Navigating kernel selections in kernel-based methods: The issues and possible solutions

Kehinde S. Oyetunde* and Rhea P. Liem†
The Hong Kong University of Science and Technology, Kowloon, Hong Kong

Kernel-based methods are employed in complex cases where the patterns are perceived to be nonlinear. These methods capture the nonlinearity in data and project such data into spaces where linear methods can be used. Kriging is one of the commonly used kernel-based methods. In kriging, kernel selection is a non-trivial issue. Poor selection of kernels could lead to inaccurate approximations especially in complex problems. Hence, it is important to select an appropriate kernel for specific applications. In this paper, we investigate the intuitions behind the behaviours of four main kernels that are commonly used within the engineering community. We then propose two possible methods to automate kernel selections and complete model training for different tasks. The results from benchmarking the proposed methods with both algebraic and real-world problems are presented. The models trained with the proposed approaches are shown to perform better than other models in most of the benchmark cases. An instance is seen in the significant improvement of the model accuracy on the axial transonic problem with NASA rotor 37 as its datum shape. The overall results are promising and demonstrate the potentials of these methods to automate kernel selections in kernel-based methods such as kriging.

Nomenclature

N_s	=	number of sample points
κ	=	covariance function
R	=	correlation matrix
r	=	cross correlation matrix
θ	=	hyperparameter
N_d	=	problem dimensions
$\mathbf{1}$	=	$N_s \times 1$ vector of one
$\mathrm{m}(\mathbf{x})$	=	prediction mean
$s^2(\mathbf{x})$	=	prediction variance
μ	=	kriging mean trend
$\sigma^2(\boldsymbol{x})$	=	kriging process variance
h_{ij}	=	$\lvert x_i - x_j \rvert$

I. Introduction

Kernel-based methods are a class of machine learning methods that are suited for nonlinear problems, and are sometimes known as *kernel machines*. These methods can detect nonlinear patterns from the data while retaining the computational elegance of matrix algebra [1]. Some of the well-known kernel-based methods include support vector machine (SVM) [2], Kriging [3, 4], kernel principal component analysis (KPCA) [5] and kernel Fisher discriminant analysis [6]. Kernel-based methods have been used in several areas such as geostatistics [7], signal processing [8], bioinformatics [9], data compression [10], information extraction [11] and pattern recognition [12]. In spite of their extensive use, selection of appropriate kernels remains a major bottleneck. The role of the kernel in such methods is to project data onto higher-dimensional spaces where linear methods are more likely to be applicable [13, 14]. In fact, the

*Ph.D. Candidate, Mechanical and Aerospace Department
†Assistant Professor, Mechanical and Aerospace Department, AIAA Member

performance of kernel-based methods is highly dependent on the choice of the kernel function [15]. In this paper, we will focus our investigations mainly on Kriging [16], which is one of the widely used kernel-based method. A detailed theoretical background on kriging will be given in section II.D. Key to the model structures of kriging are the kernels. There are different types of kernels but they all have basically two components, namely the distance which is a measure of the separation of points $|x - x'|$ and the hyperparameter θ. Kriging-based models often rely on the use of predefined kernels in their construction. A multitude of possible kernels exist and each has a number of free hyperparameters whose values also need to be determined to define the model structure. Choosing the appropriate kernel/covariance function for a particular application thus comprises both kernel selection (i.e., by comparing across different kernels) and determining the kernel hyperparameters [17]. The use of unadapted kernels or models can have negative impact on model accuracy [18]. Hence, it becomes a necessity to have Kriging models that are flexible enough to avoid model misspecification. For a particular application, a kriging kernel might be selected from a pool of kernels. Choosing the most likely model or kernel from a group of competing models can be difficult, especially when using black box methods where a thorough understanding of the physics is not obvious [19]. Within the engineering community, i.e., kriging users, the kernel pool is usually limited to the Gaussian, exponential, Matérn 3/2 and Matérn 5/2 [20]. Recently, the cubic kernel has seen more usage because of its compact support [21]. Kernels with compact supports are effective when the dataset is very large because they lead to sparse matrices [22]. For example, Liem *et al.* used the cubic kernel for predicting aerodynamic performance [23]. In this paper, however, we will not focus on problems with large datasets and therefore, detailed discussion of such kernels is outside the scope of this paper.

From literature, many methods have been proposed towards selecting the appropriate kernels for specific applications. These methods are known as *automatic kernel learning methods*. Some of the techniques employed include trial-and-error, greedy search, cross validation and deep learning. More details about the different techniques can be found in section III. With the multitude of challenges faced by researchers with the different automatic kernel learning methods, there has been a shift towards the use of multiple kernel learning (MKL) methods [24–26] in recent times. There are two main reasons towards this shift. Firstly, the need to improve model accuracy beyond what is achievable with models trained using single kernels. Secondly, the desire to retain information gathered from all kernels considered in the learning process. Multiple kernel learning methods can be done by combining predictions from models trained with different kernels. This is known as *ensemble learning* [27–31]. Intrinsically combining kernels during model training is another commonly used technique. This method is known as *composite kernel learning method* [20, 21, 32]. However, multiple kernel learning method is beyond the scope of this paper. Common to both automatic kernel learning and multiple kernel learning methods from literature is the lack of extensive study on the behaviour of the kernels in different context. In this paper, we will focus on understanding the behaviour of different kernels within the kriging framework. Based on the intuitions gained from the preliminary studies, we will propose two methods to assist in a more systematic kernel selection procedure.

The organization of the remainder of this paper is as follows: Section II starts by giving a detailed overview on kernel-based methods and common examples of the method. Section III provides a brief review on methods that have been proposed towards kernel selections. We then present the design and outcome from the study on kernels in section IV. Section V gives an overview of the proposed approaches based on our insights from the preceding section. The overview include both the formulations and algorithm. We then introduce the different test cases to be used for the study in Section VI and present the results and discussions in Section VII. In section VIII, we highlight the conclusions from the studies and briefly discuss potential future directions.

II. Kernel-based methods

One major complexity in modeling real-world systems is nonlinearity [33], where the relationship between the variables of the systems are mostly nonlinear. If the analysis of the data involves linear methods alone, chances are the predictions may be inaccurate [14]. Having established the need of non-linear methods, it is important to mention that even the use of linear methods is adequate in some cases. Linear approaches are usually preferred because they rely solely on the mathematical simplicity and proven efficiency of linear algebra and matrix theory. However, if a system is operating at a wide range of conditions as seen in many real-world cases, the resulting nonlinear dynamic behavior must be addressed with more advanced techniques such as kernel methods. Kernel methods are a group of learning machines which are used for fast recognition of nonlinear patterns in a given dataset [14]. Kernel approaches are based on the idea of pre-processing data by transferring it to higher-dimensional spaces by a novel projection technique. In these spaces, the application of linear methods is more feasible [13]. As a result, kernel approaches are more likely to extract nonlinear patterns from data while maintaining the computational simplicity of matrix algebra [1]. The feature mapping

that is done by kernel methods can be visualized in Figure 1.

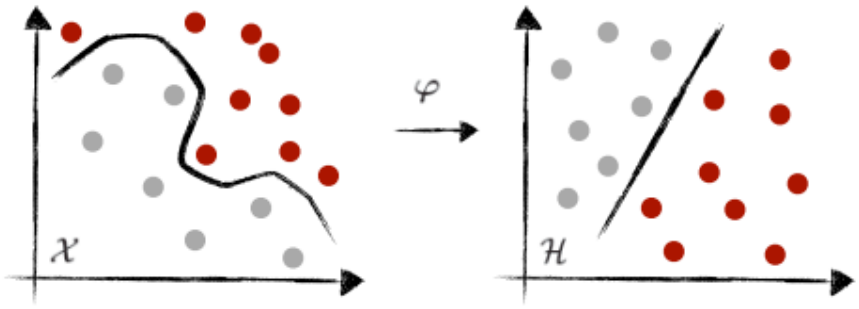


Fig. 1 Illustration of feature space mapping in kernel methods [34]

Kernel-based methods have been used extensively in many applications such as media data analysis [35], process monitoring [14], and remote sensing data analysis [36, 37], to name a few. To be specific, it is used for feature extraction in industrial process monitoring. Common to all the applications is that linear modeling approaches suffer from poor performances. Hence, non-linear methods become more imperative [38].

At the core of these methods are the kernel functions which are used to capture the inherent complexity in real-world systems. The purpose of kernels in kernel-based methods is to map data to a high-dimensional induced feature space by formulating nonlinear extensions of classical linear models. By doing this, the nonlinearities inherent in such data/problem are captured. A kernel function, $\kappa(.,.)$ is defined on $\chi \times \chi$ where all $\boldsymbol{x}, \boldsymbol{x'} \in \chi$ satisfy Equation 1;

$$\kappa\left(\boldsymbol{x}, \boldsymbol{x'}\right) = \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x'}) \rangle, \tag{1}$$

where $\varphi : \chi \mapsto \mathcal{H}$ is a nonlinear map from D_χ-dimensional input space, χ, to the $D_\mathcal{H}$-dimensional reproducing kernel Hilbert space (RKHS) associated with the kernel. The definition given in Equation 1 is the kernel trick, which is crucial in making kernel methods feasible. This is because it implies that the data never have to be represented explicitly in the RKHS as long as the data only appear as inner products in the model formulation. It is mandatory for a kernel to satisfy certain conditions. Key of it all is that a valid kernel must be positive definite and forms a symmetric matrix. In early years, kernels were required to fulfill Mercer's Theorem [39] in order to be valid. In recent time, the requirements can be relaxed so that a kernel is considered valid if and only if it is symmetric and positive semi-definite (PSD). When this is fulfilled, it is guaranteed that there exists a map $\varphi : \chi \mapsto \mathcal{H}$ such that Equation 1 holds and the feature space, $\mathcal{H}$ will have the structure of a RKHS.

Choosing a suitable kernel in kernel-based methods for a particular application is a non-trivial task [17]. The selection of the kernel in non-parametric models such as kriging is a black art [40]. Model misspecification as a result of using unadapted kernels or models can cause a loss of accuracy [18]. In Section III, a brief review on the various works that have been done towards effective kernel selection in kriging is given. In the following subsections, we give a brief overview into common kernel-based methods within the engineering community.

A. Kernel principal component analysis

Principal component analysis (PCA) is a well-known method for obtaining the major modes of variation in a dataset. The method is often used in applications involving dimensionality reduction and data compression. With time, it became a necessity to improve its performance on nonlinear manifolds. As a result, Kernel PCA (KPCA) amongst other methods was developed [1]. Kernel PCA, a generalization of linear PCA to nonlinear manifolds has proven a powerful tool for nonlinear dimensionality reduction, feature extraction, or denoising. Common for all applications is that the goal is to only retain a subset of the principal components. As opposed to other nonlinear feature extraction methods, kernel PCA does not require nonlinear optimization; instead, it relies on the solution of an eigenvalue problem [41]. Similar to linear PCA, the aim of kernel PCA is to project data onto a set of orthonormal bases that maximizes the explained variance. However, this should hold in $\mathcal{H}$ not χ. The bases are determined by the leading eigenvectors of the covariance matrix C in the RKHS. For a given sample points, N_s, C is given by Equation 2,

$$\boldsymbol{C} = \frac{1}{N_s} \sum_{n=1}^{N_s} \tilde{\varphi}(\boldsymbol{x}_n)\tilde{\varphi}(\boldsymbol{x}_n)^T. \tag{2}$$

The magnitude of the i'th eigenvalue, λ_i, measures the amount of variation in the direction of the corresponding eigenvector, ν_i, which is also known as the i-th Principal Component (PC). Analogous to linear PCA, the i-th PC can be found as the normal direction that maximizes the variance of the projection while being orthogonal to all previous PCs. This can be formulated as a quadratic optimization problem:

$$\begin{aligned} \min_{\nu_i \in \mathcal{H}} \quad & \nu_i^T \boldsymbol{C} \nu_i \\ \text{s.t.} \quad & \|\nu_i\|^2 = 1 \\ & \sum_{n=1}^{i-1} \langle \nu_n, \nu_i \rangle^2 = 0 \end{aligned} \tag{3}$$

Since C is a covariance matrix, and hence PSD, the problem is known to be convex, and similar to linear PCA it can be expressed as an eigenvalue equation:

$$\lambda_i \nu_i = \boldsymbol{C} \nu_i, \tag{4}$$

where $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$ are ordered eigenvalues and $\nu_i \in \mathcal{H}$ are the corresponding eigenvectors of $\boldsymbol{C}$.

B. Support vector machines

In the late 1980s, Vapnik and colleagues developed the Support vector machines (SVM) as a learning machine technique [42]. SVM is rooted in statistical learning theory and exploits Vapnik-Chervonenkis (VC) theory and the VC dimension in particular as a measure of a algorithms capacity to learn from data. It is a proven classification algorithm in machine learning that was first proposed by Cortes and Vapnik [43]. It is based on the principle of structural risk minimization, in which the complexity of the metamodel is controlled to avoid overfitting [44, 45]. Even though the method was originally developed to solve binary classification problems, it has since been extended to multi-class problems and regression [46]. According to Domingos, SVMs are the most used kernel approach for classification tasks [47]. However, it is encouraged to try out simpler methods such as k-nearest neighbour first before adopting SVM[47]. SVM defines the optimal decision function as the maximum margin hyperplane that separates the two classes. Any hyperplane in some kernel induced feature space, $\mathcal{H}$, can be formulated as:

$$\{\varphi(\boldsymbol{x} \in \mathcal{H} : \langle \boldsymbol{w}, \varphi(\boldsymbol{x}) \rangle + b = 0\} \boldsymbol{w} \in \mathcal{H}, \boldsymbol{b} \in \mathbb{R}, \tag{5}$$

where **w** is the weight vector and b is the bias term.

C. Radial basis functions

Radial basis functions (RBFs) are linear combinations of basis functions, where each basis function depends on the distance from the prediction point to each training point. The coefficients of the linear combination are obtained by solving a dense linear system.To capture broad patterns, radial variable basis functions are frequently supplemented with polynomial functions [48]. The prediction using RBFs is given by Equation 6,

$$f(\boldsymbol{x}, \boldsymbol{w}) = \sum_{i=1}^{n_p} p_i(\boldsymbol{x}) a_i + \sum_{i=1}^{n_t} \phi_i(\boldsymbol{x}) b_i, \tag{6}$$

where $a_i \in \mathbb{R}$ and $b_i \in \mathbb{R}$ are the coefficients for the i-th polynomial trend p_i (**x**)) and basis functions ϕ_i (**x**) respectively. There are several factors affecting the performance of the RBF interpolation process including kernel function, center distribution, shape parameter and the unknown function [49]. As for the kernel function, there are many choices such as the multiquadric [50], thin plate splines [51] and the Gaussian. In spite of the numerous advantages of RBFs, they have been reported to not been adequate as interpolants based on energy minimization. A specific case is when the data points are significantly spaced [48].

