not exceed a threshold value (set at 0.01 in this work). A metamodel built with $\mathcal{Q}_{t0}$ database is used to perform the convergence analysis for the validation set. From the results presented in the validation convergence plots of Fig. 15, it can be seen that at least 21 sample points are needed in the validation database to satisfy the selected threshold, but at least 41 points are needed to reach convergence on the results. Accordingly, in the following of this section the validation dataset $\mathcal{Q}_c$ is composed of 41 samples obtained with LHD.

Regarding the training of the metamodel, the TEAD adaptive sampling technique is used and coupled with ABAQUS software. Each FE simulation has a computational time of 2 h with a time increment of $\Delta t = 0.5$ s. All calculations are performed on a 11th

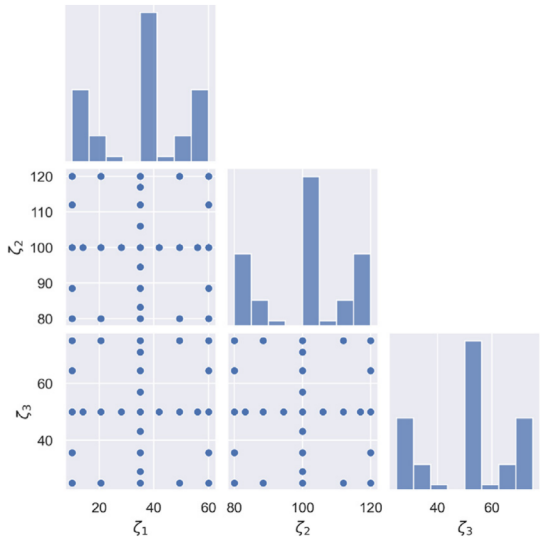


Fig. 14. Pairwise plot for the initial $\mathcal{Q}_{t0}$ using a Leja knots distribution for $N = 3$.

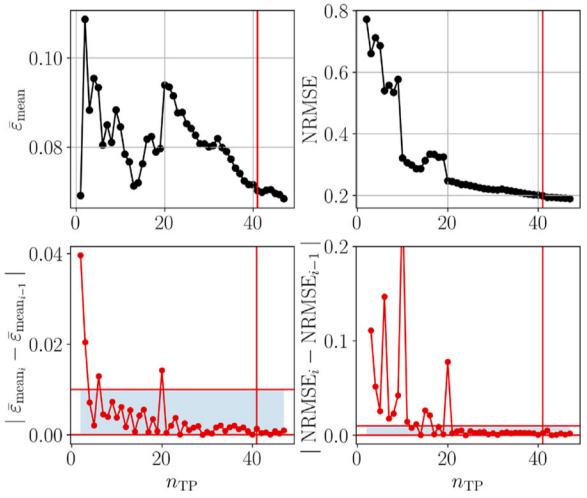


Fig. 15. Convergence plots representing accuracy metrics of Eqs. (6) and (8) as a function of number of samples in $\mathcal{Q}_c$.

Gen Intel(R) Core(TM) i7-11850H @ 2.50 GHz and with 8 CPUs. The stop criterion in terms of maximum number of samples within the training database is set at $n_{\text{TP}}^{\max} = 200$ to limit the computational costs.

Fig. 16 shows the 4D scatter plot of the maximum vertical displacement as a function of the input parameters.

The accuracy of the metamodel in terms of $\bar{\varepsilon}$ and NRMSE for the three consecutive optimisation phases at convergence is listed in Table 11. The considered accuracy criterion is achieved for a number of samples equal to $n_{\text{TP}} = 181$ and a number of $n_{\text{CP}} = 160$ CPs is considered with $n_1 = 4$, $n_2 = 7$, $n_3 = 3$. The performance improvement between the beginning and end of training is greater than 50 % with an initial NRMSE= 20% and a relative mean error $\bar{\varepsilon} = 8\%$. The local relative error at the beginning and end of training is shown in Fig. 17. It is evident that after the training the metamodel is able to better approximate the maximum displacement reducing the maximum relative error of 58%, as shown in Fig. 17(c).

Lastly, the effect of the smoothing term J_α of Eq. (14) in the problem formulation on the results is illustrated in Fig. 18. It is important to note that without the smoothing term, overfitting occurs. Specifically, the relative mean error on the validation set is over 35% at the beginning of training, compared to just 8% when the smoothing term is applied (see Fig. 17(c)).

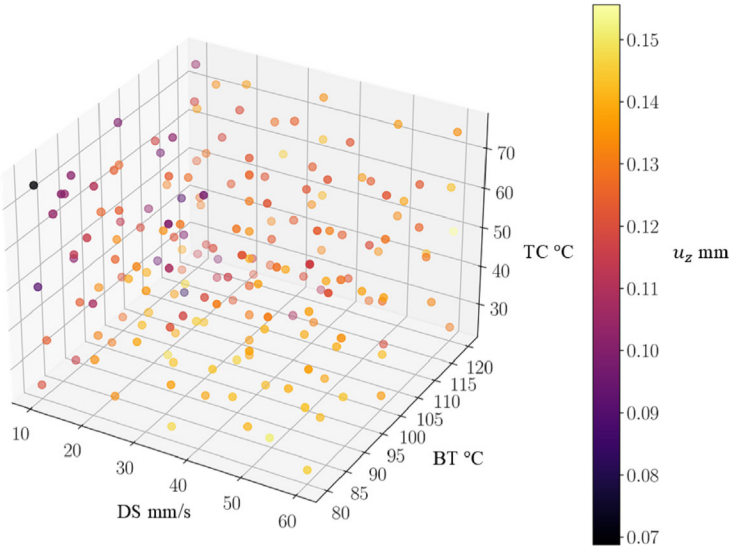


Fig. 16. Maximum vertical displacement u_z as a function of input parameter DS, BT, TC.

