Mixed Variable Gaussian Process-Based Surrogate Modeling Techniques: Application to Aerospace Design

Julien Pelamatti, Loïc Brevault, Mathieu Balesdent, El-Ghazali Talbi, Yannick Guerin

Mixed-variable Gaussian process based surrogate modeling techniques, application to aerospace design

Julien Pelamatti *, Loïc Brevault † and Mathieu Balesdent ‡
ONERA, DTIS/Université Paris Saclay, F-91123 Palaiseau Cedex, France

El-Ghazali Talbi §
Inria - Lille Nord Europe, 59650 Villeneuve d'Ascq, France

Yannick Guerin ¶
Centre National d'Etudes Spatiales, Direction des lanceurs, 75012 Paris, France

Within the framework of complex system analyses, such as aircraft and launch vehicles, the presence of computationally intensive models (*e.g.,* finite element models and multidisciplinary analyses) coupled with the dependence on discrete and unordered technological design choices results in challenging modeling problems. In this paper, the use of Gaussian process surrogate modeling of mixed continuous/discrete functions and the associated challenges are extensively discussed. A unifying formalism is proposed in order to facilitate the description and comparison between the existing covariance kernels allowing to adapt Gaussian processes to the presence of discrete unordered variables. Furthermore, the modeling performances of these various kernels are tested and compared on a set of analytical and aerospace engineering design related benchmarks with different characteristics and parameterizations. Eventually, general tendencies and recommendations for such types of modeling problem using Gaussian process are highlighted.

I. Introduction

Nowadays, a growing majority of aerospace design processes heavily relies on the use of computer models and simulations, as it usually represents a faster and cheaper alternative to physical testing, while still providing results accurate enough for most applications. Furthermore, computer simulations can also provide performance estimates in conditions which cannot reasonably be tested (*e.g.,* outer space behaviors). However, computer modeling also presents a number of limitations and drawbacks. Most notably, the computation time associated to the performance simulation of complex systems can be still considerably large. Typical examples involve computational fluid-dynamics analyses and finite element models. This issue may become particularly problematic when the models are used within an optimization

*Research Engineer, ONERA/DTIS.
†Research Engineer, ONERA/DTIS.
‡Research Engineer, ONERA/DTIS.
§Professor, INRIA-Lille Nord Europe.
¶Engineer, CNES Direction des lanceurs.

framework, as they usually must be called a large amount of times in order to determine the solution of the considered problem (*i.e.,* optimal design). In order to partially avoid this issue, a common solution consists in creating a surrogate model of the numerical simulation codes [1], which relies on a mathematical representation of the studied function. These surrogate models usually present a negligible computational cost when compared to the modeled functions, which allows them, if necessary, to be evaluated a large amount of times during the optimization process. However, the addition of this modeling layer also implies a loss of accuracy (*i.e.,* introduction of additional modeling errors) if compared to the actual simulation, the magnitude of which depends on the type of modeling technique that is used as well as on its parameterization.

Among the most popular surrogate modeling techniques used within the framework of complex system design optimization, one may find polynomial regression models [2], Artificial Neural Networks [3], Radial Basis Functions [4], [5], Support Vector Machine Regression [6] and Multivariate adaptive regression spline [7]. More comprehensive discussions on the differences and advantages of the various surrogate modeling techniques can be found in [1] and [8]. The main focus of this paper is relative to the use of specific modeling techniques known as Gaussian Processes [9] (GP), which are increasingly popular methods when modeling functions with low amount of available data and/or when dealing with computationally intensive optimization problems. The interest of GP comes from its ability to provide in addition to a model prediction, an uncertainty prediction model (under the form of a probability distribution) that may be used to quantify the surrogate model uncertainty and to refine the surrogate model via active learning strategies.

Most of the surrogate modeling techniques have originally been developed in order to model continuous problems (*i.e.,* characterized by functions depending solely on continuous variables). However, aerospace design problems are also confronted to the presence of discrete and categorical variables. The discrete choices may characterize, for instance, technological or architectural alternatives. Typical examples of such discrete choices which can be encountered within the aerospace systems are the type of material (*e.g.*, steel, aluminum, composite), the type of rocket propulsion (*e.g.*, liquid, solid, hybrid), the wing configuration (*e.g.*, straight, delta, swept), the number of stages for a launcher, the inclusion of auxiliary sub-systems and/or technologies (*e.g.*, inclusion of lifting surfaces for reusable launch vehicle), *etc*. Without loss of generality, the choices related to the architecture definition can be represented under the form of discrete design variables, sometimes referred to as categorical or qualitative variables, which characterize design alternatives and/or technological choices. Discrete variables are non-relaxable variables defined within a finite set of choices. Discrete variables are typically divided into 2 categories: quantitative and qualitative. As the name suggests, quantitative variables (sometimes also referred to as ordinal) are related to measurable values and by consequence, a relation of order between the possible values of a given variable can be defined (*i.e.*, it is possible to determine whether a value is larger, smaller or equal to another). Quantitative discrete variables are often associated to integer variables, although it is not a necessary requirement. When dealing with qualitative variables, instead, no relation of order can be

defined between the possible values of a given variable. When dealing with mixed continuous/discrete variables, a number of additional challenges, such as the absence of metrics in the discrete search space, need to be addressed in order to be able to create an accurate and reliable surrogate model. In the recent years, a few adaptations of existing continuous surrogate modeling techniques as well as novel modeling methods have been developed in order to be able to model mixed continuous/discrete functions. A few reviews of existing mixed continuous/discrete surrogate modeling techniques can be found in [10], [11] and [12].

In the context of mixed continuous/discrete problems, GP have to be created by processing a training data set $\mathcal{D}$ of n samples $\{\mathbf{x}^i, \mathbf{z}^i, y^i\}$ with $i \in \{1, ..., n\}$. $\mathcal{D}$ can be defined as follows:

$$\mathcal{D} = \left\{\mathbf{X} = \{\mathbf{x}^1, ..., \mathbf{x}^n\} \in F_x^n, \quad \mathbf{Z} = \{\mathbf{z}^1, ..., \mathbf{z}^n\} \in F_z^n, \quad \mathbf{y} = \{y^1, ..., y^n\} \in F_y^n\right\}$$

where $\mathbf{X}$ and $\mathbf{Z}$ are the matrices containing the n continuous and discrete vectors characterizing the training data set, while $\mathbf{y}$ is the vector containing the associated responses (*i.e.,* modeled function). F_x^n, F_z^n and F_y^n are the definition domains of the three previously mentioned matrices. GP are increasingly popular methods when dealing with computationally intensive functions due to their modeling performance and flexibility. They present several advantages when compared to other surrogate modeling techniques. For instance, the training of the hyperparameters can be directly performed with respect to the training data, and does not require auxiliary data sets, as is for instance the case when using cross-validation. Additionally, GP can be parameterized in order to automatically handle noisy training data sets, which can for example be encountered when considering experimental data. Furthermore, although they do not explicitly require it, GP allow the user to easily include problem specific knowledge in the model definition, if available (*e.g.,* specification of the mean function and selection of the kernel). Finally, and most importantly, GP can return an estimated error associated to prediction of the modeled function as a (virtually) computationally free bi-product, under the form of a variance value. This particular characteristic can be useful when performing Surrogate Model-Based Design Optimization, as it allows to define surrogate model refinement criteria characterized by a trade-off between exploitation (*i.e.,* refinement of the incumbent solution) and exploration (*i.e.,* reduction of the model error and uncertainty) of the design space. GP have been first defined to model function depending solely on continuous variables.

This paper focuses on the use of Gaussian Process in order to model computationally intensive functions which depend simultaneously on continuous and discrete variables. Recently, different covariance models [13–16] have been proposed to allow mixed continuous and discrete variables GP. These covariance models involve various mathematical formalisms and subtleties. The main objectives are to propose a unified formalism to facilitate the description and comparison between different covariance models that are at the core of mixed-GP. A comprehensive discussion is provided on the necessary steps required to model mixed-variable functions using GP and the advantages and drawbacks of the different techniques. Moreover, the modeling performances of the described existing approaches are assessed on

a set of analytical test-cases (five toy cases of various dimension and complexity) and aerospace engineering design problems (rocket engine performance simulation and launcher thrust frame structural analysis). General tendencies and recommendations for such a type of surrogate models are provided.

Following this introduction, in the second Section a brief theoretical overview of Gaussian Processes is provided. In the third Section, the construction of valid discrete variable kernels is discussed, and the existing kernels are re-defined within this formalism, thus allowing to better highlight and discuss the differences between the presented kernels. In the fourth Section, the discrete kernels are compared from a performance perspective by testing their modeling capabilities on five of analytical toy cases and two aerospace design test-cases of varying complexity. In the fifth Section the obtained results are described and discussed, and finally the relevant conclusions are provided in the sixth and last Section.

II. Gaussian Process surrogate models

The core concept of Gaussian Process-based surrogate modeling (sometimes also referred to as Kriging [17], [18]) is to predict the response value y^* of a black-box function $f(\cdot)$ for a generic unmapped input $\{\mathbf{x}^*, \mathbf{z}^*\}$ through an inductive procedure. Generally speaking, the larger the training data set is, the more accurate the model will be. However, it is important to note that the choice of samples that are included in the training set also influences the modeling accuracy, as is discussed later in the paper. In its most generic definition, a Gaussian Process is a collection of random variables, any finite number of which have a multivariate joint Gaussian distribution, or alternatively is a generalization of the Gaussian probability distribution[9]. In other words, instead of describing the probability distribution of random scalar or vectorial variables, GP map the probability distribution of the possible regression functions. A generic Gaussian Process $Y(\mathbf{x}, \mathbf{z})$ is characterized by its mean function μ:

$$\mu(\mathbf{x}, \mathbf{z}) = \mathbb{E}[Y(\mathbf{x}, \mathbf{z})] \tag{1}$$

with $\mathbb{E}[\cdot]$ the mathematical expectation, and its covariance function:

$$Cov\left(Y(\mathbf{x}, \mathbf{z}), Y(\mathbf{x}', \mathbf{z}')\right) = E\left[\left(Y(\mathbf{x}, \mathbf{z}) - \mu(\mathbf{x}, \mathbf{z})\right)\left(Y(\mathbf{x}', \mathbf{z}') - \mu(\mathbf{x}', \mathbf{z}')\right)\right] \tag{2}$$

and if a generic function $f(\cdot)$ follows a GP, it can be expressed as:

$$f \sim GP(\mu(\cdot), Cov(\cdot)) \tag{3}$$

The mean function parameterization can technically be defined by the user in order to better represent the modeled function. In most real-life engineering design cases, insufficient information regarding the global trend of the modeled functions is known, and it is therefore complicated to choose the appropriate trend and relative parameterization of $\mu(\cdot)$.

In these cases, it is common practice to consider the regression function $\mu(\cdot)$ as a being constant with respect to the design space [19]:

$$\mu(\mathbf{x}, \mathbf{z}) = \mu \tag{4}$$

The covariance function $Cov(\cdot)$ is usually defined through the use of a parameterized covariance *kernel*. The kernel is used to model the covariance between two elements as a symmetric positive definite function of the values of the coordinates of the elements $k(\{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\})$ (see Section IV). The most known kernels are the Squared Exponential kernel (also known as Radial Basis Function), the Rational Quadratic kernel, the Matérn kernel, *etc.*. The prior mean and prior covariance are updated by relying on the information on the modeled function provided by the data set $\mathcal{D}$, which enables to provide a more insightful model of the considered function. The predicted value f^* of this function at an unmapped location $\{\mathbf{x}^*, \mathbf{z}^*\}$ is then computed under the form of a Gaussian distribution conditioned on the data set [9]:

$$f^*|\mathbf{x}^*, \mathbf{z}^*, \mathbf{X}, \mathbf{Z}, \mathbf{Y} \sim \mathcal{N}\left(\hat{y}(\mathbf{x}^*, \mathbf{z}^*), \hat{s}^2(\mathbf{x}^*, \mathbf{z}^*)\right) \tag{5}$$

In other words, the GP provides the predicted value of the modeled function at an unmapped location $\{\mathbf{x}^*, \mathbf{z}^*\}$ under the form of a mean value $\hat{y}(\mathbf{x}^*, \mathbf{z}^*)$:

$$\begin{aligned} \hat{y}(\mathbf{x}^*, \mathbf{z}^*) &= \mathbb{E}[f^*|\mathbf{x}^*, \mathbf{z}^*, \mathbf{X}, \mathbf{Z}, \mathbf{Y}] \\ &= \mu + Cov(Y(\mathbf{x}^*, \mathbf{z}^*), Y(\mathbf{X}, \mathbf{Z}))Cov(Y(\mathbf{X}, \mathbf{Z}), Y(\mathbf{X}, \mathbf{Z}))^{-1}(\mathbf{Y} - \mu) \\ &= \mu + \boldsymbol{\psi}^T(\mathbf{x}^*, \mathbf{z}^*)\mathbf{K}^{-1}(\mathbf{y} - \mathbf{1}\mu) \end{aligned} \tag{6}$$

and associated variance $\hat{s}^2(\mathbf{x}^*, \mathbf{z}^*)$:

$$\begin{aligned} \hat{s}^2(\mathbf{x}^*, \mathbf{z}^*) &= Var(f^*|\mathbf{x}^*, \mathbf{z}^*, \mathbf{X}, \mathbf{Z}, \mathbf{Y}) \\ &= Cov(Y(\mathbf{x}^*, \mathbf{z}^*), Y(\mathbf{x}^*, \mathbf{z}^*)) - \\ &\quad Cov(Y(\mathbf{x}^*, \mathbf{z}^*), Y(\mathbf{X}, \mathbf{Z}))\, Cov(Y(\mathbf{X}, \mathbf{Z}), Y(\mathbf{X}, \mathbf{Z}))^{-1}\, Cov(Y(\mathbf{X}, \mathbf{Z}), Y(\mathbf{x}^*, \mathbf{z}^*)) \\ &= k(\{\mathbf{x}^*, \mathbf{z}^*\}, \{\mathbf{x}^*, \mathbf{z}^*\}) - \boldsymbol{\psi}^T(\mathbf{x}^*, \mathbf{z}^*)\mathbf{K}^{-1}\boldsymbol{\psi}(\mathbf{x}^*, \mathbf{z}^*) \end{aligned} \tag{7}$$

where $\mathbf{K}$ is the $n \times n$ Gram covariance matrix containing the covariance values between the n sample of the data set:

$$\mathbf{K}_{i,j} = Cov\left(Y(\mathbf{x}, \mathbf{z}), Y(\mathbf{x}', \mathbf{z}')\right) = k(\{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\}) \tag{8}$$

$\mathbf{y}$ is a $n \times 1$ vector containing the responses corresponding to the n data samples:

$$y_i = f(\mathbf{x}^i, \mathbf{z}^i) \quad \text{for } i = 1, \ldots, n \tag{9}$$

$\mathbf{1}$ is a $n \times 1$ vector of ones and finally $\boldsymbol{\psi}$ is an $n \times 1$ vector containing the covariance values between each sample of the training data set and the point at which the function is predicted:

$$\boldsymbol{\psi}_i(\mathbf{x}^*, \mathbf{z}^*) = Cov(Y(\mathbf{x}^*, \mathbf{z}^*), Y(\mathbf{x}^i, \mathbf{z}^i)) = k(\{\mathbf{x}^*, \mathbf{z}^*\}, \{\mathbf{x}^i, \mathbf{z}^i\}) \qquad \text{for } i = 1, \ldots, n \tag{10}$$

III. Mixed-variable Gaussian Process modeling

As mentioned in the introduction, there are relatively few techniques allowing to model functions which depend simultaneously on continuous and discrete variables using GP. Reviews and comparisons of the existing techniques can be found in [12] and [15]. The most commonly used approach when dealing with this kind of functions consists in creating a separate and independent GP model for every category (*i.e.,* combination of discrete variable values) of the considered problem by relying solely on the training data relative to said category. In the remainder of this paper, this approach is referred to as Category-wise surrogate modeling. However, within the framework of computationally intensive function to be modeled it is often unfeasible to provide the amount of data for each category of the problem necessary to model the considered function accurately enough. This issue becomes particularly relevant when dealing with problems characterized by a large number of categories [12]. For this reason, in this paper the concept of mixed-variable surrogate modeling is explored. The underlying idea is to maximize the use of the information provided by the training data set by creating a single mixed continuous/discrete GP with the entirety of the available data rather than distributing the available information over several independent continuous surrogate models. For illustrative purposes, the following simple mixed-variable function $f(\cdot)$ is considered:

$$\begin{aligned} f(x, z) &= \cos(x) + 0.5z \\ x &\in [0, 7], \; z \in \{0, 1\} \end{aligned} \tag{11}$$

If few data samples are available in order to model the function, the prediction provided by the separate continuous GP defined with respect to the two categories of the considered function might be highly inaccurate, as is shown in the top half of Figure 1. If instead the exact same data samples are used in order to create a mixed-variable GP, the prediction is considerably more accurate over the entirety of the search space, while the associated variance is also reduced, as is shown in the lower part of Figure 1. By relying on mixed-variable surrogate modeling rather than category-wise modeling, it is therefore possible to better exploit the information provided by the data samples defined in

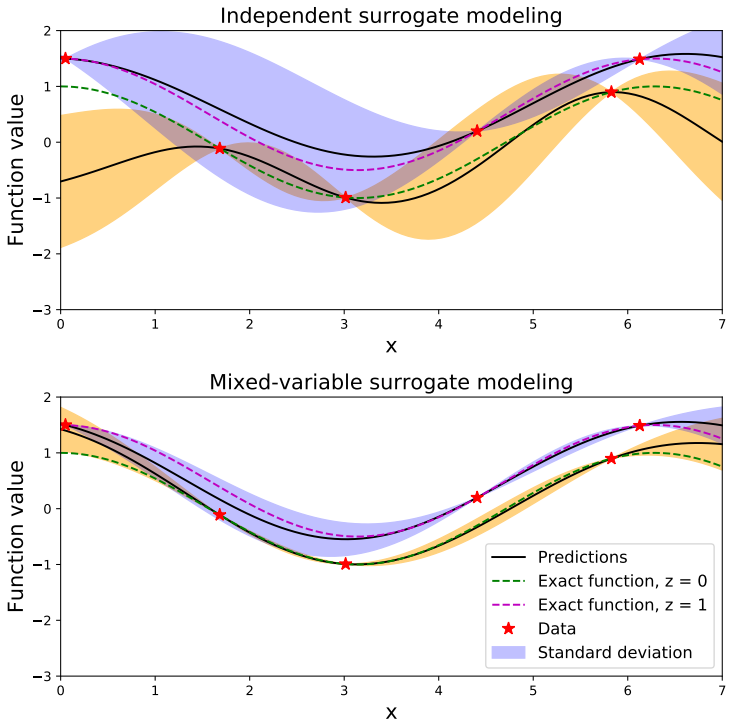


Fig. 1 Comparison between independent category-wise modeling and mixed-variable modeling

the mixed-variable search space, thus improving the modeling accuracy.

