Bayesian Optimization using Deep Gaussian Processes

Ali Hebbal, Loic Brevault, Mathieu Balesdent, El-Ghazali Talbi, Nouredine Melab

BAYESIAN OPTIMIZATION USING DEEP GAUSSIAN PROCESSES

Ali Hebbal *

ONERA, DTIS, Universit Paris Saclay, Université de Lille, CNRS/CRIStAL, Inria Lille

Loic Brevault † and **Mathieu Balesdent** ‡

ONERA, DTIS, Université Paris Saclay, F-91123 Palaiseau Cedex, France

El-Ghazali Talbi § and **Nouredine Melab** ¶

Université de Lille, CNRS/CRIStAL, Inria Lille, Villeneuve d'Ascq, France

ABSTRACT

Bayesian Optimization using Gaussian Processes is a popular approach to deal with the optimization of expensive black-box functions. However, because of the *a priori* on the stationarity of the covariance matrix of classic Gaussian Processes, this method may not be adapted for non-stationary functions involved in the optimization problem. To overcome this issue, a new Bayesian Optimization approach is proposed. It is based on Deep Gaussian Processes as surrogate models instead of classic Gaussian Processes. This modeling technique increases the power of representation to capture the non-stationarity by simply considering a functional composition of stationary Gaussian Processes, providing a multiple layer structure. This paper proposes a new algorithm for Global Optimization by coupling Deep Gaussian Processes and Bayesian Optimization. The specificities of this optimization method are discussed and highlighted with academic test cases. The performance of the proposed algorithm is assessed on analytical test cases and an aerospace design optimization problem and compared to the state-of-the-art stationary and non-stationary Bayesian Optimization approaches.

***K*eywords** Bayesian Optimization · Gaussian Process · Deep Gaussian Process · Non-stationary function · Global Optimization

*Ph.D. student, ONERA, DTIS, Université Paris Saclay, Université de Lille, CNRS/CRIStAL, Inria Lille, ali.hebbal@onera.fr

†Research Engineer, ONERA, DTIS, Université Paris Saclay, loic.brevault@onera.fr

‡Research Engineer, ONERA, DTIS, Université Paris Saclay, mathieu.balesdent@onera.fr

§Professor at Polytech'Lille - Université de Lille, el-ghazali.talbi@univ-lille1.fr

¶Professor at Polytech'Lille - Université de Lille, nouredine.melab@univ-lille1.fr

Notations:

- A scalar is represented by a lower case character: $y \in \mathbb{R}$
- A vector is represented by a bold character: $\mathbf{x} \in \mathbb{R}^d, \mathbf{x} = [x_1, \ldots, x_d]^\top$
- A matrix is represented by upper case character: $X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,d} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & \cdots & x_{i,j} & \cdots & x_{i,d} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,j} & \cdots & x_{n,d} \end{bmatrix} \in \mathcal{M}_{n,d}$
- The i^{th} row of a matrix X is noted $\mathbf{x}^{(i)\top}$
- The j^{th} column of a matrix X is noted $\mathbf{x}_j$

1 Introduction

The design of complex systems often involves computationally intensive simulation codes that provide black-box functions as objectives and/or constraints. For example, for multidisciplinary design optimization problems, disciplinary codes are often modeled as black-box functions and the optimization process requires an iterative loop between these disciplines, inducing a computational burden. Within the context of black-box functions relying on legacy codes that do not provide analytical forms of the functions or the gradients, the use of exact optimization approaches is often not tractable. Furthermore, the high computational cost makes the use of algorithms that needs a consequent number of evaluations (gradient approximation, evolutionary algorithms, *etc.*) not suitable. Moreover, the objective and constraints functions involved often have non-linear landscape with multiple local optima, hence, complicating more the optimization problem.

One popular way to deal with expensive black-box function optimization is Bayesian Optimization (BO). BO algorithms are iterative sampling procedures based on surrogate models. To avoid running excessively the expensive functions, surrogate models allow the emulation of the statistical relationship between the design variables and the responses (objective function and constraints) given a dataset also called Design of Experiments (DoE). Different surrogate models can be used in design optimization [1]. One of the most popular Bayesian Optimization algorithms is "Efficient Global Optimization" (EGO) developed by Jones *et al.* [2]. It is based on Gaussian Process (GP) regression [3] (also called Kriging). The main advantage of GP is that in addition to the prediction, it provides an uncertainty estimation of the surrogate model response. Based on these two outputs, infill criteria are constructed to iteratively add the most promising candidates to the dataset. These points are then evaluated on the exact functions and the surrogate models are updated and so on, until a stopping criterion is satisfied.

Classical GP regression is based on stationary covariance functions $k(\cdot, \cdot)$, *i.e.* the covariance function is invariant by translation: $\forall \mathbf{x}, \mathbf{x}', \mathbf{h} \in \mathbb{R}^d, k(\mathbf{x}+\mathbf{h}, \mathbf{x'}+\mathbf{h}) = k(\mathbf{x}, \mathbf{x'})$ [4]. Hence, the covariance depends only on the shift $\mathbf{x} - \mathbf{x'}$. This induces a uniform level of smoothness in the prediction over the design space. This stationary prior covariance is effective to deal with stationary functions, however, it induces a major issue when the functions to be approximated are not stationary *i.e.* the level of variability of the response changes dramatically from one region of the input space to another.

The question of non-stationarity is discussed in different fields of research. In climate science due to dramatic changes in precipitation, the stationarity assumption is dropped for modeling climate phenomena [5] [6] [7]. In signal processing and finance among other fields, non-stationary models are often used to fit time series over a long period of time [8]. Also in geostatistics non-stationarity occurs when dealing with a region with different landscapes and topographic features [9].

In design optimization, due to the abrupt change of a physical property, the objective functions or the constraints may vary with different degrees of smoothness from one region of the design space to another. Specifically, aerospace design engineering involves different disciplines which can induce non-stationary processes. For example in aerodynamics, Computational Fluid Dynamics (CFD) problems often have different specific flow regimes due to separation zones, circulating flows, vortex bursts, transitions from subsonic to transonic, supersonic and hypersonic flow regimes.

In the propulsion discipline, the combustion involves irreversible thermodynamics transformations, that are characterized by sudden and rapid changes (*e.g.*, sudden state change of the matter, spontaneous chemical reactions, spontaneous mixing of matter of different states). There can also be non-stationarities in the structure discipline, for example in the stress-strain curve of a material, the elastic region, the strain hardening region and the necking region present different behaviors.

To overcome this issue, different approaches have been developed to deal with non-stationarity in regression such as regression trees [10], neural networks [11], wavelet process modeling [12]. GP regression has also been adapted to non-stationary cases. Indeed, one can summarize the different approaches for non stationary GP regression in three main categories: direct formulation of a non-stationary covariance function [13] [14], local stationary covariance functions [15] [16] and input-space warping approaches [17] [18]. However, these approaches may have some limitations when dealing with BO problems where data is scarse, or in high dimensional problems (Section 2).

Another approach which is not among these classic methods to handle non-stationarity consists in using Deep Gaussian Processes (DGPs) [19]. DGPs correspond to a functionnal composition of GPs, which may allow the description of more complex functions than standard GPs. The key contribution of this paper is to define a BO & DGP algorithm making the coupling of DGPs and BO possible to optimize problems involving non-stationary functions. The performance of this algorithm is assessed on different analytical test functions and on an aerospace optimization design problem.

The following paper is organized in three main sections. Section 2 provides a review of literature on the different concepts used along the paper. BO with a focus on GP and infill criteria is described. Then, the different non-stationary approaches for GP are presented. Finally, the DGP modeling approach is introduced, with a review on some inference approaches used for its training. In Section 3, the coupling of BO and DGPs is discussed. This discussion covers several aspects, such as the training approach of DGP in the context of BO, uncertainty quantification, architecture of the DGP and infill criteria, in order to define the BO & DGP algorithm, Deep Efficient Global Optimization (DEGO). Section 4, presents experimentations on analytical optimization test problems and on an aerospace optimization test problem, to assess the performance of DEGO compared to traditional existing approaches.

2 State of the art

2.1 Bayesian Optimization

The context of expensive black-box optimization problems is considered throughout this paper meaning that the objective function $f^{exact} : \mathbb{X} \subseteq \mathbb{R}^d \to \mathbb{R}$ and the n_c constraint functions $g_i^{exact} : \mathbb{X} \subseteq \mathbb{R}^d \to \mathbb{R}, 1 \leq i \leq n_c$ are computationally expensive and black-box functions. Without loss of generality the following minimization problem ($\mathscr{P}$) is considered:

$$(\mathscr{P}) \left| \begin{array}{lll} \underset{\mathbf{x}}{\text{Minimize}} & y & = f^{exact}(\mathbf{x}) \\ \text{subject to} & g_i^{exact}(\mathbf{x}) & \leq 0,\ \forall i \in \{1, \ldots, n_c\} \end{array} \right. \tag{1}$$

The expensive and black-box aspects of the considered functions limit their evaluations to a sparse data-set (Design of Experiments):

$$(DoE) \left\{ \begin{array}{rcl} X & = & \left[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right]^\top, \mathbf{x}^{(i)} \in \mathbb{R}^d, \forall i \in \{1, \ldots, n\} \\ \mathbf{y} & = & \left[y^{(1)} = f^{exact}\left(\mathbf{x}^{(1)}\right), \ldots, y^{(n)} = f^{exact}\left(\mathbf{x}^{(n)}\right)\right]^\top \\ \mathbf{c}_i & = & \left[c_i^{(1)} = g_i^{exact}\left(\mathbf{x}^{(1)}\right), \ldots, c_i^{(n)} = g_i^{exact}\left(\mathbf{x}^{(n)}\right)\right]^\top, \forall i \in \{1, \ldots, n_c\} \end{array} \right.$$

where n is the size of the dataset.

BO algorithms are sequential design algorithms. The design space is filled sequentially with new candidates with the objective to improve the current minimum in the DoE:

$$y_{\min} = \min \left\{ f^{exact}(\mathbf{x}^{(i)}) | i \in \{1, \ldots, n\} \text{ and } \forall j \in \{1, \ldots, n_c\}, g_j^{exact}(\mathbf{x}^{(i)}) \leq 0 \right\} \tag{2}$$

This sequential aspect of the BO algorithms consists of two iterative operations. The first one is the modeling of the expensive black-box functions ($f^{exact}, g_1^{exact}, \ldots, g_{n_c}^{exact}$) involved in the optimization problem using the DoE X and the corresponding exact evaluations $\mathbf{y}, \mathbf{c}_1, \ldots, \mathbf{c}_{n_c}$ together with a surrogate modeling approach (Random Forest, Polynomial models, Gaussian Process, Neural Networks, *etc.*) to obtain approximations ($\hat{f}, \hat{g}_1, \ldots, \hat{g}_{n_c}$). These latter are cheaper to evaluate, which allows a greater number of evaluations than the exact functions.

The second procedure consists of the determination of the most promising candidate to add to the current DoE in order to improve the current minimum y_{min} using the information given by the surrogate models. This is achieved by

optimizing an acquisition function (also called infill sampling criterion) on the design space, which is a performance measure of a candidate's potential from a minimization point of view. Once the most promising point is added to the dataset, it is evaluated on the exact expensive functions and the surrogate models are updated, and so on until a stopping criterion is reached (Fig. 1). Hence the two key aspects in BO algorithms are the surrogate model and the infill sampling criterion.

One of the most popular BO algorithms is the Efficient Global Optimization (EGO) [2]. It uses GP as surrogate model and the Expected Improvement as the infill sampling criterion. This work is focused on BO algorithms using GP and its variants as surrogate models.

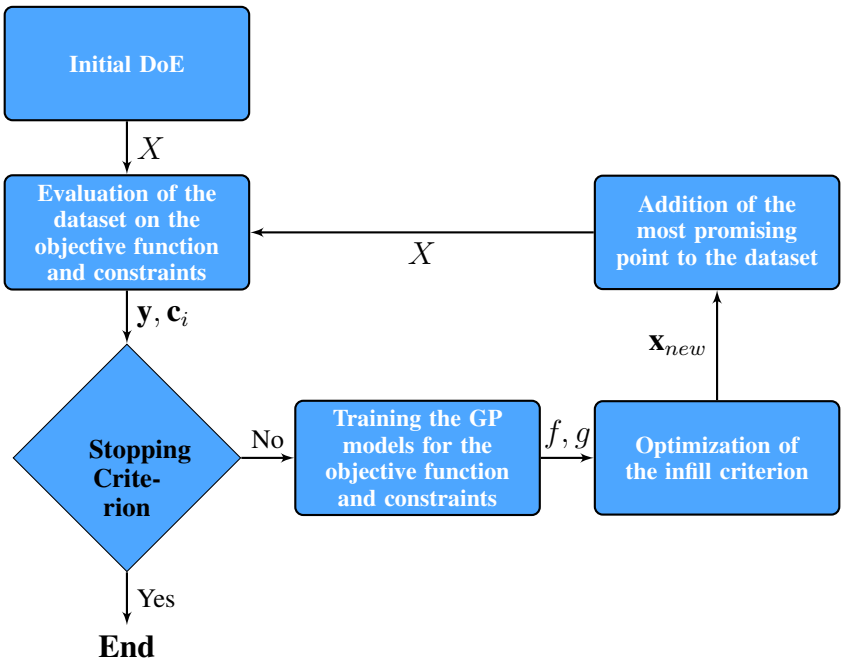


Figure 1: Bayesian Optimization with Gaussian Process framework

2.1.1 Gaussian Process

A Gaussian Process [3] f is a stochastic process indexed by a set $\mathbb{X} \subseteq \mathbb{R}^d$: $\{f(\mathbf{x}) : \mathbf{x} \in \mathbb{X}\}$ such as any finite number of random variables of the process has a joint Gaussian distribution:

$$\forall n' \in \mathbb{N}^*, \forall X' = \left[\mathbf{x}'^{(1)}, \ldots, \mathbf{x}'^{(n')}\right]^\top, f(X') \sim \mathcal{N}\left(\mu\left(X'\right), k\left(X', X'\right)\right) \tag{3}$$

with $f(X') = \left[f\left(\mathbf{x}'^{(1)}\right), \ldots, f\left(\mathbf{x}'^{(n')}\right)\right]^\top$. A GP, is completely defined by its mean and covariance functions and is noted $f(\cdot) \sim \mathcal{GP}\left(\mu(\cdot), k(\cdot,\cdot)\right)$, with $\mu(\cdot)$ the mean function and $k(\cdot,\cdot)$ the covariance function.

These properties are used to build GP regression models. Given a DoE X and its associated response values $\mathbf{y}$, this surrogate model may be used to predict in new locations X^* the output values $\mathbf{y}^*$. To do so, a GP prior is considered $f(\cdot) \sim \mathcal{GP}\left(\mu(\cdot), k^{\mathbf{\Theta}}(\cdot,\cdot)\right)$. The prior mean function $\mu(\cdot)$ can take a form that describes the trend of the approximated function if it is known (universal Kriging) otherwise a constant mean function μ may be considered (ordinary Kriging). The prior covariance function $k^{\mathbf{\Theta}}$ represents the prior belief of the unknown function to be modeled (*e.g.* smoothness, periodicity, stationarity, separability) (see Figure 2). The prior covariance function depends on a number of hyper-parameters $\mathbf{\Theta}$ allowing a better fit on the data. When dealing with noisy observations, $\mathbf{y}$ are not the exact values of the unknown function in X but a noisy version. One way to deal with this case is to introduce an identically distributed Gaussian noise with variance σ^2 such as the relationship between the latent (non-observed) function values $\mathbf{f} = f(X)$ and the observed response $\mathbf{y}$ is given by: $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{f}, \sigma^2 I_n)$ where I_n is the identity matrix of size n. Hence, the prior covariance on the noisy observations becomes: $k^{\mathbf{\Theta}}_{noisy}(X, X) = k^{\mathbf{\Theta}}(X, X) + \sigma^2 I_n$.

