Efficient Global Optimization using Deep Gaussian Processes

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

EFFICIENT GLOBAL OPTIMIZATION USING DEEP GAUSSIAN PROCESSES

Ali Hebbal [*]

ONERA, DTIS, Université 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

Efficient Global Optimization (EGO) is widely used for the optimization of computationally expensive black-box functions. It uses a surrogate modeling technique based on Gaussian Processes (Kriging). However, due to the use of a stationary covariance, Kriging is not well suited for approximating non stationary functions. This paper explores the integration of Deep Gaussian processes (DGP) in EGO framework to deal with the non-stationary issues and investigates the induced challenges and opportunities. Numerical experimentations are performed on analytical problems to highlight the different aspects of DGP and EGO.

***K*eywords** Efficient Global Optimization, non-stationary Kriging, Deep Gaussian Processes, surrogate modeling.

1 Introduction

Bayesian algorithms are widely used to deal with expensive black-box function optimization. They are based on surrogate models, allowing the emulation of the statistical relationship between the design variables and the response (objective function and constraints), to predict its behaviour using a dataset also called Design of Experiments (DoE). The evaluation cost of the surrogate models is cheaper, so it is possible to evaluate a greater number of design candidates. A complete review on the surrogate models that are widely used in design optimization is given in [1]. One of the most popular Bayesian optimization methods is "Efficient Global Optimization" (EGO) developed by Jones *et al.* [2]. It uses Kriging surrogate model [3] which is based on the Gaussian Process (GP) theory. The main advantage of Kriging is that in addition to the prediction, it provides 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 expensive functions and the surrogate model is updated and so on, until a stopping criterion is satisfied.

Classical Kriging is a GP with a stationary covariance function, inducing a uniform smoothness of the prediction. While it is effective to approximate stationary functions, it causes a major issue in the prediction of non-stationary ones. Indeed, in many design optimization problems, the objective functions or constraints vary with a completely different smoothness along the input space, due to the abrupt change of a physical property for example. Different approaches have been proposed to overcome this issue. Direct formulation of non-stationary covariance function, is one of the most explored strategies. Higdon *et al.* [4] introduced a non-stationary version of the squared exponential

[*]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 - University of Lille, el-ghazali.talbi@univ-lille1.fr

[¶]Professor at Polytech'Lille - University of Lille, nouredine.melab@univ-lille1.fr

covariance function by convolving spatially-varying kernels. Paciorek *et al.* [5] extended this formulation to covariance functions that are positive definite in the Euclidean space and especially the Matern covariance function. However, these approaches are applicable only to a maximum of 3 dimensional problems [5]. Instead of a direct formulation, another method is to use local stationary covariance functions to model non-stationary functions. For instance, Haas [6] proposed a moving window approach where the training and prediction region move along the input space so the covariance function is considered stationary within this window, while Rasmussen and Gharmani [7] used different stationary GPs in different subspaces of the input space. These strategies present some limitations for the general case. First, in computationally expensive problems, data are sparsed and using a local surrogate model with sparser data may be problematic. Second, it may induce discontinuities at the boundaries of the subspaces. Another alternative strategy using non-linear mapping, has been introduced by Sampson and Guttorp [8] and consists in deforming the input space in order to model the non-stationary response by a stationary model. Following this work, Xiong *et al.* [9] proposed a sparse and flexible parametrization of the mapping function, by considering a piece-wise density function with parametrized knots, allowing to apply the non-linear mapping for high-dimensional problems.

This paper aims at investigating a new strategy to handle the non-stationary issue in EGO based on a promising surrogate modeling class that is Deep Gaussian Processes (DGP) [10] and to raise the challenges and opportunities induced by the coupling of EGO and DGP.

The paper is organized as follows. First, a description of EGO is presented with a review on stationary GP and classical infill criteria used (Section 2). Then, DGPs are introduced, and their advantages over simple GPs are highlighted (Section 3). Next, the challenges that occur when combining DGP with EGO are discussed and a first Deep Efficient Global Optimization framework (DEGO) is presented (Section 4). Finally, a comparison on analytical test cases of the proposed approach (DEGO) with non-linear mapping developed by Xiong and stationary Kriging is performed (Section 5).

2 Efficient Global Optimization

2.1 EGO framework

EGO is a Bayesian optimization algorithm dealing with expensive black-box optimization problems. It consists in sampling iteratively, using the prediction and uncertainty giving by the Kriging model, the most promising point based on an infill sampling criterion. This point is evaluated on the black box model and the surrogate-model is updated from the new training set, and a new point is sampled, and so on, until a stopping criterion is reached (Fig. 1). Hence, the two important aspects in EGO are the Kriging surrogate model and the infill sampling criterion.

2.2 Kriging surrogate model

A surrogate model is built from a set of points called training set or design of experiment (DoE) $\mathcal{X} = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}\}$ and their associated response values $\mathcal{Y} = \{y^{(1)}, ..., y^{(N)}\}$ where N is the number of observations and $\mathbf{x}^{(i)} \in \mathbb{R}^d$. Then, it is possible to use this model to predict the output at a new point. The interest of a surrogate model is its cheap computational evaluation cost, thus instead of evaluating the expensive black-box function for a large number of points, it is possible to build a surrogate-model using fewer evaluations of the exact function, and predict its value in the other points using this model.

The particularity of Kriging surrogate model is that it gives a variance estimation of the prediction, which makes it suitable in a global optimization framework.