IV. Gaussian Process kernels

The covariance function $k(\cdot)$ is the core component of a Gaussian Process based surrogate model [9]. Loosely speaking, the purpose of this function is to characterize the similarity between distinct data samples in the design space with respect to the modeled function. In order for a function $k(\cdot)$ to represent a valid covariance, there are two main requirements [20]. More specifically, the function must be symmetric:

$$k(\{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\}) = k(\{\mathbf{x}', \mathbf{z}'\}, \{\mathbf{x}, \mathbf{z}\}) \tag{12}$$

and positive semi-definite over the input space, *i.e.*:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j k(\{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\}) \geq 0 \tag{13}$$
$$\forall n \geq 1,\ \forall (a_1, \ldots, a_n) \in \mathbb{R}^n \quad \text{and} \quad \forall \{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\} \in F_x \times F_z$$

Alternatively, a positive semi-definite function can also be defined by ensuring that the $n \times n$ matrix constructed by computing each element $\mathbf{K}_{i,j}$ as:

$$\mathbf{K}_{i,j} = k(\{\mathbf{x}^i, \mathbf{z}^i\}, \{\mathbf{x}^j, \mathbf{z}^j\}) \qquad \text{for } i, j = 1, \ldots, n \tag{14}$$

is positive semi-definite:

$$\mathbf{a}^T \mathbf{K}\mathbf{a} \geq 0 \quad \forall\, \mathbf{a} \in \mathbb{R}^n \tag{15}$$

Thanks to the characteristics mentioned above, a valid covariance function can, by construction, be defined as a parameterizable Hilbert space kernel [9], as is discussed in the following paragraphs.

In order to define a unified formalism for the different existing kernels for mixed continuous and discrete variables a brief introduction to Hilbert space kernels is discussed firstly within the purely continuous case, and then it is extended to the mixed-variable case in the subsequent paragraphs. This formalism allows to define all the various approaches under a unified form to ease the understanding and the comparison. Let $F_x \subseteq \mathbb{R}^{n_x}$ be a non-empty set, a kernel function on F_x, $k : F_x \times F_x \to \mathbb{R}$ can be defined if there exists an $\mathbb{R}$-Hilbert space and a mapping $\phi : F_x \to \mathcal{H}$ such that $\forall \mathbf{x}, \mathbf{x}' \in F_x$:

$$k(\mathbf{x}, \mathbf{x}') := \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}} \tag{16}$$

where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ is the inner product on the Hilbert space $\mathcal{H}$. By definition, an inner product defined in a Hilbert space must be [21], [22]:

- bi-linear: $\langle \alpha_1 \phi(\mathbf{x}) + \alpha_2 \phi(\mathbf{x}'), \phi(\mathbf{x}'') \rangle_{\mathcal{H}} = \alpha_1 \langle \phi(\mathbf{x}), \phi(\mathbf{x}'') \rangle_{\mathcal{H}} + \alpha_2 \langle \phi(\mathbf{x}'), \phi(\mathbf{x}'') \rangle_{\mathcal{H}}$
- symmetric: $\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}} = \langle \phi(\mathbf{x}'), \phi(\mathbf{x}) \rangle_{\mathcal{H}}$
- positive: $\langle \phi(\mathbf{x}), \phi(\mathbf{x}) \rangle_{\mathcal{H}} \geq 0, \quad \langle \phi(\mathbf{x}), \phi(\mathbf{x}) \rangle_{\mathcal{H}} = 0$ if and only if $\phi(\mathbf{x}) = 0$

for any $\mathbf{x}, \mathbf{x}', \mathbf{x}'' \in F_x$. The Hilbert space $\mathcal{H}$ can be seen as a space onto which specific features of the considered design variables are mapped. If necessary, multiple mappings and associated kernels can be defined for the same design variable in case several distinct features of the considered variable must be taken into account. In order to better capture and model the dependence of the GP covariance function with respect to the various variables of the design space, both the mapping function $\phi(\cdot)$ and the Hilbert space inner product $\langle \cdot \rangle_{\mathcal{H}}$ can depend on a number of hyperparameters parameters [9]. Depending on the characteristics of the modeled function, some specific kernel choices might be more appropriate than others. In general, a preliminary analysis of the problem at hand is necessary in order to select the kernel which can provide the most accurate and/or robust modeling. Nevertheless, the concepts presented in this work are valid and applicable regardless of the type of kernels which is chosen in order to model the influence of the continuous design variables on the considered functions.

A. Kernel operators

For complex functions, a single kernel may not be sufficient in order to capture the different influences of the various design variables. The main reason for this limitation is that the same single set of hyperparameters is used to characterize the covariance between every dimension of the compared samples. For this reason, it is common practice to combine kernels defined over various sub-spaces of the the design variables definition domain. This results in a valid kernel defined over the entire search space which provides a more accurate modeling of the considered function. It can be shown that kernels can be combined while still resulting in a valid covariance function, as long as the chosen operator allows it [21]. In this paper, the three following kernel operators are considered:

- **Sum**

  Let $k_1(\cdot)$ be a kernel defined on the input space F_{x_1} and $k_2(\cdot)$ be a kernel defined on F_{x_2}. It can be shown that $k(\cdot) = k_1(\cdot) + k_2(\cdot)$ is a valid kernel on the input space $F_x = F_{x_1} \times F_{x_2}$.

- **Product**

  Let $k_1(\cdot)$ be a kernel defined on the input space F_{x_1} and $k_2(\cdot)$ be a kernel defined on F_{x_2}. It can be shown that $k(\cdot) = k_1(\cdot) \times k_2(\cdot)$ is a valid kernel on the input space $F_x = F_{x_1} \times F_{x_2}$. Furthermore, let $k(\cdot)$ be a kernel defined on the input space F_x and $\alpha \in \mathbb{R}^+$, $\alpha k(\cdot)$ is also a valid kernel on F_x.

- **Mapping**

  Let $k(\cdot)$ be a kernel on F_x, let $\tilde{F}_x$ be a set and let $A : \tilde{F}_x \rightarrow F_x$ be a mapping function. Then $\tilde{k}(\cdot)$ defined as $\tilde{k}(\mathbf{x}, \mathbf{x}') := k(A(\mathbf{x}), A(\mathbf{x}'))$ is a kernel on $\tilde{F}_x$

The kernel operators can be used in order to combine kernels in various fashions. Two popular examples are the ANOVA and the exponential kernel [23]:

- ANOVA: $k(\{x_1, x_2\}, \{x_1', x_2'\}) = (1 + k_{x_1}(x_1, x_1')) \times (1 + k_{x_2}(x_2, x_2'))$
- Exponential: $\exp(k(x, x')) = \sum_{n=0}^{n=\infty} \frac{k(x,x')^n}{n!}$

The combination of kernels has two main advantages: first, it enables to define distinct kernels over different design variables, which allows to better capture and model the influence the various design variables on the modeled function. Furthermore, different kernels can also be combined in order to model more accurately different trends characterized by the same design variable or group of variables.

A popular approach when dealing with multidimensional problems consists in defining a single separate kernel $k_d(\cdot)$ for each dimension d the considered problem depends on, and computing the global kernel as the product between one-dimensional kernels [23]:

$$k(\mathbf{x}, \mathbf{x}') = \prod_{d=1}^{n_x} k_d(x_d, x_d') \tag{17}$$

where n_x represents the total dimension of the input space F_x. By doing so, each variable of the modeled function can be associated to a specific kernel parameterization (for instance Squared-exponential, Matérn52, Matérn32 for the

continuous variables) as well as a set of specific associated hyperparameters, thus providing a more flexible modeling of the considered function.

The same logic is relied on in order to deal with mixed-variable problems. Rather than defining a kernel on the mixed continuous/discrete design space valid by construction, the developed approach consists in defining two distinct and independent kernels: $k_x(\cdot)$ with respect to the continuous variables $\mathbf{x}$ and $k_z(\cdot)$ with respect to the discrete variables $\mathbf{z}$. Subsequently, the mixed variable kernel $k(\{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\})$ can be defined as:

$$k(\{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\}) = k_x(\mathbf{x}, \mathbf{x}') \times k_z(\mathbf{z}, \mathbf{z}') \tag{18}$$

Moreover, the kernel defined above can be further decomposed as a product of one-dimensional kernels, similarly to what is shown in Eq. 17. The resulting mixed-variable kernel can then be defined as:

$$k(\{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\}) = \prod_{d=1}^{n_x} k_{x_d}(x_d, x'_d) \prod_{d=1}^{n_z} k_{z_d}(z_d, z'_d) \tag{19}$$

Eq. 17 describes the construction of a kernel characterizing the covariance between continuous data samples. However, it can be noticed that no assumption on the nature of the considered design variables is required for the kernel to be valid [22]. The kernel construction procedure discussed above can therefore be extended to any type of design variable, as long as a proper mapping and associated Hilbert space can be defined. In the following section, the construction of kernels defined with respect to discrete variables (regardless of whether qualitative or quantitative) is discussed. Discrete variables are characterized as non-relaxable variables defined within a finite set of choices, named levels in the following (for instance for the material choices, different levels might be steel, aluminum, composite, *etc*.), and characterized by an absence of relation of order between the possible choices.

V. Discrete kernels

In order to define kernels allowing to characterize the covariance between the values of a discrete variable $\mathbf{z}$, the basic principle is the same as for continuous variables [21]. The kernel is computed as the inner product between the mappings of the two considered discrete variable samples onto a Hilbert space:

$$k_z(\mathbf{z}^i, \mathbf{z}^j) := \langle \phi(\mathbf{z}^i), \phi(\mathbf{z}^j) \rangle_{\mathcal{H}} \tag{20}$$

The main difference with what is described in the previous section lies within the fact that the discrete variables that are considered present a finite number of possible values (*i.e.,* levels) and by the fact that the numerical representation assigned to the considered variables levels is usually arbitrary, and does therefore not yield useful information for the

kernel construction. By consequence, the mapping of a discrete variable onto a Hilbert space may be different than for a continuous variable as it cannot directly depend on the exact values assigned to the considered variable levels. The discrete kernels discussed in the following paragraphs are defined for one-dimensional inputs (*i.e.,* one dimensional discrete variables). The global discrete kernel can then be computed as the product between the one-dimensional kernels, as already shown in Eq. 17:

$$k(\mathbf{z}, \mathbf{z}') = \prod_{d=1}^{n_z} k_{z_d}(z_d, z'_d) \tag{21}$$

An alternative way of describing a discrete kernel can be obtained by exploiting the fact that each discrete variable is characterized by a finite number of levels, as is discussed in [13]. By consequence, the kernel function also returns a finite number of covariance values. It is therefore possible to define a $l \times l$ matrix $\mathbf{T}$ containing the covariance values provided by the kernel:

$$\mathbf{T}_{m,d} = k_z(z^i = z_m, z^j = z_d) \tag{22}$$

By relying on this approach, the validity of a given kernel $k_z(\cdot)$ can be ensured *a posteriori* by showing that the matrix $\mathbf{T}$ is always symmetric and positive semi-definite for bounded hyperparameter values.

In the following paragraphs, existing kernels for discrete variables are presented. Furthermore, a valid construction under the kernel Hilbert space formalism described by Eq. 16 is proposed for each kernel, thus allowing to better highlight and compare their inherent characteristics.

A. Compound Symmetry

The first and most simple discrete kernel to be considered is the Compound Symmetry (CS), characterized by a single covariance value for any non-identical pair of inputs [16]:

$$k(z, z') = \begin{cases} \sigma_z^2 & \text{if} \quad z = z' \\ \theta \cdot \sigma_z^2 & \text{if} \quad z \neq z' \end{cases} \tag{23}$$

where σ_z^2 and $0 < \theta < 1$ are respectively the variance and the hyperparameter associated to the CS kernel. By definition, in case the input data samples are identical, the kernel returns the associated variance value σ_z^2, as it can be seen in Eq. 23. Alternatively, the variance computed between any pair of non-identical data samples is independent from the inputs, and is equal to a value ranging from 0 to σ_z^2. In order to show that the CS is a valid kernel, it is first necessary to

perform the same task with a delta function $\delta(z^i, z^j)$ defined as:

$$k(z, z') = \delta(z, z') = \begin{cases} 1 & \text{if} \quad z = z' \\ 0 & \text{if} \quad z \neq z' \end{cases} \tag{24}$$

Let z be a discrete variable characterized by l levels and let $\phi(\cdot)$ be a mapping of the discrete input space onto a l-dimensional Hilbert space: $\phi(z) : F_z \rightarrow \mathbb{R}^l$. The mapping is defined in such a way that the only non-zero coordinate of the image in the Hilbert space corresponds to the dimension associated to the mapped level. This is sometimes referred to as the dummy coding or one-hot coding of a discrete variable [24]. An example of the mapping described above for a generic discrete variable characterized by 4 levels is provided below:

$$z \in \{z_1, z_2, z_3, z_4\} \rightarrow \begin{cases} \phi(z = z_1) = [1, 0, 0, 0] \\ \phi(z = z_2) = [0, 1, 0, 0] \\ \phi(z = z_3) = [0, 0, 1, 0] \\ \phi(z = z_4) = [0, 0, 0, 1] \end{cases}$$

By defining the inner product on the Hilbert space presented above as a standard Euclidean scalar product, the resulting kernel, valid by construction, is equal to the delta function:

$$\langle \phi(z), \phi(z') \rangle = \phi(z)^T \phi(z') = \delta_z(z, z') \tag{25}$$

Finally, the CS kernel defined in Eq. 23 can be defined through the following combination of delta function and constant kernels:

$$k(z, z') = \sigma_z^2 \left(c_2 k_1(z, z') + c_1\right) = \sigma_z^2 \left((1 - \theta)\delta_z(z, z') + \theta\right) \tag{26}$$

The kernel above is always valid as long as the constants c_1 and c_2 are positive, which is ensured by bounding $\theta \in [0, 1]$. It should be pointed out that the CS kernel presented in the previous paragraphs is nearly identical to the one proposed by Hutter and Halstrup in [25] and [26], although the construction differs. The approach proposed in these works consists in defining a distance in the mixed-variable search space by relying on the concept of Gower distance [27]:

$$d_{gow}\left(\{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\}\right) = \frac{\sum_{d=1}^{n_x} \frac{|x_d - x'_d|}{\Delta x_d}}{n_x} + \frac{\sum_{d=1}^{n_z} \delta(z_d, z'_d)}{n_z} \tag{27}$$

where Δx_d is the domain range of the continuous variable x_d. This distance can then be rescaled and used in order to compute the covariance as a function of the Gower distance between data samples by relying on a continuous squared

exponential kernel (or any other distance based kernel):

$$k(z, z') = \sigma_z \exp\left(-\theta d_{gow}\left(\{\mathbf{x}, \mathbf{z}\}, \{\mathbf{x}', \mathbf{z}'\}\right)^2\right) \tag{28}$$

It can be easily shown that by selecting the appropriate parameterization, the results provided by the kernels defined in Eq. 23 and 28 are identical.

The CS kernel presented in the paragraph above provides a very simple method allowing to model the effect of a given discrete design variable by relying on a single hyperparameter. However, the underlying assumption which is made when considering the CS kernel is that the covariance between any pair of non identical levels of a given discrete variable is the same, regardless of the considered levels. This assumption may often be overly simplistic, especially when dealing with discrete variables which present a large number of levels. In this case, the modeling error introduced by CS kernel can become problematic, and alternative kernels should be considered. It is also important to point out that the CS kernel formulation, as presented in Eq. 23, can only return positive covariance values (by construction). This characteristic further limits the number of suitable applications for this particular covariance function. Finally, it is worth mentioning that Roustant *et al.* have extended the CS kernel in order to model mixed-variable functions characterized by discrete variables with a large number of levels [16]. The underlying idea is to group levels with similar characteristics, thus allowing to compute the covariance between these groups rather than between the levels.

B. Hypersphere decomposition kernel

A second discrete kernel considered in this paper is the hypersphere decomposition kernel, first proposed by Zhou *et al.* [13]. The working principle of the kernel is based on mapping each of the l levels of the considered discrete variable onto a distinct point on the surface of a l-dimensional hypersphere:

$$\begin{aligned} &\phi(z) : F_z \rightarrow \mathbb{R}^l \\ &\phi(z = z_m) = \sigma_z [b_{m,0}, b_{m,1}, \ldots, b_{m,l}]^T \quad \text{for } m = 1, \ldots, l \end{aligned} \tag{29}$$

where $b_{m,d}$ represents the d-th coordinate of the m-th discrete variable level mapping, and is computed as follows:

$$
\begin{aligned}
b_{m,d} &= 1 && \text{for } m \text{ and } d = 1 \\
b_{m,d} &= \cos\theta_{m,d} \prod_{k=1}^{d-1} \sin\theta_{m,k} && \text{for } d = 1, \ldots, m-1 \\
b_{m,d} &= \prod_{k=1}^{d-1} \sin\theta_{m,k} && \text{for } d = m \neq 1 \\
b_{m,d} &= 0 && \text{for } d \geq m \neq 1
\end{aligned}
$$

with $-\pi \leq \theta_{m,d} \leq \pi$. It can be noticed that in the equations above, some of the mapping coordinates are arbitrarily set to 0. This allows to avoid rotation indeterminacies (*i.e.,* an infinite number of hyperparameter sets characterizing the same covariance matrix), while also reducing the number of parameters required to define the mapping. The resulting kernel is then computed as the Euclidean scalar product between the hypersphere mappings presented above:

$$k(z, z') = \phi(z)^T \phi(z') \tag{30}$$

This kernel construction defined above is equivalent to the original formulation [13], in which the discrete kernel is defined as an $l \times l$ symmetric positive definite matrix $\mathbf{T}$ containing the covariance values between the discrete variable levels. In order to ensure the positive definiteness of this matrix, it is defined through the following Cholesky decomposition:

$$\mathbf{T} = \mathbf{L}^T \mathbf{L} \tag{31}$$

where each element of $\mathbf{L}_{i,j}$ is computed as $b_{i,j}$:

$$
\mathbf{L} = \sigma_z \begin{bmatrix}
1 & 0 & \ldots & \ldots & 0 \\
\cos\theta_{2,1} & \sin\theta_{2,1} & 0 & \ldots & \ldots \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
\cos\theta_{l,1} & \sin\theta_{l,1}\cos\theta_{l,2} & \ldots & \cos\theta_{l,l-1}\prod_{d=1}^{l-2}\sin\theta_{l,d} & \prod_{d=1}^{l-1}\sin\theta_{l,d}
\end{bmatrix} \tag{32}
$$

Differently than the CS kernel, the hypersphere decomposition kernel can return a different covariance value for each pair of levels characterizing the considered discrete variable. Furthermore, with the proper hyperparameters this kernel function can also return negative values, as each covariance value is computed as the product between a number of sine and cosine functions, which range from -1 to 1. However, it should also be highlighted that a part of the hyperparameters characterizing this kernel (*i.e.,* $\theta_{l,d}$) influence several covariance values simultaneously and that furthermore, part of the

covariance values can depend on several hyperparameters simultaneously. As a result, determining the optimal value of each hyperparameter might be more complex when compared to simpler kernel parameterizations such as CS.