Table 11
Results in terms of $\bar{\varepsilon}$ and NRMSE for the FE thermomechanical problem when considering the smoothing term. The gain is evaluated comparing the value of the metric at the beginning and end of three-step optimisation process.

	$\xi_{0,\text{opt}}$	$\xi_{1,\text{opt}}$	$\xi_{2,\text{opt}}$	Gain [%]
$\bar{\varepsilon}$ [%]	3.327	3.25	3.19	60.8
NRMSE [%]	10.26	10.2	9.11	52.2

6. Conclusions

In this paper, the metamodeling strategy based on NURBS entities developed in [5] is improved by introducing adaptive sampling techniques that optimise the distribution of training points within the design space, and generalising the three-step optimisation approach (needed to generate the metamodel) to unstructured data. The adaptive technique is integrated into the three-step optimisation process and new samples are added in the database until convergence. The adaptive sampling strategies, particularly those based on k-fold CV and geometry-based criteria, have shown significant promise in enhancing the model accuracy, while minimising computational resources.

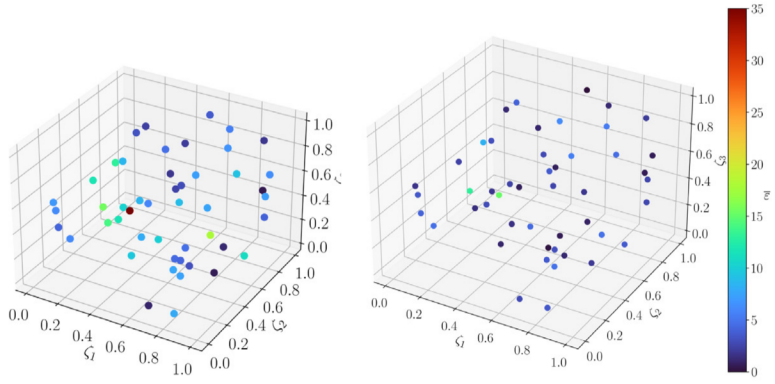
The adaptive sampling techniques find the new samples by maximising a global–local score. These classes of sampling strategies allow to maximise the utilisation of the training database while exploiting the analytical nature of NURBS entities. The effectiveness of the proposed metamodelling strategy is tested on several analytical functions and on a complex thermomechanical engineering problem, highlighting its precision, which is assessed on a validation database using relative mean error and normalised root mean square error accuracy metrics.

The results show that adaptive sampling strategies can significantly reduce the number of required training points while maintaining or improving the accuracy of the model, by adaptively adding the new sampling points to the most critical regions of the design space. This advance directly addresses one of the key challenges highlighted in previous works: the high computational costs and resource requirements for training large datasets.

Among the different techniques proposed, TEAD ensures the best results for each benchmark, as it uses the gradient information of the NURBS entities for the local search criterion in order to maximise the score and it is thus chosen as adaptive strategy in the thermomechanical test problem.

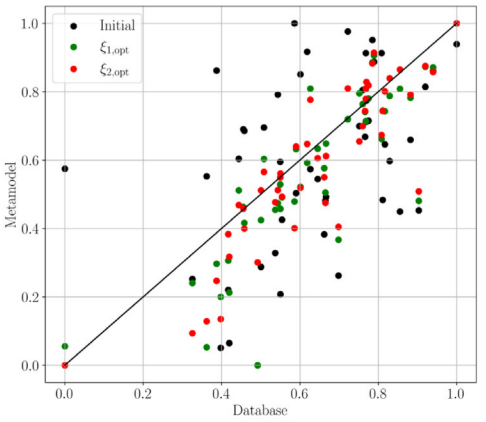
In conclusion, the main findings of this work can be resumed as follows:

- Thanks to the three-step optimisation formulation, the metamodel effectively captures strong non-linearities and geometric singularities by optimising CPs coordinates, inner knots and weights. Additionally, the general formulation enables dynamic adjustment of CPs coordinates based on weights values, improving the metamodel ability and fidelity to capture complex non-linear trends.
- The adaptive sampling search for a NURBS-based metamodel improves its ability to approximate complex non-linear behaviours by focusing computational efforts on regions of interest enhancing local accuracy and ensuring a more efficient global search across the design space. This allows achieving high accuracy with fewer samples compared to one-shot sampling



(a) Beginning of training with $n_{\mathrm{TP}} = 69$.

(b) End of training with $n_{\mathrm{TP}} = 181$.



(c) Fitting plot for the metamodel at the beginning of the training, and in the different steps of the three-step optimisation process.

Fig. 17. 4D scatter plot of the relative mean error at the beginning (Fig. 17(a)) and at the end of the training (Fig. 17(b)) by considering the smoothing term. The metric is evaluated on the validation database $\mathcal{Q}_c$. Fig. 17(c) shows the fitting plot of the metamodel.

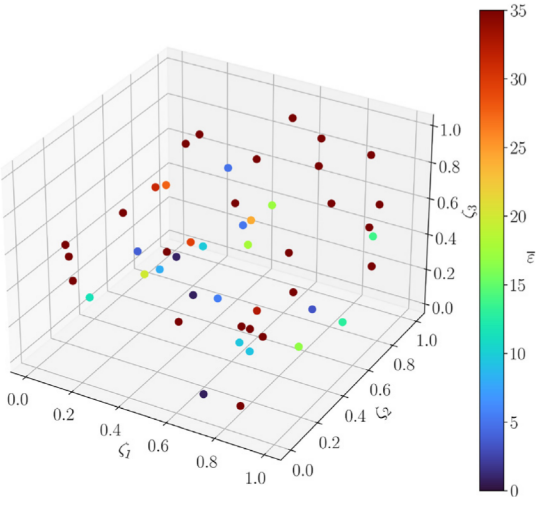


Fig. 18. Relative mean error for the problem formulation without smoothing term J_a.

methods. Nonetheless, if the problem formulation does not adequately balance local and global needs, the adaptive process may fail to improve the model effectively. Additionally, the new samples search must avoid sample clustering or poor global coverage that affect the overall robustness of the metamodel.

