An overview of gradient-enhanced metamodels with applications

Luc Laurent, Rodolphe Le Riche, Bruno Soulier, Pierre-Alain Boucard

An overview of gradient-enhanced metamodels with applications

Luc LAURENT[a]
Rodolphe LE RICHE[b,c]
Bruno SOULIER[d]
Pierre-Alain BOUCARD[d]

[a] Laboratoire de Mécanique des Structures et des Systèmes Couplés, Conservatoire National des Arts et Métiers, case courrier 2D6R10, 2 rue Conté, 75003 Paris, France
Tel.: +33-1-58-80-85-80
luc.laurent@lecnam.net

[b] CNRS LIMOS UMR 6158, France
[c] École Nationale Supérieure des Mines de Saint–Étienne, 158, cours Fauriel, F-42023 Saint-Étienne, France
leriche@emse.fr

[d] LMT Cachan (ENS Cachan/CNRS/Université Paris-Saclay), 61, avenue du Président Wilson, 94235 Cachan, France
soulier@lmt.ens-cachan.fr, boucard@lmt.ens-cachan.fr

Abstract

Metamodeling, the science of modeling functions observed at a finite number of points, benefits from all auxiliary information it can account for. Function gradients are a common auxiliary information and are useful for predicting functions with locally changing behaviors. This article is a review of the main metamodels that use function gradients in addition to function values. The goal of the article is to give the reader both an overview of the principles involved in gradient-enhanced metamodels while also providing insightful formulations. The following metamodels have gradient-enhanced versions in the literature and are reviewed here: classical, weighted and moving least squares, Shepard weighting functions, and the kernel-based methods that are radial basis functions, kriging and support vector machines. The methods are set in a common framework of linear combinations between a priori chosen functions and coefficients that depend on the observations. The characteristics common to all kernel-based approaches are underlined. A new ν-GSVR metamodel which uses gradients is given. Numerical comparisons of the metamodels are carried out for approximating analytical test functions. The experiments are replicable, as they are performed with an opensource available toolbox. The results indicate that there is a trade-off between the better computing time of least squares methods and the larger versatility of kernel-based approaches.

Keywords: gradient-enhanced – surrogate model – metamodel – cokriging

Luc Laurent, Rodolphe Le Riche, Bruno Soulier, and Pierre-Alain Boucard. An overview of gradient-enhanced metamodels with applications. Archives of Computational Methods in Engineering, *26(1):61–106, Jan 2019, First online: Jul 2017. doi: 10.1007/s11831-017-9226-3, hal: emse-01525674*

Contents

1 Introduction

Despite continuous progress in the accuracy of experimental measurements and numerical simulations of the physics of a considered system, the need for metamodels keeps increasing. Metamodels are statistical or functional models of input-ouput data that are obtained either from experimental measures or from the numerical simulation of the associated physical phenomena. Metamodels are sometimes called surrogates, proxies, regression functions, approximating functions, supervised machine learners or are referred to with specific names such as the ones described later in this article. Although not directly linked to the physics, metamodels have proven to be necessary for creating simple, computationally efficient associations between the input and output of the considered phenomena. For example, in materials sciences input may be material properties or boundary conditions and outputs are displacements, forces, temperatures, concentrations or other quantities at specific locations; in design, inputs may be the parameters describing a shape or a material and outputs specific measures of performance such as mass, strength, stiffness; in geophysics inputs may be parameterized descriptions of the underground (permeabilities, faults, reservoir shapes) and the outputs quantities observed at the surface (flow rates, displacements, accelerations, gravity). Typically, actual or numerical experiments are costly in terms of time or other resources, in which case metamodels are a key technology to perform optimization, parameter identification and uncertainty propagation.

Infering nonlinear relationships requires large amount of data particularly when the number of input parameters grows (the "curse of dimensionality" [1]) so that it is important to use all available additional information. Gradients, i.e., derivatives of the outputs with respect to the inputs, are one of the most common and most useful side knowledge to be accounted for when building the metamodels: many finite elements codes have implemented adjoint methods to calculate gradients [2–4]; automatic differentiation is another solution for computing gradients [5–7]; there are responses such as volumes for which analytical gradient calculation is accessible.

Accounting for gradients when building metamodels often allows to decrease the necessary number of data points to achieve a given metamodel accuracy, or equivalently, it allows to increase the metamodel accuracy at a given number of data points. When guessing a function with a locally changing behavior (a non stationary process in the probability terminology) from a sparse set of observations, the traditional regression techniques relying only on the function values will tend to damp out local fluctuations. This is because useful regression methods comprise regularization strategies that make them robust to small perturbations in the data. Accounting for gradients is a way to recover some of the meaningful local fluctuations. The need for gradient information has been acknowledged in geophysics for reconstructing a gravity field from stations measurements when the underground is subject to local changes [8, 9]. Further illustrations of the interest of gradients will be given in Section 10.

The purpose of this article is to review the various approaches that have been proposed to create metamodels with zero order *and* gradient information. A global view is first developed. Section 2.1 is a general introduction to metamodeling which may be skipped by readers familiar with the concept. After a Section presenting the main notations, Section 2.3 synthetizes into a unique framework the different techniques which will be covered in the review. A generic idea, which can be applied to any surrogate, for indirectly using gradients is summarized in Section 3.

The article then details, in turn, each gradient-enhanced method: the large family of least squares approaches are the focus of Section 4; Shepard weighting functions are summarized in Section 5. All the methods covered later are based on kernels. After summarizing the main concepts behind kernels in Section 6, we provide details about gradient-enhanced radial basis functions (Section 7), kriging (Section 8) and support vector regression (Section 9). Note that the formulation of the gradient-enhanced Support Vector Regression (ν-GSVR) proposed in part 9.4 is a new contribution. Multivariate cubic Hermite splines [10, 11] are not discussed in this review as they seem to date to remain a topic of mathematical research.

Finally, the different methods are applied and compared on analytical test functions in Section 10. The ensuing analysis of results and presentation of related softwares should help in choosing specific gradient-enhanced techniques. All methods described in this article have been implemented and tested with the opensource matlab/octave GRENAT Toolbox [12].

2 Build, validate and exploit a surrogate model

2.1 Surrogates and their building in a nutshell

In many contexts, the observation of the response of a parametrized system can be done only for a few parameters instances, also designated as *points* in the design space. A solution to getting an approximate response at non-sampled parameter instances is to use a *metamodel* (or *surrogate*). A metamodel is a doubly parameterized function, one set of parameters being the same as that of the studied system (i.e., the coordinates of the points), the other set of parameters allowing further control of the metamodel response to give it general representation abilities. For simplicity's sake, parameters of the second set will be designated as internal parameters. The building of the metamodel involves tuning its internal parameters in order to match, in a sense to be defined, the observations at the points.

The simplest metamodels are polynomials tuned by regression, which are part of the Response Surface Methodology (RSM, [13]) for analyzing the results of experiments. For dealing with an increase in nonlinearity of the function, rising the degree of the polynomial could seem to be a solution. However, oscillations appear and the number of polynomial coefficients, n_t, to be set grows rapidly, as

$$n_t = \binom{n_p + d^\circ}{d^\circ} = \frac{(n_p + d^\circ)!}{n_p!\, d^\circ!} \tag{1}$$

where n_p is the number of parameters and d° is the degree of the polynomial. This is why other techniques for approximating functions such as parametric kernel-based metamodels have received much attention.

The literature is already rich in contributions presenting and detailing surrogates models [14–23]. Hereafter, the basic steps in building and using surrogate models are summarized (see also Fig. 1):

- *Initial data generation.* Sampling strategies generate points in the design space (using, for instance, Latin Hypercube Sampling [24]). The responses of the actual function are calculated at each instance of the parameters. In many cases, this step is computationally intensive because the actual function involves a call to, typically, solvers of partial differential equations. Details on sampling techniques can be found in [25–27].
- *build the metamodel.* Because data is sparse, parametric surrogate models (which are reviewed in the rest of this paper) are used. This step mainly means determining the model internal parameters.
- In many situations and in particular for optimization, *enrichment (or infill) strategies* are used for adding points to the initial set of sample points. Enrichment strategies post-process the current surrogate. An example of infill method for optimization is the *Expected Improvement* [28, 29].
- Finally the quality of the surrogate model is *measured* using dedicated criteria (such as R^2 or $\mathtt{Q}_3$, cf. Section 10).

At the end the building process and during the infill steps the surrogate can provide inexpensive approximate responses and gradients of the actual function. For a large number of sample points and/or a large number of parameters, the building of a surrogate can be (computer) time consuming but it is typically less expensive than a nonlinear finite elements analysis.

In the context of optimization, metamodels are often used for approximating objective or constraint functions and the approximation contributes to localizing the potential areas of the optimum. For efficiency in optimization, metamodels are not made accurate in the whole design space but only in potentially good regions. Such family of approaches is designated as *Surrogate-Based Analysis and optimization* (SBAO) [18]. It is composed of optimization algorithms that rely on a metamodel, a classical optimization algorithm and an infill strategy. SBAO presents some similarities with Trust-Region methods [30] in the use of metamodels. However Trust-Region methods focus on proved rapid local convergence whereas SBAO targets globally optimal points. In SBAO the infill strategy looks for global optima by sequentially optimizing a criterion that is calculated directly with the metamodel (saving

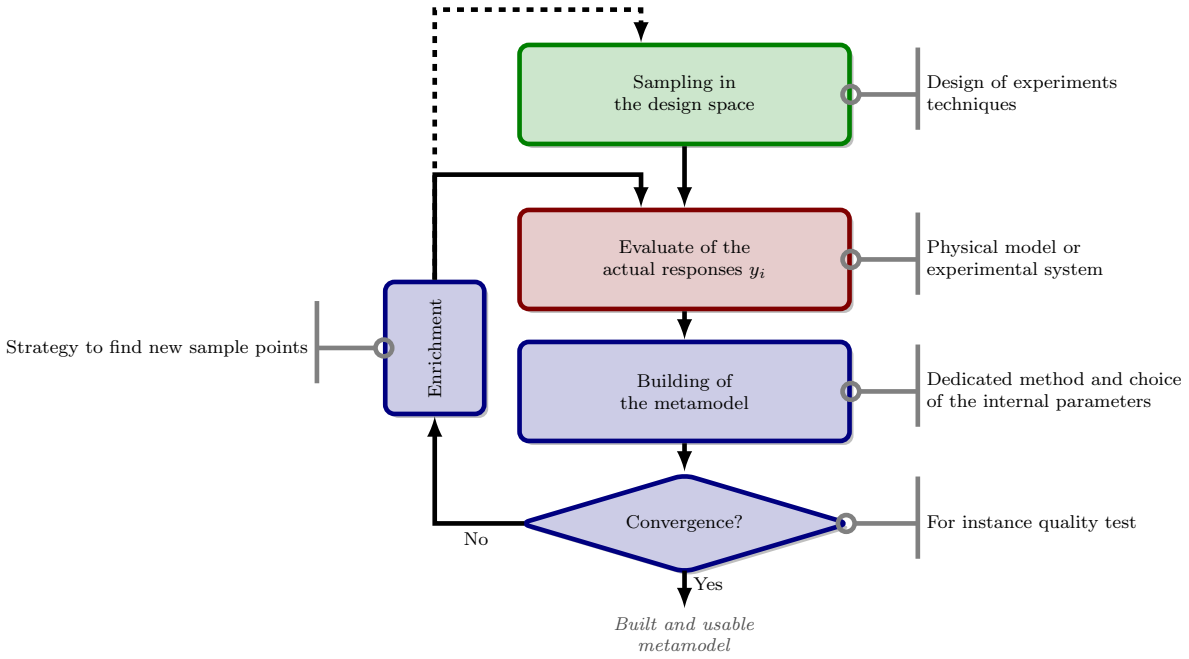


Figure 1: Schematic representation of the building process of a surrogate model (adapted from [21])

calls to the true functions) and that is a compromise between exploration and search intensification: exploration means adding points at badly known areas of the design space, intensification (also referred to as exploitation) means adding points in regions where one expects good performance. Among the many existing criteria [31, 32], *Expected Improvement* and related criteria [28, 29] are the most common and have led to the *Efficient Global Optimization* algorithm (EGO) [33]. Thus, the use of surrogates in optimization is iterative: each step of SBAO algorithms includes *i)* building a surrogate followed by *ii)* optimizing an infill criterion based on the surrogate and then *iii)* calling the actual simulation at the point output by the infill subproblem.

We now turn to the focus of this review that is gradient-enhanced metamodels also designated as gradient-assisted or gradient-based metamodels. The next sections present our notations and a global framework for gradient-enhanced metamodels.

2.2 Main notations

Let us consider an experiment parameterized by n_p continuous values grouped in the vector $\mathbf{x}^{(i)}$. n_p if often known as the dimension of the (approximation or optimization) problem. The scalar output, or response, of the experiment is the function $y(.)$. The notation $\mathbf{x}^{(i)}$ designates both sample points ($i \in [\![1, n_s]\!]$) and any non sampled point ($i = 0$). The vectors of responses (also sometimes called observations) and gradients calculated at all sample points are denoted $\mathbf{y}_g$ and are assembled according to Eq. (2)-(5).

$$\mathbf{y}_g = \begin{bmatrix} \mathbf{y}_s^\top & \mathbf{y}_{gs}^\top \end{bmatrix}^\top, \tag{2}$$

with

$$\mathbf{y}_s = \begin{bmatrix} y_1 & y_2 & \dots & y_{n_s} \end{bmatrix}^\top, \tag{3}$$

$$\mathbf{y}_{gs} = \begin{bmatrix} y_{1,1} & y_{1,2} & \dots & y_{1,n_p} & y_{2,1} & \dots & y_{n_s,n_p} \end{bmatrix}^\top, \tag{4}$$

where

$$\begin{aligned} &\forall (i,k) \in [\![0, n_s]\!] \times [\![1, n_p]\!], \\ &y_i = y\left(\mathbf{x}^{(i)}\right), \qquad y_{i,k} = \frac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right). \end{aligned} \tag{5}$$

More generally, a function y and its derivatives is written using the following index notations: $y_{,i}$, $y_{,ij}$... where i and j take values in $[\![0, n_p]\!]$ such as

$$y_{,i}(\mathbf{x}) = \begin{cases} y(\mathbf{x}) & \text{if } i = 0, \\ \dfrac{\partial y}{\partial x_i}(\mathbf{x}) & \text{if } i \in [\![1, n_p]\!]; \end{cases} \tag{6}$$

$$y_{,ij}(\mathbf{x}) = \begin{cases} y(\mathbf{x}) & \text{if } i = j = 0 \\ \dfrac{\partial y}{\partial x_i}(\mathbf{x}) & \text{if } i \in [\![1, n_p]\!] \text{ and } j = 0, \\ \dfrac{\partial y}{\partial x_j}(\mathbf{x}) & \text{if } j \in [\![1, n_p]\!] \text{ and } i = 0, \\ \dfrac{\partial^2 y}{\partial x_i \partial x_j}(\mathbf{x}) & \text{if } (i,j) \in [\![1, n_p]\!]^2. \end{cases} \tag{7}$$

Finally, the notation $\tilde{\bullet}$ designates the approximation of the quantity of interest $\bullet$ provided by the metamodel. Bold fonts mean vectors and matrices. $\| \bullet \|$ denotes the Euclidian distance.

2.3 Introduction to gradient-enhanced metamodels

This review article focuses on metamodels that, in addition to using and describing the responses, also use and model the gradient of the response with respect to the $\mathbf{x}$ parameters. Henceforth, for each sample point the value of the function and the gradients are supposed to have been observed. The following approaches will be covered: indirect approaches, Least Squares techniques (LS), Weighted Least Squares (WLS), Moving Least Squares (MLS), the Shepard Weighting function (IDW), Radial Basis Functions (GRBF), Cokriging (GKRG) and Support Vector Regression (GSVR). In these last acronyms, **G** stands for gradient-enhanced. A condensed view of the main references on which the next sections are based is given in Table 1.

Grad.-based metamodels	References
GKRG	[34–63]
GRBF	[46, 54, 56–59, 64–67]
GSVR	[54, 68–73]
IDW	[43, 46, 74]
LS	[75]
MLS	[76, 77]
WLS	[75, 78–80]

Table 1: Summary of the main references on gradient-based metamodels

Before precisely introducing each gradient-based surrogate, we give a common description of all the techniques (that can also be used for describing non-gradient-based models). It is noteworthy that all the surrogates discussed in this paper are obtained by linear combination of "coefficients" and "functions" that we will define soon. The approximation $\tilde{y}$ of an actual function y can be calculated as follows:

$$\forall \mathbf{x}^{(0)} \in \mathcal{D},\ \tilde{y}\left(\mathbf{x}^{(0)}\right) = A\left(\mathbf{x}^{(0)}; \mathbf{y}_g, \boldsymbol{\ell}\right) + \sum_{j=0}^{M} \sum_{i=1}^{N} B_{ij}\left(\mathbf{x}^{(0)}; \boldsymbol{\ell}\right) C_{ij}\left(\mathbf{x}^{(0)}; \mathbf{y}_g, \boldsymbol{\ell}\right). \tag{8}$$

$\boldsymbol{\ell}$ designates the internal parameters of the surrogate model. The terms $A()$, $B()$, and $C()$ are specific to each kind of surrogate model but share common defining properties. $A()$ is the trend term whose goal is to represent the main (average, large scale) features of the function $y()$. The $B()$'s are "functions" and the $C()$'s are "weighting coefficients". The $B()$ functions are chosen *a priori* in the sense that, assuming $\boldsymbol{\ell}$ is fixed, they do not depend on the observed responses, $y\big(\mathbf{x}^{(i)}\big)$ and their derivatives, $\frac{\partial y}{\partial x_j}\big(\mathbf{x}^{(i)}\big)$. However, the $B()$ functions can depend on the locations of the sample points, $\mathbf{x}^{(i)}$, $i = 1, \ldots, n_s$.The $B()$ functions are typically user inputs to the methods.

In contrast, the $C()$ coefficients are calculated from the observations $y\big(\mathbf{x}^{(i)}\big)$ and $\frac{\partial y}{\partial x_j}\big(\mathbf{x}^{(i)}\big)$, so that their linear combination with the $B()$ functions, eventually added to the trend, makes an approximation to $y()$, as stated in Eq. (8). The coefficients are the weights in the linear combination of the $B()$ functions. For example, if one expects that the response (for $n_p = 1$) is proportional to $1/x$ plus a quadratic term one could a priori choose $B_1(x) = 1/x$ and $B_2(x) = x^2$ and create a simple approximation with constant coefficients $y\big(\mathbf{x}^{(0)}\big) \approx \sum_{i=1}^{n_t} C_i B_i(\mathbf{x}^{(0)})$. The C_i's are then calculated from the observations, which in our context include both the response function and its derivatives at the sampled points, for example so that the approximation fits the observations in a least squares sense. When there is no a priori on the $B()$ functions, a generic choice is made: basis functions (e.g., polynomials) for LS, arctan for neural networks, kernels evaluated at a given $\mathbf{x}^{(i)}$ in kernel methods (GRBF, GKRG, GSVR here). More generally, surrogates can be created by looking, at each $\mathbf{x}^{(0)}$, for the "best" (in a certain sense) linear combination of the $B()$'s, in which case the coefficients depend on $\mathbf{x}^{(0)}$. The simplest template of such a surrogate would be $y\big(\mathbf{x}^{(0)}\big) \approx \sum_{i=1}^{n_s} \text{similarity}\big(\mathbf{x}^{(i)}, \mathbf{x}^{(0)}\big) y\big(\mathbf{x}^{(i)}\big)$ where $B_i\big(\mathbf{x}^{(0)}\big)$ is a measure of similarity between $\mathbf{x}^{(0)}$ and $\mathbf{x}^{(i)}$ (not detailed here) and the coefficients $C_i()$ are $y\big(\mathbf{x}^{(i)}\big)$. IDW and the kernel methods (GRBF, GKRG and GSVR) are refined examples of such surrogates.

Although mathematically equivalent to a single summation, the double summation in Eq. (8) emphasizes the specific structure of gradient-enhanced surrogates: in all kernel-based surrogates, the index i describes the sample point considered (therefore $N = n_s$) while j represents the variable with respect to which the derivatives are taken, $j = 0$ standing for the response without differentiation, (and $M = n_p$).

Table 2 summarizes the expressions of the trend, the functions and the coefficients such as they will appear later in the text. Note that all metamodels but LS have internal parameters, $\boldsymbol{\ell}$, that, as with non-gradient-enhanced metamodels, are computed from the known points $(\mathbf{x}^{(i)}, y(\mathbf{x}^{(i)}))$, and $\frac{\partial y}{\partial x_j}\big(\mathbf{x}^{(i)}\big)$ here, $i = 1, n_s$, in a manner which is specific to each surrogate. For the sake of clarity, the difference between the functions $B()$ and the coefficients $C()$ is made assuming that $\boldsymbol{\ell}$ is fixed, otherwise there is no clear general mathematical difference between them.

The methods that will be presented are organized in two groups, the kernel-based methods from Section 6 onward, and the rest (before). They can be distinguished in the same way as the two above examples. Kernel-based methods are built from the specification of a kernel, i.e., a function with two inputs that quantifies the similarity between what happens at these two inputs. The other approximations, which in this review are mainly variants of least squares, are made from a priori chosen single input functions that are linearly combined. Despite fundamental differences in the way they are constructed, many equivalences can be found between the methods: generalized least squares also have a kernel which is given at the end of Section 4.2; vice versa, the kernels of the GSVR are implicitely products of functions and the GSVR approximation is a linear combination of them like that of least squares; the approximate responses of GRBF and GKRG have the same form (Eq. (72) and (89) without trend are equivalent). These connections are further detailed in the paper. As a last common feature of the methods presented, it is striking that all the approaches but GSVR approximate the response by a linear combination of the observations $\mathbf{y}_g$, provided the internal parameters $\boldsymbol{\ell}$ are fixed.

The calculation of the approximate gradients will be achieved by deriving Eq. (8), i.e., calculating $\tilde{y}_{,k}(\mathbf{x})$, which is possible if A, B_{ij} and C_{ij} are differentiable functions. One could think of other ways to build $\tilde{y}_{,i}(\mathbf{x})$, like learning them directly from the gradient data $\mathbf{y}_{gs}$ independently from the response $\mathbf{y}_s$, but such techniques are instances of the usual metamodel building (just applied to the gradients)

Metamodel	A	B_{ij}	C_{ij}	M	N	$\boldsymbol{\gamma}$ tuning	Comments	Section
LS	0	$f_i(\mathbf{x}^{(0)})$	β_i	0	n_t	no $\boldsymbol{\ell}$	f_i's freely chosen, β's from Eq. (19)	4.1
WLS	0	$f_i(\mathbf{x}^{(0)})$	$\beta_i(\boldsymbol{\ell})$	0	n_t	from external knowledge	f_i's freely chosen, the β's follow Eq. (26)	4.2
MLS	0	$f_i(\mathbf{x}^{(0)})$	$\beta_i(\mathbf{x}^{(0)};\boldsymbol{\ell})$	0	n_t	Cross-validation or manually chosen	f_i's freely chosen, the β's minimize the error Eq. (35)	4.3
IDW	0	$\overline{W}_j(\mathbf{x}^{(0)};\boldsymbol{\ell})$	$Q_j(\mathbf{x}^{(0)})$	0	n_s	neighborhood radius set from sample points positions	interpolating, cf. Eq. (41) for $\overline{W}_j$ and Eq. (44) for Q_j	5
GRBF	0	$\Psi_{i,j}(\mathbf{x}^{(0)};\boldsymbol{\ell})$	$w_{ij}(\boldsymbol{\ell})$	n_p	n_s	cross-validation	interpolating, $\Psi_{i,j}$ are kernels and their derivatives, w_{ij} solution of Eq. (62)	7
GKRG	$\mathbf{f}_0^\top\hat{\boldsymbol{\beta}}$	$\left(\mathbf{r}_{0c}^\top\mathbf{K}_c^{-1}\right)_{I(i,j)}$	$\left(\mathbf{y}_g-\mathbf{F}_c\hat{\boldsymbol{\beta}}\right)_{I(i,j)}$	n_p	n_s	max. likelihood	interpolating with a trend, universal kriging in Table 6. $I(i,j)$ is the proper index in the vector product. $\mathbf{r}_{0c}^\top\mathbf{K}_c^{-1}$ depends on the kernel and point positions, not on responses	8
GSVR	$\mu(\boldsymbol{\ell})$	$\phi_{i,j}(\mathbf{x}^{(0)};\boldsymbol{\ell})$	$\vartheta_{ij}(\boldsymbol{\ell})$	n_p	n_s	Bound on cross-validation error	cf. Eq. (100), ϑ_{ij} solutions to Problem 9.1, $\phi_{i,j}$ indirectly specified through kernel choice	9

Table 2: Global framework for gradient-enhanced surrogates: Definition of trends, a priori functions and coefficients as in Eq. (8). LS means Least Squares, WLS Weighted Least Squares, MLS Moving Least Squares, IDW Shepard Weighting function, GRBF Gradient-enhanced Radial Basis Function, GKRG CoKRiGing, GSVR Gradient-enhanced Support Vector Machine.

and are out of the scope of this review.

In practice, it is common to only have access to some of the components of the gradient of the response. $\tilde{y}_{,k}(\mathbf{x})$ would be known while $\tilde{y}_{,l}(\mathbf{x})$, $k \neq l$, would not. All the techniques reviewed in this article apply only to the components where the derivatives are known. However, to keep notations simple, the derivations will always be carried out for all of the variables, as if all components of the gradient were accessible. Further remarks about missing data and higher order derivatives are given in Section 10.5.

3 Indirect gradient-based metamodels

For taking into account the derivative information of the response in the making of a metamodel, the most basic idea is to use a first order Taylor's series at sample point $\mathbf{x}^{(j)}$ to generate additional data points. For each sample point, for each of the n_p parameters, a neighboring point is created,

$$\forall (i,k) \in [\![1, n_s]\!] \times [\![1, n_p]\!], \ \mathbf{x}^{(i)} + \Delta x_k \mathbf{e}_k, \tag{9}$$

where $\mathbf{e}_k$ is an orthonormal basis vector of the design space. Under the assumption that Δx_k remains small ($|\Delta x_k| \ll 1$), the Taylor's serie provides the extrapolated value of the function y at the neighboring point :

$$y\Big(\mathbf{x}^{(i)} + \Delta x_k \mathbf{e}_k\Big) \approx y\Big(\mathbf{x}^{(i)}\Big) + \frac{\partial y}{\partial x_k}\Big(\mathbf{x}^{(i)}\Big)\Delta x_k. \tag{10}$$

Finally the non-gradient based metamodel can be built with $n_s \times (n_p + 1)$ sample points and associated responses.

This approach has been used with kriging approximation [43, 48, 81] and has been called "Indirect Cokriging". The main drawback of this method is that it requires a good choice of the Δx_k parameters: if the value is too small, the kriging correlation matrix can be ill-conditioned and too large a value leads to a degraded extrapolation by Taylor's expansion. In both cases, the metamodel provides an incorrect approximation. In previous works, Liu [81] used Maximum Likelihood Estimation for estimating the value of each parameter Δx_k and in [48], the Δx_k are chosen equal to 10^{-4}.

The indirect gradient-based approach does not scale well with dimension as the number of sample points is multiplied by $n_p + 1$ when compared to a direct approach. Moreover, because the n_p new sample points are very close to the initial sample point, numerical issues (such as the bad conditioning of covariance matrices in KRG) occur that complicate the determination of the internal parameters. Regularization or filtering techniques should be brought in. Therefore, indirect gradient-based approaches should only be used in low dimension and for problems where dedicated techniques for determining Δx_k and the internal parameters exist. In other cases, it is better not to use gradients or to opt for a direct gradient-based approach. Examples of indirect gradient-based Kriging and RBF are proposed in Fig. 6 and 9. In these figures, the derivatives of RBF and KRG are determined analytically by deriving their predictors. In such low dimension, the indirect gradient-based approaches, InRBF and InOK, seem to perform as well as the direct gradient-based approaches, GRBF and OCK, in terms of approximating the true response derivative, $dy/dx(x)$. However, as will be seen in Section 10 (Figures 23 and 25), such indirect strategies are not competitive in higher dimensions.

4 Least Squares approaches

4.1 Non weighted Least Squares (LS and GradLS)

Least Squares regression is the most common technique for approximating functions. Mainly applied in the context of Response Surface Methodology (RSM, [13]), the classical regression [82] can be extended for taking into account gradient information [75]. In this text, the acronym LS designates least squares regression without the use of gradients and GradLS is the gradient-enhanced version of it. Notice that this acronym differs from GLS that will designate *Generalized Least Squares*.

The linear model used for gradient-based formulations remains the same as for non-gradient-based versions:

$$\mathbf{y}_g = \mathbf{F}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{11}$$

but this time the vector $\mathbf{y}_g$ contains $n_s \times (n_p+1)$ terms (responses and gradients), the matrix $\mathbf{F}$ contains evaluations of the n_t a priori chosen functions f_j and their derivatives at each sample points $\mathbf{x}^{(i)}$, the vector $\boldsymbol{\beta}$ contains n_t polynomial regression coefficients β_j, and the vector $\boldsymbol{\varepsilon}$ is made of the $n_s \times (n_p+1)$ errors of the model.

For gradient-based least squares models, at each point $\left\{\mathbf{x}^{(i)}, y\left(\mathbf{x}^{(i)}\right), \dfrac{\mathrm{d}y}{\mathrm{d}\mathbf{x}}\left(\mathbf{x}^{(i)}\right)\right\}$, n_p+1 errors can be written:

$$\forall (i,k) \in [\![1,n_s]\!] \times [\![1,n_p]\!], \forall \mathbf{x}^{(i)} \in \mathcal{D},$$

$$\varepsilon_i = y\left(\mathbf{x}^{(i)}\right) - \tilde{y}\left(\mathbf{x}^{(i)}\right), \tag{12}$$

$$\varepsilon_{ik} = \frac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right) - \frac{\partial \tilde{y}}{\partial x_k}\left(\mathbf{x}^{(i)}\right). \tag{13}$$

The matrices and vectors of Eq. (11) are now further defined:

$$\mathbf{F} = \begin{bmatrix}\mathbf{F}_s^\top & \mathbf{F}_{gs}^\top\end{bmatrix}^\top, \tag{14}$$

$$\boldsymbol{\beta} = \begin{bmatrix}\beta_1 & \beta_2 & \dots & \beta_{n_t}\end{bmatrix}^\top \quad \boldsymbol{\varepsilon} = \begin{bmatrix}\varepsilon_1 & \dots & \varepsilon_{n_s} & \varepsilon_{11} & \varepsilon_{12} & \dots & \varepsilon_{1n_p} & \varepsilon_{21} & \dots & \varepsilon_{n_s n_p}\end{bmatrix}^\top, \tag{15}$$

where

$$\mathbf{F}_s = \begin{bmatrix} f_1\left(\mathbf{x}^{(1)}\right) & \dots & f_{n_t}\left(\mathbf{x}^{(1)}\right) \\ \vdots & \ddots & \vdots \\ f_1\left(\mathbf{x}^{(n_s)}\right) & \dots & f_{n_t}\left(\mathbf{x}^{(n_s)}\right) \end{bmatrix}, \tag{16}$$

$$\mathbf{F}_{gs} = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1}\left(\mathbf{x}^{(1)}\right) & \dots & \dfrac{\partial f_{n_t}}{\partial x_1}\left(\mathbf{x}^{(1)}\right) \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_1}{\partial x_{n_p}}\left(\mathbf{x}^{(1)}\right) & \dots & \dfrac{\partial f_{n_t}}{\partial x_{n_p}}\left(\mathbf{x}^{(1)}\right) \\ \dfrac{\partial f_1}{\partial x_1}\left(\mathbf{x}^{(2)}\right) & \dots & \dfrac{\partial f_{n_t}}{\partial x_1}\left(\mathbf{x}^{(2)}\right) \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_1}{\partial x_{n_p}}\left(\mathbf{x}^{(n_s)}\right) & \dots & \dfrac{\partial f_{n_t}}{\partial x_{n_p}}\left(\mathbf{x}^{(n_s)}\right) \end{bmatrix}. \tag{17}$$

The sizes of the previous matrices $\mathbf{F}_s$ and $\mathbf{F}_{gs}$ are $n_s \times n_t$ and $n_p n_s \times n_t$, respectively.

The metamodel is built by determining the vector $\hat{\boldsymbol{\beta}}$ which minimizes the following Mean Squares Error:

$$\mathrm{MSE}(\boldsymbol{\beta}) = \sum_{i=1}^{n_s}\left[\varepsilon_i^2 + \sum_{k=1}^{n_p}\varepsilon_{ik}^2\right] = \left\|\mathbf{F}\boldsymbol{\beta} - \mathbf{y}_g\right\|_2^2. \tag{18}$$

Minimizing MSE over $\boldsymbol{\beta}$ yields the function approximation,

$$\forall \mathbf{x}^{(0)} \in \mathcal{D},\ \tilde{y}\left(\mathbf{x}^{(0)}\right) = \mathbf{f}\left(\mathbf{x}^{(0)}\right)\hat{\boldsymbol{\beta}}, \tag{19}$$

where

$$\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \begin{bmatrix} f_1\left(\mathbf{x}^{(0)}\right) & \dots & f_{n_t}\left(\mathbf{x}^{(0)}\right)\end{bmatrix}, \tag{20}$$

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{F}^\top\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{y}_g. \tag{21}$$

This metamodel leads to the familiar expression of regression approximation. Notice however that here, the gradients affect the coefficients $\hat{\boldsymbol{\beta}}$.

The derivation of the GradLS model by minimization of the quadratic norm MSE can be interpreted as making the orthogonal projection of the vector of observed responses and gradients $\mathbf{y}_g$ onto the space spanned by the columns of $\mathbf{F}$. The result of the orthogonal projection is $\mathbf{F}\hat{\boldsymbol{\beta}}$. The projection is itself defined by the inner product it relies on. In least squares without derivatives, the inner product is the usual dot product between vectors in an Euclidean space. Accounting for derivatives extends this inner product to an inner product in a Sobolev space [83]:

$$\forall\ (\mathbf{u}_g, \mathbf{v}_g),\ \langle \mathbf{u}_g, \mathbf{v}_g \rangle = \langle \mathbf{u}_s, \mathbf{v}_s \rangle + \langle \mathbf{u}_{gs}, \mathbf{v}_{gs} \rangle = \sum_{i=1}^{n_s} u_i v_i + \sum_{i=1}^{n_s} \sum_{k=1}^{n_p} u_{i,k} v_{i,k}\ , \tag{22}$$

where the g, s and gs subscripts have the same meaning as above with $\mathbf{F}$. While both inner products account for the euclidean distance between the response and the metamodel, the Sobolev inner product further accounts for the difference in response and metamodel regularities through their gradients. In other words, the usual geometrical interpretations of least squares generalize to the least squares with derivatives formulation of Eq. (18) by moving from the Euclidean inner product to a product with additional derivative terms.

The derivatives of the GradLS approximation are directly obtained by deriving Eq. (19),

$$\forall k \in [\![1, n_p]\!],\ \forall \mathbf{x}^{(0)} \in \mathcal{D},\ \frac{\partial \tilde{y}}{\partial x_k}\left(\mathbf{x}^{(0)}\right) = \frac{\partial \mathbf{f}}{\partial x_k}\left(\mathbf{x}^{(0)}\right)\hat{\boldsymbol{\beta}}\ , \tag{23}$$

As required for building the gradient-enhanced least squares model, the functions f_j must be differentiable at least once.

Although the empirical mean square error Eq. (18) can be reduced by increasing the degree of the polynomial basis, $\tilde{y}()$ will increasingly oscillate between the n_s data points, which degrades the prediction quality. This oscillatory phenomenon, known as Runge's phenomenon [84], is illustrated in Fig. 2a and Fig. 3d in 1 and 2 dimensions, respectively. Runge's oscillations are mitigated when the actual function is polynomial, the number of sample points n_s increases, when gradients are accounted for like here, or when a regularization strategy is added to the MSE minimization. For example, when approximating a 4th degree polynomial function using sufficiently many sample points in a dimension low enough so that Eq. (19) can be computed, a 4th degree least squares approach is exact. Regarding the effect of gradients, observe in Fig. 2b and Fig. 3g how gradient-enhanced least squares have a more stable response than LS which only uses function values.