Training the GP consists in finding the optimal values of the hyper-parameters $\mathbf{\Theta}$, σ and μ (for ordinary kriging). These values are obtained by a standard maximum likelihood procedure:

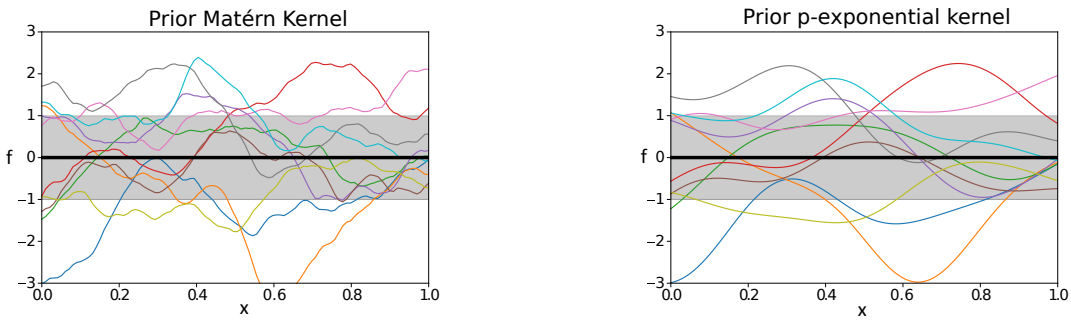


Figure 2: One-dimensional samples from the prior of a $3/2$ Matérn Kernel (left) and a p-exponential kernel (right)

$$\begin{cases} \text{Maximize} & : & p(\mathbf{y}|X) = \mathcal{N}(\mathbf{1}\mu, k^{\mathbf{\Theta}}(X,X) + \sigma^2 I_n) \\ \text{According to} & : & \mathbf{\Theta}, \sigma, \mu \end{cases} \tag{4}$$

Once the hyper-parameters optimized $\hat{\mathbf{\Theta}}, \hat{\sigma}, \hat{\mu}$, the prediction in new locations $X^* = \left[\mathbf{x}^{*(1)}, \ldots, \mathbf{x}^{*(n^*)}\right]^\top$ is done in two steps. Firstly, using the property of a GP, Eq.(3), the joint distribution of the predicted outputs $\mathbf{f}^* = f(X^*)$ and the observed outputs $\mathbf{y}$ is given by:

$$\begin{pmatrix} \mathbf{y} \\ \mathbf{f}^* \end{pmatrix} \sim \mathcal{N}\left(\mathbf{1}\hat{\mu}, \begin{matrix} k^{\hat{\mathbf{\Theta}}}(X,X) + \hat{\sigma}^2 I_n & , k^{\hat{\mathbf{\Theta}}}(X,X^*) \\ k^{\hat{\mathbf{\Theta}}}(X^*,X) & , k^{\hat{\mathbf{\Theta}}}(X^*,X^*) \end{matrix}\right) \tag{5}$$

Then, by using the conditional distribution of a joint Gaussian distribution, which is equivalent here to conditioning the prior distribution on the observations $\mathbf{y}$, the posterior distribution is obtained:

$$\mathbf{f}^*|X^*, \mathbf{y}, X \sim \mathcal{N}\left(\hat{f}(X^*), \hat{\Sigma}(X^*)\right) \tag{6}$$

where $\hat{f}(X^*)$ and $\hat{\Sigma}(X^*)$ are respectively the mean and the covariance of the posterior distribution and are defined as:

$$\hat{f}(X^*) = \mathbf{1}\hat{\mu} + k^{\hat{\mathbf{\Theta}}}(X^*,X)\left(k^{\hat{\mathbf{\Theta}}}(X,X) + \hat{\sigma}^2 I_n\right)^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}) \tag{7}$$

$$\hat{\Sigma}(X^*) = k^{\hat{\mathbf{\Theta}}}(X^*,X^*) - k^{\hat{\mathbf{\Theta}}}(X^*,X)\left(k^{\hat{\mathbf{\Theta}}}(X,X) + \hat{\sigma}^2 I_n\right)^{-1} k^{\hat{\mathbf{\Theta}}}(X,X^*) \tag{8}$$

Notice that in the special case of a single test point $\mathbf{x}^*$, the posterior distribution comes back to a univariate Gaussian distribution: $f^*|\mathbf{x}^*, \mathbf{y}, X \sim \mathcal{N}\left(\hat{f}(\mathbf{x}^*), \hat{s}^2(\mathbf{x}^*)\right)$, where $\hat{f}(\mathbf{x}^*)$ is the mean prediction on $\mathbf{x}^*$ and $\hat{s}$ its associated standard deviation. Obtaining a Gaussian posterior distribution gives along with the prediction, an uncertainty measure as a Gaussian error which is useful in the construction of infill criteria for Bayesian Optimization.

The steps of GP regression are summarized in (Figure 3). Henceforth, for notation simplifications, the dependence of the prior covariance function on $\mathbf{\Theta}$ is dropped, and $k(X,Z)$ is written $K_{X,Z}$. Moreover, the prior GP is considered with a zero mean function $\mu = 0$.

2.1.2 Infill Criteria

For selecting infill sample candidates, a variety of criteria has been developed [20]. Each criterion performs a trade-off between exploration i.e. investigating regions where the variance is large and exploitation i.e. investigating regions where the prediction is minimized. The Probability of Improvement (PI) criterion samples the location where the probability of improving the current minimum $y_{\min}$ (*cf.* Eq.(2)) is maximized.

$$PI(\mathbf{x}) = \mathbb{P}\left[f(\mathbf{x}) \leq y_{\min}\right] = \Phi\left(\frac{y_{\min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) \tag{9}$$

where $\Phi(\cdot)$ is the Cumulative Distribution Function (CDF)of the univariate Gaussian probability distribution. The higher values of $PI(\mathbf{x})$, the higher chances that $\hat{f}(\mathbf{x})$ is better than $y_{\min}$. The drawback of this criterion is that only

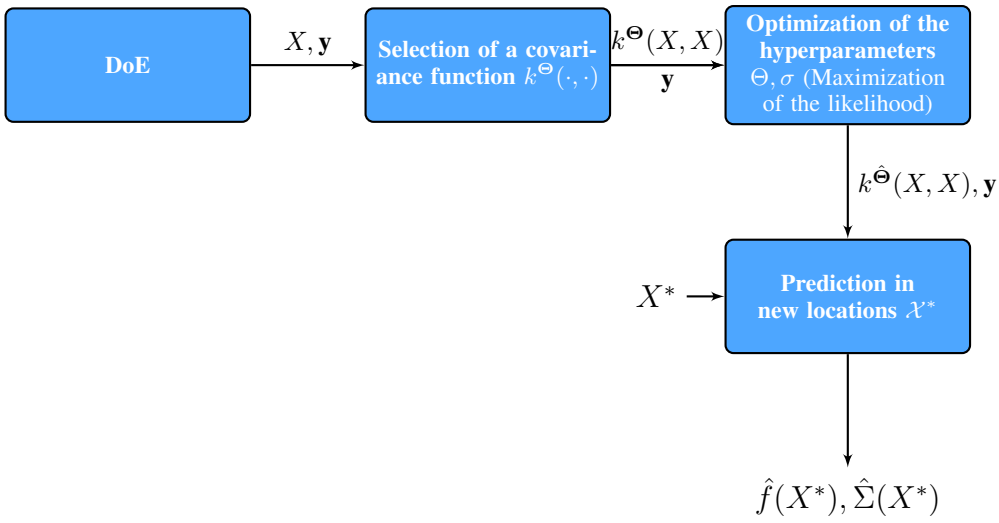


Figure 3: Gaussian Process Regression framework

the probability is taken into account and not how much a point may improve the current minimum value. This will add a large number of points around the current best point. The Expected Improvement (EI) overcomes this issue by taking into account the improvement induced by a candidate that is defined as: $I(\mathbf{x}) = \max\{0, y_{\min} - f(\mathbf{x})\}$. EI is then computed as the expectation taken with respect to the posterior distribution:

$$
\begin{aligned}
EI(\mathbf{x}) &= \mathbb{E}\left[I(\mathbf{x})\right] && (10)\\
&= \int_{\mathbb{R}} \max\{0, y_{\min} - t\} p(t|\mathbf{x}, X, \mathbf{y}) \mathrm{d}t && (11)\\
&= (y_{\min} - \hat{f}(\mathbf{x}))\Phi\left(\frac{y_{\min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) + \hat{s}(\mathbf{x})\phi\left(\frac{y_{\min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) && (12)
\end{aligned}
$$

where $\phi(\cdot)$ denotes the Probability Density function (PDF) of the univariate Gaussian probability distribution. Two important terms constitute the EI formula. The first part is the same as in PI, but multiplied by a factor that scales the EI value on the supposed improvement value. The second part expresses the uncertainty. It tends to be large when the uncertainty on the prediction is high. So, the EI is large for regions of improvement and also for regions of high uncertainty, allowing global refinement properties. The maximization of the EI can be performed using multi-start of gradient-based optimization algorithms, Monte-Carlo simulations or evolutionary algorithms [21].

Thompson sampling has also been adopted as an infill criterion [22]. It consists in drawing a sample from the posterior distribution and choosing the index of the minimum of this sample as an infill candidate. Other methods can also be mentioned as confidence bound criteria [23] or information theory based infill criteria [24]. Recently, portfolio methods combining between these different infill criteria have been developed [25] [26]. This multitude of methods shows that there is no single infill criterion that performs better over all problem instances.

To handle constraints in BO, different techniques are used [27] [28]. The direct method [29] which consists in the optimization of the unconstrained infill criterion under approximated constraints. The Expected Violation strategy [30] which considers the optimization of the unconstrained infill criterion under the constraint of an expected violation inferior to a threshold. The Probability of Feasibility approach [31] which optimizes the product of an unconstrained infill criterion with the probability of feasibility of the constraints.

2.2 Non-stationary approaches

Standard GP regression is based on the *a priori* that the variation in the output depends only on the variation in the design space and not in the region considered. This is induced by the use of stationary covariance functions as *a priori* $\forall \mathbf{x}, \mathbf{x}', \mathbf{h} \in \mathbb{R}^d, k(\mathbf{x}+\mathbf{h}, \mathbf{x'}+\mathbf{h}) = k(\mathbf{x}, \mathbf{x'}) = k_*(\mathbf{x}-\mathbf{x}')$ where $k_*(\cdot)$ is a scalar function defined on $\mathbb{R}^d$. This *a priori* is generally valid for functions where there is no change in the smoothness of the function considered in the design space. However, this is not suitable for functions with drastic variations. For example, the modified Xiong function

(*cf.* Eq.(27) in Appendix A, Figure 4), has two regions with different level of variations. It presents one region where the function varies with a high frequency $x \in [0, 0.3]$ and the other where the function varies slowly $x \in [0.3, 1]$. This makes the GP regression not suitable for this function.

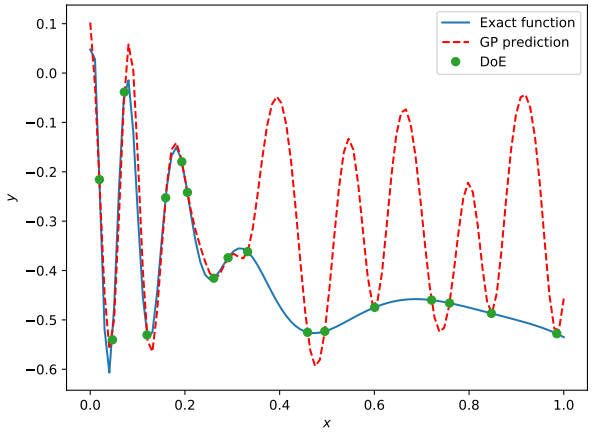


Figure 4: Approximation of the modified Xiong-function, a non-stationary 1-dimensional function by a standard GP model. The model can not capture the stability of the region $[0.4, 1]$ and continues to oscillate.

To overcome this issue different GP adaptations to non-stationarity have been proposed. These adaptations are presented in the present Section.

2.2.1 Direct formulation of non-stationary kernels

Most of the methods in literature that use a direct formulation of a non-stationary covariance function, follow the work of Higdon *et al.* [13]. The main idea is to use a convolution product of spatially-varying kernel functions to define a new class of kernels that is non-stationary:

$$k^{NS}(\mathbf{x}_i, \mathbf{x}_j) = \int_{\mathbb{R}^d} k^S(\mathbf{x}_i, \mathbf{u}) k^S(\mathbf{x}_j, \mathbf{u}) \mathrm{d}\mathbf{u} \tag{13}$$

where k^S is a stationary covariance function, $\mathbf{x}_i, \mathbf{x}_j$ are locations in $\mathbb{R}^d$. Higdon *et al.* [13] give an analytical form of the non-stationary covariance resulting from convolving Gaussian kernels. Paciorek [14] extends this approach by giving the analytical form of the non-stationary covariance function resulting from convolving any stationary kernels:

$$k^{NS}(\mathbf{x}_i, \mathbf{x}_j) = |\Sigma_i|^{\frac{1}{4}} |\Sigma_j|^{\frac{1}{4}} \left| \frac{\Sigma_i + \Sigma_j}{2} \right|^{-\frac{1}{2}} k_*^S(\sqrt{Q(\mathbf{x}_i, \mathbf{x}_j)}) \tag{14}$$

where:

$$Q(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T \left(\frac{\Sigma_i + \Sigma_j}{2} \right)^{-1} (\mathbf{x}_i - \mathbf{x}_j) \tag{15}$$

and $\Sigma_i = \Sigma(\mathbf{x}_i)$ is a $d \times d$ matrix-valued function which is positive definite for all $\mathbf{x}$ in $\mathbb{R}^d$. In the stationary case $\Sigma(\cdot)$ is a constant arbitrary matrix (often a diagonal matrix containing the lengthscales of each dimension). The interesting observation is that in the resulting non-stationary covariance function $k^{NS}(\cdot, \cdot)$, the Mahalanobis distance $\sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \Sigma^{-1} (\mathbf{x}_i - \mathbf{x}_j)}$ is not used in the stationary covariance function $k_*^S(\cdot)$. Instead, the square root of $Q(\mathbf{x}_i, \mathbf{x}_j)$ is used, that is a quadratic form with the average of the kernel matrices $\Sigma(\cdot)$ in the two locations. Paciorek [14] gives the special case of a non-stationary Matérn covariance function using Eq.(14). The construction of the kernel matrix $\Sigma(\cdot)$ for each $\mathbf{x}$ in the domain is performed *via* an eigendecomposition parametrization, which can be difficult when increasing the space dimension. Gibbs [32] proposes a simpler parametrization by choosing the matrix $\Sigma(\mathbf{x})$ as a diagonal matrix of lengthscales, hence, obtaining lengthscales depending on the location of $\mathbf{x}$. This class of approaches due to its high parametrization requirements (defining a kernel matrix for each location) may not be suitable for high-dimensional problems.