- TEAD effectively exploits the analytical nature of NURBS by using gradient-based criteria for local search. This allows for a more precise identification of regions in the design space that require additional sampling, leading to improved accuracy in the metamodel. Furthermore, this strategy consistently achieved better results in terms of error reduction and convergence criteria compared to other CV-based methods, and allows generating more accurate and reliable metamodels.
- The success of TEAD sampling technique heavily depends on the initial set of samples. If the initial distribution of samples is poor, the method may not perform well. To improve the search, for a NURBS-based metamodel, the initial dataset must include the bounds of the design domain.

In addition to the outlined features of the proposed metamodeling strategy, there are promising perspectives for further developing and exploration. Firstly, even if less samples are needed compared to one-shot strategies, the *curse of dimensionality* remains a significant issue, especially for high-dimensional design spaces. In fact, computational costs escalate rapidly with the increase in the number of CPs required to accurately represent the model. Moreover, due to the mathematical nature of NURBS entities, the *control net* has to be always structured, limiting the approximation of the design space for high dimensional problems. To address these issues, one can employs parallel computing techniques and problem dimension reduction strategies can be used to accelerate the calculations. Furthermore, hybrid sampling strategies and adaptive weighting mechanisms, which dynamically adjust the importance of exploration and exploitation during the sampling process, can help in achieving a more balanced samples distribution and avoiding samples clustering and poor global coverage.

CRediT authorship contribution statement

M. Zani: Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **E. Panettieri:** Writing – review & editing, Supervision. **M. Montemurro:** Writing – review & editing, Validation, Supervision, Software, Resources, Project administration, Methodology, Funding acquisition, Formal analysis, Conceptualization.

Funding

M. Zani is grateful to French National Research Agency for supporting this work through the research project KAM4AM (Fabrication Assistée Par La Connaissance Et L'intelligence Artificielle) ANR-20-CE10-0012-01.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix. Analytical expression of the gradient of the objective function

A.1. Optimisation of knot vectors components

Consider the general formulation of the problem with objective function expressed in Eq. (12). The gradient with respect to coefficient β_{i_k}, where $i_k = s_k + 1, \dots, m_k - s_k - 1$ and $k = 1, \dots, N$, reads:

$$\frac{\partial f_{\text{obj}}}{\partial \beta_{i_k}} = \sum_{\alpha=1}^{M} \left(2 \left(\mathbf{N}\mathbf{p}_\alpha - \mathbf{q}_\alpha \right)^{\mathrm{T}} \frac{\partial \mathbf{h}_\alpha}{\partial v_{i_k}} (v_{i_k-1} - v_{i_k+1}) + \lambda \frac{\partial \boldsymbol{J}_\alpha}{\partial \beta_{i_k}} \right), \tag{A.1}$$

where the term $\dfrac{\partial \mathbf{h}_\alpha}{\partial v_{i_k}}$ reads:

$$\frac{\partial \mathbf{h}_\alpha}{\partial v_{i_k}} = \frac{\partial \mathbf{N}}{\partial v_{i_k}} \mathbf{p}_\alpha + \mathbf{N} \left[\left(\mathbf{N}^{\mathrm{T}}\mathbf{N} + \lambda \mathbf{E} \right)^{-1} \left(\frac{\partial \mathbf{N}^{\mathrm{T}}}{\partial v_{i_k}} \mathbf{q}_\alpha - \frac{\partial}{\partial v_{i_k}} \left(\mathbf{N}^{\mathrm{T}}\mathbf{N} + \lambda \mathbf{E} \right) \mathbf{p}_\alpha \right) \right], \tag{A.2}$$

whilst $\dfrac{\partial \boldsymbol{J}_\alpha}{\partial \beta_{i_k}}$ can be obtained from Eq. (14) as follows:

$$\frac{\partial \boldsymbol{J}_\alpha}{\partial \beta_{i_k}} = \frac{\partial \boldsymbol{J}_\alpha}{\partial v_{i_k}} \frac{\partial v_{i_k}}{\partial \beta_{i_k}} = \left(2 \mathbf{p}_\alpha \mathbf{E} \frac{\partial \mathbf{p}_\alpha}{\partial v_{i_k}} + \mathbf{p}_\alpha^{\mathrm{T}} \frac{\partial \mathbf{E}}{\partial v_{i_k}} \mathbf{p}_\alpha \right) (v_{i_k-1} - v_{i_k+1}). \tag{A.3}$$

The development of the gradient of the objective function when considering knots as design variables for both the general and dimension separation formulations is detailed in [5].

The term $\frac{\partial \mathbf{E}}{\partial v_{i_k}}$ has a more complicate expression as it contains the first and second derivatives of Bernstein polynomial as a function of ζ_k, with the partial derivative with respect to a function of v_{i_k} calculated using the expression in [61].