D. Kriging

Kriging is a method originally developed to solve problems in *geostatistics*. After 1989, it became widely used in computational modeling and design applications [52]. Kriging [3] approximates the relationship between the input, $\mathbf{x}$, and the output, $\mathbf{y}$ of a blackbox function, which can be expressed as shown in Equation 7.

$$Y(\mathbf{x}) = \mu(\mathbf{x}) + Z(\mathbf{x}), \tag{7}$$

where $\mu(\mathbf{x})$ is the deterministic function approximating the mean trend of the output and $Z(\mathbf{x})$ is the stochastic process which deviates from this trend [21]. $Z(\mathbf{x})$ has a zero mean and the covariance $\mathrm{Cov}[Z(\mathbf{x}), Z(\mathbf{x}')] = \sigma^2 \mathbf{R}(\mathbf{x}, \mathbf{x}')$. σ^2 is the process variance and $\mathbf{R}(\cdot,\cdot)$ is the correlation matrix. The global model is assumed constant here, as we only consider the *ordinary kriging* model for this work. The selection of kernel for the correlation matrix is critical for kriging construction, as it encodes assumptions about the dependence structure of the function of interests [26]. For higher-dimensional problems, the product correlation rule is typically used to construct the correlation matrix, where the correlation function can be expressed as a product of stationary, one-dimensional correlations. The correlation between the i-th and j-th samples can then be expressed as shown in Equation 8.

$$R_{ij}(\boldsymbol{h}, \boldsymbol{\theta}) = \prod_{k=1}^{N_d} R\left(h_{ij}^{(k)}, \theta^{(k)}\right). \tag{8}$$

The vector of correlation parameters is denoted as $\boldsymbol{\theta} = \left\{\theta^{(k)}\right\}$, $k = 1, \ldots N_d$, where N_d denotes the problem dimension and k is the dimension index. The value $h_{ij}^{(k)}$ is the distance between two points in the k^{th} dimension, $\left|x_i^{(k)} - x_j^{(k)}\right|$. The kernel (or covariance) function is one of the important ingredient of Kriging, as it encodes assumptions about the function to be approximated [26] . It describes the correlation of the responses of design points. For a simple kriging model with a constant trend, the mean and the variance of the prediction at a point $\mathbf{x}$ are expressed as shown in Equations 9 and 10 respectively.

$$m(\mathbf{x}) = \mu + \mathbf{r}(\mathbf{x})^T \mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}\mu) \tag{9}$$

$$s^2(\mathbf{x}) = \hat{\sigma}^2 \left(1 - \mathbf{r}(\mathbf{x})^T \mathbf{R}^{-1} \mathbf{r}(\mathbf{x})\right) \tag{10}$$

To construct a kriging model, it is essential to choose the type of kernel function that models best the correlation between the design points [31]. In this article, four kernels (Gaussian, Exponential, Matérn 3/2 and Matérn 5/2 kernels) are considered for performance comparison and also to develop the proposed approaches. The mathematical expression of the kernel functions are shown in Equations 11, 12, 13 and 14

- Gaussian

$$\kappa(h, \theta) = \exp -\frac{1}{2}\left(\frac{h}{\theta}\right)^2 \tag{11}$$

- Exponential

$$\kappa(h, \theta) = \exp -\frac{1}{2}\left(\frac{h}{\theta}\right) \tag{12}$$

- Matérn 3/2

$$\kappa(h, \theta) = \left(1 + \frac{\sqrt{3}|h|}{\theta}\right) exp\left(-\frac{\sqrt{3}|h|}{\theta}\right) \tag{13}$$

- Matérn 5/2

$$\kappa(h, \theta) = \left(1 + \frac{\sqrt{5}|h|}{\theta} + \frac{5h^2}{3\theta^2}\right) exp\left(-\frac{\sqrt{5}|h|}{\theta}\right) \tag{14}$$

where h is a function of distance $|x - x'|$ and $\boldsymbol{\theta}$ is the length scale

1. Hyperparameter estimation

Hyperparameters refer to a number of parameters of the model structures, such as the kernel parameters, which determine how each sample affects the function approximation. In the context of kriging, the hyperparameters are also known as the lengthscales, which measure the rate of decay of correlation [53]. In kriging, the correlation function can either be isotropic or anisotropic. In isotropic correlation functions, a single θ value is used for different dimensions [17]. This helps simplify the hyperparameter optimization as it reduces to a one-dimensional problem. Anisotropic correlation functions are advised when each variable has a distinct physical meaning [54]. In such cases, each variable dimension has a unique θ value. This approach gives more flexibility in modeling at the expense of more complexity [55, 56].

Cross-validation and maximum likelihood estimation (MLE) methods are two main methods to estimate these hyperparameters [17, 20, 57, 58]. The MLE method is advised because of its computational efficiency [59]. The cross-validation method, on the other hand, tends to be less biased which makes it more convenient in some applications [60]. In this work, we will only consider the MLE method for its computational efficiency [59]. The likelihood function can be expressed as shown in Equation 15.

$$L(\boldsymbol{\theta}) = \frac{1}{(2\pi\sigma^2)^{n/2}|\mathbf{R}(\boldsymbol{\theta})|^{1/2}} \exp\left[-\frac{(y-\mathbf{1}\mu)^T\mathbf{R}(\boldsymbol{\theta})^{-1}(\mathbf{y}-\mathbf{1}\mu)}{2\sigma^2}\right] \tag{15}$$

The maximum likelihood estimates of regression coefficients and variance can be obtained by taking the derivatives of the natural logarithm of the likelihood function and setting the result to zero.

$$\mu(\boldsymbol{\theta}) = (\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1})^{-1}(\mathbf{1}^T\mathbf{R}^{-1}\mathbf{y}) \tag{16}$$

and,

$$\hat{\sigma}^2 = \frac{1}{N_s}(\mathbf{y}-\mathbf{1}\mu)^T\mathbf{R}(\boldsymbol{\theta})^{-1}(\mathbf{y}-\mathbf{1}\mu) \tag{17}$$

Substituting Equations 16 and 17 into the natural logarithm of the likelihood function of Equation 15, the concentrated log likelihood function is obtained, as shown in Equation 18.

$$LL(\boldsymbol{\theta}) = -\frac{N_s}{2}ln(2\pi) - \frac{N_s}{2}ln\hat{\sigma} - \frac{1}{2}ln|\boldsymbol{R}|, \tag{18}$$

which depends on the set of hyperparameters $\boldsymbol{\theta}$. The optimum value of $\boldsymbol{\theta}$ is obtained by maximizing the function.

2. Hyperparameter optimization

Hyperparameter optimization [61] is the process involved in finding the optimum hyperparameters within a defined search space. In optimization, it is important to understand the problem and be able to choose the appropriate optimization technique. For detailed information on optimization techniques, readers are referred to the book on engineering design optimization by Martins and Ning [62]. In the context of kriging, the hyperparameters are chosen by selecting the values that optimizes a defined loss function, e.g., cross validation error or maximum likelihood estimate. To help with the selection of optimum hyperparameters for kriging, several optimization algorithms have been used. In kriging and other surrogate modeling technique, it is mandatory for the hyperparameters to be non-negative hence the need to ensure the variables of the optimization algorithm are bounded to be greater than zero at any point. The implication is that, with bounded variables, the points can leave the domain after either the reflection or the expansion operation. Constraints and limits on variables are usually taken into account through adaptive penalization and projection, respectively.

In this paper, we consider using Nelder-Mead algorithm [63, 64] and Optuna [65] to optimize the hyperparameters. The Nelder-Mead method, also known as downhill simplex method, is a deterministic and local direct search algorithm first proposed by John Nelder and Roger Mead for non-linear function optimization problems [66]. The algorithm is capable of adjusting itself to the landscape of the objective function, thereby reducing the search space to converge easily. However, it is very sensitive to the selection of initial points. There is the tendency of the search converging to a local optimum. Furthermore, there is no guarantee that the algorithm converges to stationary points [67]. For Nelder-Mead algorithm, we use the `constNMPy` package [64] because it accounts for variable bounds and nonlinear inequality constraints.

Optuna [65, 68] is a software framework for automatic hyperparameter optimization that was created with machine learning in mind. The framework has a lightweight, versatile, and platform agnostic architecture. This makes it able to handle a wide variety of tasks without issues. It also adopt state-of-the-art algorithms for sampling hyperparameters and efficiently pruning unpromising trials. We select this framework because of the capability of running many trials in parallel with ease.

III. Review on kernel selection in kriging-based applications

The choice of kernel is crucial to the success of kernel-based methods [15]. Over the years, many approaches have been proposed towards this selection. Initially, researchers and engineers use their intuition and experience to select the kernel during model development. Over time, this approach has proved to be less transferable, especially to other problems. Towards finding out the best kernels for different problem sets, researchers tend to imbibe a trial-and-error approach towards kernel selection. Viana *et al.* [69] proposed a cross-validation approach in selecting the best kernel in surrogate modeling applications. In the method, all available kernels are used for model training and prediction, then the best model is selected. The issue with this approach is that all methods need to first be trained and used for prediction, before the best can be selected. One clear limitation of this approach is that the selection process is sensitive to the chosen test set. Biswas *et al.* [70] in their paper developed a mean error-based method to select the optimal correlation function for Kriging-based applications. In the method, different kernels are used during model training and that which minimizes the overall mean error ϵ is selected. While this method could help select a good performing kernel, it does not scale well with large sample sizes and problem dimension. This is because a leave-one-out cross validation (LOOCV) is employed in the computation of the overall mean error from the different kernels. Another common approach is the greedy search [40, 71]. A greedy search is performed to identify the kernel offering the best representation of the data. Usually, the quality of the kernel is quantified by the Bayesian model evidence, which can be computed by optimizing the marginal likelihood over the defined hyperparameters. However, since the integration is challenging to compute, each kernel's suitability is instead typically determined via a proxy for the model evidence, such as the Akaike Information Criterion (AIC) [72] or Bayesian Information Criterion (BIC) [73, 74]. For a more systematic Bayesian approach to kernel design, instead of selecting a single kernel, one ought to consider multiple candidates. In other words, it is desirable to marginalise over the space of kernels, not only over the space of functions for a single kernel. A conventional kernel search makes three key assumptions. First, that contributions from all but the single chosen kernel κ^* can be neglected; second, that contributions from all but the maximum likelihood hyperparameters θ^* can be neglected; third, that the proxy for the model evidence is a reliable one. There are several regimes, for example where the data are sparse or noisy, where all three of these assumptions do not hold [75]. Ali and Smith-Miles [76] adopted a meta-learning approach to automatic kernel selection for support vector machines. In the method proposed, decision tree algorithm was used to learn the relationship between dataset characteristics and the name of the best performing kernel. The data characteristics considered include the normal, gamma and Rayleigh probability distribution function estimates of the input. From the study, the authors recommended the use of the RBF kernel due to its good performance across board. However, there is no clear explanation on how the chosen statistical test aimed at capturing the data characteristics truly affect kernel selection. Also, the relationship between the input $\mathbf{x}$ and output, $\mathbf{y}$, is not considered because the data characteristics were only extracted from the input $\mathbf{x}$. While this might be sufficient for classification problems, it might not be applicable to regression problems. Wilson and Adams [77] introduced simple closed form kernels that could be used with Gaussian processes to discover patterns and enable extrapolation. These kernels are created by using a Gaussian mixture to model a spectral density.

Abdessalem *et al.* proposed the use of approximate Bayesian computation (ABC) algorithm to help with kernel selection and parameter estimation for machine learning applications [19]. Key advantages of the method are that it is independent of the problem dimension and it bypasses the evaluation of the likelihood function. Major drawbacks are the possibility of overfitting in high-dimensional parameter spaces and the wider application domain of ABC algorithm itself worsens the challenges of parameter estimation and model selection. Salem and Tomaso (2018) introduced a universal criterion based on internal accuracy, predictive performance and a roughness penalty to select best models [78]. This approach guarantees an improvement in accuracy and a limitation on overfitting. However, it is exhaustive, which limits its extensive adoption. Fischer developed a framework for model selection to rank kernels for Gaussian process regression using principle of approximation set coding (ASC) [79]. Approximation set coding (ASC) determines the best trade-off between a model's expressiveness and its inference repeatability [80]. Deep learning has also been explored in the selection of kernels for kernel-based methods such as Gaussian process. Salakhutdinov and Hinton in 2007 proposed using deep belief net (DBN) to learn covariance kernels for Gaussian processes [81]. The authors reported that using backpropagation through DBN to discriminately fine-tune the covariance kernel improved the performance on both regression and classification problems. Simpson *et al.* [75] proposed a method to identify the kernel for specific applications using transformers. In the proposed method, kernel recommendation is performed by a decoder with access to a large vocabulary of primitive kernels and products of primitive kernels. The decoder maps an encoded representation of a dataset to a kernel that can be used to model it. By training Kernel Identification Through Transformers (KITT) with a sufficiently rich vocabulary of kernels, it can predict suitable kernels for a diverse array of real datasets. Even though the authors claimed that kernel identification was done within 0.1 seconds, one limitation of

KITT is that it relied on the use of a supercomputer. Ideally, kernel selection and model parameter estimation (especially within the kernel-based method framework) should not rely on high-performance computing, as it can limit the method applicability.

IV. Kernel Studies

In this section, we aim to gain more intuition towards kernel selections in kernel-based methods such as Kriging by investigating the behaviour of kernels in different scenarios. In particular, we study the effect of problem profiles and sample sizes on kernel selection. We also study the sensitivity of different kernels to hyperparameter values. We achieve this by varying the hyperparameter value over an extensive range and evaluating the effect on the likelihood estimate in both one and two dimensional algebraic problems.

A. Effect of problem profiles on kernel selections

In this experiment, we benchmark the performance of the four kernels commonly used within the surrogate modeling community. The benchmarks are done using the Branin, Haupt and robot arm functions, which have varying characteristics in their function profiles. For each of the synthetic problems, we draw 40 sample points within their individual input domains with Halton sampling sequence [82]. Figure 2 shows the varying accuracy levels (expressed in terms of normalized root mean squared errors or RMSE) obtained by using different kernel functions for all benchmark cases.

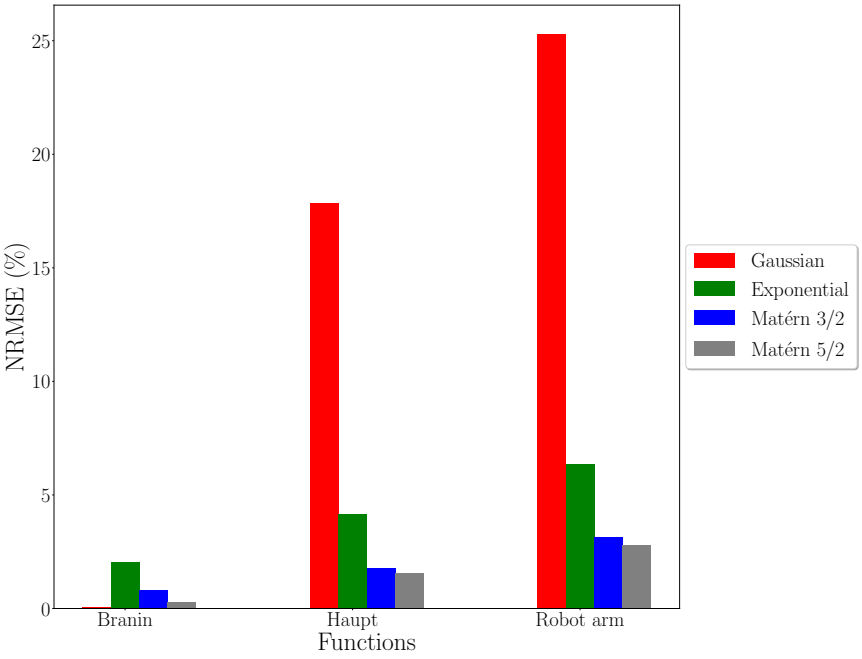


Fig. 2 Choice of kernel can be problem dependent

From Figure 2, the four kernels are observed to model the Branin function well, with the highest error measurement less than 2.5%, which is notably lower than in other cases. Note that among the three cases, the Branin function has the smoothest profile. For this function, the worst performance is recorded with the exponential kernel and the Gaussian kernel performing best. The Matérn 5/2 kernel is observed to be the second best performer, followed closely by the Matérn 3/2 kernel.

For the Haupt function, both Matérn kernels performed best and also have similar model accuracy. This is shown by their NRMSE values which is less than 2%. In this case, the Gaussian kernel is seen to have the worst model performance with its modeling error peaking approximately 17.5%. The exponential kernel performance is observed to have a NRMSE of approximately 4.75% making it the next best performer following the two Matérn kernels.