A GP is used to describe a distribution over functions, it is a collection of infinite random variables, any finite number of which has a joint Gaussian distribution [11]. It is defined by its mean function and covariance function. In GP regression, a GP prior is placed on the unobserved function (or latent) $f(\cdot)$ with a prior covariance function $k_\Theta(\mathbf{x}, \mathbf{x'})$ that depends on a number of hyper-parameters Θ and due to the fact that the trend of the response is *a priori* unknown a constant mean function is considered (ordinary Kriging) *i.e.*, $f(\mathbf{x}) \sim \mathcal{N}(\mu, k_\Theta(\mathbf{x}, \mathbf{x'}))$. Therefore, $f(\cdot)$ has a multivariate distribution on any finite subset of variables, in particular in $\mathcal{X}$ *i.e.* $\mathbf{f}|\mathcal{X} \sim \mathcal{N}(\mathbf{1}\mu, \mathbf{K}^\Theta_{NN})$ where $\mathbf{K}^\Theta_{NN}$ is the covariance matrix constructed from the parametrized covariance function k on $\mathcal{X}$ (further the dependence on Θ is dropped for notation simplicity). The choice of the covariance function is important, because it determines our prior assumptions of the function to be modeled.

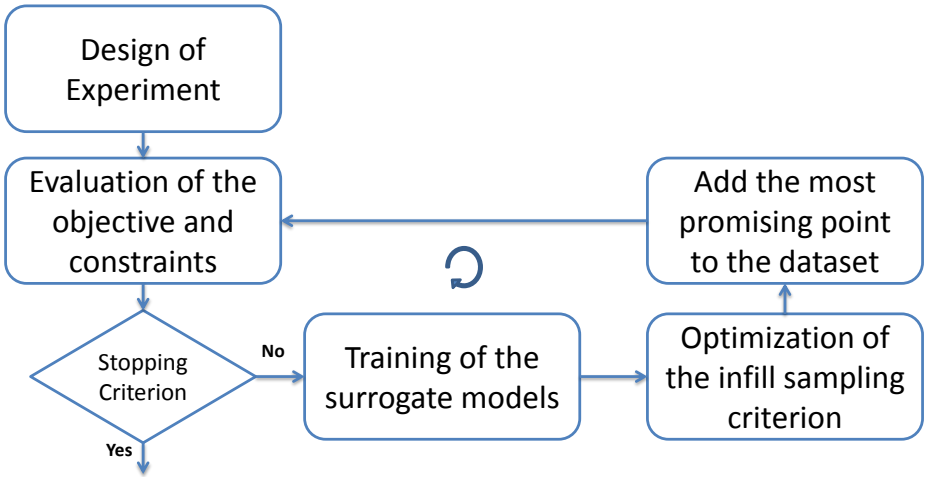

Figure 1: EGO framework

A Gaussian noise variance is considered, such that the relationship between the latent function values $f(\mathcal{X})$ and the observed response $\mathcal{Y}$ is given by : $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2\mathbf{I})$. The marginal likelihood is obtained by integrating out the latent function $f(\cdot)$:

$$p(\mathbf{y}|\mathcal{X}, \Theta) = \mathcal{N}(\mathbf{y}|\mu, \mathbf{K}_{NN} + \sigma^2\mathbf{I}) \tag{1}$$

Maximizing the marginal likelihood allows to train the GP by finding the optimal values of the hyper-parameters Θ, μ and σ. After the training, the prediction is made by considering a new point $\mathbf{x}^*$ and using the conditional properties of a multivariate normal distribution [11]:

$$p(y^*|\mathbf{x}^*, \mathcal{X}, \mathcal{Y}, \Theta) = \mathcal{N}(y^*|\hat{y^*}, \hat{s^*}^2) \tag{2}$$

with $\hat{y}^*$ the mean prediction and $\hat{s^*}^2$ the associated variance :

$$\hat{y^*} = \mu + \mathbf{k}_{\mathbf{x}^*}^T(\mathbf{K}_{NN}^{-1} + \sigma^2\mathbf{I})^{-1}(\mathbf{y} - \mathbf{1}\mu) \tag{3}$$

$$\hat{s^*}^2 = k_{\mathbf{x}^*\mathbf{x}^*} - \mathbf{k}_{\mathbf{x}^*}^T(\mathbf{K}_{NN}^{-1} + \sigma^2\mathbf{I})^{-1}\mathbf{k}_{\mathbf{x}^*} + \sigma^2 \tag{4}$$

where $k_{\mathbf{x}^*\mathbf{x}^*} = k(\mathbf{x}^*, \mathbf{x}^*)$ and $\mathbf{k}_{\mathbf{x}^*} = [k(\mathbf{x}^{(i)}, \mathbf{x}^*)]_{i=1,\dots,N}$.

2.3 Infill Sampling Criteria

Different criteria have been developed for selecting infill sample candidates [12]. They are based on a trade-off between exploration, by searching where prediction variance is high and exploitation by searching where prediction is minimized. The Probability of Improvement (PI) criterion samples the point where the probability of improving beyond the current minimum y_{min} is maximum. That is $PI(\mathbf{x}) = P\left[y \leq y_{min}\right] = \Phi\left(\frac{y_{min} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right)$, where $\Phi(\cdot)$ is the Gaussian Cumulative Distribution Function (CDF). The higher values of $PI(\mathbf{x})$ the higher chances that $\hat{y}(\mathbf{x})$ is better than y_{min}. The inconvenient of this criterion is that only the probability is taken into account and not how much a point may improve the current best. This will add a lot of points around the current best point. To overcome the inconvenience of the Probability of Improvement, the Expected Improvement (EI) takes into account the improvement induced by a candidate that is defined as: $I(\mathbf{x}) = \max\{0, y_{\min} - f(\mathbf{x})\}$. The EI is then computed as:

$$EI(\mathbf{x}) = \int t.\mathrm{PDF}_{I(\mathbf{x})}(t)\mathrm{d}t \tag{5}$$

$$= (y_{\min} - \hat{y}(\mathbf{x}))\Phi\left(\frac{y_{\min} - \hat{y}(\mathbf{x})}{\hat{s}}\right) + \hat{s}\phi\left(\frac{y_{\min} - \hat{y}(\mathbf{x})}{\hat{s}}\right) \tag{6}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ denote the Gaussian probability density function (PDF) and the Gaussian cumulative distribution function (CDF). The EI formula reveals two important terms. The first part is the same as in the 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.
For constrained optimization problems, the feasibility of the constraints must be taken into account. Different techniques exist [12], optimization of the sub-problem (maximizing an unconstrained infill criterion) under approximated constraints (Direct method), under the constraints of an expected violation (Expected violation method), or maximizing the product of an unconstrained infill criterion with the probability of feasibility of the constraints (Probability of Feasibility method) are strategies widely used.