Specifically, for a problem of dimension N using B-spline entities, the matrix $\mathbf{E} \in \mathbb{R}^{n_{\mathrm{CP}} \times n_{\mathrm{CP}}}$ can be expressed, according to Eq. (19), as a function of Bernstein polynomials only and their derivatives. The expression of its derivative with the respect of knot vectors components becomes:

$$\frac{\partial \mathbf{E}}{\partial v_{i_l}} = \frac{\partial}{\partial v_{i_l}} \sum_{k=1}^{N} \left(\mathbf{A}_k + 2 \sum_{q>k}^{N} \mathbf{B}_{k,q} \right) = \sum_{k=1}^{N} \left(\frac{\partial \mathbf{A}_k}{\partial v_{i_l}} + 2 \sum_{q>k}^{N} \frac{\partial \mathbf{B}_{k,q}}{\partial v_{i_l}} \right). \tag{A.4}$$

The index $i_l = s_l + 1, \dots, m_l - s_l - 1$ and $l = 1, \dots, N$, is here used instead of i_k to avoid confusion between the terms of $\mathbf{E}$ and the components of the knot vectors to optimise. The partial derivatives of terms $\mathbf{A}_k, \mathbf{B}_{k,q}$ with respect to v_{i_l} exists only along direction l and they are defined as follows. For derivative direction l that coincide with k (or q), the partial derivatives of the components of matrix $\mathbf{A}$ read:

$$\begin{aligned}
&\frac{\partial A_{k,\gamma_1,\gamma_2}}{\partial v_{i_l}} \\
&= \left(\prod_{g=1}^{k-1} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \int_0^1 \frac{\partial}{\partial v_{i_l}} \left(\frac{\partial^2 N_{i_k,s_k}}{\partial \zeta_k^2} \right) d\zeta_k \prod_{g=k+1}^{N} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \right) \\
&\times \left(\prod_{r=1}^{k-1} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \int_0^1 \frac{\partial^2 N_{j_k,s_k}}{\partial \zeta_k^2} d\zeta_k \prod_{r=k+1}^{N} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \right) \\
&+ \left(\prod_{g=1}^{k-1} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \int_0^1 \frac{\partial^2 N_{i_k,s_k}}{\partial \zeta_k^2} d\zeta_k \prod_{g=k+1}^{N} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \right) \\
&\times \left(\prod_{r=1}^{k-1} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \int_0^1 \frac{\partial}{\partial v_{i_l}} \left(\frac{\partial^2 N_{j_k,s_k}}{\partial \zeta_k^2} \right) d\zeta_k \prod_{r=k+1}^{N} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \right),
\end{aligned} \tag{A.5}$$

and

$$\begin{aligned}
&\frac{\partial B_{k,q,\gamma_1,\gamma_2}}{\partial v_{i_l}} \\
&= \left(\prod_{g=1}^{k-1} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \int_0^1 \frac{\partial}{\partial v_{i_l}} \left(\frac{\partial N_{i_k,s_k}}{\partial \zeta_k} \right) d\zeta_k \right. \\
&\left. \prod_{g=k+1}^{q-1} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \int_0^1 \frac{\partial N_{i_q,s_q}}{\partial \zeta_q} d\zeta_q \prod_{g=q+1}^{N} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \right) \\
&\times \left(\prod_{r=1}^{k-1} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \int_0^1 \frac{\partial N_{j_k,s_k}}{\partial \zeta_k} d\zeta_k \right. \\
&\left. \prod_{r=k+1}^{q-1} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \int_0^1 \frac{\partial N_{j_q,s_q}}{\partial \zeta_q} d\zeta_q \prod_{r=q+1}^{N} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \right) \\
&+ \left(\prod_{g=1}^{k-1} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \int_0^1 \frac{\partial N_{i_k,s_k}}{\partial \zeta_k} d\zeta_k \right. \\
&\left. \prod_{g=k+1}^{q-1} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \int_0^1 \frac{\partial N_{i_q,s_q}}{\partial \zeta_q} d\zeta_q \prod_{g=q+1}^{N} \int_0^1 N_{i_g,s_g}(\zeta_g) d\zeta_g \right) \\
&\times \left(\prod_{r=1}^{k-1} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \int_0^1 \frac{\partial}{\partial v_{i_l}} \left(\frac{\partial N_{j_k,s_k}}{\partial \zeta_k} \right) d\zeta_k \right. \\
&\left. \prod_{r=k+1}^{q-1} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \int_0^1 \frac{\partial N_{j_q,s_q}}{\partial \zeta_q} d\zeta_q \prod_{r=q+1}^{N} \int_0^1 N_{j_r,s_r}(\zeta_r) d\zeta_r \right).
\end{aligned} \tag{A.6}$$

In the case of l equal to q, expression of Eq. (A.6) modifies deriving the terms $\int_0^1 \frac{\partial}{\partial v_{i_l}} \left(\frac{\partial N_{i_q,s_q}}{\partial \zeta_q} \right) d\zeta_q$ instead of terms in k direction.

Conversely, if $l \neq k$,

$$
\begin{aligned}
&\frac{\partial A_{k,\gamma_1,\gamma_2}}{\partial v_{i_l}} \\
&= \left(\prod_{g=1}^{k-1}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g \int_0^1 \frac{\partial^2 N_{i_k,s_k}}{\partial \zeta_k^2} d\zeta_k \prod_{g=k+1}^{l-1}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g\right. \\
&\left.\int_0^1 \frac{\partial}{\partial v_{i_l}}(N_{i_l,s_l}(\zeta_l))d\zeta_l \prod_{g=l+1}^{N}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g\right) \\
&\times \left(\prod_{r=1}^{k-1}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r \int_0^1 \frac{\partial^2 N_{j_k,s_k}}{\partial \zeta_k^2} d\zeta_k \prod_{r=k+1}^{N}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r\right) \\
&+ \left(\prod_{g=1}^{k-1}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g \int_0^1 \frac{\partial^2 N_{i_k,s_k}}{\partial \zeta_k^2} d\zeta_k \prod_{g=k+1}^{N}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g\right) \\
&\times \left(\prod_{r=1}^{k-1}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r \int_0^1 \frac{\partial^2 N_{j_k,s_k}}{\partial \zeta_k^2} d\zeta_k\right. \\
&\left.\prod_{r=k+1}^{l-1}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r \int_0^1 \frac{\partial}{\partial v_{i_l}}(N_{j_l,s_l}(\zeta_l))d\zeta_l \prod_{r=l+1}^{N}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r\right),
\end{aligned}
\tag{A.7}
$$

