Non-uniform rational basis spline hyper-surfaces for metamodelling

Yohann Audoux, Marco Montemurro, Jérôme Pailhès

Non-Uniform Rational Basis Spline hyper-surfaces for metamodelling

Yohann Audoux[a], Marco Montemurro[a,*], Jérôme Pailhès[a]

[a]*Arts et Métiers ParisTech, I2M CNRS UMR 5295, F-33400 Talence, France*

Abstract

This study presents an original metamodelling technique based on Non-Uniform Rational Basis Spline (NURBS) hyper-surfaces. The proposed approach is able to fit general non-convex sets of target points (TPs) by extending the NURBS formalism to the N-dimensional (N-D) case. The shape of such a hyper-surface is tuned by several parameters: the number of control points (CPs), their coordinates and the related weights, the degrees of the blending functions and the knot-vector components defined along each direction. The goal of the proposed strategy is to automatically determine (i.e. without the user's intervention) the full set of parameters defining the NURBS hyper-surface approximating a given set of TPs, without considering simplifying hypotheses. To this purpose, the problem is formulated as a constrained nonlinear programming problem (CNLPP) wherein the optimization variables are all the parameters tuning the shape of the NURBS hyper-surface. Nevertheless, when the number of CPs and the degrees of the basis functions are included among the design variables, the resulting problem is defined over a space having a changing dimension. This problem is solved by means of an original genetic algorithm able to determine, simultaneously, the optimum value of both the design space size (related to the integer variables of the NURBS hyper-surface) and the NURBS hyper-surface continuous parameters. The effectiveness of the proposed approach is shown by means of two meaningful test case. In addition, the proposed method has been applied to a benchmark taken from the literature and the results have been compared to those provided by the PGD approach.

Keywords: Optimization, Genetic Algorithm, NURBS hyper-surface, Surrogate model, Metamodel

1. Introduction

The computer performances significantly increased over the last two decades mainly due to the enhancement of the number of processors transistors. Although these performances are widely used to improve simulation complexity, there are still many scientific and engineering problems that remain intractable because either of their numerical complexity or their peculiar requirements, such as real-time processing. This is the case of inverse problems (and, in a more general sense, of all optimization problems), which need to compute the outputs of a model a huge number of times, or the case of deployed platforms, such as virtual and / or augmented reality, or on-line control of *Multiple-Input Multiple-Output* (MIMO) systems, that require real-time processing capabilities. To deal with this kind of problems, metamodelling techniques (also called surrogate modelling techniques) are well-

*Corresponding author. Tel.: +33 55 68 45 422, Fax.: +33 54 00 06 964.
Email address: marco.montemurro@ensam.eu, marco.montemurro@u-bordeaux.fr (Marco Montemurro)

established methods, which allow reducing the computational costs, without degrading too much the accuracy level required for the problem at hand.

The metamodelling process consists of defining a suitable *surrogate* model requiring less *resources* to be executed than the original model from which it is obtained. The meaning of resources depends on the problem at hand. As an example, image reduction aims to reduce the *number of data* needed to evaluate the metamodel [1], while a metamodel used within an optimization process aims to reduce the computatonal effort (i.e. the elapsed time) to evaluate the outputs for a given set of inputs [2, 3]. It is noteworthy that all current methods need a calibration phase that is excluded when evaluating the overall computational effort. For some complex real-world engineering problems, the calibration step can require a huge amount of time: in these circumstances the engineer should carefully evaluate the opportunity of formulating a metamodel. Consider a MIMO system characterized by N inputs and M outputs defined as:

$$\begin{array}{cccc} & \mathbb{R}^N & \rightarrow & \mathbb{R}^M \\ \mathbf{z}: & \mathbf{x} & \rightarrow & \mathbf{z}(\mathbf{x}), \end{array} \tag{1}$$

where $\mathbf{x} = \left(x^{(1)}, \dots, x^{(N)}\right)$ is the vector collecting the N inputs of the system and $\mathbf{z}(\mathbf{x}) = \left(z^{(1)}(\mathbf{x}), \dots, z^{(M)}(\mathbf{x})\right)$ is the transfer function of the MIMO system containing the M outputs. In some cases, this function may be completely known but, for real-world engineering problems, this is not always true. From a mathematical viewpoint, the metamodelling process consists of determining a function $\hat{\mathbf{z}}(\mathbf{x})$ that needs less *resources* to be evaluated than $\mathbf{z}(\mathbf{x})$:

$$\begin{array}{cccc} & \mathbb{R}^N & \rightarrow & \mathbb{R}^M \\ \hat{\mathbf{z}}: & \mathbf{x} & \rightarrow & \hat{\mathbf{z}}(\mathbf{x}) = \mathbf{z}(\mathbf{x}) + \boldsymbol{\varepsilon}(\mathbf{x}), \end{array} \tag{2}$$

where $\hat{\mathbf{z}}(\mathbf{x})$ is the function approximating the real transfer function $\mathbf{z}(\mathbf{x})$ and $\boldsymbol{\varepsilon}(\mathbf{x})$ is the approximation error at point $\mathbf{x}$. The function $\boldsymbol{\varepsilon}(\mathbf{x})$ is a bounded function, whose bounds are linked to the desired accuracy.

Metamodelling strategies can be classified into *a priori* and *a posteriori* methods. The only known *a priori* method is the Proper Generalised Decomposition (PGD) that builds a surrogate model without the need of a database resulting from the original model [1]. However, to achieve this task, a suitable mathematical formulation of the problem at hand is needed. This approach is based on a separated representation of the space: the basis functions are iteratively enhanced by adding terms until a given accuracy is reached [1, 4–6]. This method has been successfully applied to inverse problems [7], to composite damage prediction [8–12], to structural optimization problems [13, 14] and also to quantum chemical problems [15], by overcoming the so-called curse of dimensionality. Despite the wide range of applications, the main limitations of this approach are essentially four: (a) the user must define a discretization of the space and the nature of the basis functions, as well; (b) an interpolation (or approximation) technique must be foreseen to get the values that are not in the discretization step (which are generally evaluated with an accuracy worse than that affecting those falling into the discetization step); (c) in relation to the previous point, PGD gives less accurate results for nonlinear problems; (d) inasmuch as PGD is based upon variable separation, it is not suited for non-convex sets of data points.

Among the *a posteriori* metamodelling techniques, the Proper Orthogonal Decomposition (POD) [16, 17] and the kriging [18–20] are the most common methods available in the literature. POD aims to provide a reduced order problem of dimension K, by solving an eigenvalue problem and by keeping only the most representative modes of the original problem of dimension $M \gg K$ [17, 21]. The most representative eigenvalues are set by means of a suitable criterion and the method can automatically fit the given data. This approach is massively employed in computational fluid dynamics problems [16, 21–28], but it has also been used in inverse problems [29, 30] and pattern recognition [31]. The main limitation is that the online phase is related to the inversion of matrix of dimension K, which can be costly even if $K \ll M$.

Kriging [32], is known as the best unbiased linear predictor and has been developed by Matheron [33, 34]. Initially developed in the framework of geostatistics, the basis of this approach is to consider the spatial dependency structure of the data. One of the first applications of kriging is related to the well-known Design and Analysis of Computer Experiments (DACE) [20, 35–37], which focuses on the covariance computation by means of an user-defined Spatial Correlation Function (SCF). Although kriging is known to be an unbiased method because no bias is introduced during the computation, it must be stressed that some bias is inevitably introduced when the user must sets the SCF parameters. As far as the choice of the SCF is concerned, two approaches are used: cross validation and maximum likelihood that may require a big amount of time for the calibration phase of the surrogate model when the number of data points becomes important. Moreover, this method is not able to reduce the number of data points and the on-line phase requires the inversion of a matrix, whose size is directly linked to the number of data points. These limitations made this method suitable for small sets of data points.

Currently, Radial Basis Functions (RBFs) [38, 39] and Artificial Neural Networks (ANNs) [40, 41] are the only methods able to deal with problems characterized by hundreds of inputs. In particular, ANNs have been used to build surrogate models in optimization of microwave circuits [42], predictions on water salinity concentrations due to groundwater extraction [43], or injection molding process simulations [44]. The main limitations characterizing ANNs are mainly related to their intrinsic algorithmic complexity. Three steps are necessary to build an ANN: selecting the function to include (i.e. the number and the nature of neurons); selecting the neural architecture (i.e. the number of layers and the number of neurons within each layer); and training the ANN (i.e. the calibration phase, that is called learning step in the case of ANN). These tasks are anything but trivial and, even if some rules are proposed in the literature to find an appropriate neural architecture [45], they are mostly based on trial and error approaches that increase the time needed for the learning phase. Some nonlinear optimization methods exist to find the ANN parameters, but the solution is not unique and strongly problem-dependent [45]. Moreover, ANNs can suffer from the over-fitting phenomenon [46].

As a general remark, setting the number and the value of the parameters tuning the behaviour of classical metamodelling techniques could be a quite difficult task, which often needs a trial-and-error approach. For instance, improving arbitrarily the number of layers and/or neurons for ANNs, or improving the modes number for PGD, can lead to over-fitting.

The metamodelling approach proposed in this study relies on the utilisation of M-dimension (M-D) Non-Uniform Rational Basis Spline (NURBS) hyper-surfaces, defined on an a N-D parametric space, to fit a given set of data points, also called target points (TPs). Up to now, only few research works focus on the formulation/implementation of surrogate models based on the NURBS formalism [47–49]. NURBS curves and surfaces are standard geometrical entities widely used in Computer Aided Design (CAD) software [50, 51]. This particular feature made them of particular interest in the field of shape optimization, known as *isogeometric* shape optimization [52–55]. NURBS hyper-surfaces represent a generalisation of these entities. A NURBS hyper-surface is defined through the number of dimensions (related to the size of the problem at hand), the degree of blending-functions along each dimension, the overall number of control points (CPs), the coordinates of each CPs and the related weights, the knot vector components along each dimension. The large number of parameters tuning the shape of a NURBS hyper-surface makes it a versatile tool for many mathematical and engineering applications, not only for formulating surrogate models [56–63]. However, the significant amount of parameters defining the NURBS hyper-surface shape also constitutes its main drawback: it is very hard to properly tune all these parameters without making some simplifying assumptions or preliminary choices, as done in [47–49]. Up to now, hyper-surface fitting problems are solved by means of iterative procedures generalising those used in the surface fitting framework [51]. These procedures can be grouped into two categories. On the one hand, some procedures start from a minimal number of (CPs), which is iteratively increased until the algorithm reaches a given accuracy. On the other hand, some procedures make use of the opposed approach, i.e. they start from the maximum allowable number of CPs, which are iteratively removed without degrading too much (i.e. within reasonable limits) the desired accuracy.

To go beyond the aforementioned restrictions, the metamodelling approach based on NURBS hyper-surfaces is here coupled with a special genetic algorithm (GA) [64, 65] able of determining both the number and the value of the parameters affecting the NURBS hyper-surface shape, without introducing simplyfing hypotheses or empirical rules to set the values of both integer and continuous variables. In particular, when the number of parameters is included among the unknowns, the hyper-surface fitting problem can be formulated as a Constrained NonLinear Programming Problem (CNLPP) defined over a space , whose dimension is included among the variables of the problem at hand. Of course, when dealing with such a problem, a particular care must be put in the choice of the proper numerical tool to perform the solution search. To this purpose, in this study, the ERASMUS code (EvolutionaRy Algorithm for optimiSation of ModUlar Systems) [64], which is able to deal with problems characterized by a variable number of design variables, is used as an optimization tool. The effectiveness of the proposed approach is illustrated through two meaningful benchmarks.

The paper is organized as follows: the theoretical framework of NURBS hyper-surfaces and the literature survey on NURBS entities used to build metamodels are presented in Section 2. The mathematical formulation of the CNLPP is given in Section 3, while the hybrid optimization tool used to perform the solution search is briefly presented in Section 4. Numerical results on two benchmarks are shown in Section 5, whilst Section 6 ends the paper with some concluding remarks and prospects.

2. A NURBS-based surrogate model

2.1. *The NURBS hyper-surfaces theory*

The fundamentals of NURBS entities are briefly provided in the most general case of NURBS hyper-surfaces. Curves and surfaces formulae, widely discussed in [50, 51, 66, 67], can be easily deduced from the following relations. A NURBS hyper-surface is a polynomial-based function ($\mathbf{H} : \mathbb{R}^N \to \mathbb{R}^M$), defined over a *parametric space* (domain), taking values in the *NURBS space* (codomain). The mathematical formula of a generic NURBS hyper-surface is

$$\mathbf{H}\left(u^{(1)}, ..., u^{(N)}\right) = \sum_{i_1=0}^{n_1} \cdots \sum_{i_N=0}^{n_N} \frac{\prod_{k=1}^{N} N_{i_k,p_k}\left(u^{(k)}\right) \times \omega_{i_1,...,i_N} \mathbf{P}_{i_1,...,i_N}}{\sum_{j_1=0}^{n_1} \cdots \sum_{j_N=0}^{n_N} \left[\omega_{j_1,...,j_N} \prod_{k=1}^{N} N_{j_k,p_k}\left(u^{(k)}\right)\right]}, \quad (3)$$

where $\mathbf{H}\left(u^{(1)}, ..., u^{(N)}\right)$ is a M-dimension vector-valued rational function, $\left(u^{(1)}, ..., u^{(N)}\right)$ are scalar dimensionless parameters defined in the interval $[0, 1]$, whilst $\mathbf{P}_{i_1,...,i_N}$ are the so called *control points* and $\omega_{i_1,...,i_N}$ are the associated weights. The j-th control point coordinate ($P^{(j)}_{i_1,...,i_N}$) is stored in the array $\mathbf{P}^{(j)}$, whose dimension is $(n_1+1)\times\cdots\times(n_N+1)$. The explicit expression of control points coordinates in $\mathbb{R}^M$ is:

$$\begin{aligned} &\mathbf{P}_{i_1,...,i_N} = \{P^{(1)}_{i_1,...,i_N}, \ldots, P^{(M)}_{i_1,...,i_N}\}, \\ &\mathbf{P}^{(j)} \in \mathbb{R}^{(n_1+1)\times\cdots\times(n_N+1)},\ j = 1, \ldots, M. \end{aligned} \tag{4}$$

In Eq. (3), $N_{i_k,p_k}(u^{(k)})$ is the k-th *blending function* of degree p_k [51] recursively determined as follows:

$$\begin{gathered} N_{i_k,0}\left(u^{(k)}\right) = \begin{cases} 1 \text{ if } U^{(k)}_{i_k} \leq u^{(k)} < U^{(k)}_{i_k+1}, \\ 0 \text{ otherwise,} \end{cases} \\ N_{i_k,q}\left(u^{(k)}\right) = \frac{u^{(k)} - U^{(k)}_{i_k}}{U^{(k)}_{i_k+q} - U^{(k)}_{i_k}} N_{i_k,q-1}\left(u^{(k)}\right) + \frac{U^{(k)}_{i_k+q+1} - u^{(k)}}{U^{(k)}_{i_k+q+1} - U^{(k)}_{i_k+1}} N_{i_k+1,q-1}\left(u^{(k)}\right), \\ q = 1, ..., p_k. \end{gathered} \tag{5}$$

where each constitutive blending function is defined on the knot vector

$$\mathbf{U}^{(k)} = \left\{\underbrace{0, \cdots, 0}_{p_k+1}, U^{(k)}_{p_k+1}, \cdots, U^{(k)}_{m_k-p_k-1}, \underbrace{1, \cdots, 1}_{p_k+1}\right\}, \tag{6}$$

whose dimension is $m_k + 1$, with

$$m_k = n_k + p_k + 1. \tag{7}$$

For more details on NURBS hyper-surfaces the reader is addressed to [61, 68].

2.2. NURBS hyper-surfaces as metamodels: a brief literature survey

The first attempt of using NURBS hyper-surfaces to formulate suitable metamodels goes back to the works of Turner [47, 49]. In particular, in [49] a NURBS-based surrogate model used in the framework of the characterization of composite material properties is presented. In this background, Turner and Crawford developed an iterative procedure for the hyper-surface fitting problem focusing on the determination of a suitable number of CPs for the NURBS hyper-surface. However, the method proposed by Turner and Crawford is based upon an empirical rule to determine the position of new CPs that are added on the basis of the value of the cost function of the problem at hand (typically the maximum approximation error). Indeed, their approach is characterized by some restrictions:

- the degrees of the NURBS blending functions are set *a priori*;
- the knot vectors components are uniformly distributed or calculated by means of simple empirical rules;
- the weights are evaluated through empirical formulae and only for those CPs whose local support contains TPs affected by a relative error greater than a given threshold.

Of course, these empirical rules (and the parameters tuning the behaviour of the related formulae) are strongly problem-dependent and the user must have a deep knowledge of the problem at hand and of the NURBS hyper-surfaces fundamentals, as well.