4.2 Generalized Least Squares (GLS)

Initially introduced for addressing the uncertainties and correlations in measured responses, *Generalized Least Squares* (GLS) follow the same logic as the previous LS and GradLS models except that weights are introduced in the error, MSE. The generalized least squares error which incorporates gradient information now reads [75, 78–80],

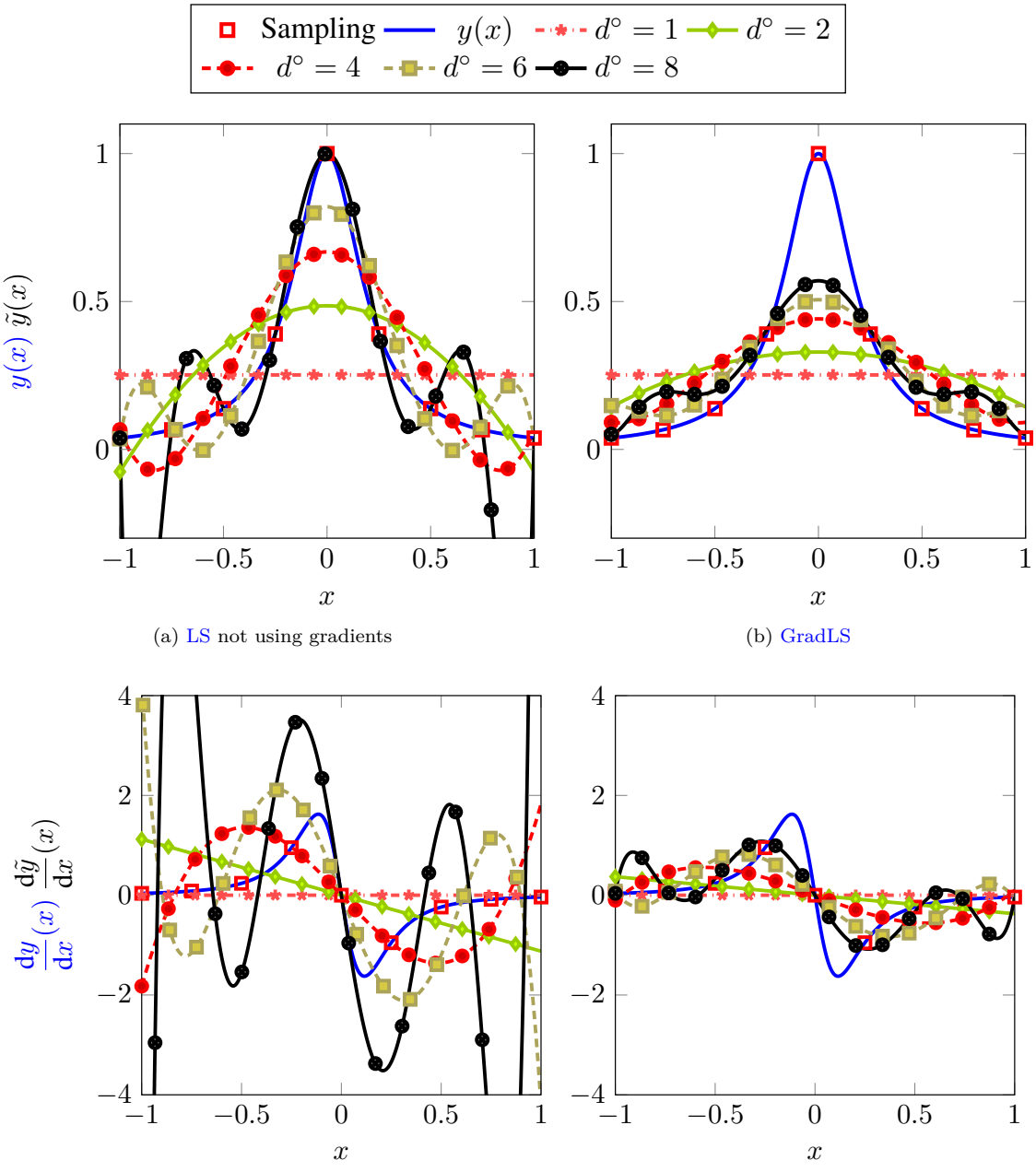
$$\mathrm{E}(\boldsymbol{\beta}) = (\mathbf{y}_s - \tilde{\mathbf{y}}_s)^\top \mathbf{W}_s (\mathbf{y}_s - \tilde{\mathbf{y}}_s) + \left(\mathbf{y}_{gs} - \tilde{\mathbf{y}}_{gs}\right)^\top \mathbf{W}_{gs} \left(\mathbf{y}_{gs} - \tilde{\mathbf{y}}_{gs}\right) \tag{24}$$

$$= \left(\mathbf{y}_g - \tilde{\mathbf{y}}_g\right)^\top \mathbf{W}_g \left(\mathbf{y}_g - \tilde{\mathbf{y}}_g\right), \tag{25}$$

where $\mathbf{W}_s$ and $\mathbf{W}_{gs}$ are positive definite weight matrices. The minimization of the error leads to the regression coefficients,

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{F}^\top \mathbf{W}_g \mathbf{F}\right)^{-1} \mathbf{F}^\top \mathbf{W}_g \mathbf{y}_g, \tag{26}$$

where $\mathbf{F}$ and $\mathbf{y}_g$ are the same as in the GradLS approach (see above) and $\mathbf{W}_g = \mathrm{diag}\begin{bmatrix}\mathbf{W}_s & \mathbf{W}_{gs}\end{bmatrix}$. The *Weighted Least Squares* (WLS) [82] approach is a special case of the *Generalized Least Squares* (GLS)



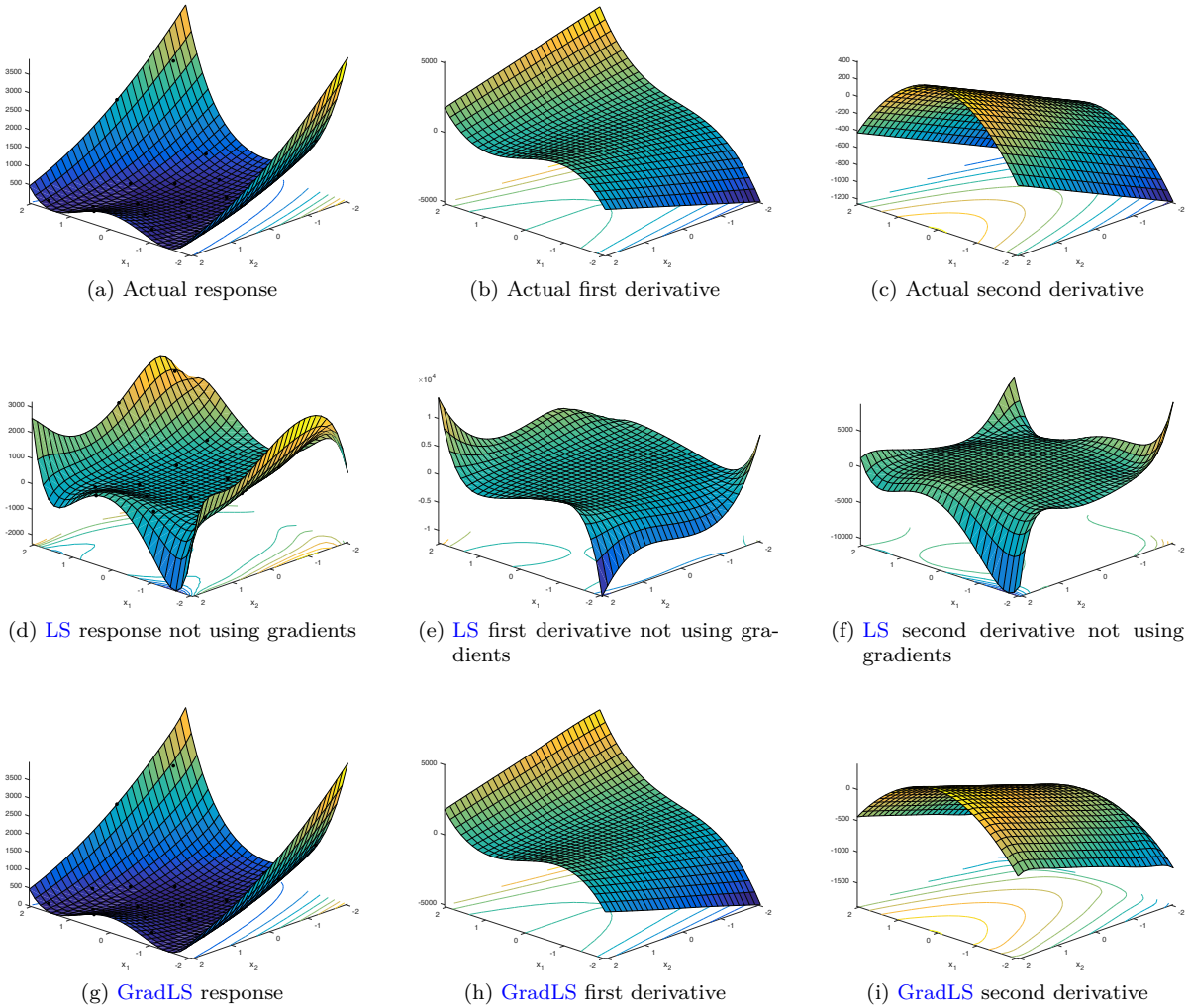
(a) LS not using gradients

(b) GradLS

(c) Derivatives from LS not using gradients

(d) Derivatives from GradLS

Figure 2: Response-only and gradient-enhanced least squares (LS and GradLS) with polynomials of degrees (d°) 1, 2, 4, 6 and 8. The actual function is $y(x) = 1/(1 + 25x^2)$, it is computed at $n_s = 9$ sample points.



(a) Actual response (b) Actual first derivative (c) Actual second derivative

(d) LS response not using gradients (e) LS first derivative not using gradients (f) LS second derivative not using gradients

(g) GradLS response (h) GradLS first derivative (i) GradLS second derivative

Figure 3: Rosenbrock's function, response-only and gradient-enhanced least squares approximations (LS and GradLS) from polynomials of degree 9 built using $n_s = 25$ sample points.

where $\mathbf{W}_g$ is a diagonal matrix. Note that Eq. (26) encompasses traditional GLS without gradients by setting $\mathbf{W}_{gs} = 0$.

In traditional GLS (models without gradients), the definition of the weight matrix $\mathbf{W}_s$ depends on the context of the study:

- If no *a priori* information on the covariance structure is available, the weights can come from a chosen weighting function $R()$: $\mathbf{W}_s = \left[R(\mathbf{x}^{(i)} - \mathbf{x}^{(j)})\right]_{1\leq i,j\leq n_s}$. $R()$ must be such that $\mathbf{W}_s$ is positive definite, a condition shared with kernels and further discussed in Section 6.
- If a covariance structure is known: $\mathbf{W}_s = \mathbf{C}^{-1}$ where $\mathbf{C} = \left[\mathrm{cov}\left[Y\left(\mathbf{x}^{(i)}\right), Y\left(\mathbf{x}^{(j)}\right)\right]\right]_{1\leq i,j\leq n_s}$. In the case of uncorrelated errors, $\mathbf{C}$ is reduced to the diagonal matrix
  $\mathrm{diag}\left[\sigma_1 \quad \sigma_2 \quad \dots \quad \sigma_{n_s}\right]$ where $\sigma_i = \mathrm{Var}[\varepsilon_i]$, and GLS degenerates into WLS.

The geometrical interpretation of gradient-enhanced GLS is similar to that of GradLS made in the previous Section, the only difference being that the projection of the vector of observations onto the space spanned by the regression functions is no longer orthogonal but oblique, following the null space of the projection matrix $\mathbf{F}\hat{\boldsymbol{\beta}}$, $\hat{\boldsymbol{\beta}}$ given by Eq. (26).

In [78], normalization methods are proposed for calculating the weight matrices of gradient-enhanced GLS:

- A *standard normalization* of responses and gradients where

$$\mathbf{W}_s = \mathrm{diag}\left[\frac{\mu_1}{y_1^2} \quad \cdots \quad \frac{\mu_{n_s}}{y_{n_s}^2}\right], \tag{27}$$

$$\mathbf{W}_{gs} = \mathrm{diag}\left[w_1 \quad \dots \quad w_1 \quad w_2 \quad \dots \quad \dots \quad w_{n_s}\right] \text{ with } w_i = \frac{\mu_i \lambda_i}{\eta_i}. \tag{28}$$

  The coefficients λ_i and μ_i are meant to balance the influence of the derivatives and responses at each sample point, respectively. η_i are normalization coefficients calculated as

$$\eta_i = \sum_{k=1}^{n_p} \frac{\partial y\left(\mathbf{x}^{(i)}\right)}{\partial x_k}. \tag{29}$$

  In this case, $\mathbf{W}_s$ contains n_s non-null terms and the diagonal of $\mathbf{W}_{gs}$ contains n_s blocks of n_p terms, $\frac{\mu_i \lambda_i}{\eta_i}$.

- A normalization using *logarithmic derivatives* where $\mathbf{W}_s$ is like that of the standard normalization above and

$$\mathbf{W}_{gs} = \mathrm{diag}\left[w_1 \quad \dots \quad w_1 \quad w_2 \quad \dots \quad \dots \quad w_{n_s}\right] \text{ with } w_i = \frac{\mu_i \lambda_i \delta_i^2}{y_i^2}. \tag{30}$$

  The δ_k coefficients, which are further described in [78], are based on the *logarithmic derivatives* introduced in [85]. μ_i and λ_i have the same expressions as in the standard normalization.

To close the presentation on gradient-enhanced generalized least squares, following [86], we show how the approach can be looked at as a kernel-based method. This comment uses explanations given in Section 8 so that readers not familiar with kernels as covariances of Gaussian processes may wish to read that Section first. The kernel is the covariance between two responses at different locations when the responses are considered as random processes,

$$\begin{aligned} \mathrm{cov}\left[\hat{Y}(\mathbf{x}), \hat{Y}(\mathbf{x}')\right] &= \mathrm{cov}\left[\mathbf{f}(\mathbf{x})\hat{\boldsymbol{\beta}}, \mathbf{f}(\mathbf{x}')\hat{\boldsymbol{\beta}}\right] = \mathbb{E}\left[\mathbf{f}(\mathbf{x})\hat{\boldsymbol{\beta}}\hat{\boldsymbol{\beta}}^\top \mathbf{f}(\mathbf{x}')^\top\right] \\ &= \mathbf{f}(\mathbf{x})\left(\mathbf{F}^\top \mathbf{W}_g \mathbf{F}\right)^{-1} \mathbf{f}(\mathbf{x}')^\top . \end{aligned} \tag{31}$$

This relation is the expression of the kernel associated to the gradient-enhanced GLS. It is obtained assuming that the responses are centered (i.e., $\mathbb{E}[Y(\mathbf{x})] = 0$) and the weight matrix is the inverse covariance of the responses and their derivatives, $\mathbf{W}_g^{-1} = \mathbb{E}\left[\mathbf{Y}_g \mathbf{Y}_g^T\right]$. It can then be checked that by using this kernel in the general prediction equation of kriging (with a null trend, see Table 6, or equivalently the GRBF prediction formula Eq. (72)),

$$\tilde{y}\left(\mathbf{x}^{(0)}\right) = \left[\mathrm{cov}\left[Y\left(\mathbf{x}^{(0)}\right), Y\left(\mathbf{x}^{(1)}\right)\right], \ldots, \mathrm{cov}\left[Y\left(\mathbf{x}^{(0)}\right), Y\left(\mathbf{x}^{(n_s)}\right)\right]\right] \mathbf{C}^{-1}\mathbf{y}_g \ , \tag{32}$$

one gets back the GLS prediction formula, $\tilde{y}\left(\mathbf{x}^{(0)}\right) = \mathbf{f}\left(\mathbf{x}^{(0)}\right)\hat{\boldsymbol{\beta}}$ with $\hat{\boldsymbol{\beta}}$ given by Eq. (26).

4.3 Moving Least Squares (MLS)

Classical response surface methods like LS, GradLS or GLS approximate functions by combining once and for all a priori functions, $f_i() \ , \ i = 1, \ldots, n_t$, that are globally defined throughout the design space. When it is not possible to decide beforehand which functions to combine, as it is the case when the function is expected to have local variations, it can be useful to proceed with local approximations. For example, it was proposed in [87] to apply the classical RSM in neighborhoods of the points of interest. *Moving Least Squares* (MLS) [88] is a generalization of GLS that builds a different metamodel at each $\mathbf{x}^{(0)}$:

$$\forall \mathbf{x}^{(0)} \in \mathcal{D}, \ \tilde{y}\left(\mathbf{x}^{(0)}\right) = \mathbf{f}\left(\mathbf{x}^{(0)}\right)\hat{\boldsymbol{\beta}}\left(\mathbf{x}^{(0)}\right). \tag{33}$$

The difference with previous approximations lies in the non constant regression coefficients $\hat{\boldsymbol{\beta}}\left(\mathbf{x}^{(0)}\right)$ (compare Eqs. (19) and (33)). Like other least squares techniques, the gradient-based MLS (also designated as Hermite version of MLS) [76, 77] is built by minimizing an error function, which here is

$$\begin{aligned}
\mathrm{E}\left(\boldsymbol{\beta}; \mathbf{x}^{(0)}\right) &= \alpha \sum_{i,j=1}^{n_s} w_{ij}\left(\mathbf{x}^{(0)}\right)\varepsilon_i\varepsilon_j + (1-\alpha)\sum_{i,j=1}^{n_s}\sum_{k,l=1}^{n_p} w_{ijkl}\left(\mathbf{x}^{(0)}\right)\varepsilon_{ik}\varepsilon_{jl} \\
&= \left(\mathbf{y}_s - \tilde{\mathbf{y}}_s\right)^\top \mathbf{W}_{Ms}\left(\mathbf{x}^{(0)}\right)\left(\mathbf{y}_s - \tilde{\mathbf{y}}_s\right) \qquad (34) \\
&\quad + \left(\mathbf{y}_{gs} - \tilde{\mathbf{y}}_{gs}\right)^\top \mathbf{W}_{Mgs}\left(\mathbf{x}^{(0)}\right)\left(\mathbf{y}_{gs} - \tilde{\mathbf{y}}_{gs}\right) \\
&= \left(\mathbf{y}_g - \tilde{\mathbf{y}}_g\right)^\top \mathbf{W}_M\left(\mathbf{x}^{(0)}\right)\left(\mathbf{y}_g - \tilde{\mathbf{y}}_g\right). \qquad (35)
\end{aligned}$$

The weights $w_{ij}\left(\mathbf{x}^{(0)}\right)$ and $w_{ijlk}\left(\mathbf{x}^{(0)}\right)$ depend of the location of $\mathbf{x}^{(0)}$. These coefficients have the following properties:

$$\forall \mathbf{x}^{(0)} \in \mathcal{D}, \ \forall (i,j,k,l) \in [\![1, n_s]\!]^2 \times [\![1, n_p]\!]^2,$$

$$w_{ij}\left(\mathbf{x}^{(0)}\right) = \begin{cases} h(\|\mathbf{x}^{(i)} - \mathbf{x}^{(0)}\|) & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases} \tag{36}$$

$$w_{ijkl}\left(\mathbf{x}^{(0)}\right) = \begin{cases} h_{kl}(\|\mathbf{x}^{(i)} - \mathbf{x}^{(0)}\|) & \text{if } i = j \text{ and } k = l, \\ 0 & \text{if } i \neq j \text{ or } k \neq l, \end{cases} \tag{37}$$

where $h()$ and $h_{kl}()$ are weight functions. Although different weight functions could be chosen for the responses and gradients, the simplest solution is to take $\forall (k,l) \in [\![1, n_p]\!]^2$, $h_{kl}(r) = h(r)$ (See [76]). α is a coefficient for managing the influence of the derivatives. $\alpha = 1$ leads to a MLS approximation without gradients.

The matrix $\mathbf{W_M}\left(\mathbf{x}^{(0)}\right)$ is diagonal, $\mathbf{W_M}\left(\mathbf{x}^{(0)}\right) = \mathrm{diag}\left[\mathbf{W}_{Ms}\left(\mathbf{x}^{(0)}\right) \ \mathbf{W}_{Mgs}\left(\mathbf{x}^{(0)}\right)\right]$, where

$\mathbf{W}_{Ms}\left(\mathbf{x}^{(0)}\right)$ and $\mathbf{W}_{Mgs}\left(\mathbf{x}^{(0)}\right)$ are $n_s \times n_s$ and $n_s n_p \times n_s n_p$ matrices, respectively:

$$\forall \in [\![1, n_s]\!],$$

$$\mathbf{W}_{Ms}\left(\mathbf{x}^{(0)}\right) = \alpha \ \mathrm{diag}\left[w_{11}\left(\mathbf{x}^{(0)}\right) \quad w_{22}\left(\mathbf{x}^{(0)}\right) \quad \dots \quad w_{n_s n_s}\left(\mathbf{x}^{(0)}\right)\right], \tag{38}$$

$$\mathbf{W}_{Mgs}\left(\mathbf{x}^{(0)}\right) = \alpha \ \mathrm{diag}\left[w_{1111}\left(\mathbf{x}^{(0)}\right) \quad w_{1122}\left(\mathbf{x}^{(0)}\right) \quad \dots \quad w_{11 n_p n_p}\left(\mathbf{x}^{(0)}\right)\right]. \tag{39}$$

The weight functions are non-negative piecewise functions chosen among the non-exhaustive list provided in Table 3.

Name	$h(r)$	Parameters
Linear	$\max\left(0, 1 - \frac{\lvert r\rvert}{\ell}\right)$	$\ell > 0$
Zhou's function [89]	$\frac{\left(1-(r/\ell)^2\right)^\alpha}{\exp\left(-(\lvert r\rvert/\ell D)^2\right) - \exp\left(-1/\ell^2\right)}$	$\ell > 0,\ \alpha$
Häussler-Combe's function [90]	$\begin{cases} \frac{\exp\left(-(\lvert r\rvert/\ell D)^2\right) - \exp\left(-1/\ell^2\right)}{1 - \exp\left(-1/\ell^2\right)} & \text{if } \lvert r\rvert/\ell < D \\ 0 & \text{if } \lvert r\rvert/\ell \geq D \end{cases}$	$\ell > 0,\ D > 0$
Cubic polynomial	$1 - 3(r/\ell)^2 + 2(\lvert r\rvert/\ell)^3$	$\ell > 0$
Fourth polynomial	$1 - 6(r/\ell)^2 + 8(\lvert r\rvert/\ell)^3 - 3(r/\ell)^4$	$\ell > 0$
Fifth polynomial	$1 - 10(\lvert r\rvert/\ell)^3 + 15(r/\ell)^4 - 6(\lvert r\rvert/\ell)^5$	$\ell > 0$
Seventh polynomial	$1 - 35(r/\ell)^4 + 84(\lvert r\rvert/\ell)^5 - 70(\lvert r\rvert/\ell)^6 + 20(\lvert r\rvert/\ell)^7$	$\ell > 0$
Squared exponential (Gaussian)	$\exp\left(-\frac{r^2}{2\ell^2}\right)$	$\ell > 0$
Generalized exponential	$\exp\left(-\left(\frac{\lvert r\rvert}{\ell}\right)^p\right)$	$\ell > 0,\ 0 \leq p \leq 2$
Cubic spline	$\begin{cases} 1 - 6\left(\frac{r}{\ell}\right)^2 + \left(\frac{\lvert r\rvert}{\ell}\right)^3 & \text{if } \lvert r\rvert < \frac{\ell}{2} \\ 2\left(1 - \frac{\lvert r\rvert}{\ell}\right) & \text{if } \frac{\ell}{2} \leq \lvert r\rvert < \ell \\ 0 & \text{if } \lvert r\rvert \geq \ell \end{cases}$	$\ell > 0$

Table 3: Examples of weighting functions, $h()$, for MLS approximation.

Finally, the MLS surrogate value at a non-sampled point $\mathbf{x}^{(0)}$ is given by Eq. (33) where the coefficients $\hat{\boldsymbol{\beta}}\left(\mathbf{x}^{(0)}\right)$ are obtained by minimizing the weighted mean squares error of Eq. (35). Because the

computation of these coefficients has to be done at each requested new point, MLS are computationally more expensive than other least squares techniques.

5 Shepard Weighting Function (IDW)

Also designated as *Inverse Distance Weighting* method (IDW), the Shepard Weighting method was introduced in [91]. The gradient-enhanced version of [46] is based on the modified Shepard Weighting method of [74]. The IDW approximation to the function is written as local linear combinations of local approximations to the true function around point $\mathbf{x}^{(i)}$, $Q_i()$. Initially chosen as a quadratic function in [74], $Q_i()$ are taken here as the first order Taylor approximation at the sampled point $\mathbf{x}^{(i)}$ for the gradient-enhanced version of IDW [43, 46].

The IDW metamodel is formulated as,

$$\forall \mathbf{x}^{(0)} \in \mathcal{D},\ \tilde{y}\left(\mathbf{x}^{(0)}\right) = \sum_{j=1}^{n_s} \overline{W_j}\left(\mathbf{x}^{(0)}\right) Q_j\left(\mathbf{x}^{(0)}\right) . \tag{40}$$

The relative weights,

$$\overline{W_j}(\mathbf{x}^{(0)}) = \frac{W_j\left(\mathbf{x}^{(0)}\right)}{\sum_{k=1}^{n_s} W_k\left(\mathbf{x}^{(0)}\right)} , \tag{41}$$

are made of the inverse distance functions,

$$W_j(\mathbf{x}) = \left[\frac{\left(R_w - \left\|\mathbf{x} - \mathbf{x}^{(j)}\right\|\right)_+}{R_w \left\|\mathbf{x} - \mathbf{x}^{(j)}\right\|} \right]^2 , \tag{42}$$

where $\forall d \in \mathbb{R},\ (d)_+ = \max(0, d)$, and R_w is a radius of influence around $\mathbf{x}^{(j)}$. The weight functions W_j are such that $Q_j(\mathbf{x})$ has an influence on the approximation only in a (hyper)sphere of center $\mathbf{x}^{(j)}$ and radius R_w. R_w is set so that the hypershere includes N_w sample points. A discussion on R_w and N_w can be found in [74].

The weight functions of Eqs. (41,42) have the following properties:

$$\begin{aligned} &\forall (i,j) \in [\![0, n_s]\!] \times [\![1, n_s]\!],\ \forall \mathbf{x}^{(i)} \in \mathcal{D}, \\ &\overline{W_j}\left(\mathbf{x}^{(i)}\right) = \delta_{ij} = \begin{cases} 0 & \text{if } j \neq i \ , \\ 1 & \text{if } j = i \ . \end{cases} \end{aligned} \tag{43}$$

The function $Q_j(\mathbf{x})$ is a first order Taylor approximation of y at $\mathbf{x}^{(j)}$,

$$\forall \mathbf{x} \in \mathcal{D},\ Q_j(\mathbf{x}) = y\left(\mathbf{x}^{(j)}\right) + \sum_{k=1}^{n_p} \frac{\partial y\left(\mathbf{x}^{(j)}\right)}{\partial x_k} \left(x_k - x_k^{(j)}\right) . \tag{44}$$

The IDW approximation interpolates responses and gradients of the actual function at the sample points. To prove it, the IDW prediction and its derivatives are now calculated at the sample points:

$$\begin{aligned} &\forall i \in [\![1, n_s]\!],\ \forall \mathbf{x}^{(i)} \in \mathcal{D}, \\ \tilde{y}\left(\mathbf{x}^{(i)}\right) &= \sum_{j=1}^{n_s} \overline{W_j}\left(\mathbf{x}^{(i)}\right) Q_j\left(\mathbf{x}^{(i)}\right) = Q_i\left(\mathbf{x}^{(i)}\right) \\ &= y\left(\mathbf{x}^{(i)}\right) \ ; \end{aligned} \tag{45}$$

$$\begin{aligned}
&\forall (i,l) \in [\![1,n_s]\!] \times [\![1,n_p]\!],\ \forall \mathbf{x}^{(i)} \in \mathcal{D},\\
\frac{\partial \tilde{y}\left(\mathbf{x}^{(i)}\right)}{\partial x_l} &= \sum_{j=1}^{n_s} \left[\frac{\partial \overline{W_j}\left(\mathbf{x}^{(i)}\right)}{\partial x_l} Q_j\left(\mathbf{x}^{(i)}\right) + \overline{W_j}\left(\mathbf{x}^{(i)}\right) \frac{\partial Q_j\left(\mathbf{x}^{(i)}\right)}{\partial x_l} \right]\\
&= \frac{\partial Q_i\left(\mathbf{x}^{(i)}\right)}{\partial x_l} = \frac{\partial y\left(\mathbf{x}^{(i)}\right)}{\partial x_l}\ ,
\end{aligned} \tag{46}$$

because,

$$\begin{aligned}
&\forall (i,j,l) \in [\![1,n_s]\!]^2 \times [\![1,n_p]\!], \forall \mathbf{x}^{(i)} \in \mathcal{D},\\
&\frac{\partial \overline{W_j}\left(\mathbf{x}^{(i)}\right)}{\partial x_l} = 0\ .
\end{aligned} \tag{47}$$

The IDW metamodel bears similarities to the kernel methods of Sections 6, 7, 8, 9: $\overline{W_j}(\mathbf{x})$ is a double input function that grows with proximity between $\mathbf{x}$ and $\mathbf{x}^{(j)}$; In IDW, $\overline{W_j}(\mathbf{x})$ is multiplied with response estimates (the $Q_j()$'s) in a way that is reminiscent of kriging, cf. GKRG in Table 2. Note also that, when compared to the other metamodels reviewed in this paper, IDW is the only approach that neither requires the inversion of large ($n_s(n_p+1)$ by $n_s(n_p+1)$) systems of linear equations nor the resolution of optimization problems as GSVR will. For this reason, IDW is computationally inexpensive. We now turn to the already mentioned kernel methods.

6 Kernel functions for gradient-enhanced kernel-based metamodels

Most kernel-based metamodels have been developped in the field of machine learning. While Support Vector Machines are arguably the most well-known, other approximation techniques belong to kernel-based techniques. In this article, we will focus on Radial Basis functions (See Section 7), Kriging (See Section 8) and Support Vector Regression (See Section 9). These three surrogate models, like all kernel-based metamodels, require choosing a kernel function or kernel, Ψ, which measures a similarity, $\Psi\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right)$, between any two points $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$, and is therefore a double input function. Kernel functions are examples of the functions $B()$ of the general metamodel framework, Eq. (8).

As will be done in Section 8 about kriging, one can look at the responses at each point $\mathbf{x}$ as a random process, $Y(\mathbf{x})$. With this point of view, since a kernel is a similarity measure, it is natural to define a kernel as the correlation between the responses at different locations, $\Psi\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) = \mathrm{corr}\left[Y\left(\mathbf{x}^{(i)}\right), Y\left(\mathbf{x}^{(j)}\right)\right]$.

Kernels must satisfy Mercer's conditions [92] which means that they must be continuous, symmetric and positive definite, a necessary condition for correlation functions. This is most easily done by taking the kernel function in a list of known Mercer's kernels [86, 92, 93].

In the case of gradient-enhanced approximations, a great simplification comes from the fact that the kernels involving gradients are deduced from the kernel involving only the responses: the correlation functions between a response and a gradient is the derivative of the kernel and the correlation between two gradients is the second derivative of the correlation, cf. Eq. (92).

An additional condition on the kernel functions has then to be satisfied: the kernels used in gradient-enhanced metamodels must be twice differentiable.

Multidimensional kernel functions Ψ are usually built from unidimensional kernels h by taking the product,

$$\begin{aligned}
&\forall (\mathbf{x}^{(i)}, \mathbf{x}^{(j)}, \boldsymbol{\ell}) \in (\mathbb{R}^{n_p})^3,\\
&\Psi\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\ell}\right) = \prod_{k=1}^{n_p} h(x_k^{(i)} - x_k^{(j)}; \ell_k),
\end{aligned} \tag{48}$$

where $\boldsymbol{\ell}$ is the vector of the kernel internal parameters. In the above formula, we have introduced the stationarity assumption that the similarity between two points depends only on the vector separating them and not on where they are located, $h(x_k^{(i)}, x_k^{(j)}) = h(x_k^{(i)} - x_k^{(j)}) = h(r)$. The sign of r is kept to simplify the calculations of the kernel derivatives. For gradient-enhanced metamodels, common twice differentiable kernel functions are summarized in Table 4 (See for example [62, 63, 94, 95]).

Name	$h(r)$	Parameters
Squared exponential	$\exp\left(-\frac{r^2}{2\ell^2}\right)$	$\ell > 0$
Cubic spline 1	$\begin{cases} 1 - 15\left(\frac{r}{\ell}\right)^2 + 30\left(\frac{\lvert r\rvert}{\ell}\right)^3 & \text{if } \lvert r\rvert < 0.2\ell \\ 2\left(1 - \frac{\lvert r\rvert}{\ell}\right) & \text{if } 0.2\ell \le \lvert r\rvert < \ell \\ 0 & \text{if } \lvert r\rvert \ge \ell \end{cases}$	$\ell > 0$
Cubic spline 2	$\begin{cases} 1 - 6\left(\frac{r}{\ell}\right)^2 + 6\left(\frac{\lvert r\rvert}{\ell}\right)^3 & \text{si } \lvert r\rvert < \frac{\ell}{2} \\ 2\left(1 - \frac{\lvert r\rvert}{\ell}\right) & \text{si } \frac{\ell}{2} \le \lvert r\rvert < \ell \\ 0 & \text{si } \lvert r\rvert \ge \ell \end{cases}$	$\ell > 0$
Matérn	$\frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}\lvert r\rvert}{\ell}\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}\lvert r\rvert}{\ell}\right)$	$(\ell, \nu) \in (\mathbb{R}^+)^2$
Matérn 3/2	$\left(1 + \frac{\sqrt{3}\lvert r\rvert}{\ell}\right)\exp\left(-\frac{\sqrt{3}\lvert r\rvert}{\ell}\right)$	$\ell > 0$
Matérn 5/2	$\left(1 + \frac{\sqrt{5}\lvert r\rvert}{\ell} + \frac{5r^2}{3\ell^2}\right)\exp\left(-\frac{\sqrt{5}\lvert r\rvert}{\ell}\right)$	$\ell > 0$

Table 4: Examples of kernel functions, $r = x_k^{(i)} - x_k^{(j)}$

Introduced by Stein [96] in the context of approximation, the Matérn class [97] of kernels have parameters that make them highly adjustable. Matérn kernels use a modified Bessel function of the second kind K_ν normalized by a Gamma function $\Gamma(\nu)$. Thanks to the parameter ν, the smoothness of the kernel function can be accurately controlled. Matérn functions and their derivatives for 3 values of ν and $\ell = 0.8$ are plotted in Fig. 4. In practice, two specific values of ν leads to the most often used Matérn 3/2 and Matérn 5/2 ($\nu = 3/2$ or $5/2$) functions. Figures 5b and 5c show these functions and their derivatives for 3 values of the parameter ℓ. They can be compared with the squared exponential kernel presented on Fig. 5a. The Matérn function is $\lceil \nu \rceil$ times differentiable [96] (where $\lceil \bullet \rceil$ denotes the ceiling function). A stronger result is that the second derivative is continuous in 0 and its asymptotic value is [98]

$$\frac{\mathrm{d}^2 h(r)}{\mathrm{d}r^2} \underset{r\to 0}{\sim} -\frac{\nu}{\ell^2}\frac{\Gamma(\nu-1)}{\Gamma(\nu)} \ , \tag{49}$$

Because the k-th derivative of the metamodel exist if the $k+1$-th derivative of the kernel at 0 exist and is finite ([86] for Gaussian Processes), the Matérn function with $\nu > 1$ can be used for building gradient-based ($k = 1$) metamodels. This assessment confirms the validity of the choice $\nu \ge 3/2$ proposed in [63, 94, 95, 99]. The squared exponential kernel has a very simple expression and is often encountered in practice. It should be noted that it yields extremely smooth metamodels: it is infinitely differentiable at $r = 0$ and so are the associated surrogates. Such smoothness is often not representative of the true function and, worse, it causes ill-conditioning of matrices in radial basis functions and kriging (cf. Sections 7 and 8). This is the reason why Matérn kernels should generally be preferred.