whilst the mixed terms components read

$$
\begin{aligned}
&\frac{\partial B_{k,q,\gamma_1,\gamma_2}}{\partial v_{i_l}} \\
&= \left(\prod_{g=1}^{k-1}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g \int_0^1 \frac{\partial N_{i_k,s_k}}{\partial \zeta_k} d\zeta_k \prod_{g=k+1}^{l-1}\int_0^1 (N_{i_g,s_g}(\zeta_g))d\zeta_g\right. \\
&\int_0^1 \frac{\partial}{\partial v_{i_l}}(N_{i_l,s_l}(\zeta_l))d\zeta_l \prod_{g=l+1}^{q-1}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g \int_0^1 \frac{\partial N_{i_q,s_q}}{\partial \zeta_q} d\zeta_q \\
&\left.\prod_{g=q+1}^{N}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g\right) \times \left(\prod_{r=1}^{k-1}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r \int_0^1 \frac{\partial N_{j_k,s_k}}{\partial \zeta_k} d\zeta_k\right. \\
&\left.\prod_{r=k+1}^{q-1}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r \int_0^1 \frac{\partial N_{j_q,s_q}}{\partial \zeta_q} d\zeta_q \prod_{r=q+1}^{N}\int_0^1 N_{j_g,s_g}(\zeta_g)d\zeta_g\right) \\
&+ \left(\prod_{g=1}^{k-1}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g \int_0^1 \frac{\partial N_{i_k,s_k}}{\partial \zeta_k} d\zeta_k \prod_{g=k+1}^{q-1}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g\right. \\
&\left.\int_0^1 \frac{\partial N_{i_q,s_q}}{\partial \zeta_q} d\zeta_q \prod_{g=q+1}^{N}\int_0^1 N_{i_g,s_g}(\zeta_g)d\zeta_g\right) \times \left(\prod_{r=1}^{k-1}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r\right. \\
&\int_0^1 \frac{\partial N_{j_k,s_k}}{\partial \zeta_k} d\zeta_k \prod_{r=k+1}^{l-1}\int_0^1 (N_{j_r,s_r}(\zeta_r))d\zeta_r \int_0^1 \frac{\partial}{\partial v_{i_l}}(N_{j_l,s_l}(\zeta_l))d\zeta_l \\
&\left.\prod_{r=l+1}^{q-1}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r \int_0^1 \frac{\partial N_{j_q,s_q}}{\partial \zeta_q} d\zeta_q \prod_{r=q+1}^{N}\int_0^1 N_{j_r,s_r}(\zeta_r)d\zeta_r\right).
\end{aligned}
\tag{A.8}
$$

This last equation concludes the analytical development of $\dfrac{\partial \mathbf{E}}{\partial v_{i_l}}$.

Remark A.1. From a numerical point of view, the above formulæ are analytically implemented and the product of integrals is done using Kronecker product.

A.2. Optimisation of weights

Consider now the gradient of the cost function with respect to the weights. This optimisation process corresponds to the last optimisation phase in Fig. 1. This optimisation differs from the previous ones mainly because it is based on a NURBS hyper-surface, and it involves modifying the importance of certain CPs by manipulating the impact of rational basis function polynomials of the associated local support. The variables to be optimised are collected in the vector $\boldsymbol{\xi}_2^T = \{\omega_{0\ldots0}, \ldots, \omega_{i_1\ldots i_N}, \ldots, \omega_{n_1\ldots n_N}\}$, $\boldsymbol{\xi}_2 \in \mathbb{R}^{n_{\mathrm{CP}}}$.

For a NURBS entity, when using the general problem formulation, the optimal values of the CPs coordinates resulting from the first optimisation step are expressed as

$$\mathbf{p}_\alpha = \left(\mathbf{R}^{\mathrm{T}}\mathbf{R} + \lambda\mathbf{E}\right)^{-1}\mathbf{R}^{\mathrm{T}}\mathbf{q}_\alpha, \tag{A.9}$$

with $\mathbf{R}$ the matrix of rational basis functions with components of Eq. (13), and $\mathbf{E}$ the matrix of second derivative of the NURBS basis functions with respect to the parametric variables. Compared to previous works, the CPs coordinates are updated for each step of the optimisation process depending on the current value of the weights according to Eq. (A.9). The analytical expression of the gradient with respect to the weights and its analytical development can be found in [26] for the case of surface fitting problems.

Proposition A.1. *Consider the general formulation of the problem with objective function of Eq. (9) without smoothing term ($\lambda = 0$). Employing the linear index of Eq. (11), the gradient with respect to weights $\omega_{i_1\dots i_N}$ reads:*

$$\frac{\partial f_{\mathrm{obj}}}{\partial \omega_{i_1\dots i_N}} = \sum_{\alpha=1}^{M} 2\left(\mathbf{R}\mathbf{p}_\alpha - \mathbf{q}_\alpha\right)\frac{\partial \mathbf{h}_\alpha}{\partial \omega_{i_1\dots i_N}}, \tag{A.10}$$

where the term $\dfrac{\partial \mathbf{h}_\alpha}{\partial \omega_{i_1\dots i_N}}$ reads:

$$\frac{\partial \mathbf{h}_\alpha}{\partial \omega_{i_1\dots i_N}} = \frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha + \mathbf{R}\frac{\partial \mathbf{p}_\alpha}{\partial \omega_{i_1\dots i_N}} = \left[\mathbf{I} - \mathbf{R}\left(\mathbf{R}^{\mathrm{T}}\mathbf{R}\right)^{-1}\mathbf{R}^{\mathrm{T}}\right]\frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha. \tag{A.11}$$

The proof of Proposition A.1 is detailed here below.