2.2.2 Local stationary covariance functions

The local stationary approaches are based on the idea that non-stationary functions have a locally stationary behavior. Haas [16] proposes a moving window approach where the training and prediction regions move along the input space using a stationary covariance function. This window has to be restrained enough, so that along this window the function is stationary. Other methods consist in dividing the input space into different subsets and using a different model for each subset, this is also known as mixture of experts. Rasmussen and Gharmani [15] propose a mixture of GP experts, that is different stationary GPs in different subspaces of the input space. The division of the input space is performed by a gating network. In this approach the learning dataset is also partitioned, meaning that each model is trained using the dataset in its own region. Using the same concept of local GPs, the Tree Gaussian Process approach of Gramacy [33] can be cited.

However, these approaches present some limitations. Indeed, in computationally expensive problems, data are sparse and using a local surrogate model with sparser data may be problematic, due to the poor prediction capability especially for high dimensional problems.

2.2.3 Warped GPs

These approaches first introduced by Sampson *et al.* [17], also called non-linear mapping, consist of a deformation of the input space in order to express the non-stationarity using a stationary covariance function. Specifically, a stationary covariance function $k^S(\cdot,\cdot)$, and a function $w(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^d$ are considered, then the non-stationary covariance function is obtained by simply combining $w(\cdot)$ and $k^S(\cdot,\cdot)$:

$$k^{NS}(\mathbf{x}_i, \mathbf{x}_j) = k^S(w(\mathbf{x}_i), w(\mathbf{x}_j)) \tag{16}$$

The difficult task in this class of approaches is the estimation of $w(\cdot)$. Gibbs approach that was cited in the direct formulation methods can also be obtained *via* non linear mapping. It consists in considering $w(\cdot)$ as a multidimensional integral of non-negative density functions. These density functions are defined as a weighted sum of l positive Gaussian radial basis functions. The drawback of this approach is the number of parameters equal to $l \times d$. Moreover, the number of radial basis functions l needed to capture the non-stationarity increases with the dimension of the space d, inducing an over-parametrized structure of the covariance function in high-dimensional situations. To overcome this issue, the non-linear mapping approach proposed by Xiong *et al.* uses a piece-wise linear density function with parametrized knots. Hence, reducing the number of parameters. However, the deformation in this approach is done only along canonical axes. Marmin *et al.* [34] adress this issue by introducing a parametrized matrix A allowing a linear mapping of the input space before undergoing the non-linear mapping of $w(\cdot)$. The non-linear mapping approach was studied in the context of BO in [35] where it was compared to regular GP. This allowed the authors to set up a new approach mixing regular GP with non-linear mapping when dealing with BO called Adaptive Partial Non-Stationary (APNS).

These approaches have some limitations when dealing with sparse data or relatively high-dimensional problems. To overcome these issues, another approach may be the use of Deep Gaussian Processes to handle non-stationarity.

2.3 Deep Gaussian Processes

2.3.1 Definition

The intuition behind using the concept of Deep Gaussian Process (DGP) for non-stationary functions, comes from the Deep Learning theory. The basic idea is to capture multiple variations through the composition of multiple functions. Hence, learning highly varying functions using composition of simpler ones [36].
Following this intuition, Damianou and Lawrence [19] developed DGP, a nested structure of GPs considering the relationship between the inputs and the final output as a functional composition of GPs (Figure 5):

$$y = f_{L-1}(\dots \mathbf{f}_l(\dots(\mathbf{f}_1(\mathbf{f}_0(\mathbf{x}))))) + \epsilon \tag{17}$$

where L is the number of layers and $\mathbf{f}_l(\cdot)$ is an intermediate GP. Each layer l is composed of an input node H_l, an output node H_{l+1} and a multi-output GP $\mathbf{f}_l(\cdot)$ mapping between the two nodes, getting the recursive equation: $H_{l+1} = \mathbf{f}_l(H_l)$. A Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is introduced such as $\mathbf{y} = f_{L-1}(H_{L-1}) + \epsilon$. The one column matrix $H_L = f_{L-1}(H_{L-1})$ refers to the noise free version of $\mathbf{y}$. An exploded view showing the multidimensional aspect of DGPs is illustrated in Figure 6.

This hierarchical composition of GPs presents better results than regular GPs in the approximation of complex functions [19] [37] [38] (see Figure 7). In fact, DGP allows a flexible way of kernel construction through input warping and dimensionality expansion to better fit data.

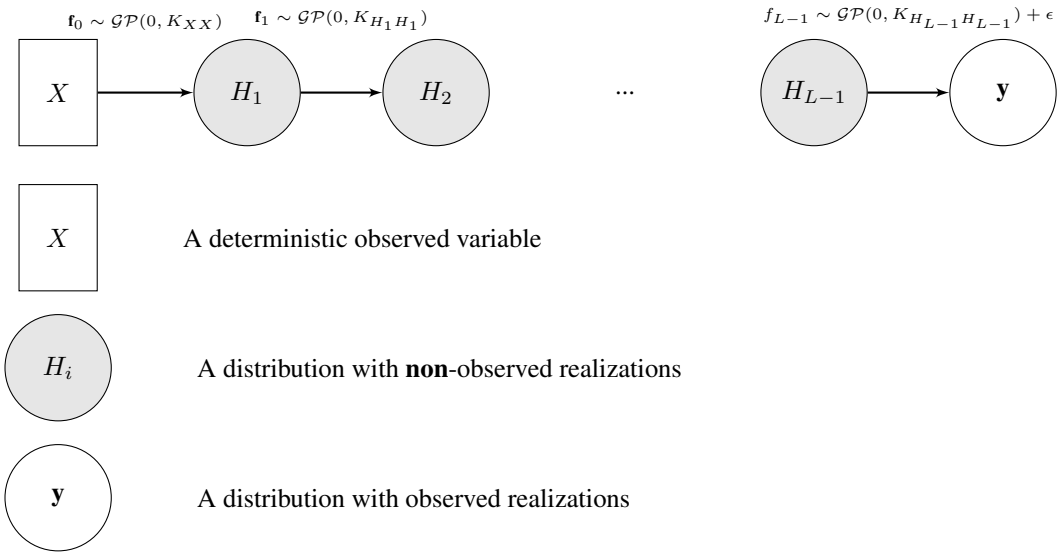


Figure 5: A representation of the structure of a DGP

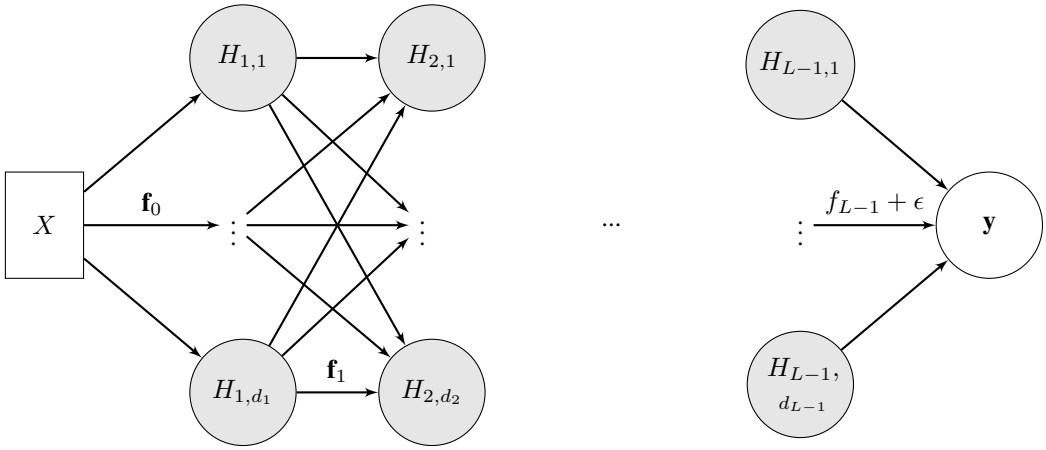


Figure 6: An exploded view of the structure of a DGP

In GP regression models, the hyper-parameters involved are the kernel parameters, the mean function parameters and the noise variance. The optimization of these hyper-parameters in the training of GPs is analytically tractable for a Gaussian noise variance. In DGPs, in addition to these parameters considered for each layer, non-observable variables $H_1, \ldots, H_l, \ldots, H_L$ are involved. Hence, the marginal likelihood for DGP can be written as:

$$\begin{aligned} p\left(\mathbf{y}|X\right) &= \int_{H_1} \ldots \int_{H_l} \ldots \int_{H_L} p\left(\mathbf{y}, H_1, \ldots, H_l, \ldots, H_L|X\right) \mathrm{d}H_1 \ldots \mathrm{d}H_l \ldots \mathrm{d}H_L \\ &= \int_{\{H_l\}_1^L} p\left(\mathbf{y}, \{\mathbf{h}_l\}_1^L|X\right) \mathrm{d}\{H_l\}_1^L \\ &= \int_{\{H_l\}_1^L} p(\mathbf{y}|H_L)p(H_L|H_{L-1}) \ldots p(H_1|X)\mathrm{d}\{H_l\}_1^L \end{aligned} \tag{18}$$

where $\{H_l\}_1^L$ is the set of non-observable (latent) variables $\{H_1, \ldots, H_L\}$.
The computation of this marginal likelihood is not analytically tractable. Indeed, $p(H_{l+1}|H_l)$ non-linearly involves the inverse of the covariance matrix $K_{H_l H_l}$, which makes the integration of the conditional probability $p(H_{l+1}|H_l)$ with respect to H_l analytically not tractable.

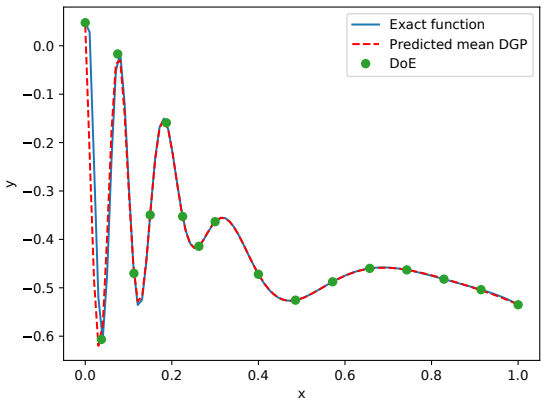


Figure 7: Approximation of the modified Xiong-function, a non-stationary 1-dimensional function by a 2 layers DGP model. The model captures the non-stationarity of the real function.

To overcome this issue the marginal likelihood is approximated using an approximate inference technique. Several approaches such as variational inference [19], Expectation propagation [39], Markov chain Monte Carlo techniques [38] [40] have been developed and are briefly discussed in this Section.

2.3.2 Direct variational inference approach

2.3.2.1 A Variational Bayes approach. Let Y and Z be respectively a set of observed and latent variables. Variational Bayes methods consist in approximating the posterior distributions of the set of latent variables $p(Z|Y)$ by a variational distribution $q(Z)$ belonging to a family of distributions which has a simple form (usually the exponential family). By marginalizing over the latent variables, $p(Y)$ can be rewritten as:

$$\begin{aligned} p(Y) &= \int_Z p(Y,Z)\mathrm{d}Z \\ &= \int_Z p(Y,Z)\frac{q(Z)}{q(Z)}\mathrm{d}Z \\ &= \mathbb{E}_q\left[\frac{p(Y,Z)}{q(Z)}\right] \end{aligned} \tag{19}$$

Since the log function is a concave function using Jensen's inequality on the expectation gives:

$$\begin{aligned} \log p(Y) &= \log\left(\mathbb{E}_q\left[\frac{p(Y,Z)}{q(Z)}\right]\right) \\ &\geq \mathbb{E}_q\left[\log\left(\frac{p(Y,Z)}{q(Z)}\right)\right] \end{aligned} \tag{20}$$

This bound obtained is called the Evidence Lower Bound (ELBO). Then, the ELBO is maximized in order to obtain a tight bound on the likelihood $p(\mathbf{Y})$. It can be shown that maximizing the ELBO is equivalent to minimizing the Kulback-Liebler (KL) divergence between $q(Z)$ and $p(Z|Y)$ [41].

Therefore, by introducing a variational distribution $q\left(\{H_l\}_1^L\right)$ in the case of DGPs and by directly applying the previous inequality Eq.(20), the following result is obtained:

$$\begin{aligned} \log p(\mathbf{y}|X) &\geq \mathbb{E}_q\left[\log\left(\frac{p(\mathbf{y},\{H_l\}_1^L|X)}{q\left(\{H_l\}_1^L\right)}\right)\right] \\ &\geq \mathbb{E}_q\left[\log\left(p\left(\mathbf{y}|\{H_l\}_1^L,X\right)\right)\right] + \mathbb{E}_q\left[\log\left(\frac{p\left(\{H_l\}_1^L|X\right)}{q\left(\{H_l\}_1^L\right)}\right)\right] \\ &\geq \mathbb{E}_q\left[\log\left(p\left(\mathbf{y}|\{H_l\}_1^L,X\right)\right)\right] - KL\left(q\left(\{H_l\}_1^L\right)||p\left(\{H_l\}_1^L|X\right)\right) \end{aligned} \tag{21}$$

The second term in Eq.(21) is the KL divergence between the variational distribution and the prior distribution on the latent variables. The KL divergence is analytically tractable if the prior and the variational distributions on the latent variables are restrained to Gaussian distributions. However, the first term is still analytically intractable since it still involves the integration of the inverse of the covariance matrices with respect to the latent variables. To overcome this issue, Damianou *et al.* [41] followed the work of Titsias and Lawrence [42] in the context of Bayesian Gaussian Process Latent Variable Model by introducing a set of inducing variables to obtain an analytical tractable lower bound.

2.3.2.2 Introduction of inducing variables. Inducing variables were first introduced in the context of sparse GP [43] [44]. To overcome the drawback of regular GP that involves the inversion of the covariance matrix of the whole dataset $K_{XX} \in \mathcal{M}_{nn}$, sparse GPs introduce a set of latent variables consisting of an input-output pairs Z and $\mathbf{u}$:

$$\text{(Induced variables)} \begin{cases} Z &= \left[\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\right]^\top, \mathbf{z}^{(i)} \in \mathbb{R}^d, \forall i \in \{1, \dots, m\} \\ \mathbf{u} &= \left[u^{(1)} = f\left(\mathbf{z}^{(1)}\right), \dots, u^{(m)} = f\left(\mathbf{z}^{(m)}\right)\right]^\top \end{cases}$$

with $m << n$. The idea is to choose Z and $\mathbf{u}$ in order to explain the statistical relationship between X and $\mathbf{y}$ by the statistical relationship between Z and $\mathbf{u}$. Then, for the training and prediction of the sparse GP, the matrix to be inverted belongs to $\mathcal{M}_{mm}$, hence, achieving reduction in the computational complexity of GP.

In each layer of a DGP, a set of inducing variables is introduced $Z_l = \left[\mathbf{z}_l^{(1)}, \dots, \mathbf{z}_l^{(m_l)}\right]^\top$, $\mathbf{z}^{(i)} \in \mathbb{R}^{d_l}, \forall i \in \{1, \dots, m_l\}$ and $U_l = \mathbf{f}_l(Z_l)$ (Figure 8) (notice here that since the intermediate layers are multi-output GPs, U_l are matrices $\in \mathcal{M}_{nd_l}$ and not vectors, except in the last layer where U_L corresponds to a one column matrix). Henceforth, for notation simplicity, the number of induced inputs in each layer are considered equal to m.