In this paper, a different approach is used to build the surrogate model based on a NURBS hyper-surface for approximating the behaviour of a MIMO system. The proposed method aims to provide the function $\hat{\mathbf{z}}(\mathbf{x})$, approximating the real transfer function $\mathbf{z}(\mathbf{x})$ of the MIMO system, as follows:

$$\begin{array}{cccc} & E = \left[x^{(1)}_{\min}, x^{(1)}_{\max}\right] \times \ldots \times \left[x^{(N)}_{\min}, x^{(N)}_{\max}\right] & \rightarrow & \mathbb{R}^M \\ \hat{\mathbf{z}}: & \mathbf{x} & \rightarrow & \mathbf{H}\left(\mathbf{f}\left(\mathbf{x}\right)\right). \end{array} \tag{8}$$

where $\mathbf{x} = \left(x^{(1)}, \ldots, x^{(N)}\right)$ is the vector collecting the N inputs of the system, $\mathbf{H}\left(\mathbf{f}\left(\mathbf{x}\right)\right) = \left(X^{(1)}(\mathbf{x}), \ldots, X^{(M)}(\mathbf{x})\right)$ is the vector containing the M approximated outputs and $\mathbf{f}\left(\mathbf{x}\right)$ is a bijective function realising the mapping of the space E into the parametric domain $[0,1]^N$ of the NURBS hyper-surface, i.e.

$$\begin{array}{cccc} & E & \rightarrow & [0,1]^N \\ \mathbf{f}: & \mathbf{x} & \rightarrow & \mathbf{f}\left(\mathbf{X}\right) = \left(f^{(1)}\left(X^{(1)}\right), \ldots, f^{(N)}\left(X^{(N)}\right)\right) = \mathbf{u}, \end{array} \tag{9}$$

where $\mathbf{u} = \left(u^{(1)}, \ldots, u^{(N)}\right)$ are the parametric coordinates of the NURBS hyper-surface related to the inputs $\mathbf{x}$ and $f^{(k)}\left(x^{(k)}\right)$ are the bijective functions defined as,

$$\begin{array}{cccc} & \left[x^{(k)}_{\min}, x^{(k)}_{\max}\right] & \rightarrow & [0,1] \\ f^{(k)}: & x^{(k)} & \rightarrow & f^{(k)}\left(x^{(k)}\right) = \dfrac{x^{(k)} - x^{(k)}_{min}}{x^{(k)}_{max} - x^{(k)}_{min}} = u^{(k)}, \quad k = 1, ..., N. \end{array} \tag{10}$$

Here the goal is to formulate a surrogate model based on NURBS entities without introducing neither simplifying hypotheses nor empirical rules to set *all the parameters* affecting the shape of the NURBS hyper-surface. Only in this way, it is possible to implement a problem-independent metamodelling strategy: this point constitutes the kernel of the proposed approach.

As explained in the Section 3, this ambitious goal can be achieved through a pertinent formulation of the metamodel generation problem. Such a problem is formulated as a constrained hyper-surface fitting problem and the surrogate model based on NURBS hyper-surfaces is efficiently coupled to a special GA [64, 65] able of determining both the number and the value of the parameters involved in the definition of the NURBS hyper-surface.

3. Problem formulation

3.1. Design variables

Eq. (3) shows that the parameters involved in the definition of a NURBS hyper-surface are of different nature:

- *integer variables*, like the number of both knot vector components and CPs, (m_k+1) and (n_k+1) respectively, as well as the degrees of the blending functions p_k along each dimension;

- *continuous variables*, like internal knot vector values $U^{(k)}_{p_k+1}, ..., U^{(k)}_{m_k-p_k-1}$, CPs coordinates $\mathbf{P}_{i_1,...,i_N}$, weights $\omega_{i_1,...,i_N}$ and the dimensionless parameters $u^{(k)}_s$ at which the NURBS hyper-surface is evaluated.

Some of these parameters are interdependent, whereas other can be smartly chosen. In particular, as far as the dimensionless parameters are concerned, i.e. $u_s^{(k)}$, they are obtained by applying Eq. (10) to the inputs of the MIMO for each TP, i.e.

$$u_s^{(k)} = \frac{x_s^{(k)} - x_{min}^{(k)}}{x_{max}^{(k)} - x_{min}^{(k)}}, \quad k = 1, \ldots, N, \quad s = 1 \ldots, n_{TP}, \tag{11}$$

where n_{TP} is the overall number of TPs. Therefore, the NURBS dimensionless parameters do not belong to the set of design variables. Moreover, the number of CPs along each parametric direction can be determined once the size of the knot vector and the degree of the blending functions along the same direction are known, according to Eq. (7). Accordingly, the number of CPs is excluded from the design variables vector.

The determination of the optimum value of CPs coordinates is carried out by a dedicated algorithm. In particular, when the size of the knot vector as well as its internal components along each direction are known, if the degree of the blending functions (along each direction) is given and if the values of $u_s^{(k)}$ have been computed by means of Eq. (11), finding the optimum value of the CPs coordinates is a quite trivial task. Indeed, the NURBS hyper-surface fitting problem is convex in terms of CPs coordinates. In the case of B-Spline hyper-surface fitting, it can be shown that the optimum value of the CPs coordinates can be determined by generalizing the well-known surface fitting algorithm $A9.7$, presented in [51], to the N-D case. However, two cases must be considered. If TPs are collected into a *sorted* set, i.e.

$$\mathbf{Q}_{sort} : \left\{ \mathbf{Q}_{s_1,\ldots,s_N} = \left(Q^{(1)}_{s_1,\ldots,s_N}, \ldots, Q^{(M)}_{s_1,\ldots,s_N} \right), \quad s_k = 0, \ldots, r_k, \quad k = 1, \ldots, N \right\}, \tag{12}$$

the algorithm structure is very simple and it is illustrated in Fig 1. In Eq.(12), $\mathbf{Q}_{s_1,\ldots,s_N}$ is the generic TP of the set and $(r_k + 1)$ is the number of TPs along the direction k. For the sake of brevity, it can be stated that the optimum value of CPs coordinates can be computed as a succession of curve fitting problems along each direction. In particular, Fig. 1 highlights that *temporary* CP coordinates are recursively computed along each direction. The final CPs coordinates are stored in the array $\mathbf{P}$. More details can be found in [51].

When considering the fitting in direction l, the size of the temporary CPs coordinates array $\mathbf{P}^{(l-1)}$ is reduced from $(r_l + 1)$ to $(n_l + 1)$ in the next temporary CPs coordinates array $\mathbf{P}^{(l)}$. This is achieved by solving $(n_1+1)\times\cdots\times(n_{l-1}+1)\times(r_{l+1}+1)\times\cdots\times(r_N+1)$ times the following problem:

$$\left(\mathbf{N}_l^T \mathbf{N}_l\right) \mathbf{P}^{(l)}_{i_1,\ldots,i_{l-1},:,s_{l+1},\ldots,s_N} = \mathbf{N}_l^T \mathbf{P}^{(l-1)}_{i_1,\ldots,i_{l-1},:,s_{l+1},\ldots,s_N}, \tag{13}$$

where $\mathbf{P}^{(l)}_{i_1,\ldots,i_{l-1},:,s_{l+1},\ldots,s_N}$ is the matrix collecting the temporary CPs coordinates at iteration l given by

$$\mathbf{P}^{(l)}_{i_1,\ldots,i_{l-1},:,s_{l+1},\ldots,s_N} = \begin{pmatrix} P^{(l,1)}_{i_1,\ldots,i_{l-1},0,s_{l+1},\ldots,s_N} & \cdots & P^{(l,M)}_{i_1,\ldots,i_{l-1},0,s_{l+1},\ldots,s_N} \\ \vdots & \ddots & \vdots \\ P^{(l,1)}_{i_1,\ldots,i_{l-1},n_l,s_{l+1},\ldots,s_N} & \cdots & P^{(l,M)}_{i_1,\ldots,i_{l-1},n_l,s_{l+1},\ldots,s_N} \end{pmatrix}, \tag{14}$$

$$i_k = 0, \ldots, n_k, \;\; k = 1, \ldots, l-1 \quad ; \quad s_k = 0, \ldots, r_k, \;\; k = l+1, \ldots, N,$$

whilst $\mathbf{P}^{(l-1)}_{i_1,\ldots,i_{l-1},:,s_{l+1},\ldots,s_N}$ is the matrix collecting the temporary CPs coordinates at iter-

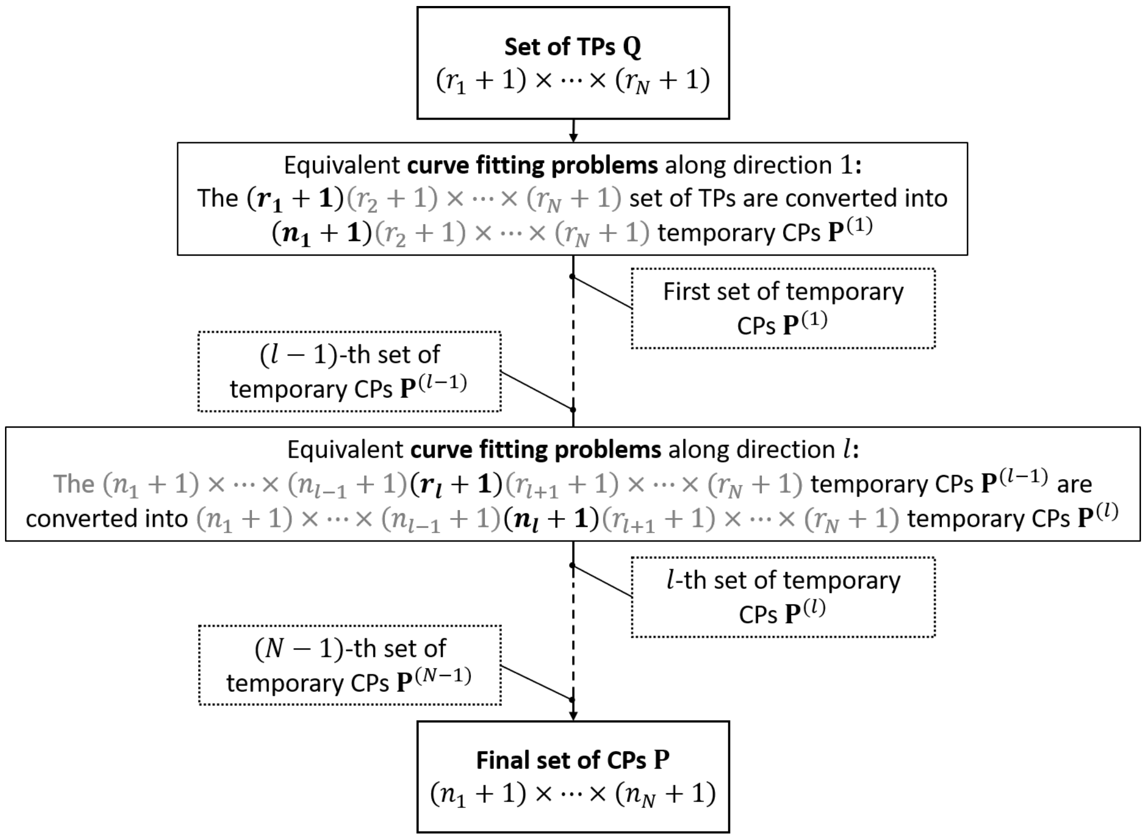

Set of TPs Q
$(r_1+1)\times\cdots\times(r_N+1)$
Equivalent curve fitting problems along direction 1:
The $(\boldsymbol{r_1}+\mathbf{1})(r_2+1)\times\cdots\times(r_N+1)$ set of TPs are converted into $(\boldsymbol{n_1}+\mathbf{1})(r_2+1)\times\cdots\times(r_N+1)$ temporary CPs $\mathbf{P}^{(1)}$
First set of temporary CPs $\mathbf{P}^{(1)}$
$(l-1)$-th set of temporary CPs $\mathbf{P}^{(l-1)}$
Equivalent curve fitting problems along direction l:
The $(n_1+1)\times\cdots\times(n_{l-1}+1)(\boldsymbol{r_l}+\mathbf{1})(r_{l+1}+1)\times\cdots\times(r_N+1)$ temporary CPs $\mathbf{P}^{(l-1)}$ are converted into $(n_1+1)\times\cdots\times(n_{l-1}+1)(\boldsymbol{n_l}+\mathbf{1})(r_{l+1}+1)\times\cdots\times(r_N+1)$ temporary CPs $\mathbf{P}^{(l)}$
l-th set of temporary CPs $\mathbf{P}^{(l)}$
$(N-1)$-th set of temporary CPs $\mathbf{P}^{(N-1)}$
Final set of CPs P
$(n_1+1)\times\cdots\times(n_N+1)$

Figure 1: Logical flow of the surface fitting algorithm [51] extended to the N-D case.

ation $l-1$ given by

$$\mathbf{P}^{(l-1)}_{i_1,\dots,i_{l-1},:,s_{l+1},\dots,s_N} = \begin{pmatrix} P^{(l-1,1)}_{i_1,\dots,i_{l-1},0,s_{l+1},\dots,s_N} & \cdots & P^{(l-1,M)}_{i_1,\dots,i_{l-1},0,s_{l+1},\dots,s_N} \\ \vdots & \ddots & \vdots \\ P^{(l-1,1)}_{i_1,\dots,i_{l-1},r_l,s_{l+1},\dots,s_N} & \cdots & P^{(l-1,M)}_{i_1,\dots,i_{l-1},r_l,s_{l+1},\dots,s_N} \end{pmatrix}, \tag{15}$$
$$i_k = 0,\dots,n_k,\ k=1,\dots,l-1 \quad ; \quad s_k = 0,\dots,r_k,\ k=l+1,\dots,N.$$

In Eq. (13), the matrix $\mathbf{N}_l$ collects the B-Spline basis functions as follows

$$\mathbf{N}_l = \begin{pmatrix} N_{0,p_l}\left(u_0^{(l)}\right) & \cdots & N_{n_l,p_l}\left(u_0^{(l)}\right) \\ \vdots & \ddots & \vdots \\ N_{0,p_l}\left(u_{r_l}^{(l)}\right) & \cdots & N_{n_l,p_l}\left(u_{r_l}^{(l)}\right) \end{pmatrix}, \quad l=1,\dots,N. \tag{16}$$

It is noteworthy that when $l=1$, $\mathbf{P}^{(0)}_{:,s_2,\dots,s_N}$ is directly given by the TPs coordinates (i.e. $\mathbf{Q}_{:,s_2,\dots,s_N}$), while $\mathbf{P}^{(N)}_{i_1,\dots,i_{N-1},:}$ gives directly the CPs coordinates when $l=N$.

The problem becomes more difficult when the TPs cannot be stored in a sorted set along each dimension. In particular, let $L<N$ be the number of dimensions along which TPs cannot be arranged in a sorted set and (r_L+1) the number of the related TPs. In this case, which can be considered as a hybrid situation, the structure, of the algorithm is presented in Fig. 2. The main difference is that the temporary values of CPs coordinates, at the first iteration, are computed as a succession of L-D hyper-surface fitting problems

instead of curve fitting problems. Fig. 2 shows that the first temporary array of CPs coordinates $\mathbf{P}^{(L)}$ reduces the size of the TPs set $\mathbf{Q}$ from (r_L+1) to $(n_1+1)\times\cdots\times(n_L+1)$ along the L first directions. This implies to solve $(r_{L+1}+1)\times\cdots\times(r_N+1)$ times the following problem

$$\left(\mathbf{N}^T\mathbf{N}\right)\mathbf{P}^{(L)}_{:,\dots,:,s_{L+1},\dots,s_N} = \mathbf{N}^T\mathbf{Q}_{:,s_{L+1},\dots,s_N}, \tag{17}$$

where $\mathbf{P}^{(L)}_{:,\dots,:,s_{L+1},\dots,s_N}$ is the matrix collecting the temporary CPs coordinates at the first iteration (i.e. the L-D hyper-surface) given by

$$\mathbf{P}^{(L)}_{:,\dots,:,s_{L+1},\dots,s_N} = \begin{pmatrix} P^{(L,1)}_{0,\dots,0,s_{L+1},\dots,s_N} & \cdots & P^{(L,M)}_{0,\dots,0,s_{L+1},\dots,s_N} \\ \vdots & \ddots & \vdots \\ P^{(L,1)}_{n_1,\dots,n_L,s_{L+1},\dots,s_N} & \cdots & P^{(L,M)}_{n_1,\dots,n_L,s_{L+1},\dots,s_N} \end{pmatrix}, \tag{18}$$
$$s_k = 0,\dots,r_k, \quad k = L+1,\dots,N,$$

while $\mathbf{Q}_{:,s_{L+1},\dots,s_N}$ is the matrix of TPs coordinates:

$$\mathbf{Q}_{:,s_{L+1},\dots,s_N} = \begin{pmatrix} Q^{(1)}_{0,s_{L+1},\dots,s_N} & \cdots & Q^{(M)}_{0,s_{L+1},\dots,s_N} \\ \vdots & \ddots & \vdots \\ Q^{(1)}_{r_L,s_{L+1},\dots,s_N} & \cdots & Q^{(M)}_{r_L,s_{L+1},\dots,s_N} \end{pmatrix}, \quad s_k = 0,\dots,r_k,\ k = L+1,\dots,N. \tag{19}$$

In Eq. (17), the matrix $\mathbf{N}$ collects the B-Spline basis functions products as follows

$$\mathbf{N} = \begin{pmatrix} N_{0,p_l}\left(u_0^{(l)}\right)\times\cdots\times N_{0,p_L}\left(u_0^{(L)}\right) & \cdots & N_{n_l,p_l}\left(u_0^{(l)}\right)\times\cdots\times N_{n_L,p_L}\left(u_0^{(L)}\right) \\ \vdots & \ddots & \vdots \\ N_{0,p_l}\left(u_{r_L}^{(l)}\right)\times\cdots\times N_{0,p_L}\left(u_{r_L}^{(L)}\right) & \cdots & N_{n_l,p_l}\left(u_{r_L}^{(l)}\right)\times\cdots\times N_{n_L,p_L}\left(u_{r_L}^{(L)}\right) \end{pmatrix}. \tag{20}$$

When the first temporary CPs coordinates array $\mathbf{P}^{(L)}$ is assessed, the algorithm of Fig. 2 computes the next temporary CPs coordinates array $\mathbf{P}^{(l)}$ according to the strategy illustrated in Fig. 1. This results in a succession of curve fitting problems, solving Eq. (13) for $l = L+1,\dots,N$. Note that the special case $L = N$ implies to directly realize the hyper-surface fitting which can be very costly when compared to the proposed algorithms.