Proof. The gradient of the cost function with respect to $\omega_{i_1\dots i_N}$ without smoothing term is defined in Eq. (A.10), and the term $\dfrac{\partial \mathbf{h}_\alpha}{\partial \omega_{i_1\dots i_N}}$ reads:

$$\frac{\partial \mathbf{h}_\alpha}{\partial \omega_{i_1\dots i_N}} = \frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha + \mathbf{R}\frac{\partial \mathbf{p}_\alpha}{\partial \omega_{i_1\dots i_N}}. \tag{A.12}$$

This term can be further simplified starting from definition of $\mathbf{p}_\alpha$ of Eq. (22). Particularly, by differentiating Eq. (A.9) one obtains:

$$\frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha + \mathbf{R}\frac{\partial \mathbf{p}_\alpha}{\partial \omega_{i_1\dots i_N}} = \mathbf{0} \iff \frac{\partial \mathbf{p}_\alpha}{\partial \omega_{i_1\dots i_N}} = -\left(\mathbf{R}^{\mathrm{T}}\mathbf{R}\right)^{-1}\mathbf{R}^{\mathrm{T}}\frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha. \tag{A.13}$$

By injecting Eq. (A.13) in Eq. (A.12) one obtains:

$$\frac{\partial \mathbf{h}_\alpha}{\partial \omega_{i_1\dots i_N}} = \frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha - \mathbf{R}\left(\mathbf{R}^{\mathrm{T}}\mathbf{R}\right)^{-1}\mathbf{R}^{\mathrm{T}}\frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha, \tag{A.14}$$

that can be simplified as:

$$\frac{\partial \mathbf{h}_\alpha}{\partial \omega_{i_1\dots i_N}} = \left[\mathbf{I} - \mathbf{R}\left(\mathbf{R}^{\mathrm{T}}\mathbf{R}\right)^{-1}\mathbf{R}^{\mathrm{T}}\right]\frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha, \tag{A.15}$$

where $\mathbf{I} \in \mathbb{R}^{n_{\mathrm{TP}} \times n_{\mathrm{TP}}}$. This last passage concludes the proof. □

The term $\dfrac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha$ is defined as follows:

$$\frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha = p_{\alpha,i_1\dots i_N}\mathbf{a} - \mathbf{b}. \tag{A.16}$$

In the above equation, vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{n_{\mathrm{TP}}}$ are defined as follows:

$$a_{\kappa_{\mathrm{TP}}} := \frac{\prod_{k=1}^{N} N_{i_k,s_k}(\zeta_{k,\kappa_{\mathrm{TP}}})}{W_{\kappa_{\mathrm{TP}}}}, \quad b_{\kappa_{\mathrm{TP}}} := a_{\kappa_{\mathrm{TP}}} h_{\alpha,\kappa_{\mathrm{TP}}}, \quad \kappa_{\mathrm{TP}} = 1, \dots, n_{\mathrm{TP}}, \tag{A.17}$$

where $h_{\alpha,\kappa_{\mathrm{TP}}}$ is the generic component of the vector $\mathbf{h}_\alpha = \mathbf{R}\mathbf{p}_\alpha$ collecting the αth coordinate of the hyper-surface evaluated at the generic TP, while the function $W_{\kappa_{\mathrm{TP}}}$ is defined as:

$$W_{\kappa_{\mathrm{TP}}} := \sum_{j_1=0}^{n_1} \cdots \sum_{j_N=0}^{n_N} \omega_{j_1\dots j_N} \prod_{k=1}^{N} N_{j_k,s_k}(\zeta_{k,\kappa_{\mathrm{TP}}}). \tag{A.18}$$

The linear index κ_{TP} is defined in Eq. (11).

The formulation based on dimension separation can be applied to the term $\dfrac{\partial \mathbf{h}_\alpha}{\partial \omega_{i_1\dots i_N}}$ with the hypothesis of not revaluating the CPs coordinates at each iteration step of the optimisation process:

$$\frac{\partial \mathbf{h}_\alpha}{\partial \omega_{i_1\dots i_N}} = \frac{\partial \mathbf{R}}{\partial \omega_{i_1\dots i_N}}\mathbf{p}_\alpha = p_{\alpha,i_1\dots i_N}\mathbf{a} - \mathbf{b}, \tag{A.19}$$

with terms $\mathbf{a}$, $\mathbf{b}$ components defined as in Eq. (A.17).