The performance benchmark in the case of the robot arm problem is similar to the Haupt function. This is because

the Matérn kernels both have the best model performance, followed by the exponential kernel. When the performance of the kernels here is compared to that on Haupt, poorer model accuracy is observed all across with the Gaussian kernel having the highest noticeable change. The NRMSE of the model trained with the Gaussian kernel here is approximately 25% which reflects the unsuitability of the kernel for this particular case. An explanation for the poorer performance observed with the Gaussian kernel might be the shape of the function's response surface as shown in Figure 18b in the Appendix.

Overall, the Gaussian kernel is seen to perform well with Branin function and worst in the other two functions. Hence, generalizing the performance of the Gaussian kernel based on its good fit with a problem might lead to the development of very poor models on other problems.

B. Effect of sample size on kernel selections

In this second experiment, we use the Haupt function to benchmark the performance of the kernels with different sample sizes. For model training, we vary the sample size from 20 to 40 with an increment of 5. For each sample size, Halton sequence is used to draw the samples within the defined input space of the function. Thereafter, the model is trained using the four kernels and then validated on 200 test points drawn randomly. In Figure 3, the performance

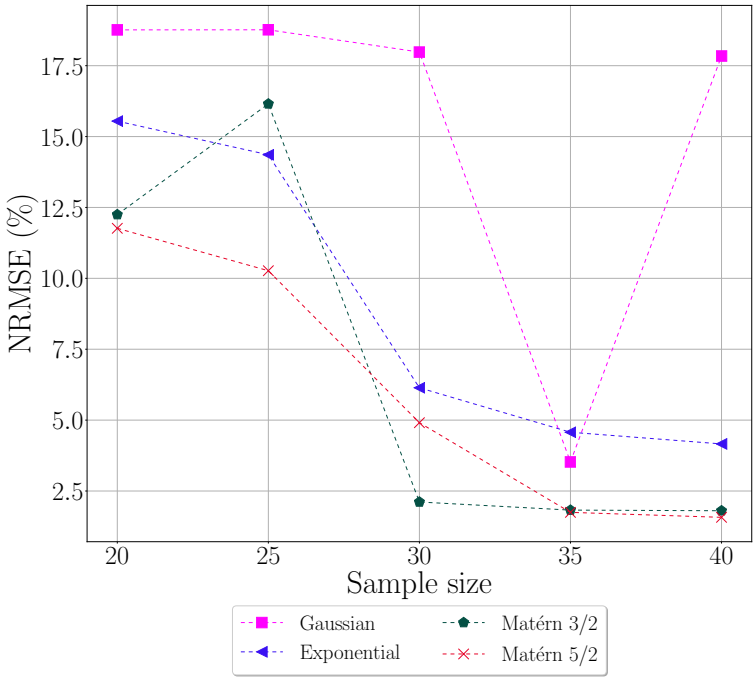


Fig. 3 Choice of kernel can be dependent on sample size

of the Gaussian kernel is seen to improve gradually as more samples are added. With more sample points added after 30 sample points, a great improvement is recorded with the error measurement reducing to as low as 3.75%. However, as more points are added to the sample size of 35, a plunge in performance is observed. Similar worsening in model performance is observed when the training point is increased from 20 to 25 for the Matérn 3/2 kernel. After that point, more addition of sample points led to a significant improvement in the model accuracy. After 30 training points, more addition of points did have little improvement on model performance. The performance of the exponential and Matérn 5/2 are similar with increase in sample points. A significant increase in performance is recorded as more points are added from 20 points up to 35 sample points. Thereafter, a minimum improvement is observed. Overall, the performance of a kernel might improve or worsen with the increase in sample size as shown in Figure 3. Hence, selecting a kernel based on its good performance with a specific sample size does not guarantee that the performance will still be good with a change in the sample size.

C. Kernel sensitivity to hyperparameter values

The choice of hyperparameter is important in the construction of a model and it remains a key constituent of the kernel function. Kriging is known to be sensitive to the choice of hyperparameters. Wrong choice of hyperparameter could lead to huge misspecification. Misspecification here refers to the poor performance of models. Usually, the hyperparameter is estimated using methods discussed in subsection II.D.1 but is it always important to spend lot of computation budget training a model? For a Kriging model, the main hyperparameters affecting the model are the length scales. These parameters can have an impact on the eventual accuracy and applicability of the method. The selection of a suitable range for the lengthscale of kriging models is still a subject of research in the surrogate modeling community [83]. Therefore, in this subsection, a methodical approach is undertaken to quantify the sensitivity of different kernels with respect to these hyperparameters using the algebraic functions. To be specific, we examine the influence of the hyperparameter on different functions with varying function profiles using the four kernels discussed in subsection II.D.

To investigate the patterns in complex problems, we consider using algebraic problems with profiles having distinct characteristics. However, we limit the test cases to only one and two-dimensional problems for ease of interpretation. For the one dimensional profiles, we use six (6) algebraic functions with varying characteristics. The functions consist of algebraic problems with contrasting, constrained and a symmetric-constrained profiles. For ease, we label the functions as Function 1, 2 and 3 respectively. More information about these three functions are given in Appendix VIII.I, VIII.J and VIII.K for Functions 1, 2 and 3 respectively. The cantilever beam, tensor product hyperbolic tangent (TPHT) and robot arm functions are also used in the one-dimensional study. The cantilever beam problem is known to have a linear profile which is assumed to be smooth. More information about the cantilever beam problem is given in Appendix VIII.D. The TPHT and robot arm functions, on the other hand, are known to be non-linear, with the robot arm function exhibiting vertical symmetry. More information about other functions used in this study is provided in the appendix.

We study the effects of the hyperparameter values on the negative log-likelihood estimate (NLL). This is obtained by negating the value estimated from equation 18. The study also includes the effect of the hyperparameter values on the model performance in 1D. The normalized root mean square (NRMSE) is used to assess this performance.

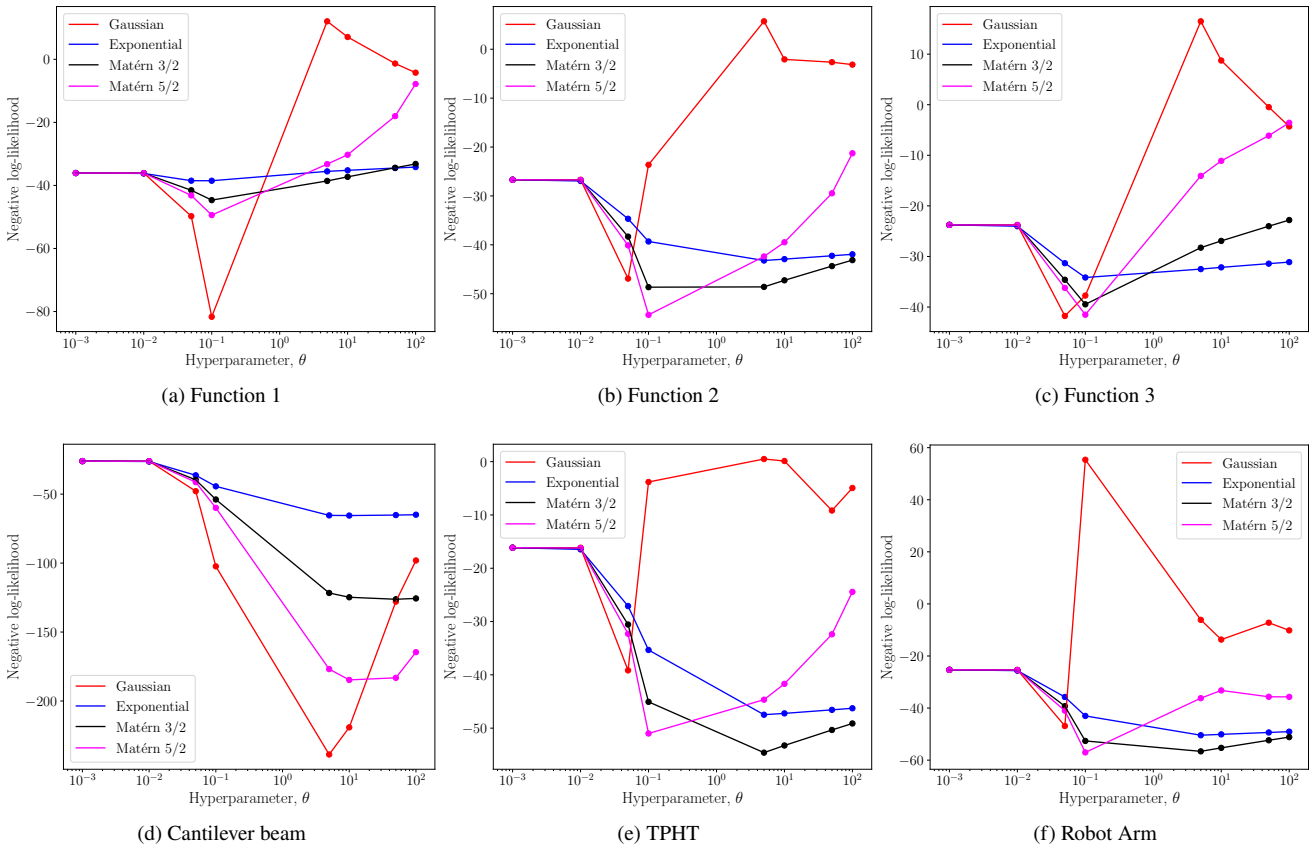
1. Effect of hyperparameter value on the likelihood estimate in one-dimensional problems

For the six 1-D algebraic problems introduced, we inspect the response of the negative log-likelihood estimate to changes in the hyperparameter values. In each case, we define an extensive input space $\theta \in [10^{-4} - 10^{2}]$ and draw 25 sample point for model development using Halton sequence [82]. With the drawn sample points, we obtain their corresponding response $\mathbf{y}$ by evaluating the function. After this, we build the correlation matrix using the different kernels and then evaluate the negative of equation 18. We present the outcome of the experiment in Figure 4.

From Figure 4d, the NLL ranged between approximately -70 and -250. The kernels tend to have lesser NLL with higher hyperparameter values, with the exception of the Gaussian kernel. For the kernels, a sort of convergence is recorded at some point and that remained for a large hyperparameter range. As the hyperparameter value of the Gaussian kernel got to the optimum, further increase in hyperparameter value only worsened the model. Similar behaviour is observed with the kernels in with the Function 1, Function 2, Function 3, TPHT and Robot arm plots as shown in Figures 4a, 4b, 4c, 4e and 4f respectively. However, the Matérn 5/2 is also observed to have similar behaviour as the Gaussian kernel. In some cases as shown in Figure 4c, the estimated NLL from the Matérn 3/2 worsened with increased hyperparameter value after the optimum mark. We expect that the negative log-likelihood estimate to be a positive value. However, this might not always be the case. We observe that for some cases as seen in Figures 4c and 4f, the Gaussian kernel has positive NLL within certain hyperparameter range. Hence, care must be taken in defining the hyperparameter range for a given problem.

2. Effect of hyperparameter on model performance for different function profiles

Having studied the response of the negative log-likelihood to changes in hyperparameter value in 1-D, we attempt to see the influence on model performance. For the six 1-D algebraic problems introduced, we inspect the response of the model performance estimate to changes in the hyperparameter values. In each case, we define an extensive input space $\theta \in [10^{-4} - 10^{2}]$ and draw 25 sample point for model development using Halton sequence [82]. With the drawn sample points, we obtain their corresponding response $\mathbf{y}$ by evaluating the function. After this, we build the correlation matrix using the different kernels and then compute their respective model parameters using equations 16 and 17. We then draw 100 validation points at random and evaluated their respective functional responses. We compute the

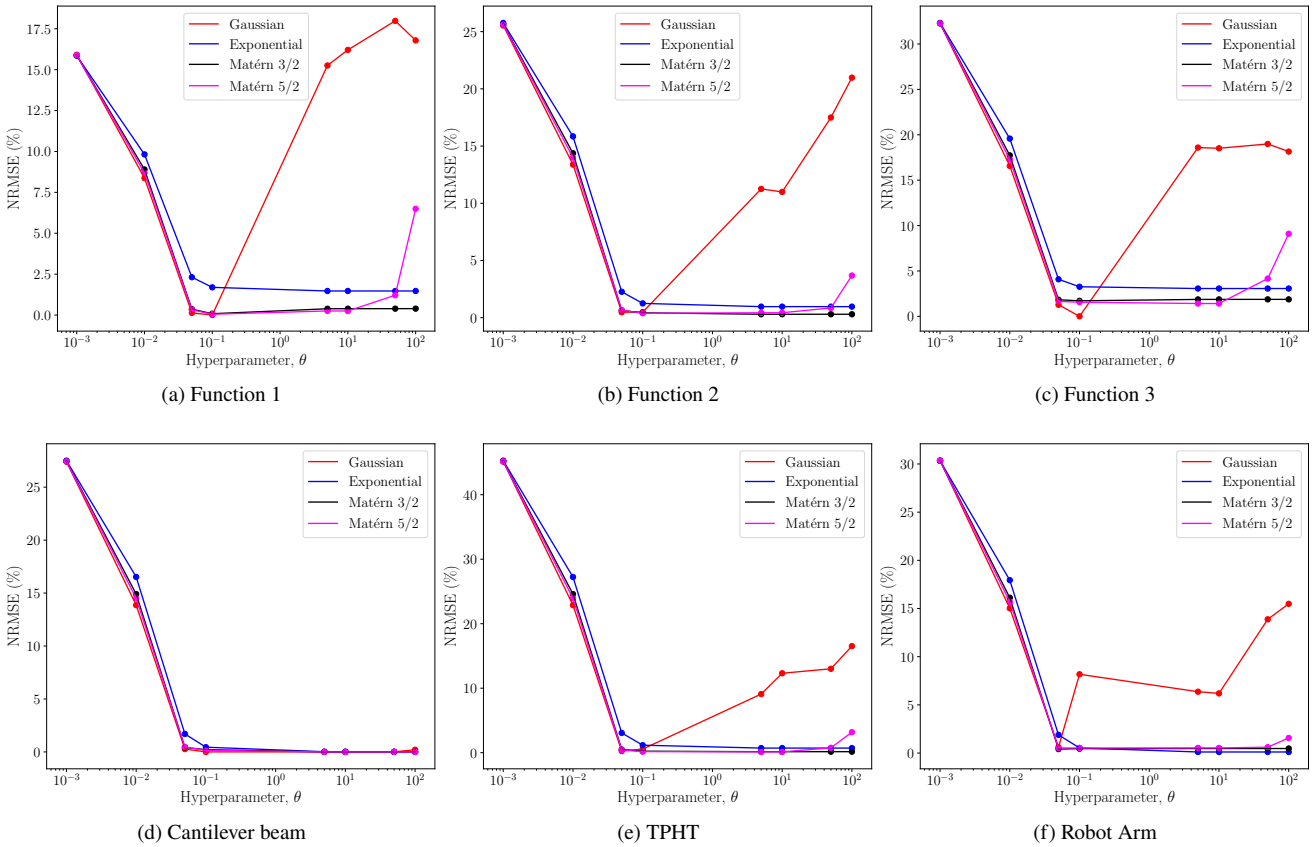


(a) Function 1 (b) Function 2 (c) Function 3

(d) Cantilever beam (e) TPHT (f) Robot Arm

Fig. 4 Effect of hyperparameter values on likelihood estimates

cross-correlation matrix and predicted the output of the validation points using equation 9. After obtaining these values, we estimate the model performance using the normalised root mean square (NRMSE). We present the outcome of this experiment in Figure 5. As expected, all kernels including the Gaussian kernel in the cantilever beam problem



(a) Function 1 (b) Function 2 (c) Function 3

(d) Cantilever beam (e) TPHT (f) Robot Arm

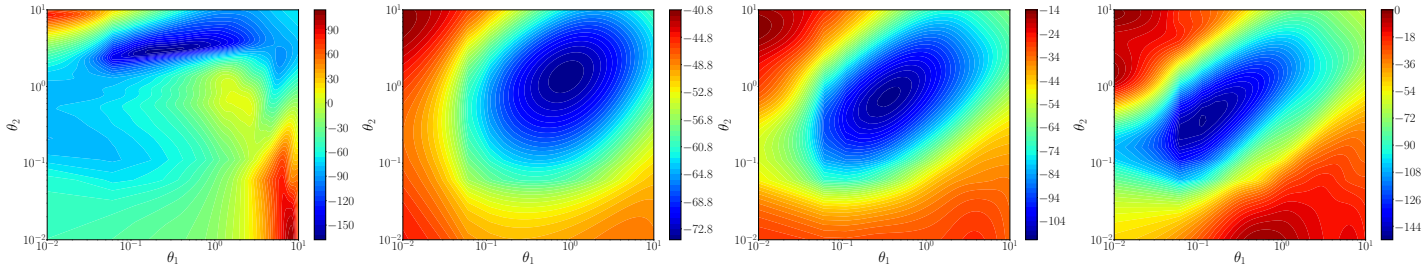
Fig. 5 Effect of hyperparameter values on model performance

performed well regardless of the hyperparameter value as shown in Figure 5d. However, a contrasting performance is observed for the Gaussian kernel. The model worsen when the hyperparameter values deviates from the optimal.