Finally, for both cases, the independent parameters tuning the NURBS hyper-surface shape are:

- the N degrees p_k;
- the N knot lengths $m_k + 1$;
- the $m_k - 2p_k - 1$ internal components of the knot vector $\mathbf{U}^{(k)}$, $k = 1,\dots,N$;
- the $n_{CP} = \prod_{k=1}^{N}(n_k+1)$ weights $\omega_{i_1,\dots,i_N}$.

These parameters are collected in the vector of design variables $\boldsymbol{\xi}$ as

$$\begin{aligned} \boldsymbol{\xi} \;=\; & \Big(p_1,\dots,p_N,m_1,\dots,m_N,U^{(1)}_{p_1+1},\dots,U^{(1)}_{m_1-p_1-1},\dots, \\ & U^{(N)}_{p_N+1},\dots,U^{(N)}_{m_N-p_N-1},\omega_{0,\dots,0},\dots,\omega_{n_1,\dots,n_N}\Big). \end{aligned} \tag{21}$$

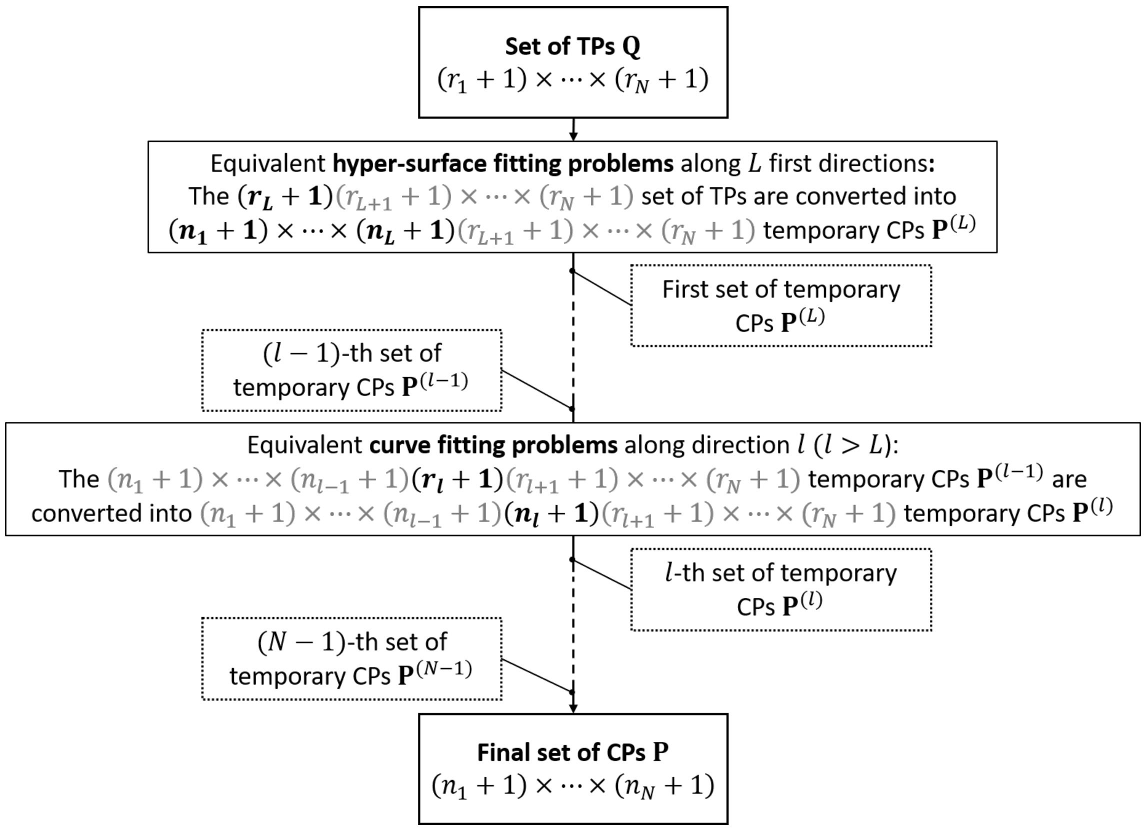

Set of TPs $\mathbf{Q}$
$(r_1+1)\times\cdots\times(r_N+1)$
Equivalent hyper-surface fitting problems along L first directions:
The $(r_L+1)(r_{L+1}+1)\times\cdots\times(r_N+1)$ set of TPs are converted into
$(n_1+1)\times\cdots\times(n_L+1)(r_{L+1}+1)\times\cdots\times(r_N+1)$ temporary CPs $\mathbf{P}^{(L)}$
First set of temporary CPs $\mathbf{P}^{(L)}$
$(l-1)$-th set of temporary CPs $\mathbf{P}^{(l-1)}$
Equivalent curve fitting problems along direction l $(l>L)$:
The $(n_1+1)\times\cdots\times(n_{l-1}+1)(r_l+1)(r_{l+1}+1)\times\cdots\times(r_N+1)$ temporary CPs $\mathbf{P}^{(l-1)}$ are
converted into $(n_1+1)\times\cdots\times(n_{l-1}+1)(n_l+1)(r_{l+1}+1)\times\cdots\times(r_N+1)$ temporary CPs $\mathbf{P}^{(l)}$
l-th set of temporary CPs $\mathbf{P}^{(l)}$
$(N-1)$-th set of temporary CPs $\mathbf{P}^{(N-1)}$
Final set of CPs $\mathbf{P}$
$(n_1+1)\times\cdots\times(n_N+1)$

Figure 2: Logical flow of the surface fitting algorithm [51] extended to the N-D case for unsorted set of TPs.

It is noteworthy that the number of independent parameters defining the NURBS hyper-surface is given by the following equation

$$n_{\text{var}} = 2N + \sum_{k=1}^{N}(m_k - 2p_k - 1) + \prod_{k=1}^{N}(n_k+1)\,, \tag{22}$$

which depends upon the integer variables of the hyper-surface.

3.2. Problem formulation

The goal of the proposed approach is to obtain an optimized surrogate model based on NURBS hyper-surfaces. Therefore, a pertinent choice would be to formulate the metamodelling problem as a hyper-surface fitting problem by minimizing the Euclidean distance (in M dimensions) between the NURBS hyper-surface and the set of TPs. Nevertheless, the main idea is to search for the best value of the parameters tuning the shape of the NURBS hyper-surface by optimizing the overall number of CPs (n_{CP}) and blending functions degrees, by keeping a sufficient accuracy (according to the problem requirements). Of course, the number of CPs is related to the number of data needed to evaluate the outputs of the metamodel, whereas the blending function degrees are related to the processing time of the metamodel (recall that the NURBS blending functions are recursively evaluated according to Eq. (5)). Accordingly, the objective function is defined as:

$$\Phi(\boldsymbol{\xi}) = a\sum_{k=0}^{N}\frac{n_k}{n_{k_{\max}}} + (1-a)\sum_{k=0}^{N}\frac{p_k}{p_{k_{\max}}}, \tag{23}$$

where a is a suitable weighting coefficient balancing the number of CPs and the NURBS hyper-surface processing time (via the degrees). As stated above, the idea is to obtain an optimized surrogate model minimizing the number of resources by ensuring a given level of accuracy. The requirement on the model accuracy can be formalized as:

$$\varepsilon_{\text{max}}^{(j)}\left(\boldsymbol{\xi}\right) \leq \varepsilon_{\text{th}}^{(j)},\ j = 1, \ldots, M, \tag{24}$$

where $\varepsilon_{\text{th}}^{(j)}$ is the maximum relative error threshold related to the required accuracy for the j-th output of the surrogate model, whilst $\varepsilon_{\text{max}}^{(j)}\left(\boldsymbol{\xi}\right)$ is the maximum relative error for the NURBS hyper-surface, i.e.

$$\varepsilon_{\text{max}}^{(j)}\left(\boldsymbol{\xi}\right) = \max_{\mathbf{u}_s}\left(\varepsilon^{(j)}\left(\mathbf{u}_s\right)\right), \tag{25}$$

where $\varepsilon^{(j)}\left(\mathbf{u}_s\right)$ is the relative error at parametric coordinates $\mathbf{u}_s = \left(u_s^{(1)}, ..., u_s^{(N)}\right)$ given by

$$\varepsilon^{(j)}\left(\mathbf{u}_s\right) = \frac{|H^{(j)}\left(\mathbf{u}_s\right) - z^{(j)}\left(\mathbf{x}_s\right)|}{z_{\text{max}}^{(j)} - z_{\text{min}}^{(j)}}, \quad j = 1, \ldots, M, \quad s = 1, ..., n_{TP}. \tag{26}$$

In Eq. (26), $z^{(j)}\left(\mathbf{x}_s\right)$ is the j-th output of the s-th TP at inputs $\mathbf{x}_s = \left(x_s^{(1)}, \ldots, x_s^{(N)}\right)$, while $H^{(j)}\left(\mathbf{u}_s\right)$ is the j-th counterpart evaluated by the metamodel (i.e. the j-th coordinate of the NURBS hyper-surface) at the parametric coordinates $\mathbf{u}_s = \left(u_s^{(1)}, \ldots, u_s^{(N)}\right)$. The scalar quantities $z_{\text{max}}^{(j)}$ and $z_{\text{min}}^{(j)}$ are the maximum and minimum values of the j-th output over the set of TPs, respectively.

The hyper-surface fitting problem can be stated in the form of an unconventional CNLPP as:

$$\begin{gathered}
\min_{\boldsymbol{\xi}} \Phi(\boldsymbol{\xi}) = a \sum_{k=0}^{N} \frac{n_k}{n_{k_{\text{max}}}} + (1-a) \sum_{k=0}^{N} \frac{p_k}{p_{k_{\text{max}}}}, \\
\text{subject to :} \\
\begin{cases}
\varepsilon_{\text{max}}^{(j)}\left(\boldsymbol{\xi}\right) \leq \varepsilon_{\text{th}}^{(j)},\ j = 1, \ldots, M, \\
0 \leq U_{l_k}^{(k)} \leq 1,\ l_k = p_k + 1, \ldots, m_k - p_k + 1,\ k = 1, \ldots, N, \\
U_{l_k}^{(k)} \leq U_{l_k+1}^{(k)},\ l_k = p_k + 1, \ldots, m_k - p_k + 1,\ k = 1, \ldots, N, \\
n_k - p_k \geq 0,\ k = 1, \ldots, N, \\
\omega_{i_1, \ldots, i_N} \geq 0,\ i_k = 0, \ldots, n_k.
\end{cases}
\end{gathered} \tag{27}$$

Problem (27) is a non-standard CNLPP for different reasons. Firstly, unlike the methods available in the literature [47, 51], the proposed strategy aims at providing all the design variables $\boldsymbol{\xi}$ without introducing neither simplifying hypotheses nor empirical rules on the parameters involved in the NURBS hyper-surface definition. Secondly, the number of design variables is integrated into the design variables vector $\boldsymbol{\xi}$ and depends upon the integer parameters of the NURBS hyper-surface. As explained in [64], the CNLPP (27) can be efficiently stated as an optimization problem of modular systems belonging to different families. Generally speaking, a *modular system* is composed by *elementary units*, i.e. the *modules*. Each module is characterized by the same vector of unknowns, i.e. the design variables of the module, which can take, a priori, different values for every module (in the most general case of different modules). For problem (27), two different classes of modules can be identified: the knot vector components and the weights. CNLPPs dealing with modular systems are unconventional because they are defined over a domain having a dimension which depends upon a linear combination of the integer variables characterizing the modular system. In particular, for problem (27) the problem dimension is given by Eq. (22).

4. Numerical strategy

Considering the peculiar nature of problem (27) a hybrid optimization tool developed at I2M laboratory in Bordeaux and called HERO (Hybrid EvolutionaRy Optimization) has been developed. It is composed of a genetic algorithm ERASMUS (EvolutionaRy Algorithm for optimiSation of ModUlar Systems) [64], interfaced with *active-set* algorithm of MATLAB *fmincon* family, available in the MATLAB *optimization toolbox* [69], see Fig. 3.

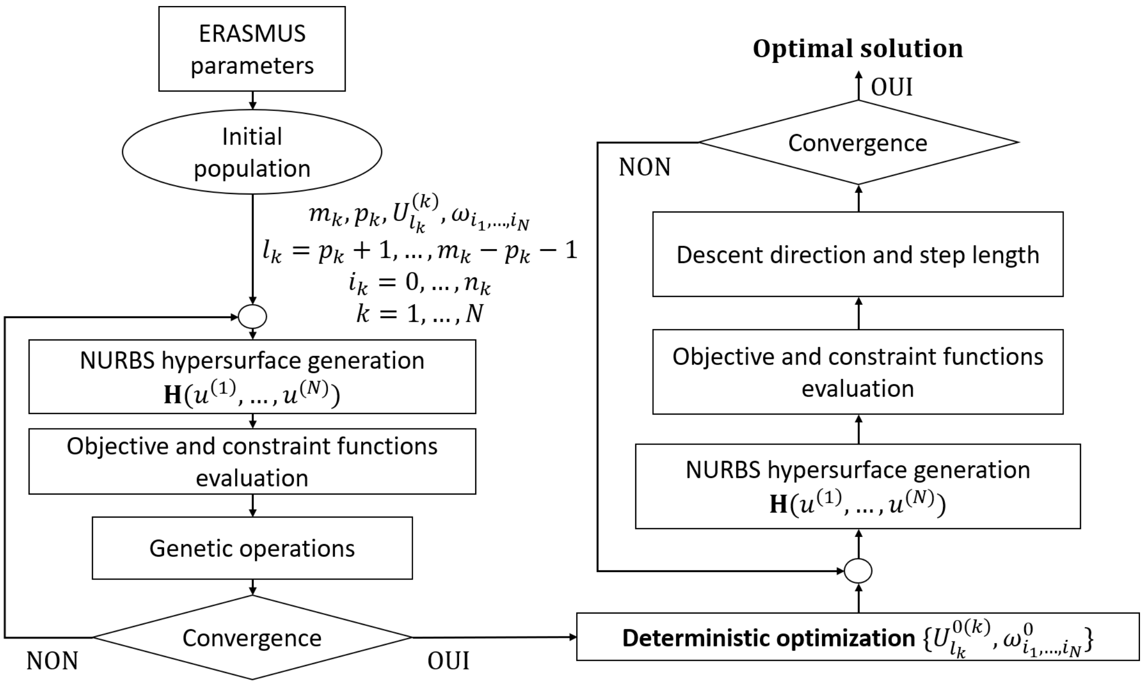

ERASMUS parameters
Initial population
$m_k, p_k, U_{l_k}^{(k)}, \omega_{i_1,\dots,i_N}$
$l_k = p_k+1,\dots,m_k-p_k-1$
$i_k = 0,\dots,n_k$
$k = 1,\dots,N$
NURBS hypersurface generation $\mathbf{H}(u^{(1)},\dots,u^{(N)})$
Objective and constraint functions evaluation
Genetic operations
Convergence
NON
OUI
Deterministic optimization $\{U_{l_k}^{0(k)}, \omega_{i_1,\dots,i_N}^{0}\}$
NURBS hypersurface generation $\mathbf{H}(u^{(1)},\dots,u^{(N)})$
Objective and constraint functions evaluation
Descent direction and step length
Convergence
NON
OUI
Optimal solution

Figure 3: Hybrid EvolutionaRy Optimization (HERO) algorithm.