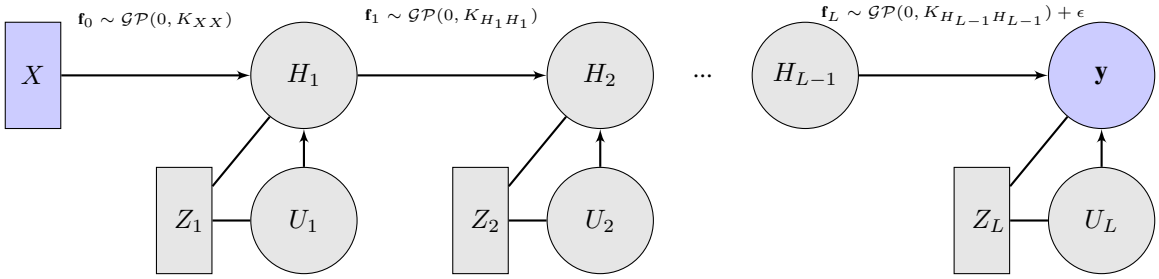


Figure 8: Representation of the introduction of the inducing variables in DGPs

Now that the latent space has been augmented with the inducing variables, the posterior of the joint distribution of the latent variables $p\left(\{H_l, U_l\}|\mathbf{y}, X\right)$ is approximated by a variational distribution $q\left(\{H_l, U_l\}\right)$ with the assumption of independency between layers:

$$q\left(\{H_l, U_l\}_1^L\right) = \prod_{l=1}^{L} q(H_l) q(U_l) \tag{22}$$

Moreover, for the sake of simplicity the variational distributions are restrained to the exponential family. Then, by using Eq.(20) it holds:

$$\log p(\mathbf{y}|X) \quad \geq \quad \mathbb{E}_{q(\{H_l\}_1^L, \{U_l\}_1^L)} \left[\log \frac{p\left(\mathbf{y}, \{H_l\}_1^L, \{U_l\}_1^L | X, \{Z_l\}_1^L\right)}{q(\{H_l\}_1^L, \{U_l\}_1^L)}\right] = \mathcal{L} \tag{23}$$

After using some results from variational sparse GP [44], an analytical tractable bound is obtained for kernels that are feasibly convoluted with the Gaussian density such as the Automatic Relevance Determination (ARD) exponential kernel. The analytical optimal form of $q(U_l)$ as a function of $q(H_l)$ can be obtained *via* the derivative of the variational lower bound $\mathcal{L}$ with respect to $q(U_l)$. Hence, collapsing $q(U_l)$ in the approximation by injecting its optimal form, allows to obtain a tighter lower bound depending on the following parameters: the kernel parameters $\{\mathbf{\Theta}_l\}_{l=1}^{l=L}$, the inducing inputs $\{Z_l\}_{l=1}^{l=L}$ and the variational distributions parameters $\{\bar{H}_l, S_l\}_{l=1}^{l=L}$ respectively the mean and covariance matrix defining the variational distributions $\{q(H_l) \sim \mathcal{N}(\bar{H}_l, S_l)\}_{l=1}^{l=L}$

Therefore, training a DGP model comes back to maximizing the evidence lower bound with respect to these parameters:

$$\begin{array}{ll} \text{Maximize:} & \mathcal{L} \\ \text{According to:} & \{\boldsymbol{\Theta}_l\}_{l=1}^{l=L}, \{Z_l\}_{l=1}^{l=L}, \{\bar{H}_l\}_{l=1}^{l=L}, \{S_l\}_{l=1}^{l=L} \end{array}$$

The number of hyperparameters to optimize in the training of a DGP is more important than in the training of a regular GP where only the kernel hyperparameters are considered, and more important than variational sparse GP since the number of hyperparameters is considered for each layer.

2.3.3 The Doubly stochastic approach

The Doubly Stochastic approach proposed by Salimbeni *et al.* [38] drops the assumption of independence between layers and the special form of kernels. Indeed, the posterior approximation maintains the exact model conditioned on U_l:

$$q\left(\{H_l, U_l\}_1^L\right) = \prod_{l=1}^{L} p(H_l|H_{l-1}, U_l) q(U_l) \tag{24}$$

However, the analytical tractability of the lower bound $\mathcal{L}$ is not maintained. The variational lower bound is then rewritten as follows (the mention of the dependence on X and Z_l is omitted for the sake of simplicity):

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{q(\{H_l,U_l\}_1^L)}\left[\log \frac{p\left(\mathbf{y}, \{H_l\}_1^L, \{U_l\}_1^L\right)}{q(\{H_l\}_1^L, \{U_l\}_1^L)}\right] \\ &= \mathbb{E}_{q(\{H_l,U_l\}_1^L)}\left[\log \frac{p\left(\mathbf{y}|\{H_l\}_1^L, \{U_l\}_1^L\right) \prod_{l=1}^L p(H_l|H_{l-1}, U_l) p(U_l)}{\prod_{l=1}^L p(H_l|H_{l-1}, U_l) q(U_l)}\right] \\ &= \mathbb{E}_{q(\{H_l,U_l\}_1^L)}\left[\log \frac{\prod_{i=1}^N p(y^{(i)}|h_L^{(i)}) \prod_{l=1}^L p(U_l)}{\prod_{l=1}^L q(U_l)}\right] \\ \mathcal{L} &= \sum_{i=1}^{N} \mathbb{E}_{q(h_L^{(i)})}\left[\log p(y^{(i)}|h_L^{(i)})\right] - \sum_{l=1}^{L} KL\left[q(U_l)||p(U_l)\right] \end{aligned} \tag{25}$$

Keeping the $\{U_l\}_{l=1}^L$ in this formulation of the ELBO instead of collapsing then allows factorization over the data $X, \mathbf{y}$ which enables parallelization. The computation of this bound is done by approximating the expectation with Monte Carlo sampling, which is straightforward using the propagation of each data-point $\mathbf{x}^{(i)}$ through all the GPs:

$$q(h_L^{(i)}) = \int \prod_{l=1}^{L-1} q\left(\mathbf{h}_l^{(i)}|U_l, \mathbf{h}_{l-1}^{(i)}, Z_{l-1}\right) \mathrm{d}\mathbf{h}_l^{(i)} \tag{26}$$

with $\mathbf{h}_0^{(i)} = \mathbf{x}^{(i)}$. The optimization of this formulation of the bound is done according to the kernel parameters $\{\boldsymbol{\Theta}_l\}_{l=1}^{l=L}$, the inducing inputs $\{Z_l\}_{l=1}^{l=L}$ and the variational distribution hyperparameters of the inducing variables: $\{q(U_l) \sim \mathcal{N}(\bar{U}_l, \Sigma_l)\}_{l=1}^{l=L}$

2.3.4 Other approaches

Alternative methods for training a DGP have been proposed. Dai *et al.* [37] improved the direct variational approach by instead of considering the parameters of the variational posteriors $q(H_l)$ as individual parameters, considered them as a transformation of observed data $\mathbf{y}$ by multi-layers perceptron. Bui *et al.* [39] proposed a deterministic approximation for DGP based on an approximated Expectation Propagation energy function, and a probabilistic back-propagation algorithm for learning. Havasi *et al.* [40] gave up the Gaussian approximation of the posterior distribution $\{q(U_l)\}_{l=1}^L$, and proposed the use of Stochastic Gradient Hamiltonian Monte Carlo for this approximation. Moreover, a Markov Chain Expectation Maximization algorithm is developed for the optimization of the hyper-parameters.

3 Bayesien Optimization using Deep Gaussian Processes

In this Section, a deep investigation is followed in order to develop the Deep Efficient Global Optimization algorithm (DEGO) a BO & DGP algorithm. To lead to that, the choices needed to make this coupling possible are discussed. These choices concern the training approach for the DGP, the uncertainty model of the DGP prediction, the infill

criteria, the induced variables in each layer and the configuration of the architecture of the DGP (number of layers, number of units, *etc.*).

In this Section different functions are used to illustrate the analyses made. These functions are presented in Appendix A.

3.1 Training

In [45], in the experimentations a DGP is used with the variational auto-encoded inference method [37]. This approach of training assumes that the variational distributions of the latent variables are factorizable over the layers. As mentioned in the previous Section, this assumption may not be realistic. The doubly stochastic inference approach proposed in [38] is preferred in the present study since it keeps the dependence between layers. The loss of analytical tractability may be compromising, since a Monte Carlo sampling approach is required. However, the form of the Evidence Lower Bound (ELBO) is fully factorizable over the dataset allowing important parallelization.
The optimization of the ELBO (Eq.(25)) is performed using a loop procedure consisting of an optimization step with the natural gradient to perform the optimization with respect to the parameters of the variational distributions $\{q(U_l)\}_{l=1}^{L}$ while fixing the other variables, then an optimization step using a stochastic gradient descent optimizer (Adam optimizer [46]) to perform the optimization with respect to the hyperparameters. This optimization procedure has been done in the case of sparse variational GPs and has shown better results than using only a gradient descent optimizer for all the variables [47]. However, using the natural gradient for all the distributions of the inner layers in the case of DGPs is tricky. Indeed, the optimization of the distribution of the variational parameters of the first layers is highly multi-modal. It is therefore more likely to take over large step size (inducing non-positive definite matrix in the natural space). One way to deal with this issue is to use different step sizes for each layer decreasing from the last layer to the first one.
The evolution of the ELBO using three different optimization approaches is presented in Figure 9 for three different problems. The optimization using a loop with a natural gradient step for all the variational distributions parameters and a step with the Adam optimizer for the hyperparameters gives the best results. However, for the Hartmann 6d and the Trid functions the size of the step of the natural gradient for the first layers is reduced compared to the step size of the last layer, in order to avoid overlarge step size.

A test set to estimate the Root Mean Square Error (RMSE) and the test log-likelihood is used to assess the prediction and uncertainty performance of the models (Table 1). The models optimized by the loop natural gradient for the variational distributions parameters of all layers and an Adam optimizer for the hyperparameters, give the best results. It is interesting to notice that the models optimized by the Adam optimizer on all the variables give comparable results on the prediction. However, it happen that they underestimate the uncertainty on the prediction (Figure 10). This explains the value obtained of the test log-likelihood in the case of the DGP model optimized by only an Adam Optimizer compared to the ones given by GP or DGP with natural gradient on all the variational parameters. In the context of BO this uncertainty measure is important for the construction of infill criteria. An underestimated uncertainty will make the BO algorithm sampling around the current minimum limiting thereby its exploration capabilities. Hence, a combination of the natural gradient on all the variational parameters and the Adam optimizer on the hyperparameters is used in DEGO for training the models.

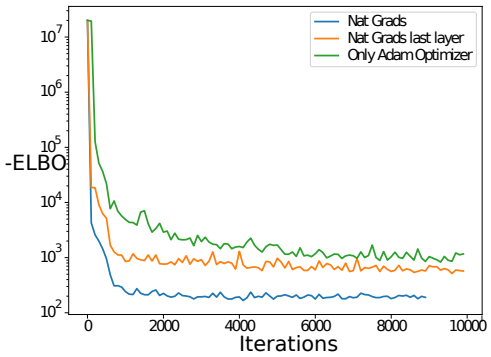
In the context of BO, this optimization procedure has to be repeated after each added point, which may be time consuming. To overcome this issue, the optimization is initialized using the optimal values of the previously trained DGP model. As shown in Figure 11 this allows faster convergence. However, this can make the optimization tricked in bad local optima. Therefore, a complete training of the model is performed after a certain number of iterations depending on the problem at hand.

The pseudo algorithm (Algorithm 1) describes the training of the DGP model. The initialization can be done from a previous model optimal parameter values to allow faster optimization or can be done from scratch.

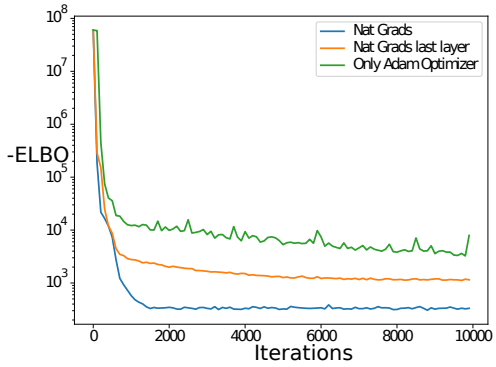
3.2 Architecture of the DGP

The architecture of the DGP is a key question when using a DGP in BO. The configuration of the architecture of the DGP includes the number of layers, the number of units in each layer and the number of induced variables in each layer.

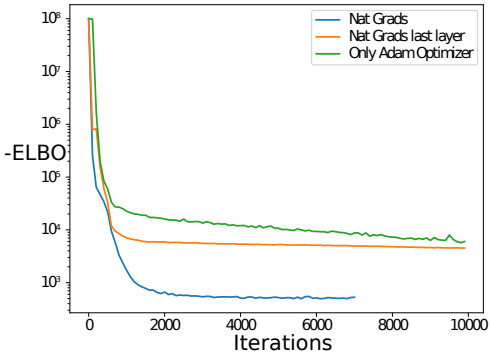
Increasing these architecture parameters enables a more powerful ability of representation. However, these variables are directly related to the computational complexity of the algorithm. Indeed, the computational complexity of a BO with DGP is given by $\mathcal{O}(j \times s \times t \times n \times (m_1^2 d_1 + \ldots + m_l^2 d_l + \ldots + m_L^2 d_L))$ where j is the number of added points,



(a) TNK constraint (d=2,n=20)
$\gamma^{Adam} = 10^{-2}, \gamma_l^{nat} = 10^{-1}, \forall 0 \leq l \leq 2$



(b) Hartmann 6d (d=6,n=60)
$\gamma^{Adam} = 10^{-2}, \gamma_2^{nat} = 10^{-1}, \gamma_0^{nat} = \gamma_1^{nat} = 10^{-2}$



(c) Trid (d=10,n=100)
$\gamma^{Adam} = 10^{-2}, \gamma_2^{nat} = 10^{-1}, \gamma_0^{nat} = \gamma_1^{nat} = 10^{-2}$

Figure 9: Comparison of the evolution of the optimization of the ELBO on three different problems using three different optimizations: an alternate optimization using the natural gradient for all the variational parameters and the Adam optimizer for the hyperparameters, an alternate optimization using the natural gradient for the variational parameters of the last layer and the Adam optimizer for the other parameters, and an optimization using the Adam optimizer for all the parameters. γ^{adam} is the step size of the Adam optimizer and γ_l^{nat} is the step size of the natural gradient for the variational parameters at layer l.

Function	Approach	mean RMSE	std RMSE	mean test log-likelihood	std test log-likelihood
TNK constraint	GP	0.18832	0.0305	-8866.59	20166.95
	DGP Adam	0.17746	0.0482	-467207	400777
	DGP Nat	**0.1659**	**0.013**	**-3671**	**1766.48**
Hartmann 6d	GP	0.3010	0.0311	**-566.27**	**451.798**
	DGP Adam	0.3166	0.0252	-2595.70	2393.99
	DGP Nat	**0.2921**	**0.0200**	-1386.12	1111.29
Trid	GP	12934	965	-10912	114
	DGP Adam	11978	496.71	-12690	808
	DGP Nat	**11151**	**388**	**-10342**	**109**

Table 1: Comparison of the Root Mean Squared Error (RMSE) and the test log-likelihood and their standard deviations (std) on three different problems with a training data size of density 10 ($n = 10 \times d$ where n is the data size and d the input dimension of the function) on 50 repetitions. GP: Gaussian Process with an RBF kernel. DGP Adam: DGP with 2 hidden layers with all its parameters optimized by an Adam Optimizer. DGP Nat: DGP with 2 hidden layers with all the variational parameters optimized by a Natural gradient and the hyperparameters by an Adam Optimizer.