3. Effect of hyperparameter value on the likelihood estimate in two-dimensional problems

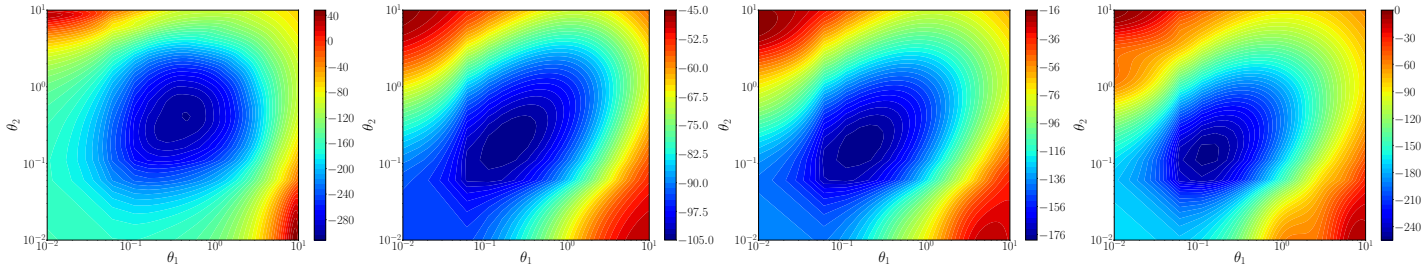
In Subsection IV.C.1, we studied the effect of hyperparameter value on the likelihood estimate using one-dimensional functions. Here, we attempt to extend the study to two-dimensional problems with the aim of gaining more insights into the behaviour of kernels in more dimensional problems. We use eight 2-D problems for the study to inspect the response of the negative log-likelihood estimate to changes in the hyperparameter values. In each case, we define an extensive input space $\theta \in [10^{-2} - 10^{1}]$ and draw 40 sample point for model development using Halton sequence [82]. With the drawn sample points, we obtain their corresponding response $\mathbf{y}$ by evaluating the function. After this, we build the correlation matrix using the different kernels and then evaluate the negative of equation 18. We present the outcome of the experiment using contour plots. On each plot, θ_1 and θ_2 are plotted on the $x-$ and $y-$ axes respectively while the colour bar stands for the negative log-likelihood estimate (NLL).

Besides the issues discussed so far, Figures 6- 13 show that the selection of a starting point will also affect the convergence to the optimal solution. This phenomenon may not be obvious for the Branin and cantilever function as shown in Figures 6 and 7 respectively. In their case, the solution space is approximately convex. However, as we look at subsequent problems, the Gaussian kernel and sometimes the Matérn kernel show a deviation from convex behaviour. In those cases, we see a definite separation of the solution space into multiple (usually two) distinct regions of asymmetric optimum behaviour. We clearly see that they are now local minima regions separated by non-optimum



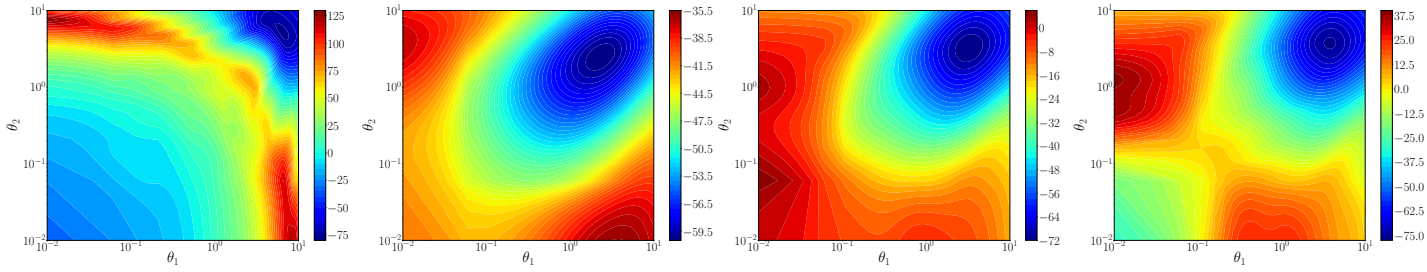
(a) Gaussian (b) Exponential (c) Matérn 3/2 (d) Matérn 5/2

Fig. 6 Effect of hyperparameter values on the likelihood estimate of Branin function



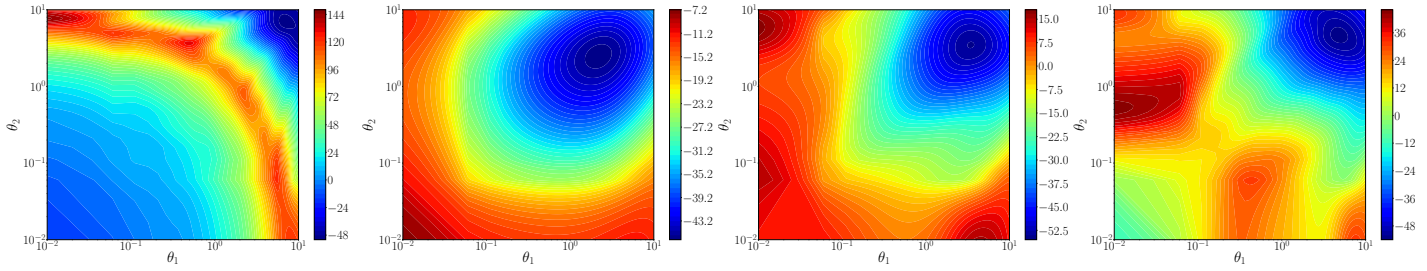
(a) Gaussian (b) Exponential (c) Matérn 3/2 (d) Matérn 5/2

Fig. 7 Effect of hyperparameter values on the likelihood estimate of cantilever beam function



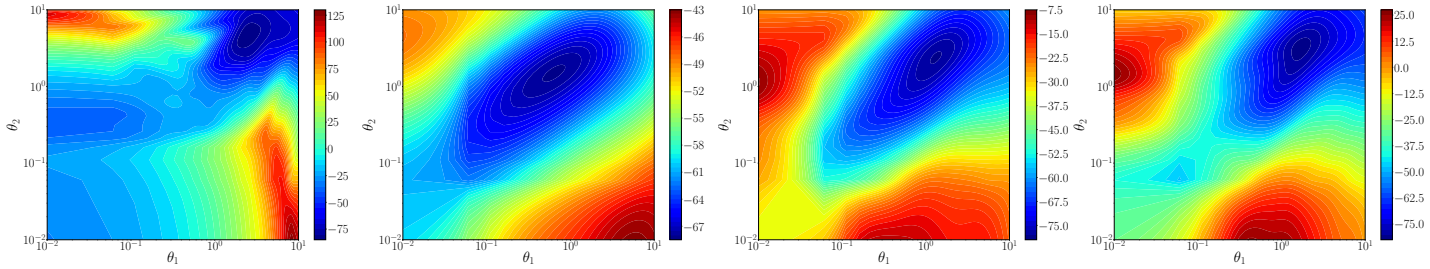
(a) Gaussian (b) Exponential (c) Matérn 3/2 (d) Matérn 5/2

Fig. 8 Effect of hyperparameter values on the likelihood estimate of robot arm function



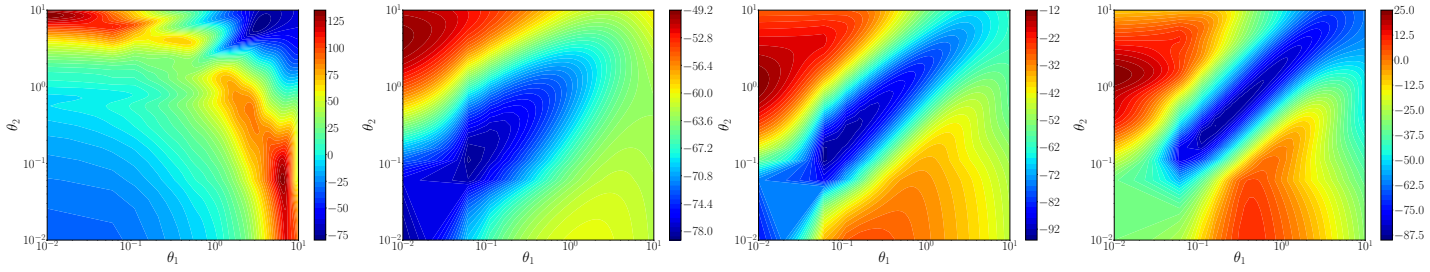
(a) Gaussian (b) Exponential (c) Matérn 3/2 (d) Matérn 5/2

Fig. 9 Effect of hyperparameter values on the likelihood estimate of TPHT function



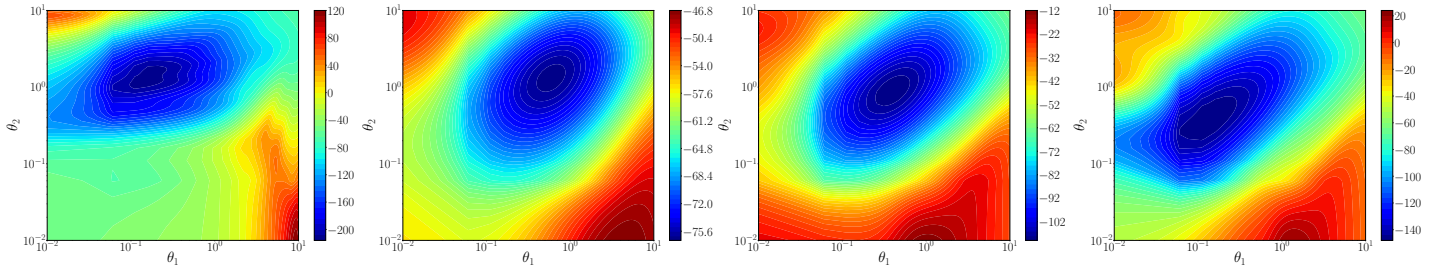
(a) Gaussian (b) Exponential (c) Matérn 3/2 (d) Matérn 5/2

Fig. 10 Effect of hyperparameter values on the likelihood estimate of Hosaki function



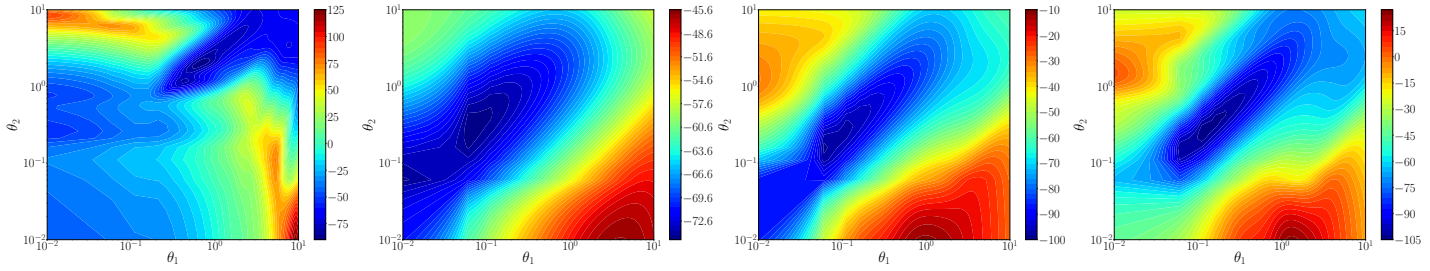
(a) Gaussian (b) Exponential (c) Matérn 3/2 (d) Matérn 5/2

Fig. 11 Effect of hyperparameter values on the likelihood estimate of Haupt function



(a) Gaussian (b) Exponential (c) Matérn 3/2 (d) Matérn 5/2

Fig. 12 Effect of hyperparameter values on the likelihood estimate of Rosenbrock function



(a) Gaussian (b) Exponential (c) Matérn 3/2 (d) Matérn 5/2

Fig. 13 Effect of hyperparameter values on the likelihood estimate of camel back function

ridges. Depending on which side of the ridge we select our initial point, our solution converges to the nearest minimum. It thus becomes obvious that if our starting point is on the side of the ridge closer to a local minimum, our eventual solution will be sub-optimum.

Overall, choosing the appropriate kernel for a particular task can be challenging. From the studies above, many factors could affect the choice of kernel. The performance of a model trained with a specific kernel could be dependent on a problem. A particular kernel that performs well on a task, might give poor performance when used on another task. To make matters worse, the choice of kernel can also be dependent on the sample size of a problem as shown in Figure 3. The consequence of this is that care must be taken when kernel-based methods are used in applications where more sample points are continuously added. Perhaps, the use of adaptive sampling techniques [84–87] to add more sample points intelligently might guarantee improved model performance. However, this could be computationally expensive as it usually involves building a surrogate model and optimizing a defined acquisition function for each new point added. We have also been able to establish the increased sensitivity of some kernels especially the Gaussian kernel to hyperparameter values.

V. Proposed Methods

The observations and results presented in Section IV suggest that the Gaussian kernel tends to be more sensitive to the choice of the hyperparameter. Also, it is observed that sensitivity of the kernel increases with the increasing complexity of the problem. A direct consequence of this behaviour of the Gaussian kernel is the need to employ an optimization algorithm to obtain the hyperparameter values that optimizes the likelihood function, even in one-dimensional problems. As the complexity of the problem to be modeled increases, using simple optimizers such as the pattern search might lead to sub-optimal results. A clue is seen in the negative log-likelihood plots of the robot arm, TPHT and Haupt and two-dimensional problems in Figures 8a, 9a and 11a, respectively. With the use of a pattern search or other commonly used optimization algorithms, the optimal hyperparameters are sensitive to the selection of initial points. In spite of this limitation, we cannot neglect the use of the Gaussian kernel even in real-world problems for two key reasons. The use of Gaussian kernel remains essential in many applications. For example, the Kriging with partial least squares (KPLS) [88–90] used to efficiently model high-dimensional problems without loss in accuracy is known to be best applied with the Gaussian and exponential kernels. An improvement of the KPLS model, the KPLSK model [91] is declared by the author to be best suited with only the Gaussian kernel. Secondly, expressly choosing other kernels over the Gaussian kernel might not lead to any improvement in the performance of the model. The Matérn 5/2 kernel is observed to have similar complications in its likelihood profile in complex cases like the Robot arm and TPHT functions as shown in Figures 8d and 9d respectively. The complication has been shown to limit the general model capability of the kernel in such cases. Given the reasons above, efficient methods to improve the stability and performance of the Gaussian kernel in complex cases are desirable.