C. Latent variable kernel

Similarly to the hypersphere decomposition kernel, the Latent Variable (LV) kernel, first proposed by Zhang *et al.* [28], is constructed by mapping the discrete variable levels onto a continuous Hilbert space. However, in the latent variable approach, the discrete variable levels are mapped onto a 2-dimensional Euclidean space regardless of the number of discrete levels, rather than on the surface of an l-dimensional hypersphere. The notion of latent variables comes from the fact that the latent variables are unobserved, they correspond to a mathematical representation of the discrete levels into a 2-dimensional Euclidean space. This mapping can be defined as follows:

$$\begin{aligned} &\phi(z) : F_z \rightarrow \mathbb{R}^2 \\ &\phi(z = z_m) = [\theta_{m,1}, \theta_{m,2}]^T \quad \text{for } m = 1, \ldots, l \end{aligned} \tag{33}$$

where $\theta_{m,1}$ and $\theta_{m,2}$ are the hyperparameters representing coordinates in the 2-dimensional latent variable space onto which the discrete variable level m is mapped. By consequence, the set of hyperparameters characterizing this kernel is represented by the l pairs of latent variable coordinates associated to a given discrete variable. For clarity purposes, an example of the mapping of a discrete variable characterizing the material choice property of a hypothetical system is provided in Figure 2. For instance, each material choice represented in Figure 2 corresponds to a point into a 2-dimensional Euclidean space and the coordinates of the point are the hyperparameters that are optimized (corresponding to the latent variable coordinates).

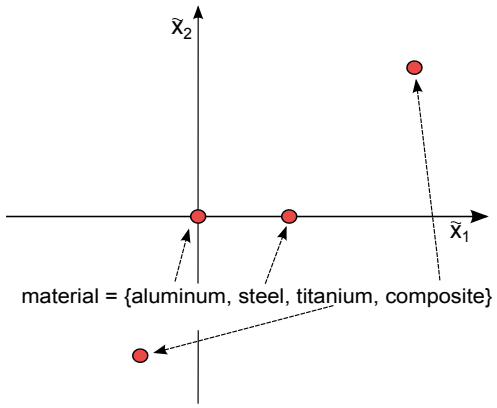


Fig. 2 Example of latent variable mapping for a generic discrete variable characterizing the 'material choice' characteristic.

Let $\phi : F_z \rightarrow \mathbb{R}^2$ be the mapping defined in Eq. 33, by applying the mapping rule discussed in Section IV.A, it can be

shown that a kernel $\tilde{k}(\cdot)$ valid on $\mathbb{R}^2$ can be used in order to define a kernel $k(\cdot)$ valid on F_z in the following fashion [21]:

$$k(z, z') = \tilde{k}(\phi(z), \phi(z')) \tag{34}$$

By consequence, any of the continuous kernels valid on $\mathbb{R}^2$ can be coupled with the latent variable mapping in order to define a valid kernel on the discrete search space F_z. Although in the original formulation of the method, the following squared exponential kernel is considered:

$$k(z, z') = \sigma_z^2 \exp(||\phi(z) - \phi(z')||_2^2) \tag{35}$$

alternative continuous kernels could technically be used without loss of generality. It can be noticed that no lengthscale parameter θ appears in the squared exponential kernel as defined in Eq. 35. The reason behind this is that the distance between the latent variables already directly depends on the hyperparameter values (*i.e.,* the latent variables coordinates), and by consequence a lengthscale parameter would be redundant. Similarly to the hypersphere decomposition kernel previously discussed, the latent variable kernel requires removing the translation and rotation indeterminacies on the latent variable value estimations. Zhang *et al.* suggest fixing one of the latent variables coordinate pair to the origin of the 2-dimensional latent search space (*e.g.,* $\{\theta_{0,1}, \theta_{0,2}\} = \{0, 0\}$), and a second pair on the θ_1 axis (*i.e.,* $\theta_{1,1} = 0$).

The latent variable kernel construction discussed above relies on mapping the discrete variable levels onto a 2-dimensional Hilbert space. This choice is arbitrary as the mapping can technically be performed onto a higher dimensional space in order to provide a larger flexibility and improve the modeling accuracy of the model. However, Zhang *et al.* [28] stated that the theoretical improvement of modeling performance does not compensate the increase in the number of hyperparameters to be tuned, and identify therefore the 2-dimensional latent space as the optimal trade-off between the kernel modeling capabilities and the number of associated hyperparameters. Similarly to the hypersphere decomposition kernel, the latent variable kernel allows to define a distinct covariance value between every pair of levels. However, due to the fact the covariance values are computed as a function of the distance between latent variables in a continuous space, as shown in Eq. 35, the returned values can not be negative, which partially limits the modeling capabilities of the LV discrete kernel.

By mapping the levels of the considered discrete variable onto a 2-dimensional latent space and by characterizing the covariance between these levels as the Euclidean distance between the latent variables, the latent variable kernel provides an intuitive visual representation of how the levels are correlated to each other, as is shown in Figure 2. In this figure, the distance between the latent variables (*i.e.,* red dots) associated to the various materials is inversely proportional to the covariance between the relative levels. For instance, with the considered hyperparameter values, the computed covariance between the aluminum and steel choices would be larger than the one between the aluminum and

the composite choices.

D. Coregionalization

The last discrete kernel considered in this paper is based on the definition of a coregionalization matrix [14]. This approach is originally developed for the modeling of vector valued functions with respect to a continuous search space, *i.e.,* functions returning multidimensional outputs rather than scalar values, $\mathbf{f} : F_x \rightarrow \mathbb{R}^{n_{outputs}}$, where $n_{outputs}$ is the size of the output vector. The underlying idea of the original formulation is to exploit the existing correlation between the various outputs in order to improve the modeling accuracy with respect to the separate and independent modeling of each output. For purely continuous functions, the Linear Model of Coregionalization (LMC) approach, as defined in [29], consists in computing each component prior f_d of the prediction vector as a sum of Q groups of independent latent GP models $u_q^i(\mathbf{x})$, where each GP group q of size R_q shares the same covariance function:

$$f_d(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{Rq} a_{d,q}^i u_q^i(\mathbf{x}) \tag{36}$$

with $a_{d,q}^i$ being scalar coefficients. The notion of latent GP models comes from the fact that the observed output $f(\cdot,\cdot)$ is defined as a linear combination of unobserved GP models $u_q^i(\cdot)$. When considering this kind of models, the resulting vector valued kernel (*i.e.,* one covariance function value per output) can be written as:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \sum_{q=1}^{Q} \mathbf{B}_q k_q(\mathbf{x}, \mathbf{x}') \tag{37}$$

where $k_q(\cdot)$ is the kernel associated to the group of GP u_q, while $\mathbf{B}_q$ is the coregionalization matrix, containing the elements $b_{d,d'}^q$ which characterize the cross-covariance between the predictions d and d' associated to u_q. The LMC can be simplified into the Intrinsic Coregionalization Model (ICM) by considering a single kernel (*i.e.,* $Q = 1$) [30], which results in the following expression of the multiple output covariance matrix:

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B} \otimes \mathbf{k}(\mathbf{X}, \mathbf{X}) \tag{38}$$

where $\otimes$ is the Kronecker product between matrices and $\mathbf{k}(\mathbf{X}, \mathbf{X})$ is the Gram covariance matrix computed on the continuous part of the training data set. The equation above is equivalent to:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \mathbf{B} \times k(\mathbf{x}, \mathbf{x}') \tag{39}$$

Without loss of generality, the concept of coregionalization can be applied to the modeling of mixed variable problems by considering the modeled function as returning an independent output for each level of the considered discrete variable. In this case, the coregionalization matrix becomes the matrix containing the covariance values between the discrete variable levels.

$$\mathbf{B}_{m,d} = k_z(z = z_m, z' = z_d) = \mathbf{T}_{m,d} \tag{40}$$

In order for the ICM to represent a valid kernel, it is necessary for the coregionalization matrix $\mathbf{B}$ to be symmetric and positive semi-definite. Similarly to the hypersphere decomposition, the kernel defined above can be obtained by mapping the levels of the considered discrete variable onto an l-dimensional continuous space:

$$\begin{aligned} &\phi : F_z \rightarrow \mathbb{R}^l \\ &\phi(z = z_m) = [\theta_{m,0}, \theta_{m,1}, \ldots, \theta_{m,l}]^T \qquad \text{for } m = 1, \ldots, l \end{aligned} \tag{41}$$

However, differently than for the hypersphere decomposition kernel, the Cartesian coordinates of the locations onto which the discrete levels are mapped are directly considered as hyperparameters. By defining the inner product on the l-dimensional Hilbert space a Euclidean scalar product, a valid discrete kernel can then be defined:

$$k(z, z') = \phi(z)^T \phi(z') \tag{42}$$

It can be shown that Eq 38 and Eq. 42 yield the same covariance values by exploiting the fact that any positive semi-definite matrix $\mathbf{B}$ can be obtained as the product between a real valued matrix $\mathbf{W}$ and its transpose [31]:

$$\mathbf{B} = \mathbf{W}^T \mathbf{W} \tag{43}$$

If $\mathbf{W}$ is defined as the $l \times l$ matrix containing the hyperparameters onto which each level of the considered discrete variable is mapped, the equivalence between Eq. 38 and Eq. 42 within the scope of coregionalization applied to discrete variable kernels is shown.

It is important to point out that when adapting the ICM to the modeling of mixed-variable functions, the underlying assumption is that only one discrete variable is considered, the levels of which are associated to independent outputs of a vector valued function. However, the kernel defined in the paragraphs above is, by construction, valid on a discrete dimension, regardless of the presence of other discrete variables. It can therefore be applied in a multi-dimensional discrete search space framework without loss of generality. Similarly to the hypersphere decomposition case, the coregionalization approach allows to characterize both positive and negative covariance values between the discrete variable levels. Finally, it can be noticed that differently than the hypersphere decomposition and latent variable kernels,

the coregionalization kernel provides by construction an independent variance value for each level of the modeled variable. This resulting property, known as heteroscedasticity, is further discussed in Section VI.B.

E. Discrete kernel comparison

The various discrete kernels described in the previous paragraphs present specific advantages and weaknesses which must be taken into account when selecting the most suitable kernel parameterization for the modeling of a given problem. For clarity purposes, the main characteristics of the presented discrete kernels are summarized in Table 1. In Figure 3, the number of hyperparameters characterizing each kernel as a function of the number of levels are represented.

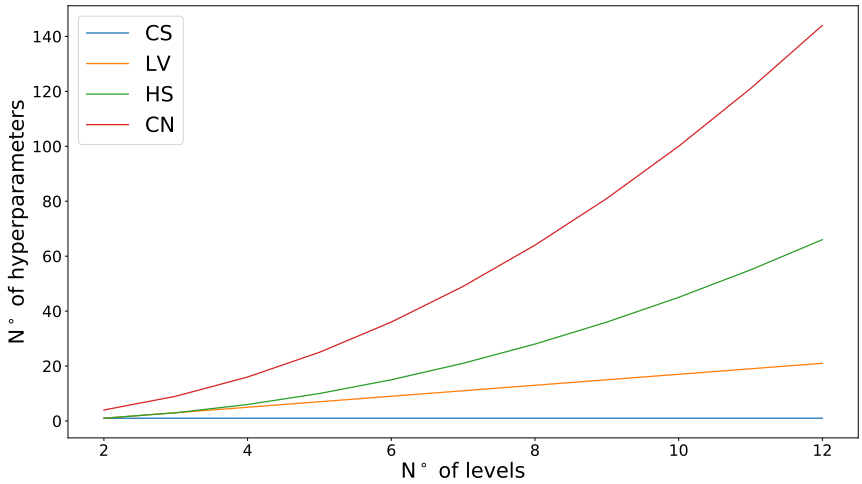


Fig. 3 Number of hyperparameters characterizing each kernel as a function of the number of levels.

Table 1 Advantages and weaknesses of discrete kernel parameterizations. The acronyms refer to the following kernels: CS: Compound Symmetry, HS: Hypersphere Decomposition, LV: Latent Variable, CN: Coregionalization.

Kernels	Hyperparameter scaling w.r.t. l	Different covariance per level pair	Negative covariance values	Inherently heteroscedastic
CS	1	No	No	No
HS	$l(l-1)/2$	Yes	Yes	No
LV	$2l-3$	Yes	No	No
CN	l^2	Yes	Yes	Yes

It can be noticed that the scaling of the number of hyperparameters with respect to the number of levels l of the considered discrete variable varies considerably between the various kernels. This must be taken into account when selecting the appropriate kernel, as a large set of hyperparameters might be difficult to tune when relying on an insufficient amount of training data. On the contrary, simplistic kernel parameterizations, such as the CS, might

be inadequate for variables with large number of levels in case sufficient data is available. It can also be noticed that the coregionalization kernel is the only one which allows to provide a heteroscedastic GP model. However, this limitation can be partially solved by including an additional term in the considered kernel which results in a different and independent variance associated to each discrete level. The description of this extension can be found in Section VI.B.

VI. Considerations on mixed-variable Gaussian Processes

A. Category-wise and level-wise mixed-variable kernels

In the previous paragraphs, only kernels for one-dimensional discrete vectors are discussed. However, most of the considered problems depend on multiple discrete variables, in which case two different approaches can be chosen: treating each dimension (*i.e.,* each discrete variable) separately and independently or defining a kernel on the combinatorial discrete search space. In this paper, these approaches are referred to as Level-wise and Category-wise kernels.

The Level-wise approach is based on considering each discrete variable separately and defining an independent kernel for each one of them. In this case, the various kernels are tasked with computing the covariance between the various levels of each variable. This allows to more easily identify the specific influence of the considered discrete variables and thus select the most suitable kernels, although this might require some previous knowledge on the modeled function. Furthermore, the resulting model usually requires fewer hyperparameters in order to be defined. However, dealing with each dimension separately implies considering that each dimension is independent from the others [32]. In practice, this means that the covariance between two levels (*i.e.,* possible values) of a given discrete variable is independent from the similarity of the 2 compared samples with respect to the other discrete variables. In some cases, this assumption may be false.

Alternatively, the Category-wise kernel definition allows to avoid this kind of issues. The main idea is to define the kernel on the combinatorial discrete search space rather than separately on each dimension of the input space. By doing so, the resulting kernel allows to compute the covariance between discrete variable levels combinations (*i.e.,* categories) rather than between discrete variable levels. However, the drawback of this approach is that the number of hyperparameters required to describe such a kernel tends to scale exponentially with the number of discrete variables. By consequence, larger amounts of data are usually required in order to properly learn the modeled function trends. If this condition is not satisfied, this solution tends to provide worse modeling performance than the level-wise approach. A second drawback of the category-wise approach is that data samples belonging to every category of the considered problem are necessary in order to train the model. This can be problematic when dealing with computationally intensive problems characterized by a large number of categories. Instead, the level-wise approach uses learned level correlations to predict in a particular category which is not present within the data set. It means that, as a category is defined as

a combination of a specific level for each discrete variable, it is not necessary to have in the initial data set a data for each category but only a data for each level of the discrete variables. When only a single discrete variable is considered, level-wise and category-wise approaches are equivalent. Finally, it can be noted that when considering a category-wise approach, the input of the global discrete kernel can be represented under the form of a scalar (*i.e.,* a single discrete variable) characterizing the category the considered sample belongs to rather than the level values of its discrete variables.

It is important to keep in mind that category-wise modeling and category-wise kernel are two distinct and different approaches for the modeling of mixed-variable functions. Indeed, category-wise modeling refers to the creation of a separate and independent continuous GP for every category of the considered function, whereas the category-wise kernel approach refers to the creation of a discrete kernel allowing to compute the covariance between the categories of the considered discrete variables. Separate and independent continuous GP, category-wise kernel and level-wise approach are evaluated in the benchmark of analytical toy cases and aerospace engineering design problems (see Section VII).