As shown in Fig. 3, the optimization procedure for problem (27) is split in two phases. During the first step, solely the GA ERASMUS is employed to perform the solution search and the full set of design variables is taken into account. ERASMUS is a special GA able to deal with optimization problems characterized by a non-constant number of design variables, and more specifically, optimization problems of *modular systems*. This goal can be achieved thanks to the original representation of information in ERASMUS, i.e. the individual's *genotype*, which is organized in *modular parts*, each one composed of *chromosomes* which are in turn made of *genes* (each gene codes a specific design variable).

In agreement with the paradigms of natural sciences, individuals characterized by a different number of chromosomes (i.e. modular structures composed of a different number of modules) belong to different *species*. ERASMUS has been conceived for crossing also different species, thus making possible (and without distinction) the *simultaneous optimization of species and individuals*. This task can be attained thanks to some special genetic operators that have been implemented to perform the reproduction phase between individuals belonging to different species: the general architecture of ERASMUS is illustrated in Fig. 4. Accordingly, ERASMUS is able to optimize both the number of modules (for each class of modules) and the values of the design variables characterizing each module simultaneously. The effectiveness of ERASMUS has been shown on a large number of real-world engineering problems [57–59, 70–75].

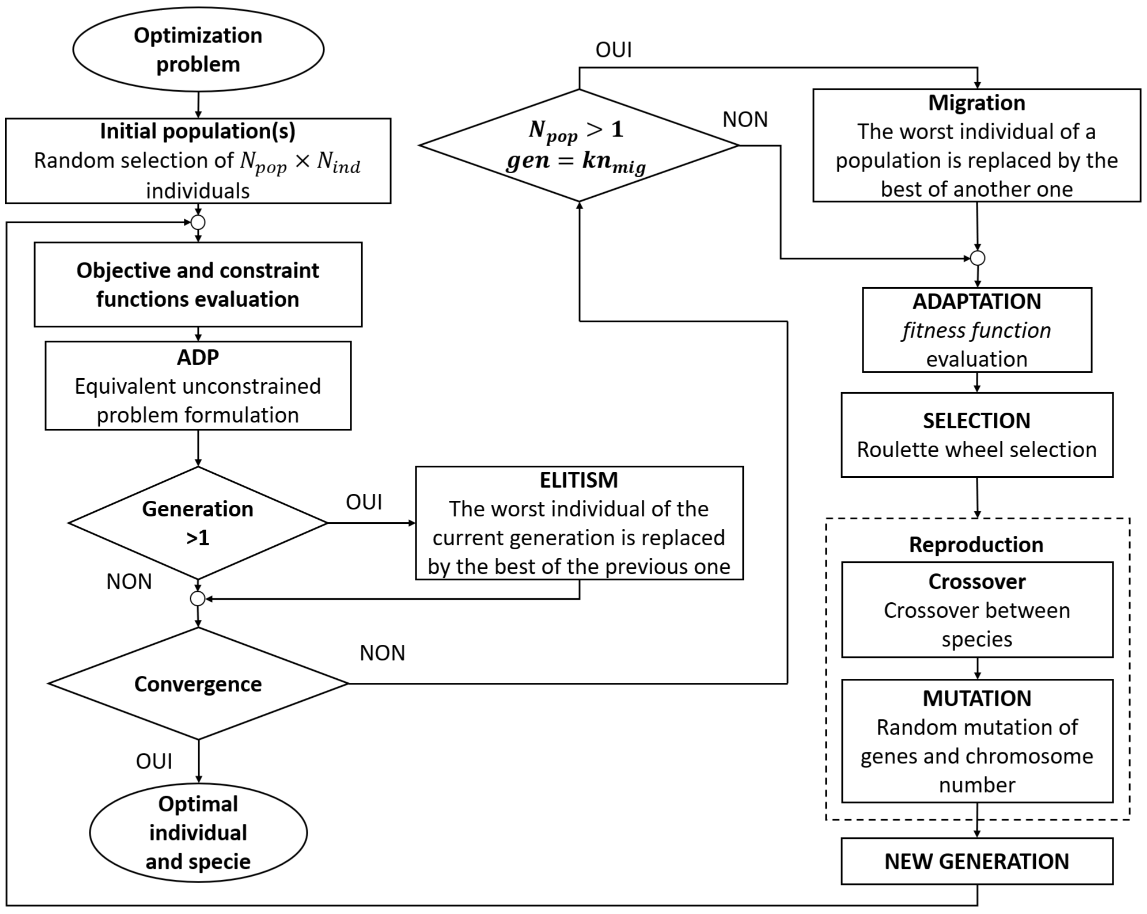

Optimization problem
Initial population(s)
Random selection of $N_{pop} \times N_{ind}$ individuals
Objective and constraint functions evaluation
ADP
Equivalent unconstrained problem formulation
Generation >1
OUI
NON
ELITISM
The worst individual of the current generation is replaced by the best of the previous one
Convergence
NON
OUI
Optimal individual and specie
$N_{pop} > 1$
$gen = kn_{mig}$
OUI
NON
Migration
The worst individual of a population is replaced by the best of another one
ADAPTATION
fitness function evaluation
SELECTION
Roulette wheel selection
Reproduction
Crossover
Crossover between species
MUTATION
Random mutation of genes and chromosome number
NEW GENERATION

Figure 4: ERASMUS algorithm [64].

Due to the strong nonlinearity of problem (27), the aim of the genetic calculation is to provide a potential sub-optimal point in the design space, which constitutes the initial guess for the subsequent phase, i.e. the local optimization performed via the *active-set* algorithm. During this second phase, only the components of the knot vector along each dimension and the weights are considered as design variables, see Fig. 3. The second phase of HERO allows finding a local minimum for the number of parameters resulting from the first GA exploration of the design space.

4.1. Meta-heuristic exploration

As stated above, when the number of internal knots and the degree along each parametric direction are included among the design variables, problem (27) is defined over a space of changing dimension. Moreover, this CNLPP is characterized by a large number of design variables. In particular, when all the weights are included into the design variables vector, the computational cost could become prohibitive. To this purpose, a dedicated strategy able to determine which weights should be integrated in the optimization process has been developed. This task is achieved by splitting the exploration of the design space into two steps. In the first step, all weights are set to one, i.e. only B-Spline hyper-surfaces are used to fit the set of TPs. Then, if the error threshold is no satisfied, the *local support* property [51] of NURBS hyper-surfaces is used to assess the weights that must be integrated as design variables for the second step of the exploration. If the approximation error is satisfied after the first step, all weights are kept equal to one and the B-Spline hyper-surface obtained by ERASMUS is used as a starting point for the subsequent deterministic optimization.

4.1.1. Meta-heuristic exploration: first step

In this first step, all weights $\omega_{i_1,\dots,i_N}$ are set to one and only B-Spline entities are used to fit the TPs. As a result, the CNLPP is stated as:

$$\min_{\boldsymbol{\xi}_{\mathrm{I}}} \Phi(\boldsymbol{\xi}_{\mathrm{I}}) = a\sum_{k=0}^{N}\frac{n_k}{n_{k_{\max}}} + (1-a)\sum_{k=0}^{N}\frac{p_k}{p_{k_{\max}}},$$
$$\text{subject to}$$
$$\begin{cases} \varepsilon^{(j)}_{\max}(\boldsymbol{\xi}_{\mathrm{I}}) \leq \varepsilon^{(j)}_{\mathrm{th}},\ j=1,\dots,M,\\ 0 < U^{(k)}_{l_k} < 1,\ l_k = p_k+1,\dots,m_k-p_k+1,\ k=1,\dots,N,\\ U^{(k)}_{l_k} \leq U^{(k)}_{l_k+1},\ l_k = p_k+1,\dots,m_k-p_k,\ k=1,\dots,N,\\ n_k - p_k \geq 0,\ k=1,\dots,N,\\ \omega_{i_1,\dots,i_N} = 1,\ i_k = 0,\dots,n_k,\ k=1,\dots,N. \end{cases} \tag{28}$$

The vector $\boldsymbol{\xi}_{\mathrm{I}}$ collects the optimization variables as follows,

$$\boldsymbol{\xi}_{\mathrm{I}} = \left(p_1,\dots,p_N,m_1,\dots,m_N,U^{(1)}_{p_1+1},\dots,U^{(1)}_{m_1-p_1-1},\dots,U^{(N)}_{p_N+1},\dots,U^{(N)}_{m_N-p_N-1}\right). \tag{29}$$

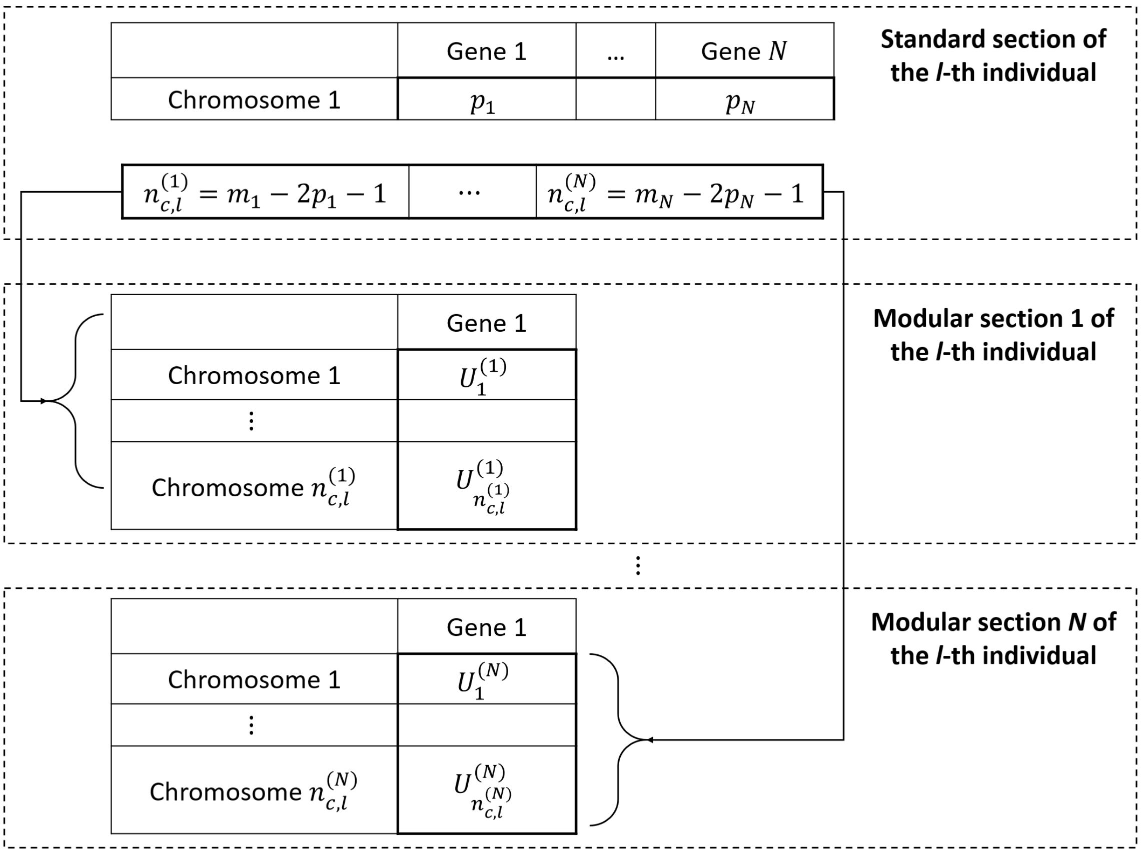
It is noteworthy that a B-Spline hyper-surface can be considered as a modular system, where m_k and p_k are standard design variables, whilst each knot vector $\mathbf{U}^{(k)}$ represents the generic module, whose variables are $U^{(k)}_{l_k}$, $l_k = p_k+1,\dots,m_k-p_k-1$, $k=1,\dots,N$. This system is thus composed of N modules corresponding to the N knot vectors of the B-Spline hyper-surface. The individual genotype of ERASMUS, for problem (28), is given in Fig. 5, where $n^{(k)}_{c,l}$ is the number of chromosomes of the k-th modular part (related to the knot vector $\mathbf{U}^{(k)}$) of the l-th individual. This quantity corresponds to the number of internal components of the knot vector $\mathbf{U}^{(k)}$ and is related to m_k and p_k by the following relation:

$$n^{(k)}_{c,l} = m_k - 2p_k - 1. \tag{30}$$

This step aims at setting up the number of optimization variables (i.e. the number of the NURBS hyper-surface parameters). Thus, the degrees p^{I}_k and the knot vector lengths m^{I}_k+1 given by ERASMUS remain constant for the rest of the optimization process. Moreover, by means of the user-defined error threshold $\varepsilon^{(j)}_{\mathrm{th}}$ and the *local support* property of NURBS hyper-surfaces, the following set $\boldsymbol{\Omega}$ can be defined

$$\boldsymbol{\Omega} = \left\{\omega_{i_1,\dots,i_N},\ i_k = 0,\dots,n_k \quad | \quad \exists \mathbf{u}_s \in \mathbb{U}\cap\mathbb{S}^{(i_1,\dots,i_N)}\right\}, \tag{31}$$

where $\mathbb{S}^{(i_1,\dots,i_N)}$ is the local support of control point $\mathbf{P}_{i_1,\dots,i_N}$ (and therefore of the weight



Gene 1
...
Gene N
Standard section of the l-th individual
Chromosome 1
p_1
p_N
$n_{c,l}^{(1)} = m_1 - 2p_1 - 1$
$\cdots$
$n_{c,l}^{(N)} = m_N - 2p_N - 1$
Modular section 1 of the l-th individual
Gene 1
Chromosome 1
$U_1^{(1)}$
Chromosome $n_{c,l}^{(1)}$
$U_{n_{c,l}^{(1)}}^{(1)}$
Modular section N of the l-th individual
Gene 1
Chromosome 1
$U_1^{(N)}$
Chromosome $n_{c,l}^{(N)}$
$U_{n_{c,l}^{(N)}}^{(N)}$

Figure 5: Individual genotype of ERASMUS for the NURBS hyper-surface fitting problem.

$\omega_{i_1,\ldots,i_N}$) defined as follow

$$\mathbb{S}^{(i_1,\ldots,i_N)} = \left[U_{i_1}^{(1)}, U_{i_1+p_1+1}^{(1)}\right[\times \ldots \times \left[U_{i_N}^{(N)}, U_{i_N+p_N+1}^{(N)}\right[, \tag{32}$$

while $\mathbb{U}$ is the set of TPs at which the maximum relative error does not meet the user-defined threshold

$$\mathbb{U} = \left\{\mathbf{u}_s, \ s = 1, \ldots, n_{TP} \quad | \quad \exists j \in [1, M] : \varepsilon^{(j)}(\mathbf{u}_s) > \varepsilon_{\text{th}}^{(j)}\right\}. \tag{33}$$

If $\boldsymbol{\Omega}$ is not empty, the weights that belong to $\boldsymbol{\Omega}$ are introduced in the optimization process for the second step of the exploration. Therefore, our approach is able to automatically switch between B-Spline and NURBS formalism to best fit the set of TPs. Moreover, if $\boldsymbol{\Omega}$ is empty, the second step of the meta-heuristic exploration does not occur since no optimization variables are added to the optimization process.

It is noteworthy that function Φ of Eq. (27) depends only upon the integer parameters of the NURBS hyper-surface m_k and p_k. As a result, this function cannot be used within a deterministic algorithm because these parameters are set equal to the values provided by the genetic step. Moreover, deterministic algorithms are not able to deal with modular systems optimization. To this purpose, a different problem formulation is used during the local optimization carried out by means of *active-set* algorithm.