With respect to improving the performance of the Gaussian kernel or totally avoiding it if it cannot be improved in poor cases, we propose two methods to automate kernel selection within the kriging framework. The first proposed method considers using the general Matérn kernel in equation 19 as it forms the basis of the commonly used kernels in kriging within the engineering community. We base our proposition on the common knowledge that four major kernels used within the engineering community can be traced to the general Matérn equation. These kernels have varying values of ν, as summarized in Table 1. The ν-parameter is optimized simultaneously with the model hyperparameters. To ensure that the method is a efficient as possible, the upper bound of ν is capped at 3.0. In cases where the absolute difference between the optimal ν and the upper bound ν_{ub} are less that a defined value ($\epsilon < 0.05$), the Gaussian kernel is used for model training. Also to ensure that the model training time is minimal, the optimal hyperparameter from the first stage is used for initialization with the Gaussian kernel. This proposed method is summarized in Algorithm 1 and the model training is represented in a flowchart as shown in Figure 14.

In some machine learning algorithms, hyperparameters to be optimized are usually a mix of continuous, discrete and categorical variables. In the context of kriging, the hyperparameters are usually continuous and optimization of such parameters are done within a defined search space. We propose a method that introduces the kernel as a variable to be optimized within the kriging framework. This way, both kernel selection and model training are carried out simultaneously. To do this, a list of kernels are provided as categories and the defined search space is provided for the continuous hyperparameter values. Commonly used optimization algorithms such as Nelder Mead or Hooke-Jeeves [92] are incapable of handling the optimization of categorical variables. In this method, the optuna optimization framework will be used to solve the optimization problem. With the chosen framework, we aim to achieve non-dependence of the kriging model performance on the initial hyperparameter before optimization, appropriate kernel selection and

improvement of model performance. In our approach, t unique combinations of kernel and hyperparameter values are generated. t is the total number of trials to run in parallel and p is the number of kernel categories available. In this paper, we use $t = 20$ because we only use low dimensional test cases. The value can be adjusted as desired. However, we recommend that to achieve faster model training, a reasonably low value should be used. We summarize the algorithm used by the method for model training and prediction in Algorithm 2.

$$\kappa(h, \theta, \nu) = \frac{1}{2^{\nu-1}\Gamma(\nu)} \left(2\sqrt{\nu}\frac{|h|}{\theta}\right)^{\nu} K_{\nu}\left(2\sqrt{\nu}\frac{|h|}{\theta}\right) \tag{19}$$

where $\nu \geq 1/2$ is the shape parameter, Γ is the Gamma function, and K_{ν} is the modified Bessel function of the second kind.

Table 1 ν for different kernels

ν	Kernel
$\frac{1}{2}$	Exponential
$\frac{3}{2}$	Matérn 3/2
$\frac{5}{2}$	Matérn 5/2
∞	Gaussian

$$\begin{aligned} \max_{\nu,\boldsymbol{\theta}} \quad & -\frac{N_s}{2} ln(2\pi) - \frac{N_s}{2} ln\hat{\sigma}(\nu, \boldsymbol{\theta}) - \frac{1}{2} ln|R(\nu, \boldsymbol{\theta})| \\ \text{s.t.} \quad & \frac{1}{2} \leq \nu \leq \frac{6}{2} \\ & \theta^{(d)} \geq 0 \end{aligned} \tag{20}$$

Algorithm 1 Optimal-ν method

Input: $\mathbf{x}$, $\mathbf{y}$, $\mathbf{X}$
Output: $\boldsymbol{\theta}^f$, ν, $\hat{\mathbf{y}}$

Initialize $\boldsymbol{\theta}$, ν
Assemble $\boldsymbol{R}(\boldsymbol{\theta}) = \prod_{d=1}^{n} \kappa\left(\theta^{(d)}, \nu\right)$
Obtain ν^{opt}, $\boldsymbol{\theta}^f$ by maximizing the likelihood estimate (equation 20) using constrained Nelder-mead algorithm.
if $\|\nu^f$ - $\nu_{ub}\| \geq \epsilon$ **then**
    $\nu^f \leftarrow \nu^{opt}$
else
    $\nu^f \leftarrow \infty$
    Obtain $\boldsymbol{\theta}^f$ by maximizing the likelihood estimate (Equation 18) using constrained Nelder-mead algorithm.
end if
Assemble $\mathbf{R}(\boldsymbol{\theta}) = \prod_{d=1}^{n} \kappa\left(\theta_d^f, \nu^f\right)$
Cholesky factorize $\mathbf{R}(\boldsymbol{\theta})$
Compute the kriging weights $\boldsymbol{\omega} = \mathbf{R}^{-1}(\mathbf{y} - \boldsymbol{\mu})$
Estimate $\hat{\mathbf{y}} = \boldsymbol{\mu} + r'(\mathbf{X}) \cdot \boldsymbol{\omega}$ =0

$$\begin{aligned} \max_{\kappa,\boldsymbol{\theta}} \quad & -\frac{N_s}{2} ln(2\pi) - \frac{N_s}{2} ln\hat{\sigma}(\kappa, \boldsymbol{\theta}) - \frac{1}{2} ln|R(\kappa, \boldsymbol{\theta})| \\ \text{s.t.} \quad & \theta^{(d)} \geq 0 \end{aligned} \tag{21}$$

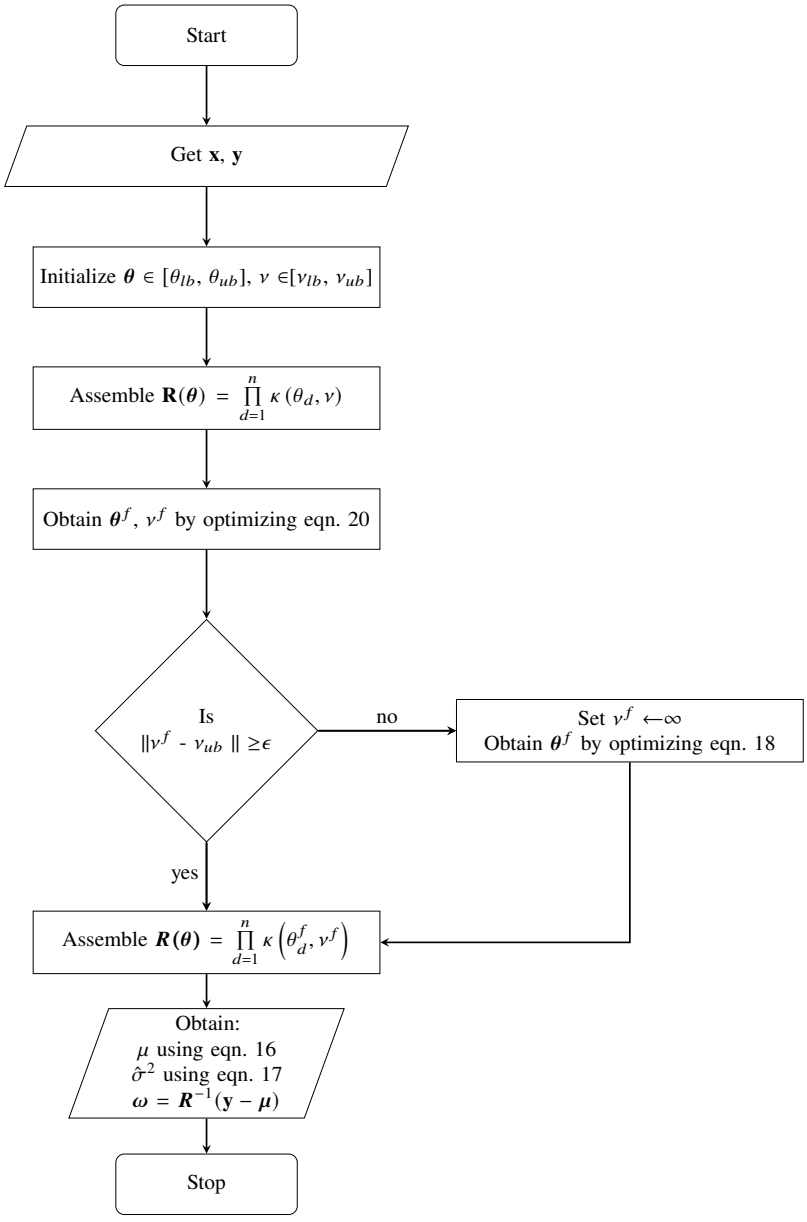


Fig. 14 Flowchart showing the use of proposed Optimal-ν method in model training

Algorithm 2 Optimal-κ method

Input: $\mathbf{x}$, $\mathbf{y}$, $\mathbf{X}$, kernel list $\boldsymbol{K} = [\kappa_1, \kappa_2, ...\kappa_p]$, t
Output: θ^f, κ^f, $\hat{\mathbf{y}}$

1: Generate t unique combinations
2: $\mathcal{S} = [\,]$; $LL_S = [\,]$
3: **for** each combination **do**
4: Set $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_t$
5: Assemble $\mathbf{R}(\boldsymbol{\theta}) = \prod_{d=1}^{n} \kappa_t \left(\theta^{(d)}\right)$
6: Obtain $\{\kappa_t^{opt}, \boldsymbol{\theta}_t^{opt}\}$ by maximizing the likelihood estimate (Equation 21)
7: Obtain the maximum likelihood estimate, LL from Step 6.
8: $\mathcal{S}$.append($\{\kappa^{opt}, \boldsymbol{\theta}^{opt}\}$)
9: LL_S.append(LL)
10: **end for**
11: $\{\kappa^f, \boldsymbol{\theta}^f\} = \arg\max_{\kappa,\boldsymbol{\theta}} LL_S$
12: Assemble $\mathbf{R}(\boldsymbol{\theta}) = \prod_{d=1}^{n} \kappa^f \left(\theta^{(d)}\right)$
13: Cholesky factorize $\mathbf{R}(\boldsymbol{\theta})$
14: Compute the kriging weights $\boldsymbol{\omega} = \mathbf{R}^{-1}(\mathbf{y} - \boldsymbol{\mu})$
15: Estimate $\hat{\mathbf{y}} = \boldsymbol{\mu} + r'(\mathbf{X}) \cdot \boldsymbol{\omega}$

VI. Test cases

To evaluate the performance of our proposed approach towards kernel selections, we benchmark using both algebraic functions and a real-world problem. To be specific, we use three 2-D algebraic, three 3-D algebraic and two real-world engineering problems. The 2-D problems were chosen from the study in Section IV. the problems are Branin, TPHT and the Haupt functions. A description of the problems are provided in the Appendix. For the 3-D algebraic problems, we consider, shear stress from a welded-beam [93], the flow of water through a borehole [94] and the Hartmann 3-dimensional function. For the real world problem, we use the axial transonic rotor case with NASA rotor 37 [21].

A. 3-D Welded Beam Problem

The problem of welded beam design is a popular example of some complex designs issues that arises in structural engineering [95]. It involves with designing the form of steel beams and with connecting them to form complex structures. This problem is a three-dimensional function which represents the shear stress from a welded beam. The input variables are the thickness, height and length of the beam. The detailed description of the variables and output is provided in Table 2. The analytical form of this function is given by Equation 22.

$$\tau = \sqrt{\tau'^2 + \tau''^2 + \frac{l\tau'\tau''}{\sqrt{0.25(l^2 + (h+t)^2)}}} \tag{22}$$

where,

$$\tau' = \frac{6000}{\sqrt{2}hl}$$

$$\tau'' = \frac{6000(14 + 0.5l)\sqrt{0.25(l^2 + (h+t)^2)}}{2[0.707hl(l^2/12 + 0.25(h+t)^2)]}$$

B. 3-D Water Flow Problem

The water flow problem is a eight-dimensional function which represents the volume of water flow in a borehole [94]. We formulate the function into a three-dimensional analytical problem by choosing the top three variables that contributes most to the volume of water in the borehole tank. The input variables are the radius of the borehole, radius of the influence and the length of the borehole. The values of the other variables are fixed. To be specific, we fixed the

Table 2 3-D Welded Beam Problem

Input/Output	Quantity	Description
y_1	τ	Shear stress
x_1	$5 \leq t \leq 10$	Beam thickness
x_2	$0.125 \leq h \leq 1$	Beam height
x_3	$5 \leq l \leq 10$	Beam length

values of the transmissivity of upper aquifer, potentiometric head of upper aquifer, transmissivity of lower aquifer, potentiometric head of lower aquifer, hydraulic conductivity of borehole to be 70,000 m^2/y, 1,000 m, 80m^2/y, 750 m and 10,000 m/y respectively. The detailed description of the variables and output is provided in Table 3. The analytical form of this function is given by Equation 23.

$$f = \frac{2\pi T_u (H_u - H_l)}{ln(r/r_w)\left[1 + \frac{2LT_u}{ln(r/r_w)r_w^2 K_w} + \frac{T_u}{T_l}\right]} \tag{23}$$

Table 3 3-D Water Flow Problem

Input/Output	Quantity	Description
y_1	f	Water flow
x_1	$0.05 \leq r_w \leq 0.15$	Radius of borehole (m)
x_2	$100 \leq r \leq 50000$	Radius of influence (m)
x_3	$1120 \leq L \leq 1680$	Length of borehole (m)
	$63070 \leq T_u \leq 115600$	Transmissivity of upper aquifer (m^2/y)
	$990 \leq H_u \leq 1110$	Potentiometric head of upper aquifer (m)
	$63.1 \leq T_l \leq 116$	Transmissivity of lower aquifer (m^2/y)
	$700 \leq H_l \leq 820$	Potentiometric head of lower aquifer (m)
	$9855 \leq K_w \leq 12045$	Hydraulic conductivity of borehole (m/y)

C. Hartmann 3D problem

The Hartmann 3D function is a three dimensional function which is usually evaluated in the domain: $x_i \in [0, 1]$. The function is known to have four local minima and a global minimum. The global minimum $f(\boldsymbol{x}^*) = -3.86278$ where $\boldsymbol{x}^*$ is (0.1146,0.5556,0.8525). The analytical form of this function is given by Equation 24.

$$f(x) = -\sum_{i=1}^{4} \alpha_i exp\left(-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2\right) \tag{24}$$

where,

$$\alpha = (1.0, 1.2, 3.0, 3.2)^T$$

$$A = \begin{bmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}$$

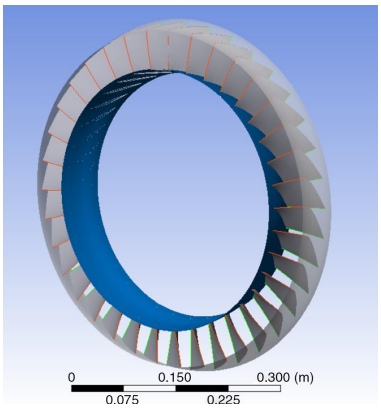
$$P = 10^{-4}\begin{bmatrix} 3689 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{bmatrix}$$

D. Axial transonic rotor case

This case is chosen because it is a typical real-world problem. Figure 15 shows the three-dimensional computer-aided design of the axial transonic rotor. For this case, the inputs are the twist and sweep of the datum. The input values range are summarized in the Table 4. The quantities of interest (QOIs) are the total pressure ratio and adiabatic efficiency. The QOIs are evaluated using the ANSYS Reynolds-averaged Navier Stokes (RANS) solver. For both outputs, the CFD simulations are performed at a speed of 1.7 $\times 10^4$ rpm on the rotor. A dataset consisting of 147 samples was generated using Latin hypercube sampling (LHS). The approximated response surfaces of the adiabatic efficiency and total pressure ratio are shown in Figure 16 (a) and (b) respectively.