B. Surrogate model scedasticity

Within the framework of continuous GP, it is common practice to consider the model as being homoscedatic, which translates to a constant variance value with respect to the design space:

$$k(\mathbf{x}^*, \mathbf{x}^*) = \sigma^2 \tag{44}$$

When the GP kernel is defined as the product of one-dimensional continuous and discrete kernels, as in Eq. 19, the homoscedastic variance can be interpreted as the product between the variances associated to each one of the kernels:

$$\sigma^2 = \prod_d^{n_x} \sigma_x^2 \prod_d^{n_z} \sigma_z^2 \tag{45}$$

Although the homoscedasticity assumption is often acceptable in the purely continuous case, it may be overly simplistic in the mixed continuous/discrete case, as the discrete variables may be associated to large variations of the modeled function global trend. In these cases, it may be necessary to model the kernel variance so that its value depends on the input data sample, thus obtaining what is referred to as a heteroscedastic GP. In the most general case, the heteroscedastic mixed-variable GP variance should vary as a function of both continuous and discrete variables. However, as this work paper is mainly on the discrete aspects of mixed-variable GP and in order to reduce the model training complexity, in this paper the variance is only considered to vary as a function of the discrete variables $\mathbf{z}$:

$$\sigma(\{\mathbf{x}^*, \mathbf{z}^*\}, \{\mathbf{x}^*, \mathbf{z}^*\}) = \sigma^2(\mathbf{z}^*, \mathbf{z}^*) \tag{46}$$

In general, a prior knowledge of the global trend of the modeled function allows to determine which choice, between a homoscedastic and a heteroscedastic model is more suitable for a given problem. In this paper, it is assumed that no prior information on the modeled function is known, and both alternatives are tested for the considered kernel parameterizations. Finally, it is important to note that considering a heteroscedastic discrete kernel has a different influence for Level-Wise and Category-Wise kernels, due to the fact that in the first case a different variance value is associated each level of each discrete variable, whereas in the latter a different variance value is associated to each category of the considered problem.

The heteroscedasticity of a mixed-variable function can be easily shown by considering the simple test-function shown in Figure 4 and defined as follows:

$$f(x,z) = \begin{cases} \sin(7x) & \text{if} \quad z = 0 \\ 2\sin(7x) & \text{if} \quad z = 1 \end{cases} \tag{47}$$

The variances associated to the two categories of the considered function (which in this case are equivalent to the

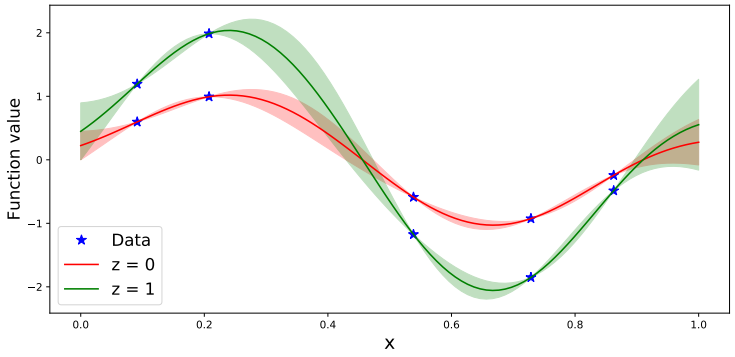


Fig. 4 Category-wise independent modeling of a heteroscedastic mixed variable function

levels of the variable z) can be compared by modeling it with two independent GP models obtained by considering identical continuous data sets, as shown in Figure 4. In general, the optimal variance value depends on the size and on the values of the specific data set with respect to which the models are trained. However, by repeating the comparison over several data sets it can be shown that on average the variance associated to the first category of the problem (*i.e.,* $z = 0$) is approximately 4 times smaller than the one associated to the second category of the problem (*i.e.,* $z = 1$). This is highlighted in Figure 5, where the estimated optimal variance associated to the two categories computed over 20 repetitions is provided for 3 different data set sizes. The results show that in general, a heteroscedastic mixed-variable kernel should therefore yield more accurate predictions when dealing with functions presenting similar characteristics.

The heteroscedastic variance function defined in Eq. 46 can be characterized as a discrete kernel, similar to the ones

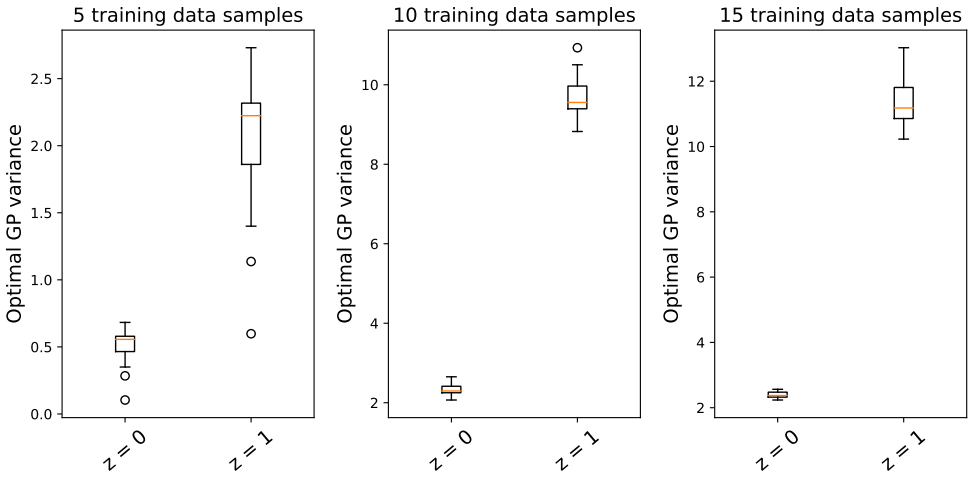


Fig. 5 Estimated optimal variance associated to the two categories of a heteroscedastic example for various data set sizes over 20 repetitions per size

presented in the previous paragraphs:

$$\sigma^2(\mathbf{z}^i, \mathbf{z}^j) = \prod_{d=1}^{n_z} k_{z_d}(z_d^i, z_d^j) \tag{48}$$

where each kernel k_{z_d} can be constructed by mapping each level characterizing the discrete variable z_d onto a hyperparameter characterizing the variance associated to the level:

$$\begin{aligned} &\phi(z) : F_z \rightarrow \mathcal{R} \\ &\phi(z_m) = \theta_m \quad \text{for } m = 1, \ldots, l \end{aligned} \tag{49}$$

and by defining the inner product as a standard product between scalars:

$$k_(z^i, z^j) = \langle \phi(z^i), \phi(z^j) \rangle = \phi(z^i)\phi(z^j) \tag{50}$$

The resulting variance function provides more flexible and accurate modeling when dealing with functions characterized by heteroscedastic behaviors (with respect to the discrete design variables). However, considering a heteroscedastic discrete kernel also results in a larger number of hyperparameters to be trained. Therefore, relying on a heteroscedastic kernel in order to model a homoscedastic function when insufficient data is provided might be counterproductive and yield worse results than a homoscedastic model.

C. Training of the models

The majority of the continuous and discrete kernels presented in the previous paragraphs depends on a number of hyperparameters, such as the latent variables coordinates of Eq. 35, which influence the value returned by the kernel

function, independently from the input data samples. Furthermore, the global kernel function also depends on additional parameters: namely the kernel and likelihood variances (σ^2, σ_n^2) as well as the GP mean μ. The results obtained in this paper are obtained by training the hyperparameters through the optimization of the marginal likelihood function. Let $\boldsymbol{\theta}$ be the vector containing all the hyperparameters characterizing the global kernel $k(\cdot,\cdot)$, the contained values are determined by maximizing the log marginal likelihood:

$$\begin{aligned} \boldsymbol{\theta}^* &= \underset{\boldsymbol{\theta}}{\text{argmax}} \left(\log p(\mathcal{Y}|\mathbf{X},\mathbf{Z},\boldsymbol{\theta})\right) \\ &= \underset{\boldsymbol{\theta}}{\text{argmax}} \left(-\frac{1}{2}\mathbf{y}^T\mathbf{K}^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}| - \frac{n}{2}\log 2\pi\right) \end{aligned} \tag{51}$$

where $\mathbf{K}$ is computed by relying on kernels which are parameterized as a function of $\boldsymbol{\theta}$. Although the hyperparameter vector $\boldsymbol{\theta}$ characterizes both continuous and discrete kernels, all of its values are defined within a continuous search space, thus allowing to rely on common continuous optimization methods. For all the results presented in this paper, the optimization problem defined in Eq. 51 is solved with the help of a Bounded Limited memory Broyden - Fletcher - Goldfarb a Shanno (L-BFGS-B) algorithm [33]. Like most optimization methods of this family, the convergence towards the global optimum of the problem is dependent on the initialization in the design space and cannot always be ensured. This issue is particularly relevant when considering the discrete kernel parameterizations. Multiple random initializations of the optimization algorithm with the selection of the result characterized by the largest likelihood value is therefore considered. Heuristic optimization methods, such as the Differential Evolution [34] or the Covariance Matrix Adaptation - Evolution Strategy [35], could also be tested. However this option is often not computationally tractable due to the prohibitive increase of the overhead computational cost related to the GP training when dealing with large Gram covariance matrices (usually associated to large dimensional problems).

D. Mixed-variable design of experiments

The modeling accuracy which can be provided by a GP is closely related to the training data set given as an input to the model, which is sometimes referred to as Design of Experiments (DoE). In general, larger data sets result in a better modeling accuracy. However, given that this paper lies within the scope of complex and computationally intensive system design, the amount of data available for the creation of the surrogate models is usually very limited. It might therefore be necessary to distribute the available data over the design space in such a way to maximize the information that the GP model can use. In the purely continuous case, the data samples can either be sampled on given distributions, such as a uniform distribution, or alternatively slightly more advanced sampling methods, such as the Latin Hypercube Sampling (LHS) [36] method can be used. In order to extend the concept of design of experiments in

the mixed continuous/discrete search space, the assumption that is made in this paper is that in the absence of problem specific-knowledge, the same amount of data should be placed in each category characterizing the modeled function. Under this assumption, 3 possible alternatives for the creation of the design of experiment are considered:

- Evenly and randomly distributing the data obtained through a single continuous sampling between all of the problem categories.
- Replicating the same continuous sampling in each category of the considered problem (meaning that for each category, the continuous variables take the same values).
- Performing an independent continuous sampling in each category of the considered problem.

In general, samples which are close one another in the continuous search space while presenting different discrete variable values allow the GP model to better determine the covariance functions between the various discrete levels and categories. On the other hand, spacing the samples within the continuous design space allows to better capture the global trend of modeled function in the continuous domain. The first mixed-variable DoE option allows to optimize the usable information within the continuous part of the design space, while limiting the GP ability to determine the covariance between the problem categories. Alternatively, the second option can provide a more accurate computation of the covariance between categories, however it also tends to increase the modeling error for unmapped locations which are distant from the training data set samples, due to the poor coverage of the continuous search space. Finally, the third alternative represents a compromise between the first two options. Further analysis of the challenges related to the mixed-variable sampling issues can be found in [37].

Usually, an analysis of the considered problem and of the amount of available computational resources is required in order to select the most appropriate approach for the creation of a mixed-variable DoE. However, this type of analysis extends beyond the scope of the paper. In this work, the first approach combined with a continuous LHS method is considered for the DoE creation.

VII. Numerical comparison of the different mixed GP-based models

In the previous section, different kernels allowing to characterize the covariance function between discrete variables levels are presented, discussed and compared. In order to assess the modeling performance of the various kernels as well as its dependence on the characteristics of the considered function, a number of analytical and engineering related test-cases characterized by different dimensions, complexities and characteristics are considered. The comparison between the kernels is based on an analysis of the modeling error (*i.e.,* difference between the actual function value and the GP prediction) which is provided under the form of a Root Mean Squared Error (RMSE) calculated on a validation

data set distinct from the training one:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}(\mathbf{x}^i, \mathbf{z}^i) - y(\mathbf{x}^i, \mathbf{z}^i)\right)^2} \tag{52}$$

where N is the size of the validation data set and the values of $y(\cdot)$ and $\hat{y}(\cdot)$ are normalized between 0 and 1:

$$\hat{y}_{norm} = \frac{\hat{y} - y_{min}}{y_{max} - y_{min}} \tag{53}$$

y_{min} and y_{max} represent respectively the smallest and largest values present within the training data set (*i.e.,* in the vector $\mathbf{y}$). In order to analyze the change in absolute and relative performance of the various kernels, the modeling benchmark is performed for different training data set sizes, when possible. Furthermore, in order to take into account the variability of the obtained results caused by the random nature of the training DOE, the benchmark is repeated multiple times for each training data set size (varying as a function of the computational cost and complexity of the problem). The validation data set size is fixed to 1000, and its elements are generated randomly by combining a continuous LHS and a sampling over a uniform discrete distribution in the discrete search space.

In the presented benchmark, the compared discrete kernels are the following: Compound Symmetry (CS), Hypersphere decomposition (HS), Latent Variables (LV) and Coregionalization (CN). Furthermore, for each one of these kernels, 4 alternative variants are considered: the homoscedastic level-wise approach, the heteroscedastic level-wise approach (referred to as _He), the homoscedastic category-wise approach (referred to as _C) and finally the heteroscedastic category-wise approach (referred to as _C_He). However, due to the inherently heteroscedastic nature of the CN kernel, only the level-wise and category-wise variants are considered in the presented results. In order to provide a modeling performance reference, the modeling benchmark is also performed by relying on a Category-Wise (CW) GP, which is obtained by defining an independent purely continuous GP for every category of the considered function. Each one of these GP is trained by relying solely on the data samples of the training set which belong to the corresponding category. The CW GP prediction of the modeled function at an unmapped location is then computed by evaluating the continuous GP corresponding to the sample category at the location characterized by the sample continuous variables. This results in a total number of compared methods equal to 15. Given that the main focus and contribution of this paper is related to the discrete part of mixed-variable kernel, the continuous kernel considered in order to obtain the results presented in the following and throughout this paper is the squared exponential kernel [9]. Finally, it might be worth mentioning that in the heteroscedastic CS parameterization case, the hyperparameters characterizing the heteroscedastic kernel outnumber and outweigh the CS kernel-specific ones. By consequence, the results obtained with the level-wise and category-wise heteroscedastic CS kernel might be mainly driven by the heteroscedastic aspect rather than by the specific kernel parameterization. For details about the benchmark settings and implementations, please refer to the

appendix.

A. Branin function

The first analytical benchmark function to be considered is a modified version of the Branin function [38] characterized by two continuous variables and two discrete variables, each one with 2 levels, thus resulting in overall 4 discrete categories. The analytical definition of this mixed-variable Branin function is the following:

$$f(x_1, x_2, z_1, z_2) = \begin{cases} h(x_1, x_2) & \text{if } z_1 = 0 \text{ and } z_2 = 0 \\ 0.4h(x_1, x_2) + 1.1 & \text{if } z_1 = 0 \text{ and } z_2 = 1 \\ -0.75h(x_1, x_2) + 5.2 & \text{if } z_1 = 1 \text{ and } z_2 = 0 \\ -0.5h(x_1, x_2) - 2.1 & \text{if } z_1 = 1 \text{ and } z_2 = 1 \end{cases} \tag{54}$$

where:

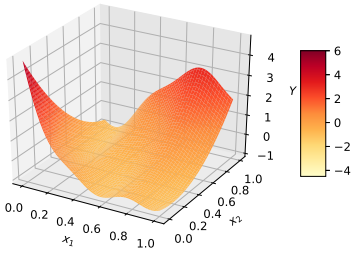
$$\begin{aligned} h(x_1, x_2) =& (((15x_2 - \frac{5}{4\pi^2}(15x_1 - 5)^2 + \frac{5}{\pi}(15x_1 - 5) - 6)^2 + \\ & 10\left(1 - \frac{1}{8\pi}\right)\cos(15x_1 - 5) + 10) - 54.8104)/51.9496 \end{aligned} \tag{55}$$

with

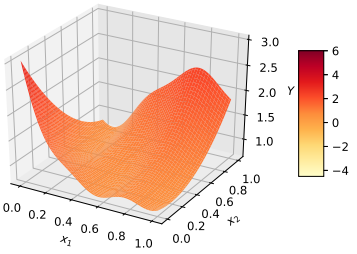
$$x_1 \in [0, 1], \quad x_2 \in [0, 1], \quad z_1 \in \{0, 1\}, \quad z_2 \in \{0, 1\}$$

For illustrative purposes, the responses associated to each category of the mixed-variable Branin function are presented in Figure 6. By analyzing the function definition, two particular characteristics which may influence the modeling performance of the various kernels can be highlighted. The first noticeable characteristic is the presence of a negative correlation between the responses characterized by the 2 levels of the variable z_1: $z_1 = 0$ and $z_1 = 1$ (*i.e.,* the global trends present opposite variations with respect to the continuous variables). Secondly, although the function depends on 2 independent discrete variables, it can be noticed that its categories are defined as a function of the discrete level combinations, rather than as a function of the discrete levels independently.