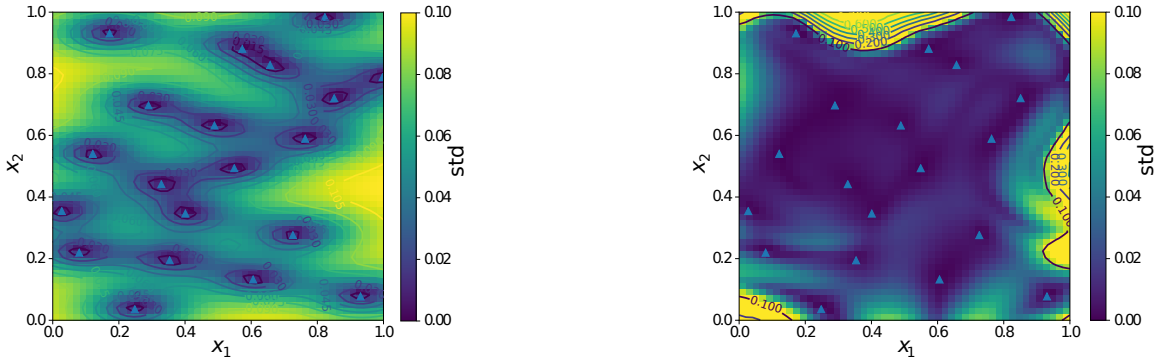


Figure 10: Standard deviation on the prediction given by a 2 hidden layers DGP model on the TNK constraint function. (left) standard deviation given by a model optimized using natural gradient on all the variational parameters. (right) standard deviation given by a model optimized using natural gradient only on the variational parameters of the last layer. An underestimation of the uncertainty happens in the second approach.

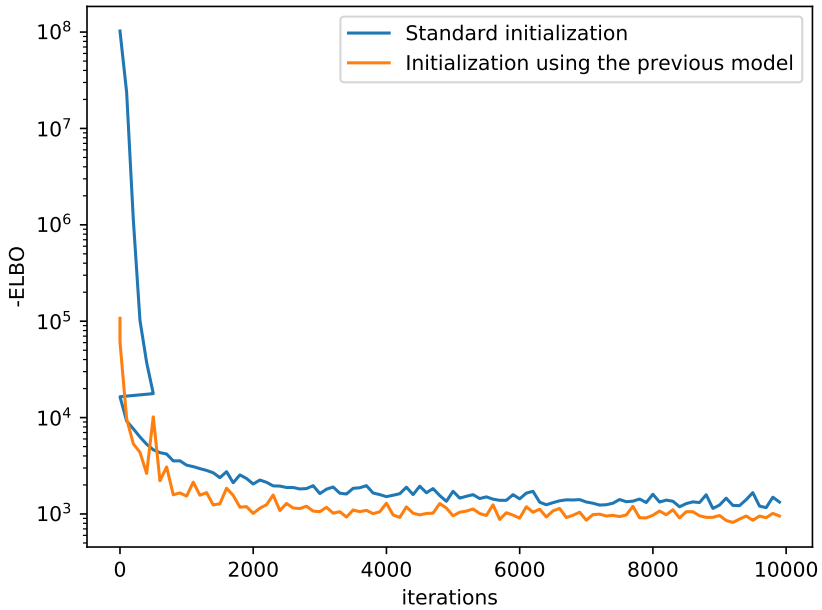


Figure 11: Comparison of the evolution of the optimization of the ELBO in the case of using the standard initialization procedure and in the case of using the previous model optimal parameters as the initialization. Using the previous model allows better and faster convergence

Algorithm 1: DGP model optimization

Require: $X, \mathbf{y}$.
Require: The number of induced variables, m.
Require: The number of layers, L.
Require: The number of iterations, $iter$.
Require: The tolerance on the variation of the ELBO Δ_k, tol.
Require: The step sizes γ^{Adam}, $\gamma_l^{Nat}, \forall l = 0, \ldots, L$
Require: $\mathbf{x}^{*var}, \mathbf{x}^{*hyper}$ a previous model optimal variational parameters and hyperparameters to initialize from if available.
if $\boldsymbol{x}^{*var}, \boldsymbol{x}^{*hyper}$ *available* **then**
    Initialize parameters:
    $\mathbf{x}_0^{hyper} \leftarrow \mathbf{x}^{*hyper}$
    $\mathbf{x}_0^{var} \leftarrow \mathbf{x}^{*var}$
else
    Initialize using another procedure (random initialization, PCA, *ect.*)
end
$ELBO_0 \leftarrow X, \mathbf{y}, \mathbf{x}_0^{hyper}, \mathbf{x}_0^{var}$
$t \leftarrow 0$
while $t < iter$ ***and*** $\Delta_k > tol$ **do**
    $\mathbf{x}_{t+1}^{hyper} \leftarrow$ Adam optimizer step$(ELBO_t, \mathbf{x}_t^{hyper}, \gamma^{Adam})$
    if *oversized natural step* $\boldsymbol{\gamma}^{nat}$ **then**
        $\gamma_j^{Nat} \leftarrow \gamma_j^{Nat}/10, \forall j = 1, \ldots l-1$
    end
    $\mathbf{x}_{t+1}^{var} \leftarrow$ Nat Grad Optimizer step$(ELBO_t, \mathbf{x}_t^{var}, \gamma_0^{Nat}, \ldots, \gamma_l^{Nat})$
    $\Delta_k \leftarrow \min\{ELBO_j, j = 0, \ldots, t-k\} - \min\{ELBO_j, j = 0, \ldots, t\}$ $t \leftarrow t+1$
end
return $model(X, \mathbf{y}, \mathbf{x}_{t-1}^{var}, \mathbf{x}_{t-1}^{hyper})$

s the number of propagated samples, t the number of optimization steps in the training, n the size of the data-set, l the number of layers, m_l the number of induced inputs at the layer l and d_l the number of hidden units at the layer l. Moreover, the number of optimization steps l increases according to the number of variables. Hence, the configuration of the DGP has to be adapted to the budget of computational time available and the complexity of the problem at hand (see Section 4 for computational times). Usually, in the early iterations of BO there is not enough information to use complex model, therefore a standard GP may be sufficient. Then, along the iterations the number of layers is increased.

It is interesting to observe that the number of inducing variables is the preponderant term in the complexity of the BO with DGP. Induced inputs were first introduced in the framework of sparse GPs. By choosing a number of induced inputs m with $m << n$ and n the number of data points, the complexity of the inference becomes $O(nm^2)$ instead of $O(n^3)$. Hence, completing computational speed ups in the construction of the model. In sparse GP, increasing the number of inducing inputs allows more precision until reaching $m = n$ when the the full GP model is recovered. In DGPs, the interpretation of the induced inputs is more complicated. Firstly, it is essential to use induced inputs to obtain the evidence lower bound for the inference in DGP. Secondly, the variables $H_l, l = 1 \ldots L$ are random variables and not deterministic as X. So, it is possible to gain more precision even if $m_l > n$.

However, the functional composition of GPs within a DGP makes each layer an approximation of a simpler function. In Figure 12, a 2-hidden layers DGP is used to approximate the modified Xiong function, the input-output of each layer is plotted. The intermediate layers try to deform the input space by stretching it, in order that the last layer approximates a stationary function, achieving an unparameterized mapping. Hence, the inner layers have a less complex behavior than the whole model. Thus, only a reduced number of induced inputs can capture the features of the hidden layers, hence, allowing computational speed ups.

3.3 Infill criteria

To use DGP in BO, it is essential to adapt the considered infill criteria to DGP. In fact, some infill criteria can not be used directly with DGP. For example, the EI formula (Eq.(12)) is based on the fact that the prediction is Gaussian.

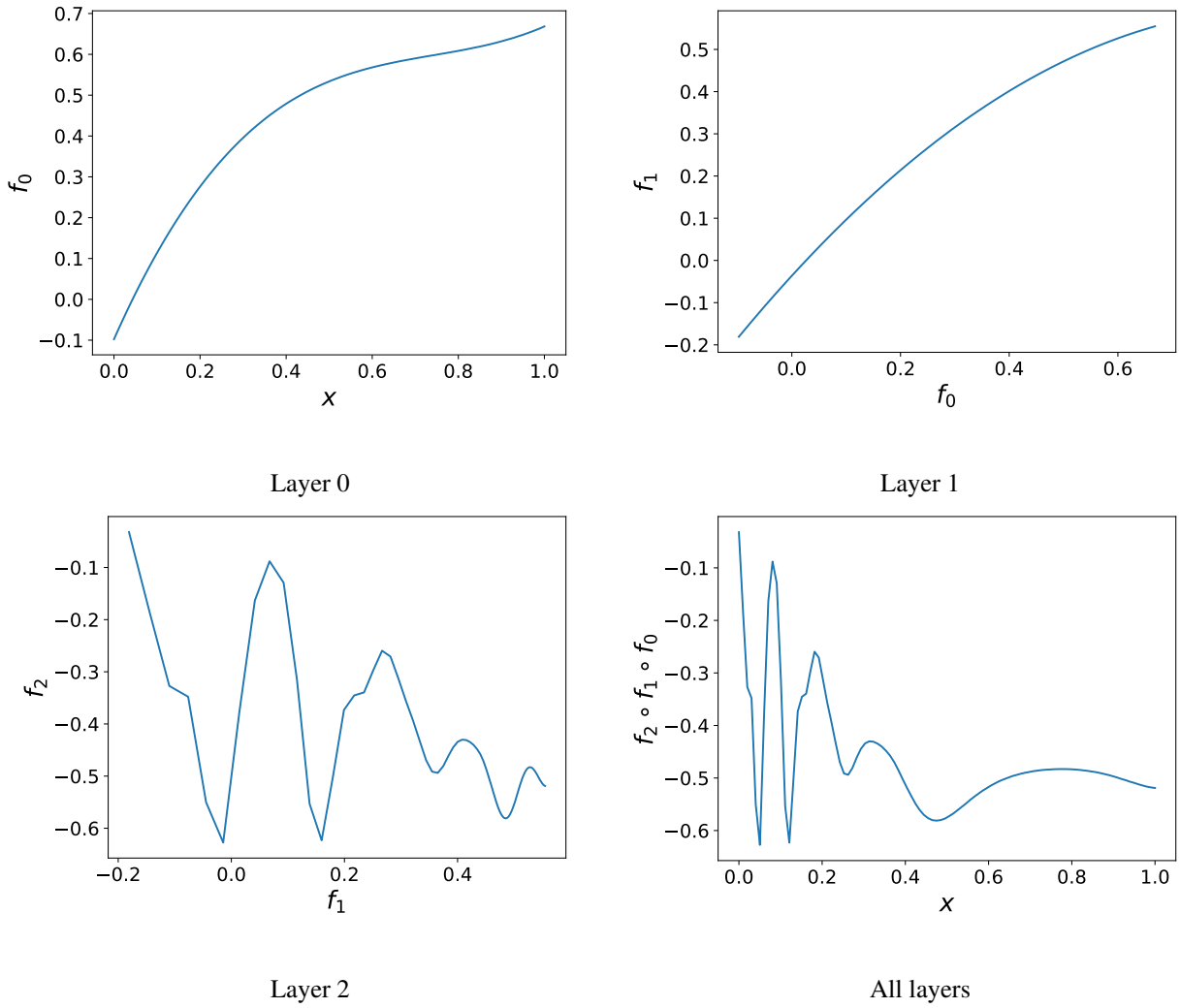


Figure 12: The input-output signal of each layer of a 2 hidden layers DGP used to approximate the modified Xiong function.

However, in DGP the prediction does not *a priori* follows a normal distribution. The expected improvement is the expected value of $I(\mathbf{x}) = max(0, y_{min} - f(\mathbf{x}))$. Therefore, the direct approach is to use sampling techniques to approximate EI (Eq.(12)). However, as observed in Figure 12 the inner layers are often simple functions, almost linear, with a last layer that approximates a deformed stationary function. This allows the prediction from the composition of GPs to be reasonably considered as Gaussian (see Figure 13). Hence, to predict using DGPs a Gaussian approximation can be made, in order to directly use the analytical formula of the infill criteria used for GPs. Hence, sampling directly on the value of the improvement is avoided.

Infill criteria such as EI are highly multi-modal, especially in high-dimensional problems. For this reason, an evolutionary algorithm such as the Differential Evolution algorithm [48] is preferred. The DGP allows parallel prediction which makes it possible to evaluate the infill criteria for all the population simultaneously. The result obtained using the evolutionary algorithm can then be optimized by a local optimizer. This hybridization is preferred to the use of multiple local searches whose number increases exponentially with the dimension of the problem.

3.4 Summary of DEGO

To summarize the proposed BO & DGP algorithm, DEGO, Algorithm 2 describes the steps previously discussed. The Expected Improvement is used as the infill criterion, but other infill criteria may be used. If approximation of the DGP prediction by a Gaussian is not valid, sampling is used to approximate the infill criterion. Some empirical rules can be used to determine the number of initial points and the number of added points depending on the dimension of the

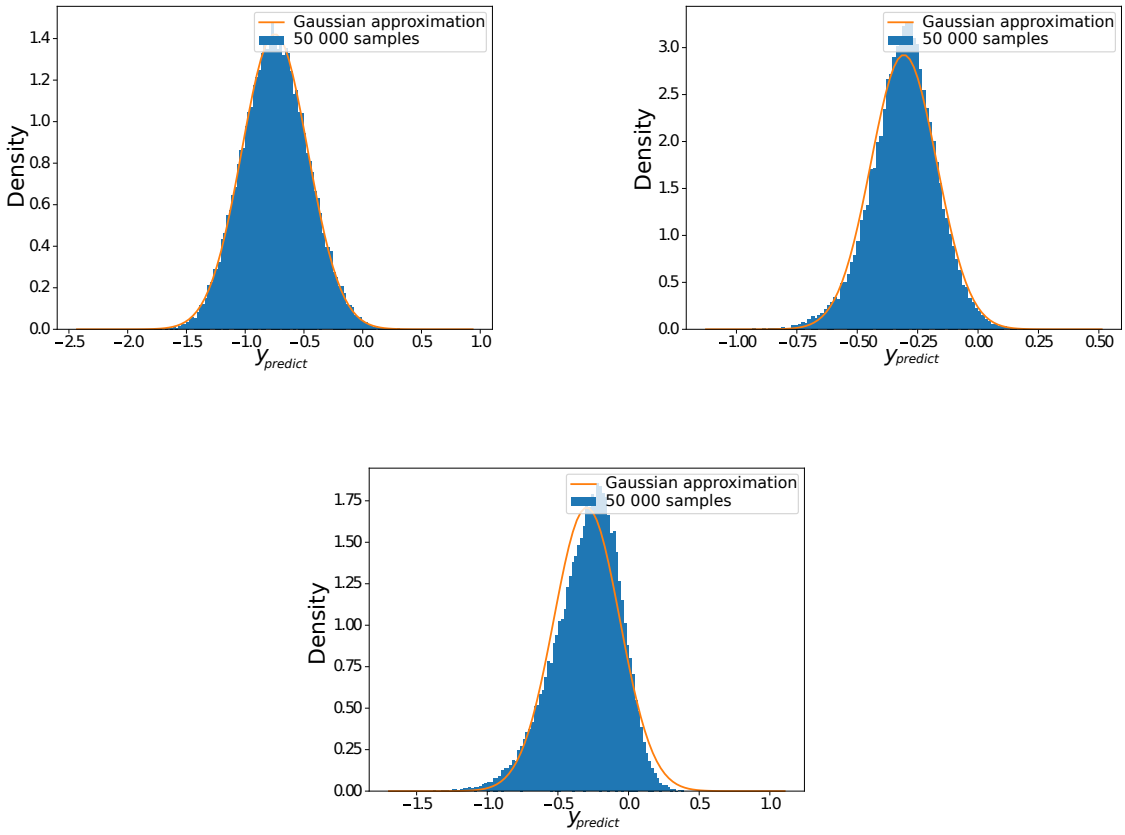


Figure 13: Samples on the value of the prediction of a DGP model in three different locations. In the first two figures, the prediction is almost Gaussian. In the third figure the distribution of the prediction is slightly asymmetric, but it is still well approximated by a Gaussian distribution.

problem d (for the experimentations in Section 4, for all the problems an initial DoE of size $5 \times d$ is considered and $10 \times d$ points are added in the BO process.). The size of the induced variables is fixed along all the iterations to the total number of points at the end of the BO. This allows the models to keep the same number of parameters along the iterations, making it possible to initialize them from the previous models. Moreover, as discussed previously, setting the number of induced variables to a number larger than the number of points in DGP may allow a better representation. The model is updated from the previous model optimal parameter values for a certain number of consecutive iterations allowing speed ups in the optimization, and then initialized from scratch every n_{update} iterations to avoid being tricked in some bad local minima.
In this algorithm the unconstrained case is considered. However, the generalization to the constrained case is straightforward, since it comes back to create DGP models also for the constraints and to use a constrained infill criterion as the Probability of Feasibility or the Expected Violation.