Data availability

The raw/processed data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

References

[1] Z. Yang, D. Eddy, S. Krishnamurty, I. Grosse, P. Denno, F. Lopez, Investigating predictive metamodeling for additive manufacturing, 2016.
[2] G. Pan, P. Ye, P. Wang, Z. Yang, A sequential optimization sampling method for metamodels with radial basis functions, Sci. World J. 2014 (2014) 192862.
[3] P. Lagouanelle, F. Freschi, L. Pichon, Adaptive sampling for fast and accurate metamodel-based sensitivity analysis of complex electromagnetic problems, IEEE Trans. Electromagn. Compat. PP (2023) 1–9.
[4] M. Zani, E. Panettieri, M. Montemurro, A metamodel of the wire arc additive manufacturing process based on basis spline entities, Eng. Comput. (2024).
[5] B. Vuillod, M. Zani, L. Hallo, M. Montemurro, Handling noise and overfitting in surrogate models based on non-uniform rational basis spline entities, Comput. Methods Appl. Mech. Engrg. 425 (2024) 116913.
[6] G. Tapia, S. Khairallah, M. Matthews, W.E. King, A. Elwany, Gaussian process-based surrogate modeling framework for process planning in laser powder-bed fusion additive manufacturing of 316L stainless steel, Int. J. Adv. Manuf. Technol. 94 (2018) 3591–3603.
[7] B.M. de Gooijer, J. Havinga, H.J. Geijselaers, A.H. van den Boogaard, Evaluation of POD based surrogate models of fields resulting from nonlinear FEM simulations, Adv. Model. Simul. Eng. Sci. 8 (1) (2021) 25.
[8] F. Chinesta, A. Ammar, E. Cueto, Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models, Arch. Comput. Methods Eng. 17 (4) (2010) 327–350.
[9] J. Sacks, W.J. Welch, T.J. Mitchell, H.P. Wynn, Design and analysis of computer experiments, Statist. Sci. 4 (4) (1989) 409–423.
[10] A.I. Forrester, A.J. Keane, Recent advances in surrogate-based optimization, Prog. Aerosp. Sci. 45 (1–3) (2009) 50–79.
[11] L. Wang, X. Chen, S. Kang, X. Deng, R. Jin, Meta-modeling of high-fidelity FEA simulation for efficient product and process design in additive manufacturing, Addit. Manuf. 35 (2020) 101211.
[12] J. Kudela, R. Matousek, Recent advances and applications of surrogate models for finite element method computations: a review, Soft Comput. 26 (2022).
[13] J.C. García-Merino, C. Calvo-Jurado, E. Martínez-Pañeda, E. García-Macías, Multielement polynomial chaos Kriging-based metamodelling for Bayesian inference of non-smooth systems, Appl. Math. Model. 116 (2023) 510–531.
[14] P. Saves, R. Lafage, N. Bartoli, Y. Diouane, J. Bussemaker, T. Lefebvre, J.T. Hwang, J. Morlier, J.R.R.A. Martins, SMT 2.0: A surrogate modeling toolbox with a focus on hierarchical and mixed variables Gaussian processes, Adv. Eng. Sofw. 188 (2024).
[15] G. Chen, K. Zhang, X. Xue, L. Zhang, C. Yao, J. Wang, J. Yao, A radial basis function surrogate model assisted evolutionary algorithm for high-dimensional expensive optimization problems, Appl. Soft Comput. 116 (2022).
[16] F. Li, Z. Shang, Y. Liu, H. Shen, Y. Jin, Inverse distance weighting and radial basis function based surrogate model for high-dimensional expensive multi-objective optimization, Appl. Soft Comput. 152 (2024).
[17] J.T. Hwang, J.R. Martins, A fast-prediction surrogate model for large datasets, Aerosp. Sci. Technol. 75 (2018) 74–87.
[18] A. Parnianifard, A. Azfanizam, M. Ariffin, M. Ismail, Comparative study of metamodeling and sampling design for expensive and semi-expensive simulation models under uncertainty, Simul. 96 (1) (2020) 89–110.
[19] H. Liu, Y. Ong, J. Cai, A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design, Struct. Multidiscip. Optim. 57 (2018).
[20] T. Mackman, C. Allen, M. Ghoreshi, K. Badcock, Comparison of adaptive sampling methods for generation of surrogate aerodynamic models, AIAA J. 51 (2013) 797–808.
[21] J. Fuhg, A. Fau, U. Nackenhorst, State-of-the-art and comparative review of adaptive sampling methods for Kriging, Arch. Comput. Methods Eng. 28 (2020).
[22] H. Xu, H. Li, G. Xiang, X. Zhang, An improved adaptive surrogate model and application in thermal management system design, Mater. Des. 208 (2021) 109883.
[23] B. Pickett, C.J. Turner, A review and evaluation of existing adaptive sampling criteria and methods for the creation of NURBs-based metamodels, International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, vol. 2, 2011, pp. 609–618, http://dx.doi.org/10.1115/DETC2011-47288.
[24] C. Turner, R. Crawford, M. Campbell, Multidimensional sequential sampling for NURBS-based metamodel develpment, Eng. Comput. (Lond.) 23 (2007) 155–174.
[25] S. Mo, D. lu, X. Shi, G. Zhang, M. Ye, J. Wu, J. Wu, A Taylor expansion-based adaptive design strategy for global surrogate modeling with applications in groundwater modeling, Water Resour. Res. 53 (2017).
[26] G. Bertolino, M. Montemurro, N. Perry, F. Pourroy, An efficient hybrid optimization strategy for surface reconstruction, in: Computer Graphics Forum, Vol. 40, Wiley Online Library, 2021, pp. 215–241.
[27] M.J. Kazemzadeh-Parsi, A. Ammar, F. Chinesta, Domain decomposition involving subdomain separable space representations for solving parametric problems in complex geometries, Adv. Model. Simul. Eng. Sci. 9 (1) (2022) 2.
[28] X. Ying, An overview of overfitting and its solutions, in: Journal of Physics: Conference Series, Vol. 1168, IOP Publishing, 2019.
[29] H. Nguyen, I. Couckuyt, L. Knockaert, T. Dhaene, D. Gorissen, Y. Saeys, An alternative approach to avoid overfitting for surrogate models, Proc. - Winter Simul. Conf. (2011) 2760–2771.
[30] Y. Zhang, W. Yao, S. Ye, X. Chen, A regularization method for constructing trend function in Kriging model, Struct. Multidiscip. Optim. 59 (2019) 1221–1239.
[31] X. Xie, X. Lu, Theta-regularized Kriging: Modeling and algorithms, Appl. Math. Model. 136 (2024) 115627.
[32] D. Lenz, R. Yeh, V. Mahadevan, I. Grindeanu, T. Peterka, Customizable adaptive regularization techniques for B-spline modeling, J. Comput. Sci. 71 (2023) 102037.
[33] C. Valencia, M. Yuan, Radial basis function regularization for linear inverse problems with random noise, J. Multivariate Anal. 116 (2013) 92–108.
[34] P. Jiang, L. Shu, Q. Zhou, H. Zhou, X. Shao, J. Xu, A novel sequential exploration-exploitation sampling strategy for global metamodeling, IFAC- Pap. 48 (28) (2015) 532–537, 17th IFAC Symposium on System Identification SYSID 2015.
[35] G. Li, V. Aute, S. Azarm, An accumulative error based adaptive design of experiments for offline metamodeling, Struct. Multidiscip. Optim. 40 (2010) 137–155.
[36] B. Kim, Y. Lee, D.-H. Choi, Construction of the radial basis function based on a sequential sampling approach using cross-validation, J. Mech. Sci. Technol. - J. MECH. SCI. TECHNOL. 23 (2009) 3357–3365.
[37] Y. Audoux, M. Montemurro, J. Pailhès, Non-uniform rational basis spline hyper-surfaces for metamodelling, Comput. Methods Appl. Mech. Engrg. 364 (2020) 112918.