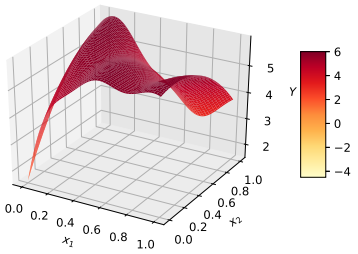
The modeling performance benchmark obtained for the mixed-variable Branin function are provided in Figure 7. These results are obtained over 20 different training data sets of 20, 40 and 80 samples (*i.e.*, 5, 10 and 20 samples per discrete category). Overall, it can be seen that for this test-case most of the considered mixed-variable surrogate modeling methods provide a better modeling accuracy than the independent CW GP. The only exceptions being the homoscedastic and heteroscedastic category-wise CS kernels when insufficient data is provided (*i.e.,* the 20 training data samples case). It can also be noticed that for all 3 training data set sizes, both variants of the level-wise CS and LV



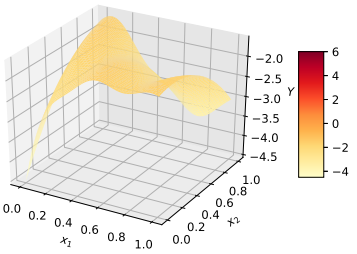
Category 1: $z_1 = 0, z_2 = 0$



Category 2: $z_1 = 0, z_2 = 1$



Category 3: $z_1 = 1, z_2 = 0$



Category 4: $z_1 = 1, z_2 = 1$

Fig. 6 The 4 discrete categories of the Branin function.

kernels provide nearly identical results. This is due to the fact that in case the modeled variable is only characterized by 2 levels, both kernel parameterizations rely on a single hyperparameter to compute the covariance between the levels and therefore yield very similar results. The slight difference between the results obtained with the 2 methods is due to the different hyperparameter initialization in the GP model training. The same similarity is not found for CS and LV category-wise variants, as in this case the discrete kernels model 4 levels instead of 2. A further noticeable trend in the presented results is that both level-wise and category-wise CS and LV kernels tend to provide worse results when compared to the rest of the considered methods. This can be explained by the fact that these kernels can not return negative covariance values, which makes it so that they cannot properly model the negative correlation trends characterizing the levels of z_1. Furthermore, the results show that overall the heteroscedastic variants of the considered kernels tend to produce better results with respect to their homoscedastic counterparts. This is closely related to the heteroscedastic nature of the mixed-variable Branin function which is easily noticeable from its analytical definition. Finally, the results show that for large enough training data sets, the best performing kernels in terms of modeling accuracy are the HS_C, HS_C_He and the CN_C. Additionally to the previous considerations, these results can be explained by the fact that when sufficient data is provided, the category-wise modeling of the discrete variables allows to better capture the fact that each category of the considered mixed-variable Branin is defined as a function of the discrete level combinations, rather than as a function of the discrete levels independently. When sufficient data are available, by

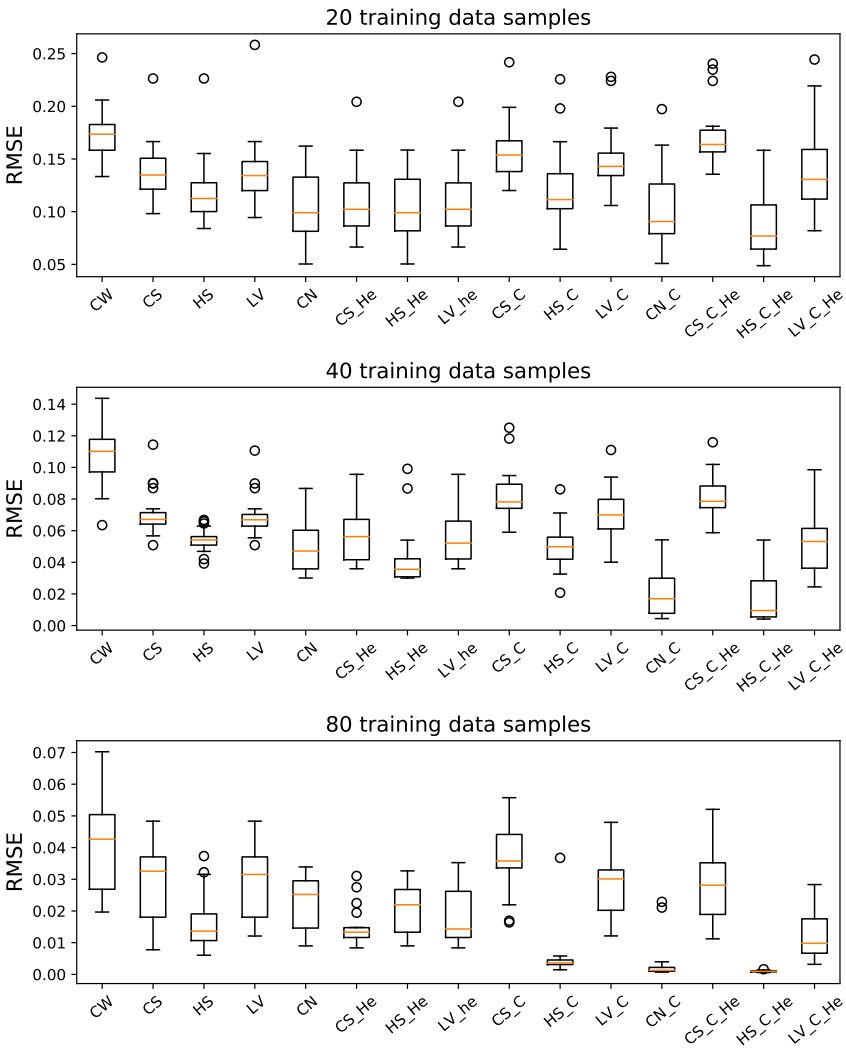


Fig. 7 Comparison of various discrete kernels modeling performance on the mixed-variable Branin function for various training data set sizes over 20 repetitions.

comparing HS_C and HS_C_He modeling, it may be noticed that both offer a small RMSE, but the heteroscedastic variant offer a better modeling illustrating the interest of considering the scedasticity in this case. This difference is less noticeable for smaller data sets, in which case the information provided to the GP is not sufficient to properly model each function category.

B. Augmented Branin function

The second analytical benchmark function to be considered is an augmented (*i.e.,* larger dimension) version of the previously described Branin function, characterized by 10 continuous variables and 2 discrete variables, each one with 2 levels, thus resulting in 4 discrete categories. The main purpose of this test-case is to assess the dependency of the

various discrete kernel parameterizations performance with respect to the size of the continuous design space of the considered problem. In practice, the augmented Branin function is defined as follows:

$$f(x_1, ..., x_{10}, z_1, z_2) = \begin{cases} \tilde{h}(x_1, ..., x_{10}) & \text{if } z_1 = 0 \text{ and } z_2 = 0 \\ 0.4\tilde{h}(x_1, ..., x_{10}) + 1.1 & \text{if } z_1 = 0 \text{ and } z_2 = 1 \\ -0.75\tilde{h}(x_1, ..., x_{10}) + 5.2 & \text{if } z_1 = 1 \text{ and } z_2 = 0 \\ -0.5\tilde{h}(x_1, ..., x_{10}) - 2.1 & \text{if } z_1 = 1 \text{ and } z_2 = 1 \end{cases} \tag{56}$$

where:

$$\tilde{h}(x_1, ..., x_{10}) = \frac{h(x_1, x_2) + h(x_3, x_4) + h(x_5, x_6) + h(x_7, x_8) + h(x_9, x_{10})}{5} \tag{57}$$

with:

$$\begin{aligned} h(x_i, x_j) =& (((15x_j - \frac{5}{4\pi^2}(15x_i - 5)^2 + \frac{5}{\pi}(15x_i - 5) - 6)^2 + \\ & 10\left(1 - \frac{1}{8\pi}\right)\cos(15x_i - 5) + 10) - 54.8104)/51.9496 \end{aligned} \tag{58}$$

and

$$x_1, ..., x_{10} \in [0, 1], \quad z_1 \in \{0, 1\}, \quad z_2 \in \{0, 1\}$$

For this benchmark function, the testing is performed with training data sets of 80, 160 and 320 samples (*i.e.*, 20, 40 and 80 samples per discrete category). The results are provided in Figure 8.

An analysis of these results shows that for small training data sets (*i.e.,* 80 samples), no real difference between the various considered kernels can be highlighted due to the insufficient information to train the GP model. For larger training data sets, instead, it is shown that mixed-variable GP yield overall better results when compared to the independent CW GP, with the exception of the category-wise modeling with a CS kernel. Furthermore, similarly to what is shown for the previously discussed mixed-variable Branin function, heteroscedastic approaches tend to provide a better modeling accuracy, due to the ability to better capture the heteroscedastic trends of the considered function. Finally, it can be seen that if sufficient data is provided, a category-wise approach of the mixed-variable surrogate modeling yields the best results, if combined with heteroscedastic kernels (*i.e.,* CN_C and HS_C_He). This is related to the fashion in which the categories of this particular benchmark function are constructed, as is discussed in the previous paragraphs. In general, it is shown that the relative performance of the considered kernel parameterizations does not significantly vary when the size of the continuous search space is increased. The main noticeable difference is that larger training data sets are required in order to obtain the same range of modeling accuracy, which can be explained by the larger number of hyperparameters required to characterize the continuous kernels.

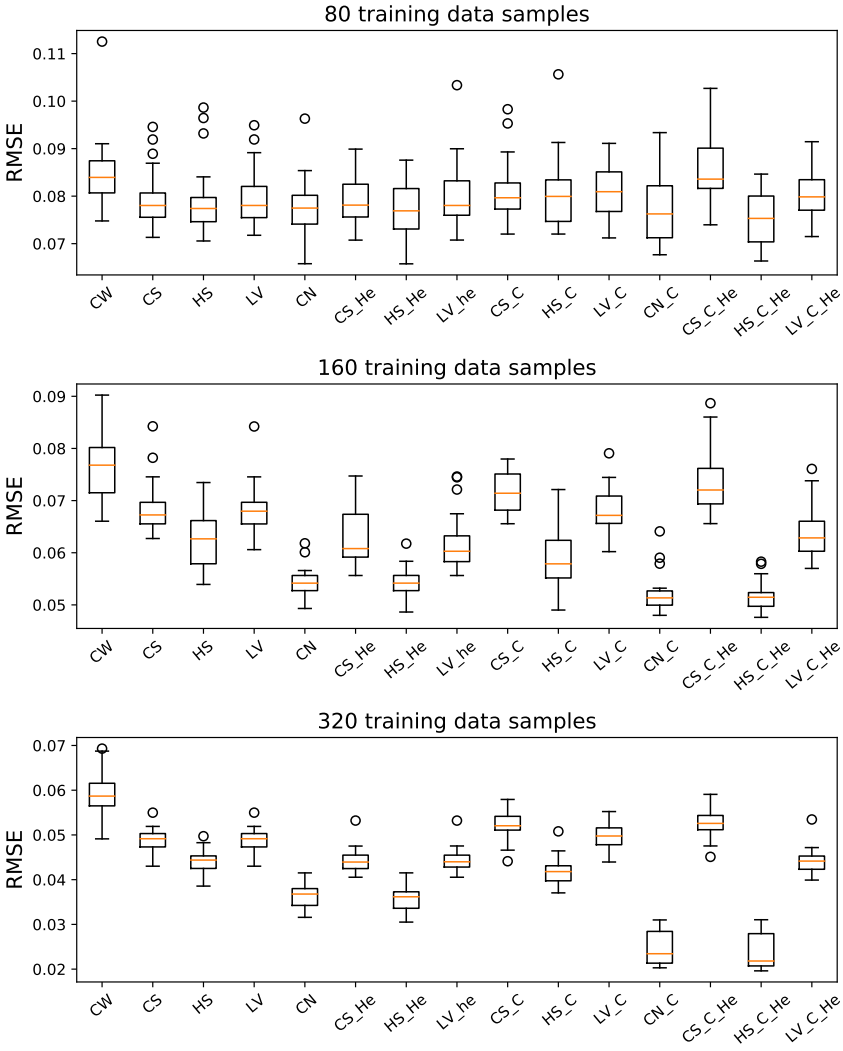


Fig. 8 Comparison of various discrete kernels modeling performance on the augmented mixed-variable Branin function for various training data set sizes over 20 repetitions.

C. Goldstein function

The third analytical benchmark function to be considered in this work is a modified mixed-variable variant of the Goldstein [39] function characterized by 2 continuous variables and 2 discrete variables, each one with 3 levels, thus resulting in 9 discrete categories. This mixed-variable Goldstein function is defined as follows:

$$f(x_1, x_2, z_1, z_2) = h(x_1, x_2, x_3, x_4) \tag{59}$$

with

$$x_1 \in [0, 100], \quad x_2 \in [0, 100], \quad z_1 \in \{0, 1, 2\}, \quad z_2 \in \{0, 1, 2\}$$

The values of x_3 and x_4 are determined as a function of z_1 and z_2 according to the relations defined in Table 2.

	$z_1 = 0$	$z_1 = 1$	$z_1 = 2$
$z_2 = 0$	$x_3 = 20, x_4 = 20$	$x_3 = 50, x_4 = 20$	$x_3 = 80, x_4 = 20$
$z_2 = 1$	$x_3 = 20, x_4 = 50$	$x_3 = 50, x_4 = 50$	$x_3 = 80, x_4 = 50$
$z_2 = 2$	$x_3 = 20, x_4 = 80$	$x_3 = 50, x_4 = 80$	$x_3 = 80, x_4 = 80$

Table 2 Characterization of the Goldstein function discrete categories

$$\begin{aligned} h(x_1, x_2, x_3, x_4) = \; & 53.3108 + 0.184901x_1 - 5.02914x_1^3 10^{-6} + 7.72522x_1^4 10^{-8} - \\ & 0.0870775x_2 - 0.106959x_3 + 7.98772x_3^3 10^{-6} + \\ & 0.00242482x_4 + 1.32851x_4^3 10^{-6} - 0.00146393x_1x_2 - \\ & 0.00301588x_1x_3 - 0.00272291x_1x_4 + 0.0017004x_2x_3 + \\ & 0.0038428x_2x_4 - 0.000198969x_3x_4 + 1.86025x_1x_2x_3 10^{-5} - \\ & 1.88719x_1x_2x_4 10^{-6} + 2.50923x_1x_3x_4 10^{-5} - \\ & 5.62199x_2x_3x_4 10^{-5} \end{aligned} \tag{60}$$

This benchmark function presents similar continuous and discrete design space dimensions to the previously discussed mixed-variable Branin function, however, a few key characteristics differ. First of all, although both functions depend on the same number of discrete variables, the Goldstein function presents a larger number of levels per discrete variable, thus resulting in more than twice as many categories. On the other hand, the Goldstein function can be expected to be less challenging to model due to the absence of negative correlations between discrete levels as well as the presence of a more homoscedastic global trend when compared to the Branin function. Finally, it can also be noticed that each discrete variable has an independent influence on the definition of each problem category, which is not the case for the

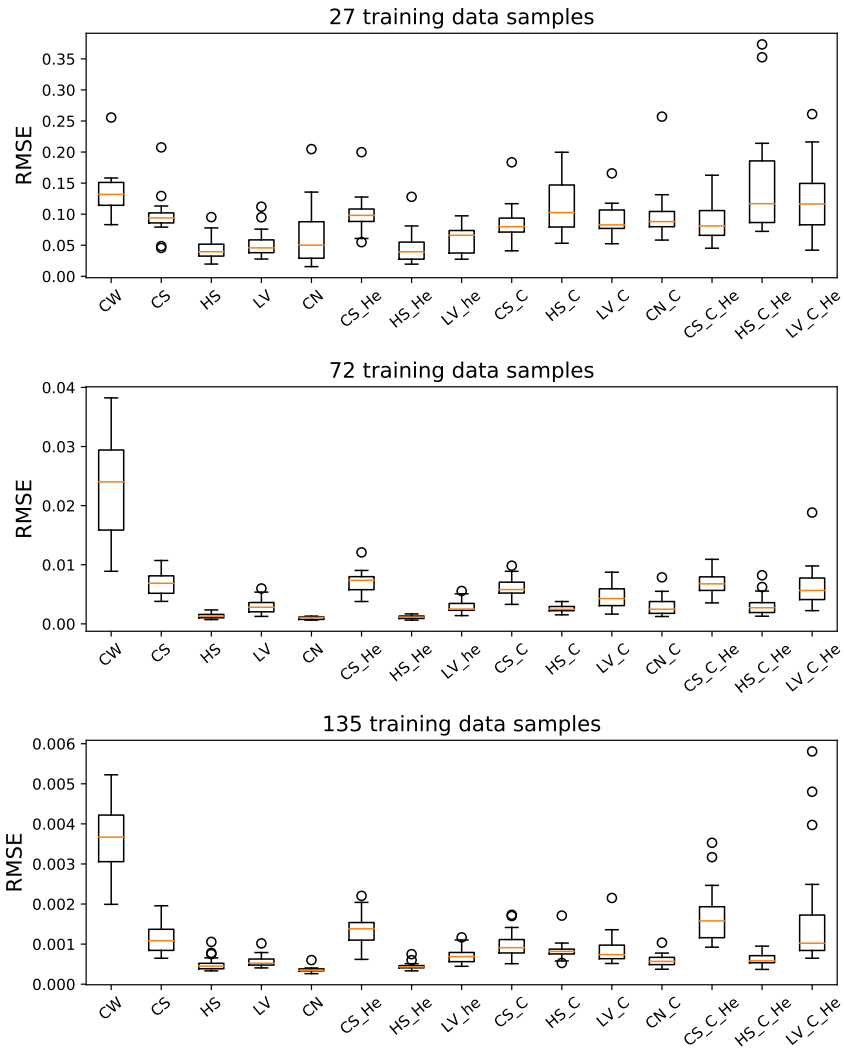


Fig. 9 Comparison of various discrete kernels modeling performance on the mixed-variable Goldstein function for various training data set sizes over 20 repetitions.

two Branin function variants previously considered. For this benchmark function, the testing is repeated with data sets of 27, 72 and 135 samples (*i.e.*, 3, 8 and 15 samples per discrete category). The results obtained for the modeling of the Goldstein function with the considered discrete kernels are provided in Figure 9. As for the previous test-cases, the independent CW GP approach tends to provide considerably worse modeling performance, most notably for larger data sets, due to lower amount of exploited information. A second noticeable trend which can be identified in the results is that for small sized training data sets (*i.e.,* 27 data samples), the category-wise kernels tend to provide a worse modeling accuracy when compared to the ones based on a level-wise approach. This can be explained by the relatively large number of categories with respect to the available data, which can be challenging to model independently. Additionally, it can also be seen that the heteroscedastic variants of the considered kernels do not seem to provide a sizable advantage

in terms of modeling performance with respect to their homoscedastic counterpart. As previously mentioned, this is due to the fact that the Goldstein function is not properly heteroscedastic, and by consequence, considering a heteroscedastic model results in a number of unnecessary hyperparameters to be tuned. Finally, for large enough training data sets the LV, HS and CN kernel parameterizations tend to provide the most promising modeling results, with a slightly better performance for the latter two.

D. Fourth benchmark function

The fourth analytical benchmark function which is considered is an adaptation of a function proposed in [16], characterized by a single continuous variable and 2 discrete variables, presenting respectively 4 and 2 levels, for a total of 8 categories. The considered function is defined as follows:

$$f(x, z_1, z_2) = \begin{cases} \cos\left(7\pi\frac{x}{2} - \frac{z_1}{20}\right) & \text{if } z_2 = 0 \\ \cos\left(7\pi\frac{x}{2} + \left(0.4 + \frac{z_1}{15}\right)\pi - \frac{z_1}{20}\right) & \text{if } z_2 = 1 \end{cases} \tag{61}$$

with $x \in [0, 1]$, $z_1 \in \{6, 7, 8, 9\}$ and $z_2 \in \{0, 1\}$. For illustrative purposes, the 8 categories of the function defined above are plotted in Figure 10. This benchmark is expected to be characterized by homoscedastic trends combined with a

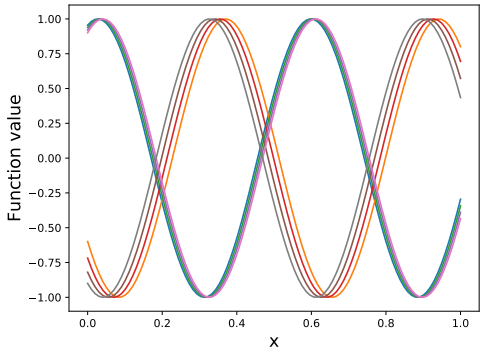


Fig. 10 Categories of the 4th modeling analytical benchmark.

negative covariance between the levels of the discrete variable z_2. For this function, the testing is repeated with data sets of 40, 80 and 120 samples (*i.e.*, 5, 10 and 15 samples per discrete category). The obtained modeling results are provided in Figure 11. Overall, it can be seen that most of the considered discrete kernels provide a considerably better modeling accuracy when compared to the independent category-wise GP, the only exception being the homoscedastic and heteroscedastic category-wise CS and LV kernels, which tend to poorly model the negative correlation trends. The results also show that due to the relatively homoscedastic nature of this particular test-case, the heteroscedastic variants of the considered discrete kernels do not yield sizable advantages in terms of modeling accuracy.