3 Deep Gaussian Processes

3.1 Introduction and Bayesian Training

DGPs [10] are a class of surrogate models based on the structure of neural networks, where each layer is a GP. It considers that the statistical relationship between the inputs and the response is expressed by a functional composition of GPs :

$$y = f_L(\mathbf{f}_{L-1}(...(\mathbf{f}_1(\mathbf{f}_0(\mathbf{x}))) + \epsilon_L \tag{7}$$

where L is the number of layers and $\mathbf{f}_l(.)$ is an intermediate GP. Each layer is composed of an input node $\mathbf{h}_l$, an output node $\mathbf{h}_{l+1}$ and a GP $\mathbf{f}_l(.)$ mapping between the two nodes, getting the recursive equation: $\mathbf{h}_{l+1} = \mathbf{f}_l(\mathbf{h}_l) + \epsilon_l$ where ϵ_l is a Gaussian noise introduced in each layer (Fig. 2).

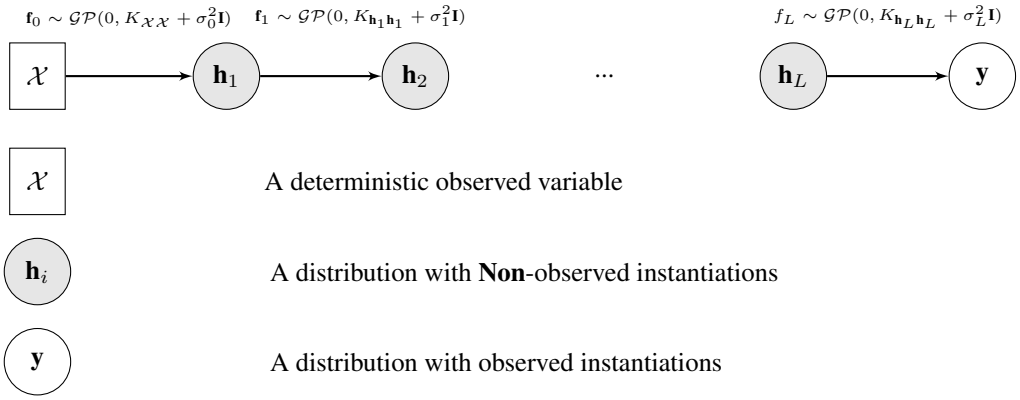

Figure 2: A representation of the structure of a DGP

The main difficulty induced by this cascade of GPs is that the intermediate nodes $\mathbf{h}_l$ are latent variables *i.e.* they are not observable as in a standard GP. Moreover, using non-linear covariance functions makes the overall composition no longer a GP. Hence, learning the hyperparameters of the model is challenging. To illustrate this, consider a DGP with one hiddden layer $\mathbf{h}_1$. In this configuration a DGP is equivalent to a Gaussian Process Latent Variable Model (GPLVM)[13]. The likelihood $p(\mathbf{y}|\mathcal{X})$ is given by integrating over the latent variables $\mathbf{h}_1$:

$$\begin{aligned} p(\mathbf{y}|\mathcal{X}) &= \int p(\mathbf{y}|\mathbf{h}_1)p(\mathbf{h}_1|\mathcal{X})\mathrm{d}\mathbf{h}_1 \\ &= \int \mathcal{N}(0, K_{\mathbf{h}_1\mathbf{h}_1} + \sigma_1^2\mathbf{I})\mathcal{N}(0, K_{XX} + \sigma_0^2\mathbf{I})\mathrm{d}\mathbf{h}_1 \end{aligned} \tag{8}$$

for simplicity $\mathbf{h}_1$ is taken one dimensional. The generalization to multi-dimensional hidden layer is done with the assumption of independence between dimensions. Unfortunately, the integrals of Gaussians with respect to non-linear kernel functions are analytically intractable. To overcome this issue, the variational Bayesian approach is used. It consists of approximating the posterior distribution of the latent variables $p(\mathbf{h}_1|\mathcal{Y})$ by a variational distribution $q(\mathbf{h}_1)$ which has a factorized Gaussian form over the latent variables $\prod_{i=1}^{N}\mathcal{N}(\mathbf{h}_1^{(i)}|\mathbf{m}_1^{(i)}, S_1^{(i)})$, where $\{\mathbf{m}_1^{(i)}, S_1^{(i)}\}_{i=1}^{N}$ are the variationnal parameters. Using Jensen's inequality, a first variational lower bound of the log likelihood is obtained:

$$\begin{aligned}\log p(\mathbf{y}|\mathcal{X}) &\geq \int q(\mathbf{h}_1)\log\frac{p(\mathbf{y}|\mathbf{h}_1)p(\mathbf{h}_1|\mathcal{X})}{q(\mathbf{h}_1)}\mathrm{d}\mathbf{h}_1 \\ &= \int q(\mathbf{h}_1)\log p(\mathbf{y}|\mathbf{h}_1)\mathrm{d}\mathbf{h}_1 - \int q(\mathbf{h}_1)\log\frac{p(\mathbf{h}_1|\mathcal{X})}{q(\mathbf{h}_1)}\mathrm{d}\mathbf{h}_1 \\ &= \tilde{F}(q) - KL(q||p)\end{aligned} \tag{9}$$