4.1.2. Meta-heuristic exploration: second step

During the second step of the meta-heuristic exploration, the number of optimization variables is set (because the number of knot vector components and the degree along each direction are the result of the first optimization step). Therefore, only continuous design variables are considered at this stage, namely the weights belonging to the set $\mathbf{\Omega}$ as well as the knot vectors components. They are grouped as follows:

$$\boldsymbol{\xi}_{\mathrm{II}} = \left(U^{(1)}_{p_1+1}, \ldots, U^{(1)}_{m_1-p_1-1}, \ldots, U^{(N)}_{p_N+1}, \ldots, U^{(N)}_{m_N-p_N-1}, \mathbf{\Omega} \right). \tag{34}$$

This step takes place only if the set $\mathbf{\Omega}$ is not empty and aims at providing a potential sub-optimal point constituting the initial guess for the subsequent phase, i.e. the local optimization performed via the deterministic algorithm. When $\mathbf{\Omega}$ is empty, the solution provided by ERASMUS at the end of the first step is used as the starting point for the deterministic algorithm. For this second step the CNLPP reads:

$$\begin{gathered}
\min_{\boldsymbol{\xi}_{\mathrm{II}}} \psi^{(\chi)}(\boldsymbol{\xi}_{\mathrm{II}}) = \frac{1}{M} \sum_{j=1}^{M} \frac{\left[\sum_{s=1}^{n_{TP}} \left(H^{(j)}_{\boldsymbol{\xi}_{\mathrm{II}}}(\mathbf{u}_s) - Q^{(j)}_s \right)^{\chi} \right]^{(\frac{1}{\chi})}}{\left[\sum_{s=1}^{n_{TP}} \left(H^{(j)}_{\mathrm{I}}(\mathbf{u}_s) - Q^{(j)}_s \right)^{\chi} \right]^{(\frac{1}{\chi})}}, \\
\text{subject to} \\
\begin{cases}
0 < U^{(k)}_{l_k} < 1, \ l_k = p_k + 1, ..., m_k - p_k + 1, \ k = 1, ..., N, \\
U^{(k)}_{l_k} \leq U^{(k)}_{l_k+1}, \ l_k = p_k + 1, ..., m_k - p_k, \ k = 1, ..., N, \\
\omega_{i_1,...,i_N} \geq 0, \ i_k = 0, ..., n_k.
\end{cases}
\end{gathered} \tag{35}$$

In Eq. (35), $\psi^{(\chi)}$ is the χ-norm function used to approximate the maximum relative error, while $\mathbf{H}_{\mathrm{I}}$ is the hyper-surface given by the first step exploration. This approximation gives good results for $\chi \geq 20$. Obviously, the hyper-surface $\mathbf{H}_{\mathrm{I}}$, corresponding to the optimization variables given by $\boldsymbol{\xi}^{(\mathrm{opt})}_{\mathrm{I}}$ found at the end of the first step, is introduced among the initial population of this second step.

4.2. Deterministic optimization: third step

If the set $\mathbf{\Omega}$ is not empty, the pseudo-optimal solution found at the end of the second step $\boldsymbol{\xi}^{\mathrm{opt}}_{\mathrm{II}}$ is used as initial guess for the deterministic optimization phase. Also in this case, only continuous design variables are considered. The resulting CNLPP is

$$\begin{gathered}
\min_{\boldsymbol{\xi}_{\mathrm{III}}} \psi^{(\chi)}(\boldsymbol{\xi}_{\mathrm{III}}) = \frac{1}{M} \sum_{j=1}^{M} \frac{\left[\sum_{s=1}^{n_{TP}} \left(H^{(j)}_{\boldsymbol{\xi}_{\mathrm{III}}}(\mathbf{u}_s) - Q^{(j)}_s \right)^{\chi} \right]^{(\frac{1}{\chi})}}{\left[\sum_{s=1}^{n_{TP}} \left(H^{(j)}_{\mathrm{II}}(\mathbf{u}_s) - Q^{(j)}_s \right)^{\chi} \right]^{(\frac{1}{\chi})}}, \\
\text{subject to} \\
\begin{cases}
0 < U^{(k)}_{l_k} < 1, \ l_k = p_k + 1, ..., m_k - p_k + 1, \ k = 1, ..., N, \\
U^{(k)}_{l_k} \leq U^{(k)}_{l_k+1}, \ l_k = p_k + 1, ..., m_k - p_k, \ k = 1, ..., N, \\
\omega_{i_1,...,i_N} \geq 0, \ i_k = 0, ..., n_k,
\end{cases}
\end{gathered} \tag{36}$$

where $\mathbf{H}_{\text{II}}$ is the NURBS hyper-surface associated to the vector of optimization variables $\boldsymbol{\xi}_{\text{II}}^{(\text{opt})}$ given by ERASMUS at the end of the second exploration step. Note that when $\boldsymbol{\Omega} = \emptyset$, $\boldsymbol{\xi}_{\text{II}}^{(\text{opt})} = \boldsymbol{\xi}_{\text{I}}^{(\text{opt})}$ and the vector $\boldsymbol{\xi}_{\text{III}}$ collects design variables as $\boldsymbol{\xi}_{\text{II}}$ does. Solving the CNLPP (36) via a deterministic method allows finding a local minimizer from the starting point provided by the meta-heuristic exploration.

5. Numerical results

The effectiveness of the proposed metamodelling strategy is here shown by means of three test cases: the first two benchmarks belong to the field of solid mechanics, whilst the last one belongs to the field of image reduction. These benchmarks have been chosen to illustrate some interesting properties of the surrogate model based on NURBS hyper-surfaces.

The first example deals with the approximation of the displacement field of a thin plate under plane stress hypothesis. The goal is to approximate the displacement field of the plate when varying, as input data, the thickness and the force applied. Of course, the displacement field depends also upon the Cartesian coordinates x and y. In this case the metamodel is characterized by two outputs: the components of the displacement field along x and y axis, respectively. The peculiar feature of this example is that the applied load (i.e. one boundary condition) has been included among the inputs of the metamodel.

The second benchmark focuses on finding the displacement field of a thin plate with a hole: in this case the input data of the metamodel are the plate thickness, the hole radius and the Cartesian coordinates x and y. The outputs are the components of the displacement field. The interesting feature of this benchmark is related to the topology of the TPs used to formulate the hyper-surface fitting problem, which varies when changing the hole radius.

The results have been compared to those provided by the iterative procedure presented in [47, 48] by using basis function of degree 2 and empirical rules to assess the knot vector components by means of the TPs parametric coordinates:

$$\begin{gathered}
d_k = \frac{r_k + 1}{n_k - p_k + 1}, \qquad l_k = \lfloor j_k d_k \rfloor, \qquad \alpha_k = j_k d_k - l_k, \\
U^{(k)}_{p_k + j_k} = (1 - \alpha_k)\, u^{(k)}_{l_k - 1} + \alpha_k u^{(k)}_{l_k}, \\
j_k = 1, ..., n_k - p_k,\ \ k = 1, ..., N,
\end{gathered} \tag{37}$$

where $\lfloor \cdot \rfloor$ is the floor function. As far as the iterative procedure is concerned, starting from the minimal number of CPs, some CPs are added until the desired accuracy is achieved. CPs are added in the parametric direction k with the following ratio:

$$q_k = \frac{n_k}{r_k}, \quad k = 1, ..., N. \tag{38}$$

q_k is a measure of the loading pressure of CPs in the direction k, thus CPs are added in the direction with the lowest loading pressure. More details about the CPs addition/deletion strategies are available in [47, 51].

The last benchmark is taken from the literature and deals with an image reduction problem [1]. This benchmark has been solved by using both the proposed strategy and the PGD-based approach presented in [1]. The aim of this test case is to compare the results provided by the surrogate model based on NURBS hyper-surfaces to those provided by the PGD.

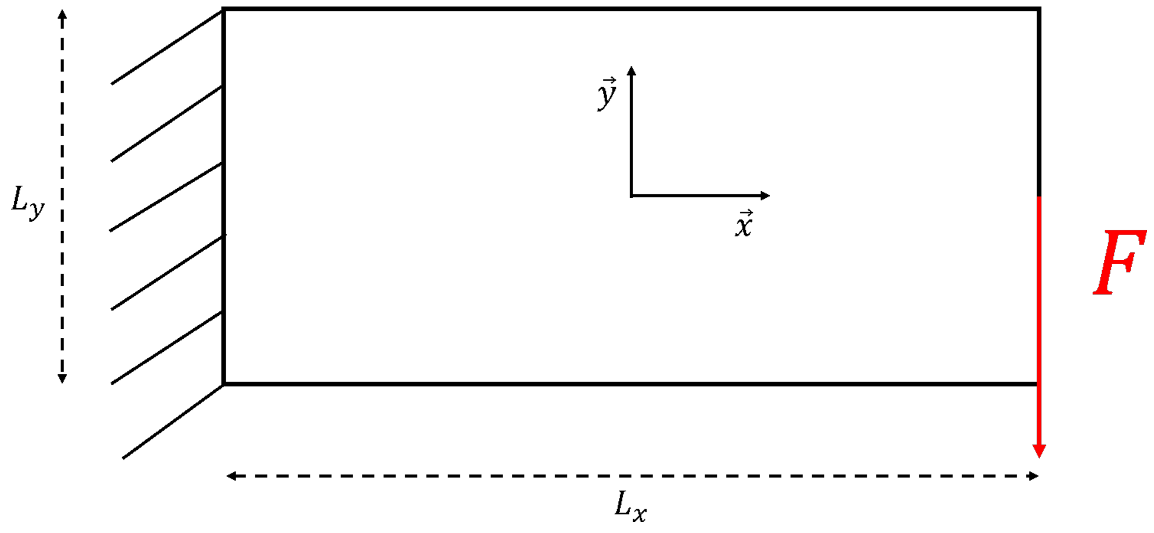


Figure 6: Geometry and boundary conditions for benchmark 1.

The geometry of the thin plate is illustrated in Fig. 6. The plate along x and y axes are $L_x = 0.2$m and $L_y = 0.1$m, respectively. The thickness t varies in the range $[t_{\min}, t_{\max}]$, with $t_{\min} = 0.001$m and $t_{\max} = 0.01$m. A force is applied along the negative direction of y axis, at location $(x, y) = \left(\frac{L_x}{2}, 0\right)$, while the plate is clamped at $x = \frac{-L_x}{2}$. The intensity of the force F varies in the interval $[F_{\min}, F_{\max}]$, with $F_{\min} = 50$N and $F_{\max} = 200$N. The metamodelling process aims at providing the displacement field of the plate for the different values of the thickness t and the force intensity F, as follows:

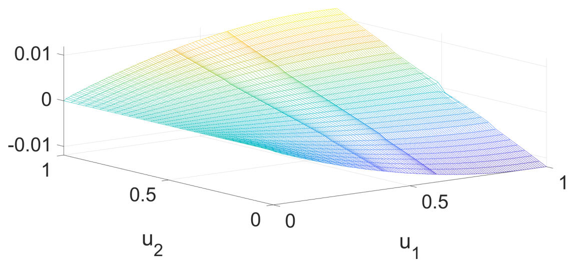
$$\hat{\mathbf{z}}(x, y, t, F) = \begin{pmatrix} \hat{u}_x(x, y, t, F) \\ \hat{u}_y(x, y, t, F) \end{pmatrix}, \tag{39}$$

where $\hat{\mathbf{z}}(x, y, t, F)$ is the surrogate model function approximating the real displacement field of the plate, while $\hat{u}_x$ and $\hat{u}_y$ are the approximated displacement components along x and y axes, respectively.

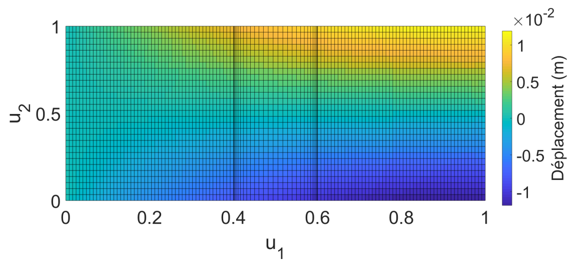
The MIMO system here is, hence, characterized by $N = 4$ inputs and $M = 2$ outputs. The TPs have been obtained through a finite element (FE) static analysis carried out via tha ANSYS code. The mesh of the FE model is composed of 126×30 PLANE182 four-node elements. Of course, a preliminary sensitivity analysis of the displacement field to the mesh size has been conducted, but it is not reported here for the sake of brevity. Fig. 7 shows the displacement field components for a thickness $t = 0.001$m and a force intensity $F = 50$N. It is noteworthy that, due to the NURBS formalism, derivatives of the NURBS hyper-surface can be easily obtained, thus strain and stress fields can be directly computed from the approximated displacement field.

The database of TPs is composed by the displacement field components u_x and u_y computed for different values of x, y, t and F and are collected in a 4-D array $\mathbf{Q}$ as follows:

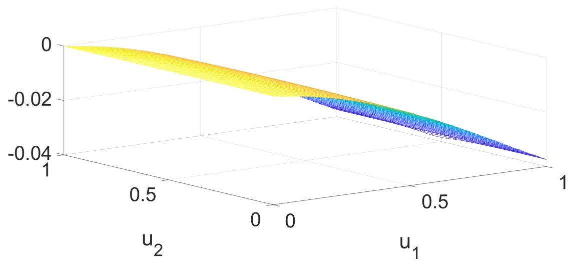
$$\mathbf{Q}_{s_1,s_2,s_3,s_4} = \left(u_x\left(x_{s_1}, y_{s_2}, t_{s_3}, F_{s_4}\right), u_y\left(x_{s_1}, y_{s_2}, t_{s_3}, F_{s_4}\right)\right), \quad s_k = 0, \ldots, r_k, \quad k = 1, ..., N. \tag{40}$$



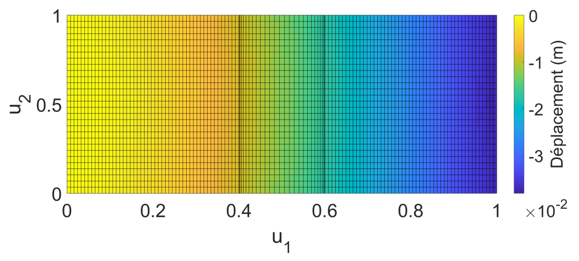
(a) Displacement field form u_x



(b) Displacement field u_x



(c) Displacement field form u_y



(d) Displacement field u_y

Figure 7: Displacement field for benchmark 1 at thickness $t = 0.001$m and force intensity $F = 10$N.

The dimensionless parameters $u^{(k)}$ are defined as:

$$u^{(1)} = \frac{x - \frac{-L_x}{2}}{\frac{L_x}{2} - \frac{-L_x}{2}}, \quad x \in \left[\frac{-L_x}{2}, \frac{L_x}{2}\right], \tag{41}$$

$$u^{(2)} = \frac{y - \frac{-L_y}{2}}{\frac{L_y}{2} - \frac{-L_y}{2}}, \quad y \in \left[\frac{-L_y}{2}, \frac{L_y}{2}\right], \tag{42}$$

Genetic parameters				
	Test case 1	Test case 2 (1st analysis)	Test case 2 (2nd analysis)	Test case 3
N. of population	3	3	3	2
Population size	100	100	200	150
N. of generations	100	100	100	100
Crossover probability	0.85	0.85	0.85	0.85
Mutation probability	0.005	0.005	0.005	0.005
Shift probability	0.5	0.5	0.5	0.5
Isolation time	5	5	5	5
Selection	roulette-wheel			
Elitism	active			

Table 1: Genetic parameters of the GA ERASMUS for test cases 1, 2 and 3.

Active-set algorithm parameters	
Maximum number of objective function evaluations	100 $\times$ number of variables
Maximum number of iterations	400
Minimum objective function improvement	10^{-6}
Minimum optimization variable change	10^{-6}
Maximum constraint violation	10^{-6}

Table 2: Optimization parameters of the *active-set* algorithm for test cases 1, 2 and 3.

$$u^{(3)} = \frac{t - t_{min}}{t_{max} - t_{min}}, \quad t \in [t_{min}, t_{max}], \tag{43}$$

$$u^{(4)} = \frac{F - F_{min}}{F_{max} - F_{min}}, \quad F \in [F_{min}, F_{max}]. \tag{44}$$

The parameters of the GA ERASMUS, used to solve problem (27), are listed in Table 1. Selection is performed by the roulette-wheel operator and optimization constraints are handled via Automatic Dynamic Penalisation (ADP) method [76]. The parameters of the *active-set* algorithm, used to solve problem (36), are listed in Table 2. The whole optimization process requires a computational time of approximately 6000s on a machine with a four cores intel i7 processor (2.9 GHz).

Variable		x	y	t	F	Overall number of points	Maximal error (TPs)		Mean error (TPs)	
$r_k + 1$ (TPs)		126	30	21	21	1 666 980	$\varepsilon_{\max}^{(u_x)}$	$\varepsilon_{\max}^{(u_y)}$	$\varepsilon_{mean}^{(u_x)}$	$\varepsilon_{mean}^{(u_y)}$
Iterative method	p_k	2	2	2	2	1 062 423	$2.36e^{-3}$	$2.05e^{-3}$	$4.04e^{-5}$	$6.91e^{-5}$
	n_k	108	26	18	18		$5.24e^{-4}$ (*fmincon*)	$8.46e^{-4}$ (*fmincon*)	$1.95e^{-5}$ (*fmincon*)	$4.25e^{-5}$ (*fmincon*)
HERO	p_k	4	4	1	1	43 500	$4.11e^{-3}$ (ERASMUS)	$3.20e^{-3}$ (ERASMUS)	$1.08e^{-4}$ (ERASMUS)	$2.09e^{-4}$ (ERASMUS)
	n_k	28	24	14	3		$2.33e^{-3}$ (*fmincon*)	$3.02e^{-3}$ (*fmincon*)	$9.42e^{-5}$ (*fmincon*)	$2.09e^{-4}$ (*fmincon*)

Table 3: Comparison of results provided by HERO and those resulting from the iterative procedure [47, 51].