The implementation of multidimensionnal kernel functions and their first and second derivatives can lead to a complicated and time consuming code. In order to improve both aspects, [99] has proposed

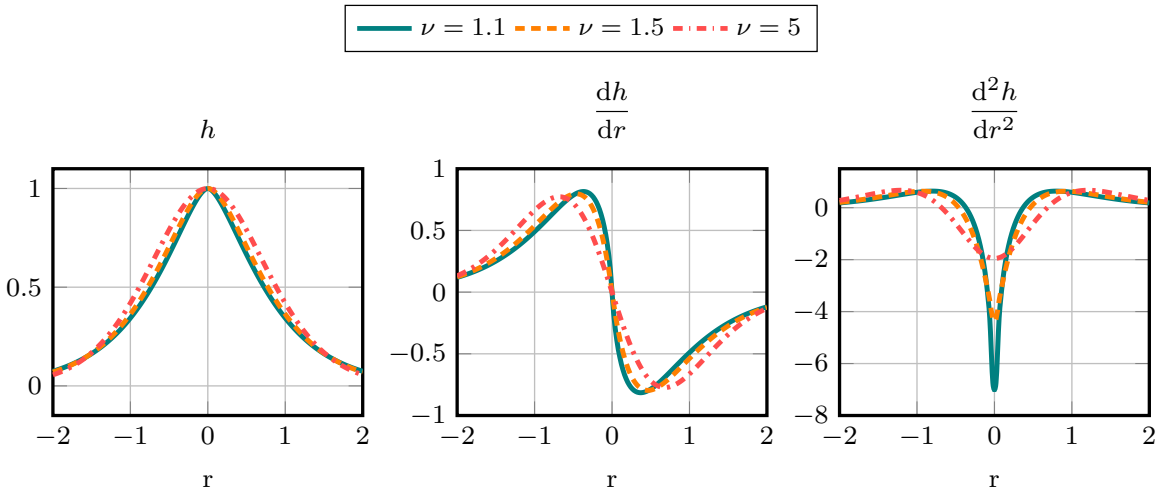


Figure 4: Matérn function for $\ell = 0.8$ and 3 values of ν

the following formulation:

$$L_m = \prod_{k=1}^{m} h(x_k^{(i)} - x_k^{(j)}; \ell_k); \tag{50}$$

$$U_m = \prod_{k>m}^{n_p} h(x_k^{(i)} - x_k^{(j)}; \ell_k); \tag{51}$$

$$M_{m,n} = \prod_{m<k<n}^{n_p} h(x_k^{(i)} - x_k^{(j)}; \ell_k) \text{ with } m < n. \tag{52}$$

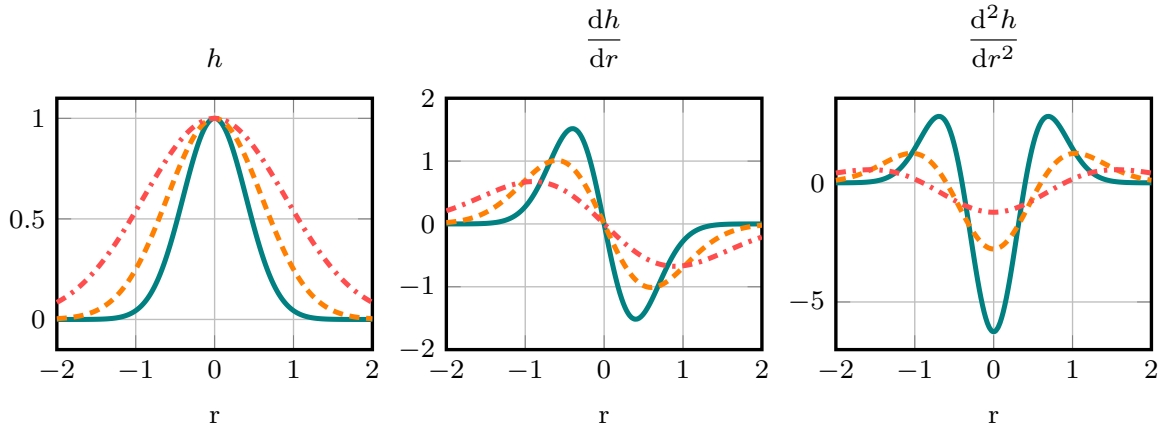
The derivatives can then be computed as shown below where only the derivatives of the unidimensional correlation function are needed,

$$\frac{\partial \Psi}{\partial x_m^{(i)}}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\ell}\right) = L_m U_m \frac{\mathrm{d}h}{\mathrm{d}x_m^{(i)}}\left(x_m^{(i)} - x_m^{(j)}; \ell_m\right); \tag{53}$$

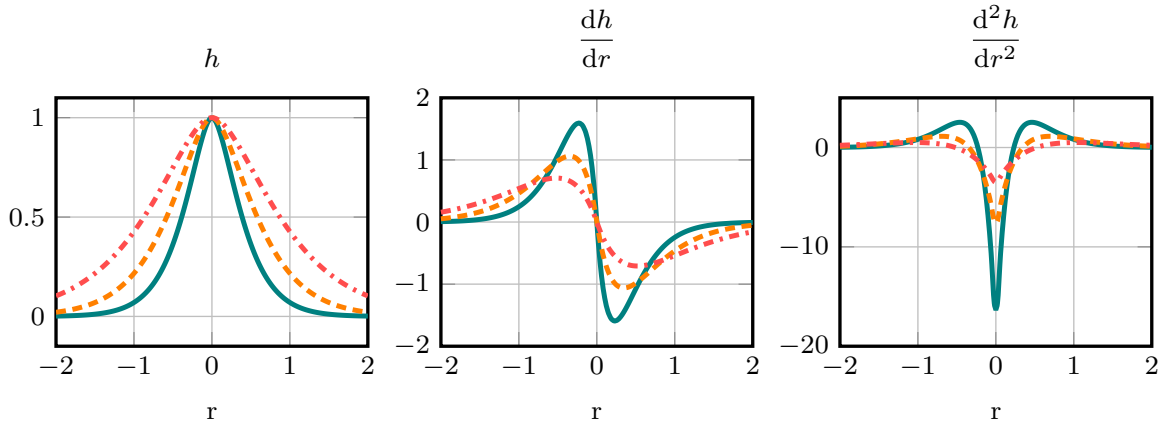
$$\frac{\partial^2 \Psi}{\partial x_m^{(i)} \partial x_n^{(j)}}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\ell}\right) = \begin{cases} L_m M_{m,n} U_n \dfrac{\mathrm{d}h}{\mathrm{d}x_m^{(i)}}\left(x_m^{(i)} - x_m^{(j)}; \ell_m\right) \dfrac{\mathrm{d}h}{\mathrm{d}x_n^{(j)}}\left(x_n^{(i)} - x_n^{(j)}; \ell_n\right) & \text{if } m \neq n, \\ L_m U_m \dfrac{\partial^2 h}{\partial x_m^{(i)} \partial x_m^{(j)}}\left(x_m^{(i)} - x_m^{(j)}; \ell_m\right) & \text{if } m = n. \end{cases} \tag{54}$$

7 Gradient-enhanced Radial Basis Function (GRBF)

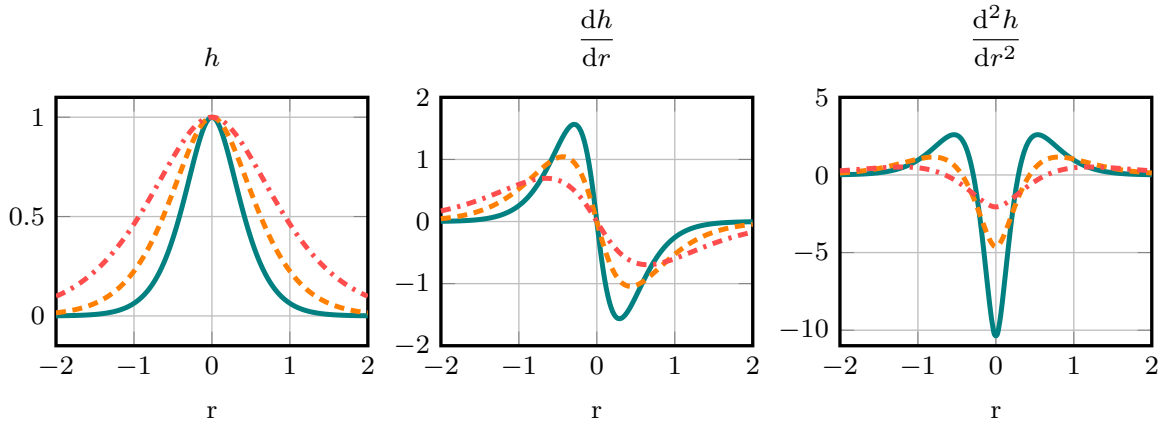
Gradient-enhanced Radial Basis Function (GRBF) has also been designated as Hermite-Birkhoff or Hermite interpolation [64]. This method was introduced in the more global context of Artificial Neural Networks [65, 66] and it was used for dealing with optimization problems involving expensive solvers in the context of computational fluid dynamics [46, 67] and assembly design [56–59].



(a) Squared exponential kernel



(b) Matérn 3/2 kernel



(c) Matérn 5/2 kernel

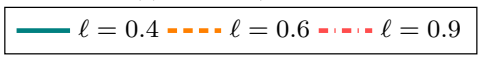


Figure 5: Examples of several kernel functions recommended for gradient-based metamodels

7.1 Building process

The principle of GRBF is similar to that of classical RBF approach [100–102] with an extended basis of functions. The added functions are chosen as the derivatives of the radial basis functions Ψ. Thus, the GRBF approximation reads,

$$\begin{aligned}\forall \mathbf{x}^{(0)} \in \mathcal{D},\\ \tilde{y}\left(\mathbf{x}^{(0)}\right) &= \sum_{i=1}^{n_s} w_i \Psi\left(\mathbf{x}^{(0)}, \mathbf{x}^{(i)}\right) + \sum_{j=1}^{n_p}\sum_{i=1}^{n_s} w_{ij} \frac{\partial \Psi}{\partial x_j^{(0)}}\left(\mathbf{x}^{(0)}, \mathbf{x}^{(i)}\right)\\ &= \sum_{j=0}^{n_p}\sum_{i=1}^{n_s} w_{ij}\Psi_{0i,j},\end{aligned} \tag{55}$$

where

$$\forall \mathbf{x}^{(0)} \in \mathcal{D},\ \forall (i,j,k) \in [\![0, n_s]\!]^2 \times [\![1, n_p]\!],$$

$$w_{ij} = \begin{cases} w_{i0} = w_i & \text{if } j = 0,\\ w_{ij} & \text{otherwise;}\end{cases} \tag{56}$$

$$\Psi_{ij,k} = \begin{cases} \Psi_{ij,0} = \Psi_{ij} = \Psi\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) & \text{if } k = 0,\\ \Psi_{ij,k} = \dfrac{\partial \Psi_{ij}}{\partial x_k^{(i)}} = \dfrac{\partial \Psi}{\partial x_k^{(i)}}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) & \text{otherwise.}\end{cases} \tag{57}$$

Only one half of the first derivatives needs to be calculated because they are odd functions:

$$\forall (i,j,k) \in [\![0, n_s]\!]^2 \times [\![1, n_p]\!],$$

$$\frac{\partial \Psi}{\partial x_k^{(i)}}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) = -\frac{\partial \Psi}{\partial x_k^{(j)}}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right). \tag{58}$$

The second derivatives of the radial basis functions will be denoted

$$\forall (i,j,k,l) \in [\![0, n_s]\!]^2 \times [\![0, n_p]\!]^2,\ \forall (\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \in \mathcal{D},$$

$$\Psi_{ij,kl} = \frac{\partial^2 \Psi}{\partial x_k \partial x_l}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right). \tag{59}$$

The GRBF building process consists in the determination of the w_{ij}'s coefficients by ensuring that the GRBF approximation interpolates the responses and gradients of the actual function at the sample points:

$$\forall (k,l) \in [\![1, n_s]\!] \times [\![1, n_p]\!], \forall \mathbf{x}^{(k)} \in \mathcal{D},$$

$$\tilde{y}\left(\mathbf{x}^{(k)}\right) = \tilde{y}_k = y_k = y\left(\mathbf{x}^{(k)}\right), \tag{60}$$

$$\frac{\partial \tilde{y}}{\partial x_l}\left(\mathbf{x}^{(k)}\right) = \tilde{y}_{k,l} = y_{k,l} = \frac{\partial y}{\partial x_l}\left(\mathbf{x}^{(k)}\right). \tag{61}$$

Equations (60) and (61) lead to the following matrix formulation:

$$\boldsymbol{\Psi}_g \mathbf{w}_g = \mathbf{y}_g. \tag{62}$$

The vectors $\mathbf{w}_g$ and $\mathbf{y}_g$ contain the RBF coefficients and the responses and gradients of the actual function, respectively. The matrix $\boldsymbol{\Psi}_g$ is built from the classical RBF matrix $\boldsymbol{\Psi}$ and the first and second derivatives of the radial basis functions matrices, denoted $\boldsymbol{\Psi}_d$ and $\boldsymbol{\Psi}_{dd}$, respectively:

$$\boldsymbol{\Psi}_g = \begin{bmatrix} \boldsymbol{\Psi} & -\boldsymbol{\Psi}_d \\ \boldsymbol{\Psi}_d^\top & \boldsymbol{\Psi}_{dd} \end{bmatrix}; \tag{63}$$

$$\boldsymbol{\Psi} = \begin{bmatrix} \Psi_{11} & \Psi_{12} & \dots & \Psi_{1n_s} \\ \Psi_{21} & \Psi_{22} & \dots & \Psi_{2n_s} \\ \vdots & & \ddots & \vdots \\ \Psi_{n_s1} & \Psi_{n_s2} & \dots & \Psi_{n_sn_s} \end{bmatrix}; \tag{64}$$

$$\boldsymbol{\Psi}_d = \begin{bmatrix} \Psi_{11,1} & \Psi_{11,2} & \dots & \Psi_{11,n_p} & \Psi_{12,1} & \dots & \Psi_{1n_s,n_p} \\ \Psi_{21,1} & \Psi_{21,2} & \dots & \Psi_{21,n_p} & \Psi_{22,1} & \dots & \Psi_{2n_s,n_p} \\ \vdots & & \ddots & \vdots & & \ddots & \\ \Psi_{n_s1,1} & \Psi_{n_s1,2} & \dots & \Psi_{n_s1,n_p} & \Psi_{n_s2,1} & \dots & \Psi_{n_sn_s,n_p} \end{bmatrix}; \tag{65}$$

$$\boldsymbol{\Psi}_{dd} = \begin{bmatrix} \Psi_{11,11} & \Psi_{11,12} & \dots & \Psi_{11,1n_p} & \Psi_{12,11} & \dots & \Psi_{1n_s,1n_p} \\ \Psi_{11,21} & \Psi_{11,22} & \dots & \Psi_{11,2n_p} & \Psi_{12,21} & \dots & \Psi_{1n_s,2n_p} \\ \vdots & & \ddots & \vdots & & \ddots & \\ \Psi_{11,n_p1} & \Psi_{11,n_p2} & \dots & \Psi_{11,n_pn_p} & \Psi_{12,n_p1} & \dots & \Psi_{1n_s,n_pn_p} \\ \Psi_{21,11} & \Psi_{21,12} & \dots & \Psi_{21,1n_p} & \Psi_{22,11} & \dots & \Psi_{2n_s,1n_p} \\ \vdots & & \ddots & \vdots & & \ddots & \\ \Psi_{n_s1,n_p1} & \Psi_{n_s1,n_p2} & \dots & \Psi_{n_s1,n_pn_p} & \Psi_{n_s2,n_p1} & \dots & \Psi_{n_sn_s,n_pn_p} \end{bmatrix}. \tag{66}$$

The sizes of the $\boldsymbol{\Psi}$, $\boldsymbol{\Psi}_d$ and $\boldsymbol{\Psi}_{dd}$ matrices are $n_s \times n_s$, $n_s n_p \times n_s$ and $n_s n_p \times n_s n_p$, respectively. So, matrix $\boldsymbol{\Psi}_g$ contains $n_s(1+n_p) \times n_s(1+n_p)$ terms. The other terms in Eq. (62) are

$$\mathbf{w}_g = \begin{bmatrix} w_1 & \dots & w_{n_s} & w_{11} & w_{12} & \dots & w_{1n_p} & w_{21} & \dots & w_{n_sn_p} \end{bmatrix}^\top; \tag{67}$$

$$\mathbf{y}_g = \begin{bmatrix} y_1 & \dots & y_{n_s} & y_{11} & y_{12} & \dots & y_{1n_p} & y_{21} & \dots & y_{n_sn_p} \end{bmatrix}^\top. \tag{68}$$

The determination of the w_{ij}'s finally consists in the inversion of the $\boldsymbol{\Psi}_g$ matrix. This square symmetrical matrix is larger than the $\boldsymbol{\Psi}$ of the classical RBF approach.

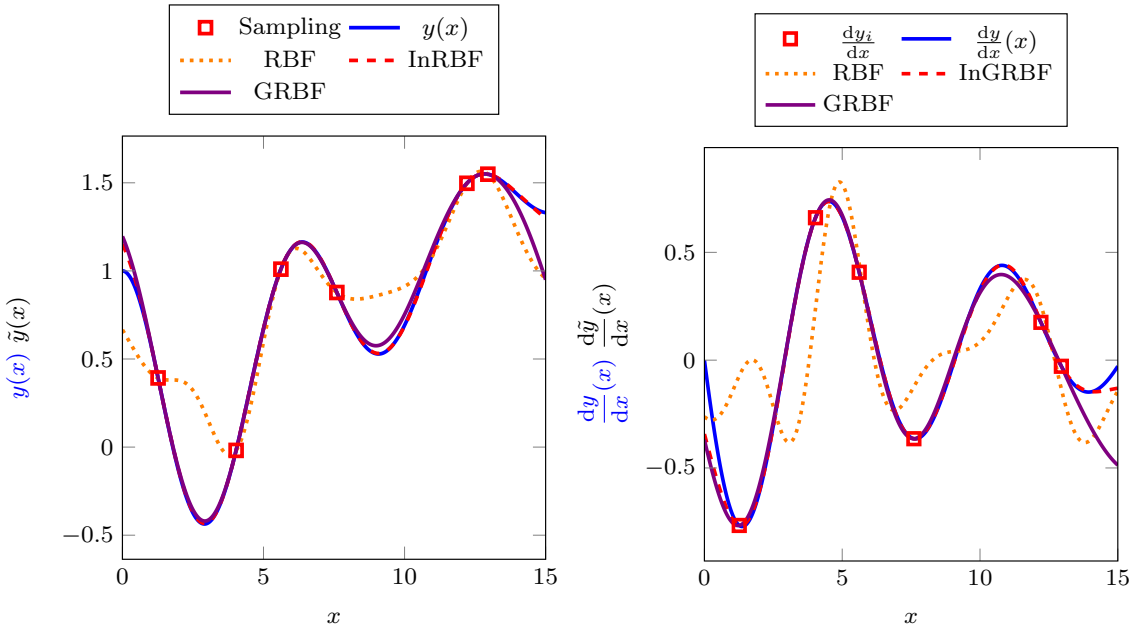
In order to reduce the computation time, LU or Cholesky factorisation of the $\boldsymbol{\Psi}_g$ matrix can be used.

Finally, the derivatives of the GRBF can be easily calculated by deriving Eq. (55).

Figures 6, 7 and 8 provide illustrations on analytical functions. In the figures, the points have been generated by an Improved Hypercube Sampling technique, IHS, [103]. An indirect version of the gradient-enhanced RBF is proposed in 1D. This approach works in unidimensional problems but it becomes unstable as the number of sample points and the number of parameters of the problem increase.

7.2 RBF kernels and conditioning

Many radial basis functions have been proposed in the literature (see for example [104]) and they can be completed by the kernels presented in Section 6. In the case of gradient-based RBF, the kernels must be at least twice differentiable to comply with the expressions of Eq. (60) and (61). Thus, Matérn or squared exponential kernels can be used in GRBF. The matrix made of 0, 1 and 2nd order derivatives $\boldsymbol{\Psi}_g$ is guaranteed to be positive definitite as will be explained in Section 8.3 about the $\mathbf{C}_c$ matrix which has the same form. However, the conditioning of the matrix may be bad. As already discussed in Section 6, the squared exponential kernel is likely to yield an ill-conditioned $\boldsymbol{\Psi}_g$ matrix, an issue that can be addressed through any of the following techniques: use more distant sampled points or equivalently decrease the value of the internal parameters $\boldsymbol{\ell}$; replace the squared exponential kernel with a Matérn kernel. Another solution can be to add a very small value (with an order fo magnitude of the machine accuracy) to the diagonal of the $\boldsymbol{\Psi}_g$ matrix. In this case the GRBF approximation will not interpolate the responses and gradients at the sample point.



(a) Responses (b) Derivatives

Figure 6: RBF, Indirect RBF (inRBF) and GRBF approximations of the unidimensional analytical function $y(x) = \exp(-x/10)\cos(x) + x/10$. $n_s = 6$ sample points, squared exponential function.

7.3 Estimation of parameters

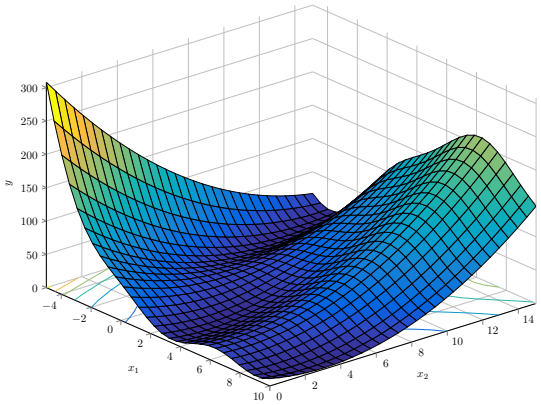
The internal parameters of the RBF metamodel can be determined by minimizing the *Leave-One-Out* error (LOO) with respect to $\boldsymbol{\ell} = (\ell_i)_{1\leq i\leq n_p}$ (and ν in the case of the Matérn kernel). Based on the principle of *Cross-validation* [105, 106], the classical LOO error is detailed hereafter where $\tilde{y}_{-i}(\mathbf{x}^{(i)})$ is the RBF approximation at point $\mathbf{x}^{(i)}$ without taking into account the response and the gradient of the actual function at that sample point $\mathbf{x}^{(i)}$:

$$\mathrm{LOO}(\boldsymbol{\ell}) = \frac{1}{n_s}\sum_{i=1}^{n_s}\Big(\tilde{y}_{-i}(\mathbf{x}^{(i)}) - y\Big(\mathbf{x}^{(i)}\Big)\Big)^2. \tag{69}$$

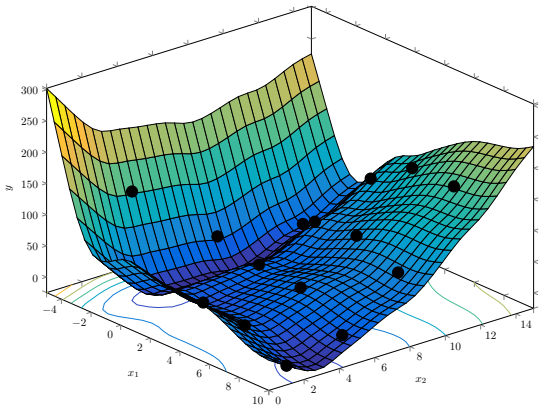
Bompard *et al.* [107] propose an extended LOO criterion by adding the derivatives information:

$$\mathrm{LOO}(\boldsymbol{\ell}) = \frac{1}{n_s(n_p+1)}\sum_{i=1}^{n_s}\left[\Big(\tilde{y}_{-i}(\mathbf{x}^{(i)}) - y\Big(\mathbf{x}^{(i)}\Big)\Big)^2 + \sum_{k=1}^{n_p}\left(\frac{\partial\tilde{y}_{-i,k}}{\partial x_k}(\mathbf{x}^{(i)}) - \frac{\partial y}{\partial x_k}(\mathbf{x}^{(i)})\right)^2\right], \tag{70}$$

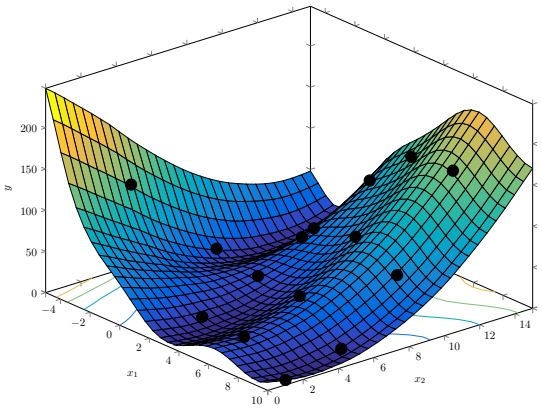
where $\dfrac{\partial\tilde{y}_{-i,k}}{\partial x_k}$ is the approximation of the derivative provided by the metamodel built without taking into account the true k-th component of the gradient at point $\mathbf{x}^{(i)}$. The approximations $\tilde{y}_{-i}$ and $\dfrac{\partial\tilde{y}_{-i,k}}{\partial x_k}$ are therefore obtained through Eq.(60) and partial use of (61) when a gradient is omitted from the LOO error. In order to avoid the building of numerous metamodels associated to each value of the internal parameters, an efficient way for computing the LOO was proposed in [108] and extended to GRBF in [107]. It implies estimating the LOO criterion by inverting the kernel matrix once and for all and calculating a vector product instead of completely building the metamodel each time a data is



(a) Actual Branin's response surface

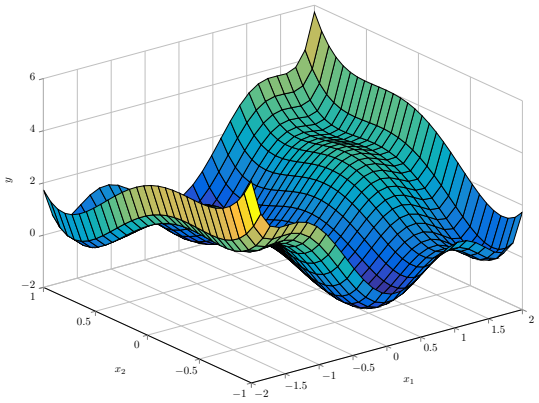


(b) RBF

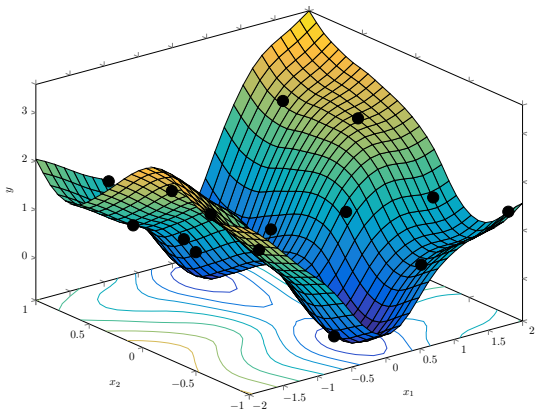


(c) GRBF

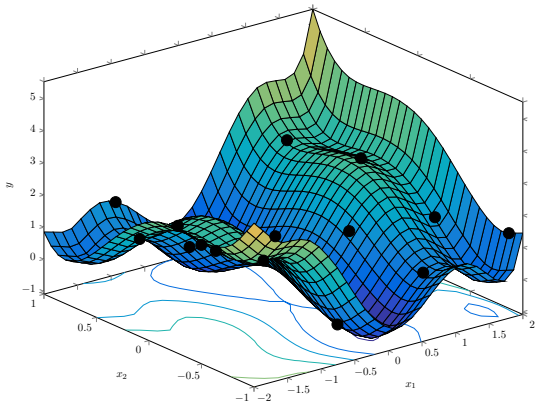
Figure 7: RBF and GRBF approximations of the two-dimensional Branin's function, $\forall (x_1, x_2) \in [-5, 10] \times [0, 15]$, $y(x_1, x_2) = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 1$, IHS sampling with $n_s = 20$, Matérn 3/2 kernel function.



(a) Actual Six-hump Camel Back response surface



(b) RBF



(c) GRBF

Figure 8: RBF and GRBF approximations of the two-dimensional Six-hump Camel Back function, $\forall (x_1, x_2) \in [-2, 2] \times [-1, 1]$, $y(x_1, x_2) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right) x_1^2 + x_1 x_2 + \left(-4 + 4x_2^2\right) x_2^2$, IHS sampling with $n_s = 20$, Matérn 3/2 kernel function

removed. Finally, due to the multimodality of the LOO, a global optimizer has to be used such as a stochastic optimizer (e.g., an evolution strategy or a particle swarm algorithm).

7.4 Variance of a stochastic process obtained with GRBF

As an extension to an idea given in [29], Bompard [54] proposes to look at the deterministic response y as an instance of a stationary Gaussian stochastic process Y whose correlation is given by the GRBF kernel and whose constant variance is $\sigma_Y^2 = \mathrm{Var}[Y(\mathbf{x})]$. This allows to describe the mean and variance of the GRBF prediction. Let $\mathbf{\Psi}\left(\mathbf{x}^{(0)}\right)$ be the vector containing the evaluations and first derivatives of the RBF kernels $\Psi_{0i,j}$ (from Eq. (55)) at the new point $\mathbf{x}^{(0)}$. By solving Eq. (62) for the weights, the GRBF estimation is expressed as a linear combination of the true responses and their derivatives,

$$\widetilde{Y}(\mathbf{x}^{(0)}) = \mathbf{\Psi}\left(\mathbf{x}^{(0)}\right)^\top \mathbf{\Psi}_g^{-1} \mathbf{Y}_g \ . \tag{71}$$

A mean and variance expressions are then calculated in a manner similar to kriging:

$$\begin{aligned}
&\forall \mathbf{x}^{(0)} \in \mathcal{D}, \\
&\tilde{y}\left(\mathbf{x}^{(0)}\right) = \mathbb{E}\left[\widetilde{Y}(\mathbf{x}^{(0)})\right] = \mathbf{\Psi}\left(\mathbf{x}^{(0)}\right)^\top \mathbf{\Psi}_g^{-1} \mathbb{E}[\mathbf{Y}_g] = \mathbf{y}_g^\top \mathbf{\Psi}_g^{-1} \mathbf{\Psi}\left(\mathbf{x}^{(0)}\right), && (72) \\
&s^2_{GRBF} = \mathbb{E}\left[\widetilde{Y}(\mathbf{x}^{(0)}) - Y(\mathbf{x}^{(0)})\right]^2 \\
&\qquad = \sigma_Y^2 \left(1 - \mathbf{\Psi}\left(\mathbf{x}^{(0)}\right)^\top \mathbf{\Psi}_g^{-1} \mathbf{\Psi}\left(\mathbf{x}^{(0)}\right)\right) \ . && (73)
\end{aligned}$$

The expression for the mean makes use of the further specification of the observations at the sample points, $\mathbb{E}[\mathbf{Y}_g] = \mathbf{y}_g$, i.e., one considers the conditional process Y knowing the observations at the sample points.

This variance could be used in infill criteria such as the *expected improvement* [33]. Unfortunately, as was said earlier, this variance calculation will often fail due to loss of positiveness of the GRBF matrix $\mathbf{\Psi}_g$ unless specific measures are undertaken.

8 Gradient-enhanced cokriging (GKRG)

Kriging, an alternative name for conditional Gaussian Processes, is today one of the main techniques for approximating functions and optimizing expensive to calculate functions. Cokriging is an extension of kriging for dealing with several correlated functions. Initially introduced for geostatistics [34, 35], many works focus on the assumptions, principles and formulations of cokriging [36–38, 42]. Gradient-based cokriging was introduced by Morris et al. [39] as a way to account for gradient information in kriging, and has since then been applied to many fields. Table 5 summarizes the references and the kind of applications that concern *gradient-enhanced cokriging*. Because the concepts underlying gradient-enhanced cokriging have received various names, the last column of the Table lists the original keywords employed by the cited authors. It is seen that gradient-enhanced cokriging has been largely used in the context of fluid problems. The efforts made to calculate gradients in fluid simulations explain this observation.

8.1 Formulation of gradient-enhanced cokriging

Gradient-enhanced cokriging is very similar to the classical kriging approach. Random processes associated with the deterministic objective function and its gradients are first defined through the primary

References	Contents	Original Keywords
[39]	Initial developments for taking into account gradients, application to water flow through a borehole	*Bayesian prediction using derivatives, Gaussian Process*
[40]	Approximation of functions using gradients	*Kriging to model gradients*
[41]	Theoretical developments and application to analytical functions	*First Order Kriging*
[43, 44]	Gradient-based metamodel for minimizing the drag of an aerofoil (CFD)	*Direct and indirect cokriging*
[45]	Gradient-based cokriging for optimization, infill strategy and application to structural optimization	*Kriging model including derivative information*
[46]	Comparison with other gradient-enhanced metamodels and application to fluids	*Kriging, gaussian process including derivative*
[21]	Developments for using gradient and hessian information (code available)	*Gradient-/Hessian-Enhanced Kriging*
[47]	Application to aerodynamic optimization	*Cokriging, Gradient-Enhanced Kriging*
[48]	Comparison of kriging with and without gradient, infill strategy and application to aerodynamic optimization	*Cokriging*
[49]	Uncertainty quantification and application to analytical and CFD examples	*Gradient-Enhanced Kriging (GEK)*
[50]	Application to aircraft aerodynamic shape optimization	*Gradient-Based Kriging (GBK)*
[51]	Uncertainty quantification, approximation quality of analytical functions and application to design of nuclear plants	*Gradient-Enhanced Universal Kriging (GEUK)*
[52]	Application to structural and aerodynamic optimization with multi-fidelity approach	*Cokriging*
[53]	Taking into account gradient and hessian information, application to analytical functions and aerodynamic optimization problems	*Gradient/Hessian-enhanced Direct/Indirect Kriging (GEK)*
[55]	Approximation quality of analytical functions, uncertainty quantification of a transonic aerofoil	*Cokriging, Gradient and Hessian enhanced Kriging*
[54, 107]	Comparison of gradient-based metamodels, application to analytical functions and CFD problems for shape optimization	*Direct/Indirect Co-kriging*
[60]	Multi-fidelity approach to aerofoil design	*direct Gradient-Enhanced Kriging (GEK)*
[61]	Comparison between direct and indirect gradient-based kriging using an analytical function and airfoil model. Study of the internal parameters estimation by Likelihood maximization.	*direct/indirect gradient-enhanced kriging*
[56–59, 95, 109]	Application to assembly design (nonlinear structural problems), comparison with gradient-based RBF, comparison with multi-fidelity method	*Gradient-Enhanced/-Based cokriging*
[62, 63, 110]	Gradient-enhanced kriging with and without multi-fidelity models	*Gradient-Enhanced Kriging*

Table 5: Summary of works using gradient-enhanced cokriging.

response, Y, and the n_p auxiliary responses, W^i [35]:

$$\begin{aligned}
&\forall i \in [\![1, n_p]\!], \forall \mathbf{x}^{(0)} \in \mathcal{D}, \\
&Y\left(\mathbf{x}^{(0)}\right) = \mu_0\left(\mathbf{x}^{(0)}\right) + Z_0\left(\mathbf{x}^{(0)}\right), \qquad (74)\\
&W^i\left(\mathbf{x}^{(0)}\right) = \mu_i\left(\mathbf{x}^{(0)}\right) + Z_i\left(\mathbf{x}^{(0)}\right). \qquad (75)
\end{aligned}$$

In the particular case of gradient-enhanced cokriging, the auxiliary responses W^i correspond to the components of the gradient:

$$\forall i \in [\![1, n_p]\!],\ \forall \mathbf{x}^{(0)} \in \mathcal{D},\ W^i\left(\mathbf{x}^{(0)}\right) = \frac{\partial Y}{\partial x_i}(\mathbf{x}^{(0)}). \qquad (76)$$

As in regular kriging, μ_i and Z_i represent, for each response, the deterministic trends and the fluctuations around the trends:

$$\begin{aligned}
&\forall \mathbf{x}^{(0)} \in \mathcal{D}, \\
&\mathbb{E}\left[Y\left(\mathbf{x}^{(0)}\right)\right] = \mathbb{E}\left[\mu_0\left(\mathbf{x}^{(0)}\right) + Z_0\left(\mathbf{x}^{(0)}\right)\right] = \mu_0\left(\mathbf{x}^{(0)}\right), \qquad (77)\\
&\sigma_Y^2 = \mathrm{Var}\left[Y\left(\mathbf{x}^{(0)}\right)\right] = \mathrm{Var}\left[\mu_0\left(\mathbf{x}^{(0)}\right) + Z_0\left(\mathbf{x}^{(0)}\right)\right] = \mathrm{Var}\left[Z_0\left(\mathbf{x}^{(0)}\right)\right] = \sigma_{Z_0}^2; \qquad (78)\\
&\forall i \in [\![1, n_p]\!], \forall \mathbf{x}^{(0)} \in \mathcal{D}, \\
&\mathbb{E}\left[Z_i(\mathbf{x}^{(0)})\right] = 0,\ \mathbb{E}\left[W^i\left(\mathbf{x}^{(0)}\right)\right] = \mu_i(\mathbf{x}^{(0)}), \qquad (79)\\
&\sigma_{W^i}^2 = \mathrm{Var}\left[W^i\left(\mathbf{x}^{(0)}\right)\right] = \mathrm{Var}\left[\mu_i\left(\mathbf{x}^{(0)}\right) + Z_i\left(\mathbf{x}^{(0)}\right)\right] = \mathrm{Var}\left[Z_i\left(\mathbf{x}^{(0)}\right)\right] = \sigma_{Z_i}^2. \qquad (80)
\end{aligned}$$

All Z_i's are centered stationary Gaussian Processes. As in usual kriging, the covariance of Z_0 is a function of a generalized distance among the sample points. Some other cross-covariance relations have to be introduced for the auxiliary variables. These covariances and cross-covariances are defined in Sections 8.3 and 8.5.