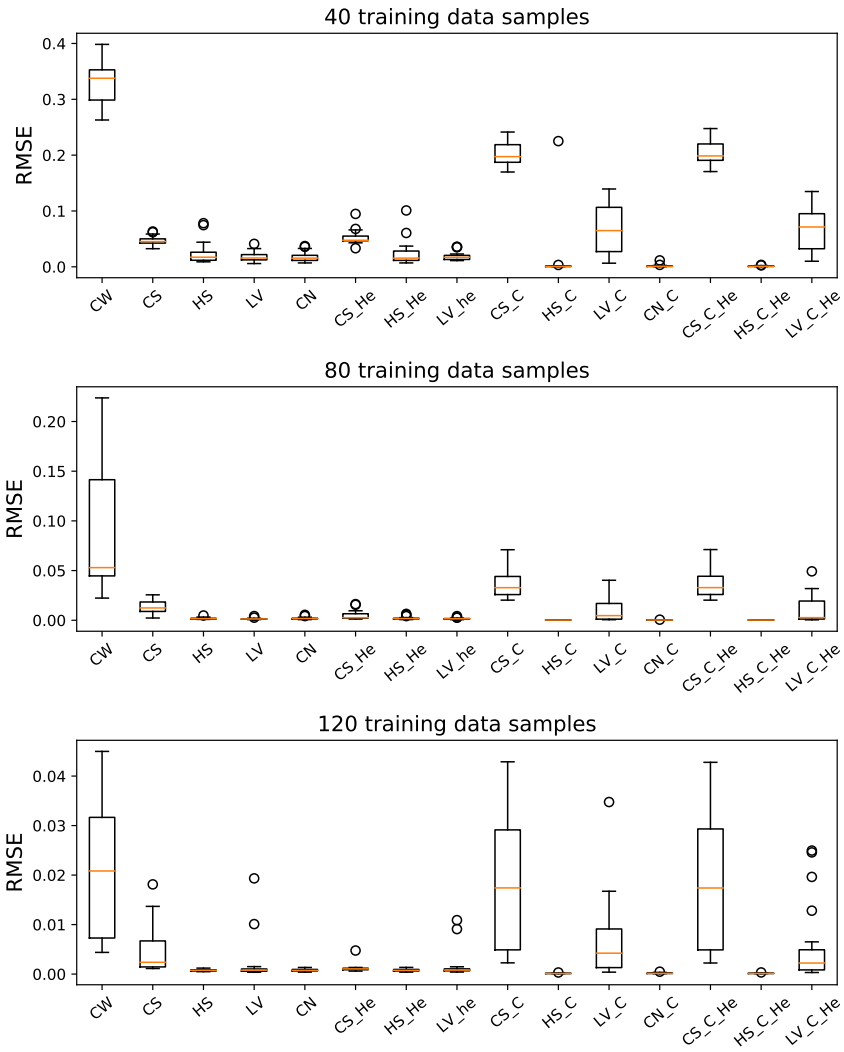


Fig. 11 Comparison of various discrete kernels modeling performance on the fourth analytical benchmark function for various training data set sizes over 20 repetitions.

E. Fifth benchmark function

The fifth and final analytical benchmark function which is considered is an adaptation of a function proposed in [40], characterized by 5 continuous variables and 5 discrete variables. Each one of the discrete variables is associated to 3 levels, which results in a total of 243 categories. This test-function is defined as follows:

$$f(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{5} \frac{x_i(z_{6-i} - 2)}{80} \prod_{i=1}^{5} \cos\left(\frac{x_i}{\sqrt{i}}\right) \sin\left(\frac{50(z_{6-i} - 2)}{\sqrt{i}}\right) \tag{62}$$

with $\mathbf{x} = \{x_1, \ldots, x_5\} \in [0, 1]^5$, and $z_i \in \{1, 2, 3\}$ for $i = \{1, ..., 5\}$. The mixed-variable function defined above is characterized by a large number of categories relatively to the total dimension of its design space. As a consequence,

both the independent category-wise GP and category-wise discrete kernels cannot reasonably be considered for this benchmark, as the number of training data samples would be smaller than the total number of categories, and by extension also lower than the number of hyperparameters to be tuned. For this benchmark the testing is repeated with data sets of 60, 90 and 120 samples. The results obtained with the considered level-wise discrete kernels are provided in Figure 12. The results show that for small training data sets, not enough information is provided in order to properly

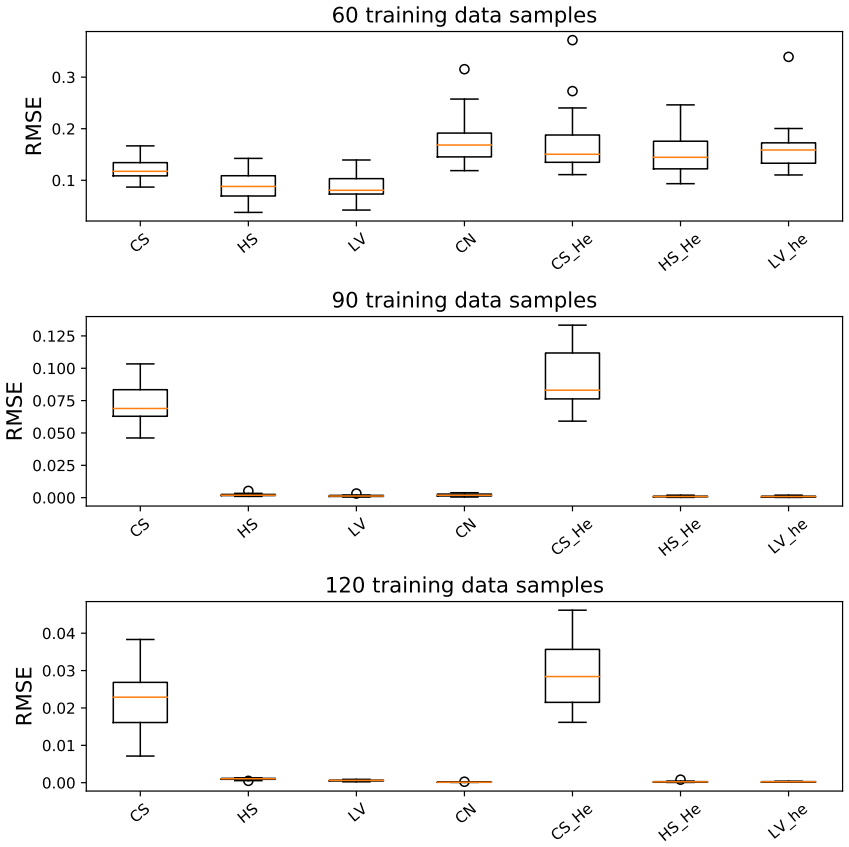


Fig. 12 Comparison of various discrete kernels modeling performance on the fifth analytical benchmark function for various training data set sizes over 20 repetitions.

train heteroscedastic kernels, and by consequence the homoscedastic ones yield slightly better results. For larger training data sets, instead, both variants of the CS kernel are too simplistic and thus unable to properly capture the trends of the considered function. In order to be able to distinguish the differences in performance between the remaining kernel parameterizations, the same results of Figure 12 are presented in Figure 13 without the 2 CS kernel variants, It can then be seen that when sufficient information is provided to the model, the heteroscedastic kernels provide the most accurate modeling of the considered function, with the best results being associated to the CN kernel.

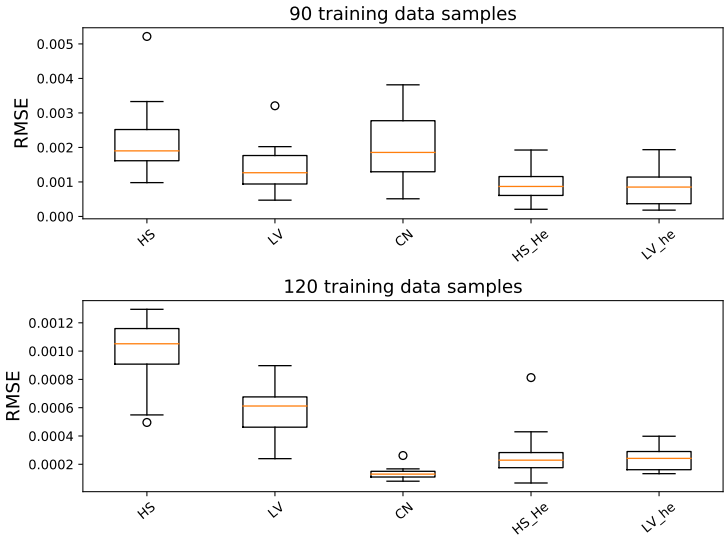


Fig. 13 Comparison of various discrete kernels modeling performance on the fifth analytical benchmark function for various training data set sizes over 20 repetitions.

F. Propulsion performance simulation

In order to better assess the surrogate modeling performance of the discrete kernels discussed in this paper, two aerospace design related test-cases are considered as benchmarks. The first representative test-case is a simulation of the combustion performance of a launcher engine. More specifically, the modeled variable is the specific impulse I_{SP} provided by the engine. The I_{SP} is modeled as a function of the reductant to oxidant ratio O_F, the combustion chamber pressure P_c, the nozzle area ratio ϵ (defined as the ratio between the nozzle throat diameter and the nozzle exit diameter) and the type of reductant and oxidant. The first three variables characterizing the I_{SP} are continuous, while the type of reductant and oxidant are discrete choices which can be represented with the use of discrete variables. A summary of the variables characterizing the problem is provided in Table 3.

Variable	Nature	Min	Max	Levels
O_F	continuous	2.5	5.5	[-]
P_c [bar]	continuous	20	80	[-]
ϵ	continuous	10	60	[-]
Oxidant	discrete	[-]	[-]	O2, N204, F2, H2O2
Reductant	discrete	[-]	[-]	CH4, N2H4, JP-4, H2

Table 3 Variables characterizing the combustion performance simulation test-case

Although this problem is theoretically characterized by a total of 16 categories, it is important to note that not all the combinations of reductant and oxidant can realistically be simulated and therefore only 7 categories are modeled. The combustion simulations necessary to create the training data sets are performed by using the thermo-chemical simulation software 'Chemical Equilibrium with Applications' (CEA) created by NASA [41]. This software simulates

the combustion of gases in a combustion chamber and their expansion in an engine nozzle. The conditions for chemical motion equilibrium are stated in terms of Gibbs and Helmholtz energies [42] or the maximization of the entropy. The system of equations which characterizes the equilibrium and describes its composition is non-linear and therefore iterative methods (such as the Newton-Rhapson method [43]) are used to solve it. In Figure 14, the I_{SP} profiles for the 7 considered combinations of reductant and oxidant are shown as a function of the nozzle ratio ε and the reductant to oxidant ratio O_F.

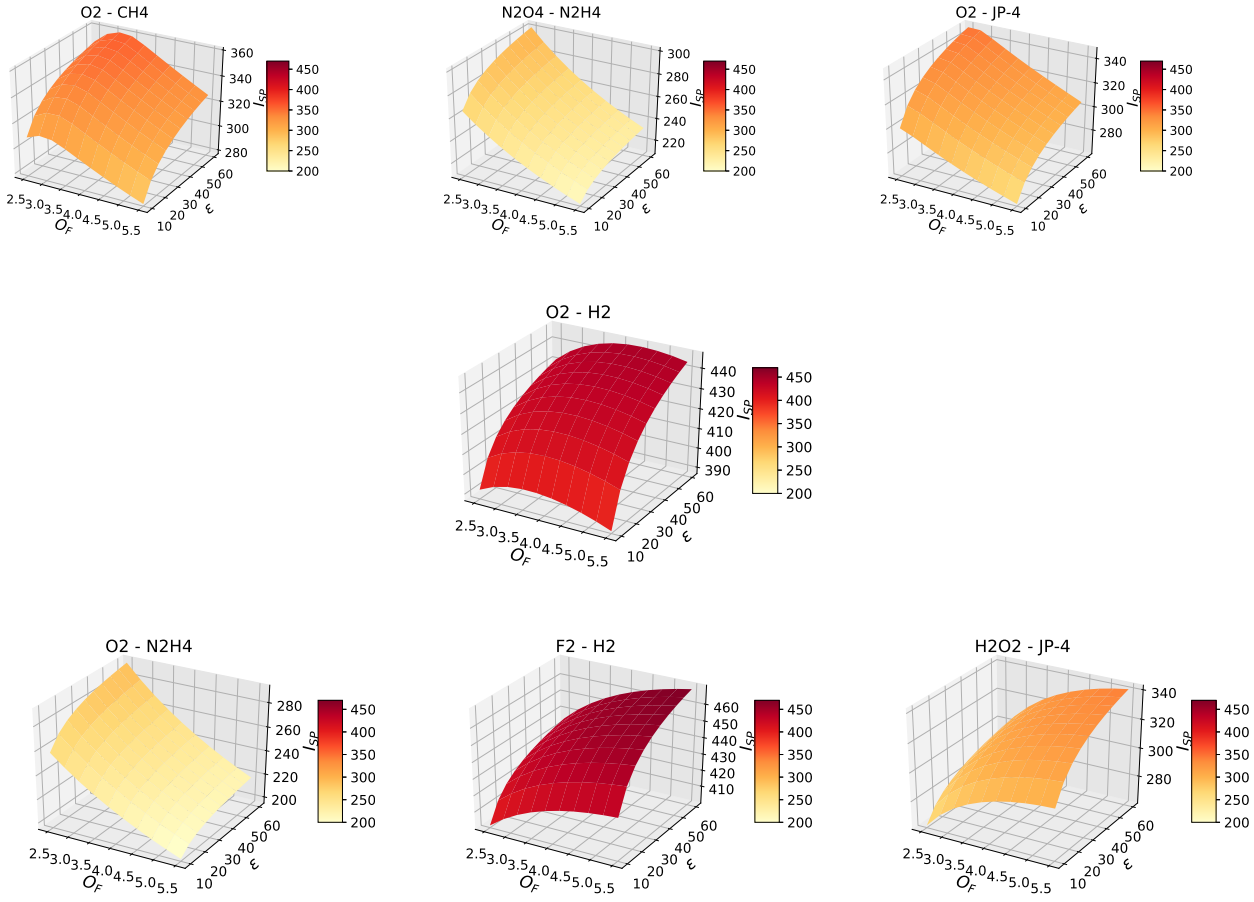


Fig. 14 Specific impulse of a launcher engine as a function of the nozzle ratio ϵ and the reductant to oxidant ratio O_F for various combinations of oxidant and reductant.

The results obtained when modeling the launcher engine specific impulse are provided in Figure 15. These results are obtained over 20 different training data sets of 21, 56 and 105 samples (*i.e.*, 3, 8 and 15 samples per discrete category). As for the previous test-cases, it is shown that mixed-variable modeling provides a more accurate prediction of the modeled function values with respect to independent continuous CW GP. Furthermore, it can be seen that for small training data sets (*i.e.,* 21 samples), heteroscedastic kernels tend to yield worse results when compared to the homoscedastic ones due to the larger number of hyperparameters to train with insufficient data. However, it can be

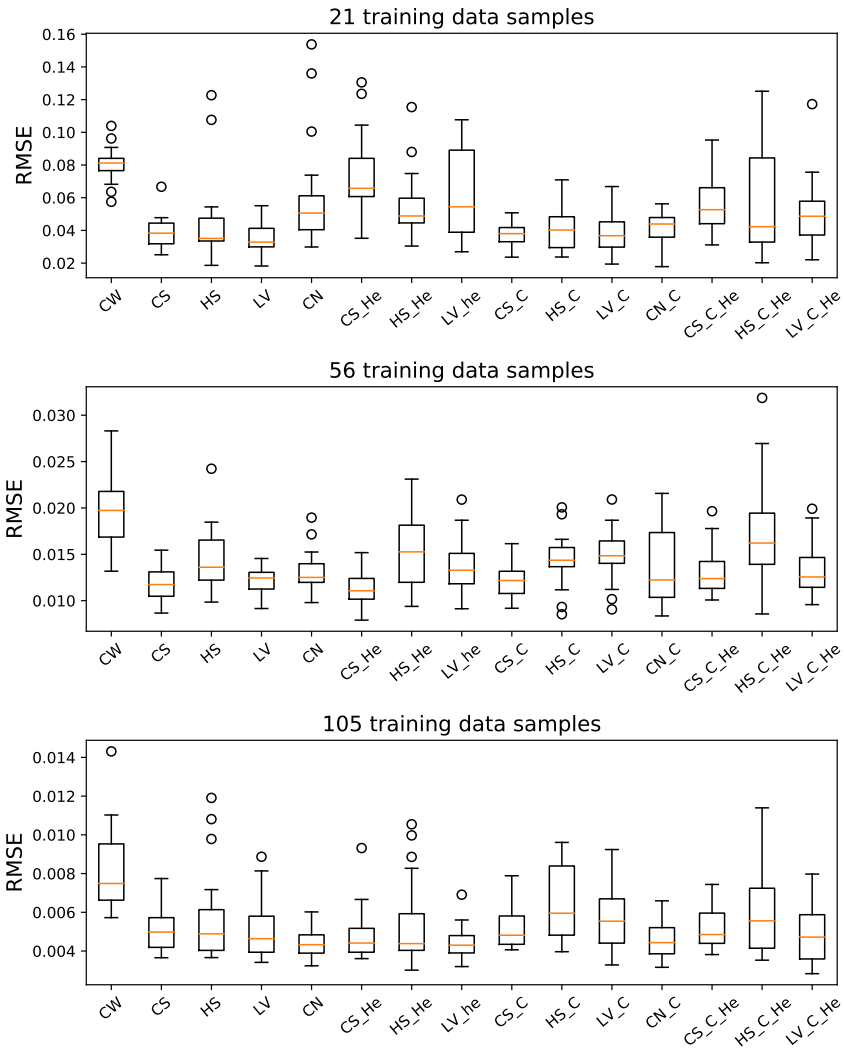


Fig. 15 Comparison of various discrete kernels modeling performance on the launcher engine specific impulse test-case for various training data set sizes over 20 repetitions.

noticed that when sufficient data is provided (*i.e.,* 105 data samples), the relative difference in performance between the compared surrogate models diminishes. This can be explained by the fairly smooth and linear trends of the modeled function with respect to the design variables (as is shown in Figure 14) that often characterizes physical phenomena, which result in an easier modeling process.