Algorithm 2: DEGO algorithm

Require: Expensive black-box problem of dimension d to optimize, f^{exact}
Require: Number of initial points n.
Require: Number of total added points n_{add}.
Require: Number of layers l.
Require: Number of iterations in the optimization of the model $iter$.
Require: Number of consecutive model updates using the previous model optimal values n_{update}.
$X_0 \leftarrow LHS(d, n)$ (or another design of experiments method)
$\mathbf{y}_0 \leftarrow f^{exact}(X_0)$ (**evaluate**)
$m \leftarrow n + n_{add}$ (set the number of induced variables to the final number of points)
$t \leftarrow 0$
$model_0 \leftarrow$ **DGP model optimization**$(X_0, \mathbf{y}_0, m, l, iter)$ (optimize model from scratch)
while $t \leq n_{add}$ **do**
    $t \leftarrow t + 1$
    $\mathbf{x}^{(t)} \leftarrow argmax(EI_{model_{t-1}}(\mathbf{x}))$(use sampling to estimate the EI)
    $y^{(t)} \leftarrow f^{exact}(\mathbf{x}^{(t)})$ (**evaluate**)
    $X_t \leftarrow \begin{bmatrix} X_{t-1} \\ \mathbf{x}_t \end{bmatrix}$ (add a row to the matrix)
    $\mathbf{y}_t \leftarrow \begin{bmatrix} \mathbf{y}_{t-1} \\ y^{(t)} \end{bmatrix}$ (add an element to the vector)
    if $t \% n_{update} \neq 0$ **then**
        $model_t \leftarrow$ **DGP model training**$(X_t, \mathbf{y}_t, m, l, iter, model_{t-1})$ (optimize model using the optimal parameter values of the previous model as initialization)
    else
        $model_t \leftarrow$ **DGP model training**$(X_t, \mathbf{y}_t, m, l, iter)$ (optimize model from scratch)
    end
end
return $X_t, \boldsymbol{y}_t$

4 Experimentations

4.1 Analytical test problems

Experimentations on different analytical problems have been performed to assess the performance of DGPs in BO. Details on the experimental setup are presented in Appendix B.

4.1.1 2d constrained problem

The function to optimize is a simple two dimensional quadratic function. While the constraint is non-stationary and feasible when equal to zero. An important discontinuity between the feasible and non feasible regions breaks the smoothness of the constraint (Figure 14). Therefore, the problem is challenging for standard GP, since the optimal region is exactly at the boundary of the discontinuity, requiring an accurate modeling of the non-stationarity.

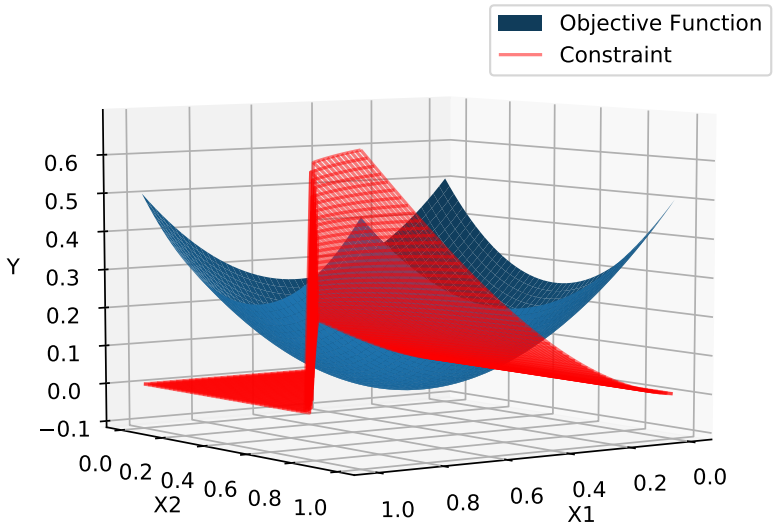


Figure 14: Objective and constraint functions 2d problem

A DoE of 10 initial data points is initialized using a Latin Hypercube Sampling. Then, 20 points are added using the Expected Violation criterion (EV) to handle the constraint. A standard Gaussian Process with an RBF kernel is used to approximate the objective function. The DGPs are used with an RBF kernel in each layer and are optimized using 5000 optimization steps. To assess the robustness of the algorithms, 50 repetitions are performed.

The convergence plots of the BO algorithms with GP, DGP 2, 3, 4 and 5 layers are displayed in Figure 15. As expected, the BO & GP is not well-suited for this problem, actually at the end of the algorithm, the median is still far from the actual minimum and there is an important variation. This is due to the fact that the GP can not capture the discontinuity and the feasible tray region of the constraint and considers an important region as unfeasible (Figure 17). However, DEGO is able to capture the feasible region (Figure 18), which makes it able to give efficient results with a median at the end of the algorithms near to the actual minimum and better robustness. The interesting observation concerns the mean and standard deviation given in Table 2, where the 3 layers DGP gives the best results. Increasing the number of layers deteriorates the quality of the results. This is explained by the fact that 5000 steps in the optimization for DGPs with more than three layers in this case is insufficient. Hence, the necessity to increase the number of optimization steps in the training of deeper models. However, increasing the number of layers and the number of steps induces additional computational time (Fig 16) which quickly becomes an important burden for high dimensional problems. For the remaining test cases only a DGP with two layers is considered.

4.1.2 Trid function

The Trid function is considered in 10 dimensions Eq.(29). The range of variation of the $10d$ Trid function values is large. It varies from values of 10^5 to its global minimum $f(\mathbf{x}^*) = -210$ (Figure 26). This large variation range makes

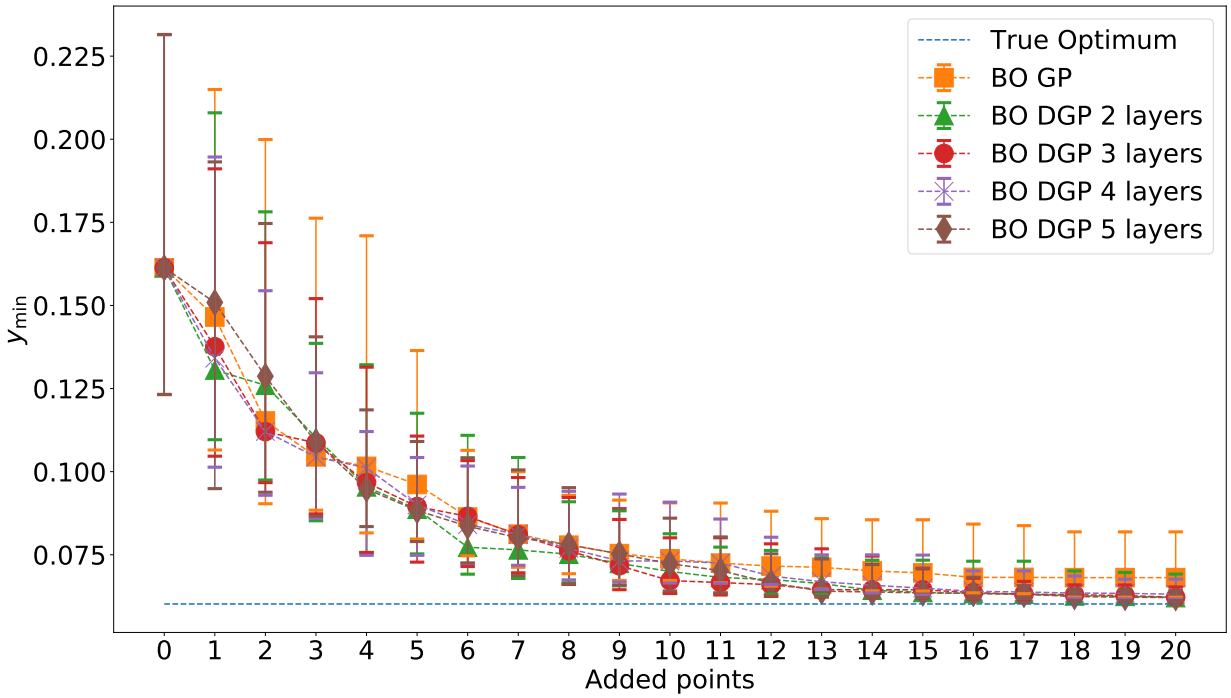


Figure 15: Plot of convergence of BO using different architectures of DGPs with 5000 training steps for 50 different initial DoE and a standard GP. The markers indicate the median of the minimum obtained while the error-bars indicate the first and third quartiles.

Table 2: Performance of BO (values of the minimum found) with standard GP and different DGP configurations on the constrained 2d problem. 50 repetitions are performed.

BO &	**mean minimum obtained**	**standard deviation**	**gap between the mean minimum and the global minimum.**
GP	0.09356	0.0605	0.03336
DGP 2 L	0.08468	0.059793	0.02448
DGP 3 L	**0.073918**	**0.04293**	**0.01371**
DGP 4 L	0.08066	0.05073	0.02046
DGP 5 L	0.08204	0.05707	0.02184
Global minimum	0.0602	-	-

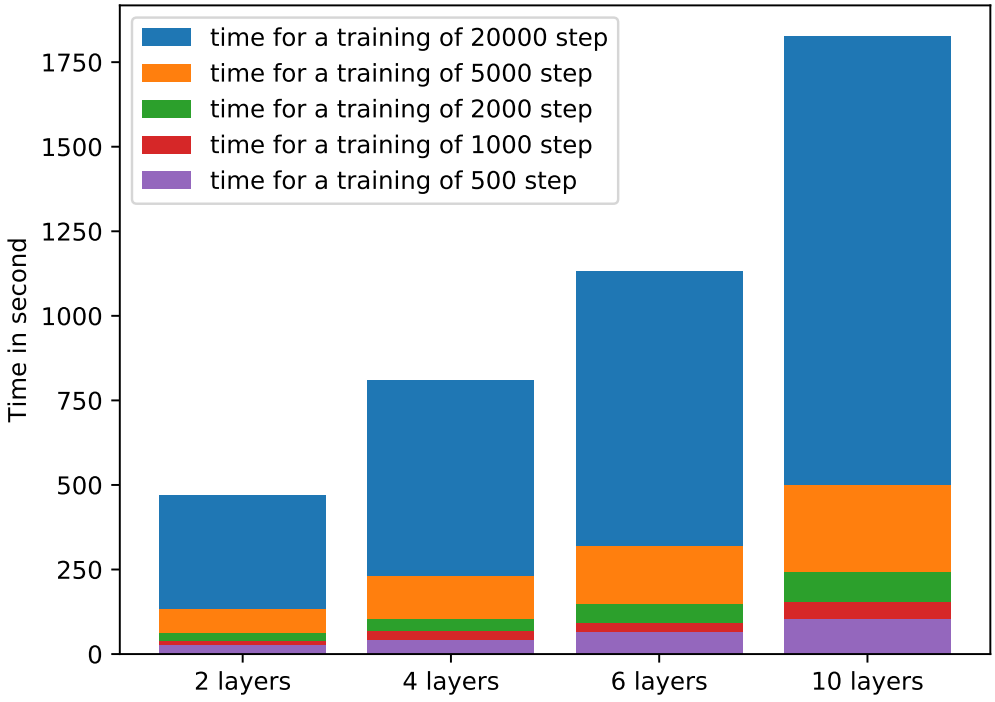


Figure 16: Mean time in one iteration of BO according to the number of layers in DGP and the number of steps in the training of the model.

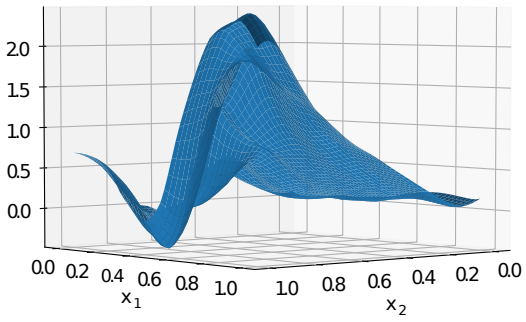


Figure 17: GP approximation of the constraint with a DoE of 40 points.

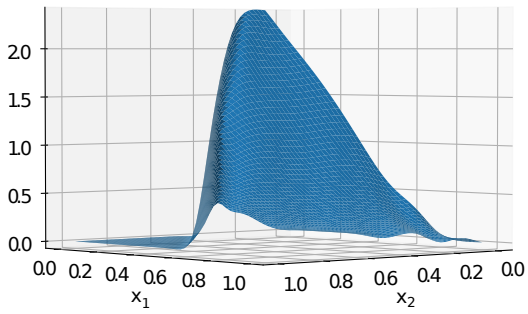


Figure 18: DGP 2 layers approximation of the constraint with a DoE of 40 points.

Table 3: Performance of BO (values of the minimum found) with standard GP, non-linear mapping with two knots (NLM), Adaptive partial non-stationary kriging (APNS), Deep Gaussian Processes with two hidden layers (DGP) on the Trid function.

BO &	**mean minimum obtained**	**standrad deviation**	**gap between the mean minimum and the global minimum.**
GP	-20.730	75.654	189.27
NLM	-57.727	59.920	152.273
APNS	-49.112	62.746	160.888
DGP (DEGO)	**-206.739**	**1.5521**	**3.261**
Global minimum	-210	-	-

it difficult for BO with stationary GP to find the global minimum. This function was also used in [35] to assess the performance of BO with non-stationary GP using the non-linear mapping and a mixture of the non-linear mapping and standard GP called Adaptive partial non-stationary kriging.

The results of BO with a DGP of 2 hidden layers are compared to the results found in [35] on 50 different DoE (Table 3). The initial DoE is initialized with a Latin Hypercube Sampling with 50 initial points, and 100 points are added during the BO using the EI criterion.

The minimum given by BO & GP, NLM (Non-Linear Mapping) and APNS (Adaptive Partial Non-Stationary kriging) for this problem are not close to the global minimum. Moreover, there is a high variation in the obtained results, showing the difficulty of these approaches to handle this optimization problem. However, there is an important difference between DEGO (BO & DGP) compared to the other approaches. The approximated mean minimum obtained -206.739 is very close to the actual global minimum -210 with a low standard deviation of 1.5521, hence confirming the robustness of the approach. The convergence plot of DEGO (Figure 19) also shows the fast convergence of this approach. In fact, after only 65 iterations the algorithm is stabilized around the global optimum.