The results of the NURBS hyper-surface fitting process are shown in Table 3. A maximum relative error of $\varepsilon_{\text{th}}^{(u_x)} = \varepsilon_{\text{th}}^{(u_y)} = 5e^{-3}$ has been used for both HERO and the iterative procedure in [47, 51]. It is noteworthy that, for this benchmark, the set $\mathbf{\Omega}$ was empty after the first genetic exploration, i.e all weights are equal to 1, thus a B-Spline hyper-surface is used in the subsequent deterministic optimization.

As it can be seen on Table 3, the proposed strategy is able to strongly reduce the number of CPs when compared to the iterative procedure: 43 500 when using HERO vs.

1 062 423 when the strategy presented in [51] is employed. This result is quite expected since the empirical rules used in [47, 51] was conceived to make computation faster and not for minimising the number of NURBS entity parameters. To highlight the fact that using empirical rules leads to non-optimal solutions, the same third step, i.e. problem (36) is solved by considering as a starting point the result from the iterative method.

In the case of the proposed approach, when comparing the value of the error at the starting point (which is the best solution resulting from the genetic exploration) and at the local optimum, found at the end of the process, the approximation error has not been decreased so much, showing that the starting point is really closer to the local optimum. The same consideration does not apply for the result provided by the iterative strategy which is far away from the corresponding local optimum. Even if the results of the proposed approach are better than those provided by the iterative strategy, it must be noticed that the optimized solution may not be the global optimum. In fact, when all NURBS entity parameters are integrated as design variables during the first optimization step, the resulting CNLPP is strongly non-convex and, at the end of the iterations, the genetic algorithm provides, often, multiple equivalent optimal solutions which can be used as a starting guest for the deterministic optimization (and which converges towards different minimisers). Accordingly, the solution presented in Table 3 corresponds to the best individual provided by ERASMUS at the end of the genetic exploration.

5.2. Test case 2

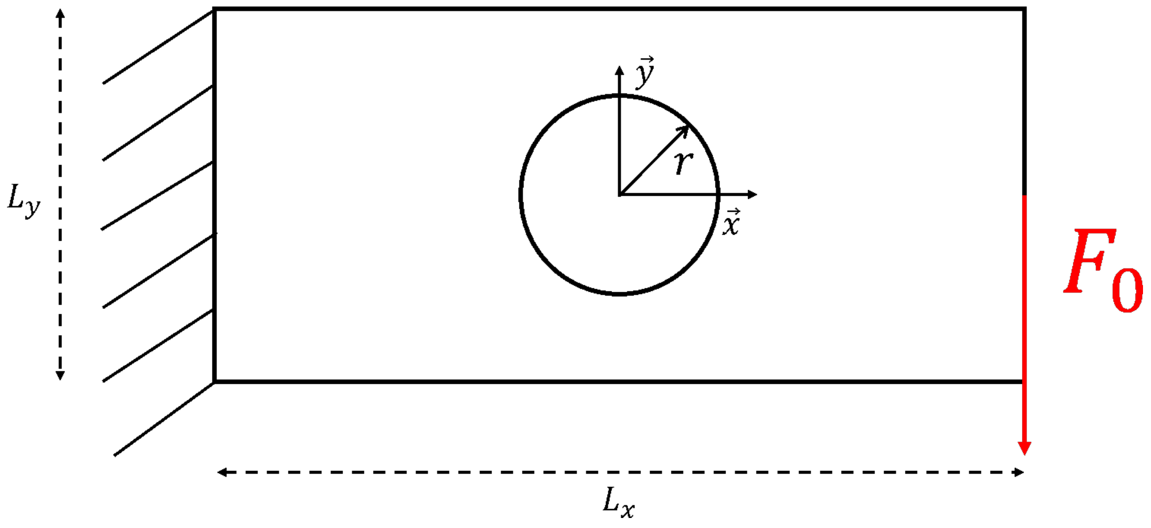
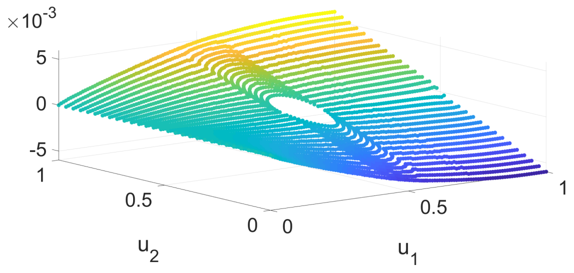


Figure 8: Geometry and boundary conditions for benchmark 2.

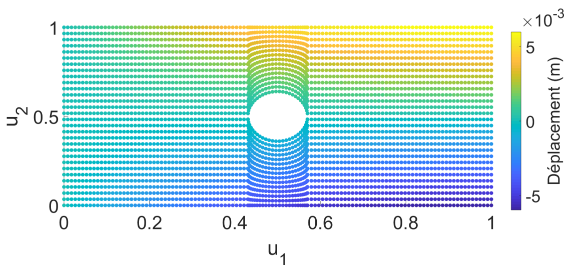
The geometry of the second benchmark is illustrated in Fig. 8. The plate is rectangular with lengths $L_x = 0.2$m and $L_y = 0.1$m along x and y axes, respectively. The thickness t varies in the interval $[t_{\min}, t_{\max}]$, with $t_{\min} = 0.001$m and $t_{\max} = 0.01$m. The plate has a hole of radius R varying in the range $[R_{\min}, R_{\max}]$, with $R_{\min} = 0.005$m and $R_{\max} = 0.025$m. A force is applied in the negative direction of y axis, at location $(x, y) = \left(\frac{L_x}{2}, 0\right)$, while the plate is clamped at $x = \frac{-L_x}{2}$. The intensity of the force is $F_0 = 100$N. The goal is to obtain a surrogate model of the displacement field of the structure for the different values of t and R:

$$\hat{\mathbf{z}}(x,y,t,R) = \begin{pmatrix} \hat{u}_x(x,y,t,R) \\ \hat{u}_y(x,y,t,R) \end{pmatrix}, \tag{45}$$

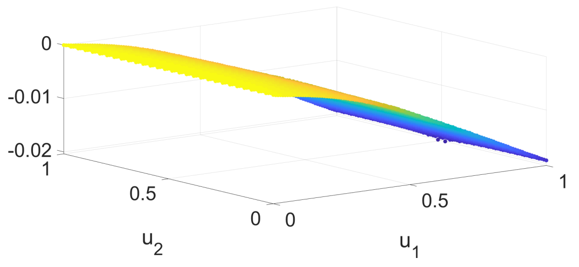
where $\hat{\mathbf{z}}(x,y,t,R)$ is the NURBS hyper-surface approximating the real displacement field of the plate, while $\hat{u}_x$ and $\hat{u}_y$ are the approximated displacement fields along x and y axes, respectively.



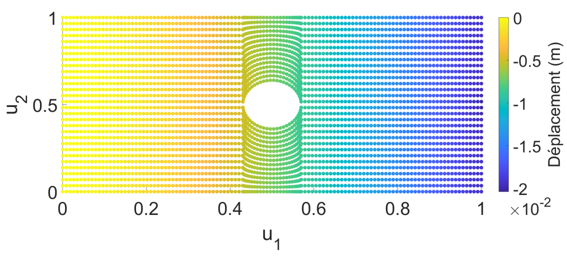
(a) Forme du champ de dépalcement u_x



(b) Champ de dépalcement u_x



(c) Forme du champ de dépalcement u_y



(d) Champ déplacement u_y

Figure 9: Displacement field for benchmark 2 at thickness $t = 0.001$m and hole radius $r = 0.005$m.

The thickness t and the radius R are design parameters, while x and y the coordinates of each point belonging to the plate. The MIMO system here is, thus, characterized by $N = 4$ inputs and $M = 2$ outputs. The TPs have been obtained by means of a static analysis carried out on a FE model made of 32400 four-node PLANE182 elements with plane stress formulation. The number of elements has been obtained after a preliminary convergence study (not reported here for the sake of brevity). Fig. 7 illustrates the displacement field for a thickness $t = 0.001$m and a hole radius $R = 0.005$m.

The set of TPs is composed by displacement field components u_x and u_y computed for different values of x, y, t and R and is stocked in the form of a 4-D array $\mathbf{Q}$:

$$\mathbf{Q}_{s_1,s_2,s_3,s_4} = \left(u_x\left(x_{s_1}, y_{s_2}, t_{s_3}, R_{s_4}\right), u_y\left(x_{s_1}, y_{s_2}, t_{s_3}, R_{s_4}\right)\right), \quad s_k = 0, \dots, r_k. \tag{46}$$

The dimensionless parameters $u^{(k)}$ are determined as follows:

$$u^{(1)} = \frac{x - \dfrac{-L_x}{2}}{\dfrac{L_x}{2} - \dfrac{-L_x}{2}}, \quad x \in \left[\frac{-L_x}{2}, \frac{L_x}{2}\right], \tag{47}$$

$$u^{(2)} = \frac{y - \dfrac{-L_y}{2}}{\dfrac{L_y}{2} - \dfrac{-L_y}{2}}, \quad y \in \left[\frac{-L_y}{2}, \frac{L_y}{2}\right], \tag{48}$$

$$u^{(3)} = \frac{t - t_{min}}{t_{max} - t_{min}}, \quad t \in [t_{min}, t_{max}], \tag{49}$$

$$u^{(4)} = \frac{R - R_{min}}{R_{max} - R_{min}}, \quad R \in [R_{min}, R_{max}]. \tag{50}$$

Variable		x	y	t	R	Overall number of points	Maximal error (TPs)		Mean error (TPs)	
$r_k + 1$ (TPs)		4 681		21	21	2 064 321	$\varepsilon^{(u_x)}_{\max}$	$\varepsilon^{(u_y)}_{\max}$	$\varepsilon^{(u_x)}_{mean}$	$\varepsilon^{(u_y)}_{mean}$
Iterative method	p_k	2	2	2	2	1 062 423	$8.27e^{-3}$	$7.54e^{-3}$	$1.41e^{-4}$	$6.84e^{-5}$
	n_k	131	27	18	18		$7.36e^{-3}$ (*fmin-con*)	$2.38e^{-3}$ (*fmin-con*)	$1.31e^{-4}$ (*fmin-con*)	$5.76e^{-5}$ (*fmin-con*)
HERO 1	p_k	1	1	4	1	12 726	$1.30e^{-2}$ (ERAS-MUS)	$8.13e^{-3}$ (ERAS-MUS)	$8.44e^{-4}$ (ERAS-MUS)	$5.12e^{-4}$ (ERAS-MUS)
	n_k	100	8	6	1		$1.10e^{-2}$ (*fmin-con*)	$8.13e^{-3}$ (*fmin-con*)	$7.99e^{-4}$ (*fmin-con*)	$5.08e^{-4}$ (*fmin-con*)
Iterative method	p_k	2	2	2	2	811 512	$1.21e^{-2}$	$2.12e^{-2}$	$2.24e^{-4}$	$9.10e^{-5}$
	n_k	116	23	16	16		$6.79e^{-3}$ (*fmin-con*)	$2.00e^{-3}$ (*fmin-con*)	$2.20e^{-4}$ (*fmin-con*)	$9.10e^{-5}$ (*fmin-con*)
HERO 2	p_k	1	1	4	1	2 736	$2.08e^{-2}$ (ERAS-MUS)	$2.45e^{-2}$ (ERAS-MUS)	$9.59e^{-4}$ (ERAS-MUS)	$8.04e^{-4}$ (ERAS-MUS)
	n_k	11	18	5	1		$2.05e^{-2}$ (*fmin-con*)	$2.14e^{-2}$ (*fmin-con*)	$9.54e^{-4}$ (*fmin-con*)	$7.31e^{-4}$ (*fmin-con*)

Table 4: Comparison of results provided by HERO and those resulting from iterative procedure [47, 51].

The parameters of the GA ERASMUS used to solve problem (28) for this second benchmark are listed in Table 1. Selection is performed by the roulette-wheel operator, whilst the ADP method is used as a constraint-handling technique [76]. The parameters of the *active-set* algorithm, used to solve problem (36), are listed in Table 2. The whole optimization process requires a computational time of approximately 6500s on a machine with a four cores intel i7 processor (2.9 GHz).

The results of the NURBS hyper-surface fitting process are shown in Table 4. A maximum relative error of $\varepsilon_{\text{th}}^{(u_x)} = \varepsilon_{\text{th}}^{(u_y)} = 3e^{-2}$ has been used for both the iterative method and the proposed approach. As in the case of benchmark 1, a B-Spline hyper-surface is sufficient to fit the displacement field components over the structure for each value of t and R. As it can be seen from Table 3, two optimal solutions are reported corresponding to different combinations of parameters tuning the behaviour of the genetic algorithm ERASMUS as reported in Table 1 (i.e. the number of individuals per population is different). Also in this case, Table 4 shows an interesting result: the number of CPs for the proposed strategy is drastically decreased when compared to the iterative methods using empiric rules. Again, the starting points for the deterministic optimization found by ERASMUS are really close to a local optimum. The same function has been used by considering as initial guess the solutions provided by the iterative procedure: the maximum relative error for solution with 1 062 423 CPs is greater than that characterizing the solution with 811 512 CPs. However, the mean relative error is still greater for the solution with 811 512.

5.3. *Test case 3*

This benchmark focuses on an image reduction problem and has been solved by means of both the surrogate model based on NURBS hyper-surfaces and PGD method [1, 6]. In this context, the goal is to minimize the number of data needed to display the original picture to save memory space, without degrading too much the quality of the image. The results of the proposed metamodelling strategy have been compared to those obtained by the PGD method. Unlike the image reduction problem presented in [1], dealing with a grayscale picture, the benchmark proposed in this study deals with the image reduction of a colour picture (courtesy of Mélissande Labadie).

For this test case, the parameter a of Eq. (28) has been set to 0.99 to increase the weight related to the minimization of the CPs number. The maximum approximation error threshold for problem (28) is:

$$\varepsilon_{\text{th}}^{(j)} = 0.35, \quad j = \text{R}, \text{G}, \text{B}, \tag{51}$$

where R, G and B represents red, green and blue outputs respectively. The parameters tuning the behaviour of the GA are reported in Table 1, whilst those governing the behaviour of the active-set algorithm are in Table 2. The interested reader is addressed to [1] for the details about the PGD approach in relation to the proposed benchmark.

An additional stopping criterion for the GA has been considered for this example. Since the genetic calculation focuses on the CPs number minimization, a threshold value on the expected compression ration has been added among the stopping criteria for the meta-heuristic exploration.