The trend models μ_i can be chosen independently of one another [111] and this choice leads to different kinds of (co)kriging (*simple* when μ_i is a known constant, *ordinary* when μ_i is an unknown constant and *universal* in the general case that it is both unknown and a function of $\mathbf{x}$). In this paper, the *universal* cokriging model where the trend is built using polynomial regression will be detailed, see Eq. (81)). In order to limit the amount of required inputs, the trend models of the auxiliary responses, μ_i, $i \in [\![1, n_p]\!]$, will be obtained by differentiation of the primary response trend, μ_0 (See [39] and Eq. (82)).

$$\begin{aligned}
&\forall \mathbf{x}^{(0)} \in \mathcal{D}, \\
&\mu_0\left(\mathbf{x}^{(0)}\right) = \sum_{j=1}^{n_t} \beta_j f_j\left(\mathbf{x}^{(0)}\right) = \mathbf{f}_0^\top \boldsymbol{\beta}, \qquad (81)\\
&\forall i \in [\![1, n_p]\!],\ \mu_i\left(\mathbf{x}^{(0)}\right) = \frac{\partial \mu_0}{\partial x_i}\left(\mathbf{x}^{(0)}\right) = \sum_{j=1}^{n_t} \beta_j \frac{\partial f_j}{\partial x_i}(\mathbf{x}^{(0)}) = \mathbf{f}_0^{iT} \boldsymbol{\beta}, \qquad (82)
\end{aligned}$$

where

$$\begin{aligned}
\boldsymbol{\beta} &= \begin{bmatrix} \beta_1 & \beta_2 & \dots & \beta_{n_t} \end{bmatrix}^\top; \\
\mathbf{f}_0 &= \begin{bmatrix} f_1(\mathbf{x}^{(0)}) & f_2(\mathbf{x}^{(0)}) & \dots & f_{n_t}(\mathbf{x}^{(0)}) \end{bmatrix}^\top; \\
\forall i \in [\![1, n_p]\!],\ \mathbf{f}_0^i &= \begin{bmatrix} \frac{\partial f_1}{\partial x_i}(\mathbf{x}^{(0)}) & \frac{\partial f_2}{\partial x_i}(\mathbf{x}^{(0)}) & \dots & \frac{\partial f_{n_t}}{\partial x_i}(\mathbf{x}^{(0)}) \end{bmatrix}^\top.
\end{aligned}$$

The *Best Linear Unbiased Predictor* (BLUP) of the response using both primary and auxiliary responses makes the cokriging model [35]. This predictor is a linear combination of deterministic functions called $\lambda()$'s and the evaluations of primary and auxiliary responses at the sample points:

$$\forall \mathbf{x}^{(0)} \in \mathcal{D},\ \hat{Y}\left(\mathbf{x}^{(0)}\right) = \sum_{i=1}^{n_s} \lambda_i^0\left(\mathbf{x}^{(0)}\right) Y\left(\mathbf{x}^{(i)}\right) + \sum_{i=1}^{n_s}\sum_{j=1}^{n_p} \lambda_i^j\left(\mathbf{x}^{(0)}\right) W^j\left(\mathbf{x}^{(i)}\right). \tag{83}$$

The functions $\lambda()$ are evaluated by minimizing the variance of the estimation error $\varepsilon\left(\mathbf{x}^{(0)}\right) = \hat{Y}\left(\mathbf{x}^{(0)}\right) - Y\left(\mathbf{x}^{(0)}\right)$ while accounting for the unbiasedness condition. Finally, the expressions of the cokriging prediction and variance are obtained. These steps are further explained in the next sections.

8.2 No bias condition

The condition for the cokriging estimator to be unbiased is

$$\begin{aligned}
&\forall \mathbf{x}^{(0)} \in \mathcal{D},\\
&\mathbb{E}\left[\hat{Y}\left(\mathbf{x}^{(0)}\right) - Y\left(\mathbf{x}^{(0)}\right)\right] = 0\\
&\mathbb{E}\left[\sum_{i=1}^{n_s} \lambda_i^0\left(\mathbf{x}^{(0)}\right) Y\left(\mathbf{x}^{(i)}\right) + \sum_{i=1}^{n_p}\sum_{j=1}^{n_p} \lambda_i^j\left(\mathbf{x}^{(0)}\right) W^j\left(\mathbf{x}^{(i)}\right) - Y\left(\mathbf{x}^{(0)}\right)\right] = 0\\
&\sum_{i=1}^{n_s} \lambda_i^0\left(\mathbf{x}^{(0)}\right)\mathbb{E}\left[Y\left(\mathbf{x}^{(i)}\right)\right] + \sum_{i=1}^{n_s}\sum_{j=1}^{n_p} \lambda_i^j\left(\mathbf{x}^{(0)}\right)\mathbb{E}\left[W^j\left(\mathbf{x}^{(i)}\right)\right] - \mathbb{E}\left[Y\left(\mathbf{x}^{(0)}\right)\right] = 0\\
&\sum_{i=1}^{n_s} \lambda_i^0\left(\mathbf{x}^{(0)}\right)\mu_0\left(\mathbf{x}^{(i)}\right) + \sum_{i=1}^{n_s}\sum_{j=1}^{n_p} \lambda_i^j\left(\mathbf{x}^{(0)}\right)\mu_j\left(\mathbf{x}^{(i)}\right) - \mu_0\left(\mathbf{x}^{(0)}\right) = 0.
\end{aligned} \tag{84}$$

Inserting the expression of the trend (Eq. (81) and (82)) leads to

$$\begin{aligned}
&\sum_{i=1}^{n_s} \lambda_i^0\left(\mathbf{x}^{(0)}\right) \sum_{k=1}^{n_t} \beta_k f_k\left(\mathbf{x}^{(i)}\right)\\
+&\sum_{i=1}^{n_s}\sum_{j=1}^{n_p} \lambda_i^j\left(\mathbf{x}^{(0)}\right) \sum_{k=1}^{n_t} \beta_k \frac{\partial f_k}{\partial x_j}(\mathbf{x}^{(i)}) - \sum_{k=1}^{n_t} \beta_k f_k\left(\mathbf{x}^{(0)}\right) = 0 \text{ , or,}\\
&\boldsymbol{\lambda}_0^\top \mathbf{F}\boldsymbol{\beta} + \boldsymbol{\lambda}_W^\top \mathbf{F}_W \boldsymbol{\beta} - \mathbf{f}_0^\top \boldsymbol{\beta} = 0.
\end{aligned} \tag{85}$$

with

$$\boldsymbol{\lambda}_0 = \begin{bmatrix}\lambda_1^0 & \lambda_2^0 & \dots & \lambda_{n_s}^0\end{bmatrix}^\top \qquad \text{size } 1 \times n_s;$$

$$\boldsymbol{\lambda}_W = \begin{bmatrix}\lambda_1^1 & \lambda_1^2 & \dots & \lambda_1^{n_p} & \lambda_2^1 & \dots & \lambda_{n_s}^{n_p}\end{bmatrix}^\top \qquad \text{size } 1 \times n_s n_p;$$

$$\mathbf{F} = \begin{bmatrix} f_1\big(\mathbf{x}^{(1)}\big) & f_2\big(\mathbf{x}^{(1)}\big) & \dots & f_{n_t}\big(\mathbf{x}^{(1)}\big) \\ f_1\big(\mathbf{x}^{(2)}\big) & \ddots & & \\ \vdots & & & \\ f_1\big(\mathbf{x}^{(n_s)}\big) & \dots & \dots & f_{n_t}\big(\mathbf{x}^{(n_s)}\big) \end{bmatrix} \qquad \text{size } n_t \times n_s;$$

$$\mathbf{F}_W = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}\big(\mathbf{x}^{(1)}\big) & \frac{\partial f_2}{\partial x_1}\big(\mathbf{x}^{(1)}\big) & \dots & \frac{\partial f_{n_t}}{\partial x_1}\big(\mathbf{x}^{(1)}\big) \\ \vdots & & & \vdots \\ \frac{\partial f_1}{\partial x_{n_p}}\big(\mathbf{x}^{(1)}\big) & \frac{\partial f_2}{\partial x_{n_p}}\big(\mathbf{x}^{(1)}\big) & \dots & \frac{\partial f_{n_t}}{\partial x_{n_p}}\big(\mathbf{x}^{(1)}\big) \\ \frac{\partial f_1}{\partial x_1}\big(\mathbf{x}^{(2)}\big) & \frac{\partial f_2}{\partial x_1}\big(\mathbf{x}^{(2)}\big) & \dots & \frac{\partial f_{n_t}}{\partial x_1}\big(\mathbf{x}^{(2)}\big) \\ \vdots & & & \vdots \\ \frac{\partial f_1}{\partial x_{n_p}}\big(\mathbf{x}^{(n_s)}\big) & \frac{\partial f_2}{\partial x_{n_p}}\big(\mathbf{x}^{(n_s)}\big) & \dots & \frac{\partial f_{n_t}}{\partial x_{n_p}}\big(\mathbf{x}^{(n_s)}\big) \end{bmatrix} \qquad \text{size } n_t \times n_s n_p.$$

Equation (85) can be further condensed after a simplification with respect to $\boldsymbol{\beta}$:

$$\boldsymbol{\lambda}_c \mathbf{F}_c = \mathbf{f}_0^\top, \tag{86}$$

where the vector $\boldsymbol{\lambda}_c = \begin{bmatrix}\boldsymbol{\lambda}_0^\top & \boldsymbol{\lambda}_W^\top\end{bmatrix}^\top$ includes $(n_p+1) \times n_s$ cokriging coefficients and $\mathbf{F}_c = \begin{bmatrix}\mathbf{F}^\top & \mathbf{F}_W^\top\end{bmatrix}^\top$ is a $n_t \times (n_s+1)n_s$ matrix. It should be remembered that $\boldsymbol{\lambda}_c$ and $\mathbf{f}_0$ depend on the non-sampled point $\mathbf{x}^{(0)}$. For simplicity's sake the functions $\lambda()$ have and will been written without specifying that they are defined only at the non-sampled point $\mathbf{x}^{(0)}$.

8.3 Formulation of the variance

The variance of the cokriging error estimate is

$$
\begin{aligned}
&\forall \mathbf{x}^{(0)} \in \mathcal{D},\\
s^2_{UCK}(\mathbf{x}^{(0)}) &= \mathrm{Var}\left[\hat{Y}\left(\mathbf{x}^{(0)}\right) - Y\left(\mathbf{x}^{(0)}\right)\right]\\
&= \mathrm{Var}\left[\hat{Y}\left(\mathbf{x}^{(0)}\right)\right] + \mathrm{Var}\left[Y\left(\mathbf{x}^{(0)}\right)\right] - 2\mathrm{cov}\left[\hat{Y}\left(\mathbf{x}^{(0)}\right), Y\left(\mathbf{x}^{(0)}\right)\right]\\
&= \mathrm{Var}\left[\sum_{i=1}^{n_s} \lambda_i^0(\mathbf{x}^{(0)}) Z_0\left(\mathbf{x}^{(i)}\right) + \sum_{i=1}^{n_s}\sum_{j=1}^{n_p} \lambda_i^j(\mathbf{x}^{(0)}) Z_j\left(\mathbf{x}^{(i)}\right)\right]\\
&+ \mathrm{Var}\left[Z_0\left(\mathbf{x}^{(0)}\right)\right]\\
&- 2\mathrm{cov}\left[\sum_{i=1}^{n_s} \lambda_i^0(\mathbf{x}^{(0)}) Z_0\left(\mathbf{x}^{(i)}\right) + \sum_{i=1}^{n_s}\sum_{j=1}^{n_p} \lambda_i^j(\mathbf{x}^{(0)}) Z_j\left(\mathbf{x}^{(i)}\right), Z_0\left(\mathbf{x}^{(0)}\right)\right]\\
&= \sigma^2_{Z_0} + \sum_{i=1}^{n_s}\sum_{j=1}^{n_p} \lambda_i^0\lambda_j^0 \mathrm{cov}\left[Z_0\left(\mathbf{x}^{(i)}\right), Z_0\left(\mathbf{x}^{(j)}\right)\right]\\
&+ \sum_{i=1}^{n_s}\sum_{k=1}^{n_s}\sum_{j=1}^{n_p}\sum_{l=1}^{n_p} \lambda_i^j\lambda_j^l \mathrm{cov}\left[Z_j\left(\mathbf{x}^{(i)}\right), Z_l\left(\mathbf{x}^{(k)}\right)\right]\\
&+ 2\sum_{i=1}^{n_s}\sum_{k=1}^{n_s}\sum_{j=1}^{n_p} \lambda_i^0\lambda_k^j \mathrm{cov}\left[Z_0\left(\mathbf{x}^{(i)}\right), Z_j\left(\mathbf{x}^{(k)}\right)\right]\\
&- 2\sum_{i=1}^{n_s}\sum_{j=1}^{n_p} \lambda_i^j \mathrm{cov}\left[Z_j\left(\mathbf{x}^{(i)}\right), Z_0\left(\mathbf{x}^{(0)}\right)\right]\\
&- 2\sum_{i=1}^{n_s} \lambda_i^0 \mathrm{cov}\left[Z_0\left(\mathbf{x}^{(i)}\right), Z_0\left(\mathbf{x}^{(0)}\right)\right].
\end{aligned}
$$

The following notations are introduced for simplifying the covariances :

$$
\begin{aligned}
&\forall (i,j,k,l) \in [\![0, n_s]\!]^2 \times [\![1, n_p]\!]^2,\\
&\mathrm{cov}\left[Z_0\left(\mathbf{x}^{(i)}\right), Z_0\left(\mathbf{x}^{(j)}\right)\right] = \mathrm{cov}\left[Y\left(\mathbf{x}^{(i)}\right), Y\left(\mathbf{x}^{(j)}\right)\right] = c_{ij},\\
&\mathrm{cov}\left[Z_0\left(\mathbf{x}^{(i)}\right), Z_k\left(\mathbf{x}^{(j)}\right)\right] = \mathrm{cov}\left[Y\left(\mathbf{x}^{(i)}\right), W^k\left(\mathbf{x}^{(j)}\right)\right] = c_{ij,k},\\
&\mathrm{cov}\left[Z_k\left(\mathbf{x}^{(i)}\right), Z_l\left(\mathbf{x}^{(j)}\right)\right] = \mathrm{cov}\left[W^k\left(\mathbf{x}^{(i)}\right), W^l\left(\mathbf{x}^{(j)}\right)\right] = c_{ij,kl}.
\end{aligned}
\tag{87}
$$

Now the variance of the cokriging error estimation can be written in matrix notation,

$$
\forall \mathbf{x}^{(0)} \in \mathcal{D},\ s^2_{UCK}(\mathbf{x}^{(0)}) = \sigma^2_{Z_0} + \boldsymbol{\lambda}_c^\top \mathbf{C}_c \boldsymbol{\lambda}_c - 2\boldsymbol{\lambda}_c^\top \mathbf{c}_{0c}, \tag{88}
$$

where $\mathbf{C}_c = \begin{bmatrix} \mathbf{C} & \mathbf{C}_{WY} \\ \mathbf{C}_{WY}^\top & \mathbf{C}_{WW} \end{bmatrix}$ is the cokriging covariance/cross-covariance matrix. It is composed of the classical kriging covariance matrix $\mathbf{C}$, the cross-covariance matrix $\mathbf{C}_{WY}$ made of covariances between primary and auxiliary responses and the cross-covariance matrix $\mathbf{C}_{WW}$ between the auxiliary responses.

Using the notations introduced in Eq. (87), these matrices are defined as:

$$(\mathbf{C})_{ij} = c_{ij};$$

$$\mathbf{C}_{WY} = \begin{bmatrix} c_{11,1} & c_{11,2} & \cdots & c_{11,n_p} & c_{12,1} & \cdots & c_{1n_s,n_p} \\ c_{21,1} & c_{21,2} & \cdots & c_{21,n_p} & c_{22,1} & \cdots & c_{2n_s,n_p} \\ c_{31,1} & c_{31,2} & \cdots & & & & \\ \vdots & & \ddots & & & & \vdots \\ c_{n_s 1,1} & \cdots & & & & & c_{n_s n_s,n_p} \end{bmatrix} \quad \text{size } n_s \times n_s n_p;$$

$$\mathbf{C}_{WW} = \begin{bmatrix} \mathbf{C}_{WW}^{11} & \mathbf{C}_{WW}^{12} & \cdots & \mathbf{C}_{WW}^{1n_s} \\ \mathbf{C}_{WW}^{21} & \mathbf{C}_{WW}^{22} & \cdots & \vdots \\ \vdots & & \ddots & \vdots \\ \mathbf{C}_{WW}^{n_s 1} & \mathbf{C}_{WW}^{n_s 2} & \cdots & \mathbf{C}_{WW}^{n_s n_s} \end{bmatrix} \quad \text{size } n_s n_p \times n_s n_p,$$

$$\text{with } \forall (k,l) \in [\![1, n_s]\!]^2,\ \mathbf{C}_{WW}^{kl} = \begin{bmatrix} c_{kl,11} & c_{kl,12} & \cdots & c_{kl,1n_p} \\ c_{kl,21} & c_{kl,22} & \cdots & \vdots \\ \vdots & & \ddots & \vdots \\ c_{kl,n_p 1} & c_{kl,n_p 2} & \cdots & c_{kl,n_p n_p} \end{bmatrix}.$$

The global cokriging covariance matrix $\mathbf{C}_c$ obtained is symmetric and contains $n_s(n_p+1)$ rows and columns. $\mathbf{c}_{0c}$ is the vector of covariances and cross-covariances between the sampled and any non-sampled points and it is expressed as $\mathbf{c}_{0c} = \begin{bmatrix} c_{10} & \cdots & c_{n_s 0} & c_{10,1} & c_{10,2} & \cdots & c_{20,1} & \cdots & c_{n_s 0,n_p} \end{bmatrix}^\top$ (size $1 \times n_s n_p$). The matrix $\mathbf{C}_c$ is positive definite. The proof is the following: $\forall \mathbf{v} \in \mathbb{R}^{n_s(n_p+1)}$, $\mathbf{v}^\top \mathbf{C}_c \mathbf{v} = \mathrm{Var}\left[\mathbf{v}^\top \begin{pmatrix} Y \\ W \end{pmatrix}\right] \geq 0$ since a variance is always positive. The matrix $\mathbf{\Psi}_g$ of GRBF (see Eq. (63)) is also positive definite because it has the same structure and is made of the same kernels. Above, positive definitness is not strict so that bad conditionning and even non invertibility may happen (e.g., when two sample points are identical).

8.4 Constrained optimization problem for cokriging building

Using the notations introduced in Eq. (86) and (88), a cokriging model is built by solving the following constrained optimization problem.

Problem 8.1 (Universal cokriging)**.** *Find* $\boldsymbol{\lambda}_c \in \mathbb{R}^{n_s(n_p+1)}$ *that minimizes*

$$\boldsymbol{\lambda}_c^\top \mathbf{C}_c \boldsymbol{\lambda}_c - 2\boldsymbol{\lambda}_c^\top \mathbf{c}_{0c} + \sigma_{Z_0}^2,$$

subject to $\mathbf{F}_c^\top \boldsymbol{\lambda}_c = \mathbf{f}_0$

Universal kriging and universal cokriging lead to the same constrained optimization problem. In the case of cokriging however, additional cross-covariances are taken into account. This constrained optimization problem is solved by the lagrangian technique which yields the following expressions for cokriging prediction and variance:

$$\forall \mathbf{x}^{(0)} \in \mathcal{D},$$

$$\tilde{y}_{UCK}\left(\mathbf{x}^{(0)}\right) = \left[\mathbf{c}_{0c} + \mathbf{F}_c \left(\mathbf{F}_c^\top \mathbf{C}_c^{-1} \mathbf{F}_c\right)^{-1} \left(\mathbf{f}_0 - \mathbf{F}_c^\top \mathbf{C}_c^{-1} \mathbf{c}_{0c}\right)\right]^\top \mathbf{C}_c^{-1} \mathbf{y}_g, \tag{89}$$

with $\mathbf{y}_g = \left[y_1 \quad \dots \quad y_{n_s} \quad \frac{\partial y_1}{\partial x_1} \quad \frac{\partial y_1}{\partial x_2} \quad \dots \quad \frac{\partial y_{n_s}}{\partial x_{n_p}} \right]^\top$, and

$$\forall \mathbf{x}^{(0)} \in \mathcal{D},\ s^2_{UCK}\left(\mathbf{x}^{(0)}\right) = \sigma^2_{Z_0} - \mathbf{c}_{0c}{}^\top \mathbf{C}_c^{-1} \mathbf{c}_{0c} + \mathbf{u}_0^\top \left(\mathbf{F}_c^\top \mathbf{C}_c^{-1} \mathbf{F}_c\right)^{-1} \mathbf{u}_0, \tag{90}$$
$$\text{with } \mathbf{u}_0 = \mathbf{u}\left(\mathbf{x}^{(0)}\right) = \mathbf{F}_c^\top \mathbf{C}_c^{-1} \mathbf{c}_{0c} - \mathbf{f}_0.$$

Like usual kriging, cokriging interpolates the responses at the data points by having the prediction equal to the response and the variance null there. The proof of this property is based on $\mathbf{c}_{0c}$ being equal to the ith column of $\mathbf{C}_c$ at $\mathbf{x}^{(0)} = \mathbf{x}^{(i)}$.

Simple and ordinary cokriging can be easily deduced from the previous equations by considering $\mu_0(\mathbf{x}) = m$ where m is a known real or $\mu_0(\mathbf{x}) = \beta$ where β is an unknow real. In both cases and according to Eq. (82), $\forall i \in [\![1, n_p]\!]$, $\mu_i(\mathbf{x}) = 0$. So, $\mathbf{F}_c = \begin{bmatrix} \mathbf{1}_{n_s}^\top & \mathbf{0}_{n_s \times n_p}^\top \end{bmatrix}^\top$ where $\mathbf{1}_{n_s}$ and $\mathbf{0}_{n_s \times n_p}$ are matrices containing n_s 1's and $n_s n_p$ 0's, respectively.

8.5 Covariance structure

The most critical choice when creating a cokriging model is that of the covariance functions. In applications such as geostatistics (see for instance [35]), this choice can be governed by expert information. In the more general context of computer experiments, there is a wide range of covariance functions to choose from. However, noting that covariance functions are kernel functions such as introduced in Section 6, multidimensional kernels can be formed by multiplying unidimensional kernels. Continuing this strategy for gradient-enhanced cokriging, Morris et al. [39] have proposed a general form for the cross-covariance relations:

$$\forall k \in [\![1, n_p]\!],\ \forall (a_k, b_k) \in \mathbb{N}^2,\ \forall (\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \in \mathcal{D},$$
$$\operatorname{cov}\left[Y^{(a_1, a_2, \dots, a_{n_p})}\left(\mathbf{x}^{(i)}\right), Y^{(b_1, b_2, \dots, b_{n_p})}\left(\mathbf{x}^{(j)}\right) \right] =$$
$$\sigma_Y^2 (-1)^{\sum b_j} \prod_{k=1}^{n_p} \left[h^{(a_k + b_k)}(x_k^{(i)} - x_k^{(j)}; \ell_k) \right], \tag{91}$$

where

$$Y^{(a_1, a_2, \dots, a_{n_p})}\left(\mathbf{x}^{(i)}\right) = \frac{\partial^{a_1 + a_2 \cdots + a_{n_p}} Y}{\partial x_1^{a_1} \partial x_2^{a_2} \dots \partial x_{n_p}^{a_{n_p}}}\left(\mathbf{x}^{(i)}\right),$$

and $h(r; \ell)$ is a unidimensional correlation function depending on the real r and the correlation length ℓ and $h^{(k)}$ is its k-th derivative.

Readers can note that kernels are even functions but their first derivative are odd, cf. for example Fig. 5. Therefore, referring to the covariance notation of Eq. (87), the following relation is found: $\forall (i, j, k) \in [\![1, n_s]\!]^2 \times [\![1, n_p]\!]$, $c_{ij,k} = -c_{ji,k}$. More generally, in the case of gradient-enhanced cokriging, the covariances satisfy,

$$\begin{aligned}
&\forall (i, j, k, l) \in [\![0, n_s]\!]^2 \times [\![1, n_p]\!]^2, \\
&\operatorname{cov}\left[Y\left(\mathbf{x}^{(i)}\right), Y\left(\mathbf{x}^{(j)}\right) \right] = c_{ij} = c_{ji} = \sigma_Y^2 \Psi\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\ell}\right), \\
&\operatorname{cov}\left[Y\left(\mathbf{x}^{(i)}\right), \frac{\partial Y}{\partial x_k}\left(\mathbf{x}^{(j)}\right) \right] = c_{ij,k} = -\sigma_Y^2 \frac{\partial \Psi}{\partial r_k}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\ell}\right), \\
&\operatorname{cov}\left[\frac{\partial Y}{\partial x_k}\left(\mathbf{x}^{(i)}\right), Y\left(\mathbf{x}^{(j)}\right) \right] = c_{ji,k} = \sigma_Y^2 \frac{\partial \Psi}{\partial r_k}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\ell}\right), \\
&\operatorname{cov}\left[\frac{\partial Y}{\partial x_k}\left(\mathbf{x}^{(i)}\right), \frac{\partial Y}{\partial x_l}\left(\mathbf{x}^{(j)}\right) \right] = c_{ij,kl} = -\sigma_Y^2 \frac{\partial^2 \Psi}{\partial r_k \partial r_l}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\ell}\right),
\end{aligned} \tag{92}$$

where $r_k = x_k^{(i)} - x_k^{(j)}$, and $\Psi\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\ell}\right) = \prod_{k=1}^{n_p} h(x_k^{(i)} - x_k^{(j)}; \ell_k) = \prod_{k=1}^{n_p} h(r_k; \ell_k)$ is the multidimensional correlation function.

In the literature, mainly squared exponential functions have been used for building kriging and cokriging approximations. Recently, many works [63, 86, 95, 96] have focused on Matérn [97] covariances, in particular Matérn $\frac{3}{2}$ and $\frac{5}{2}$ [63, 95]. Similarly to RBF and GRBF approximations, Matérn kernels improve the condition number of the covariance matrix, therefore improving the stability of the method.

With the product covariances introduced (see Eq. (92)), the process variance can be factored out of the different covariances in Eq. (89):

$$\mathbf{C}_c = \sigma_Y^2 \mathbf{K}_c; \tag{93}$$

$$\mathbf{c}_{0_c} = \sigma_Y^2 \mathbf{r}_{0_c}. \tag{94}$$

8.6 Summary of cokriging formulations and first illustrations

Table 6 summarizes the different cokriging formulations which look similar to kriging formulations with an extended definition of the correlation matrix and vector. If we only consider the metamodel predictions and not its variance or internal parameter learning, the functional forms of simple cokriging without trend and gradient-enhanced radial basis functions are identical (compare Eq. (71) with SCK where $m = 0$ in Table 6).

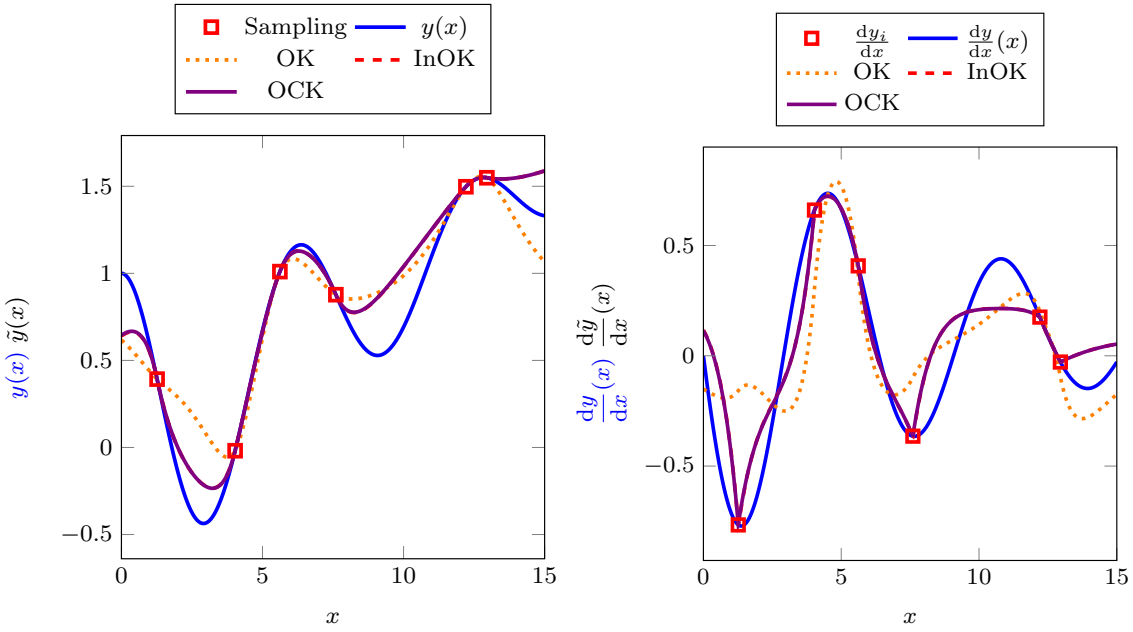
	Type	Formulation	with
$\tilde{y}_\bullet\left(\mathbf{x}^{(0)}\right)$	SCK	$m + \mathbf{r}_{0_c}^\top \mathbf{K}_c^{-1}\left(\mathbf{y}_g - m\mathbf{F}_{\mathbf{10}}^\top\right)$	–
	OCK	$\hat{\beta} + \mathbf{r}_{0_c}^\top \mathbf{K}_c^{-1}\left(\mathbf{y}_g - \hat{\beta}\mathbf{F}_{\mathbf{10}}^\top\right)$	$\hat{\beta} = \left(\mathbf{F}_{\mathbf{10}}^\top \mathbf{K}_c^{-1} \mathbf{F}_{\mathbf{10}}\right)^{-1} \mathbf{F}_{\mathbf{10}}^\top \mathbf{K}_c^{-1} \mathbf{y}_g$
	UCK	$\mathbf{f}_0^\top \hat{\boldsymbol{\beta}} + \mathbf{r}_{0_c}^\top \mathbf{K}_c^{-1}\left(\mathbf{y}_g - \mathbf{F}_c\hat{\boldsymbol{\beta}}\right)$	$\hat{\boldsymbol{\beta}} = \left(\mathbf{F}_c^\top \mathbf{K}_c^{-1} \mathbf{F}_c\right)^{-1} \mathbf{F}_c^\top \mathbf{K}_c^{-1} \mathbf{y}_g$
$s_\bullet^2\left(\mathbf{x}^{(0)}\right)$	SCK	$\sigma_Y^2\left(1 - \mathbf{r}_{0_c}^\top \mathbf{K}_c^{-1} \mathbf{r}_{0_c}\right)$	–
	OCK	$\sigma_Y^2\left(1 - \mathbf{r}_{0_c}^\top \mathbf{K}_c^{-1} \mathbf{r}_{0_c} + \mathbf{u}_0^\top \left(\mathbf{F}_{\mathbf{10}}^\top \mathbf{K}_c^{-1} \mathbf{F}_{\mathbf{10}}\right)^{-1} \mathbf{u}_0\right)$	$\mathbf{u}_0 = \mathbf{F}_{\mathbf{10}}^\top \mathbf{K}_c^{-1} \mathbf{r}_{0_c} - 1$
	UCK	$\sigma_Y^2\left(1 - \mathbf{r}_{0_c}^\top \mathbf{K}_c^{-1} \mathbf{r}_{0_c} + \mathbf{u}_0^\top \left(\mathbf{F}_c^\top \mathbf{K}_c^{-1} \mathbf{F}_c\right)^{-1} \mathbf{u}_0\right)$	$\mathbf{u}_0 = \mathbf{F}_c^\top \mathbf{K}_c^{-1} \mathbf{r}_{0_c} - \mathbf{f}_0$

Table 6: Cokriging predictions and variances (SCK, Simple CoKriging; OCK, Ordinary CoKriging; and UCK, Universal CoKriging) with $\mathbf{F}_{\mathbf{10}} = [\mathbf{1}_{n_s}^\top \mathbf{0}_{n_s \times n_p}^\top]$.

The Figures 9, 10 and 11 illustrate how kriging, indirect kriging (the principle of indirect gradient-enhanced metamodels is presented in Section 3) and ordinary cokriging approximate one and two-dimensional functions. The indirect version of the gradient-enhanced cokriging is only proposed in 1D, in Figure 9, where it can be seen that it yields very accurate results (the line cannot be visually separated from the true function on the plot). The Figures 9, 10 and 11 show that, like for RBF approximation, the use of the gradient information improves the approximation of the analytical function, in particular for multimodal functions such as the Six-hump Camel Back in Fig. 11 .

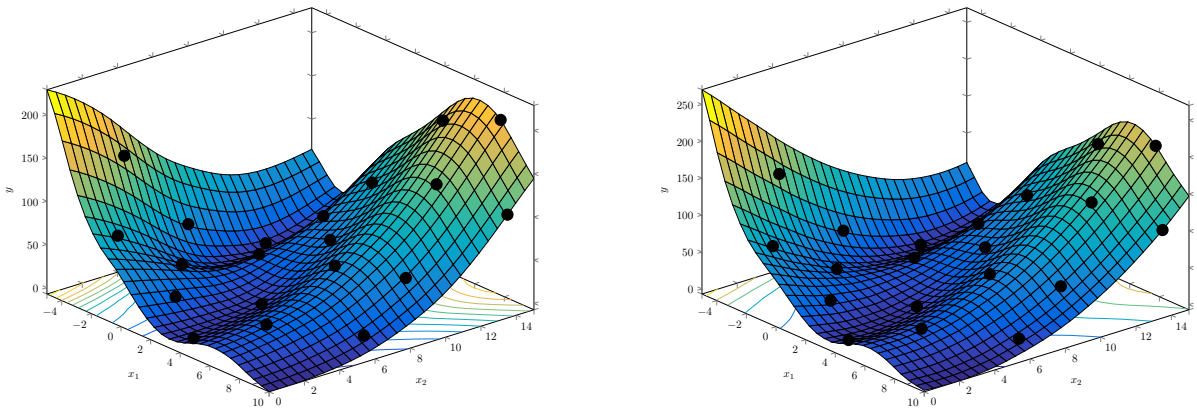
Figure 12 shows confidence intervals calculated with the predictions and the variances of ordinary kriging and cokriging. Remark that the use of the gradients reduces the approximation uncertainty.

When compared to GRBF, GKRG has the same covariance structure: $\mathbf{C}_c$ is the same matrix as $\boldsymbol{\Psi}_g$. Without trend and when the kernels are the same, the GRBF approximation of Eq. (72) is the same as that of GKRG (cf. Table 6). Differences arise because of the trend and the way the internal parameters are tuned. As a result, as will be observed in section 10, gradient-enhanced cokriging and GRBF have very similar performances with a slight advantage for the cokriging.