Table 4 Value ranges for the axial transonic rotor input variable

	Lower limit	Upper limit
Twist (radians)	-0.125	0.22
Sweep (radians)	-0.02	0.05



(a) 2D

Fig. 15 Computer-aided design of the axial transonic rotor [21]

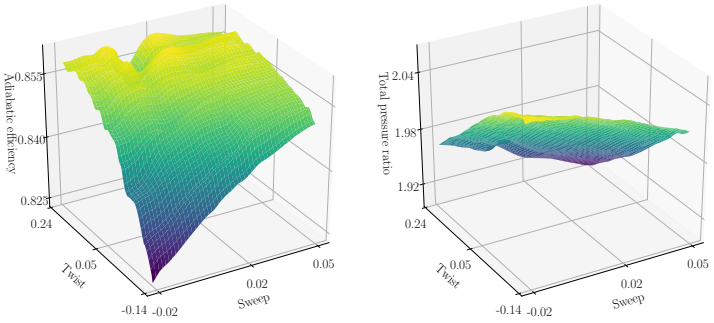
VII. Results

In this Section, we present a description of the different performance metrics and the benchmark results for the various test cases presented in Section VI.

A. Performance metrics

In comparing the performance of the proposed approach with models trained with single kernels, we employ the use of two different metrics. The metrics are the normalized root mean square error (NRMSE) and coefficient of determination (R^2). NRMSE and R^2 are expressed in Equations 25 and 26 respectively.

$$NRMSE = 100 * \sqrt{\frac{1}{m}\sum_{i}^{m}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2} \tag{25}$$



(a) Adiabatic efficiency (b) Total pressure ratio

Fig. 16 Generated response surfaces of the quantities of interest

$$R^2 = 1 - \frac{\sum_i^m (y_i - \hat{y}_i)^2}{\sum_i^m (y_i - \bar{y})^2} \tag{26}$$

where,

$$\bar{y} = \frac{1}{m} \sum_i^m y_i \tag{27}$$

From the Equations above, m is the number of points to be predicted. The true and predicted values are denoted as y_i and $\hat{y}_i$ respectively.

B. Results

The benchmark results for each test case explained in Section VI are provided in this subsection. For the cases, we benchmark the four single kernels in II.D with the proposed method. In the case of the algebraic functions, we use adaptive sampling to choose the optimal sample points. For our adaptive sampling technique, we initialized with a sample size N_s, of $10d$, where d is the problem dimension. This is as recommended by Jones *et al.* [96]. These initial sample points were drawn using Latin hypercube sampling [97]. In our sampling method for each problem, new points are continuously added till the defined convergence criteria is met. The final sampling points for the Branin, TPHT, Haupt, water flow, welded beam and Hartmann problems are 44,117,50,36,47 and 64, respectively. For axial transonic rotor, we use a sample size of 40 to train our Kriging models for both total pressure ratio (TPR) and adiabatic efficiency problems. After the training, the models are benchmarked against a well-defined test set. The results for the algebraic problems are summarized in tables 5 and 6. For the real-world problems, the results for the NRMSE and R^2 are provided in tables 7 and 8 respectively. In the tables, we present the selected kernels chosen by each of the proposed models by putting them in brackets after the benchmarked values. We represent the Gaussian, exponential, Matérn 3/2 and Matérn 5/2 kernels with G, E, M3 and M5 respectively. In the benchmarking results, the two top performing models are presented in bold texts.

Table 5 NRMSE (%) results for the algebraic test cases

	Branin (2D)	TPHT (2D)	Haupt (2D)	Water Flow (3D)	Welded Beam (3D)	Hartmann (3D)
Gaussian	0.0739	12.0957	24.2145	**0.0388**	1.7552	1.5491
Exponential	9.8358	2.9078	1.9755	3.1023	4.1305	3.7306
Matérn 3/2	2.2200	**1.4607**	0.5201	0.3642	2.1062	**1.5197**
Matérn 5/2	**0.0469**	15.7560	**0.3450**	0.0924	**1.4122**	20.7910
Proposed method A	**0.0651(G)**	3.2820(G)	1.2208(G)	0.054(G)	**1.5677(G)**	**1.5491(G)**
Proposed method B	0.0662(G)	**2.5830(M3)**	**0.4686(M5)**	**0.042(G)**	1.8183(M5)	1.5837(M3)

Table 6 R^2 results for the algebraic test cases

	Branin (2D)	TPHT (2D)	Haupt (2D)	Water Flow (3D)	Welded Beam (3D)	Hartmann (3D)
Gaussian	0.9999	0.8599	0.4008	**0.9999**	0.9919	0.9965
Exponential	0.7919	0.9956	0.9902	0.9835	0.9550	0.9798
Matérn 3/2	0.9894	**0.9989**	0.9993	0.9997	0.9883	**0.9967**
Matérn 5/2	**0.9999**	0.6623	**0.9997**	0.9999	**0.9947**	0.5674
Proposed method A	**0.9999(G)**	0.9948(G)	0.9962(G)	0.9999(G)	**0.9935(G)**	**0.9965(G)**
Proposed method B	0.9999(G)	**0.9967(M3)**	**0.9994(M5)**	**0.9999(G)**	0.9912(M5)	0.9964(M3)

The Branin function is known to be a smooth function as seen in its function profile in Figure 21. The likelihood of the function with the four kernels given in Figure 6 also agrees with this. In the benchmark, the Gaussian kernel performed best with a NRMSE of 0.0739% and a R^2 value of 0.9999 compared to the remaining three kernels. The proposed methods also selected the Gaussian kernel and gave similar benchmarking result of approximately 0.066% NRMSE.

The tensor-product hyperbolic tangent (TPHT) problem approximates a step function. The problem is known to cause oscillations with some surrogate modeling approaches [48]. This difficulty is seen in the likelihood profiles especially in the Gaussian kernel likelihood profile as shown in Figure 9a. Similar complexity is also observed in the likelihood profile of the Matérn 5/2 kernel, as shown in Figure 9d. In the figures, the chances of the hyperparameter optimization converging in a local optima is heightened. This complexity is seen to affect the performance of both kernels. For the problem, the Gaussian and Matérn kernels have a R^2 values of 0.8599 and 0.6623 respectively as against the good performance observed with the exponential and Matérn 3/2 kernels. It is interesting that our first proposed method also chose to use the Gaussian kernel and performed well. This improvement in performance can be attributed to the use of informative hyperparameter initialization in our methods. The second method was able to avoid the poorly performing kernels and selected the Matérn 3/2 kernel for model training. The method upon benchmarking gave a NRMSE and R^2 of 2.5830% and 0.9967 respectively.

The Gaussian kernel likelihood profile of the Haupt function as shown in Figure 11a is similar with that of the TPHT problem. The complexity in the likelihood profile for the Gaussian kernel is seen to affect the performance of the kernel in the benchmarking study. With the kernel, a poor performance of 24.2145% NRMSE is observed. Other kernels are seen to reasonable perform well with the Matérn 5/2 performing best with a NRMSE of approximately 0.35%. With our proposed methods, a significant improvement in performance is observed. To be specific, even though the first proposed method chose the Gaussian kernel, the performance improvement is quite significant with a R^2 value of 0.9962 compared to the R^2 single kernel benchmark estimate of just 0.40.

Moving to the three dimensional test cases, the four kernels performed well on the water flow problem. Regardless of the good modeling performance, the Gaussian kernel is observed to have the most superior performance with a NRMSE and R^2 values of 0.0388% and 0.9999 respectively. The proposed methods both selected the Gaussian kernel for model training. The performance on the chosen test set is also similar to the single Gaussian kernel. Proposed methods A and B have a NRMSE values of 0.054% and 0.042% respectively.

The benchmark carried out using the welded beam showed the superior performance of the Gaussian and Matérn 3/2 kernels. The Matérn 3/2 performed best with NRMSE and R^2 values of 1.4122% and 0.9947 respectively. The proposed methods differ in the selection of the best kernels. The first proposed method selected the Gaussian kernel and gave a much better performance than the single Gaussian kernel upon benchmark. The method improved the performance by reducing the NRMSE by a significant percentage of approximately 10.7% when compared with the single Gaussian kernel. The second proposed method selected the Matérn 5/2 kernel upon training, yielding a model which has 0.9912 R^2 value. For this benchmark, the proposed methods avoiding the less performing kernels: the exponential and Matérn 3/2 kernels.

The performance of the Matérn 5/2 kernel on the Hartmann 3D problem set is very poor with a low R^2 and NRMSE values of 0.5674 and 20.79%. It would have been a bad generalization if the Matérn 5/2 kernel is used for training a function with similar characteristics to the Hartmann 3D. Benchamarking the proposed methods against the single kernel methods gave good results. The proposed methods were also able to avoid poorly performing kernels in this test case. The first proposed method selected the Gaussian kernel upon model training and had similar performance with a

R^2 of approximately 0.9965. The second method selected the Matérn 3/2 kernel and gave similar performance when the kernel is used separately during model training.

Table 7 NRMSE (%) results for the real-world test cases

	Total Pressure Ratio	Adiabatic Efficiency
Gaussian	**0.7786**	1.7655
Exponential	1.1650	2.2920
Matérn 3/2	**0.6906**	2.7961
Matérn 5/2	1.6718	3.6433
Proposed method A	0.7793(G)	**1.7639(G)**
Proposed method B	0.8067(M3)	**1.6713(G)**

Table 8 R^2 results for the real-world test cases

	Total Pressure Ratio	Adiabatic Efficiency
Gaussian	**0.9989**	0.9943
Exponential	0.9977	0.9904
Matérn 3/2	**0.9992**	0.9857
Matérn 5/2	0.9953	0.9757
Proposed method A	0.9989(G)	**0.9943(G)**
Proposed method B	0.9989(M3)	**0.9949(G)**

For the real-world problem sets, we evaluate the performance of the proposed methods against the single kernel methods. For the total pressure ratio, the proposed methods differed in their choice of kernels. The first method selected the Gaussian kernel during model training while the Matérn 3/2 kernel is used by the second method. The performance of the two methods are similar with their R^2 value being 0.9989 approximately. This performance is similar to the performance of the Gaussian and Matérn 3/2 kernels in the single kernel benchmark. Again, our methods selected the top performing kernels for training leading to a generally acceptable model performance.

In the case of the adiabatic efficiency test problem, the Gaussian kernel outperformed other kernels with a low NRMSE of 1.7655%. The proposed methods both selected the Gaussian kernel during model training. The proposed methods upon benchmark yielded NRMSE values of 1.7639% and 1.6713%, which are similar to the Gaussian kernel model performance.

VIII. Conclusion and future direction

In this paper, we expose the difficulties that can be encountered when selecting kernels in kernel-based methods such as Kriging. We attempt to gain more knowledge on the characteristics of different Kriging models when used to capture various complex problems. We start by presenting the results for several studies wrapped around kernel selections. To be specific, we studied the effects of problem profiles, sample sizes on kernel selections. We also investigated the sensitivity of the four common kernels to hyperparameter values. We did this by inspecting the response of the likelihood and error estimates to changes in hyperparameter values for each kernel. From the studies, the complexity of the likelihood profile of the Gaussian kernel is shown. The complication has been shown to limit the general model capability of the kernel in some cases. Our benchmark studies also showed cases where the Matérn 5/2 performed poorly as seen in Tables 5 and 6. Towards improving model performance, we proposed two methods and examined the behaviour and performance of our proposed methods on both algebraic and real-world test cases. For the algebraic problem benchmark, we chose three unique cases from the kernel studies and three more benchmark problems which are all three-dimensional. We considered the total pressure ratio and adiabatic efficiency from the axial transonic rotor problem for the real-world test case. Upon benchmarking our proposed methods against the four kernels, our methods performed well across board. The methods successfully avoided poorly performing kernels and in some cases the first proposed method significantly improved the performance of the Gaussian kernel. From our benchmark, we also observed the good modeling performance of the Matérn 3/2 kernel and would recommend that it is considered as the first point of call in kernel selections.

In the future, it will be worth exploring more methods that can help in selecting the best kernel for engineering problems. A meta-approach towards kernel selection would be an interesting direction. To achieve good results with such methods, it will be desirable to define a set of statistical descriptors that can effectively capture the characteristics of problems. Such characteristics can then help in creating a rule-based system for kernel selections. In defining the statistical descriptors, capturing the relationship between the input and output values especially for regression problems is essential. For the limitation observed when the Gaussian kernel is used to model complex problems, it might be worth the attempt to examine the effect of using an ensemble of Gaussians. In the method, few Gaussian kernel models (two or three) might be initialized at random before training. After training, the models are used for prediction. The predicted outputs can then be combined using information derived from the different models. The optimal likelihood is one that can be used to derive the weights for combination. The Akaike information criteria (AIC) [98] or the Bayesian information criteria (BIC) [73, 74], which are dependent on the optimal likelihood. These criterion can be used for weight computations. Lastly, It will be desirable to study the effect of pre-known hyperparameter range on the training time of kriging models using the four common kernels. Also, it will be interesting to study the influence of distribution of sampling points on model performance for different kernels.

As a general recommendation for kernel selections in other kernel-based methods, either one of two options is proposed. Firstly, finding a common parameter between the commonly used kernels and optimizing such parameter while carrying out model training. Also, it will be interesting to see the outcome of framing the hyperparameter optimization for kernel-based methods as a mixed-optimization problem. This is to facilitate the simultaneous learning of kernels and optimization of their hyperparameters.

Acknowledgments

The authors would like to acknowledge receipt of financial support from the Research Grants Council (RGC) of the Hong Kong Special Administrative Region under the Hong Kong Ph.D. Fellowship Scheme (HKPFS).

References

[1] Scholkopf, B., Smola, A., and Muller, K. R., "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput*, 1998, pp. 1299–1319.

[2] Hofmann, T., Scholkopf, B., and Smola, A. J., "A review of kernel methods in machine learning," , 2006.

[3] Cressie, N., "The origins of kriging," Vol. 22, 1990, pp. 239–252. https://doi.org/101007/bf00889887.

[4] Cressie, N., *Statistics for spatial data*, Wiley series in probability and statistics, 1993.

[5] Kanagawa, M., Hennig, P., Sejdinovic, D., and Sriperumbudur, B. K., "Gaussian Processes and Kernel Methods:A Review on Connections and Equivalences," , 2018.

[6] Mika, S., Ratsch, G., Weston, J., Scholkopf, B., and Mullers, K.-R., "Fisher discriminant analysis with kernels," *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop (cat no 98th8468)*, IEEE, 1999, pp. 41–48.

[7] Matheron, G., "Principles of geostatistics," Vol. 58, 1963, pp. 1246–1266. https://doi.org/102113/gsecongeo5881246.

[8] Pérez-Cruz, F., and Bousquet, O., "Kernel methods and their potential use in signal processing," *IEEE Signal Processing Magazine*, Vol. 21, No. 3, 2004, pp. 57–65.

[9] Gönen, M., and Margolin, A. A., "Localized data fusion for kernel k-means clustering with application to cancer biology," *Advances in Neural Information Processing Systems*, Vol. 27, 2014, pp. 1305–1313.

[10] Fukumizu, K., and Leng, C., "Gradient-based kernel dimension reduction for regression," *Journal of the American Statistical Association*, Vol. 109, No. 505, 2014, pp. 359–370.

[11] Jiang, J., "Information extraction from text," *Mining text data*, Springer, 2012, pp. 11–41.

[12] Bishop, C. M., *Pattern Recognition and Machine Learning*, softcover reprint of the original 1st edition 2006 (corrected at 8th printing 2009) ed., Information Science and Statistics, Springer New York, New York, NY, 2016.

[13] Cover, T. M., "Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition," *IEEE Trans Electron Comput*, 1965, pp. 326–334.

[14] Pilario, K. E., Shafiee, M., Cao, Y., and Yang, S.-H., "A Review of Kernel Methods for Feature Extraction in Nonlinear Process Monitoring," *Processes*, 2020. https://doi.org/103390/pr8010024.

[15] Afkanpour, A., Szepesvári, C., and Bowling, M., "Alignment based kernel learning with a continuous set of base kernels," *Machine learning*, Vol. 91, No. 3, 2013, pp. 305–324.

[16] Palar, P. S., Liem, R. P., Zuhal, L. R., and Shimoyama, K., "On the use of surrogate models in engineering design optimization and exploration," , 2019. https://doi.org/101145/33196193326813.