G. Thrust frame structural analysis

The second aerospace design test-case that is considered is the modeling of a launcher thrust frame stiffening, commonly performed for preliminary sizing purposes. More specifically, the Ariane 6 aft bay structural characteristics are analyzed. The thrust frame is located at the bottom of the launcher first stage and has the purpose of withstanding

the weight of the launcher as well as the thrust of the two solid rocket boosters during the lift-off phase. A schematic representation of the Ariane 6 aft bay and its location within the launch vehicle system is provided in Figure 16. This structure is composed of a cylindrical outer skin, stiffened by frames and stringers, and an inner body comprising three major frames and an inner skin.

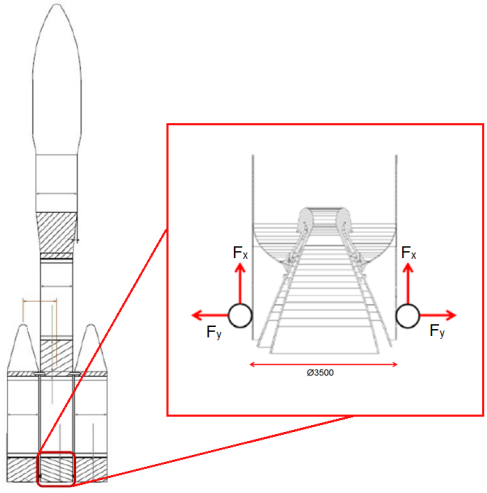


Fig. 16 Ariane 6 PPH launcher aft bay.

The static loads that are considered for this simulation are the longitudinal and lateral thrust of the two solid rocket boosters (see Figure 16). The boundary conditions are defined by modeling the first stage composite bottom skirt clamped to the upper interface of the thrust frame. In this test-case, two different parameters of the considered system are modeled: the maximum von Mises stress on the inner skin of the structure [44] and the upper interface longitudinal over-flux. These are modeled as a function of the inner and outer skins thicknesses t_i, t_o of the 6 regions in which the thrust frame is divided as well as of the number of stringers N_s and the number of frames N_f. The 12 thicknesses are continuous variables while the number of stringers and frames are discrete variables, each one characterized by 3 levels, thus resulting in a total of 9 categories. For illustrative purposes, structural responses on the entire thrust frame structure considering the maximum von Mises stress and the over-flux are provided in Figure 17. For the sake of simplicity, the same range is considered for all the thicknesses variables characterizing the problem. More specifically, the minimum and maximum bounds are 1 and 30 mm, respectively. A summary of the variables characterizing the studied problem is provided in Table 4.

In order to generate the training and testing data sets for this analysis, the MSC Nastran Finite Element Method (FEM) software [45] is used. In practice, a separate finite element model is created for every considered category of the problem (*i.e.*, for every combination of the number of stringers and frames) due to the need of a distinct meshing for every configuration. A number of static FEM analyses is then performed with varying inner and outer skins thicknesses, according to the values present in the data sets, and using the finite element model corresponding to the category the

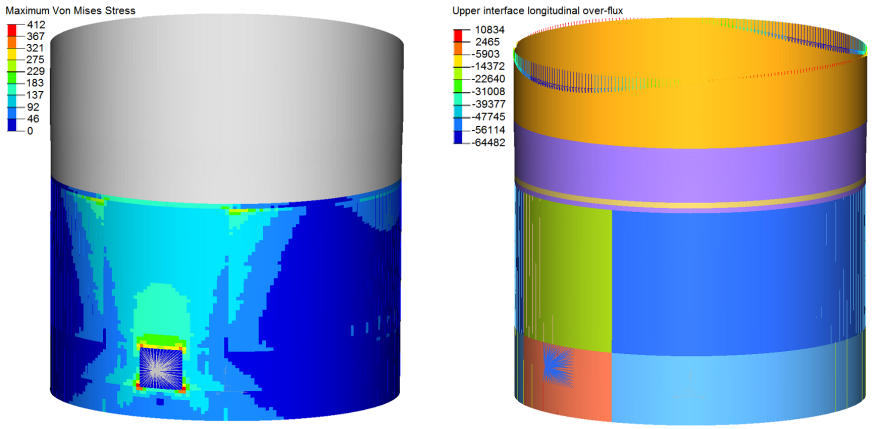


Fig. 17 Examples of structural responses on the entire thrust frame structure. On the left figure the maximum von Mises stress is illustrated, while on the right figure the upper interface longitudinal over-flux is shown.

Variable	Nature	Min	Max	Levels
$t_{i_{1,6}}$ [mm]	continuous	1	30	[-]
$t_{o_{1,6}}$ [mm]	continuous	1	30	[-]
N_s	discrete	[-]	[-]	36, 72, 144
N_f	discrete	[-]	[-]	2, 4, 8

Table 4 Variables characterizing the thrust frame structural analysis test-case

considered sample belongs to. The lift-off conditions are simulated by considering two aligned vertical loads F_x and two opposing horizontal loads F_y, applied on the side of the thrust frame, as illustrated in Figure 16.

Due to the larger computation cost of the static load simulation when compared to the analytical test-cases, the modeling performance benchmark is only performed over 10 repetitions, with each training data set containing 135 samples. The results obtained when performing the surrogate modeling of the maximum inner skin von Mises stress and the upper interface longitudinal over-flux on the thrust frame are shown in the upper and lower plots of Figure 18, respectively. The results show that overall, relying on mixed-variable kernels allows to considerably reduce the modeling error if compared to independent CW GP. It can also be noticed that category-wise mixed-variable kernels yield worse results with respect to a level-wise modeling, due to the nature of the modeled functions as well as the relatively small data sets size. No significant difference in performance between CS/LV kernels and HS/CN kernels can be noticed, which can be explain by a lack of negative correlation trends between the discrete levels and/or categories of the considered problem. Finally, heteroscedastic and homoscedastic kernels yield similar results, which again suggests the absence of considerably heteroscedastic trends in the modeled function.

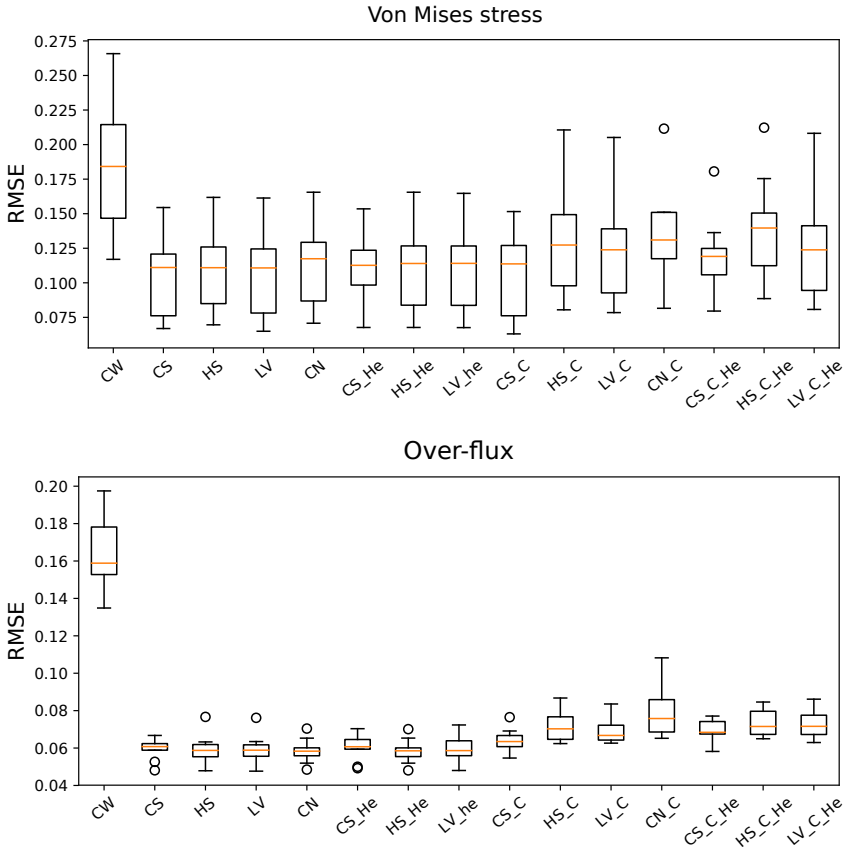


Fig. 18 Comparison of various discrete kernels modeling performance on the thrust frame structural analysis test-case. The modeled values are the maximum inner skin von Mises stress (top) and the upper interface longitudinal over-flux (bottom) over 10 repetitions.

H. Model error estimation accuracy

Within the context of function modeling, the main property which is usually considered is the modeling accuracy, *i.e.,* the difference between the actual and the predicted function value, which can be estimated through criteria such as the RMSE. However, in some cases the validity of the error model (*i.e.,* variance prediction $\hat{s}^2(\cdot)$) may also be relevant. A notable example is the role of the GP models within the Bayesian optimization framework [46], in which the variance prediction drives the exploration aspect of the optimization process. This highlights the necessity for both the prediction to be accurate and the error model to be coherent. As a measure of the error model coherence, the Mean Negative test Log-Likelihood (MNLL) is considered. Similarly to the RMSE, this measure (for the Gaussian case) is computed on a test data set of N samples as:

$$\text{MNLL} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{\sqrt{2\pi\hat{s}^2(\mathbf{x}^i,\mathbf{z}^i)}}\exp\left(-\frac{\left(y(\mathbf{x}^i,\mathbf{z}^i)-\hat{y}(\mathbf{x}^i,\mathbf{z}^i)\right)^2}{2\hat{s}^2(\mathbf{x}^i,\mathbf{z}^i)}\right)\right) \tag{63}$$

and it represents the (negative) likelihood of predicting the exact value of the data set samples with the considered GP model prediction (in terms of both mean prediction and associated variance). As for the RMSE, lower values of the MNLL tend to characterize a better likelihood of explaining the test set and therefore reflect the higher quality of the surrogate model uncertainty estimation. For illustrative purposes, the error model produced by the various discrete kernels considered in this paper is compared on the most representative analytical and design related test-cases.

First the mixed-variable Branin function is considered. As for the modeling accuracy benchmark, the test is repeated 20 times over data sets of 20, 40 and 80 samples and validated on a data set of a 1000 samples. The obtained results are provided in Figure 19.

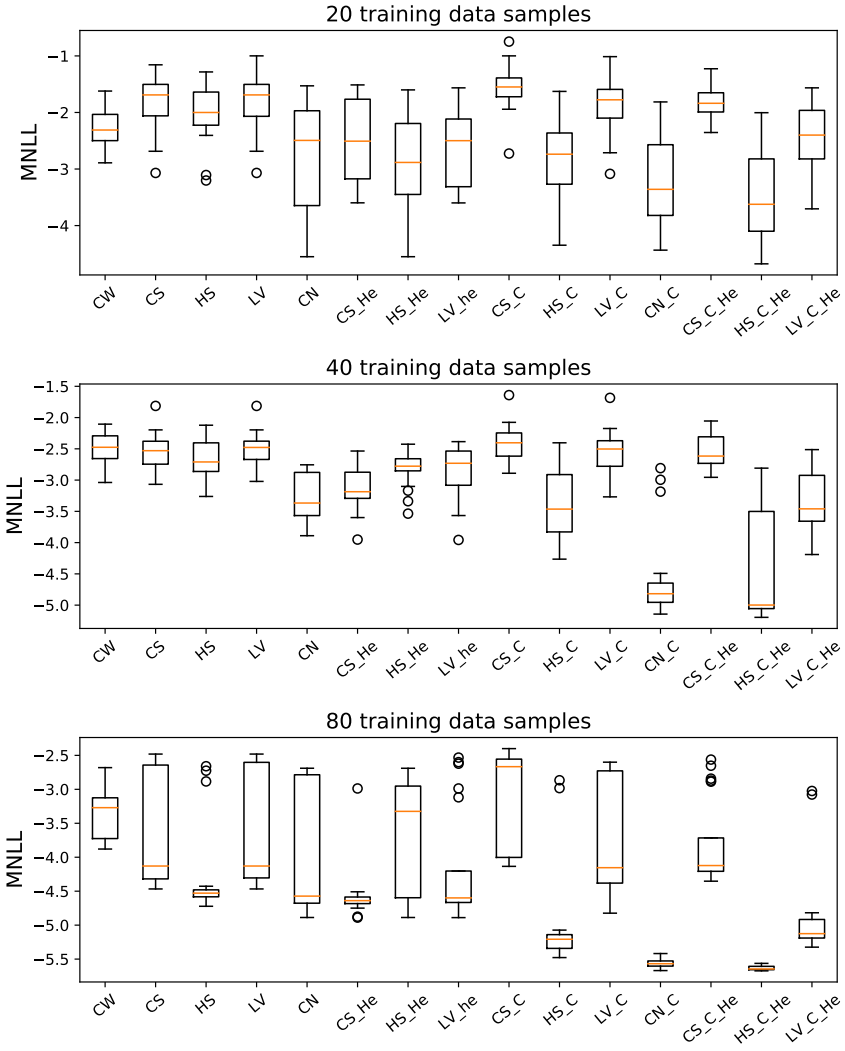


Fig. 19 Comparison of various discrete kernels error model on the mixed-variable Branin function for various training data set sizes over 20 repetitions.

Overall, the results show a similar relative performance between the various kernels as for the prediction benchmark.

The main difference is represented by the independent CW approach which tends to provide an error model accuracy comparable to most of the considered kernels for small sized data sets (*i.e.,* 20 and 40 data samples). A slightly worse relative performance of the heteroscedastic hypersphere decomposition kernel can also be noticed.

The second considered benchmark is the thrust frame structural analysis. In this case, the test is repeated 10 times over data sets of 135 data samples. The obtained results for the modeling of the maximum von Mises stress on the inner skin of the structure and the upper interface longitudinal over-flux are provided in the top and bottom parts of Figure 20, respectively.

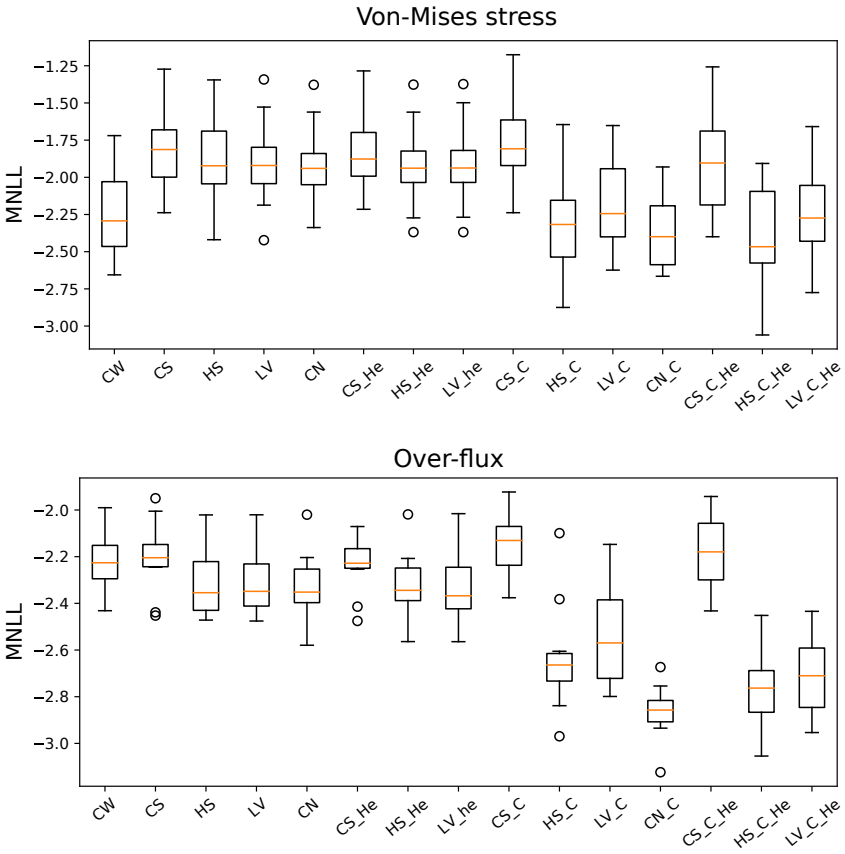


Fig. 20 Comparison of various discrete kernels modeling MNLL on the thrust frame structural analysis test-case. The modeled values are the maximum inner skin von Mises stress (top) and the upper interface longitudinal over-flux (bottom) over 10 repetitions.

Also for this benchmark, the results show that the MNLL provided by the independent CW approach, relatively to the rest of the discrete kernels, is considerably better than its prediction RMSE. Furthermore, it can also be noticed that differently from the modeling benchmark, most of the category-wise approaches overall tend to provide a more accurate error model when compared to the level-wise approaches. This might be explained by the fact that sufficient data is provided in order to properly optimize the variance associated to each category of the problem, resulting in uncertainty

model associated to the prediction that is more likely to contained the unobserved exact simulation results. It illustrates the importance of considering both the accuracy of the prediction and the accuracy of the error model calibration.

VIII. Result synthesis

In the previous sections, the different discrete kernel parameterizations described in Section V have been tested on several analytical and engineering related benchmark functions. Overall, it is shown that mixed-variable GP tend to provide a considerably more accurate modeling performance when compared to the independent category-wise GP as they can rely on the entirety of the training data rather than only the samples associated to a given category. This difference becomes more noticeable when the number of categories associated to the considered problem increases. This can, for instance, be seen when comparing the results obtained for the Branin and Goldstein functions in Figures 7 and 9, characterized respectively by 4 and 9 categories, but the same number of discrete variables.