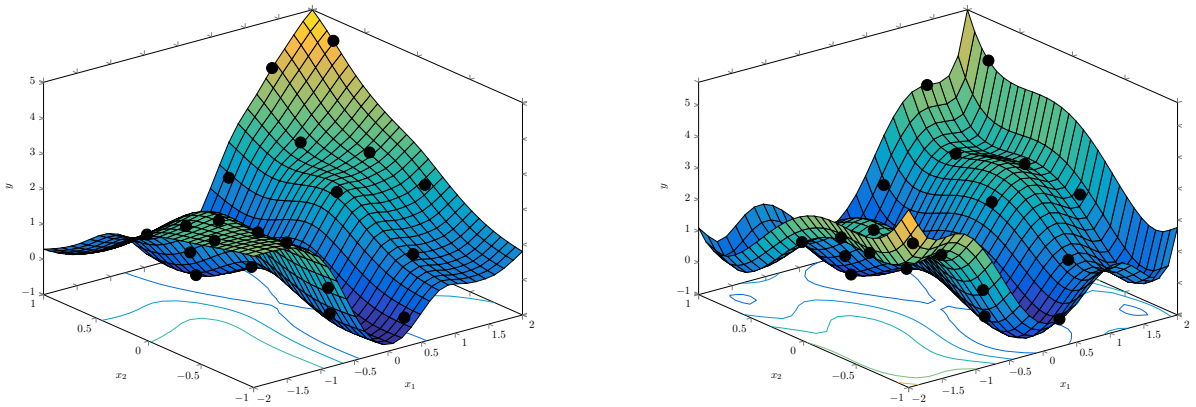
(a) Responses (b) Derivatives

Figure 9: Ordinary KRG, Indirect ordinary KRG (InOK) and ordinary GKRG (OCK) approximations of an unidimensional analytical function ($y(x) = \exp(-x/10)\cos(x) + x/10$, sampling with $n_s = 6$) using squared exponential function.



(a) Ordinary kriging (b) Ordinary gradient-enhanced cokriging

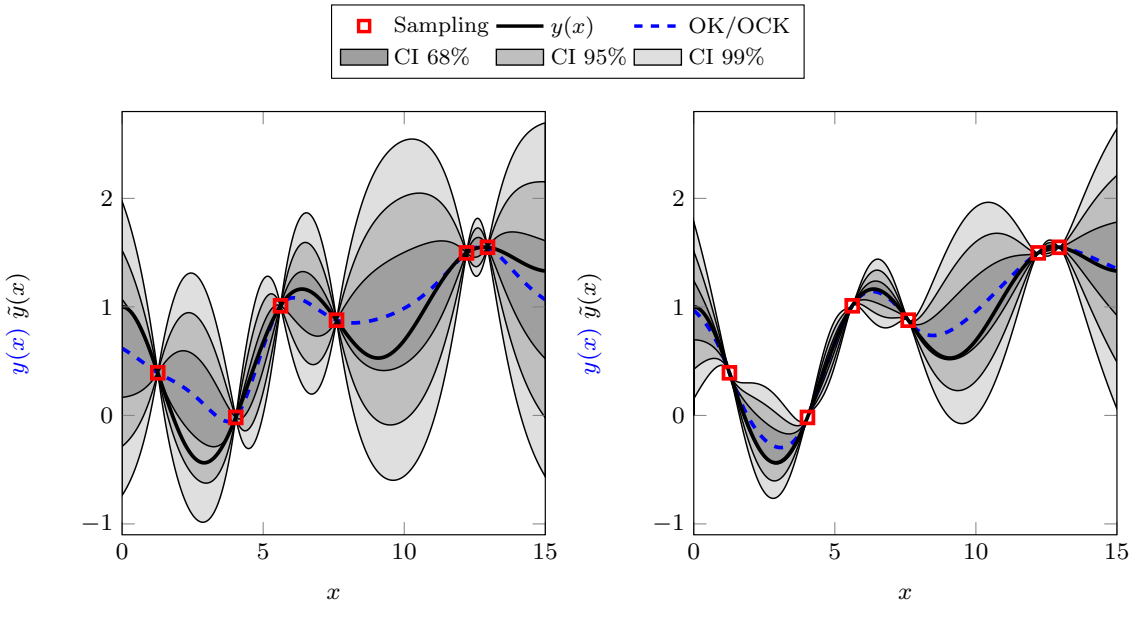
Figure 10: Ordinary kriging and gradient-enhanced cokriging approximations of the two-dimensional Branin's function (See Fig. 7a, IHS sampling with $n_s = 30$) using Matérn 3/2 kernels.



(a) Ordinary kriging

(b) Ordinary gradient-enhanced cokriging

Figure 11: Ordinary kriging and gradient-enhanced cokriging approximations of the two-dimensional Six-hump Camel Back function (See Fig. 8a, IHS sampling with $n_s = 20$) using Matérn 3/2 kernels.



(a) Kriging

(b) Cokriging

Figure 12: Confidence intervals of kriging (a) and cokriging (b), $n_s = 6$, $y(x)$ given in Fig. 9.

8.7 Derivatives of the cokriging approximation

Derivatives of the cokriging approximation to the response, $\frac{\partial \hat{Y}}{\partial x_i}\big(\mathbf{x}^{(0)}\big)$, $i = 1, ..., n_p$, can be obtained in two equivalent ways.

Firstly, Eq. (83) can be differentiated with respect to x_i which means taking the derivatives of the functions $\lambda()$. Substituting the expression of the $\lambda()$'s amounts to differentiating the correlation vectors $\mathbf{r}_{0c}$ (and the trend functions $\mathbf{f}_0$ for universal cokriging) in the expressions for the approximation $\tilde{y}()$ given in Table 6. To do so, the second derivatives of the kernel functions, which appear in the derivatives of $\mathbf{r}_{0c}$, are needed. The choice of the kernel must be adapted to this goal: squared exponential or Matérn (with $\nu > 1$) kernels are appropriate. It is remarkable that the second derivatives of the kernel functions were already required in the making of the cross-covariance matrix $\mathbf{C}_{WW}$, so approximating the derivative does not add requirements to the kernels.

Secondly and in an equivalent manner, the cokriging equations for predicting the response derivatives, $\frac{\partial \hat{Y}}{\partial x_i}\big(\mathbf{x}^{(0)}\big)$, can be obtained following the same path as that followed for the response: the cokriging estimate to the derivative is written as a linear combination of both the responses and their derivatives at sample points like in Eq. (83); the no bias condition of Eq. (84) is replaced by a no bias on the derivatives, $\mathbb{E}\left[\frac{\partial \hat{Y}}{\partial x_i}\big(\mathbf{x}^{(0)}\big) - \frac{\partial Y}{\partial x_i}\big(\mathbf{x}^{(0)}\big)\right] = 0$, and results in a relation like Eq. (86) with $\frac{\partial \mathbf{f}_0}{\partial x_i}$ instead of $\mathbf{f}_0$; similarly, the variance minimized is that of the error between derivatives, $\mathrm{Var}\left[\frac{\partial \hat{Y}}{\partial x_i}\big(\mathbf{x}^{(0)}\big) - \frac{\partial Y}{\partial x_i}\big(\mathbf{x}^{(0)}\big)\right]$, leading to an equation identical to Eq. (88) but $\mathbf{c}_{0c}$ is replaced by the vector $\frac{\partial \mathbf{c}_{0c}}{\partial x_i}$. Therefore, the cokriging models summarized in Table 6 provide models for the derivatives by just differentiating the trend and the correlation vectors. As a result in these (differentiated) models, the kriging interpolation property also applies to the derivatives. A notable feature of such gradient-enhanced cokriging is that the uncertainty of the estimated response derivative is also calculated [99]. This property was not used in previous works but it should turn out to be useful in the context of uncertainty quantification or reliability-based optimization.

8.8 Estimation of the cokriging parameters

As in the case of kriging, the estimation of the cokriging parameters, ℓ_i, σ_Y and $\boldsymbol{\beta}$ (and ν for the general Matérn kernel), can be achieved using *Leave-One-Out* or *Maximum likelihood* techniques. Leave-One-Out (LOO) was already introduced for GRBF in Section 7.3 and has also been applied to Gradient-Based cokriging (see for example [54]). The *Maximum likelihood* approach [112] is made possible by the probabilistic interpretation of cokriging and more common than LOO.

Maximum likelihood estimation operates by maximizing the following likelihood function (or minimizing the opposite of its log):

$$L(\boldsymbol{\beta}, \sigma_Y^2, \boldsymbol{\ell}) = \big(2\pi\sigma_Y^2\big)^{-n_s(n_p+1)/2} |\mathbf{K}_c(\boldsymbol{\ell})|^{-1/2} \exp\left[-\frac{1}{2\sigma_Y^2}\big(\mathbf{y}_g - \mathbf{F}\boldsymbol{\beta}\big)^\top \mathbf{K}_c(\boldsymbol{\ell})^{-1}\big(\mathbf{y}_g - \mathbf{F}\boldsymbol{\beta}\big)\right]. \quad (95)$$

At a given $\boldsymbol{\ell}$, $L()$ can be analytically maximized over $\boldsymbol{\beta}$ and σ_Y^2 which yields the expression of their estimates:

$$\hat{\boldsymbol{\beta}}(\boldsymbol{\ell}) = \Big(\mathbf{F}^\top \mathbf{K}_c(\boldsymbol{\ell})^{-1}\mathbf{F}\Big)^{-1} \mathbf{F}^\top \mathbf{K}_c(\boldsymbol{\ell})^{-1}\mathbf{y}_g; \quad (96)$$

$$\hat{\sigma}_Y^2(\boldsymbol{\ell}) = \frac{1}{n_s}\big(\mathbf{y}_g - \mathbf{F}\boldsymbol{\beta}\big)^\top \mathbf{K}_c(\boldsymbol{\ell})^{-1}\big(\mathbf{y}_g - \mathbf{F}\boldsymbol{\beta}\big). \quad (97)$$

The correlation lengths ℓ_i are obtained by a numerically minimizing the following expression which is the relevant part of minus the log-likelihood where $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}_Y^2$ have been input:

$$\hat{\boldsymbol{\ell}} = \underset{\boldsymbol{\ell}\in\mathbb{L}}{\arg\min}\ \psi(\boldsymbol{\ell}) \text{ where } \psi(\boldsymbol{\ell}) = \hat{\sigma}_Y^2(\boldsymbol{\ell})|\mathbf{K}_c(\boldsymbol{\ell})|^{1/n_s(n_p+1)}. \tag{98}$$

Because $\psi()$ is multimodal, it is essential to perform the minimization with a global optimization algorithm [113]: for example, a stochastic optimizer such as the Particle Swarm Optimizer can be employed [114]. In order to reduce the number of optimization iterations, the gradient of the likelihood function is sometimes calculated and accounted for in the optimization [63, 115]. During the numerical optimization for finding the correlation lengths, $\boldsymbol{\ell}$, the correlation matrix $\mathbf{K}_c$ has to be rebuilt, factorized and inverted at each iteration, which goes along with a noticeable computational cost. However, in most practical situations where metamodels are called in, the objective function relies on numerical simulation such as nonlinear finite elements and remains much more costly than the metamodel. Furthermore, the gain in accuracy of the gradient-based approximations allows in many cases to contain the computational time by reducing the necessary number of sample points [94, 95, 116].

9 Gradient-enhanced Support Vector Regression (GSVR)

Support Vector Regression (SVR) is a nonlinear regression method that appeared within the framework of statistical learning theory. It is an extension of the Support Vector Machines (SVMs) originally designed for nonlinear classification [117] and pattern recognition [118].

The literature on SVR is already rich and general introductions may be found in [119–121]. Initially built for learning from function responses at sample points, many extensions of SVR to additionally account for derivatives have been proposed. In compliance with the rest of the text, we shall call them gradient-enhanced SVR or GSVR. Initially introduced in [68] with an iteratively re-weighted least squares procedure, GSVR has then been revisited, again with a least squares approach in [70], with regularized least squares in [71], and by the Twin SVR technique in [69, 73]. A general framework for incorporating prior knowledge in SVR which has been applied to function derivatives was also put forward in [72]. More recently, GSVR has been applied to shape optimization in CFD problems [54].

9.1 Building procedure

We now present the method introduced by [68] and applied in [54]. The approximation is built from a linear combination of the basis functions $\phi_i()$ and their derivatives (all of which are independent of the observations) added to a constant trend term μ. The ϑ's are the coefficients of the combination, and will be adjusted using the observed responses $\mathbf{y}_g$:

$$\begin{aligned}
&\forall \mathbf{x}^{(0)} \in \mathcal{D},\\
&\tilde{y}\left(\mathbf{x}^{(0)}\right) = \mu + \sum_{i=1}^{n_s} \vartheta_i \phi\left(\mathbf{x}^{(0)}, \mathbf{x}^{(i)}\right) + \sum_{j=1}^{n_p}\sum_{i=1}^{n_s} \vartheta_{ij} \frac{\partial \phi}{\partial x_j}\left(\mathbf{x}^{(0)}, \mathbf{x}^{(i)}\right)\\
&\qquad = \mu + \sum_{j=0}^{n_p}\sum_{i=1}^{n_s} \vartheta_{ij}\phi_{0i,j} \qquad (99)\\
&\qquad = \mu + \boldsymbol{\vartheta}^\top \boldsymbol{\phi}_g\left(\mathbf{x}^{(0)}\right), \qquad (100)
\end{aligned}$$

where $\boldsymbol{\vartheta}$ and $\boldsymbol{\phi}_g\left(\mathbf{x}^{(0)}\right)$ contain the $n_s \times (n_p + 1)$ coefficients and evaluations of the basis function and its derivatives at the sample points, respectively.

At this point, the expression of the approximation $\tilde{y}\left(\mathbf{x}^{(0)}\right)$ is the same as that in any least squares approach, cf. Eq. 19 for example with $\boldsymbol{\vartheta}$ and $\hat{\boldsymbol{\beta}}$, and $\boldsymbol{\phi}_g\left(\mathbf{x}^{(0)}\right)$ and $\mathbf{f}\left(\mathbf{x}^{(0)}\right)$ playing the same roles, respectively. However, the coefficients $\boldsymbol{\vartheta}$ will be calculated through a different approach and the a

priori functions $\phi_{i,j}(\mathbf{x})$ will never be used as such but will always occur in products and hence they will be indirectly specified through a kernel and its derivatives, cf. Section 9.2.

Support vector regression seeks to approximate the function responses, $y\left(\mathbf{x}^{(i)}\right)$, within a ε_0 accuracy and, additionally, GSVR requires the derivatives, $\dfrac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right)$, to be approximated within ε_k accuracy. The SVR approximation is made more stable to changes in data by minimizing the vector norm $\|\vartheta\|^2$ (cf. [117, 120] for explanations on how reducing $\|\vartheta\|^2$ makes the approximation less flexible, therefore more stable). These considerations lead the constrained convex quadratic optimization problem 9.1 where $\boldsymbol{\xi}^+, \boldsymbol{\xi}^-, \boldsymbol{\tau}^+$ and $\boldsymbol{\tau}^-$ are slack variables on the accuracies for avoiding problems with no feasible solution:

Problem 9.1 (GSVR as a minimization problem)**.** *Find* $\left(\boldsymbol{\vartheta}, \mu, \boldsymbol{\xi}^+, \boldsymbol{\xi}^-, \boldsymbol{\tau}^+, \boldsymbol{\tau}^-\right)$[a] *that minimize*

$$\frac{1}{2}\|\boldsymbol{\vartheta}\|^2 + \frac{\Gamma_0}{n_s}\sum_{i=1}^{n_s}\left(\xi^{+(i)} + \xi^{-(i)}\right) + \sum_{k=1}^{n_p}\frac{\Gamma_k}{n_s}\sum_{i=1}^{n_s}\left(\tau_k^{+(i)} + \tau_k^{-(i)}\right),$$

subject to

$$\forall (i,k) \in [\![1, n_s]\!] \times [\![1, n_p]\!], \quad \begin{cases} y\left(\mathbf{x}^{(i)}\right) - \boldsymbol{\vartheta}^\top \boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) - \mu & \leq \varepsilon_0 + \xi^{+(i)}, \\ \boldsymbol{\vartheta}^\top \boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) + \mu - y\left(\mathbf{x}^{(i)}\right) & \leq \varepsilon_0 + \xi^{-(i)}, \\ \dfrac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right) - \boldsymbol{\vartheta}^\top \dfrac{\partial \boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right) & \leq \varepsilon_k + \tau_k^{+(i)}, \\ \boldsymbol{\vartheta}^\top \dfrac{\partial \boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right) - \dfrac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right) & \leq \varepsilon_k + \tau_k^{-(i)}, \\ \xi^{+(i)}, \xi^{-(i)}, \tau_k^{+(i)}, \tau_k^{-(i)} & \geq 0. \end{cases}$$

[a] Recall that bold notations designate vectors. For example, $\boldsymbol{\xi}^+ = \left[\xi^{+(1)} \quad \xi^{+(2)} \quad \dots \quad \xi^{+(n_s)}\right]^\top$

The hyper-parameters of the method, $\Gamma_k,\ k = 0, \dots, n_p$, are user-defined penalty parameters that control the trade-off between approximation regularity (low $\|\boldsymbol{\vartheta}\|^2$) and approximation accuracy in response and derivatives of the response. Geometrically, the constraints on accuracy are tubes of half-width ε_i in the space of responses and derivatives outside of which the GSVR criterion is subject to a linear loss at a rate determined by the hyper-parameters Γ_k.

Problem 9.1 can be rewritten as a saddle-point problem involving a Lagrangian and positive Lagrange multipliers $\alpha^{\pm(i)}$, $\lambda_k^{\pm(i)}$, $\eta^{\pm(i)}$ and $\theta_k^{\pm(i)}$ (a.k.a., dual variables):

Problem 9.2 (GSVR as a saddle-point problem)**.** *Find* $\left(\boldsymbol{\vartheta}, \mu, \boldsymbol{\xi}^+, \boldsymbol{\xi}^-, \boldsymbol{\tau}^+, \boldsymbol{\tau}^-\right)$ *and* $\left(\boldsymbol{\alpha}^+, \boldsymbol{\alpha}^-, \boldsymbol{\lambda}^+, \boldsymbol{\lambda}^-, \boldsymbol{\eta}^+, \boldsymbol{\eta}^-, \boldsymbol{\theta}^+, \boldsymbol{\theta}^-\right)$ *that, respectively, minimize and maximize the Lagrangian*

$$
\begin{aligned}
L =& \frac{1}{2}\|\boldsymbol{\vartheta}\|^2 + \frac{\Gamma_0}{n_s}\sum_{i=1}^{n_s}\left(\xi^{+(i)} + \xi^{-(i)}\right) + \sum_{k=1}^{n_p}\frac{\Gamma_k}{n_s}\sum_{i=1}^{n_s}\left(\tau_k^{+(i)} + \tau_k^{-(i)}\right) \\
&- \sum_{i=1}^{n_s}\alpha^{+(i)}\left[\varepsilon_0 + \xi^{+(i)} - y\left(\mathbf{x}^{(i)}\right) + \boldsymbol{\vartheta}^\top\boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) + \mu\right] \\
&- \sum_{i=1}^{n_s}\alpha^{-(i)}\left[\varepsilon_0 + \xi^{-(i)} + y\left(\mathbf{x}^{(i)}\right) - \boldsymbol{\vartheta}^\top\boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) - \mu\right] \\
&- \sum_{k=1}^{n_p}\sum_{i=1}^{n_s}\lambda_k^{+(i)}\left[\varepsilon_k + \tau_k^{+(i)} - \frac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right) + \boldsymbol{\vartheta}^\top\frac{\partial\boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right)\right] \\
&- \sum_{k=1}^{n_p}\sum_{i=1}^{n_s}\lambda_k^{-(i)}\left[\varepsilon_k + \tau_k^{-(i)} + \frac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right) - \boldsymbol{\vartheta}^\top\frac{\partial\boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right)\right] \\
&- \sum_{i=1}^{n_s}\left(\eta^{+(i)}\xi^{+(i)} + \eta^{-(i)}\xi^{-(i)}\right) \\
&- \sum_{k=1}^{n_p}\sum_{i=1}^{n_s}\left(\theta_k^{+(i)}\tau_k^{+(i)} + \theta_k^{-(i)}\tau_k^{-(i)}\right).
\end{aligned}
$$

At a solution, the partial derivatives of the Lagrangian with respect to the primal variables have to vanish:

$$
\begin{aligned}
\frac{\partial L}{\partial \boldsymbol{\vartheta}} &= \boldsymbol{\vartheta} - \sum_{i=1}^{n_s}\left(\alpha^{+(i)} - \alpha^{-(i)}\right)\boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) \\
&\quad - \sum_{k=1}^{n_p}\sum_{i=1}^{n_s}\left(\lambda_i^{+(k)} - \lambda_i^{-(k)}\right)\frac{\partial\boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right) = \mathbf{0}; && (101)\\
\frac{\partial L}{\partial \mu} &= -\sum_{i=1}^{n_s}\left(\alpha^{+(i)} - \alpha^{-(i)}\right) = 0; && (102)\\
\frac{\partial L}{\partial \xi^{+(i)}} &= \frac{\Gamma_0}{n_s} - \alpha^{+(i)} - \eta^{+(i)} = 0 \qquad \forall i \in [\![1, n_s]\!]; && (103)\\
\frac{\partial L}{\partial \xi^{-(i)}} &= \frac{\Gamma_0}{n_s} - \alpha^{-(i)} - \eta^{-(i)} = 0 \qquad \forall i \in [\![1, n_s]\!]; && (104)\\
\frac{\partial L}{\partial \tau_k^{+(i)}} &= \frac{\Gamma_k}{n_s} - \lambda_k^{+(i)} - \theta_k^{+(i)} = 0 \qquad \forall (i,k) \in [\![1, n_s]\!] \times [\![1, n_p]\!]; && (105)\\
\frac{\partial L}{\partial \tau_k^{-(i)}} &= \frac{\Gamma_k}{n_s} - \lambda_k^{-(i)} - \theta_k^{-(i)} = 0 \qquad \forall (i,k) \in [\![1, n_s]\!] \times [\![1, n_p]\!]. && (106)
\end{aligned}
$$

$\boldsymbol{\vartheta}$, $\boldsymbol{\eta}^\pm$ and $\boldsymbol{\theta}^\pm$ can readily be solved for from Eq. (101) and Eq. (103)-(106) which leads to the constrained convex quadratic optimization Problem 9.3. Notice that the positive sign of the Lagrangian multipliers $\alpha^{\pm(i)}$, $\lambda_k^{\pm(i)}$, $\eta^{\pm(i)}$ and $\theta_k^{\pm(i)}$ produces the inequality constraints.

Problem 9.3 (GSVR as a Convex Constrained Quadratic Optimization)**.** *Find* $\left(\boldsymbol{\alpha}^+, \boldsymbol{\alpha}^-, \boldsymbol{\lambda}^+, \boldsymbol{\lambda}^-\right)$ *that minimize*

$$\frac{1}{2}\begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\\ \boldsymbol{\lambda}^+\\ \boldsymbol{\lambda}^-\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\Psi}_r & -\boldsymbol{\Psi}_r & \boldsymbol{\Psi}_{rd} & -\boldsymbol{\Psi}_{rd}\\ -\boldsymbol{\Psi}_r & \boldsymbol{\Psi}_r & -\boldsymbol{\Psi}_{rd} & \boldsymbol{\Psi}_{rd}\\ \boldsymbol{\Psi}_{rd}^\top & -\boldsymbol{\Psi}_{rd}^\top & \boldsymbol{\Psi}_{dd} & -\boldsymbol{\Psi}_{dd}\\ -\boldsymbol{\Psi}_{rd}^\top & \boldsymbol{\Psi}_{rd}^\top & -\boldsymbol{\Psi}_{dd} & \boldsymbol{\Psi}_{dd}\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\\ \boldsymbol{\lambda}^+\\ \boldsymbol{\lambda}^-\end{bmatrix} + \begin{bmatrix}-\mathbf{y}_s\\ \mathbf{y}_s\\ -\mathbf{y}_{gs}\\ \mathbf{y}_{gs}\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\\ \boldsymbol{\lambda}^+\\ \boldsymbol{\lambda}^-\end{bmatrix} + \begin{bmatrix}\varepsilon_0\mathbf{1}\\ \varepsilon_0\mathbf{1}\\ \boldsymbol{\varepsilon}\\ \boldsymbol{\varepsilon}\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\\ \boldsymbol{\lambda}^+\\ \boldsymbol{\lambda}^-\end{bmatrix}, \quad (107)$$

subject to

$$\begin{cases} \begin{bmatrix}\mathbf{1}\\ -\mathbf{1}\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\end{bmatrix} = 0, \\ \begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix} \le \begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\\ \boldsymbol{\lambda}^+\\ \boldsymbol{\lambda}^-\end{bmatrix} \le \begin{bmatrix}\Gamma_0/n_s\mathbf{1}\\ \Gamma_0/n_s\mathbf{1}\\ \boldsymbol{\Gamma}\\ \boldsymbol{\Gamma}\end{bmatrix}. \end{cases}$$

The vectors $\mathbf{y}_s$ and $\mathbf{y}_{gs}$ contain the responses and the derivatives of the actual function at the sample points, respectively. $\boldsymbol{\varepsilon}$ is made of the ε_k values ($\forall k \in [\![1, n_p]\!]$). The matrices $\boldsymbol{\Psi}_r$, $\boldsymbol{\Psi}_{rd}$ and $\boldsymbol{\Psi}_{dd}$ consist of the evaluations and derivatives of the kernel function (see Section 9.2), and $\boldsymbol{\Gamma}$ designates the vector of Γ_k/n_s.

Responses or derivatives that are inside their ε_k tube, that is, responses and derivatives for which the accuracy constraints are satisfied, do not impact the solution of any of the above problems and could be removed altogether from the metamodel building without changing the result. This is because the Lagrange multipliers associated to these points are both (upper and lower limit) null. On the opposite, responses or derivatives at points that have one non-zero Lagrange multiplier influence the metamodel and are called *support vectors*. The dual variables $\boldsymbol{\alpha}^\pm$ and $\boldsymbol{\lambda}^\pm$ are determined by solving the Constrained Quadratic Optimization Problem 9.3. Classical Quadratic Programming algorithms [120] such as the Interior Point algorithm can be applied. For dealing with large number of sample points and parameters, dedicated algorithms, such as *Sequential Minimal Optimization* [122], are preferable. In order to reduce the computational cost of GSVR, a few works introduce new formulations: Lázaro *et al.* propose the IRWLS algorithm [68]; Jayadeva *et al.* have devised a regularized least squares approach [71]; Khemchandani *et al.* have come up with the Twin SVR [73].

9.2 Kernel functions

Problem 9.3 has the variables $\mathbf{x}^{(i)}$ involved only through products of $\boldsymbol{\phi}()$ and their derivatives. In SVR also, a kernel is defined as the inner product $\Psi(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$. The "kernel trick" consists in not explicitely giving $\phi()$ but directly working with the kernel $\Psi(,)$. As was already stated in Section 6, any function with two inputs cannot be a kernel, it has to satisfy the Mercer's conditions (see [120]) in order to be continuous, symmetric and positive definite. In the case of GSVR, the basis functions intervene

in the following products:

$$\forall (i,j,k,l) \in [\![1,n_s]\!]^2 \times [\![1,n_p]\!]^2,$$

$$\boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right)^\top \boldsymbol{\phi}\left(\mathbf{x}^{(j)}\right) = \Psi\left(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\right) = \Psi_{ij}, \tag{108}$$

$$\frac{\partial \boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right)^\top \boldsymbol{\phi}\left(\mathbf{x}^{(j)}\right) = \frac{\partial \Psi}{\partial x_k^{(i)}}\left(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\right) = \Psi_{ij,k0}, \tag{109}$$

$$\boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right)^\top \frac{\partial \boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(j)}\right) = \frac{\partial \Psi}{\partial x_k^{(j)}}\left(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\right) = \Psi_{ij,0k}, \tag{110}$$

$$\frac{\partial \boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right)^\top \frac{\partial \boldsymbol{\phi}}{\partial x_l}\left(\mathbf{x}^{(j)}\right) = \frac{\partial^2 \Psi}{\partial x_k^{(i)} \partial x_l^{(j)}}\left(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\right) = \Psi_{ij,kl}. \tag{111}$$

Therefore, in addition to the Mercer's conditions, a kernel used in GSVR must be twice differentiable. Again, like in GRBF and GKRG, squared exponential or Matérn ($\nu > 1$) functions can be used as kernels for GSVR (a list of kernels has been given in Section 6). With the notations introduced in Eq. (108)-(111), the matrices present in Problem 9.3 can now be detailed:

$$(\boldsymbol{\Psi}_r)_{ij} = \Psi_{ij}; \tag{112}$$

$$\boldsymbol{\Psi}_{rd} = \begin{bmatrix} \Psi_{11,10} & \Psi_{11,20} & \dots & \Psi_{11,n_p0} & \Psi_{12,10} & \dots & \Psi_{1n_s,0n_p} \\ \Psi_{21,10} & \Psi_{21,20} & \dots & \Psi_{21,n_p0} & \Psi_{22,10} & \dots & \Psi_{2n_s,0n_p} \\ \vdots & & \ddots & \vdots & \vdots & \ddots & \vdots \\ \Psi_{n_s1,10} & \Psi_{n_s1,20} & \dots & \Psi_{n_s1,n_p0} & \Psi_{n_s2,10} & \dots & \Psi_{n_sn_s,0n_p} \end{bmatrix}; \tag{113}$$

$$\boldsymbol{\Psi}_{dd} = \begin{bmatrix} \Psi_{11,11} & \Psi_{11,12} & \dots & \Psi_{11,1n_p} & \Psi_{12,11} & \dots & \Psi_{1n_s,1n_p} \\ \Psi_{11,21} & \Psi_{11,22} & \dots & \Psi_{11,2n_p} & \Psi_{12,21} & \dots & \Psi_{1n_s,2n_p} \\ \vdots & & \ddots & \vdots & \vdots & \ddots & \vdots \\ \Psi_{11,n_p1} & \Psi_{11,n_p2} & \dots & \Psi_{11,n_pn_p} & \Psi_{12,n_p1} & \dots & \Psi_{1n_s,n_pn_p} \\ \Psi_{21,11} & \Psi_{21,12} & \dots & \Psi_{21,1n_p} & \Psi_{22,11} & \dots & \Psi_{2n_s,1n_p} \\ \vdots & & \ddots & \vdots & \vdots & \ddots & \vdots \\ \Psi_{n_s1,n_p1} & \Psi_{n_s1,n_p2} & \dots & \Psi_{n_s1,n_pn_p} & \Psi_{n_s2,n_p1} & \dots & \Psi_{n_sn_s,n_pn_p} \end{bmatrix}. \tag{114}$$

The sizes of matrices $\boldsymbol{\Psi}_r$, $\boldsymbol{\Psi}_{rd}$ and $\boldsymbol{\Psi}_{dd}$ are $n_s \times n_s$, $n_s \times n_s n_p$ and $n_s n_p \times n_s n_p$, respectively. The full matrix of kernel functions and their derivatives at the sample points in Problem 9.3 is square and contains $2n_s(1+n_p)$ rows.

9.3 Evaluating the GSVR metamodel

Solving the Convex Constrained Quadratic problem for $\boldsymbol{\alpha}^\pm$ and $\boldsymbol{\lambda}^\pm$ allows to calculate the ϑ's from Eq. (101). The GSVR response estimate at a new point $\mathbf{x}^{(0)}$ is then given by:

$$\forall \mathbf{x}^{(0)} \in \mathcal{D},$$

$$\begin{aligned} \tilde{y}\left(\mathbf{x}^{(0)}\right) &= \mu + \sum_{i=1}^{n_s} \left(\alpha^{+(i)} - \alpha^{-(i)}\right) \Psi\left(\mathbf{x}^{(i)},\mathbf{x}^{(0)}\right) \\ &\quad + \sum_{k=1}^{n_p}\sum_{i=1}^{n_s} \left(\lambda_k^{+(i)} - \lambda_k^{-(i)}\right) \frac{\partial \Psi}{\partial x_k^{(i)}}\left(\mathbf{x}^{(i)},\mathbf{x}^{(0)}\right) \\ &= \mu + \begin{bmatrix} \boldsymbol{\alpha}^+ \\ -\boldsymbol{\alpha}^- \\ \boldsymbol{\lambda}^+ \\ -\boldsymbol{\lambda}^- \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{\Psi}\left(\mathbf{x}^{(0)}\right) \\ \boldsymbol{\Psi}\left(\mathbf{x}^{(0)}\right) \\ \boldsymbol{\Psi}_d\left(\mathbf{x}^{(0)}\right) \\ \boldsymbol{\Psi}_d\left(\mathbf{x}^{(0)}\right) \end{bmatrix}, \end{aligned} \tag{115}$$

where $\boldsymbol{\Psi}\left(\mathbf{x}^{(0)}\right)$ and $\boldsymbol{\Psi}_d\left(\mathbf{x}^{(0)}\right)$ are the vectors of kernels functions $\left(\mathbf{x}^{(i)}, \mathbf{x}^{(0)}\right)$ and their derivatives, respectively. The derivative of the approximation given by the GSVR metamodel is obtained by differentiating Eq. (115). To be able to do this, the kernel function Ψ must be at least twice differentiable. The trend term, μ, has not been calculated yet. Its value stems from the Karush-Kuhn-Tucker conditions for the Convex Constrained Problem 9.3: at a solution, the products between the dual variables and the associated constraints vanish:

$$\forall (i,k) \in [\![1, n_s]\!] \times [\![1, n_p]\!],$$

$$\alpha^{+(i)}\left(\varepsilon_0 + \xi^{+(i)} - y\left(\mathbf{x}^{(i)}\right) + \boldsymbol{\vartheta}^\top \boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) + \mu\right) = 0, \tag{116}$$

$$\alpha^{-(i)}\left(\varepsilon_0 + \xi^{-(i)} + y\left(\mathbf{x}^{(i)}\right) - \boldsymbol{\vartheta}^\top \boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) - \mu\right) = 0, \tag{117}$$

$$\lambda_k^{+(i)}\left(\varepsilon_k + \tau_k^{+(i)} + \boldsymbol{\vartheta}^\top \frac{\partial \boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right)}{\partial x_k} - \frac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right)\right) = 0, \tag{118}$$

$$\lambda_k^{-(i)}\left(\varepsilon_k + \tau_k^{-(i)} - \boldsymbol{\vartheta}^\top \frac{\partial \boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right) + \frac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right)\right) = 0, \tag{119}$$

$$\xi^{+(i)}\left(\frac{\Gamma_0}{n_s} - \alpha^{+(i)}\right) = 0, \tag{120}$$

$$\xi^{-(i)}\left(\frac{\Gamma_0}{n_s} - \alpha^{-(i)}\right) = 0, \tag{121}$$

$$\tau_k^{+(i)}\left(\frac{\Gamma_k}{n_s} - \lambda_k^{+(i)}\right) = 0, \tag{122}$$

$$\tau_k^{-(i)}\left(\frac{\Gamma_k}{n_s} - \lambda_k^{-(i)}\right) = 0. \tag{123}$$

Eq. (116)-(117) and (120)-(121) are the same as those for the classical (non-gradient based) SVR for which the following conclusions hold:

- from Eq. (120) and (121), either $\left(\Gamma_0/n_s - \alpha^{\pm(i)}\right) = 0$ and $\xi^{\pm(i)} > 0$, or $\xi^{\pm(i)} = 0$ and $\Gamma_0/n_s > \alpha^{\pm(i)}$.
- from Eq. (101) for $\boldsymbol{\vartheta}$, Eq. (116) and (117), and because $\tilde{y}\left(\mathbf{x}^{(i)}\right)$ cannot be below and above $y\left(\mathbf{x}^{(i)}\right)$ at the same time so that, in terms of the dual variables $\alpha^{+(i)}\alpha^{-(i)} = 0$, μ can be calculated:

$$\mu = y\left(\mathbf{x}^{(i)}\right) - \boldsymbol{\vartheta}^\top \boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) + \varepsilon_0 \quad \text{if} \quad \alpha^{+(i)} = 0 \text{ and } \alpha^{-(i)} \in]0, \Gamma_0/n_s[, \tag{124}$$

or

$$\mu = y\left(\mathbf{x}^{(i)}\right) - \boldsymbol{\vartheta}^\top \boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) - \varepsilon_0 \quad \text{if} \quad \alpha^{-(i)} = 0 \text{ and } \alpha^{+(i)} \in]0, \Gamma_0/n_s[. \tag{125}$$

The above bounds on $\alpha^{\pm(i)}$ are enforced as constraints of the quadratic optimization problem.