[17] Rasmussen, C. E., and Williams, C. K. I., *Gaussian Processes for Machine Learning*, 2006.

[18] David Ginsbourger, L. C., Céline Helbert, "Discrete Mixtures of Kernels for Kriging-based Optimization," , Mar. 2008.

[19] Abdessalem, A. B., Dervilis, N., Wagg, D. J., and Worden, K., "Automatic kernel selection for Gaussian processes regression with approximate Bayesian computation and sequential Monte Carlo," *Front Built Environ*, 2017. https://doi.org/103389/fbuil201700052.

[20] Palar, P. S., and Shimoyama, K., "Kriging with Composite Kernel Learning for Surrogate Modeling in Computer Experiments," 2019. https://doi.org/102514/62019-2209.

[21] Palar, P. S., Zuhal, L. R., and Shimoyama, K., "Gaussian Process Surrogate Model with Composite Kernel Learning for Engineering Design," *AIAA Journal*, 2020. https://doi.org/https://doiorg/102514/1J058807.

[22] Moreaux, G., "Compactly supported radial covariance functions," *Journal of Geodesy*, Vol. 82, No. 7, 2008, pp. 431–443.

[23] Liem, R. P., Mader, C. A., and Martins, J. R. R. A., "Surrogate models and mixtures of experts in aerodynamic performance prediction for aircraft mission analysis," *Aerospace Science and Technology*, Vol. 43, 2015, pp. 126–151.

[24] Gönen, M., and Alpaydın, E., "Multiple kernel learning algorithms," *The Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2211–2268.

[25] Liu, X., Wang, L., Zhu, X., Li, M., Zhu, E., Liu, T., Liu, L., Dou, Y., and Yin, J., "Absent multiple kernel learning algorithms," *IEEE transactions on pattern analysis and machine intelligence*, Vol. 42, No. 6, 2019, pp. 1303–1316.

[26] Liem, R. P., Oyetunde, K. S., Palar, P. S., and Shimoyama, K., "Kriging with mixed kernel (MK) for complex aerospace problems," Sixteenth International Conference on Flow Dynamics, 2019.

[27] Goel, T., Haftka, R. T., Shyy, W., and Queipo, N. V., "Ensemble of surrogates," *Structural and Multidisciplinary Optimization*, Vol. 33, No. 3, 2006, pp. 199–216. https://doi.org/101007/s00158-006-0051-9.

[28] Acar, E., and Rais-Rohani, M., "Ensemble of metamodels with optimized weight factors," *Structural and Multidisciplinary Optimization*, 2008.

[29] Yin, H., Fang, H., Wen, G., Gutowski, M., and Xiao, Y., "On the ensemble of metamodels with multiple regional optimized weight factors," Vol. 58, 2018, pp. 245–263. https://doi.org/101007/s00158-017-1891-1.

[30] Palar, P. S., and Shimoyama, K., "Ensemble of Kriging with Multiple Kernel Functions for Engineering Design Optimization," , 2018. https://doi.org/101007/978-3-319-91641-5_18.

[31] Palar, P. S., and Shimoyama, K., "Efficient global optimization with ensemble and selection of kernel functions for engineering design," Vol. 59, 2019, pp. 93–116. https://doi.org/101007/s00158-018-2053-9.

[32] Ergul, U., and Bilgin, G., "MCK-ELM: multiple composite kernel extreme learning machine for hyperspectral images," *Neural Computing and Applications*, 2019, pp. 1–11.

[33] Nelles, O., "Nonlinear dynamic system identification," *Nonlinear System Identification*, Springer, 2001, pp. 547–577.

[34] Abrahamsen, T. J., *Kernel Methods for Machine Learning with Life Science Applications*, DTU Compute, 2013.

[35] Iosifidis, A., Tefas, A., Pitas, I., and Gabbouj, M., "A Review of Approximate Methods for Kernel-based Big Media Data Analysis," 24th European Signal Processing Conference (EUSIPCO), 2016.

[36] Camps-Valls, G., and Bruzzone, L., "Kernel-based methods for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 43, No. 6, 2005, pp. 1351–1362.

[37] Gomez-Chova, L., Munoz-Mari, J., Laparra, V., Malo-Lopez, J., and Camps-Valls, G., "A Review of Kernel Methods in Remote Sensing Data Analysis," , 2011. https://doi.org/101007/978-3-642-14212-3_10.

[38] Schölkopf, B., Smola, A. J., Bach, F., et al., *Learning with kernels: support vector machines, regularization, optimization, and beyond*, MIT press, 2002.

[39] Mercer, J., "Functions of positive and negative type and their connection with the theory of integral equations," *Philos Trans Royal Soc (A)*, 1909, pp. 69–70.

[40] Duvenaud, D., Lloyd, J., Grosse, R., Tenenbaum, J., and Zoubin, G., "Structure discovery in nonparametric regression through compositional kernel search," *International Conference on Machine Learning*, PMLR, 2013, pp. 1166–1174.

[41] Braun, M. L., Buhmann, J. M., and Muller, K. L., "On relevant dimensions in kernel feature spaces," *The Journal of Machine Learning Research*, 2008, pp. 1875–1908.

[42] Moustapha, M., Bourinet, J.-M., Guillaume, B., and Sudret, B., "Comparative Study of Kriging and Support Vector Regression for Structural Engineering Applications," Vol. 4, 2018, p. 04018005. https://doi.org/101061/ajrua60000950.

[43] Cortes, C., and Vapnik, V., "Support-Vector Networks," *Machine Learning*, 1995, pp. 273–297. https://doi.org/https://doiorg/101007/BF00994018.

[44] Vapnik, V., *The Nature of Statistical Learning Theory*, Springer, New York, 1995.

[45] Vapnik, V., *Statistical Learning Theory*, Wiley, New York, 1998.

[46] Smola, A., and Scholkopf, B., "A tutorial on support vector regression," *Statistics and Computing*, 2004, pp. 199–222.

[47] Domingos, P., "A few useful things to know about machine learning," *Communications of the ACM*, 2012, pp. 78–87. https://doi.org/https://doiorg/101145/23477362347755.

[48] Hwang, J. T., and Martins, J. R. R. A., "A fast-prediction surrogate model for large datasets," *Aerospace Science and Technology*, Vol. 75, 2018, pp. 74–87. https://doi.org/101016/jast201712030.

[49] Gong, D., Wei, C., Wang, L., Feng, L., and Wang, L., "Adaptive Methods for Center Choosing of Radial Basis Function Interpolation: A Review," Information Computing and Applications, 2010, pp. 573–580.

[50] Hardy, R. L., "Multiquadric equations of topography and other irregular surfaces," *Journal of Geophysical Research*, 1971, pp. 1905–1915.

[51] Duchon, J., "Splines minimizing rotation-invariant semi-norms in Sobolev spaces," , 1977. https://doi.org/101007/BFb0086566.

[52] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and Analysis of Computer Experiments," Vol. 4, 1989, pp. 409–423. https://doi.org/101214/ss/1177012413.

[53] Gramacy, R. B., "Surrogates: Gaussian Process Modeling, Design, and Optimization for the Applied Sciences," 2020.

[54] Hang, H., and Steinwart, I., "Optimal Learning with Anisotropic Gaussian SVMs," , 2018.

[55] Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D., "Bayesian Prediction of Deterministic Functions, with Applications to the Design and Analysis of Computer Experiments," *Journal of the American Statistical Association*, Vol. 86(416), 1991, pp. 953–963.

[56] Lophaven, S. N., Nielsen, H. B., and Sondergaard, J., "DACE - A Matlab Kriging Toolbox, Version 2 0," , ????

[57] Martin, J. D., and Simpson, T. W., "Use of Kriging Models to Approximate Deterministic Computer Models," Vol. 43, 2005, pp. 853–863. https://doi.org/102514/18650.

[58] Bachoc, F., "Cross Validation and Maximum Likelihood estimation of hyper-parameters of Gaussian processes with model misspecification," *Comput Stat Data Anal*, Vol. 66, 2013, pp. 55–69.

[59] Gano, S. E., Renaud, J. E., Martin, J. D., and Simpson, T. W., "Update strategies for kriging models used in variable fidelity optimization," *Structural and Multidisciplinary Optimization*, Vol. 32, No. 4, 2006, pp. 287–298. https://doi.org/101007/s00158-006-0025-y.

[60] Dubrule, O., "Cross validation of kriging in a unique neighborhood," *Mathematical Geology*, 1983.

[61] Yu, T., and Zhu, H., "Hyper-parameter optimization: A review of algorithms and applications," *arXiv preprint arXiv:2003 05689*, 2020.

[62] Martins, J. R., and Ning, A., *Engineering design optimization*, Cambridge University Press, 2021.

[63] Luersen, M. A., Riche, R. L., and Guyon, F., "A constrained, globalized, and bounded Nelder?Mead method for engineering optimization," Vol. 27, 2004, pp. 43–54. https://doi.org/101007/s00158-003-0320-9.

[64] Blaessle, A., "constNMPy: A Python package for constrained Nelder-Mead optimization," , 2017. URL https://githubcom/alexblaessle/constrNMPy.

[65] Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M., "Optuna: A Next-generation Hyperparameter Optimization Framework," , 2019.

[66] Nelder, J. A., and Mead, R., "A simplex method for function minimization," *The computer journal*, Vol. 7, No. 4, 1965, pp. 308–313.

[67] Mohammadi, H., "Kriging-based black-box global optimization: analysis and new algorithms," Ph.D. thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne, 2016.

[68] Agrawal, T., "Optuna and AutoML," *Hyperparameter Optimization in Machine Learning*, Springer, 2021, pp. 109–129.

[69] Viana, F. A. C., Haftka, R. T., and Steffen, V., "Multiple surrogates: how cross-validation errors can help us to obtain the best predictor," *Structural and Multidisciplinary Optimization*, Vol. 39, No. 4, 2009, pp. 439–457.

[70] Biswas, S., Chakraborty, S., Chandra, S., and Ghosh, I., "Kriging-based approach for estimation of vehicular speed and passenger car units on an urban arterial," *Journal of Transportation Engineering, Part A: Systems*, Vol. 143, No. 3, 2017, p. 04016013.

[71] Duvenaud, D., "Automatic model construction with Gaussian Processes," Ph.D. thesis, 2014.

[72] Wagenmakers, E.-J., and Farrell, S., "AIC model selection using Akaike weights," *Psychonomic Bulletin & Review*, 2004.

[73] Burnham, K. P., and Anderson, D. R., "Multimodel inference: understanding AIC and BIC in Model Selection," *Sociology Methods Research*, 2004, pp. 261–304.

[74] Aho, K., Derryberry, D., and Peterson, T., "Model selection for ecologists: the worldviews of AIC and BIC," *Ecology*, 2014, pp. 631–636. https://doi.org/101890/13-1452.

[75] Simpson, F., Davies, I., Lalchand, V., Vullo, A., Durrande, N., and Rasmussen, C., "Kernel Identification Through Transformers," *arXiv preprint arXiv:2106 08185*, 2021.

[76] Ali, S., and Smith-Miles, K. A., "A meta-learning approach to automatic kernel selection for support vector machines," *Neurocomputing*, 2006, pp. 173–186.

[77] Wilson, A. G., and Adams, R. P., "Gaussian Process Kernels for Pattern Discovery and Extrapolation," , 2013.

[78] Salem, M. B., and Tomaso, L., "Automatic selection for general surrogate models," *Structural and Multidisciplinary Optimization*, 2018.

[79] Fischer, B., Gorbach, N., Bauer, S., Bian, Y., and Buhmann, J. M., "Model Selection for Gaussian Process Regression by Approximation Set Coding," 2016.

[80] Buhmann, J. M., "Simbad: Emergence of pattern similarity," *Similarity-Based Pattern Analysis and Recognition*, Springer London, 2013, pp. 45–64. https://doi.org/101007/978-1-4471-5628-4_3.

[81] Salakhutdinov, R., and Hinton, G. E., "Using Deep Belief Nets to Learn Covariance Kernels for Gaussian Processes," *NIPS*, Vol. 7, Citeseer, 2007, pp. 1249–1256.

[82] Halton, J. H., "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," Vol. 2, 1960, pp. 84–90. https://doi.org/101007/bf01386213.

[83] Gramacy, R., "laGP: large-scale spatial modeling via local approximate Gaussian processes in R," *Journal of Statistical Software*, Vol. 72(1), 2016, pp. 1–46.

[84] Lynch, J. F., "Analysis and application of adaptive sampling," *Journal of computer and system sciences*, Vol. 66, 2003, pp. 2–19.

[85] Eason, J., and Cremaschi, S., "Adaptive sequential sampling for surrogate model generation with artificial neural networks," *Computers and Chemical Engineering*, Vol. 68, 2014, pp. 220–232.

[86] Liu, H., Cai, J., and Ong, Y., "An adaptive sampling approach for Kriging metamodeling by maximizing expected prediction error," *Computers and Chemical Engineering*, Vol. 106, 2017, pp. 171–182.

[87] Zhai, Z., Li, H., and Wang, X., "An adaptive sampling method for Kriging surrogate model with multiple outputs," *Engineering with Computers*, 2020.

[88] Zuhal, L. R., Faza, G. A., Palar, P. S., and Liem, R. P., "Fast and Adaptive Reliability Analysis via Kriging and Partial Least Squares," *AIAA SciTech Forum*, 2021. https://doi.org/102514/62021-0675.

[89] Bouhlel, M. A., Bartoli, N., and Otsmane, A. M. J., "Improving kriging surrogates of high-dimensional design models by Partial Least Squares dimension reduction," *Structural and Multidisciplinary Optimization*, 2016, pp. 935–952.

[90] Bouhlel, M. A., and Martins, J. R. R. A., "Gradient-enhanced kriging for high-dimensional problems," , 2017.

[91] Bouhlel, M. A., Bartoli, N., Otsmane, A., and Morlier, J., "Efficient global optimization for high dimensional constrained problems by using the Kriging models combined with the partial least squares method," *Engineering Optimization*, 2018. https://doi.org/101080/0305215X20171419344.

[92] Moser, I., "Hooke-jeeves revisited," *2009 IEEE Congress on Evolutionary Computation*, IEEE, 2009, pp. 2670–2676.

[93] Deb, K., "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, 2000, pp. 311–338.

[94] Morris, M. D., Mitchell, T. J., and Ylvisaker, D., "Bayesian design and analysis of computer experiments: Use of derivatives in surface prediction," *Technometrics*, 1993, pp. 243–255.

[95] Mosavi, A., and Vaezipour, A., "Reactive search optimization; application to multiobjective optimization problems," *Applied Mathematics*, Vol. 3, No. 10A, 2012, pp. 1572–1582.

[96] Jones, S. M., D R, and Welch, W. J., "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, 1998, p. 455–492.

[97] Collins, J. C., and III, "Latin Hypercube Sampling in Sensitivity Analysis," , 1994. https://doi.org/1021236/ada285867.

[98] deLeeuw, J., "Introduction to Akaike (1973) information theory and an extension of the maximum likelihood principle," *Breakthroughs in Statistics I*, 1992, pp. 599–609.