where the second term is the Kullback-Leibler (KL) divergence between the variational posterior distribution $q(\mathbf{h}_1)$ and the prior over the latent variables $p(\mathbf{h}_1) = \mathcal{N}(\mathbf{h}_1|\mathbf{0}, K_{XX})$. Since both distributions are Gussian, it is computed analytically. However, the first term $\tilde{F}(q)$ is still intractable, due to the fact that there is still the integration over non linear covariance function $K_{\mathbf{h}_1\mathbf{h}_1}$. To deal with this problematic, Titsias and Lawrence [13] introduced inducing variables that consists in augmenting with additional input-output pairs $\mathcal{Z} \in \mathbb{R}^{M\times d}, \mathbf{u} \in \mathbb{R}^M$, the latent space. Originally, inducing inputs were used as part of sparse GP to faster the GP regression $M << N$. The direct methods of sparse GP [14] modify the GP prior in order to use the inverse of the covariance matrix of the inducing inputs K_{MM} instead of K_{NN}. Titsias in [15] used the inducing inputs in a variational framework to avoid changing the GP prior in the sparse approximation. Titsias and Lawrence [13] used this same framework to overcome the intractability of $\tilde{F}(q)$. This approach consisted of augmenting the joint distribution with a distribution over the inducing variables $p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, K_{MM})$ to obtain :

$$\begin{aligned}p(\mathbf{y}|\mathbf{h}_1) &= \int_{\mathbf{u},\mathbf{f}_1} p(\mathbf{y},\mathbf{u},\mathbf{f}_1|\mathbf{h}_1)\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f}_1 \\ &= \int_{\mathbf{u},\mathbf{f}_1} p(\mathbf{y}|\mathbf{f}_1)p(\mathbf{f}_1|\mathbf{u},\mathbf{h}_1)p(\mathbf{u})\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f}_1\end{aligned} \tag{10}$$

The dependence on $\mathcal{Z}$ is dropped for notation simpicity. Then, the variational approach is used by approximating the true posterior of the latent variables $p(\mathbf{f}_1, \mathbf{u}|\mathbf{y}, \mathbf{h}_1) = p(\mathbf{f}_1|\mathbf{u}, \mathbf{h}_1, \mathbf{y})p(\mathbf{u}|\mathbf{y}, \mathbf{h}_1)$ with the variational distribution $q(\mathbf{f}_1, \mathbf{u}) = p(\mathbf{f}_1|\mathbf{u}, \mathbf{h}_1)q(\mathbf{u})$. The approximation $p(\mathbf{f}_1|\mathbf{u}, \mathbf{h}_1, \mathbf{y}) \approx p(\mathbf{f}_1|\mathbf{u}, \mathbf{h}_1)$ implies that $\mathbf{u}$ is a sufficient statistic for $\mathbf{f}_1$ which is true for an optimal set of $\mathcal{Z}$ and $\mathbf{u}$, and $q(\mathbf{u})$ is a free Gaussian distribution over the inducing variables. By using the same trick as in Eq.(9) a lower bound of $\log[p(\mathbf{y}|\mathbf{h}_1)]$ that depends on $\mathcal{Z}$, $q(\mathbf{u})$ and the hyperparamters Θ_1 of $\mathbf{f}_1$ is obtained. Then, by optimizing this lower bound analyticly according to $q(\mathbf{u})$ a tighter lower bound that depends only on $\mathcal{Z}$ and Θ_1 is obtained. By replacing $\tilde{F}(q)$ in Eq.(9) by this lower bound, an overall lower bound of the log likelihood $\log[p(\mathbf{y}|\mathcal{X})]$ that depends on $\mathcal{Z}$, Θ_0, Θ_1 and $q(\mathbf{h}_1)$ is obtained. This lower bound is computable analyticly for kernels that are feasibly convoluted with the Gaussian density $q(\mathbf{h}_1)$ such as the linear, the squared exponential and the Automatic Relevence Determination (ARD) squared exponential kernel.
The generalization to L number of layers consists in using the similar approximations in each layer with the assumption of independence between layers that is $q(\{\mathbf{h}_l\}_{l=1}^L) = \prod_{l=1}^L q(\mathbf{h}_l)$ [10]. Hence, in a DGP configuration a lower bound of the likelihood is maximized according to $\{\Theta_l\}_{l=0}^L$, $\{\mathcal{Z}_l\}_{l=0}^L$ and $\{q(\mathbf{h}_l) = \prod_{i=1}^N \mathcal{N}(\mathbf{h}_l|\mathbf{m}_l^{(i)}, S_l^{(i)})\}_{l=0}^L$.

3.2 Other Training Approaches

The main issue in the previous approach is the number of variational parameters $q(\mathbf{h}_l)$ that increases linearly with the number of training datapoints, which complicates the optimization of the approximated likelihood. To overcome this issue, Dai *et al.* [16] instead of considering the variational posteriors as individual parameters, take them as a transformation of observed data. Specificly, a recursive relationship links the variational parameters that is: $\mathbf{m}_0^{(i)} = g_1(y^{(i)})$ and $\mathbf{m}_l^{(i)} = g_l(\mathbf{m}_{l-1}^{(i)})$ where g_l is a multi-layer perceptron. This backprogration mechanism transforms the initialization of the variational parameters to the initialization of neural network parameters, which has been well studied in deep learning literature [17]. Furthermore, the variational parameters are moved coherently during the optimization process. Bui *et al.* [18] proposed a deterministic approximation for DGPs based on an approximated Expectation Propagation energy function, and a probabilistic back-propagation algorithm for learning. Salimbeni and Deisenroth [19] proposed a Doubly stochastic variational inference that does not assume the independence between layers and the form of the kernel functions, hence loosing the analytical tractability, that is bypassed through a crude Monte-Carlo sampling from the variational posterior.

3.3 Prediction

Once trained, the prediction using DGPs in a new point $\mathbf{x}^*$ uses either normality assumptions in each layer or sampling strategies. The first approach consists of assuming that the predicted distribution at the layer $l-1$ is Gaussian.