9.4 Gradient-enhanced ν-SVR

The classical GSVR discussed so far requires choosing Γ_k and ε_k ($k \in [\![0, n_p]\!]$), and is sometimes called ε_k-GSVR. ε_k is typically taken as the standard deviation of the noise of the response data and its derivatives. Often though, there is no prior knowledge on an eventual noise on the response. Furthermore, if the ε_k's are taken small, there will be many non-zero Lagrange multipliers (i.e., among $\boldsymbol{\alpha}^+, \boldsymbol{\alpha}^-, \boldsymbol{\lambda}^+, \boldsymbol{\lambda}^-$), in other terms there will be many support vectors, and the SVR model will lose some of its "sparsity" in the sense that the ability to drop some of the terms when evaluating the metamodel (Eq. (115)) will decrease.

ν-GSVR is an alternative support vector regression model where the ε_k's are no longer given but calculated. ν-GSVR uses new scalars, $\nu_k \in [0,1]^{n_p+1}$, which act as upperbounds on the proportion of points that will be support vectors. This approach is inherited from ν-SVM (Support Vector Machines, [123, 124]), and has been compared with classical ε-SVR in [125]. The larger the ν_k's, the more the approximation is required to approach the data points and made flexible (or "complex" in an information theory sense). A ν-GSVR model solves the following optimization problem (compare with Problem 9.1):

Problem 9.4 (GSVR as optimization problem)**.** *Find* $(\boldsymbol{\vartheta}, \mu, \boldsymbol{\xi}^+, \boldsymbol{\xi}^-, \boldsymbol{\tau}^+, \boldsymbol{\tau}^-, \boldsymbol{\varepsilon}^a)$ *that minimize*

$$\frac{1}{2}\|\boldsymbol{\vartheta}\|^2 + \frac{\Gamma_0}{n_s}\left[\nu_0\varepsilon_0 + \sum_{i=1}^{n_s}\left(\xi^{+(i)} + \xi^{-(i)}\right)\right] + \sum_{k=1}^{n_p}\frac{\Gamma_k}{n_s}\left[\nu_k\varepsilon_k + \sum_{i=1}^{n_s}\left(\tau_k^{+(i)} + \tau_k^{-(i)}\right)\right],$$

subject to

$$\forall (i,k) \in [\![1,n_s]\!] \times [\![1,n_p]\!], \quad \begin{cases} y\left(\mathbf{x}^{(i)}\right) - \boldsymbol{\vartheta}^\top \boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) - \mu & \leq \varepsilon_0 + \xi^{+(i)}, \\ \boldsymbol{\vartheta}^\top \boldsymbol{\phi}\left(\mathbf{x}^{(i)}\right) + \mu - y\left(\mathbf{x}^{(i)}\right) & \leq \varepsilon_0 + \xi^{-(i)}, \\ \dfrac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right) - \boldsymbol{\vartheta}^\top \dfrac{\partial \boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right) & \leq \varepsilon_k + \tau_k^{+(i)}, \\ \boldsymbol{\vartheta}^\top \dfrac{\partial \boldsymbol{\phi}}{\partial x_k}\left(\mathbf{x}^{(i)}\right) - \dfrac{\partial y}{\partial x_k}\left(\mathbf{x}^{(i)}\right) & \leq \varepsilon_k + \tau_k^{-(i)}, \\ \xi^{+(i)}, \xi^{-(i)}, \tau_k^{+(i)}, \tau_k^{-(i)} & \geq 0, \\ \varepsilon_0, \varepsilon_k & \geq 0. \end{cases}$$

[a] As usual the boldface notation denotes the vector, $\boldsymbol{\varepsilon} = [\varepsilon_0, \ldots, \varepsilon_{n_p}]^\top$, which should not be mistaken with the errors in Section 4.

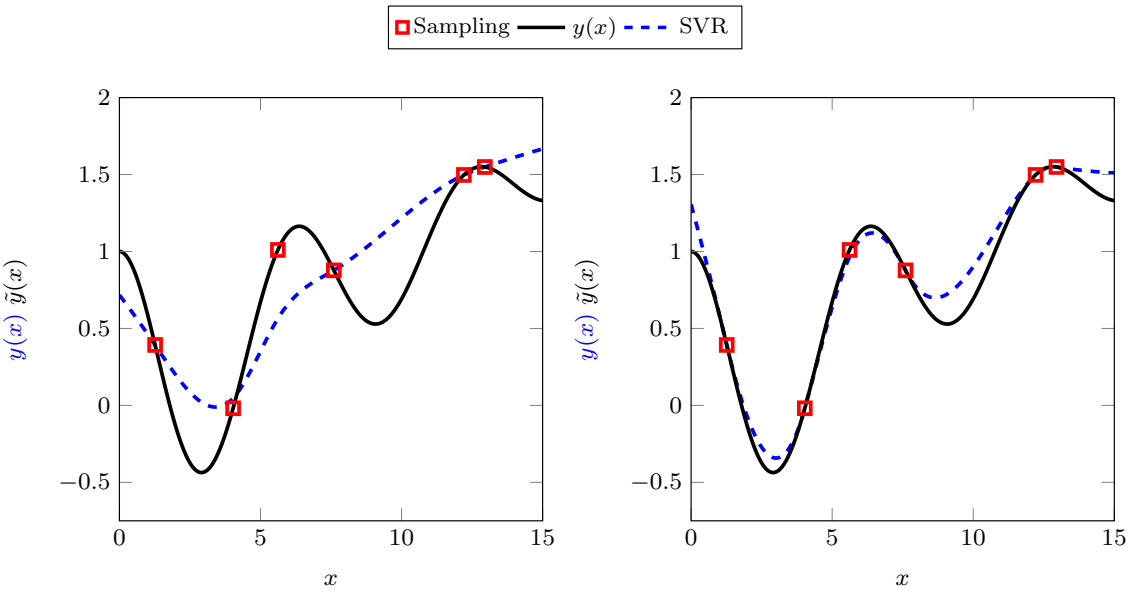
This problem is solved with a Lagrangian approach in a manner similar to that of ε_k-GSVR [54]. The $n_p + 1$ new constraints on the positivity of the ε_k's induce $n_p + 1$ new Lagrange multipliers. The resulting quadratic dual optimization problem is similar to that given as Problem 9.3 with the additional Lagrange multipliers. The ν-GSVR developped here has been implemented in the GRENAT Toolbox [12].

Figures 13 shows approximations of an analytical unidimensional function using ν-SVR, ν-GSVR and their derivatives (obtained by differentiating Eq. (115)). The filled areas correspond to the ε-tube of both approximations. Just as for the previous GRBF and GKRG metamodels, the gradient-enhanced ν-GSVR exhibits more accurate approximations than ν-SVR does.

9.5 Tuning GSVR parameters

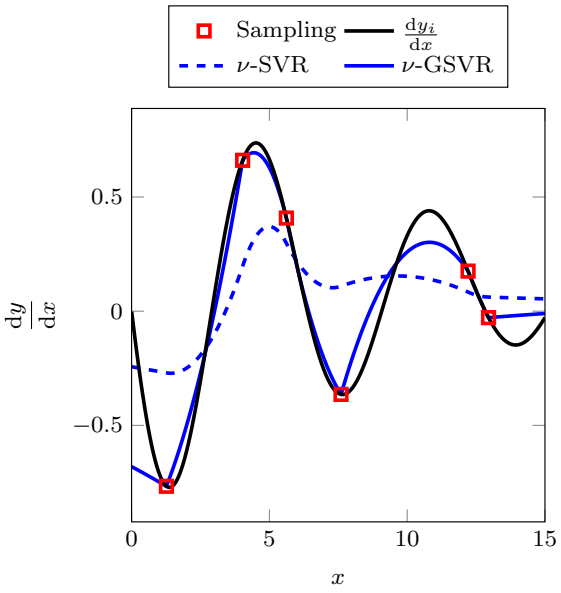
The GSVR model involves the same parameters are the version without gradients, that is the ε_i and Γ_i internal parameters, plus the parameters of the kernels $\boldsymbol{\ell}$. Several works, summarized in [126], discuss how to tune these parameters for non-gradient SVR, either in the form of empirical choices or of methodologies. Both ν-SVR and ν-GSVR (see Section 9.4) help in choosing the ε_i by replacing them by ν_i, a targeted proportion of points that are support vectors.

Algorithms have been proposed for determining the values of the aforementioned internal parameters using Leave-one-out bounds (Eq. (69)) for support vector regression. Introduced in [127], these bounds have been completed with the *Span* concept [128] that currently stands as the most accurate bound. A method for the minimization of this bound is described and studied in [129]. The gradient of the leave-one-out bound for SVR with respect to the internal parameters has been calculated in [130] and used for tuning the parameters.



(a) ν-SVR ($\nu_0 = 0.7$, $\Gamma_0 = 10^4$, $\ell = 0.13$)

(b) ν-GSVR ($\nu_0 = \nu_1 = 0.7$, $\Gamma_0 = \Gamma_1 = 10^4$, $\ell = 0.15$)



(c) Derivatives

Figure 13: Approximations of a unidimensional analytical function ($y(x) = \exp(-x/10)\cos(x) + x/10$) by ν-SVR and ν-GSVR ($n_s = 6$).

Recently, an extension of the *Span* bound to gradient-based SVR has been proposed in [54]: because the evaluation of the *Span* bound is computationally expensive, the authors have proposed to estimate the internal parameters of the kernel function as those of a gradient-enhanced RBF.

10 Applications and discussion

10.1 Procedure for comparing performances

Comparisons of response-only and gradient-enhanced metamodels will be carried out for modeling the 5 and 10 dimensional (n_p =5 or 10) *Rosenbrock* and *Schwefel* functions which are respectively defined as,

$$\forall \mathbf{x} \in [-2,2]^{n_p} \quad , \quad y(\mathbf{x}) = \sum_{i=1}^{n_p-1} \left[100 \left(x_i^2 - x_{i+1} \right)^2 + (x_i - 1)^2 \right] ; \tag{126}$$

$$\forall \mathbf{x} \in [-500,500]^{n_p} \quad , \quad y(\mathbf{x}) = 418.9829 + \sum_{i=1}^{n_p} x_i \sin\left(\sqrt{|x_i|} \right) . \tag{127}$$

Rosenbrock's function has only one basin of attraction but it is located in a long curved valley, that is the variables significantly interact with each other. Schwelfel's function is highly multimodal and, worse, the function is challenging for many surrogates in that the frequency and the amplitude of the sin() function that composes it changes through the design space, making the function non stationary. Schwefel's difficulty is nevertheless limited because it is an additively decomposable and smooth function.

Between $n_s = 5$ and $n_s = 140$ points are generated by *Improved Hypercube Sampling* [103]. Each sampling and metamodel building is repeated 50 times. The global approximation quality of the metamodel is measured by computing the mean value and the standard deviation of the R^2 and $\mathtt{Q}_3$ criteria for the 50 metamodels at $n_v = 1000$ validation points which are different from the sample points.

These metamodel quality criteria are now defined.

$$\begin{aligned} \mathrm{R}^2 &= \left(\frac{\sigma_{xy}}{\sigma_x \sigma_y} \right)^2 \quad \text{with} \quad \sigma_{xy} = \frac{1}{n_v} \sum_{i=1}^{n_v} \left(y\left(\mathbf{x}^{(i)}\right) - \overline{y} \right) \left(\tilde{y}\left(\mathbf{x}^{(i)}\right) - \overline{\tilde{y}} \right) , \\ \sigma_x &= \sqrt{\frac{1}{n_v} \sum_{i=1}^{n_v} \left(y\left(\mathbf{x}^{(i)}\right) - \overline{y} \right)^2} \, , \, \sigma_y = \sqrt{\frac{1}{n_v} \sum_{i=1}^{n_v} \left(\tilde{y}\left(\mathbf{x}^{(i)}\right) - \overline{\tilde{y}} \right)^2} . \end{aligned} \tag{128}$$

As usual, the $\bar{}$ symbol denotes the average. R^2 Pearson's correlation coefficient, measures how well the surrogate predictions are correlated to the true responses. The closer R^2 is to 1, the better. $\mathtt{Q}_3$ is a normalized leave-one-out criterion (cf. Eq. (69)):

$$\mathtt{Q}_3 = \frac{1}{n_v} \sum_{i=1}^{n_v} e_i \text{ with } e_i = \frac{\left(\tilde{y}_{-i}\left(\mathbf{x}^{(i)}\right) - y\left(\mathbf{x}^{(i)}\right) \right)^2}{\max\limits_{i \in [\![1,n_v]\!]} y\left(\mathbf{x}^{(i)}\right)^2} \tag{129}$$

The closer $\mathtt{Q}_3$ is to 0, the better the prediction accuracy of the surrogate.

The results presented next have been obtained on a computer equipped with an Intel® Xeon® E5-2680 v2 processor (20 cores at 2.8 GHz) and 128Gb of volatile memory (DDR3-1866). The execution times given are CPU times which correspond to the time required for running the computer code on a single processor loaded at 100%.

10.2 Comparison of LS and GradLS models

The response-only and gradient-enhanced least squares metamodels, LS and GradLS, are compared in details when approximating the 3 and 5 dimensional Rosenbrock's functions. The least squares fit are

carried out with polynomials of degrees $d^\circ = 1$ to 10. Fig. 15 and 16 show the results in terms of R^2 and Q_3 (mean and standard deviation) for the 3 dimensional function and Fig. 17 and 18 show the results for the 5 dimensional function. The approximation quality improves as the mean of R^2 increases and its standard deviation simultaneously decreases or, as the mean of Q_3 decreases and its standard deviation simultaneously decreases. In order to help understanding the outcome of the experiments, Fig. 14 summarizes both, *i)* the number of terms in the polynomials, n_t, which is a function of their degree as seen in Eq. (1) and, *ii)* the number of equations in the least squares approximations (Eq. (11)) which is equal to n_s and $n_s(n_p + 1)$ for the LS and GradLS models, respectively. The number of polynomial terms are plotted, for each n_p separately, with the continuous lines. The number of equations are plotted as marks, blue marks for the LS, and black marks with a dependency on n_p for GradLS: the dotted lines give the number of equations in GradLS as a function of n_p for each different n_s. The GradLS formulation uses more equations than LS does thanks to the gradients. As long as there are more independent equations than terms in the polynomial, the solution (21) to the Mean Squares Error exists and is unique. In this case in our implementation a QR factorization of the $\mathbf{F}^\top\mathbf{F}$ is performed. On the contrary, if the degree of the polynomial is such that there are more polynomial terms than equations, the problem is ill-posed and the matrix $\mathbf{F}^\top\mathbf{F}$ in Eq. (21) is no longer invertible. In our implementation of least squares, solution unicity is then recovered by using the Moore-Penrose pseudo-inverse[1] of $\mathbf{F}$, written $\mathbf{F}^+$, i.e., by solving $\hat{\boldsymbol{\beta}} = \mathbf{F}^+\mathbf{y}_g$. The portions of the solid lines that are below the marks associated to each n_s indicate the polynomial degrees for which there are sufficiently many equations to solve the original least squares problem, in other terms, the polynomials which are fully defined by the data points.

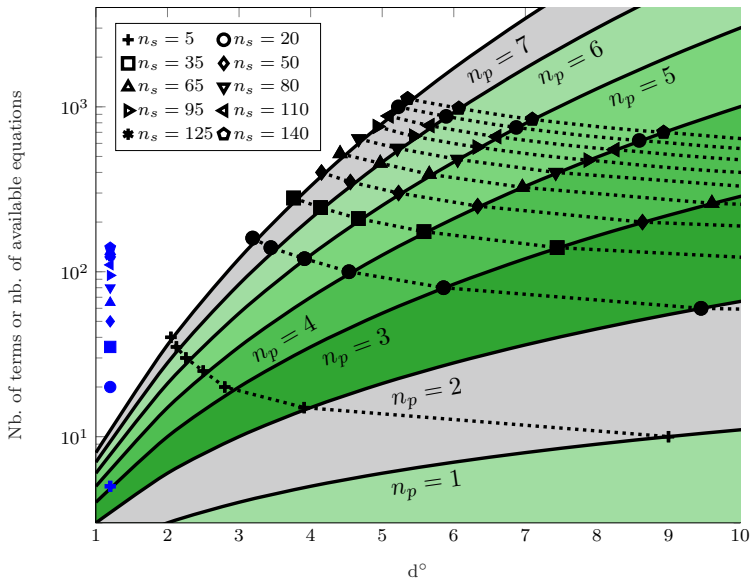
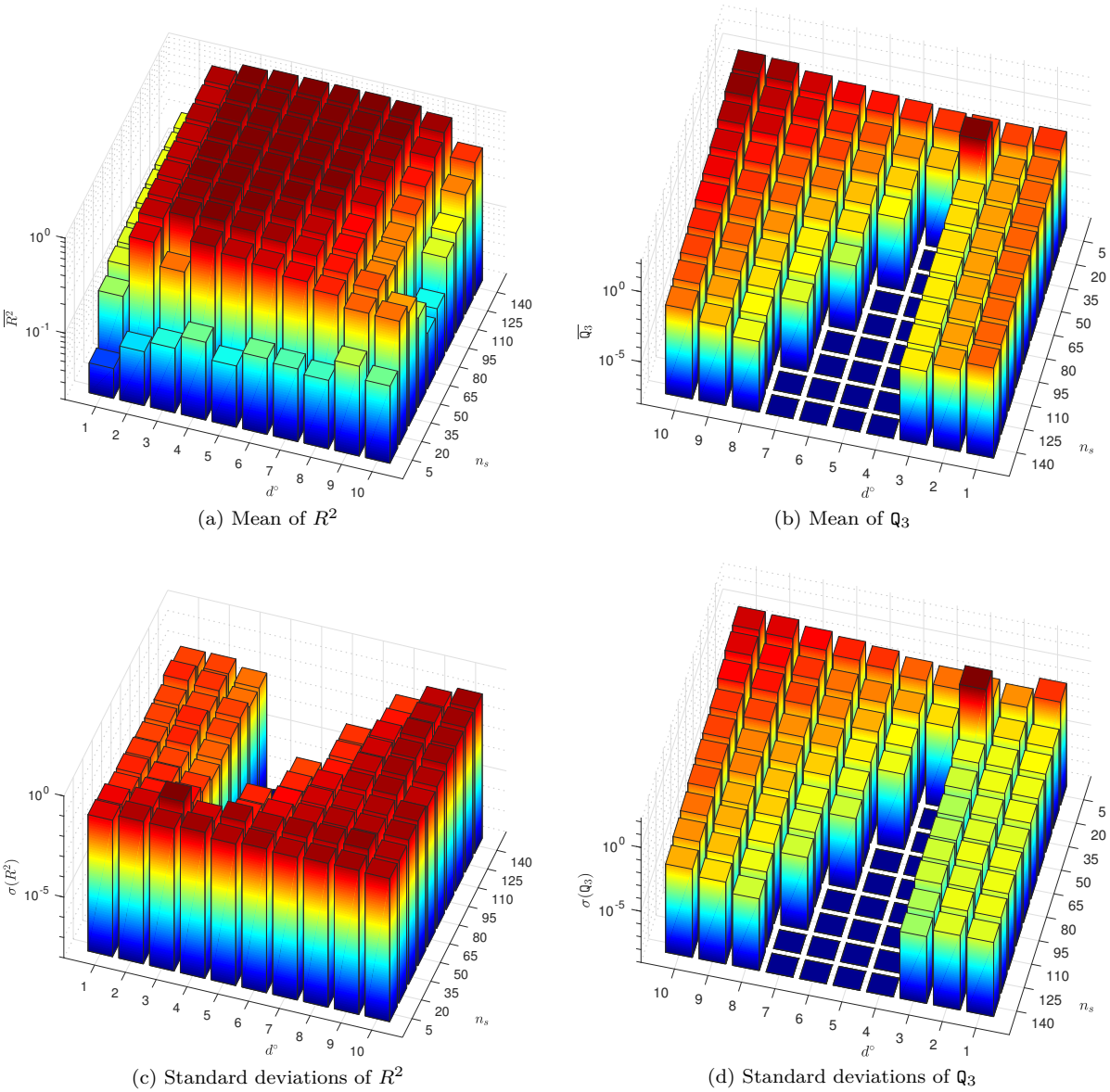


Figure 14: Number of terms in the polymonials, n_t, as a function of the polynomial degree (solid lines), and number of available equations (blue and black marks for LS and GradLS, respectively) as a function of the number of parameters n_p. The number of equations depends on n_p only for GradLS: the dotted lines join, for each number of sample points n_s, the number of equations for varying n_p.

General trends are visible in Figures 15 to 18 concerning Rosenbrock's function and in Figures 19 to 20 concerning Schwefel's function. They are particularly clear on the mean of Q_3, and they are confirmed by R^2. Not surprisingly, the quality of the approximations increases (i.e., the mean of Q_3 diminishes) with the number of sample points n_s; at a given n_s (larger than 5), the approximations improve from

[1] In the case where there are not enough equations, pseudo-inverse recovers solution unicity by choosing, out of the infinite number of solutions to $\mathbf{F}\boldsymbol{\beta} = \mathbf{y}_g$, the $\boldsymbol{\beta}$ of minimal euclidean norm.



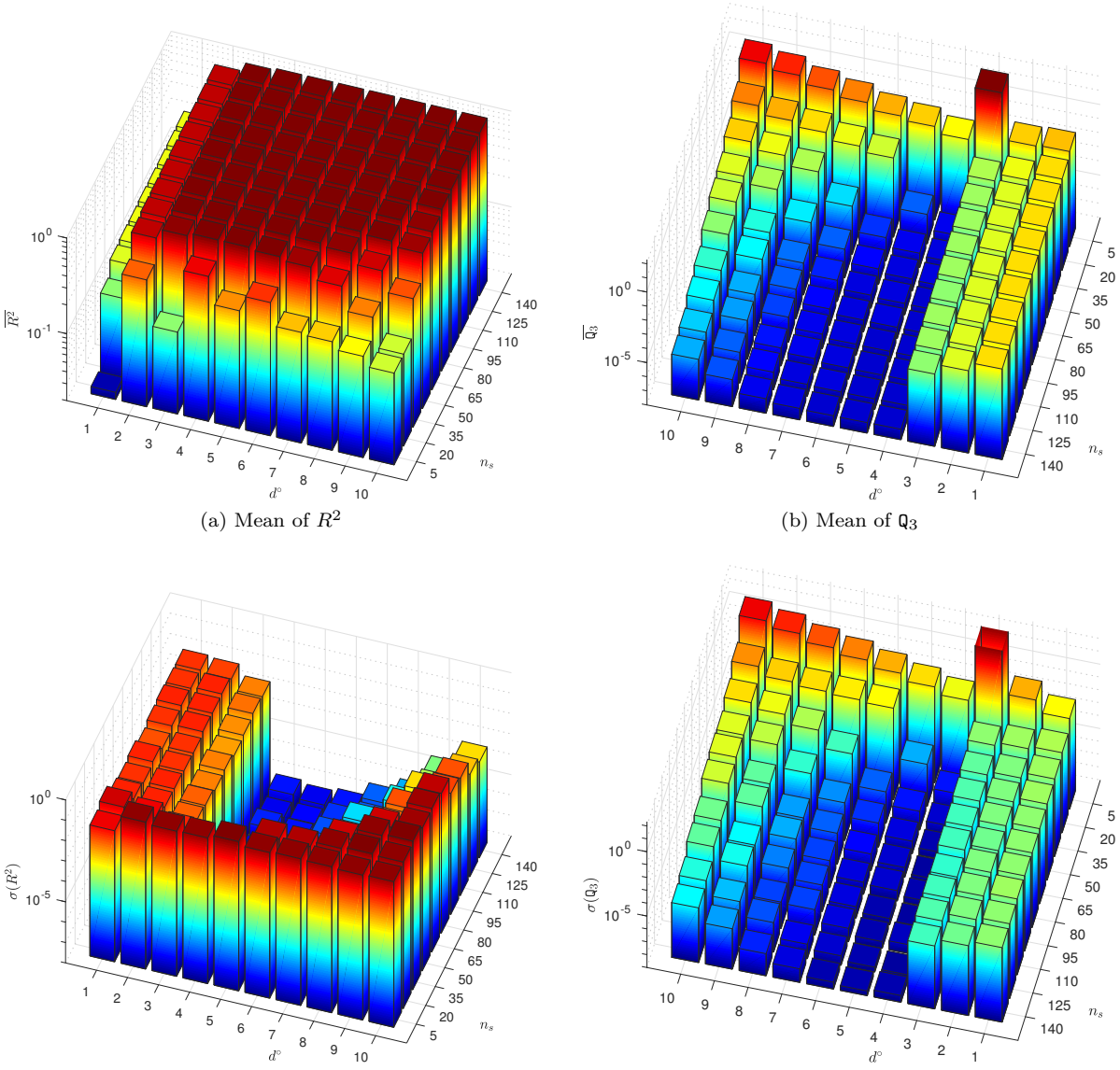
(a) Mean of R^2

(b) Mean of $\mathtt{Q}_3$

(c) Standard deviations of R^2

(d) Standard deviations of $\mathtt{Q}_3$

Figure 15: Performance of the LS metamodel in approximating the 3 dimensional Rosenbrock's function in terms of the number of sample points (n_s) and the degree of the polynomial (d°).



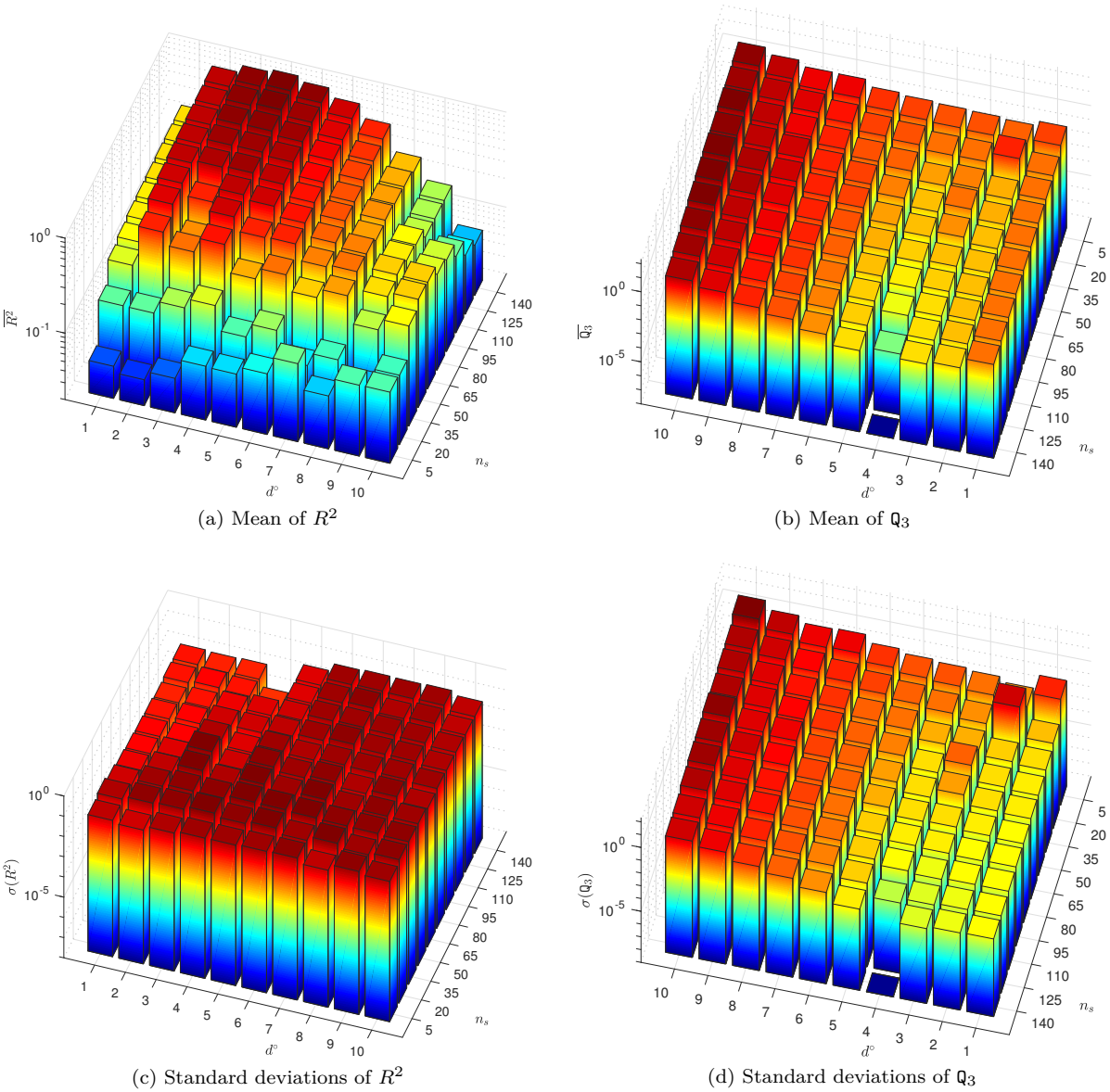
(a) Mean of R^2

(b) Mean of $\mathtt{Q}_3$

(c) Standard deviations of R^2

(d) Standard deviations of $\mathtt{Q}_3$

Figure 16: Performance of the GradLS metamodel in approximating the 3 dimensional Rosenbrock's function in terms of the number of sample points (n_s) and the degree of the polynomial (d°). Compare with the response-only LS metamodel in Fig. 15.



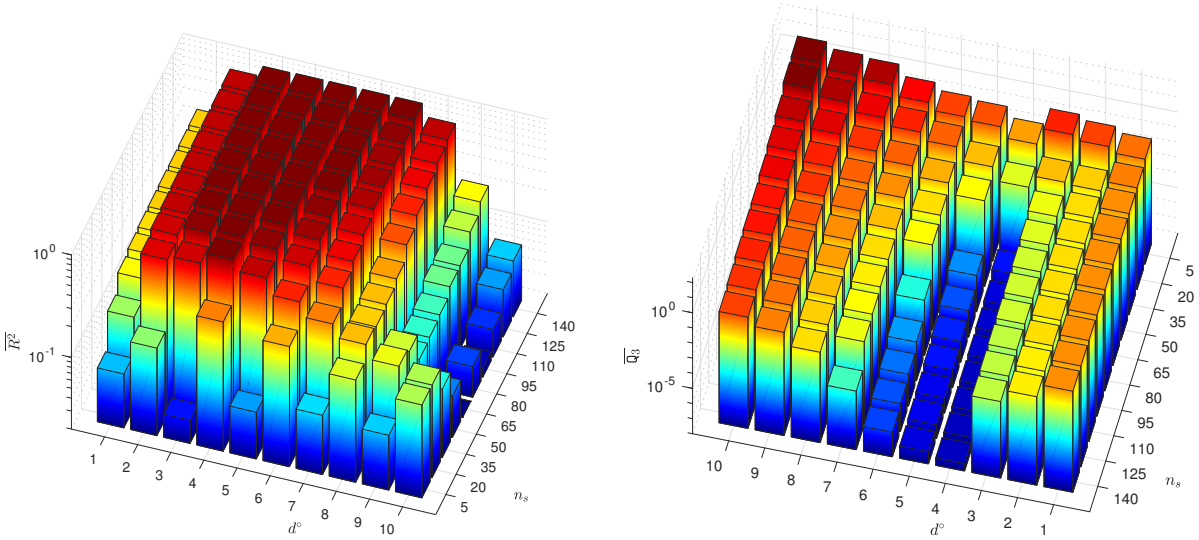
(a) Mean of R^2

(b) Mean of $\mathtt{Q}_3$

(c) Standard deviations of R^2

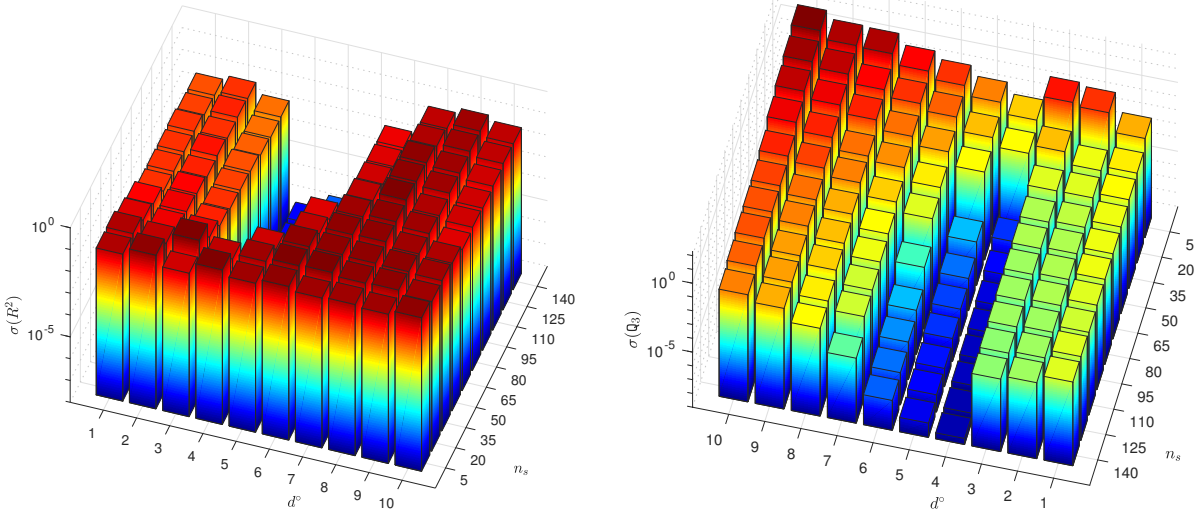
(d) Standard deviations of $\mathtt{Q}_3$

Figure 17: Performance of the LS metamodel in approximating the 5 dimensional Rosenbrock's function in terms of the number of sample points (n_s) and the degree of the polynomial (d°).



(a) Mean of R^2

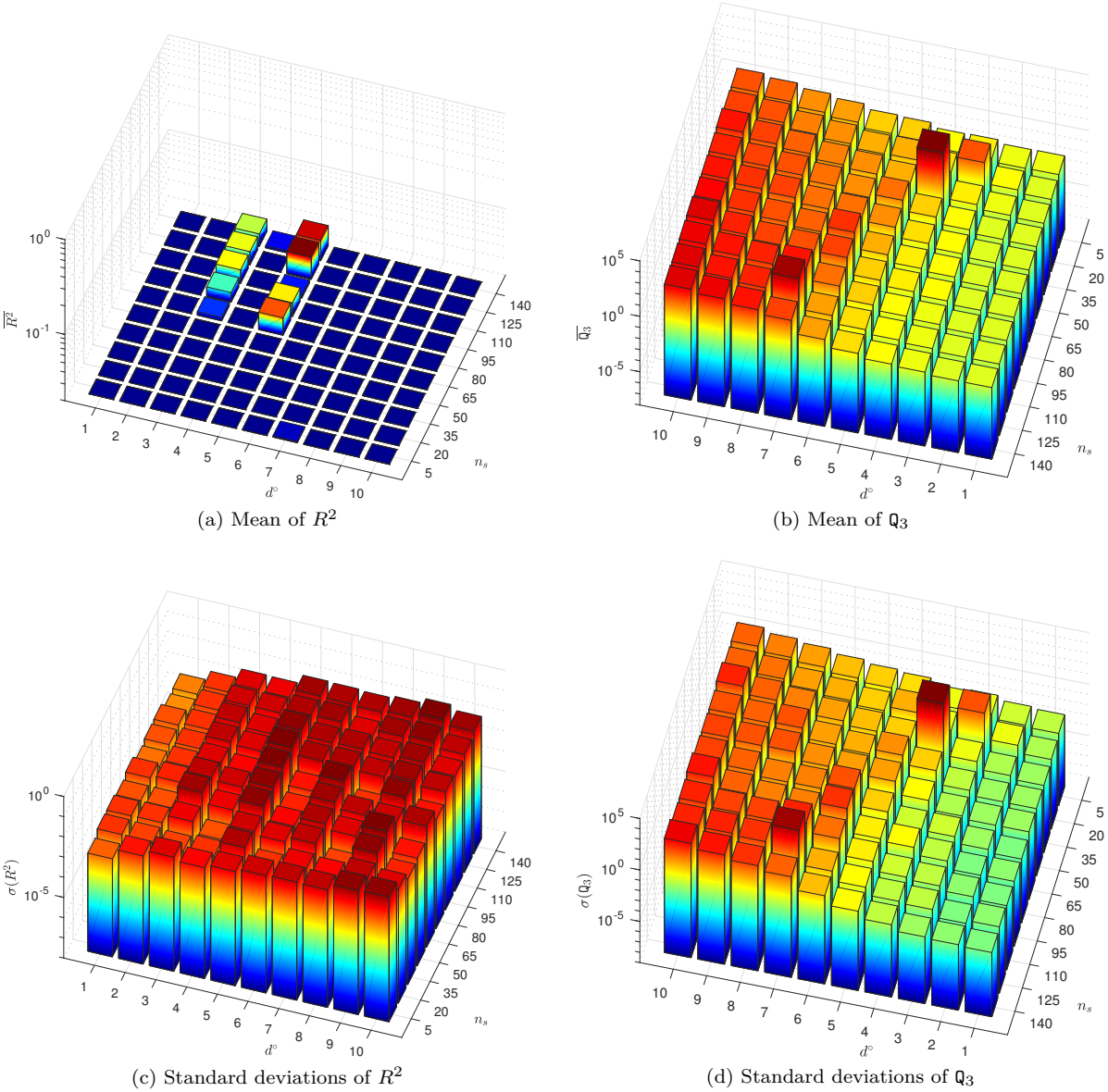
(b) Mean of $\mathtt{Q}_3$



(c) Standard deviations of R^2

(d) Standard deviations of $\mathtt{Q}_3$

Figure 18: Performance of the GradLS metamodel in approximating the 5 dimensional Rosenbrock's function in terms of the number of sample points (n_s) and the degree of the polynomial (d°). Compare with the response-only LS metamodel in Fig. 17.