The results presented in this benchmark of analytical and engineering problems are a direct illustration of the "no free lunch theorem" [47], no mixed-variable GP performs better for all analytical and engineering problems, for all dataset size and for all repetitions. However, some general trends can be identified.

Among the considered kernel parameterizations, the CS is outperformed by the other kernels for most of the test-cases when considering sufficiently large data sets. This is due to the fact that the CS models the covariance between any pair of non identical levels with the same hyperparameter value, which considerably limits its modeling capabilities when confronted with complex problems and/or large number of levels. However, this difference in performance becomes less noticeable when small data sets are used, as the model is easier to train when compared to other kernels characterized by a larger number of hyperparameters. This can, for instance, be seen for the augmented Branin functions in Figure 8. The obtained results also show that, because of its specific distance-based construction, the LV kernel is not suitable when dealing with functions which present negative correlation trends between levels or categories, as it can only return positive covariance values. This is for instance clearly shown in the results obtained for the Branin and augmented Branin functions in Figures 7 and 8. Finally, the hypersphere and coregionalization kernels show similar behaviors on the various considered test-cases, which can be explained by the similar construction and characteristics (*i.e.,* mapping of l levels onto an l-dimensional Hilbert space). The main differences in performance between the two kernels can be identified when dealing with particularly small data sets, in which case the coregionalization is limited by its larger number of hyperparameters, and when dealing with heteroscedastic functions, which the coregionalization kernel can model inherently better.

The difference in performance between the level-wise and category-wise approach based kernels varies depending on the considered function. For mixed-variable functions characterized by a low number of categories (relatively to the number of discrete variables), a category-wise approach may provide a better modeling as it can allow to separately

characterize the covariance between each category, thus better capturing the various trends of the considered function. This can for instance be seen in the results obtained for the Branin function in Figure 7. However, when considering functions characterized by larger number of categories, the level-wise approach based kernels tend to perform better than the category-wise based ones in case small training data sets are provided, as is for instance shown for the Goldstein function in Figure 9. This can be explained by the fact that the number of categories characterizing a given function tends to increase exponentially with the number of discrete variables. The number of hyperparameters necessary to characterize a category-wise kernel follows a similar trend and can therefore become considerably large with respect to the available data, thus resulting in a difficult model training process. For functions characterized by a particularly large number of categories, such as the fifth analytical benchmark, a category-wise approach becomes unfeasible, as it presents more categories than the amount of data samples which can be provided for the GP model training. An analysis of the considered function in terms of number of levels and categories with respect to the size of the available training data set might therefore be necessary in order to assess whether a level-wise or category-wise is more suitable.

The obtained results also show that in the presence of heteroscedastic trends in the modeled function, considering heteroscedastic kernels tends to results in better modeling performance, as can for instance be seen for the fifth analytical benchmark in Figure 12. However, it can also be noticed that when dealing with homoscedastic functions, considering heteroscedastic kernels usually yields results comparable with the homoscedastic ones. Therefore, unless the considered function presents a particularly large discrete design space or the training data is particularly limited, considering heteroscedastic kernels is usually the safest choice, unless problem specific knowledge is available.

Finally, the coherence of the considered discrete kernels error models is also analyzed on two representative test-cases (one analytical benchmark and one engineering design related benchmark) with different data set sizes by relying on the mean negative test log-likelihood as a comparison criterion. Overall, it is shown that the relative performance between kernels for a given modeled function is similar to the one obtained when considering the RMSE. The main noticeable difference with respect to the modeling accuracy benchmark is the good performance of the independent CW GP modeling (*i.e.,* the reference method), especially for small sized training data sets, as is shown in Figures 19 and 20. In fact, in these cases the independent CW GP provides a more coherent error model than several of the compared kernels. This can be explained by the fact that this approach relies on considerably less data in order to build each one of the independent surrogate models, and provides therefore a more conservative variance prediction due to the lack of information with respect to a large portion of the search space.

IX. Conclusions

In this paper, the Gaussian Process based surrogate modeling of fixed-size mixed-variable functions is discussed. It is shown that it is possible to define a mixed-variable kernel by combining purely continuous and purely discrete kernels. Subsequently, the construction of valid discrete kernels is discussed, and the existing alternatives are presented and compared. Furthermore, the resulting mixed-variable Gaussian processes are tested on a number of benchmarks with different characteristics. Overall, the obtained results show that relying on mixed-variable surrogate models rather than on separate and independent continuous GP for each category allows to better exploit the available data and by consequence model more accurately the considered functions. The results also show that depending on the specific characteristics of the modeled function, such as homoscedasticity, number of levels/categories and presence of negative correlations, the relative performance of the compared discrete kernel varies. As a result, the kernel choice must be adapted to the specifics of the considered problems. Overall, in this paper the modeling capabilities of mixed-variable Gaussian processes when dealing with small training data sets are shown. This characteristic, coupled with the fact that Gaussian processes can provide an estimate of the modeling error under the form of a variance as a (virtually) free bi-product of the modeled function prediction, makes mixed-variable Gaussian processes a promising candidate for the surrogate model-based optimization of mixed-variable problems[48]. One of the main limits of the current mixed-variable GPs is the curse of dimension and the explosion of combinatorial choices, which can lead to a high number of hyperparameters to be determined. Several approaches could be investigated to reduce this number such as partial least square techniques and constitutes an interesting perspective.

Funding Sources

This research was co-founded by the Centre National d'Etudes Spatiales (CNES) and by the Office National d'Etudes et de Recherches Aerospatiales (ONERA - The French Aerospace Lab) within the context of a PhD thesis.

Appendix A

In the following, the various discrete kernels presented in this paper are tested on a number of benchmarks. More specifically, 5 analytical functions and 2 engineering-related test-cases are considered. These benchmarks present different characteristics in terms of continuous and discrete design space dimensions, combinatorial design space size, complexity, presence of negative correlation and heteroscedastic trends and category-wise construction. The key properties (from a modeling perspective), simulation details and expected analyses for each benchmark are provided below:

Branin function

- 2 continuous dimensions, 2 discrete dimensions, 9 categories
- Modeling performed for 3 data set sizes: 20, 40, 80

- Presence of negative correlation trends between levels and category-wise construction of the function.

Augmented Branin function

- 10 continuous dimensions, 2 discrete dimensions, 4 categories
- Modeling performed for 3 data set sizes: 80, 160, 320
- Increase of the continuous design space size with respect to the Branin function (but identical discrete design space characteristics). Presence of negative correlation trends between levels and category-wise construction of the function.

Goldstein function

- 2 continuous dimensions, 2 discrete dimensions, 9 categories
- Modeling performed for 3 data set sizes: 27, 72, 135
- Increase in the number of categories with respect to the Branin function (but identical discrete design space size). Overall homoscedastic trend.

Analytical benchmark N.4

- 1 continuous dimensions, 2 discrete dimensions, 8 categories
- Modeling performed for 3 data set sizes: 40, 80, 120
- Overall homoscedastic with negative correlation trends between levels.

Analytical benchmark N.5

- 5 continuous dimensions, 5 discrete dimensions, 243 categories
- Modeling performed for 3 data set sizes: 60, 90, 120
- Large number of categories, impossibility of relying on category-wise approaches.

Propulsion performance simulation

- 3 continuous dimensions, 2 discrete dimensions, 7 categories (16 theoretical)
- Modeling performed for 3 data set sizes: 21, 56, 105
- Realistic simulation. Not all level combinations are physically feasible, which results in not all categories being present in the DoE.

Thrust frame structural analysis

- 12 continuous dimensions, 2 discrete dimensions, 9 categories
- Modeling performed for 1 data set sizes: 135
- Realistic simulation. Linear trends with respect to the continuous sizing variables.

Implementation

The results presented in the following paragraphs are obtained with the following implementation. The model comparison overhead routine is written in Python 3.6. The compared discrete kernels are implemented within the framework of a GPflow [49], a Python based toolbox for GP-based modeling relying on the Tensorflow machine learning platform

[50] (version 1.13). The GP training is performed with the help of a Bounded Limited memory Broyden - Fletcher - Goldfarb a Shanno (L-BFGS-B) algorithm [33].

References

[1] Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., and Kevin Tucker, P., "Surrogate-based analysis and optimization," *Progress in Aerospace Sciences*, Vol. 41, No. 1, 2005, pp. 1–28. https://doi.org/10.1016/j.paerosci.2005.02.001.

[2] Draper, N. R., and Smith, H., *Applied regression analysis*, Vol. 326, John Wiley & Sons, 1998.

[3] Papadrakakis, M., Lagaros, N. D., and Tsompanakis, Y., "Structural optimization using evolution strategies and neural networks," *Computer methods in applied mechanics and engineering*, Vol. 156, No. 1-4, 1998, pp. 309–333. https://doi.org/10.1016/S0045-7825(97)00215-6.

[4] Dyn, N., Levin, D., and Rippa, S., "Numerical procedures for surface fitting of scattered data by radial functions," *SIAM Journal on Scientific and Statistical Computing*, Vol. 7, No. 2, 1986, pp. 639–659. https://doi.org/10.1137/0907043.

[5] Fang, H., and Horstemeyer, M. F., "Global response approximation with radial basis functions," *Engineering Optimization*, Vol. 38, No. 04, 2006, pp. 407–424. https://doi.org/10.1080/03052150500422294.

[6] Smola, A. J., and Schölkopf, B., "A tutorial on support vector regression," *Statistics and computing*, Vol. 14, No. 3, 2004, pp. 199–222. https://doi.org/10.1023/B:STCO.0000035301.49549.88.

[7] Friedman, J. H., et al., "Multivariate adaptive regression splines," *The annals of statistics*, Vol. 19, No. 1, 1991, pp. 1–67. https://doi.org/https://www.jstor.org/stable/2241837.

[8] Wang, G. G., and Shan, S., "Review of Metamodeling Techniques in Support of Engineering Design Optimization," *Journal of Mechanical Design*, Vol. 129, No. 4, 2007, p. 370. https://doi.org/10.1115/1.2429697.

[9] Rasmussen, C. E., and Williams, C. K. I., *Gaussian processes for machine learning*, MIT Press, 2006.

[10] Bartz-Beielstein, T., and Zaefferer, M., "Model-based methods for continuous and discrete global optimization," *Applied Soft Computing*, Vol. 55, 2017, pp. 154–167.

[11] Meckesheimer, M., Barton, R. R., Simpson, T., Limayem, F., and Yannou, B., "Metamodeling of Combined Discrete/Continuous Responses," *AIAA JOURNAL*, Vol. 39, No. 10, 2001.

[12] Swiler, L. P., Hough, P. D., Qian, P., Xu, X., Storlie, C., and Lee, H., "Surrogate models for mixed discrete-continuous variables," *Constraint Programming and Decision Making*, Springer, 2014, pp. 181–202. https://doi.org/10.1007/978-3-319-04280-0_21.

[13] Zhou, Q., Qian, P. Z. G., and Zhou, S., "A Simple Approach to Emulation for Computer Models With Qualitative and Quantitative Factors," *Technometrics*, Vol. 53, No. 3, 2011, pp. 266–273. https://doi.org/10.1198/TECH.2011.10025.

[14] Alvarez, M. A., Rosasco, L., Lawrence, N. D., et al., "Kernels for vector-valued functions: A review," *Foundations and Trends® in Machine Learning*, Vol. 4, No. 3, 2012, pp. 195–266.

[15] Zhang, Y., and Notz, W. I., "Computer experiments with qualitative and quantitative variables: A review and reexamination," *Quality Engineering*, Vol. 27, Taylor & Francis, 2015, pp. 2–13. https://doi.org/10.1080/08982112.2015.968039.

[16] Roustant, O., Padonou, E., Deville, Y., Clément, A., Perrin, G., Giorla, J., and Wynn, H., "Group kernels for Gaussian process metamodels with categorical inputs," *arXiv preprint arXiv:1802.02368*, 2018.

[17] Oliver, M. A., and Webster, R., "Kriging: a method of interpolation for geographical information systems," *International Journal of Geographical Information Systems*, Vol. 4, No. 3, 1990, pp. 313–332. https://doi.org/10.1080/02693799008941549.

[18] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and Analysis of Computer Experiments," *Statistical Science*, Vol. 4, No. 4, 1989, pp. 409–423. https://doi.org/https://www.jstor.org/stable/2245858.

[19] Simpson, T. W., Peplinski, J. D., Koch, P. N., and Allen, J. K., "Metamodels for Computer-based Engineering Design: Survey and recommendations," *Engineering with Computers*, Vol. 17, No. 2, 2001, pp. 129–150. https://doi.org/10.1007/pl00007198.

[20] Aronszajn, N., "Theory of reproducing kernels," *Transactions of the American mathematical society*, Vol. 68, No. 3, 1950, pp. 337–404. https://doi.org/10.2307/1990404.

[21] Steinwart, I., and Christmann, A., *Support vector machines*, Springer Science & Business Media, 2008.

[22] Scholkopf, B., and Smola, A. J., *Learning with kernels: support vector machines, regularization, optimization, and beyond*, MIT press, 2001.

[23] Santner, T. J., Williams, B. J., and Notz, W. I., *The Design and Analysis of Computer Experiments*, Springer New York, 2003. https://doi.org/10.1007%2F978-1-4939-8847-1.

[24] Suits, D. B., "Use of Dummy Variables in Regression Equations," *Journal of the American Statistical Association*, Vol. 52, No. 280, 1957, pp. 548–551. https://doi.org/10.1080/01621459.1957.10501412.

[25] Halstrup, M., "Black-box optimization of mixed discrete-continuous optimization problems," Ph.D. thesis, TU Dortmund, jan 2016.

[26] Hutter, F., "Automated configuration of algorithms for solving hard computational problems," Ph.D. thesis, University of British Columbia, 2009.

[27] Gower, J. C., "A General Coefficient of Similarity and Some of Its Properties," *Biometrics*, Vol. 27, No. 4, 1971, p. 857.

[28] Zhang, Y., Tao, S., Chen, W., and Apley, D. W., "A latent variable approach to Gaussian process modeling with qualitative and quantitative factors," *Technometrics*, 2019, pp. 1–12. https://doi.org/10.1080/00401706.2019.1638834.

[29] Journel, A. G., and Huijbregts, C. J., *Mining geostatistics*, Vol. 600, Academic press London, 1978.

[30] Goovaerts, P., et al., *Geostatistics for natural resources evaluation*, Oxford University Press on Demand, 1997.

[31] Pinheiro, J., and Bates, D. M., "Unconstrained parametrizations for variance-covariance matrices," *Statistics and Computing*, Vol. 6, No. 3, 1996, pp. 289–296. https://doi.org/10.1007/BF00140873.

[32] Qian, P. Z. G., Wu, H., and Wu, C. F. J., "Gaussian Process Models for Computer Experiments With Qualitative and Quantitative Factors," *Technometrics*, Vol. 50, No. 3, 2008, pp. 383–396. https://doi.org/10.1198/004017008000000262.

[33] Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C., "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, Vol. 16, No. 5, 1995, pp. 1190–1208. https://doi.org/10.1137/0916069.

[34] Storn, R., and Price, K., "Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces," *Journal of Global Optimization*, Vol. 11, No. 4, 1997, pp. 341–359.

[35] Hansen, N., "Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms," *Towards a New Evolutionary Computation*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 75–102.

[36] McKay, M. D., Beckman, R., and Conover, W., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, Vol. 21, No. 2, 1979, p. 239. https://doi.org/10.1080/00401706.2000.10485979.

[37] Deng, X., Hung, Y., and Lin, C. D., "Design for computer experiments with qualitative and quantitative factors," *Statistica Sinica*, 2015, pp. 1567–1581.

[38] Forrester, A., Sobester, A., and Keane, A., *Engineering design via surrogate modelling: a practical guide*, John Wiley & Sons, 2008.

[39] Picheny, V., Wagner, T., and Ginsbourger, D., "A benchmark of kriging-based infill criteria for noisy optimization," *Structural and Multidisciplinary Optimization*, Vol. 48, No. 3, 2013, pp. 607–626. https://doi.org/10.1007/s00158-013-0919-4.

[40] Deng, X., Lin, C. D., Liu, K. W., and Rowe, R. K., "Additive Gaussian Process for Computer Models With Qualitative and Quantitative Factors," *Technometrics*, Vol. 59, No. 3, 2017, pp. 283–292. https://doi.org/10.1080/00401706.2016.1211554.

[41] McBride, B. J., and Gordon, S., "Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications. User Manual and Program Description," Tech. rep., NASA, 1996.

[42] Levine, I. N., *Physical chemistry*, McGraw-Hill, 2009.

[43] Bonnans, J.-F., Gilbert, J. C., and Lemarechal, C., *Numerical optimization: theoretical and practical aspects*, Springer Berlin Heidelberg, 2006. https://doi.org/10.1007/978-3-540-35447-5.

[44] Ford, H., and Alexander, J. M., *Advanced mechanics of materials*, E. Horwood, 1977.

[45] MacNeal, R. H., and McCormick, C. W., "The NASTRAN Computer Program for Structural Analysis," *SAE Technical Paper*, SAE International, 1969. https://doi.org/10.1016/0045-7949(71)90021-6.

[46] Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, Vol. 13, 1998, pp. 455–492. https://doi.org/10.1023/A:1008306431147.

[47] Ho, Y.-C., and Pepyne, D. L., "Simple explanation of the no-free-lunch theorem and its implications," *Journal of optimization theory and applications*, Vol. 115, No. 3, 2002, pp. 549–570. https://doi.org/10.1023/A:1021251113462.

[48] Pelamatti, J., Brevault, L., Balesdent, M., Talbi, E.-G., and Guerin, Y., "Efficient global optimization of constrained mixed variable problems," *Journal of Global Optimization*, Vol. 73, No. 3, 2019, pp. 583–613. https://doi.org/10.1007/s10898-018-0715-1.

[49] Matthews, A. G. d. G., van der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J., "GPflow: A Gaussian process library using TensorFlow," *Journal of Machine Learning Research*, Vol. 18, No. 40, 2017, pp. 1–6.

[50] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," , 2015. Software available from tensorflow.org.