This Gaussian distribution is used as input in the layer l and the mean and covariance of the non-Gaussian output distribution at the layer l is obtained using the GPLVM prediction formula [13]. This distribution is then assumed to be Gaussian to use the GPLVM prediction formula at layer $l+1$ and so on until reaching the final layer. The second approach uses sampling strategies (crude Monte-Carlo sampling) along the layers *i.e.* using the mean and the covariance of the first layer, k samples are generated following a Gaussian distribution, then each sample is passed to the next layer undergoing a new transformation following a Gaussian, and so on until reaching the last layer. So, k values are obtained, the mean and standard deviation of these values are the predicted mean and its associated variance. The first method does not require sampling and then is faster than the sampling strategie approach, however it is based on assumptions which may impact the accuracy of the prediction.

3.4 Advantage of DGPs over GPs

Since DGPs are a composition of GPs with different stationary kernels, the overall process is no longer a GP allowing the capture of non-stationarity (Fig. 3, 4). Moreover it has been shown that DGPs handle scarce data and overcome the overfitting issue which is interesting in design optimization problem involving expensive black functions and induced uncertainty [19].

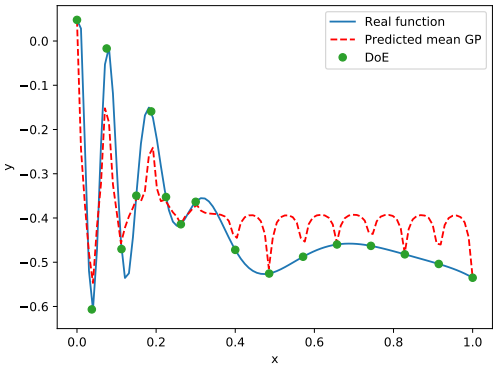

Figure 3: Regural GP

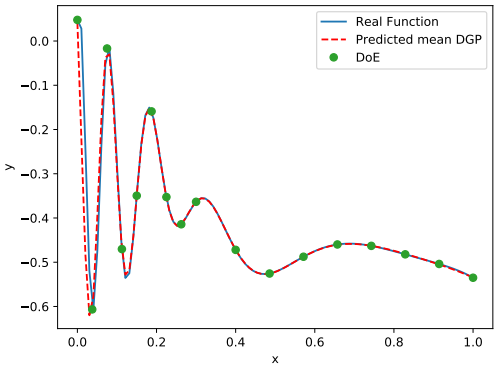

Figure 4: DGP

4 DGPs and EGO

In this section the coupling of EGO and DGPs is discussed, highlighting the different arising challenges and opportunities. In [16], an optimization experiment with DGPs was reported using the EI criterion. However, the combination EGO and DGP was not discussed, and as will be shown in this section the EI criterion cannot be used directly without prior assumptions.

4.1 Number of induced inputs

In EGO, there is an iterative procedure based on an infill criterion to add a point to the dataset and update the surrogate model. However, in DGPs the number of induced inputs is directly depending on the cardinal of the dataset. So, in DEGO the number of induced inputs may change over the iterations. A small number of inducing inputs allows computational speed ups in the construction of the model by using reduced size matrices and less optimization parameters, while a high number allows accuracy in the approximation. Therefore, a trade-off between the two has to be set. Since in computationally intensive engineering design problems, the dataset is not large, the construction of the model is relatively cheap. One approach is to set the number of induced points in each layer equal to the number of data points over the iterations, another approach to advantage even more the accuracy is to set the number of induced variables in each layer equal to the sum of the number of data points and the number of infill points. However, since EGO is an iterative update of the surrogate model, it may be interesting to investigate robust approaches to vary the number of induced inputs considering the result of previous iterations.

4.2 Number of layers

In DGPs, each layer transforms the output of the previous layer allowing the approximations of new features, and learning more complex representations. Experimentations with 2 to 5 layers were conducted in [19] and showed that the gain with increasing layers is achieved on very large dataset (one billion points). Since in engineering design problems, the dataset does not exceed the hundreds, two or three-layers configuration may be sufficient to catch the non stationarity of the model [19]. Nevertheless, an adaptive number of layers and of the dimension of a layer, to EGO may be a promising strategy to explore. Indeed, one can begin with a two layer configuration in the first iterations, then, after a certain number of iterations or based on a given criterion that expresses the accuracy of the surrogate model, switches to a higher number of layers.

4.3 Training the model

While in regular GP regression there is only kernel hyper-parameters to optimize in the training, in DGPs, in addition to kernel hyper-parameters in each layer, there are also the variational parameters. To faster the training, since in an EGO framework, the training is repeated in each iteration with an added point, it may be interesting after adding a consequent number of added points (to ensure a certain stability of hyper-parameters) to explore the use of the previous optimal configuration of the hyper-parameters as initialization for the next learning phase.

4.4 Infill Criterion

EGO is based on the balance between exploitation by searching where the value of the prediction is minimal, and exploration by searching where uncertainty is high. The EI is a widely used infill criterion. However, a direct application of the formula of the EI (Eq. 6) can not be computed analytically, since a DGP is not a Gaussian model. However, when the Gaussian assumption approach is used for prediction the direct formula of the EI can be used, since the approximated prediction is considered Gaussian. On the other hand if the sampling approach is used for prediction, then, the expected improvement $EI(\mathbf{x})$ is approximated using sampling on the value of the improvement $I(\mathbf{x})$. In the constrained case, the probability of feasibility or the expected violation can also be approximated using a sampling strategy, or using the cumulative distributive function of a Gaussian with the assumption that the whole model behaves as a Gaussian model. So, The question that arises is when the assumption of Gaussian behavior is valid and if not how many samples must be used for an accurate approximation without useless computational cost ?

5 Experimentations

In this section, experimentations on two analytical test problems are performed to compare between EGO using DGPs (DEGO), standard EGO, and EGO using Xiong's non-linear mapping (NLEGO). The first problem is an unconstrained one dimensionnal optimization problem, and the second one is a constrained two dimensionnal problem (A Python implementation is publicly available [20]).