4.1.3 Hartmann-6d function

The Hartmann-6d is a $6d$ function (Eq.(30)). Which is smooth and does not show non-stationary behavior (Figure 27). The interest of this function is that BO coupled with some non-stationary approaches can not reach its global minima while BO & classic GP presents good performance on it. Hence, using DEGO on this function allows to demonstrate the robustness of this non-stationary BO algorithm on stationary functions. This makes sense when applying BO to black-box functions when there is no information about the stationarity of the problem at hand.

The results of BO with a DGP of 2 hidden layers are compared to the results found in [35] on 50 different DoE (Table 4). The initial DoE are initialized using a Latin Hypercube Sampling with 30 initial points and 60 points are added during the BO process using the EI criterion.

The results obtained by BO & NLM and APNS are far from the global optimum and show important variation. The stationary GP gives better and more robust results, since it is adapted to the stationary behavior of the Hartmann function. However, the minimum obtained by DEGO is closer to the global optimum and the optimization is more robust to the initial DoE than standard GP even if the function is stationary. This shows the interest of using the DEGO even for functions without any information on their stationary behavior unlike BO & NLM and APNS that are well-suited for stationary functions. Moreover, the convergence curve of the DEGO (Figure 20) highlights its speed

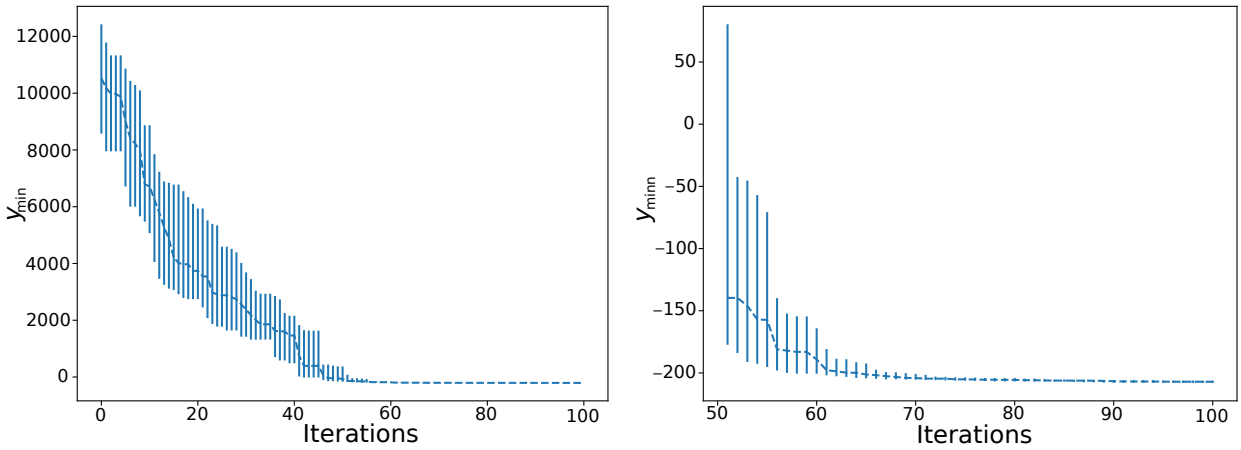


Figure 19: DEGO plot of convergence of y_{min} on the Trid function with 50 DoE. The curve represent the median of y_{min} while the vertical bars represent the first and third quartiles. (left) Convergence plot on all the iterations. (right) Convergence plot on the 50 last iterations.

Table 4: Performance of BO (values of the minimum found) with standard GP, non-linear mapping with two knots (NLM), Adaptive partial non-stationary kriging (APNS), and Deep Gaussian Processes with two hidden layers (DGP) on the Hartmann 6d function.

BO &	**mean minimum obtained**	**standard deviation**	**gap between the mean minimum and the global minimum.**
GP	-3.148	0.275	0.174
NLM	-2.818	0.570	0.504
APNS	-3.051	0.415	0.271
DGP (DEGO)	**-3.250**	**0.098**	**0.072**
Global minimum	-3.322	-	-

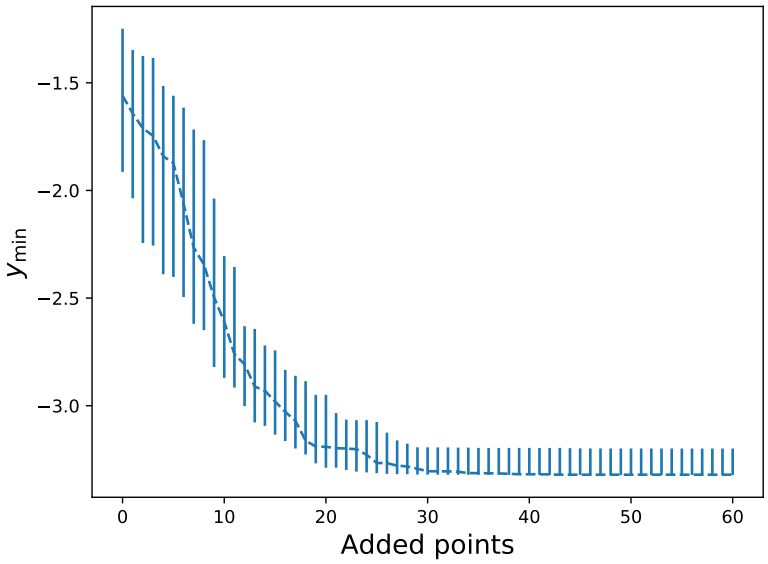


Figure 20: DEGO plot of convergence of y_{min} on the Hartmann-6d function with 50 DoE. The curve represent the median of y_{min} while the vertical bars represent the first and third quartiles.

of convergence, since after only 30 added points the results given are better and more robust than the other algorithms at the end of the BO process.

4.2 Application to industrial test case: design of aerospace vehicle

To confirm the interest of the BO & DGP approach, DEGO, a real world aerospace vehicle design optimization problem is considered. It consists of the maximization of the change in velocity (ΔV) of a solid-propellant booster engine. It is a representative physical problem for solid booster design with simulation models fast enough to provide the real minimum to compare and illustrate the efficiency of the proposed algorithm.

4.2.1 Description of the problem

The optimization of ΔV for a solid propellant booster is considered (Figure 21). Four design variables are considered:

- Propellant mass: 5 t $< m_{prop} < 15$ t
- Combustion chamber pressure: 5 bar $< p_c < 100$ bar
- Throat nozzle diameter: 0.2 m $< d_c < 1$ m
- Nozzle exit diameter: 0.5 m $< d_s < 1.2$ m

Different constraints are also considered including a structural one limiting the combustion pressure according to the motor case, 6 geometrical constraints on the internal vehicle layout for the propellant and the nozzle, a jet breakaway constraint concerning the nozzle throat diameter and the nozzle exit diameter, and a constraint on the maximum Gross Lift-Off Weight (GLOW) allowed.

$$
\begin{array}{ll}
\text{Minimize:} & -\Delta V(\mathbf{x}) \\
\text{w.r.t:} & \mathbf{X} = [m_{prop}, p_{c,c}, {}_{,s}] \\
\text{s.t:} & \left\{ \begin{array}{l} \text{1 structural constraint} \\ \text{6 geometrical constraints} \\ \text{1 jet breakaway constraints} \\ \text{maximum GLOW allowed} \end{array} \right.
\end{array}
$$

This problem is expected to have non-stationarity behaviors due to some constraints. In fact, the constraints may have a different behavior in the feasible and unfeasible regions. Moreover the objective function which is the change in velocity may have a tray region when it is equal to zero, due to an insufficient propellant mass (Figure 22).

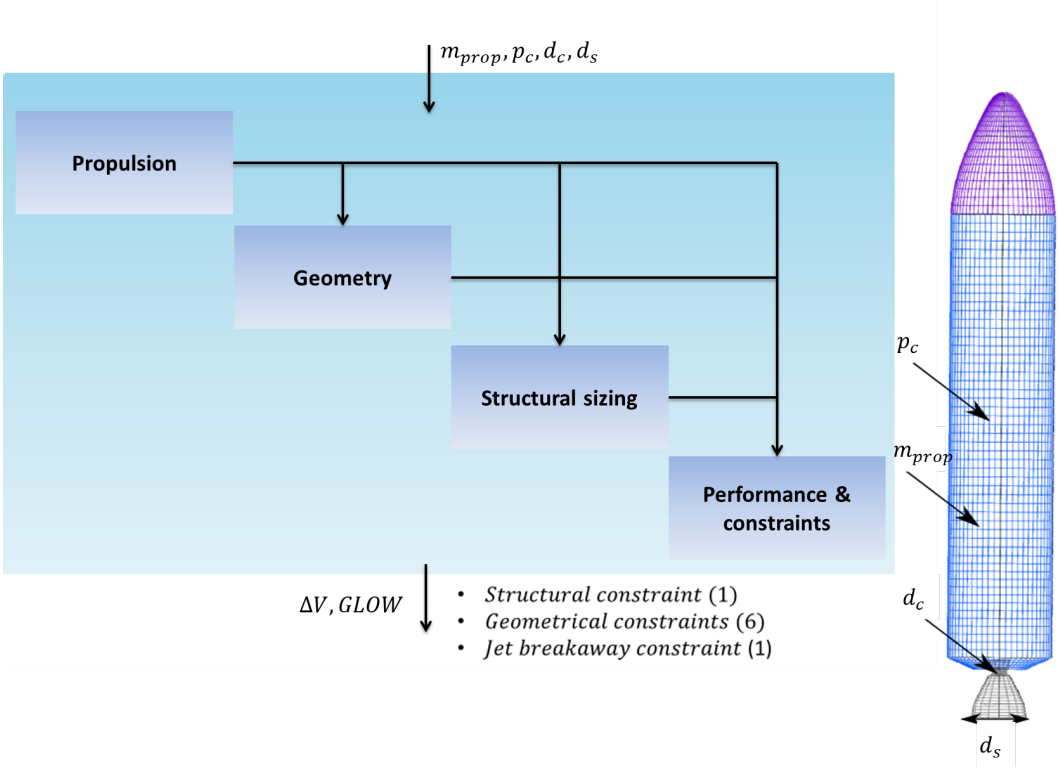


Figure 21: Optimization problem of a solid-propellant booster engine.

4.2.2 Experimental results

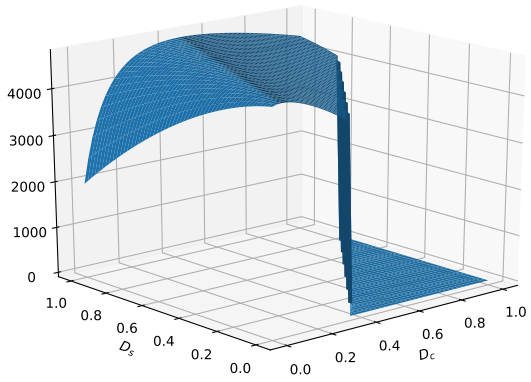
The initial DoE are set using a Latin Hypercube Sampling of 30 points and 50 points are added with BO using EI for the objective function and EV for the constraints. To assess the robustness of the results, 10 repetitions are performed.

The plots of convergence of the BO algorithms are displayed in Figure 23. After adding 50 points, both BO & DGP (DEGO) and standard GP reach the global minimum. However, DEGO is faster to converge. DEGO shows robust results near the global optimum $4738m/s$ after only 12 iterations, while the BO & GP is not stabilized until 24 iterations (Table 5).

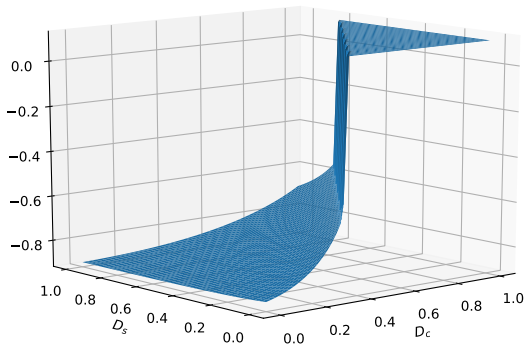
Table 5: Performance of the algorithms after 12 added points, after 24 added points and after 50 added points.

	After 12 added points		After 24 added points		After 50 added points	
Algorithm	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**
BO & GP	4656	97.12	4709	41.33	4725	10.63
DEGO	4709	27.95	4718	22.53	4736	7.49

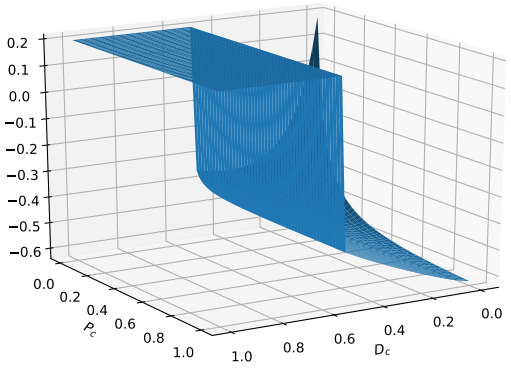
The speed of convergence is important in case of expensive black-box functions. Indeed, one evaluation of the objective function or the constraints can cost multiple hours, even multiple days. Hence, DEGO is interesting even for problems where BO & GP can reach the global minimum, due to its speed of convergence which can reduce drastically the number of evaluations needed to converge.



A sectional view of the change in velocity according to the diameters of the nozzle.



A sectional view of a constraint according to the diameters of the nozzle.



A sectional view of a constraint according to the throat nozzle diameter and the combustion chamber pressure.

Figure 22: Sectional view of the non-stationary behaviors of some functions involved in the booster problem

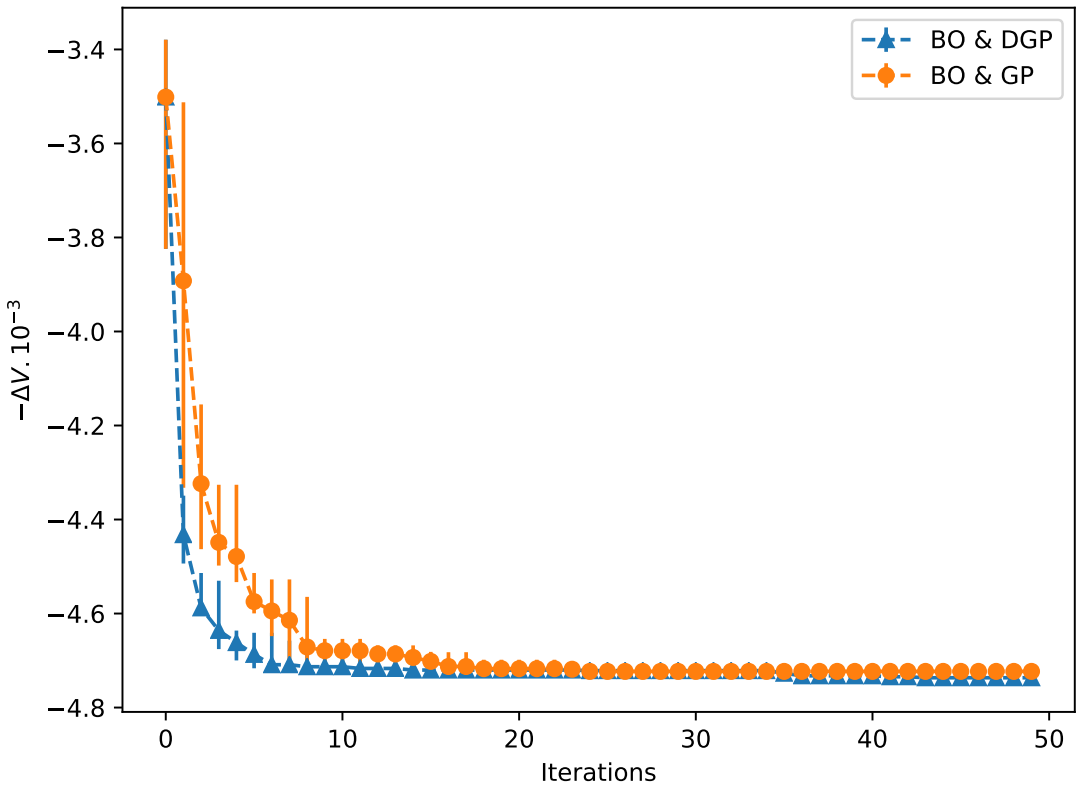


Figure 23: Convergence curve of $-\Delta V$ using DEGO and BO & GP.