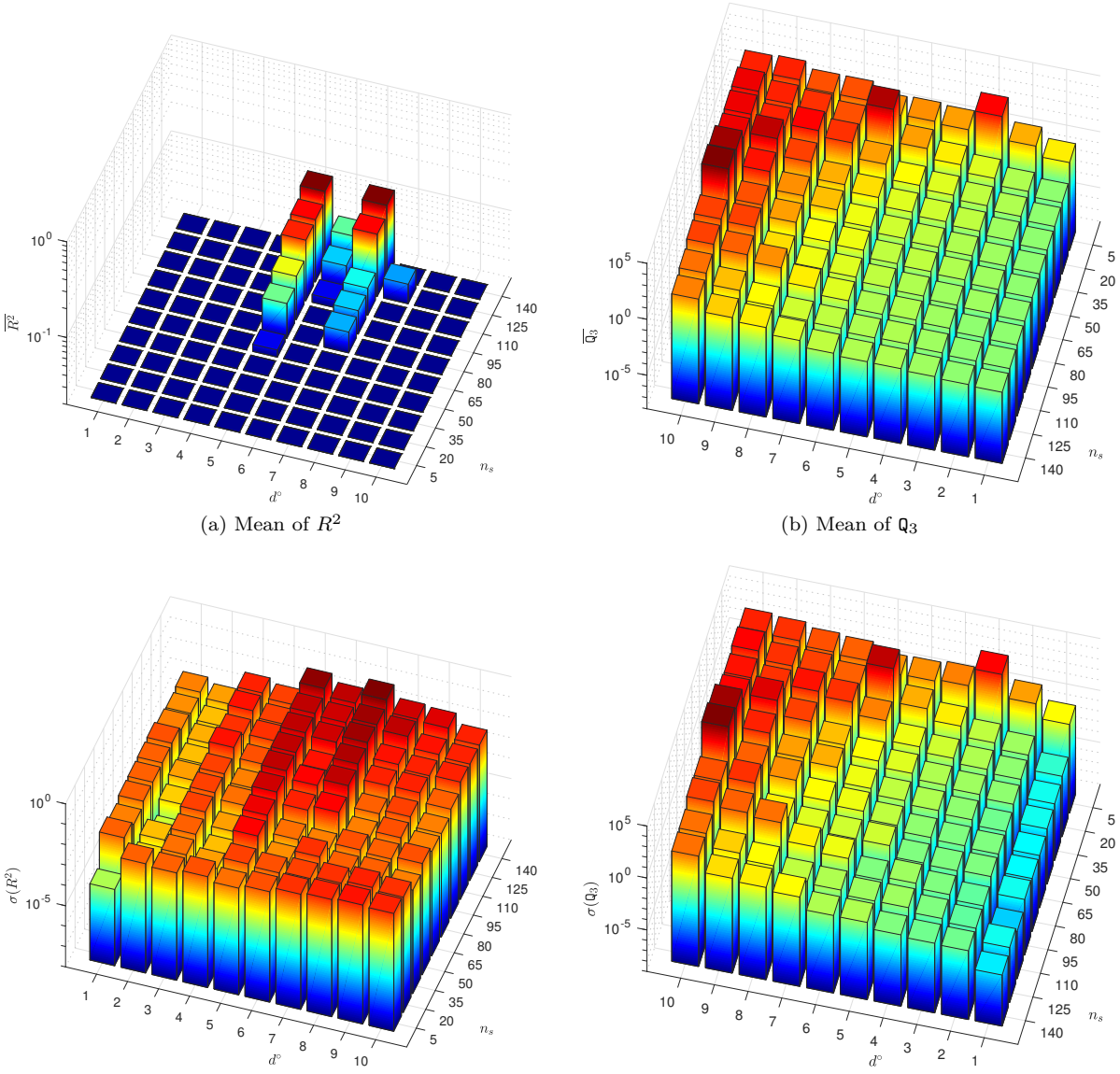
(a) Mean of R^2

(b) Mean of $\mathtt{Q}_3$

(c) Standard deviations of R^2

(d) Standard deviations of $\mathtt{Q}_3$

Figure 19: Performance of the LS metamodel in approximating the 3 dimensional Schwefel's function in terms of the number of sample points (n_s) and the degree of the polynomial (d°).



(a) Mean of R^2

(b) Mean of $\mathtt{Q}_3$

(c) Standard deviations of R^2

(d) Standard deviations of $\mathtt{Q}_3$

Figure 20: Performance of the GradLS metamodel in approximating the 3 dimensional Schwefel's function in terms of the number of sample points (n_s) and the degree of the polynomial (d°). Compare with the response-only LS metamodel in Fig. 19.

polynomial degree $d^\circ = 1$ up to 4, and then degrade as the degrees of the polynomials go to 10. The explanation is that Rosenbrock's function is a polynomial of degree 4. Below $d^\circ = 4$, the true function cannot be represented by the approximations. Beyond $d^\circ = 4$ it can, but higher order terms of the polynomial need to be cancelled, which requires sufficiently many sample points to be accurately done. An estimate of the limit on the number of sample points is n_s such that there are as many equations as polynomial terms n_t. This limit is seen in Fig. 14: for LS in dimension n_p =3, the lower bound on n_s is 35, 84 and 120 for $d^\circ = 4$, 6, 7; a ridge where $\mathtt{Q}_3$ suddenly degrades crosses the upper right plot of Fig. 15 and this ridge closely follows these limits on n_s. The same estimate (number of polynomial terms equal to number of equations) applied to GradLS yields n_s limits $n_p + 1$ smaller than those of LS: in Fig. 16, $n_p + 1 = 4$, and the $\mathtt{Q}_3$ degradation ridge follows ($d^\circ = 4$, $n_s = 35/4 \approx 9$), ($d^\circ = 6$, $n_s = 84/4 = 21$), ($d^\circ = 7$, $n_s = 120/4 = 30$). In dimension 5, the $\mathtt{Q}_3$ degradation ridges in Figs. 17 and 18 can be interpreted in the same way. In all of the figures, the less regular variation of the quality indicators with d° for n_s =5 is because it is too small a sample size.

Schwefel's function is not easily approximated with a polynomial. This is noticed in Figs. 19 and 20 which gather approximation performance indicators for the 3 dimensional version of the function and where the levels of R^2 and $\mathtt{Q}_3$ are respectively smaller and larger than those of the 3 dimensional Rosenbrock function. By comparing Fig. 19 to Fig. 20, it is also seen that the gradient-enhanced GradLS outperforms LS.

Two conclusions can already be drawn from these tests. Firstly, least squares approximations have a high performance domain characterized by a number of equation larger than the number of polynomial terms. This shows that the regularization performed by the pseudo-inverse does not produce as good least squares approximations as additional data points do. Secondly, because the gradient-enhanced least squares GradLS require sample sizes $n_p + 1$ smaller than those of classical LS, they have a much larger high performance domain.

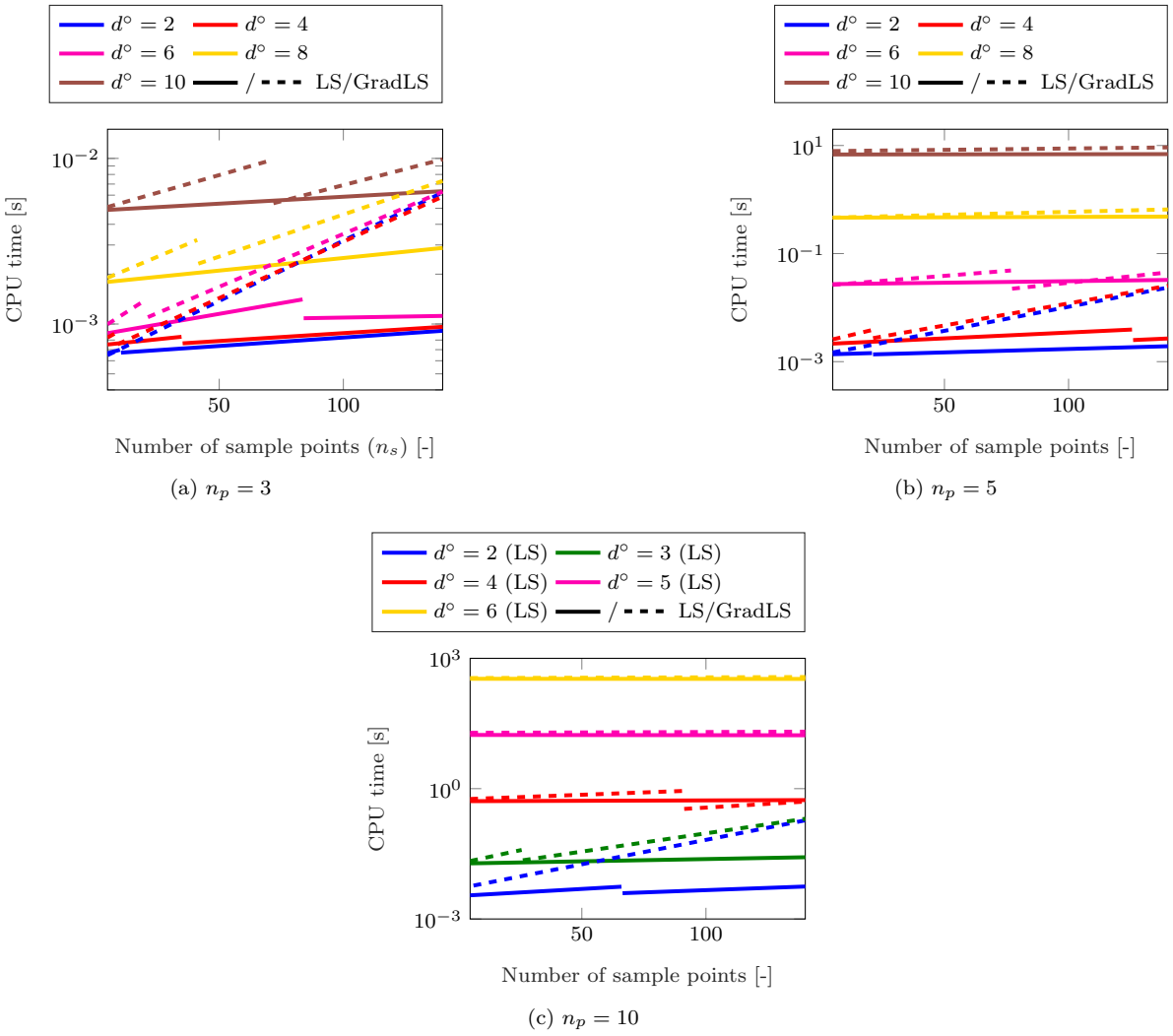
In addition, even outside of the high performance domain, it is observed in Figs. 15 to 20 that, at a given n_s and d°, GradLS consistently outperforms LS when approximating Rosenbrock's function.

Fig. 21 shows the CPU time needed to calculate the LS and GradLS metamodels as a function of the number of sample points n_s for degrees of the polynomial approximation d° ranging from 2 to 10 and in dimensions n_p =3, 5 and 10. In 10 dimension, the polynomials are limited to the degrees $\{2, 3, 4, 5, 6\}$ to keep calculation times reasonable. In both LS and GradLS models, it is observed that the CPU time grows with n_s, n_p and d° and that the main factors for CPU time increase are n_p and d° (a log scale is applied to the CPU axis). The effect of n_p and d° is comparable in both metamodels so that the CPU times are close to each other. The CPU time of the gradient-enhanced GradLS grows slightly faster than that of LS with n_s and in a manner independent of n_p and d°, which is mainly noticeable at low CPU times. It therefore turns out that enhancing least squares through the gradients is not as costly as one could fear. The reason is that, for high degree polynomials in high dimensions, the main numerical cost comes from the factorization of the $n_t \times n_t$ matrix $\mathbf{F}^\top\mathbf{F}$, which has $\mathcal{O}(n_t^3)$ complexity, and n_t is a function of n_p and d° but not of n_s (cf. Eq. (1)). The additional computation time of GradLS comes from the storage of the $(n_s(n_p + 1) \times n_t)$ matrix $\mathbf{F}$ and its product with other matrices.

Note also in Fig. 21 that the geometric evolution of the CPU time sometimes exhibits a discontinuity. For example, for LS, $n_p = 3$ and $d^\circ = 6$, there is a CPU time step at $n_s = 84$. These discontinuities correspond to the change in least squares algorithms when there are or not sufficient equations for determining the n_t polynomial terms: when the problem is well-posed, a QR factorization of $\mathbf{F}^\top\mathbf{F}$ is performed, when the problem is ill-posed, it is the pseudo-inverse method that is called. For high degree polynomials the CPU evolution curves are continuous because the problem is always ill-posed and the pseudo-inverse is the only active algorithm.

10.3 Comparison of kernel-based models

We now compare kernel-based metamodels that use or do not use gradients. These are variants of the RBF and KRG approaches. The SVR metamodels were not tested because of the large computational cost induced by tuning their internal parameters which does not allow repeated runs. The tested approaches are Ordinary Kriging (OK), Radial Basis Functions (RBF), both of which do not



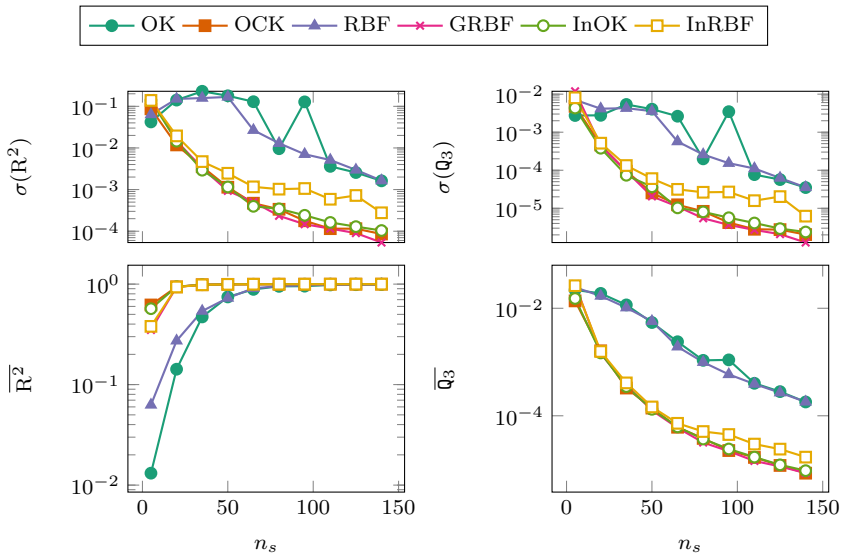
(a) $n_p = 3$

(b) $n_p = 5$

(c) $n_p = 10$

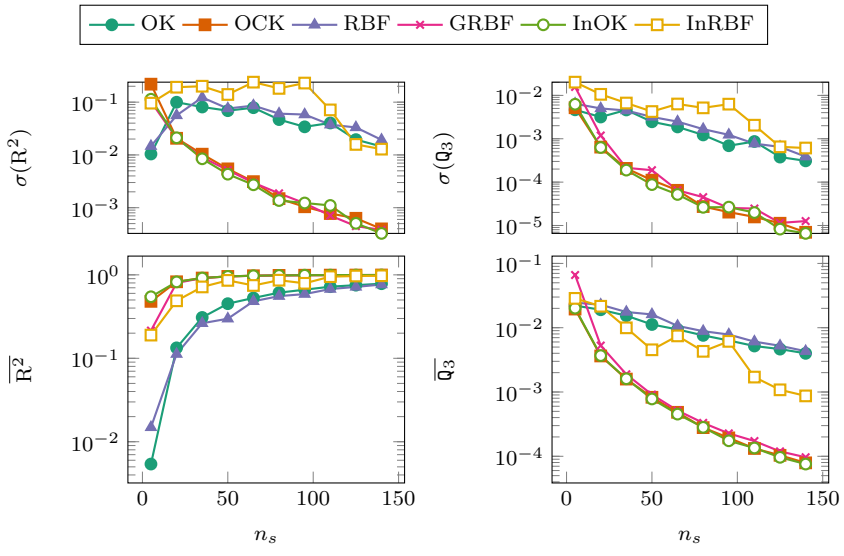
Figure 21: CPU time required for building the LS and GradLS metamodels as a function of the number of sample points n_s for dimensions $n_p = 3,\ 5$ and 10 and polynomial degrees d° ranging from 2 to 10.

utilize gradients, Indirect gradient-based Ordinary Kriging (InOK), Indirect gradient-based RBF (InRBF), gradient-enhanced Ordinary CoKriging (OCK), and Gradient-enhanced RBF (GRBF). All these examples take Matérn 3/2 as kernel function.



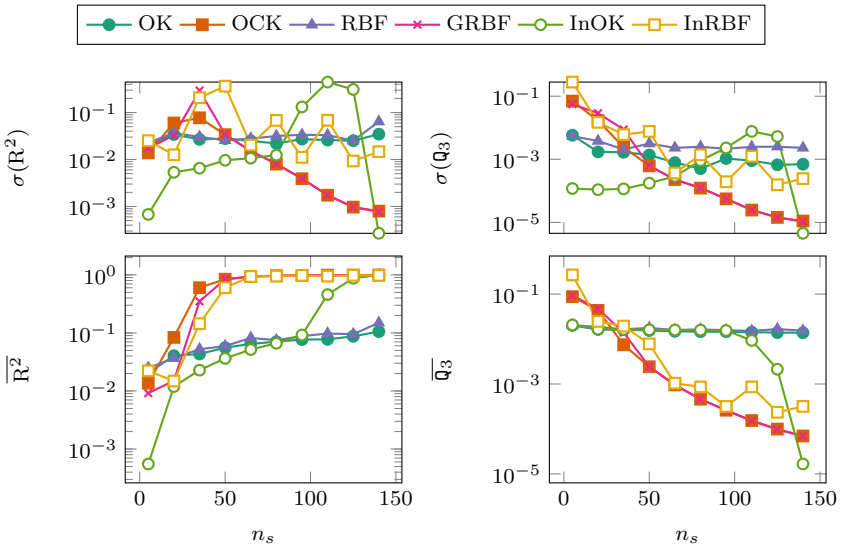
(a) R^2 mean values and standard deviations (b) Q_3 mean values and standard deviations

Figure 22: Performances of kernel-based metamodels in approximating the 5 dimensional Rosenbrock's function, obtained from 50 repetitions for each number of sample points n_s. The metamodels are: ordinary kriging (OK), ordinary gradient-enhanced cokriging (OCK), radial basis functions (RBF), gradient-enhanced radial basis functions (GRBF), indirect gradient-based ordinary kriging (OK) and indirect gradient-based RBF (InRBF.)



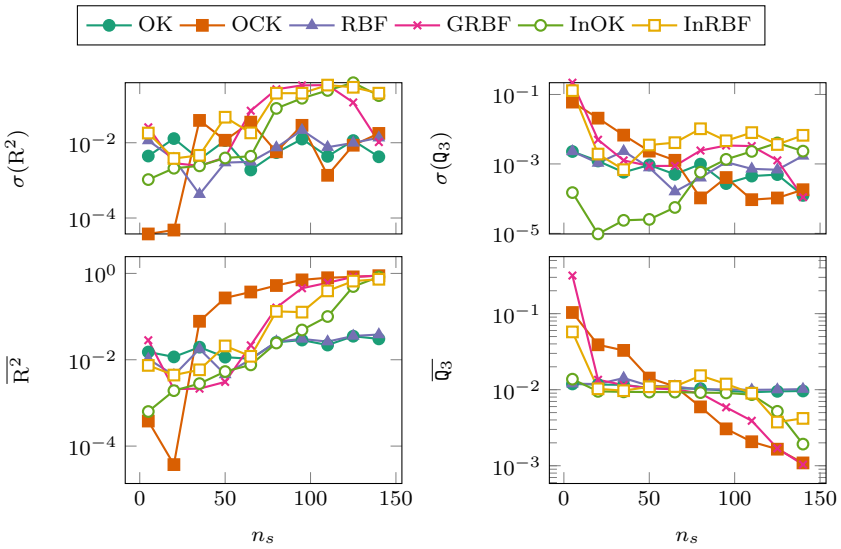
(a) R^2 mean values and standard deviations (b) Q_3 mean values and standard deviations

Figure 23: Performances of kernel-based metamodels in approximating the 10 dimensional Rosenbrock's function, obtained from 50 repetitions for each number of sample points n_s.



(a) R^2 mean values and standard deviations(b) Q_3 mean values and standard deviations

Figure 24: Performances of kernel-based metamodels in approximating the 5 dimensional Schwefel's function, obtained from 50 repetitions for each number of sample points n_s.



(a) R^2 mean values and standard deviations(b) Q_3 mean values and standard deviations

Figure 25: Performances of kernel-based metamodels in approximating the 10 dimensional Schwefel's function, obtained from 50 repetitions for each number of sample points n_s.

Figures 22 to 25 show the values of the R^2 and Q_3 criteria for the Rosenbrock and Schwefel test functions in 5 and 10 dimensions. It can be seen in Figures 22 and 23 that all the surrogates tested provide a good approximation to the Rosenbrock function in 5 and 10 dimensions, as measured by R^2 tending to 1 and Q_3 to 0 with n_s, which is likely because the function is smooth and unimodal. Nevertheless, the methods that directly or indirectly utilize gradients have a visible advantage, that is, OK and RBF converge much slower to good values of Q_3 and R^2 as n_s increases.

Schwefel's function is the approximation target in Figures 24 and 25. For modeling such a multimodal non stationary function, it is observed that directly accounting for gradients is a determining asset: The surrogates relying only on the response, OK and RBF, cannot approximate well the function, even when the number of sampled points n_s is large (equal to 140); surrogates directly using gradients, OCK and GRBF, manage to represent well Schwefel's function in both 5 and 10 dimensions. The performances of OCK and GRBF, are similar for the two test functions. The only noticeable difference is with Schwefel's function in 10 dimensions where OCK converges slightly faster than GRBF.

On the average of Figures 22 to Fig. 25, The indirect gradient-enhanced metamodels, InOK and InRBF, approximate the test functions with an accuracy that is between that of response-only and direct gradient-enhanced metamodels. A closer comparison of Fig. 22 and Fig. 24, and Fig. 23 and Fig. 25, suggests that InRBF performs better than InOK for the multimodal Schwefel function and vice versa for the smooth Rosenbrock's function. Once again, the main difference between InRBF and InOK is that InRBF tunes its internal parameters by cross-validation when InOK tunes them by maximum likelihood. Cross-validation shows a better ability to deal with multimodality than maximum likelihood does.

To sum up, these results illustrate the advantage of direct gradient-enhanced metamodels in approximating non stationary, multimodal functions. They confirm other experiments carried out in the more complete study [57].

Fig. 26 shows the CPU time taken for building the kernel-based metamodels for varying number of sample points and in dimensions 3, 5 and 10. SVR and GSVR are omitted in 10 dimensions because they take too much CPU time.

The building process includes the tuning of the metamodels' internal parameters which is performed here with an Particle Swarm Optimizer.

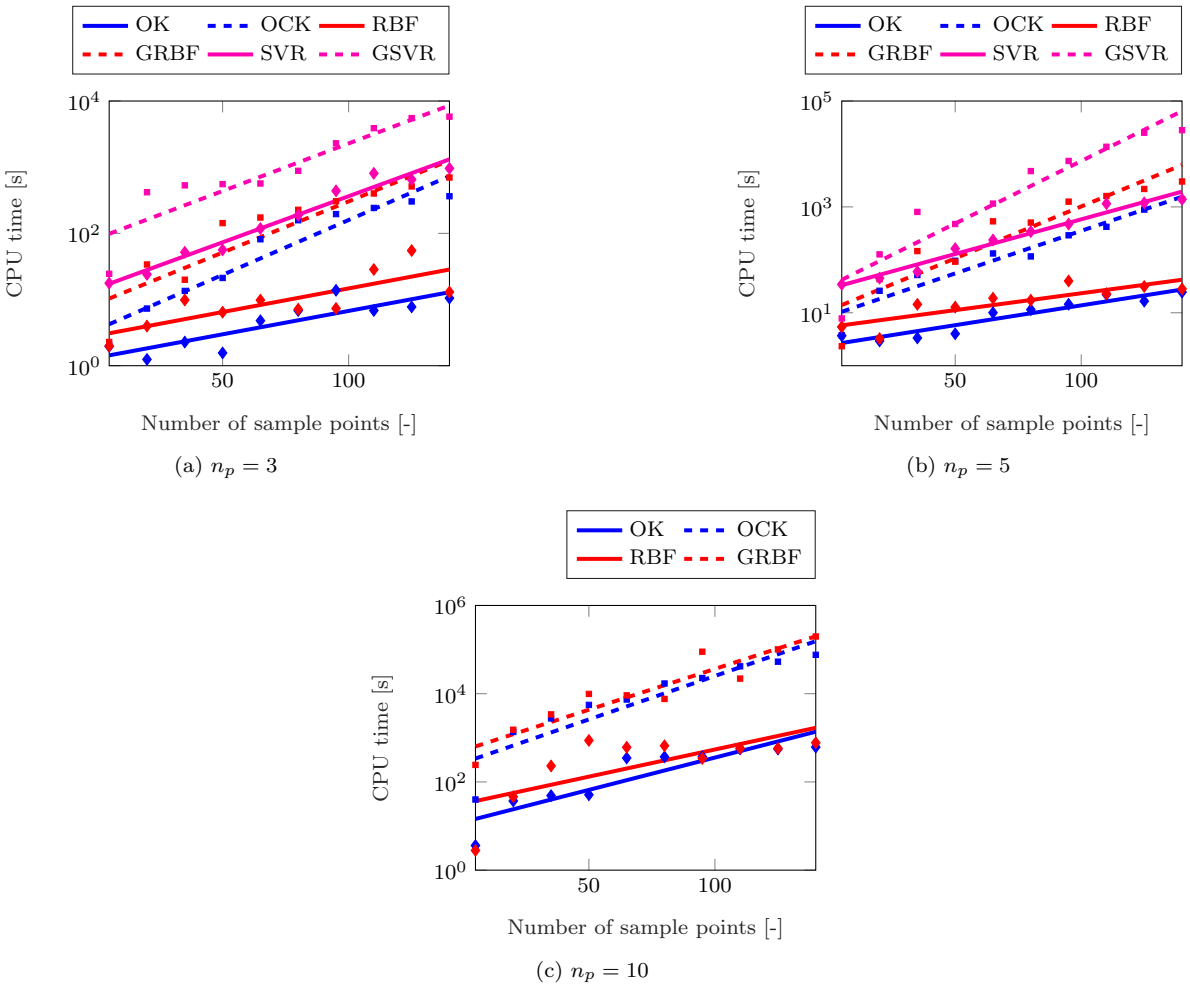
The typical CPU times of the kernel methods in Fig. 26 are larger than those of the least squares methods reported in Fig. 21. Furthermore, kernel methods show a higher sensitivity to the number of sample points n_s and a larger CPU penalty for including gradients in the model than least squares do. The main reason for the larger CPU time of kernel methods is the tuning of their internal parameters, which least squares do not do. The higher sensitivity of kernel methods to n_s and, consequently, to the presence of gradients, comes from the inversion of the $n_s \times n_s$ or $n_s(n_p+1) \times n_s(n_p+1)$ (in the gradient-enhanced version) matrices $\mathbf{\Psi}_g$ and $\mathbf{C}_c$. Similarly, SVR and GSVR have a number of constraints that scales with n_s and n_s (n_p+1), respectively. The advantage in terms of CPU time of the gradient-enhanced least squares methods should be assessed against an inferior approximation capacity as exemplified by the poor performance of GradLS on the Schwefel's function.

Among kernel methods, SVR and GSVR are the most time consuming techniques while KRG and GKRG (here OK/OCK, *i.e.* ordinary kriging/cokriging) are faster to calculate.

10.4 Available softwares for gradient-enhanced metamodels

Despite the wide use and availability of metamodels that exclusively use simulation responses, $y()$, the more recent gradient-enhanced metamodels are only proposed in a few codes.

GRENAT [12], which stands for ***GR****adient* ***EN****hanced* ***A****pproximation* ***T****oolbox*, is the toolbox that was used for generating all results and plots proposed in this review and in [59, 94, 95, 131, 132]. GRENAT is written in Matlab/Octave and follows the object oriented Matlab's syntax. It allows building and exploiting response-only, and indirect and direct gradient-enhanced kriging, radial basis functions and support vector regression. It can be linked to the MultiDOE toolbox [133] which compiles many sampling techniques.



(a) $n_p = 3$

(b) $n_p = 5$

(c) $n_p = 10$

Figure 26: CPU times taken for building kernel-based metamodels as a function of the number of sample points n_s in dimensions $n_p = 3$, 5 and 10.

The ooDACE Toolbox [134, 135], also developed in object-oriented Matlab, has an implementation of cokriging that could accomodate accounting for gradients.

In addition to mathematical descriptions, Forrester et al. [21] also propose code samples for building gradient-enhanced metamodels.

10.5 Remarks about missing data and higher order derivatives

In keeping with previous works [21, 136], the RBF, kriging and SVR metamodels and their gradient-enhanced versions that have been described in the review can readily be adapted for dealing with missing data: Hybrid version of these metamodels can be considered by removing responses, components of gradient or full gradient at certain sample points. Components of the vector $\mathbf{y}_g$ are deleted and the corresponding terms in the linear combinations making the approximations (in generalized least squares, GRBF, GKRG) are removed from the equations. In the case of GSVR, the constraints in the model defining optimization problem for which there is no longer an observation are removed. With IDW, terms of the first order Taylor approximations $Q_j(\mathbf{x})$ can be dropped, at the cost of loosing the corresponding gradient interpolation properties.

Deleting observations can even be a choice for minimizing the computational cost needed to build the metamodels and evaluate the responses and/or gradients. For example, when dealing with a function with known flat behavior in a part of the design space and a multimodal behavior in another part, accounting for gradient information is useful only in the latter. On the basis of relations like Eq. (26) for least squares, or Eq. (62) for radial basis functions, or Eq. (89) for cokriging, which all involve the inversion of a $n_s(n_p+1)$ by $n_s(n_p+1)$ covariance matrix, the computational complexity of each observation is at least cubic: assuming the number of operations required to invert a square matrix is slightly less than cubic, it will be at least cubic when multiplied by the number of repetitions of the inversion required for tuning internal parameters; then, accounting for all the gradients multiplies the complexity of the metamodels by at least $(n_p+1)^3$. This metamodel complexity, although non negligible, will typically remain orders of magnitude smaller than the complexity of calculating the true response.

The formulations of gradient-enhanced RBF, cokriging and SVR can be also extended for taking into account higher order derivatives of the objectif function. Examples of formulations of Hessian-enhanced cokriging can be found in [21, 53]. In the case of SVR, an Hessian formulation may be based on the development proposed in [72]: the "prior knowledge", which is added to the classical SVR formulation (Problem 9.1 in Section 9), consists in terms of the Hessian which are accounted for through new constraints.

11 Conclusions

We have reviewed the main surrogates (or metamodels) for approximating functions observed at a finite number of points that not only use function values but also their gradients. These surrogates are variations around the least squares methods, the Shepard weighting function, radial basis functions, kriging and support vector machines. Indirect methods, where the knowledge of the gradients produces new points to learn from, have also been covered. An effort was made to detail the logic and the formulations that led to these models. To the authors' knowledge, the ν-SVR formulation with gradients was given here for the first time. Another goal was to compare the metamodels. It was first done theoretically, in particular by casting all metamodels as linear combinations of functions chosen a priori and coefficients that depend on the observations. The comparison between metamodels was then substantiated by simple examples.

These examples, confirming other studies [9, 57, 132], show that exploiting gradient information is a determining advantage for approximating functions with locally changing behaviors. Including gradients in least squares methods comes at a negligible additional numerical cost. The more versatile kernel-based surrogates pay a numerical cost for also approximating gradients: all methods but Shepard weighting function have a complexity that scales at least with the cube of the number of observations, and each gradient at a point in a space of dimension n_p adds n_p observations.

The litterature on gradient-enhanced metamodels is recent but already rich. Today, many perspectives should be considered.

From a methodological point of view, there is a need for more robust, numerically less complex approaches that can account for large numbers (say, millions) of data points with their gradients, in higher dimensions (say, thousands). The current kernel methods can approximate a larger family of functions than least squares do, but they would not allow data sets beyond of the order of 1000 points because of bad conditionning issues and because of the rapidly growing number of operations. Beyond 10000 points, computer memory would be an additional limitation. Recent works on Gaussian Processes have introduced strategies to deal with large number of points [137, 138] and high-dimensional problems [139, 140]. However these approaches remain currently limited to response-only data.

On the applicative side, surrogates that learn and predict gradients should contribute to progress in local and global sensitivity analysis, uncertainty propagation, local trust region and global surrogate-based optimization methods.

Acronyms

ν-GSVR ν-version of the gradient-enhanced Support Vector Regression (surrogate model). 1, 45
ε-SVR ε-version of the Support Vector Regression (surrogate model). 45
ν-SVR ν-version of the Support Vector Regression (surrogate model). 45, 61
ε_k-GSVR ε_k-version of the gradient-enhanced Support Vector Regression (surrogate model). 44, 45

BLUP Best Linear Unbiased Predictor. 30

EGO Efficient Global Optimization [33]. 5

GBK Gradient-based kriging (surrogate model). 28, 63
GEK Gradient-enhanced kriging (surrogate model). 28, 63
GEUK Gradient-enhanced universal kriging (surrogate model). 28, 63
GKRG Gradient-enhanced cokriging (surrogate model, same as GBK, GEK and GEUK). 2, 6, 7, 8, 18, 27, 35, 36, 43, 45, 59, 61
GLS Generalized Least Square regression (surrogate model). 2, 9, 11, 14, 15
GradLS gradient-enhanced Least Square regression (surrogate model). 2, 9, 11, 12, 13, 14, 15, 47, 48, 50, 52, 54, 55, 56, 59
GRBF Gradient-enhanced Radial Basis Function (surrogate model). 2, 6, 7, 8, 9, 15, 20, 22, 23, 27, 33, 35, 38, 43, 45, 57, 59, 61
GRENAT GRadient-ENhanced Approximation Toolbox (Matlab/Octave's toolbox, [12]). 45, 59
GSVR Gradient-enhanced Support Vector Regression (surrogate model). 2, 6, 7, 8, 18, 39, 40, 42, 43, 44, 45, 59, 61

IDW Inverse Distance Weighting method also called Shepard Weighting method (surrogate model). 2, 6, 7, 8, 17, 18, 61
IHS Improved Hypercube Sampling (Sampling technique, [103]). 23, 25, 26
InOK Indirect gradient-enhanced ordinary kriging (surrogate model). 9, 36, 57, 59
InRBF Indirect gradient-enhanced Radial Basis Function (surrogate model). 9, 57, 59

KRG Kriging (surrogate model). 9, 36, 55, 59

LOO Leave-One-Out. 24, 27, 38
LS Least Square regression (surrogate model). 2, 6, 7, 8, 9, 11, 12, 13, 15, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56

MLS Moving Least Square regression (surrogate model). 2, 6, 8, 15, 16, 17
MSE Mean Square Error (quality criterion). 11
MultiDOE Multiple Design Of Experiments (Matlab/Octave's toolbox, [133]). 59

OCK Gradient-enhanced Ordinary cokriging (surrogate model). 9, 36, 57, 59
OK Ordinary kriging (surrogate model). 55, 57, 59

RBF Radial Basis Function (surrogate model). 2, 9, 22, 23, 24, 27, 28, 35, 47, 55, 57, 59, 61
RSM Response Surface Methodology. 4, 9, 15

SBAO Surrogate-Based Analysis and Optimization [18]. 4, 5
SVM Support Vector Machine. 39
SVR Support Vector Regression (surrogate model). 39, 40, 42, 44, 45, 47, 55, 59, 61

WLS Weigthed Least Square regression (surrogate model). 6, 8, 11, 14

References

[1] G. F. Hughes. "On the mean accuracy of statistical pattern recognizers". In: *IEEE Transactions on Information Theory* 14.1 (1968), pp. 55–63. DOI: 10.1109/TIT.1968.1054102 (cit. on p. 3).