- In standard EGO, an ARD p-exponential kernel [2] is used: $k(\mathbf{x}, \mathbf{x'}) = l * \exp\{-\sum_{i=1}^{D} \theta_i (x_i - x'_i)^{p_i}\}$. The learning of the hyperparameters (2 hyperparameters by dimension) is done with CMA-ES [21].

- In NLEGO an ARD p-exponential kernel is used with the integration of the mapping $g(\cdot)$: $k(\mathbf{x}, \mathbf{x'}) = l * \exp\{-\sum_{i=1}^{D} \theta_i (g(x_i) - g(x'_i))^{p_i}\}$. The learning of the hyper-parameters $(2 + k_i$ by dimension where k_i is the number of knots in dimension i) is done with CMA-ES.

- In DEGO, an ARD Gaussian kernel is used in each layer $k(\mathbf{x}, \mathbf{x'}) = l * \exp\{-\sum_{i=1}^{D} \theta_i (x_i - x'_i)^2\}$. A variational auto-encoded DGP with a 500 dimensionnal encoder by layer is used for learning the hyperparameters using l-BFGS-b optimization. The prediction is used with the Gaussian behaviour assumption. The learning and prediction of the DGP is performed using the toolbox PyDeepGP [22].

5.1 1D unconstrained problem

5.1.1 Objective function

The function to minimize is a one-dimensional non-stationary function presented in Equ. 11. It is a variant of the Xiong function [9] providing two-regions of interest in the minimization, one where the function varies with a high

frequency $x \in [0, 0.3]$ and the other where the function varies slowly $x \in [0.3, 1]$ (Fig. 3, 4)

$$f(x) = -0.5(\sin[40(x - 0.85)^4] \cos[2(x - 0.95)] + 0.5(x - 0.9) + 1), \; x \in [0, 1] \tag{11}$$

5.1.2 Results

20 initial DoE of five points are generated using a stochastic Latin Hypercube Sampling. 20 points are added using the EI criterion that is optimized with a differential evolution algorithm.
Table 1 displays the mean best value attained ("DEGO l HL qD dynamic/m" corresponds to DEGO with l-hidden q-dimensional layers and a number of inducing inputs that is equal to the size of the dataset at each iteration if dynamic and equal to m otherwise) with the corresponding variance and the percentage of time observed to attain the true optimum that is -0.60698. Standard EGO is the less performing algorithm, which is expected since it does not take into account the non-stationarity of the function. NLEGO gives good results, since the function is divided in two different regions, four knots catch easily the non-stationarity. DEGO with one hidden layer gives similar results as NLEGO when the number of inducing inputs change over iterations, while setting the number of inducing points to 25 gives even better results. This is explained by the fact that more inducing inputs leads to a more accurate variational approximation. Moreover, the variance of the results on the 20 repetitions is lower for DEGO 1HL, which make it robust than the other algorithms. However, adding just one other hidden layer to the configuration gives poor results, this is due to the over-fitting induced by this complex configuration for a one dimensional function.
The interesting point in this comparison is that even for one dimensional problems where the non linear mapping is efficient, DEGO outperformed it with an adequate configuration. However, it also highlights the fact that an inadequate configuration has important impacts on the result.
Fig. 5, 6 illustrate respectively an iteration where the best point maximizes the EI and the next iteration where it is added to the dataset.

Table 1: Performance of the algorithms

Algorithm	**The mean best value**	**Variance**	**% of success**
EGO	-0.5493	0.00073	15%
NLEGO 4 knots	-0.5716	0.00115	45%
DEGO 1HL 2D dynamic	-0.5717	0.00127	50%
DEGO 1HL 2D 25	**-0.581**	**0.00108**	**55** %
DEGO 2HL 2D 25	-0.546	0.00067	15 %

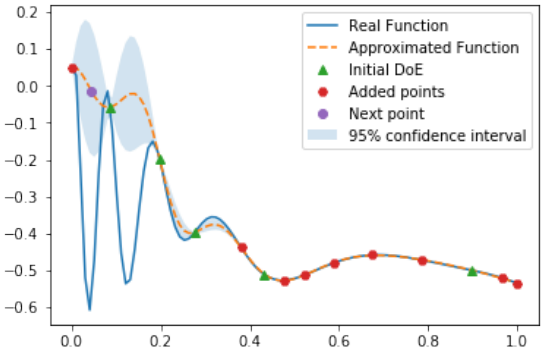

Figure 5: Best point maximizing the EI at iteration k for DEGO-2

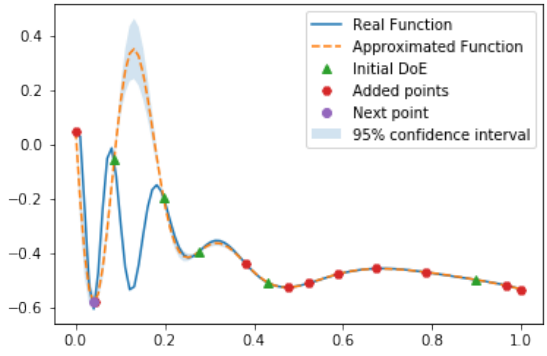

Figure 6: Best point added to the dataset at iteration $k + 1$ for DEGO-2

5.2 2D constrained problem

5.2.1 Objective function and constraint

The function to optimize is a simple two dimensional quadratic function: $f(x, y) = (x - 0.5)^2 + (y - 0.5)^2$. 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 (Fig. 7). Therefore, the problem is challenging for standard GP, since the optimal region is exactly at the boundary of the discontinuity, requiring an accurate modelisation of the non-stationarity.