When the original picture is stored in a RAW format, three matrices are stocked (i.e. red, green and blue values), whose size is related to the numbers of pixel pix_h and pix_v along horizontal and vertical axes, respectively. The number of data needed to save the picture can be expressed as:

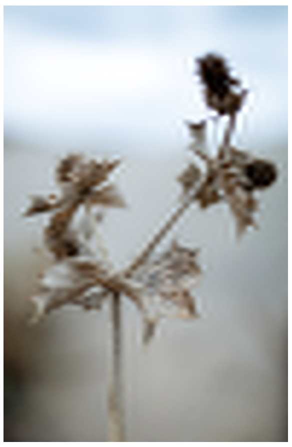
$$n_{\text{data}}^{(raw)} = 3 \times \text{pix}_h \times \text{pix}_v. \tag{52}$$

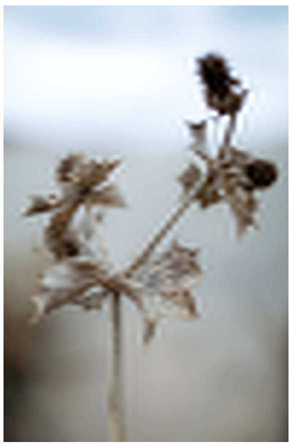
In the case of a NURBS entity, the number of data to be saved is:

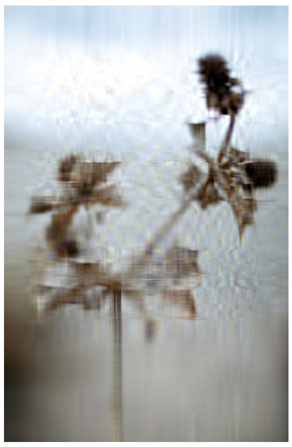
$$n_{\text{data}}^{\text{ISO}} = \begin{cases} 2 + 3n_{CP} + \sum_{k=1}^{N} (n_k - p_k), \text{ for B-Spline,} \\ 2 + 4n_{CP} + \sum_{k=1}^{N} (n_k - p_k), \text{ for NURBS.} \end{cases} \tag{53}$$

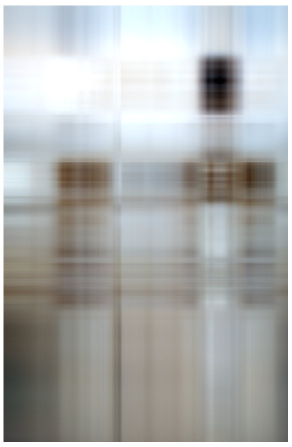
According to the above formulae, a B-Spline entity requires less data to be stored than the

(a) Original image with 2 901 176 data (pix_v = 1218, $\text{pix}_h = 794$)


(b) B-Spline with 15 601 data ($n_1 = 100$; $n_2 = 50$)


(c) NURBS with 20 752 data ($n_1 = 100$; $n_2 = 50$)


(d) PGD with 120 720 data ($N_{\text{iter}} = 20$)


(e) PGD with 18 108 data ($N_{\text{iter}} = 3$)

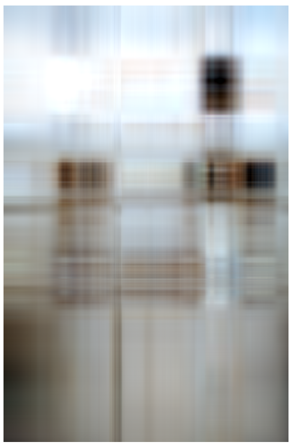

(f) PGD with 24 144 data ($N_{\text{iter}} = 4$)

Figure 10: Image reduction: comparison between PGD [1], NURBS and B-Spline results for a maximum relative error equal to 0.65.

NURBS one for a given number of CPs. Regarding the PGD, the number of data is linked to the number of enrichments N_{iter} as follows:

$$n_{\text{data}}^{(\text{PGD})} = 3 \times N_{\text{iter}} \times (\text{pix}_h + \text{pix}_v)\,. \tag{54}$$

The compression ratio can, thus, be expressed as the ratio of the number of data required by the metamodel to the number of data related to the RAW format, i.e.

$$c = \frac{n_{\text{data}}^{M}}{n_{\text{data}}^{\text{RAW}}}, \qquad M = \text{ISO}, \text{PGD}. \tag{55}$$

Fig. 10 shows the results of a compressed image for a fixed number of CPs obtained by using B-Spline (Fig. 10b) and NURBS (Fig. 10c) entities. In this comparison, the knot vectors have been computed according to Eq. (37) and only the weights of the NURBS have been optimized. Surprisingly, the approximation error related to B-Spline

Method		pix$_v$ / r_1+1	pix$_h$ / r_2+1	Number of data n_{data}	Compression ratio c (data storage)	Maximal error $\varepsilon_{\max}$	Mean error $\varepsilon_{\text{mean}}$
Original		1 218	794	2 901 276	(23.2 mo)	/	/
PGD	N_{iter}	3		18 108	$6.2e^{-3}$ (435 ko)	0.7780	0.0820
	N_{iter}	4		24 144	$8.3e^{-3}$ (579 ko)	0.7834	0.0661
	N_{iter}	20		120 720	4.2^{-2} (2.9 mo)	0.5921	0.0299
B-Spline of Fig. 10b	p_k	2	2	15 601	$5.4e^{-3}$ (374 ko)	0.6182	0.0254
	n_k	100	50				
NURBS of Fig. 10c	p_k	2	2	20 752	$7.2e^{-3}$ (498 ko)	0.6182	0.0254
	n_k	100	50				

Table 5: Approximation error comparison between PGD [1], NURBS and B-Spline results related to the images shown in Fig 10.

and NURBS metamodels were really close as it can be seen in Table 5, where maximal and mean approximation errors have been expressed as follows:

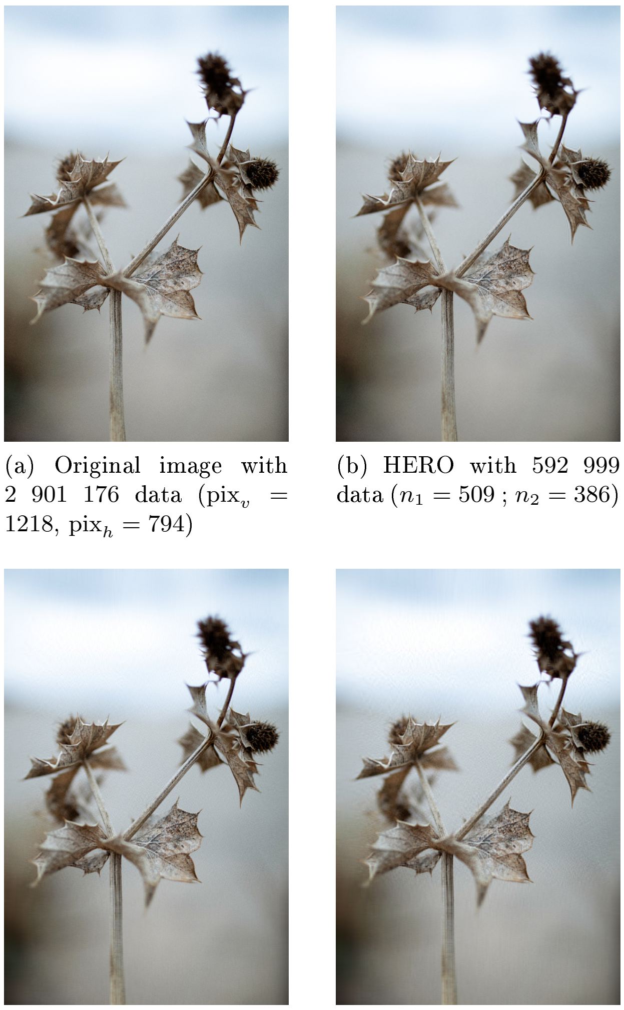
$$\varepsilon_\alpha = \max_j \varepsilon_\alpha^{(j)}, \quad \alpha = \max, \text{mean}, \quad j = \text{R}, \text{G}, \text{B}. \tag{56}$$

The results of the PGD approach, fro an equivalent number of data of both B-Spline and NURBS entities, are illustrated in Figs. 10e and 10f, respectively. As it can be inferred from these images and also from the results reported in Table 5, PGD results are less accurate than the NURBS-based metamodelling strategy for an equivalent number of data. Fig. 10d shows the reduced image obtained by means of the PGD, which is as accurate as image 10b. However, this image requires an amount of data significantly greater (about seven times) than that required for the B-Spline solution illustrated in Fig. 10b. Moreover, although the maximal approximation error of the PGD solution of Fig. 10d is slightly lower than that of the B-Spline solution of Fig. 10b, the B-Spline image look "prettier" than the PGD one. This visual aspect is due to the error repartition on the three fundamental colours (Red, Green and Blue) which is better in the case of the B-Spline solution. The results listed in table 5 also show that including weights among the optimisation variables gives a negligible improvement in the approximation error.

Method		pix$_v$ / r_1+1	pix$_h$ / r_2+1	Number of data n_{data}	Compression ratio c (data storage)	Maximal error $\varepsilon_{\max}$	Mean error $\varepsilon_{\text{mean}}$
Original		1 218	794	2 901 276	(23.2 mo)	/	/
PGD	N_{iter}	99		597 564	0.21 (4.8 mo)	0.4392	0.0139
	N_{iter}	139		839 004	0.29 (6.7 mo)	0.3473	0.0120
HERO (B-Spline)	p_k	6	6	592 999	0.20 (4.7 mo)	0.3486	0.0117
	n_k	509	386				

Table 6: Approximation error comparison between the results provided by the PGD [1] and HERO of Fig. 11.

Accordingly, only B-Spline entities have been considered in the HERO process. The results are provided in Table 6, whilst the corresponding images are presented in Fig. 11.



(a) Original image with 2 901 176 data ($\text{pix}_v = 1218$, $\text{pix}_h = 794$)

(b) HERO with 592 999 data ($n_1 = 509$; $n_2 = 386$)

(c) PGD with 839 004 data ($N_{\text{iter}} = 139$)

(d) PGD with 597 564 data ($N_{\text{iter}} = 99$)

Figure 11: Image reduction: comparison between PGD [1] and HERO results.

When looking at Fig. 11, it is rather difficult to distinguish between the original image and that provided by the surrogate modelling strategy based on NURBS hyper-surfaces when a compression ratio $c = 0.20$ is set. However, Table 6 shows that the maximal approximation error is relatively high compared to the minimal accuracy needed in a no loss case. Indeed, since RGB matrices are composed of integers varying between 0 and 255 (i.e. 8 bits per colour), the metamodel outputs are rounded to the nearest integer (for both PGD and HERO). As a result, the minimal accuracy giving negligible loss of picture quality is:

$$\varepsilon_{\text{th}}^{(\text{loss})} = \frac{0.5}{255} = 0.02, \tag{57}$$

where $\varepsilon_{\text{th}}^{(\text{loss})}$ is the maximal approximation error threshold that prevents from degrading the original picture. As it can been inferred from Table 6, the maximal approximation error related to both B-Spline and PGD solutions is higher than this value. However, the mean error is twice lower than this threshold and this explains why there are no "visible" differences between original and reconstructed picture.

The solution based on the B-Spline hyper-surfaces is compared to the results provided by the PGD, at equivalent compression ratio (Fig. 11d) and at equivalent maximal approximation error (Fig. 11c). It is noteworthy that the solution resulting from the PGD, at equivalent compression ratio, has a lower "quality" than that provided by HERO. Table 6 shows that the maximal approximation error is, in this case, significantly higher. In addition, to reach the accuracy of the B-Spline-based metamodel, the number of data needed by the PGD is 1.41 times higher. It is noteworthy that mean approximation error related to the PGD solutions is still higher than that related to the B-Spline-based metamodel.

6. Conclusions

A new metamodelling technique based on NURBS hyper-surfaces together with a hybrid optimization strategy has been presented in this paper. The proposed approach is very general: nor simplifying hypotheses neither empirical rules are utilised to select *a priori* some parameters of the metamodel. Conversely, the number of parameters and their values are automatically determined by the hybrid optimization strategy (and according to user's defined accuracy), making this approach problem-independent. Moreover, the metamodel obtained aims at being as "light" as possible since it automatically determines if rather B-Spline or NURBS hyper-surfaces are needed to fit a given set of data points and which and where (i.e. on the control hyper-net) weights have to be added during the optimization process. This result is achieved thanks to the special genetic algorithm ERASMUS able to deal with optimization problems characterized by a variable number of design variables. The effectiveness of this process has been shown through two meaningful examples and the results have been compared to those provided by an iterative procedure available in the literature. In addition, a benchmark taken from the literature has been solved by the proposed approach and the results provided has been compared to those of the PGD method.

The use of NURBS hyper-surfaces as surrogate models has revealed to be really effective in handling boundary conditions among the input parameters of the surrogate model. Moreover, the proposed approach has shown its ability to fit non-convex sets of data where the TPs space topology varies with metamodel inputs. In both cases, the obtained results have been compared to those resulting from an iterative method taken from literature. From these comparisons it seems evident that the use of empirical rules in setting some of the parameters of the NURBS entity are not adequate for minimising the metamodel resources (i.e. number of CPs and degrees) by ensuring, simultaneously, the required accuracy.

When compared to the PGD method on a benchmark taken from the literature, dealing with an image reduction problem, the proposed approach is more efficient in terms of either compression ratio or accuracy. Moreover, for this test case, including weights into the optimization process has a negligible effect.

Of course, the proposed methodology constitutes just a first attempt. One limitation is that the set of TPs needs to be sorted to use the algorithms presented in Section 3 for the computation of the CPs coordinate. When dealing with a high number of TPs, the matrix to be stored before inversion could reach terabytes of memory. The inversion in this case is anything but trivial. This problem has been addressed in the literature by adding a sorting step such as natural neighbour used in [47]. However, this strategy seems to be quite limiting. To this purpose, research is ongoing in order to develop suitable mapping techniques avoiding the use of sorted set of TPs. In this way, also unordered data can be effectively handled by the proposed approach.

Furthermore, linking NURBS-based methods with kriging could be an interesting prospect. In fact, Kriging is based upon the study of the spatial dependency of the TPs. As a result, an estimation of the approximation error is provided together with the function evaluation. Coupling Kriging to the NURBS-based metamodelling strategy could be very useful for a better assessment of the approximation error and also to exploit the local support property of the NURBS blending function to determine the correlation among inputs.

A further interesting prospect concerns the application of the proposed metamodelling strategy to eigenvalue problems and nonlinear analyses. Due to the complexity of the topology of the TPs space, in these cases some smoothing terms should be added to the problem formulation in order to avoid over-fitting.

References

[1] F. Chinesta, R. Keunings, A. Leygue, The Proper Generalized Decomposition for Advanced Numerical Simulations, SpringerBriefs in Applied Sciences and Technology, Springer International Publishing, Cham, 2014.
URL http://link.springer.com/10.1007/978-3-319-02865-1

[2] G. G. Wang, S. Shan, Review of Metamodeling Techniques in Support of Engineering Design Optimization, Journal of Mechanical Design 129 (4) (2007) 370–380. doi:10.1115/1.2429697.
URL http://mechanicaldesign.asmedigitalcollection.asme.org/article.aspx?articleid=1449318

[3] C. Bisagni, L. Lanzi, Post-buckling optimisation of composite stiffened panels using neural networks, Composite Structures 58 (2) (2002) 237–247. doi:10.1016/S0263-8223(02)00053-3.
URL http://www.sciencedirect.com/science/article/pii/S0263822302000533

[4] F. Chinesta, A. Ammar, E. Cueto, Recent Advances and New Challenges in the Use of the Proper Generalized Decomposition for Solving Multidimensional Models, Archives of Computational Methods in Engineering 17 (4) (2010) 327–350. doi:10.1007/s11831-010-9049-y.
URL https://link.springer.com/article/10.1007/s11831-010-9049-y

[5] F. Chinesta, A. Leygue, F. Bordeu, J. V. Aguado, E. Cueto, D. Gonzalez, I. Alfaro, A. Ammar, A. Huerta, PGD-Based Computational Vademecum for Efficient Design, Optimization and Control, Archives of Computational Methods in Engineering 20 (1) (2013) 31–59. doi:10.1007/s11831-013-9080-x.
URL http://link.springer.com/10.1007/s11831-013-9080-x

[6] F. Chinesta, E. Cueto, PGD-Based Modeling of Materials, Structures and Processes, Springer Science & Business, 2014.