Appendix

A. Haupt Function

The Haupt function is a two dimensional multimodal function. This function is usually evaluated in the domain: $x_i \in [0, 4]$. The analytical form of this function is given by Equation 28. Figure 17 shows the response surface plot of the function.

$$f = x_1 sin(4x_1) + 1.1x_2 sin(2x_2) \tag{28}$$

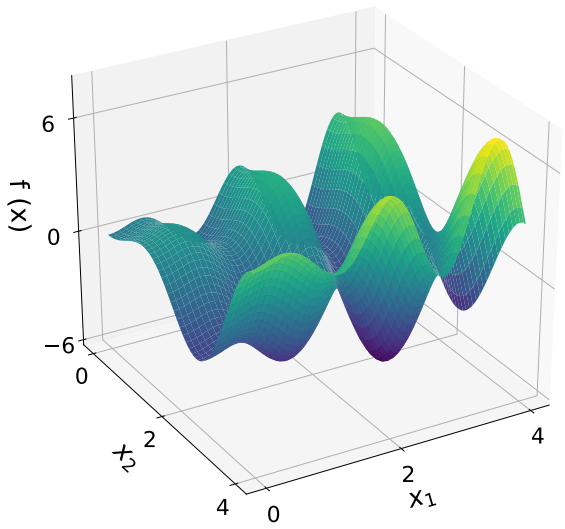


Fig. 17 Haupt function profile

B. Robot Arm Function

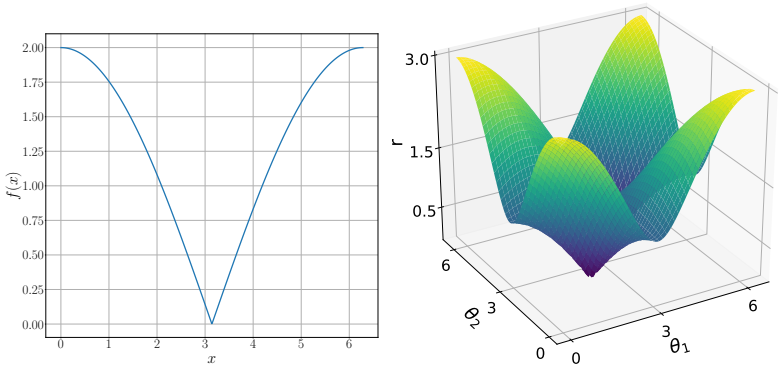
We modified the robot arm function to be a n-dimensional problem. Even though it has as inputs the angles ($\theta \in [0, 2\pi]$) and lengths ($L \in [0, 1]$) of the arm segments. We assumed a fixed length of the robot arm to be 1.0 and made the angle of the arm segment to be the variable. The output of the function still remains the distance of tip of the robot arm (r). Figure 18(a) and (b) shows the one and two dimensional plot of the function respectively. The resulting function profile can be characterized as non-linear and symmetric. The function profile is expressed as;

$$r = \sqrt{\left(\sum_{i=1}^{n+1} L_i \cos\left(\sum_{j=1}^{i} \theta_j\right)\right)^2 + \left(\sum_{i=1}^{n+1} L_i \sin\left(\sum_{j=1}^{i} \theta_j\right)\right)^2} \tag{29}$$

C. Tensor Product Hyperbolic Tangent Function

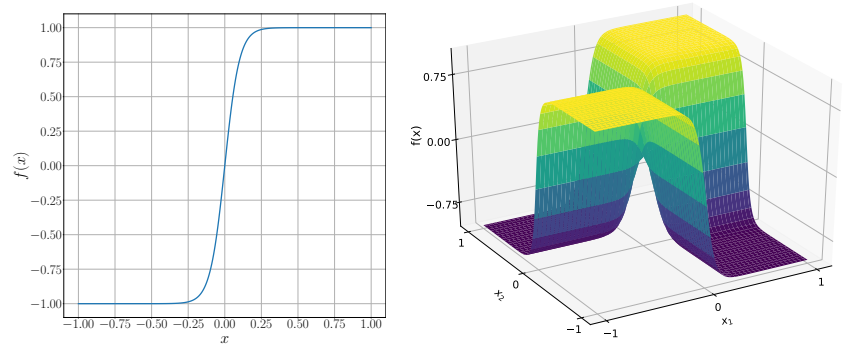
This function can fit in different dimensions n. In one dimension, the function has a unit step profile. It has as inputs $x \in [-1, 1]$ and the abruptness of the step function a. We keep the value of the abruptness fixed as 10 in all of its usage in this paper. Figure 19(a) and (b) shows the one and two dimensional plot of the function. The function profile is expressed as;

$$f = \prod_{i=1}^{n} \tanh(ax_i) \tag{30}$$



(a) 1D (b) 2D

Fig. 18 Robot arm function profile.



(a) 1D (b) 2D

Fig. 19 Tensor product hyperbolic tangent function.

D. Cantilever Beam Problem

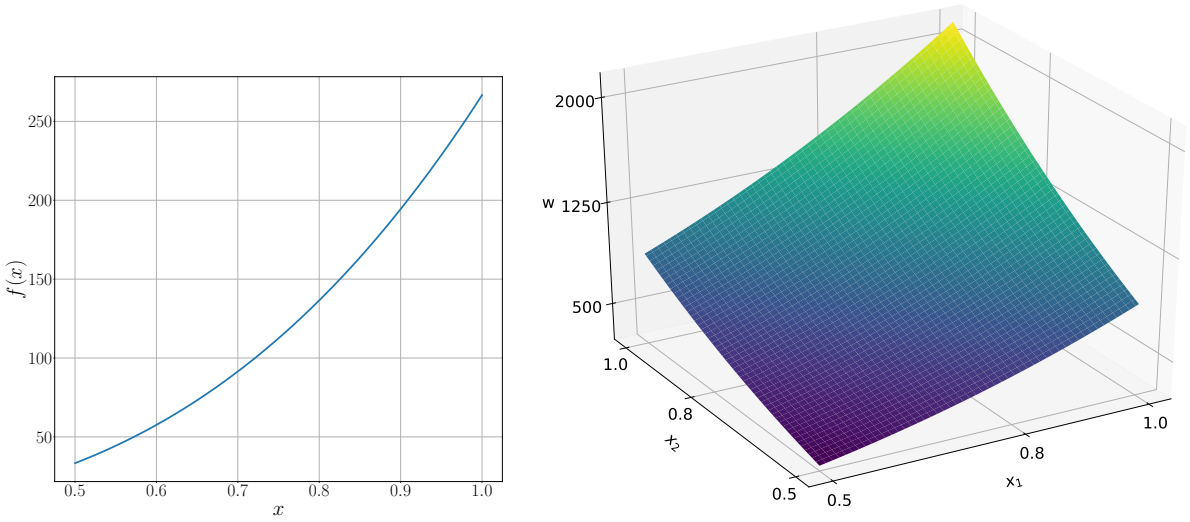
It is an n-dimensional function profile which can be characterized as linear. It has as inputs the lengths ($l \in [0.5, 1.0]$), widths ($b \in [0.01, 0.05]$) and heights ($h \in [0.30, 0.65]$) of the elements, applied force at the tip P and Young's modulus E and the tip deflection (w) as the output. Figure 20(a) and (b) shows the one and two dimensional plot of the function. The function profile is expressed as;

$$w = \frac{P}{3E} \sum_{i=1}^{n} \left[\frac{12}{b_i h_i^3} \left(\left(\sum_{j=i}^{n} l_j \right)^3 - \left(\sum_{j=i+1}^{n} l_j \right)^3 \right) \right] \tag{31}$$

E. Branin Function

The Branin or Branin-Hoo function is a commonly used analytical function of two dimensions that is continuous and non-convex. This function is usually evaluated in the domain: $x_1 \in [-5, 10]$ and $x_2 \in [0, 15]$. The analytical form of this function is given by Equation 32. Figure 21 shows the response surface plot of the function.

$$f(x_1, x_2) = \left(x_2 - \frac{5.1 x_1^2}{4\pi^2} + \frac{5 x_1}{\pi} - 6 \right)^2 + 10\left(1 - \frac{1}{8\pi}\right) \cos(x_1) + 10 \tag{32}$$



(a) 1D (b) 2D

Fig. 20 Cantilever beam function profile

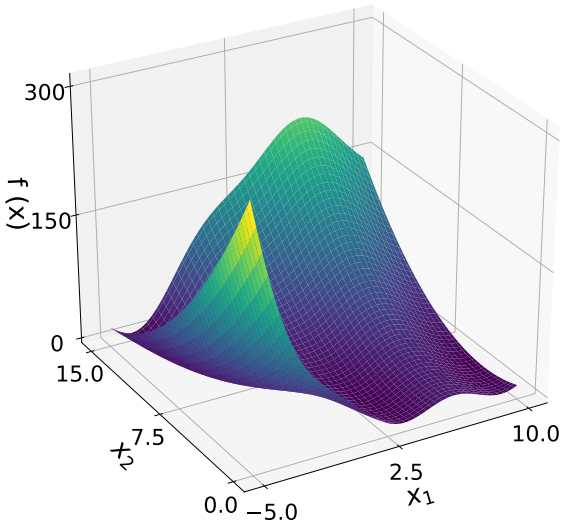


Fig. 21 Branin function profile

F. Camel back Function

The six hump camel function, simply known as the camel function is a commonly used analytical 2D function. The function is usually defined on an input domain: $x_1 \in [-3, 3]$ and $x_2 \in [-2, 2]$. The response surface of the function is shown in Figure 22. The analytical form of this function is given by Equation 33:

$$f(x_1, x_2) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2 \tag{33}$$

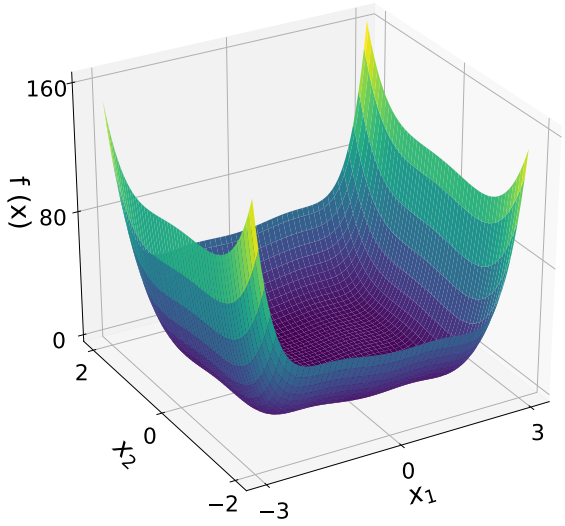


Fig. 22 Camel back function profile

G. Rosenbrock Function

The Rosenbrock function also known as Valley or Banana function, is a commonly used analytical function of two dimensions. The function is unimodal, and the global minimum lies in a narrow, parabolic valley. This function is usually evaluated in the domain: $x_i \in [-5, 10]$. The analytical form of this function is given by Equation 34. Figure 23 shows the response surface plot of the function.

$$f = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right] \tag{34}$$

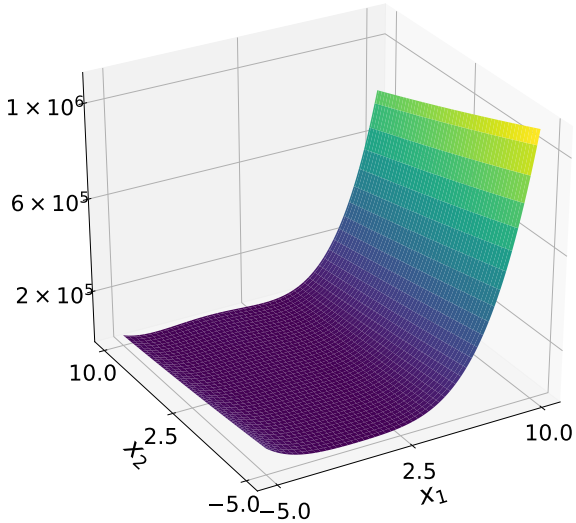


Fig. 23 Rosenbrock function profile

H. Hosaki Function

$$f = 1 - 8x_1 + 7x_1^2 - \frac{7}{3}x_1^3 + \frac{1}{4}x_1^4)x_2^2 e^{-x_2} \tag{35}$$

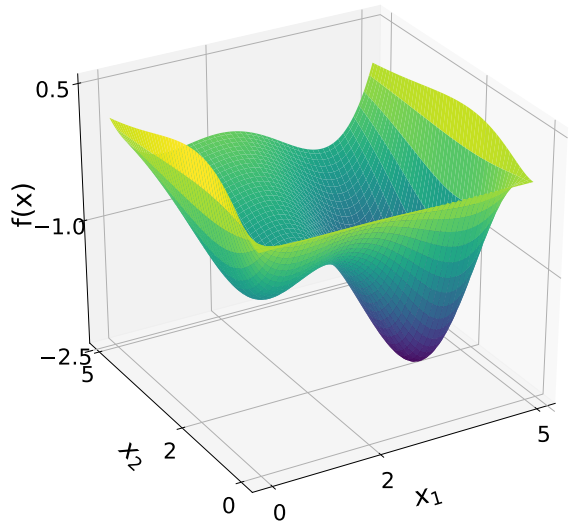


Fig. 24 Hosaki function profile

I. Function with contrasting profiles

The function is one dimensional with input $x \in [0, 2]$ and clearly multiple profiles. As shown in Figure 25, the profile is flat at smaller values, $x \in [0, 0.5]$ and an oscillation of higher magnitudes observed $x \in [0.5, 2]$.

$$f(x) = (6x - 2)^2 \sin(12x - 4) \tag{36}$$

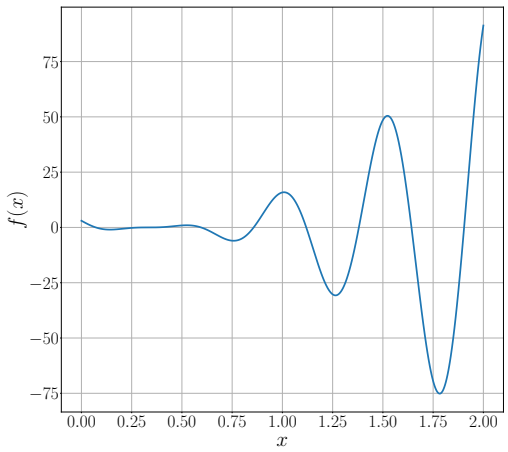


Fig. 25 Function with two distinct profiles

J. Function with constrained profile

The function is one dimensional with input $x \in [0, 1]$. The profile is constrained with the minimum and maximum function value being 0 and 1 respectively.

$$f(x) = \frac{1}{1 + (10x)^4} + \frac{1}{2} e^{-100\left(x - \frac{1}{2}\right)^2} \tag{37}$$

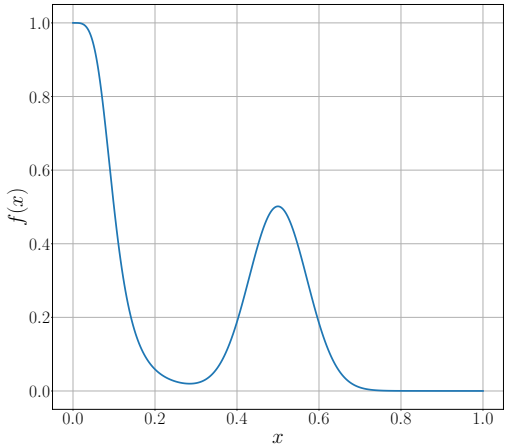


Fig. 26 Function with constrained profile

K. Function with symmetric and constrained profile

The function is one dimensional algebraic function. The profile is both symmetric and constrained. The function is evaluated in the domain: $x \in [0, 1]$.

$$f(x) = \frac{1}{100} + \frac{5}{8}(2x - 1)^4 \left((2x - 1)^2 + 4sin^2 5\pi x\right) \tag{38}$$

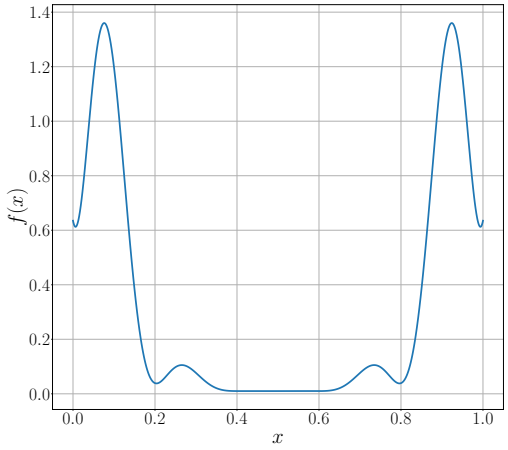


Fig. 27 Function with constrained profile