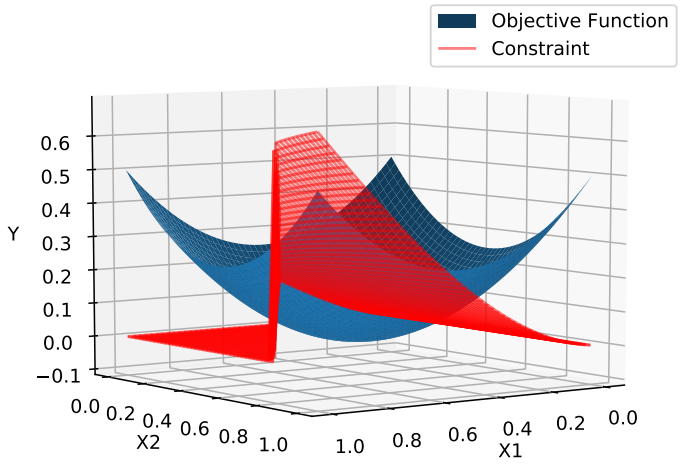

Figure 7: Objective and constraint functions 2D problem

5.2.2 Results

20 initial DoE of 15 points are generated using a stochastic Latin Hypercube Sampling. 20 points are added using the EI criterion and the expected violation criterion for the constraint with a threshold of 10^{-3}, and optimized with a differential evolution algorithm. Since the objective function is quadratic, a simple Gaussian process is used to approximate it in all the experimentations.

Table 2 displays the mean best value attained by each algorithm with the corresponding variance and the percentage of time that it attains the true optimum that is 0.0602. As in the first case standard EGO is not adapted to the problem, due to the discontinuity of the constraint. However, NLEGO does not performed as in the previous problem where its results were comparable to DEGO. This may be explained by two reasons. First, the non linear mapping is not well suited to the increase in the dimensionality of the problem, due to the fact that in the mapping the non-stationarity at a particular dimension may affect the other dimensions. Secondly, the non linear mapping can catch a change in the smoothness of the function in intervals but can not catch an abrupt change as the discontinuity in this constraint. DEGO with three layers and a dynamic number of induced points gives the better results. In contrast to standard GP, DEGO succeeds to capture the non stationarity of the model after adding a consequent number of points while classical EGO keeps a smooth modelisation and does not succeed to model accurately the discontinuity (Fig. 9, 10). The evolution of these algorithms according to the number of evaluations (Fig. 8) accentuates even more the superiority of DEGO and specially dynamic DEGO with 3 hidden layers. In fact, in the first iterations its speed of convergence is far more important than the other algorithms. Stopping the algorithms after adding 8 points would have given a more important gap between the different results. Another interesting aspect to observe is that the results given by DEGO are improved by adding hidden layers until reaching the three layers configuration. Adding more layers leads to a degradation of the results due to over-fitting (Fig. 11). Finally, unlike the first problem, setting the number of induced points to a maximum does not give better results. All these numerical experiments illustrate the importance of the settings of the DGP configuration in EGO, but the appropriate configuration may provide better results than regular GP and non linear mapping.

Table 2: Performance of the algorithms

Algorithm	The mean best value	Variance	% of success
EGO	0.09579	0.001034	5 %
NLEGO 8 knots	0.07956	0.000715	30 %
DEGO 1HL 10D dynamic	0.08699	0.000906	15 %
DEGO 2HL 10D dynamic	0.065342	$1.905\ 10^{-5}$	50 %
DEGO 3HL 10D dynamic	**0.06454**	**1.32** 10^{-5}	**75** %
DEGO 4HL 10D dynamic	0.06498	$1.22\ 10^{-5}$	60 %
DEGO 3HL 10D 35	0.066358	$1.92\ 10^{-5}$	45 %