[2] J.-L. Lions. *Optimal Control of Systems Governed by Partial Differential Equations.* Springer Verlag, New York, 1971 (cit. on p. 3).

[3] D. G. Cacuci. "Sensitivity theory for nonlinear systems. I. Nonlinear functional analysis approach". In: *Journal of Mathematical Physics* 22.12 (1981), pp. 2794–2802. DOI: 10.1063/1.525186 (cit. on p. 3).

[4] A. Jameson. "Aerodynamic design via control theory". English. In: *Journal of Scientific Computing* 3.3 (1988), pp. 233–260. DOI: 10.1007/BF01061285 (cit. on p. 3).

[5] L. Beda et al. *Programs for automatic differentiation for the machine.* Tech. rep. BESM, Inst. Precise Mechanics and Computation Techniques, Academy of Science, Moscow, 1959 (cit. on p. 3).

[6] R. E. Wengert. "A simple automatic derivative evaluation program". In: *Commun. ACM* 7.8 (Aug. 1964), pp. 463–464. DOI: 10.1145/355586.364791 (cit. on p. 3).

[7] A. Griewank. "On automatic differentiation". In: *Mathematical Programming: recent developments and applications.* Amsterdam, 1989, pp. 83–108 (cit. on p. 3).

[8] V. Paoletti et al. "Inversion of gravity gradient tensor data: does it provide better resolution?" In: *Geophysical Journal International* 205.1 (2016), pp. 192–202. DOI: 10.1093/gji/ggw003 (cit. on p. 3).

[9] P. Qin et al. "Integrated gravity and gravity gradient 3D inversion using the non-linear conjugate gradient". In: *Journal of Applied Geophysics* 126 (2016), pp. 52–73. DOI: http://dx.doi.org/10.1016/j.jappgeo.2016.01.013 (cit. on pp. 3, 61).

[10] R. Lorentz. "Multivariate Hermite interpolation by algebraic polynomials: A survey". In: *Journal of Computational and Applied Mathematics* 122.1–2 (2000). Numerical Analysis in the 20th Century Vol. II: Interpolation and Extrapolation, pp. 167–201. DOI: http://dx.doi.org/10.1016/S0377-0427(00)00367-8 (cit. on p. 3).

[11] M.-J. Lai. "Multivariate splines for data fitting and approximation". In: *Approximation Theory XII: San Antonio* (2007), pp. 210–228 (cit. on p. 3).

[12] L. Laurent. *GRENAT (Matlab/Octave Toolbox) https://bitbucket.org/luclaurent/grenat.* 2016 (cit. on pp. 3, 45, 59, 63).

[13] G. E. P. Box and K. Wilson. "On the Experimental Attainment of Optimum Conditions". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 13.1 (1951), pp. 1–45 (cit. on pp. 4, 9).

[14] T. W. Simpson et al. "Comparison Of Response Surface And Kriging Models For Multidisciplinary Design Optimization". In: *AIAA paper 98-4758. 7 th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.* 1998, pp. 98–4755. DOI: 10.2514/6.1998-4755 (cit. on p. 4).

[15] A. A. Giunta and L. T. Watson. "A comparison of approximation modeling techniques - Polynomial versus interpolating models". In: *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.* AIAA-98-4758. American Institute of Aeronautics and Astronautics, 1998. DOI: doi:10.2514/6.1998-4758 (cit. on p. 4).

[16] R. Jin, W. Chen, and T. Simpson. "Comparative studies of metamodeling techniques under multiple modeling criteria". In: *8th Symposium on Multidisciplinary Analysis and Optimization.* AIAA-2000-4801. American Institute of Aeronautics and Astronautics, 2000. DOI: doi:10.2514/6.2000-4801 (cit. on p. 4).

[17] S. Varadarajan, W. Chen, and C. J. Pelka. "Robust concept exploration of propulsion systems with enhanced model approximation capabilities". In: *Engineering Optimization* 32.3 (2000), pp. 309–334. DOI: doi:10.1080/03052150008941302 (cit. on p. 4).

[18] N. V. Queipo et al. "Surrogate-based analysis and optimization". In: *Progress in Aerospace Sciences* 41.1 (2005), pp. 1–28. DOI: DOI:10.1016/j.paerosci.2005.02.001 (cit. on pp. 4, 63).

[19] J. Peter et al. "Comparison of surrogate models for the actual global optimization of a 2D turbomachinery flow". In: *Proceedings of the 7th WSEAS International Conference on Simulation, Modelling and Optimization*. World Scientific, Engineering Academy, and Society (WSEAS). 2007, pp. 46–51 (cit. on p. 4).

[20] M. Marcelet. "Etude et mise en oeuvre d'une méthode d'optimisation de forme couplant simulation numérique en aérodynamique et en calcul de structure". PhD thesis. École Nationale Supérieure d'Arts et Métiers, 2008 (cit. on p. 4).

[21] A. I. J. Forrester, A. Sóbester, and A. J. Keane. *Engineering Design via Surrogate Modelling: A Practical Guide*. 1st ed. Wiley, Sept. 2008 (cit. on pp. 4–5, 28, 61).

[22] B.-S. Kim, Y.-B. Lee, and D.-H. Choi. "Comparison study on the accuracy of metamodeling technique for non-convex functions". In: *Journal of Mechanical Science and Technology* 23.4 (Apr. 2009), pp. 1175–1181. DOI: 10.1007/s12206-008-1201-3 (cit. on p. 4).

[23] D. Zhao and D. Xue. "A comparative study of metamodeling methods considering sample quality merits". English. In: *Structural and Multidisciplinary Optimization* 42.6 (2010), pp. 923–938. DOI: 10.1007/s00158-010-0529-3 (cit. on p. 4).

[24] M. D. McKay, R. J. Beckman, and W. J. Conover. "Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code". In: *Technometrics* 21.2 (1979), pp. 239–245. DOI: 10.1080/00401706.1979.10489755 (cit. on p. 4).

[25] T. J. Santner, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments*. Springer-Verlag, 2003, p. 283 (cit. on p. 4).

[26] K.-T. Fang, R. Li, and A. Sudjianto. *Design and modeling for computer experiments*. Chapman & Hall/CRC, 2005 (cit. on p. 4).

[27] J. Franco. "Planification d'expériences numériques en phase exploratoire pour la simulation des phénomènes complexes". PhD thesis. École Nationale Supérieure des Mines de Saint-étienne, 2008 (cit. on p. 4).

[28] M. Schonlau. "Computer Experiments and Global Optimization". PhD thesis. University of Waterloo, 1997 (cit. on pp. 4–5).

[29] A. Sóbester, S. J. Leary, and A. J. Keane. "On the Design of Optimization Strategies Based on Global Response Surface Approximation Models". English. In: *Journal of Global Optimization* 33.1 (2005), pp. 31–59. DOI: 10.1007/s10898-004-6733-1 (cit. on pp. 4–5, 27).

[30] N. M. Alexandrov et al. "A trust-region framework for managing the use of approximation models in optimization". In: *Structural optimization* 15.1 (1998), pp. 16–23. DOI: 10.1007/BF01197433 (cit. on p. 4).

[31] M. J. Sasena. "Flexibility and efficiency enhancements for constrained global design optimization with kriging approximations". PhD thesis. University of Michigan, 2002 (cit. on p. 5).

[32] A. G. Watson and R. J. Barnes. "Infill sampling criteria to locate extremes". In: *Mathematical Geology* 27.5 (1995), pp. 589–608. DOI: 10.1007/BF02093902 (cit. on p. 5).

[33] D. R. Jones, M. Schonlau, and W. J. Welch. "Efficient global optimization of expensive black-box functions". In: *Journal of Global optimization* 13.4 (1998), pp. 455–492 (cit. on pp. 5, 27, 63).

[34] G. Matheron. "La théorie des variables régionalisées et ses applications". In: *Les Cahiers du Centre de Morphologie Mathématique de Fontainebleau, école Nationale des Mines de Paris* Fascicule 5 (1970) (cit. on pp. 6, 27).

[35] H. Wackernagel. *Multivariate geostatistics: an introduction with applications*. Springer Verlag, 1995 (cit. on pp. 6, 27, 29–30, 34).

[36] E. H. Isaaks and R. M. Srivastava. *An introduction to applied geostatistics*. Oxford University Press, New York, 1989 (cit. on pp. 6, 27).

[37] J. M. Hoef and N. A. C. Cressie. "Multivariable spatial prediction". English. In: *Mathematical Geology* 25.2 (1993), pp. 219–240. DOI: 10.1007/BF00893273 (cit. on pp. 6, 27).

[38] P. Goovaerts. *Geostatistics for natural resources evaluation*. Oxford University Press, USA, 1997 (cit. on pp. 6, 27).

[39] M. D. Morris, T. J. Mitchell, and D. Ylvisaker. "Bayesian design and analysis of computer experiments: use of derivatives in surface prediction". In: *Technometrics* 35.3 (1993), pp. 243–255 (cit. on pp. 6, 27–29, 34).

[40] J. R. Koehler and A. B. Owen. "Computer experiments". In: *Handbook of statistics* 13 (1996), pp. 261–308 (cit. on pp. 6, 28).

[41] R. M. Lewis. "Using sensitivity information in the construction of kriging models for design optimization". In: *Proceedings of the 7th AIAA/USAF/NASA/ISSMO Multidisciplinary Analysis & Optimization Symposium, Saint Louis, Missouri*. 1998. DOI: doi:10.2514/6.1998-4799 (cit. on pp. 6, 28).

[42] M. Arnaud and X. Emery. *Estimation et interpolation spatiale*. Hermes Science Publications, Paris, 2000 (cit. on pp. 6, 27).

[43] H.-S. Chung and J. J. Alonso. "Using gradients to construct cokriging approximation models for high-dimensional design optimization problems". In: *40th AIAA Aerospace Sciences Meeting & Exhibit*. AIAA-2002-0317. American Institute of Aeronautics and Astronautics, 2002. DOI: doi:10.2514/6.2002-317 (cit. on pp. 6, 9, 17, 28).

[44] H.-S. Chung and J. J. Alonso. "Design of a Low-Boom Supersonic Business Jet Using Cokriging Approximation Models". In: *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*. AIAA-2002-5598. American Institute of Aeronautics and Astronautics, 2002. DOI: doi:10.2514/6.2002-5598 (cit. on pp. 6, 28).

[45] S. J. Leary, A. Bhaskar, and A. J. Keane. "A derivative based surrogate model for approximating and optimizing the output of an expensive computer simulation". In: *Journal of Global Optimization* 30.1 (2004), pp. 39–58 (cit. on pp. 6, 28).

[46] S. J. Leary, A. Bhaskar, and A. J. Keane. "Global approximation and optimization using adjoint computational fluid dynamics codes". In: *AIAA journal* 42.3 (2004), pp. 631–641 (cit. on pp. 6, 17, 20, 28).

[47] J. Laurenceau and P. Sagaut. "Building Efficient Response Surfaces of Aerodynamic Functions with Kriging and Cokriging". In: *AIAA Journal* 46.2 (2008), pp. 498–507. DOI: 10.2514/1.32308 (cit. on pp. 6, 28).

[48] J. Laurenceau and M. Meaux. "Comparison of Gradient and Response Surface Based Optimization Frameworks Using Adjoint Method". In: *49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 16th AIAA/ASME/AHS Adaptive Structures Conference,10th AIAA Non-Deterministic Approaches Conference, 9th AIAA Gossamer Spacecraft Forum, 4th AIAA Multidisciplinary Design Optimization Specialists Conference*. AIAA-2008-1889. American Institute of Aeronautics and Astronautics, 2008. DOI: doi:10.2514/6.2008-1889 (cit. on pp. 6, 9, 28).

[49] R. Dwight and Z.-H. Han. "Efficient Uncertainty Quantification Using Gradient-Enhanced Kriging". In: *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. AIAA-2009-2276. American Institute of Aeronautics and Astronautics, 2009. DOI: doi:10.2514/6.2009-2276 (cit. on pp. 6, 28).

[50] Y. Xuan et al. "Gradient-based Kriging approximate model and its application research to optimization design". In: *Science in China Series E: Technological Sciences* 52.4 (2009), pp. 1117–1124. DOI: 10.1007/s11431-009-0096-2 (cit. on pp. 6, 28).

[51] B. A. Lockwood and D. J. Mavriplis. "Parameter Sensitivity Analysis for Hypersonic Viscous Flow using a Discrete Adjoint Approach". In: *AIAA Paper* 447 (2010) (cit. on pp. 6, 28).

[52] A. March, K. Willcox, and Q. Wang. "Gradient-based multifidelity optimisation for aircraft design using Bayesian model calibration". In: *Aeronautical Journal* 115.1174 (2010), pp. 729–738. DOI: 10.1017/S0001924000006473 (cit. on pp. 6, 28).

[53] W. Yamazaki, M. Rumpfkeil, and D. Mavriplis. "Design Optimization Utilizing Gradient/Hessian Enhanced Surrogate Model". In: *28th AIAA Applied Aerodynamics Conference*. AIAA-2010-4363. American Institute of Aeronautics and Astronautics, 2010. DOI: doi:10.2514/6.2010-4363 (cit. on pp. 6, 28, 61).

[54] M. Bompard. "Modéles de substitution pour l'optimisation globale de forme en aérodynamique et méthode locale sans paramétrisation". PhD thesis. Université de Nice Sophia Antipolis, 2011 (cit. on pp. 6, 27–28, 38–39, 45, 47).

[55] M. P. Rumpfkeil, W. Yamazaki, and D. J. Mavriplis. "A Dynamic Sampling Method for Kriging and Cokriging Surrogate Models". In: *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. AIAA-2011-883. American Institute of Aeronautics and Astronautics, 2011. DOI: doi:10.2514/6.2011-883 (cit. on pp. 6, 28).

[56] L. Laurent, P.-A. Boucard, and B. Soulier. "Gradient-Enhanced Metamodels and Multiparametric Strategies for Designing Structural Assemblies". In: *B.H.V. Topping, (Editor), Proceedings of the Eleventh International Conference on Computational Structures Technology, 4-7 September*. Paper 230. Civil-Comp Press, Stirlingshire, UK, 2012. DOI: 10.4203/ccp.99.230 (cit. on pp. 6, 20, 28).

[57] L. Laurent. "Stratégie multiparamétrique et métamodèles pour l'optimisation multiniveaux de structures". PhD thesis. 61, avenue du Président Wilson, 94230 Cachan: École Normale Supérieure de Cachan, 2013 (cit. on pp. 6, 20, 28, 59, 61).

[58] L. Laurent, P.-A. Boucard, and B. Soulier. "Combining Multiparametric Strategy and Gradient-Based Surrogate Model for Optimizing Structure Assemblies". In: *10th World Congress on Structural and Multidisciplinary Optimization, Orlando, Florida, USA, 19-24 May*. Ed. by ISSMO. 2013 (cit. on pp. 6, 20, 28).

[59] L. Laurent. "Global optimisation on assembly problems using gradient-based surrogate model and multiparametric strategy". In: *PhD Olympiad ECCOMAS, 11th World Congress on Computational Mechanics, Barcelona, Spain, July 20-25*. 2014 (cit. on pp. 6, 20, 28, 59).

[60] Z.-H. Han, S. Görtz, and R. Zimmermann. "Improving variable-fidelity surrogate modeling via gradient-enhanced kriging and a generalized hybrid bridge function". In: *Aerospace Science and Technology* 25.1 (2013), pp. 177–189. DOI: http://dx.doi.org/10.1016/j.ast.2012.01.006 (cit. on pp. 6, 28).

[61] R. Zimmermann. "On the Maximum Likelihood Training of Gradient-Enhanced Spatial Gaussian Processes". In: *SIAM Journal on Scientific Computing* 35.6 (2013), A2554–A2574. DOI: 10.1137/13092229X (cit. on pp. 6, 28).

[62] S. Ulaganathan et al. "Performance study of multi-fidelity gradient enhanced kriging". English. In: *Structural and Multidisciplinary Optimization* 51.5 (2015), pp. 1017–1033. DOI: 10.1007/s00158-014-1192-x (cit. on pp. 6, 18, 28).

[63] S. Ulaganathan et al. "Performance study of gradient-enhanced Kriging". In: *Engineering with Computers* 32.1 (2016), pp. 15–34. DOI: 10.1007/s00366-015-0397-y (cit. on pp. 6, 18–19, 28, 35, 39).

[64] W. Zongmin. "Hermite-Birkhoff interpolation of scattered data by radial basis functions". English. In: *Approximation Theory and its Applications* 8.2 (1992), pp. 1–10. DOI: 10.1007/BF02836101 (cit. on pp. 6, 20).

[65] I. C. Kampolis, E. I. Karangelos, and K. C. Giannakoglou. "Gradient-assisted radial basis function networks: theory and applications". In: *Applied Mathematical Modelling* 28.2 (2004), pp. 197–209. DOI: http://dx.doi.org/10.1016/j.apm.2003.08.002 (cit. on pp. 6, 20).

[66] K. C. Giannakoglou, D. I. Papadimitriou, and I. C. Kampolis. "Aerodynamic shape design using evolutionary algorithms and new gradient-assisted metamodels". In: *Computer Methods in Applied Mechanics and Engineering* 195.44–47 (2006), pp. 6312–6329. DOI: http://dx.doi.org/10.1016/j.cma.2005.12.008 (cit. on pp. 6, 20).

[67] Y.-S. Ong, K. Lum, and P. B. Nair. "Hybrid evolutionary algorithm with Hermite radial basis function interpolants for computationally expensive adjoint solvers". English. In: *Computational Optimization and Applications* 39.1 (2008), pp. 97–119. DOI: 10.1007/s10589-007-9065-5 (cit. on pp. 6, 20).

[68] M. Lázaro et al. "Support Vector Regression for the simultaneous learning of a multivariate function and its derivatives". In: *Neurocomputing* 69.1–3 (2005), pp. 42–61. DOI: http://dx.doi.org/10.1016/j.neucom.2005.02.013 (cit. on pp. 6, 39, 42).

[69] Jayadeva, R. Khemchandani, and S. Chandra. "Regularized Least Squares Twin SVR for the Simultaneous Learning of a Function and its Derivative". In: *Neural Networks, 2006. IJCNN'06. International Joint Conference onIJCNN '06. International Joint Conference on.* 2006, pp. 1192–1197. DOI: 10.1109/IJCNN.2006.246826 (cit. on pp. 6, 39).

[70] G. Bloch et al. "Support vector regression from simulation data and few experimental samples". In: *Information Sciences* 178.20 (2008). Special Issue on Industrial Applications of Neural Networks - 10th Engineering Applications of Neural Networks 2007, pp. 3813–3827. DOI: http://dx.doi.org/10.1016/j.ins.2008.05.016 (cit. on pp. 6, 39).

[71] Jayadeva, R. Khemchandani, and S. Chandra. "Regularized least squares support vector regression for the simultaneous learning of a function and its derivatives". In: *Information Sciences* 178.17 (2008), pp. 3402–3414. DOI: http://dx.doi.org/10.1016/j.ins.2008.04.007 (cit. on pp. 6, 39, 42).

[72] F. Lauer and G. Bloch. "Incorporating prior knowledge in support vector regression". English. In: *Machine Learning* 70.1 (2008), pp. 89–118. DOI: 10.1007/s10994-007-5035-5 (cit. on pp. 6, 39, 61).

[73] R. Khemchandani, A. Karpatne, and S. Chandra. "Twin support vector regression for the simultaneous learning of a function and its derivatives". English. In: *International Journal of Machine Learning and Cybernetics* 4.1 (2013), pp. 51–63. DOI: 10.1007/s13042-012-0072-1 (cit. on pp. 6, 39, 42).

[74] R. J. Renka. "Multivariate interpolation of large sets of scattered data". In: *ACM Transactions on Mathematical Software (TOMS)* 14.2 (1988), pp. 139–148 (cit. on pp. 6, 17).

[75] S. Lauridsen et al. "Response surface approximation using gradient information". In: *Proceedings of 4th world congress on structural and multidisciplinary optimization, 4-8 June, 2001, Dalian China.* Ed. by Cheng, G. et al. (Red.) Vol. 4. 2002, p. 5 (cit. on pp. 6, 9, 11).

[76] C. Kim, S. Wang, and K. K. Choi. "Efficient Response Surface Modeling by Using Moving Least-Squares Method and Sensitivity". In: *AIAA Journal* 43.11 (2005), pp. 2404–2411. DOI: 10.2514/1.12366 (cit. on pp. 6, 15).

[77] P. Breitkopf et al. "Moving least squares response surface approximation: Formulation and metal forming applications". In: *Computers & Structures* 83.17–18 (2005). Advances in Meshfree Methods, pp. 1411–1428. DOI: http://dx.doi.org/10.1016/j.compstruc.2004.07.011 (cit. on pp. 6, 15).

[78] F. van Keulen, B. Liu, and R. T. Haftka. "Noise and discontinuity issues in response surfaces based on functions and derivatives". In: *41st Structures, Structural Dynamics, and Materials Conference and Exhibit*. AIAA-00-1363. American Institute of Aeronautics and Astronautics, 2000. DOI: doi:10.2514/6.2000-1363 (cit. on pp. 6, 11, 14).

[79] K. Vervenne and F. van Keulen. "An Alternative Approach to Response Surface Building Using Gradient Information". In: *43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. AIAA-2002-1584. American Institute of Aeronautics and Astronautics, 2002. DOI: doi:10.2514/6.2002-1584 (cit. on pp. 6, 11).

[80] F. van Keulen and K. Vervenne. "Gradient-enhanced response surface building". English. In: *Structural and Multidisciplinary Optimization* 27.5 (2004), pp. 337–351. DOI: 10.1007/s00158-004-0392-1 (cit. on pp. 6, 11).

[81] W. Liu. "Development of Gradient-Enhanced Kriging Approximations for Multidisciplinary Design Optimization". PhD thesis. University of Notre Dame, Indiana, July 2003 (cit. on p. 9).

[82] R. H. Myers and D. C. Montgomery. *Response Surface Methodology. Process and Product Optimization Using Designed Experiments*. John Wiley & Sons Inc., New York, NY, 1995 (cit. on pp. 9, 11).

[83] V. Mazja. *Sobolev spaces*. New York: Springer Verlag, 1985 (cit. on p. 11).

[84] C. Runge. "Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten". In: *Zeitschrift für Mathematik und Physik* 46 (1901), pp. 224–243 (cit. on p. 11).

[85] R. T. Haftka. "Semi-analytical static nonlinear structural sensitivity analysis". In: *AIAA Journal* 31.7 (1993), pp. 1307–1312. DOI: 10.2514/3.11768 (cit. on p. 14).

[86] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. Cambridge, MA, USA: The MIT Press, Jan. 2006, p. 248 (cit. on pp. 14, 18–19, 35).

[87] N. Stander. "The successive response surface method applied to sheet-metal forming". In: *Proceedings, First MIT Conference on Computational Fluid and Solid Mechanics*. June 2001, pp. 481–5 (cit. on p. 15).

[88] P. Lancaster and K. Salkauskas. "Surfaces generated by moving least squares methods". In: *Mathematics of computation* 37.155 (1981), pp. 141–158. DOI: 10.1090/S0025-5718-1981-0616367-1 (cit. on p. 15).

[89] L. Zhou and W. X. Zheng. "Moving least square Ritz method for vibration analysis of plates". In: *Journal of Sound and Vibration* 290.3–5 (2006), pp. 968–990. DOI: http://dx.doi.org/10.1016/j.jsv.2005.05.004 (cit. on p. 16).

[90] U. Häussler-Combe and C. Korn. "An adaptive approach with the Element-Free-Galerkin method". In: *Computer Methods in Applied Mechanics and Engineering* 162.1–4 (1998), pp. 203–222. DOI: http://dx.doi.org/10.1016/S0045-7825(97)00344-7 (cit. on p. 16).

[91] D. Shepard. "A two-dimensional interpolation function for irregularly-spaced data". In: *Proceedings of the 1968 23rd ACM national conference*. ACM. 1968, pp. 517–524 (cit. on p. 17).

[92] J. Mercer. "Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations". In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 209.441-458 (1909), pp. 415–446. DOI: 10.1098/rsta.1909.0016 (cit. on p. 18).

[93] M. G. Genton. "Classes of Kernels for Machine Learning: A Statistics Perspective". In: *J. Mach. Learn. Res.* 2 (Mar. 2001), pp. 299–312 (cit. on p. 18).

[94] L. Laurent, P.-A. Boucard, and B. Soulier. "Generation of a cokriging metamodel using a multiparametric strategy". In: *Computational Mechanics* 51.2 (Feb. 2013), pp. 151–169. DOI: 10.1007/s00466-012-0711-0 (cit. on pp. 18–19, 39, 59).

[95] L. Laurent, P.-A. Boucard, and B. Soulier. "A dedicated multiparametric strategy for the fast construction of a cokriging metamodel". In: *Computers & Structures* 124.0 (2013). Special Issue: KRETA, pp. 61–73. DOI: 10.1016/j.compstruc.2013.03.012 (cit. on pp. 18–19, 28, 35, 39, 59).

[96] M. L. Stein. *Interpolation of Spatial Data: some theory for kriging*. Springer Verlag, 1999 (cit. on pp. 19, 35).

[97] B. Matérn. *Spatial Variation (Lecture NotesStatist. 36)*. Springer, Berlin, 1960 (cit. on pp. 19, 35).

[98] M. Abramowitz and I. Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Applied mathematics series vol. 55,n ° 1972. U.S. Govt. Print. Off., 1964 (cit. on p. 19).

[99] B. A. Lockwood and M. Anitescu. "Gradient-Enhanced Universal Kriging for Uncertainty Propagation". In: *Nuclear Science and Engineering* 170.2 (Feb. 2012), pp. 168–195. DOI: 10.13182/nse10-86 (cit. on pp. 19, 38).

[100] R. L. Hardy. "Multiquadric equations of topography and other irregular surfaces". In: *Journal of Geophysical Research* 76.8 (1971), pp. 1905–1915. DOI: 10.1029/JB076i008p01905 (cit. on p. 22).

[101] M. J. Powell. *Approximation Theory and Methods*. Cambridge University Press, 1981 (cit. on p. 22).

[102] D. Broomhead and D. Lowe. "Multivariable functional interpolation and adaptive networks". In: *Complex Systems* 2 (1988), pp. 321–355 (cit. on p. 22).

[103] B. Beachkofski and R. Grandhi. "Improved Distributed Hypercube Sampling". In: *43rd AIAA/ASME/ASCE Structures, Structural Dynamics, and Materials Conference*. AIAA 2002–1274. 2002. DOI: doi:10.2514/6.2002-1274 (cit. on pp. 23, 47, 63).

[104] R. Schaback. *A Practical Guide to Radial Basis Functions*. Book for teaching. 2007. URL: http://num.math.uni-goettingen.de/schaback/teaching/sc.pdf (visited on) (cit. on p. 23).

[105] M. Stone. "Cross-validatory choice and assessment of statistical predictions". In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1974), pp. 111–147 (cit. on p. 24).

[106] S. Geisser. "The Predictive Sample Reuse Method with Applications". English. In: *Journal of the American Statistical Association* 70.350 (1975), pp. 320–328 (cit. on p. 24).

[107] M. Bompard, J. Peter, J. Desideri, et al. "Surrogate models based on function and derivative values for aerodynamic global optimization". In: *Proceedings of the V European Conference on Computational Fluid Dynamics ECCOMAS CFD 2010*. 2010 (cit. on pp. 24, 28).

[108] S. Rippa. "An algorithm for selecting a good value for the parameter c in radial basis function interpolation". In: *Advances in Computational Mathematics* 11.2 (Nov. 1999), pp. 193–210 (cit. on p. 24).

[109] B. Soulier et al. "Métamodèles à gradients et multiniveaux de fidélité pour l'optimisation d'assemblages". In: *12ème Cologue National en Calcul des Structures, Giens, France, 12-22 mai*. CSMA. 2015 (cit. on p. 28).

[110] S. Ulaganathan et al. "High dimensional Kriging metamodelling utilising gradient information". In: *Applied Mathematical Modelling* (2016 (Online)), pp. –. DOI: http://dx.doi.org/10.1016/j.apm.2015.12.033 (cit. on p. 28).

[111] P. Chauvet. "Aide mémoire de la géostatistique linéaire". In: *Cahiers de Géostatistique, école Nationale Supérieur des Mines de Paris, Centre de Géostatistique, Fontainebleau* Fascicule 2 (1999) (cit. on p. 29).

[112] K. Mardia and R. Marshall. "Maximum likelihood estimation of models for residual covariance in spatial regression". In: *Biometrika* 71.1 (1984), p. 135 (cit. on p. 38).

[113] J. Warnes and B. Ripley. "Problems with likelihood estimation of covariance functions of spatial Gaussian processes". In: *Biometrika* 74.3 (1987), p. 640 (cit. on p. 39).

[114] D. J. Toal et al. "The development of a hybridized particle swarm for kriging hyperparameter tuning". In: *Engineering Optimization* 43.6 (Jan. 2011) (cit. on p. 39).

[115] D. J. J. Toal et al. "An adjoint for likelihood maximization". In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 465.2111 (2009), pp. 3267–3287. DOI: 10.1098/rspa.2009.0096 (cit. on p. 39).

[116] L. Laurent. "Multilevel optimisation of structures using a multiparametric strategy and metamodels". PhD thesis. 61, avenue du Président Wilson, 94230 Cachan: École Normale Supérieure de Cachan, 2013 (cit. on p. 39).

[117] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995 (cit. on pp. 39–40).

[118] V. N. Vapnik and A. Y. Chervonenkis. *Theory of Pattern Recognition [in Russian]*. USSR: Nauka, 1974 (cit. on p. 39).

[119] A. J. Smola et al. "Asymptotically optimal choice of ε-loss for support vector machines". In: *Niklasson, L.F.: 8th International Conference on Artificial Neural Networks 1998. Proceedings. Vol.1 : Skövde, Sweden, 2 - 4 September 1998, ICANN 98*. Springer, 1998, pp. 105–110 (cit. on p. 39).

[120] A. J. Smola and B. Schölkopf. "A tutorial on support vector regression". English. In: *Statistics and Computing* 14.3 (2004), pp. 199–222. DOI: 10.1023/B:STCO.0000035301.49549.88 (cit. on pp. 39–40, 42).

[121] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods*. New York, NY, USA: Cambridge University Press, 2000 (cit. on p. 39).

[122] J. Platt. *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*. Tech. rep. MSR-TR-98-14. Microsoft Research, Apr. 1998, p. 21 (cit. on p. 42).

[123] B. Schölkopf et al. "Shrinking the tube: a new support vector regression algorithm". In: *Advances in Neural Information Processing Systems 11*. Max-Planck-Gesellschaft. Cambridge, MA, USA: MIT Press, June 1999, pp. 330–336 (cit. on p. 45).

[124] B. Schölkopf et al. "New Support Vector Algorithms". In: *Neural Comput.* 12.5 (May 2000), pp. 1207–1245. DOI: 10.1162/089976600300015565 (cit. on p. 45).

[125] C.-C. Chang and C.-J. Lin. "Training V-support Vector Regression: Theory and Algorithms". In: *Neural Comput.* 14.8 (Aug. 2002), pp. 1959–1977. DOI: 10.1162/089976602760128081 (cit. on p. 45).

[126] V. Cherkassky and Y. Ma. "Artificial Neural Networks — ICANN 2002: International Conference Madrid, Spain, August 28–30, 2002 Proceedings". In: ed. by J. R. Dorronsoro. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. Chap. Selection of Meta-parameters for Support Vector Regression, pp. 687–693. DOI: 10.1007/3-540-46084-5_112 (cit. on p. 45).

[127] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998 (cit. on p. 45).

[128] V. N. Vapnik and O. Chapelle. "Bounds on error expectation for support vector machines." eng. In: *Neural Comput* 12.9 (Sept. 2000), pp. 2013–2036 (cit. on p. 45).

[129] O. Chapelle et al. "Choosing Multiple Parameters for Support Vector Machines". English. In: *Machine Learning* 46.1-3 (2002), pp. 131–159. DOI: 10.1023/A:1012450327387 (cit. on p. 45).

[130] M.-W. Chang and C.-J. Lin. "Leave-one-out Bounds for Support Vector Regression Model Selection". In: *Neural Computation* (2005) (cit. on p. 45).

[131] L. Laurent, P.-A. Boucard, and B. Soulier. "Fast Multilevel Optimization using a Multiparametric Strategy and a Cokriging Metamodel". In: *Y. Tsompanakis, B.H.V. Topping, (Editors), Proceedings of the Second International Conference on Soft Computing Technology in Civil, Structural and Environmental Engineering, 6-9 September*. Paper 50. Civil-Comp Press, Stirlingshire, UK, 2011. DOI: 10.4203/ccp.97.50 (cit. on p. 59).

[132] L. Laurent et al. "On the use of gradient-enhanced metamodels for global approximation and global optimization". In: *VII European Congress on Computational Methods in Applied Sciences and Engineering, Hersonissos, Crete, Greece, June 5-10*. 2016 (cit. on pp. 59, 61).

[133] L. Laurent. *MultiDOE (Matlab/Octave Toolbox) https://bitbucket.org/luclaurent/multidoe*. 2016 (cit. on pp. 59, 63).

[134] I. Couckuyt, T. Dhaene, and P. Demeester. "ooDACE Toolbox: A Flexible Object-Oriented Kriging implementation." In: *Journal of Machine Learning Research* 15.1 (2014), pp. 3183–3186 (cit. on p. 61).

[135] S. Ulaganathan et al. "A Matlab Toolbox for Kriging Metamodelling". In: *Procedia Computer Science* 51 (2015), pp. 2708–2713. DOI: http://dx.doi.org/10.1016/j.procs.2015.05.395 (cit. on p. 61).

[136] A. I. J. Forrester, A. Sóbester, and A. J. Keane. "Optimization with missing data". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science* 462.2067 (2006), pp. 935–945. DOI: 10.1098/rspa.2005.1608 (cit. on p. 61).

[137] J. Fritz, I. Neuweiler, and W. Nowak. "Application of FFT-based Algorithms for Large-Scale Universal Kriging Problems". In: *Mathematical Geosciences* 41.5 (2009), pp. 509–533. DOI: 10.1007/s11004-009-9220-x (cit. on p. 62).

[138] J. Hensman, N. Durrande, and A. Solin. "Variational Fourier features for Gaussian processes". In: *ArXiv e-prints* (Nov. 2016) (cit. on p. 62).

[139] M. A. Bouhlel et al. "Improving kriging surrogates of high-dimensional design models by Partial Least Squares dimension reduction". In: *Structural and Multidisciplinary Optimization* 53.5 (2016), pp. 935–952. DOI: 10.1007/s00158-015-1395-9 (cit. on p. 62).

[140] M. A. Bouhlel et al. "An Improved Approach for Estimating the Hyperparameters of the Kriging Model for High-Dimensional Problems through the Partial Least Squares Method". In: *Mathematical Problems in Engineering* 2016 (2016), p. 11 (cit. on p. 62).