5 Conclusion and future works

A coupling between Bayesian Optimization (BO) and Deep Gaussian Processes (DGP) has been proposed in this paper. This coupling induces some adaptations of the handling of DGPs (training approach, uncertainty on the prediction, architecture of the DGP) and also on BO (the iterative structure of BO, infill criteria). The main propositions are the use of natural gradient on all the variational parameters of the DGP in the training which enables a better convergence of the Evidence Lower Bound, and a better uncertainty quantification on the prediction. Also, to take advantage of the iterative structure of BO, the optimal parameter values of the previous model are used as initialization for the next one to speed up the training of the model. In the considered problems, a DGP with 2 hidden layers proved to give a well-balanced compromise between the time complexity in the training and its power of representation. To use the classic infill criteria considering that the prediction of the DGP model is not necessarily Gaussian, a sampling procedure to approximate infill criterion such as the Expected Improvement was proposed. The algorithm DEGO obtained following these propositions was assessed on analytical test optimization problems. The experimentation showed its better efficiency and robustness compared with standard BO & GP and approaches using non-linear mapping to handle non-stationarity. Finally, this algorithm was applied to a real-world aerospace engineering design problem, showing its improved speed of convergence compared to standard BO & GP.

The goal of this paper was to propose a BO & DGP algorithm and to illustrate its tangible interest over the state-of-the-art approaches. However, it is necessary to explore more this coupling. For example, the handling of the step size of the Natural Gradient used in each layer when training a DGP model can be improved. Also, infill criteria such as Thompson Sampling or criteria using information theory may be more adapted to DGP than the EI. More parallelism can also be integrated at different levels of DEGO.

Acknowledgments

This work is co-funded by ONERA-The French Aerospace Lab and Université de Lille, in the context of a joint PhD thesis.
Discussions with Hugh Salimbeni and Zhenwen Dai were very helpful for this work, special thanks to them.
The Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

Appendices

A Functions

Modified Xiong function:

$$f(x) = -0.5\left(\sin\left(40(x-0.85)^4\right)\cos\left(2.5(x-0.95)\right) + 0.5(x-0.9) + 1\right),\ x \in [0,1] \tag{27}$$

Modified TNK constraint function:

$$f(\mathbf{x}) = 1.6(x_0-0.6)^2 + 1.6(x_1-0.6)^2 - 0.2\cos\left(20\arctan\left(\frac{0.3x_0}{(x_1+10^{-8})}\right)\right) - 0.4,\ \mathbf{x} \in [0,1]\times[0,1] \tag{28}$$

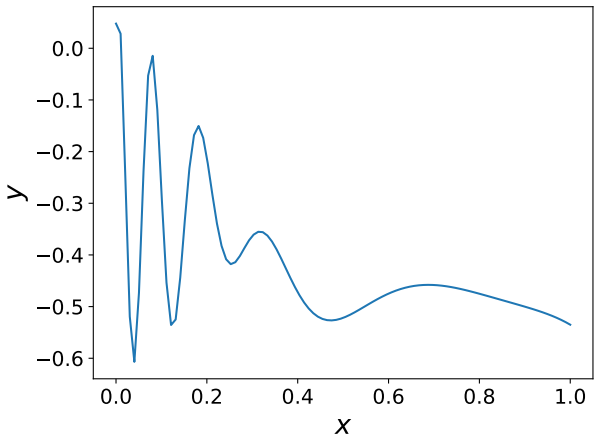


Figure 24: Modified Xiong function

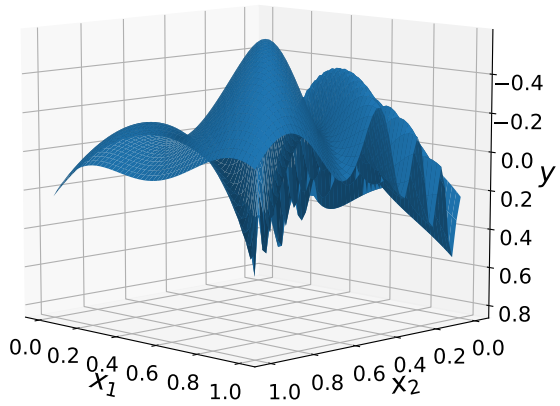


Figure 25: Modified TNK constraint

10d Trid function:

$$f(\mathbf{x}) = \sum_{i=1}^{10}(x_i-1)^2 - \sum_{i=2}^{10} x_i x_{i-1},\ x_i \in [-100,100], \forall i = 1,\ldots,10 \tag{29}$$

Hartmann-6d function:

$$f(\mathbf{x}) = \sum_{i=1}^{4}\alpha_i \exp\left(-\sum_{j=1}^{6} A_{ij}(x_j - Pij)^2\right),\ x_i \in [0,1], \forall i = 1,\ldots,6 \tag{30}$$

with:

$$\alpha = [1, 1.2, 3, 3.2]^\top$$

and

$$P = 10^{-4}\begin{bmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{bmatrix}$$

and

$$A = \begin{bmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{bmatrix}$$

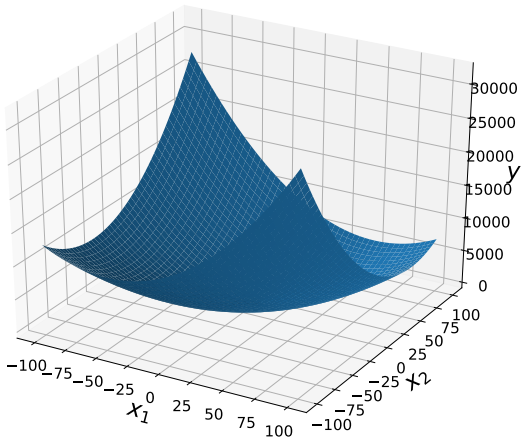


Figure 26: Sectional $2d$ view of the Trid function showing where the global minimum lie

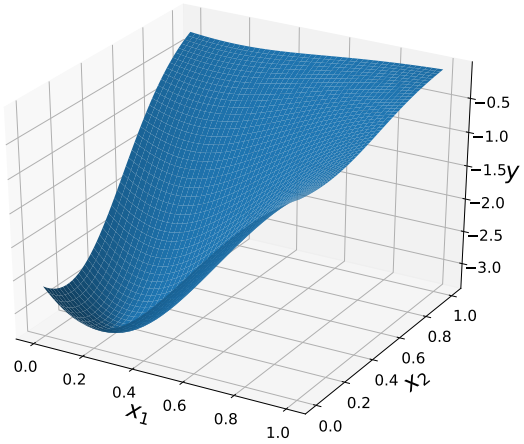


Figure 27: Sectional $2d$ view of the Hartmann-6d function showing where the global minimum lie

B Experimental setup

- All experiments were executed on Grid'5000 using a Tesla P100 GPU. The code is based on GPflow [49] and Doubly-Stochastic-DGP [38].
- For all DGPs, RBF kernels are used with a length-scale and variance initialized to 1 if it does not get an initialization from a previous DGP. The data is scaled to have a zero mean and a variance equal to 1.
- The Adam optimizer is set with $\beta_1 = 0.8$ and $\beta_2 = 0.9$ and a step size $\gamma^{adam} = 0.01$.
- The natural gradient step size is initialized for all layers at $\gamma^{nat} = 0.1$
- For DEGO the number of successive updates before optimizing from scratch is 5.
- The infill criteria are optimized using a parallel differential evolution algorithm with a population of 400 and 100 generations.
- A Github repository featuring DEGO algorithm will be available after the publication of the paper.

References

[1] Gary Wang and Songqing Shan. Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical design*, 129(4):370–380, 2007.

[2] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

[3] Carl Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

[4] Athanasios Papoulis and Pillai Unnikrishna. Probability, random variables and stochastic processes, 1991.

[5] Paul CD Milly, Julio Betancourt, Malin Falkenmark, Robert M Hirsch, Zbigniew W Kundzewicz, Dennis P Lettenmaier, and Ronald J Stouffer. Stationarity is dead: Whither water management? *Science*, 319(5863):573–574, 2008.

[6] Ian Cordery and S L. YAO. Non stationarity of phenomena related to drought. *Extreme hydrological events. Proc. international symposium, Yokohama, 1993*, 01 1993.

[7] Sahil Garg, Amarjeet Singh, and Fabio Ramos. Learning non-stationary space-time models for environmental monitoring. In *Twenty-Sixth AAAI Conference on Artificial Intelligence, Toronto*, 2012.

[8] Sreenivas Konda. *Fitting Models of Nonstationary Time Series: An Application to EEG Data*. PhD thesis, Case Western Reserve University, 2006.

[9] Peter M Atkinson and Christopher D Lloyd. Non-stationary variogram models for geostatistical sampling optimisation: An empirical investigation using elevation data. *Computers & Geosciences*, 33(10):1285–1300, 2007.

[10] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[11] Donald F Specht. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[12] Brani Vidakovic. *Statistical modeling by wavelets*, volume 503. John Wiley & Sons, 2009.

[13] Dave Higdon, Jenise Swall, and John Kern. Non-stationary spatial modeling. *Bayesian statistics*, 6(1):761–768, 1999.

[14] Christopher J Paciorek and Mark J Schervish. Spatial modelling using a new class of nonstationary covariance functions. *Environmetrics*, 17(5):483–506, 2006.

[15] Carl E Rasmussen and Zoubin Ghahramani. Infinite mixtures of gaussian process experts. In *Advances in neural information processing systems*, pages 881–888, 2002.

[16] Timothy C Haas. Kriging and automated variogram modeling within a moving window. *Atmospheric Environment. Part A. General Topics*, 24(7):1759–1769, 1990.

[17] Paul D Sampson and Peter Guttorp. Nonparametric estimation of nonstationary spatial covariance structure. *Journal of the American Statistical Association*, 87(417):108–119, 1992.

[18] Ying Xiong, Wei Chen, Daniel Apley, and Xuru Ding. A non-stationary covariance-based kriging method for metamodelling in engineering design. *International Journal for Numerical Methods in Engineering*, 71(6):733–756, 2007.

[19] Andreas Damianou and Neil Lawrence. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pages 207–215, 2013.

[20] Victor Picheny, Tobias Wagner, and David Ginsbourger. A benchmark of kriging-based infill criteria for noisy optimization. *Structural and Multidisciplinary Optimization*, 48(3):607–626, 2013.

[21] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[22] Kinjal Basu and Souvik Ghosh. Analysis of thompson sampling for gaussian process optimization in the bandit setting. *arXiv preprint arXiv:1705.06808*, 2017.

[23] Dennis D Cox and Susan John. Sdo: A statistical method for global optimization. In *in Multidisciplinary Design Optimization: State-of-the-Art*, pages 315–329, 1997.

[24] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in neural information processing systems*, pages 918–926, 2014.

[25] Bobak Shahriari, Ziyu Wang, Matthew W Hoffman, Alexandre Bouchard-Côté, and Nando de Freitas. An entropy search portfolio for bayesian optimization. *arXiv preprint arXiv:1406.4625*, 2014.

[26] Matthew D Hoffman, Eric Brochu, and Nando de Freitas. Portfolio allocation for bayesian optimization. In *UAI*, pages 327–336. Citeseer, 2011.

[27] Michael J Sasena. *Flexibility and efficiency enhancements for constrained global design optimization with kriging approximations*. PhD thesis, University of Michigan Ann Arbor, MI, 2002.

[28] JM Parr, AJ Keane, Alexander IJ Forrester, and CME Holden. Infill sampling criteria for surrogate-based optimization with constraint handling. *Engineering Optimization*, 44(10):1147–1166, 2012.

[29] Michael J Sasena, Panos Y Papalambros, and Pierre Goovaerts. The use of surrogate modeling algorithms to exploit disparities in function computation time within simulation-based optimization. *Constraints*, 2:5, 2001.

[30] Charles Audet, J Denni, Douglas Moore, Andrew Booker, and Paul Frank. A surrogate-model-based method for constrained optimization. In *8th Symposium on Multidisciplinary Analysis and Optimization*, page 4891, 2000.

[31] Matthias Schonlau, William J Welch, and D Jones. Global optimization with nonparametric function fitting. *Proceedings of the ASA, section on physical and engineering sciences*, pages 183–186, 1996.

[32] Mark N Gibbs. *Bayesian Gaussian processes for regression and classification*. PhD thesis, University of Cambridge, 1998.

[33] Robert B Gramacy and Herbert K H Lee. Bayesian treed gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130, 2008.

[34] Sébastien Marmin, David Ginsbourger, Jean Baccou, and Jacques Liandrat. Warped gaussian processes and derivative-based sequential designs for functions with heterogeneous variations. *SIAM/ASA Journal on Uncertainty Quantification*, 6(3):991–1018, 2018.

[35] David John James Toal and Andy J Keane. Non-stationary kriging for design optimization. *Engineering Optimization*, 44(6):741–765, 2012.

[36] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[37] Zhenwen Dai, Andreas Damianou, Javier González, and Neil Lawrence. Variational auto-encoded deep gaussian processes. *arXiv preprint arXiv:1511.06455*, 2015.

[38] Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes. In *Advances in Neural Information Processing Systems*, pages 4588–4599, 2017.

[39] Thang Bui, Daniel Hernández-Lobato, Jose Hernandez-Lobato, Yingzhen Li, and Richard Turner. Deep gaussian processes for regression using approximate expectation propagation. In *International Conference on Machine Learning*, pages 1472–1481, 2016.

[40] Marton Havasi, José Miguel Hernández-Lobato, and Juan José Murillo-Fuentes. Inference in deep gaussian processes using stochastic gradient hamiltonian monte carlo. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7506–7516. Curran Associates, Inc., 2018.

[41] Andreas Damianou. Deep gaussian processes and variational propagation of uncertainty. *PhD Thesis, University of Sheffield*, 2015.

[42] Michalis Titsias and Neil D Lawrence. Bayesian gaussian process latent variable model. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 844–851, 2010.

[43] Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264, 2006.

[44] Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.

[45] Ali Hebbal, Loïc Brevault, Mathieu Balesdent, Ei-Ghazali Taibi, and Nouredine Melab. Efficient global optimization using deep gaussian processes. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2018.

[46] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[47] Hugh Salimbeni, Stefanos Eleftheriadis, and James Hensman. Natural gradients in practice: Non-conjugate variational inference in gaussian process models. In *Artificial Intelligence and Statistics*, 2018.

[48] A Kai Qin, Vicky Ling Huang, and Ponnuthurai N Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE transactions on Evolutionary Computation*, 13(2):398–417, 2009.

[49] Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke. Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman. GPflow: A Gaussian process library using TensorFlow. *Journal of Machine Learning Research*, 18(40):1–6, apr 2017.