[7] A. Ammar, M. Normandin, F. Chinesta, Solving parametric complex fluids models in rheometric flows, Journal of Non-Newtonian Fluid Mechanics 165 (23–24) (2010) 1588–1601. doi:10.1016/j.jnnfm.2010.08.006.
URL http://www.sciencedirect.com/science/article/pii/S037702571000217X

[8] O. Allix, P. Ladevèze, D. Gilletta, R. Ohayon, A damage prediction method for composite structures, International Journal for Numerical Methods in Engineering 27 (2) (1989) 271–283. doi:10.1002/nme.1620270205.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.1620270205

[9] X. Aubard, C. Cluzel, L. Guitard, P. Ladevèze, Damage modeling at two scales for 4d carbon/carbon composites, Computers & Structures 78 (1-3) (2000) 83–91. doi:10.1016/S0045-7949(00)00101-2.
URL https://www.sciencedirect.com/science/article/pii/S0045794900001012

[10] P. Ladevèze, L. Guitard, L. Champaney, X. Aubard, Debond modeling for multidirectional composites, Computer Methods in Applied Mechanics and Engineering 185 (2-4) (2000) 109–122. doi:10.1016/S0045-7825(99)00254-6.
URL https://www.sciencedirect.com/science/article/pii/S0045782599002546

[11] P. Ladevèze, Multiscale modelling and computational strategies for composites, International Journal for Numerical Methods in Engineering 60 (1) (2004) 233–253. doi:10.1002/nme.960.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.960

[12] D. Violeau, P. Ladevèze, G. Lubineau, Micromodel-based simulations for laminated composites, Composites Science and Technology 69 (9) (2009) 1364–1371. doi:10.1016/j.compscitech.2008.09.041.
URL https://www.sciencedirect.com/science/article/pii/S0266353808003758

[13] A. Ammar, A. Huerta, F. Chinesta, E. Cueto, A. Leygue, Parametric solutions involving geometry: A step towards efficient shape optimization, Computer Methods in Applied Mechanics and Engineering 268 (2014) 178–193. doi:10.1016/j.cma.2013.09.003.
URL //www.sciencedirect.com/science/article/pii/S0045782513002284

[14] A. Leygue, E. Verron, A First Step Towards the Use of Proper General Decomposition Method for Structural Optimization, Archives of Computational Methods in Engineering 17 (4) (2010) 465–472. doi:10.1007/s11831-010-9052-3.
URL http://link.springer.com/10.1007/s11831-010-9052-3

[15] F. Chinesta, A. Ammar, E. Cueto, On the use of proper generalized decompositions for solving the multidimensional chemical master equation, European Journal of Computational Mechanics 19 (1-3) (2010) 53–64. doi:10.3166/ejcm.19.53-64.
URL https://doi.org/10.3166/ejcm.19.53-64

[16] G. Berkooz, P. Holmes, J. L. Lumley, The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows, Annual Review of Fluid Mechanics 25 (1) (1993) 539–575. doi:10.1146/annurev.fl.25.010193.002543.
URL https://doi.org/10.1146/annurev.fl.25.010193.002543

[17] A. Chatterjee, An introduction to the proper orthogonal decomposition, Current Science 78 (7) (2000) 808–817.
URL http://www.jstor.org/stable/24103957

[18] Y. Zhao, W. Lu, C. Xiao, A Kriging surrogate model coupled in simulation–optimization approach for identifying release history of groundwater sources, Journal of Contaminant Hydrology 185-186 (2016) 51–60. doi:10.1016/j.jconhyd.2016.01.004.
URL http://www.sciencedirect.com/science/article/pii/S0169772216300043

[19] D. E. Myers, Spatial interpolation: an overview, Geoderma 62 (1) (1994) 17–28. doi:10.1016/0016-7061(94)90025-6.
URL http://www.sciencedirect.com/science/article/pii/0016706194900256

[20] J. D. Martin, T. W. Simpson, On the Use of Kriging Models to Approximate Deterministic Computer Models, Vol. 2004, ASME, 2004, pp. 481–492. doi:10.1115/DETC2004-57300.
URL http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?articleid=1651237

[21] E. Iuliano, D. Quagliarella, Proper Orthogonal Decomposition, surrogate modelling and evolutionary optimization in aerodynamic design, Computers & Fluids 84 (2013) 327–350. doi:10.1016/j.compfluid.2013.06.007.
URL http://www.sciencedirect.com/science/article/pii/S0045793013002223

[22] K. Kunisch, S. Volkwein, Galerkin Proper Orthogonal Decomposition Methods for a General Equation in Fluid Dynamics, SIAM Journal on Numerical Analysis 40 (2) (2002) 492–515. doi:10.1137/S0036142900382612.
URL https://epubs.siam.org/doi/abs/10.1137/S0036142900382612

[23] C. W. Rowley, Model reduction for fluids, using balanced proper orthogonal decomposition, International Journal of Bifurcation and Chaos 15 (03) (2005) 997–1013. doi:10.1142/S0218127405012429.
URL https://www.worldscientific.com/doi/abs/10.1142/S0218127405012429

[24] N. Aubry, P. Holmes, J. L. Lumley, E. Stone, The dynamics of coherent structures in the wall region of a turbulent boundary layer, Journal of Fluid Mechanics 192 (1988) 115–173. doi:10.1017/S0022112088001818.
URL https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/dynamics-of-coherent-structures-in-the-wall-region-of-a-turbulent-boundary-
717E2487A5A758FF9321242328FB8FAC

[25] P. Holmes, J. L. Lumley, G. Berkooz, C. W. Rowley, Turbulence, Coherent Structures, Dynamical Systems and Symmetry, Cambridge University Press, 2012.

[26] B. R. Noack, K. Afanasiev, M. Morzyński, G. Tadmor, F. Thiele, A hierarchy of low-dimensional models for the transient and post-transient cylinder wake, Journal of Fluid Mechanics 497 (2003) 335–363. doi:10.1017/S0022112003006694.
URL https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/hierarchy-of-lowdimensional-models-for-the-transient-and-posttransient-cyli
0F114BEB5DD20B7342E99ED8D0070C01

[27] D. Rempfer, H. F. Fasel, Dynamics of three-dimensional coherent structures in a flat-plate boundary layer, Journal of Fluid Mechanics 275 (1994) 257–283. doi:10.1017/S0022112094002351.
URL https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/dynamics-of-threedimensional-coherent-structures-in-a-flatplate-boundary-la
80694CB73540252999AA023E87415736

[28] M. J. Mifsud, S. T. Shaw, D. G. MacManus, A high-fidelity low-cost aerodynamic model using proper orthogonal decomposition, International Journal for Numerical Methods in Fluids 63 (4) (2009) 468–494. doi:10.1002/fld.2085.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.2085

[29] H. T. Banks, M. L. Joyner, B. Wincheski, W. P. Winfree, Nondestructive evaluation using a reduced-order computational methodology, Inverse Problems 16 (4) (2000)

929. doi:10.1088/0266-5611/16/4/304.
URL http://stacks.iop.org/0266-5611/16/i=4/a=304

[30] S. U. Hamim, R. P. Singh, Proper Orthogonal Decomposition–Radial Basis Function Surrogate Model-Based Inverse Analysis for Identifying Nonlinear Burgers Model Parameters From Nanoindentation Data, Journal of Engineering Materials and Technology 139 (4) (2017) 041010–041010–8. doi:10.1115/1.4037022.
URL http://dx.doi.org/10.1115/1.4037022

[31] K. Fukunaga, Introduction to Statistical Pattern Recognition, Elsevier, 2013.

[32] D. G. Krige, A statistical approach to some basic mine valuation problems on the Witwatersrand, Journal of the Southern African Institute of Mining and Metallurgy 52 (6) (1951) 119–139.
URL https://journals.co.za/content/saimm/52/6/AJA0038223X_4792

[33] G. Matheron, Le krigeage universel (1969).

[34] G. Matheron, Le krigeage disjonctif, Tech. Rep. N-360, Fontainebleau (Dec. 1973).

[35] J. P. C. Kleijnen, Kriging metamodeling in simulation: A review, European Journal of Operational Research 192 (3) (2009) 707–716. doi:10.1016/j.ejor.2007.10.013.
URL http://www.sciencedirect.com/science/article/pii/S0377221707010090

[36] R. Bettinger, Inversion d'un système par krigeage: application à la synthèse des catalyseurs à haut débit, Ph.D. thesis, Université de Nice Sophia Antipolis (2009).
URL https://tel.archives-ouvertes.fr/tel-00460162/

[37] M. J. Sasena, Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations (2002) 237.

[38] M. D. Buhmann, Radial basis functions, Acta Numerica 2000 9 (2000) 1–38.
URL http://journals.cambridge.org/abstract_S0962492900000015

[39] E. Viennet, Réseaux à fonctions de base radiales, Apprentissage connexionniste (2006) 105.
URL https://hal.archives-ouvertes.fr/hal-00085092/

[40] B. YEGNANARAYANA, ARTIFICIAL NEURAL NETWORKS, PHI Learning Pvt. Ltd., 2009.

[41] V. Capecchi, M. Buscema, P. Contucci, B. D'Amore (Eds.), Applications of Mathematics in Models, Artificial Neural Networks and Arts, Springer Netherlands, Dordrecht, 2010.
URL http://link.springer.com/10.1007/978-90-481-8581-8

[42] J. Rayas-Sanchez, EM-Based Optimization of Microwave Circuits Using Artificial Neural Networks: The State-of-the-Art, IEEE Transactions on Microwave Theory and Techniques 52 (1) (2004) 420–435. doi:10.1109/TMTT.2003.820897.
URL http://ieeexplore.ieee.org/document/1262733/

[43] J. Sreekanth, B. Datta, Multi-objective management of saltwater intrusion in coastal aquifers using genetic programming and modular neural network based surrogate models, Journal of Hydrology 393 (3) (2010) 245–256. doi:10.1016/j.jhydrol.2010.08.

023.
URL http://www.sciencedirect.com/science/article/pii/S0022169410005408

[44] H. Shi, Y. Gao, X. Wang, Optimization of injection molding process parameters using integrated artificial neural network model and expected improvement function method, The International Journal of Advanced Manufacturing Technology 48 (9-12) (2010) 955–962. doi:10.1007/s00170-009-2346-7.
URL https://link.springer.com/article/10.1007/s00170-009-2346-7

[45] S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd édition, Prentice-Hall, New Jerey, 1999.

[46] R. H. Myers, D. C. Montgomery, C. M. Anderson-Cook, Response Surface Methodology: Process and Product Optimization Using Designed Experiments, John Wiley & Sons, 2009.

[47] C. J. Turner, HyPerModels: hyperdimensional performance models for engineering design, Ph.D. thesis (2005).
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.392.1513&rep=rep1&type=pdf

[48] C. J. Turner, R. H. Crawford, N-Dimensional Nonuniform Rational B-Splines for Metamodeling, Journal of Computing and Information Science in Engineering 9 (3) (2009) 031002. doi:10.1115/1.3184599.
URL http://ComputingEngineering.asmedigitalcollection.asme.org/article.aspx?articleid=1401568

[49] J. Steuben, J. Michopoulos, A. Iliopoulos, C. Turner, Inverse characterization of composite materials via surrogate modeling, Composite Structures 132 (2015) 694–708. doi:10.1016/j.compstruct.2015.05.029.
URL http://www.sciencedirect.com/science/article/pii/S0263822315003943

[50] G. Farin, Curves and Surfaces for CAGD, Elsevier, 2002. doi:10.1016/B978-1-55860-737-8.X5000-5.
URL http://linkinghub.elsevier.com/retrieve/pii/B9781558607378X50005

[51] L. Piegl, W. Tiller, The NURBS Book, Monograph in Visual Communication, Springer-Verlag Berlin Heidelberg, 1995.
URL 10.1007/978-3-642-97385-7

[52] X. Qian, Full analytical sensitivities in NURBS based isogeometric shape optimization 199 (29) 2059–2071.
URL http://www.sciencedirect.com/science/article/pii/S0045782510000812

[53] W. A. Wall, M. A. Frenzel, C. Cyron, Isogeometric structural shape optimization 197 (33) 2976–2988.
URL http://www.sciencedirect.com/science/article/pii/S0045782508000509

[54] J. Kiendl, R. Schmidt, R. Wüchner, K. U. Bletzinger, Isogeometric shape optimization of shells using semi-analytical sensitivity analysis and sensitivity weighting 274 148–167.
URL http://www.sciencedirect.com/science/article/pii/S0045782514000486

[55] L. Chamoin, H. P. Thai, Certified real-time shape optimization using isogeometric analysis, PGD model reduction, and a posteriori error estimation 119 (3) 151–176.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.6045

[56] G. Costa, M. Montemurro, J. Pailhès, A 2D topology optimisation algorithm in NURBS framework with geometric constraints, International Journal of Mechanics and Materials in Design (2017) 1–28doi:10.1007/s10999-017-9396-z.
URL https://link.springer.com/article/10.1007/s10999-017-9396-z

[57] G. Costa, M. Montemurro, J. Pailhès, A General Hybrid Optimization Strategy for Curve Fitting in the Non-Uniform Rational Basis Spline Framework, Journal of Optimization Theory and Applications (2017) 1–27doi:10.1007/s10957-017-1192-2.
URL https://link.springer.com/article/10.1007/s10957-017-1192-2

[58] M. Montemurro, A. Catapano, A New Paradigm for the Optimum Design of Variable Angle Tow Laminates, in: A. Frediani, B. Mohammadi, O. Pironneau, V. Cipolla (Eds.), Variational Analysis and Aerospace Engineering: Mathematical Challenges for the Aerospace of the Future, Springer Optimization and Its Applications, Springer International Publishing, Cham, 2016, pp. 375–400.
URL https://doi.org/10.1007/978-3-319-45680-5_14

[59] M. Montemurro, A. Catapano, On the effective integration of manufacturability constraints within the multi-scale methodology for designing variable angle-tow laminates, Composite Structures 161 (2017) 145–159. doi:10.1016/j.compstruct.2016.11.018.
URL http://www.sciencedirect.com/science/article/pii/S0263822316320712

[60] M. Montemurro, A. Catapano, A general B-Spline surfaces theoretical framework for optimisation of variable angle-tow laminates, Composite Structures 209 (2018) 561–578. doi:10.1016/j.compstruct.2018.10.094.
URL http://www.sciencedirect.com/science/article/pii/S0263822318324334

[61] G. Costa, M. Montemurro, J. Pailhès, NURBS hyper-surfaces for 3D topology optimization problems, Mechanics of Advanced Materials and Structures (2019) 1–20doi:10.1080/15376494.2019.1582826.
URL https://doi.org/10.1080/15376494.2019.1582826

[62] G. Costa, M. Montemurro, J. Pailhes, N. Perry, Maximum length scale requirement in a topology optimisation method based on NURBS hyper-surfaces, CIRP Annals - Manufacturing Technology 68 (1) (2019) 153–156.
URL https://hal.archives-ouvertes.fr/hal-02277399

[63] G. Costa, M. Montemurro, J. Pailhès, Minimum length scale control in a NURBS-based SIMP method, Computer Methods in Applied Mechanics and Engineering 354 (2019) 963–989. doi:10.1016/j.cma.2019.05.026.
URL http://www.sciencedirect.com/science/article/pii/S0045782519302968

[64] M. Montemurro, A contribution to the development of design strategies for the optimisation of lightweight structures, Habilitation à diriger des recherches, Université de Bordeaux, France (2018).
URL http://hdl.handle.net/10985/15155

[65] M. Montemurro, Optimal design of advanced engineering modular systems through a new genetic approach., Ph.D. thesis, Université Pierre et Marie Curie Paris VI, Paris (2012).
URL https://tel.archives-ouvertes.fr/tel-00955533

[66] P. Bézier, Courbes et surfaces, Hermès, Paris; Londres, 1986.

[67] C. de Boor, A Practical Guide to Splines., Mathematics of Computation 34 (149) (1980) 325. doi:10.2307/2006241.
URL https://www.jstor.org/stable/2006241?origin=crossref

[68] T. Rodriguez, M. Montemurro, P. Le Texier, J. Pailhès, Structural displacement requirement in a topology optimization algorithm based on isogeometric entities, Journal of Optimization Theory and Applications 184 (1) (2020) 250–276.
URL https://doi.org/10.1007/s10957-019-01622-8

[69] Optimization Toolbox User's Guide, Tech. rep., The Mathworks Inc., 3 Apple Hill Drive, Natick (2018).

[70] L. Cappelli, G. Balokas, M. Montemurro, F. Dau, L. Guillaumat, Multi-scale identification of the elastic properties variability for composite materials through a hybrid optimisation strategy, Composites Part B: Engineering 176 (2019) 107193. doi:10.1016/j.compositesb.2019.107193.
URL http://www.sciencedirect.com/science/article/pii/S1359836819321766

[71] G. Bertolino, M. Montemurro, G. De Pasquale, Multi-scale shape optimisation of lattice structures: an evolutionary-based approach, International Journal on Interactive Design and Manufacturing (IJIDeM) (May 2019). doi:10.1007/s12008-019-00580-9.
URL https://doi.org/10.1007/s12008-019-00580-9

[72] M. Montemurro, M. I. Izzi, J. El-Yagoubi, D. Fanteria, Least-weight composite plates with unconventional stacking sequences: Design, analysis and experiments, Journal of Composite Materials 53 (16) (2019) 2209–2227. doi:10.1177/0021998318824783.
URL https://doi.org/10.1177/0021998318824783

[73] E. Panettieri, M. Montemurro, A. Catapano, Blending constraints for composite laminates in polar parameters space, Composites Part B: Engineering 168 (2019) 448–457. doi:10.1016/j.compositesb.2019.03.040.
URL http://www.sciencedirect.com/science/article/pii/S1359836819304494

[74] L. Cappelli, M. Montemurro, F. Dau, L. Guillaumat, Characterisation of composite elastic properties by means of a multi-scale two-level inverse approach, Composite Structures 204 (2018) 767–777. doi:10.1016/j.compstruct.2018.08.007.
URL http://www.sciencedirect.com/science/article/pii/S0263822318318403

[75] M. Montemurro, A. Pagani, G. A. Fiordilino, J. Pailhès, E. Carrera, A general multi-scale two-level optimisation strategy for designing composite stiffened panels, Composite Structures 201 (2018) 968–979. doi:10.1016/j.compstruct.2018.06.119.
URL http://www.sciencedirect.com/science/article/pii/S0263822318311036

[76] M. Montemurro, A. Vincenti, P. Vannucci, The Automatic Dynamic Penalisation method (ADP) for handling constraints with genetic algorithms, Computer Methods

in Applied Mechanics and Engineering 256 (2013) 70–87. doi:10.1016/j.cma.2012.12.009.
URL http://www.sciencedirect.com/science/article/pii/S0045782512003799