[38] M.S. Floater, Meshless parameterization and B-spline surface approximation, in: R. Cipolla, M. R. (Eds.), The Mathematics of Surfaces IX: Proceedings of the Ninth IMA Conference on the Mathematics of Surfaces, Springer, 2000, pp. 1–18.
[39] L. Piegl, W. Tiller, The NURBS book, Springer Science and Business Media, 1996.
[40] T. Rodriguez, M. Montemurro, P. Le Texier, J. Pailhès, Structural displacement requirement in a topology optimization algorithm based on isogeometric entities, J. Optim. Theory Appl. 184 (1) (2020) 250–276.
[41] M. Montemurro, A. Mas, S. eddine Zerrouq, Topology and anisotropy optimisation of continua using non-uniform rational basis spline entities, Comput. Methods Appl. Mech. Engrg. 420 (2024) 116714.
[42] G. Costa, M. Montemurro, J. Pailhès, A general hybrid optimization strategy for curve fitting in the non-uniform rational basis spline framework, J. Optim. Theory Appl. 176 (2018) 225–251.
[43] M.I. Izzi, A. Catapano, M. Montemurro, Strength and mass optimisation of variable-stiffness composites in the polar parameters space, Struct. Multidiscip. Optim. 64 (4) (2021) 2045–2073.
[44] E.T. Lee, Choosing nodes in parametric curve interpolation, Comput. Aided Des. 21 (6) (1989) 363–370.
[45] C.J. Turner, R.H. Crawford, N-dimensional nonuniform rational B-splines for metamodeling, J. Comput. Inf. Sci. Eng. 9 (3) (2009).
[46] G.J.A. Loeven, J.A.S. Witteveen, H. Bijl, A probabilistic radial basis function approach for uncertainty quantification, 2007.
[47] K. Svanberg, MMA and GCMMA-two methods for nonlinear optimization, 2007, pp. 1–15, 1.
[48] M. Montemurro, On the structural stiffness maximisation of anisotropic continua under inhomogeneous Neumann–Dirichlet boundary conditions, Compos. Struct. 287 (2022) 115289.
[49] I. Kalogridis, Robust thin-plate splines for multivariate spatial smoothing, Econ. Stat. (2023).
[50] J. Westermann, A. Zea, U.D. Hanebeck, Adaptive sampling for global meta modeling using a Gaussian process variance measure, in: 2021 European Control Conference, ECC, IEEE, 2021, pp. 573–579.
[51] D. Bremner, K. Fukuda, A. Marzetta, Primal-dual methods for vertex and facet enumeration (preliminary version), in: Proceedings of the Thirteenth Annual Symposium on Computational Geometry, 1997, pp. 49–56.
[52] A. Armillotta, M. Bellotti, M. Cavallaro, Warpage of FDM parts: Experimental tests and analytic model, Robot. Comput.-Integr. Manuf. 50 (2018) 140–152.
[53] A. Cano-Vicent, M.M. Tambuwala, S.S. Hassan, D. Barh, A.A. Aljabali, M. Birkett, A. Arjunan, Á. Serrano-Aroca, Fused deposition modelling: Current status, methodology, applications and future prospects, Addit. Manuf. 47 (2021) 102378.
[54] A. Cattenone, S. Morganti, G. Alaimo, F. Auricchio, Finite element analysis of additive manufacturing based on fused deposition modeling: Distortions prediction and comparison with experimental data, J. Manuf. Sci. Eng. 141 (1) (2019) 011010.
[55] G. Gao, F. Xu, J. Xu, G. Tang, Z. Liu, A survey of the influence of process parameters on mechanical properties of fused deposition modeling parts, Micromach. 13 (4) (2022) 553.
[56] R. Gao, J. Xie, J. Yang, C. Zhuo, J. Fu, P. Zhao, Research on the fused deposition modeling of polyether ether ketone, Polymers 13 (14) (2021) 2344.
[57] B. Edison A., M. Andres, A directed energy deposition additive manufacturing process simulated with ABAQUS AM modeler, Int. J. Robot. Eng. 6 (1) (2021).
[58] X. Song, S. Feih, W. Zhai, C.N. Sun, F. Li, R. Maiti, J. Wei, Y. Yang, V. Oancea, L. Romano Brandt, A.M. Korsunsky, Advances in additive manufacturing process simulation: Residual stresses and distortion predictions in complex metallic components, Mater. Des. 193 (2020).
[59] M. Smith, ABAQUS/Standard User's Manual, Version 6.20, Dassault Systèmes Simulia Corp, United States, 2020.
[60] C. Piazzola, L. Tamellini, Algorithm X: The sparse grids matlab kit-a matlab implementation of sparse grids for high-dimensional function approximation and uncertainty quantification, ACM Trans. Math. Software (2023).
[61] L.A. Piegl, W. Tiller, Computing the derivative of NURBS with respect to a knot, Comput. Aided Geom. Design 15 (9) (1998) 925–934.