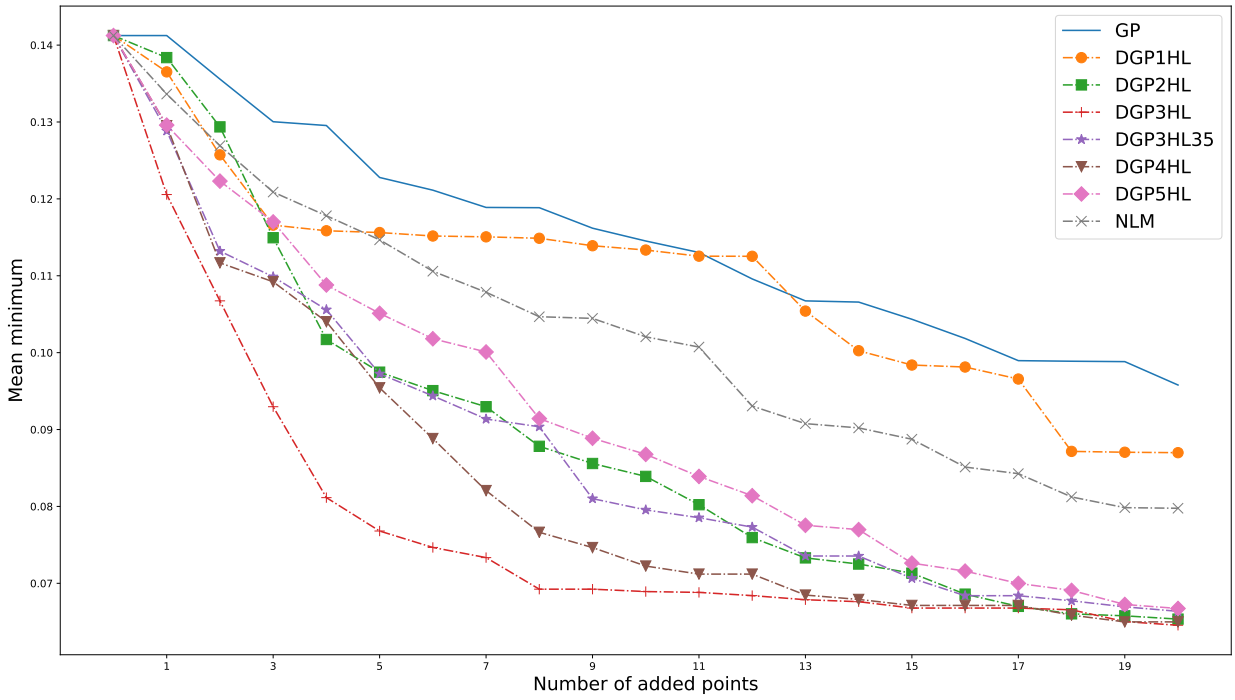

Figure 8: Evolution of the mean minimum according to the number of evaluations

6 Conclusions

The coupling of EGO and DGP is a promising strategy to deal with optimization problems including non-stationary functions. In this paper, the challenges arising from the adaptation of EGO to DGP have been highlighted, and propositions were suggested in order to make the coupling possible. Finally, numerical experimentations confirmed the interest of DEGO giving the promising results, and also the difficulty in choosing the adequate configuration of the network. Hence, the necessity to provide an adaptive framework to set the configuration of the DGP according to the dynamic of EGO and the dimensions of the problem. Future works are to give a complete parametrization of DEGO (e.g. setting the number of layers, the width of the layer, the number of inducing points) according to an optimization problem, and to adapt a parallel infill criterion, allowing multiple points to be added in each iteration of DEGO, and to use this adaptive algorithm in real world computationally expensive black-box problems.

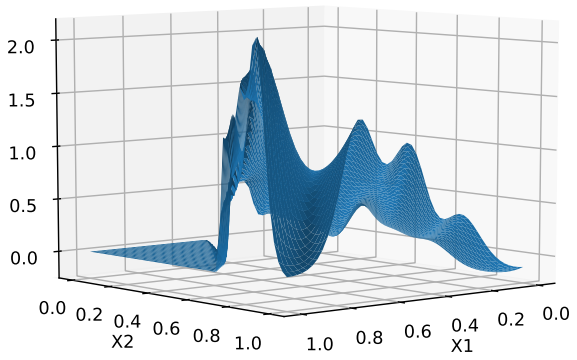

Figure 9: DGP approximation at the end of DEGO3 HL

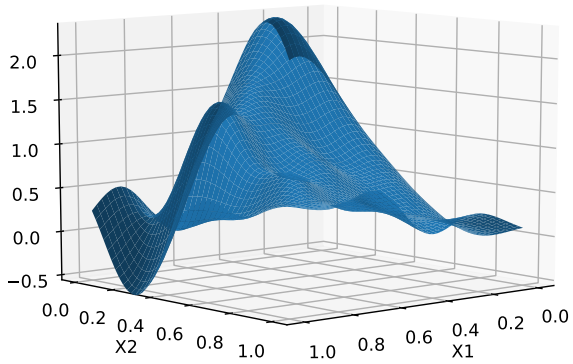

Figure 10: GP approximation at the end of EGO

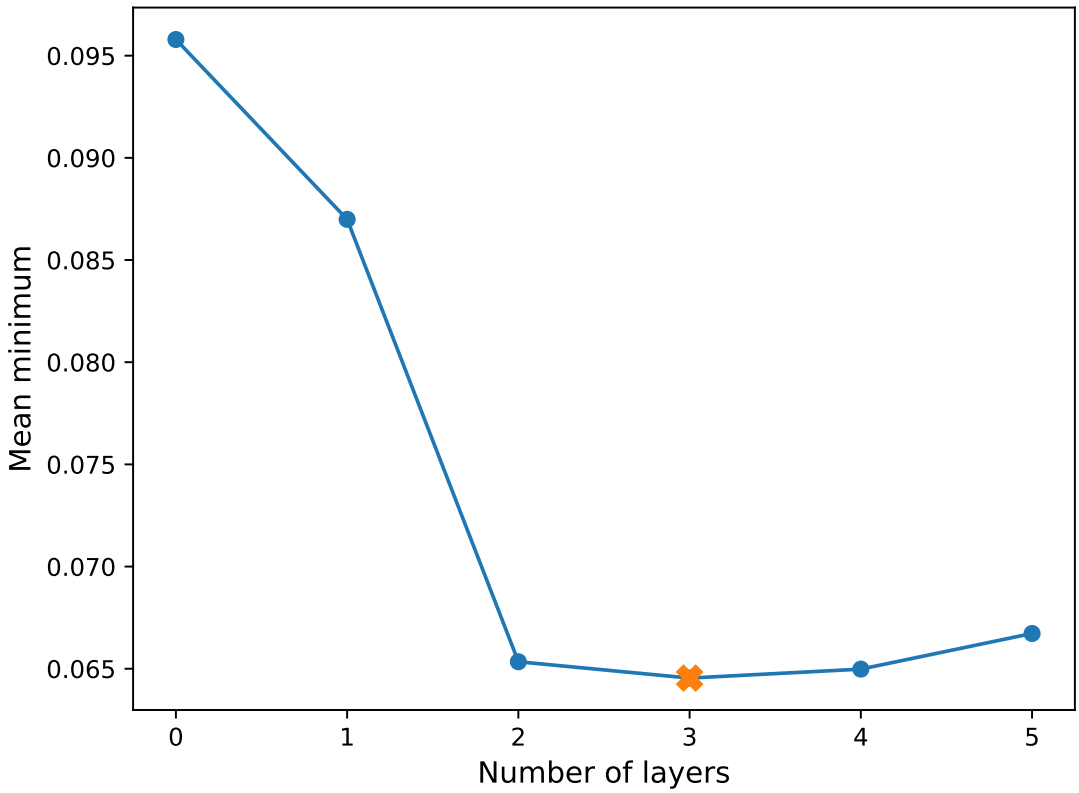

Figure 11: Evolution of the mean minimum according to the number of layers

7 ACKNOWLEDGMENT

This work was co-funded by ONERA-The French Aerospace Lab and Université de Lille/Inria Lille, in the context of a PhD thesis.

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] Michael L Stein. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media, 2012.

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

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

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

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

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

[9] 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.

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

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

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

[13] 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.

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

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

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

[17] Nico Weymaere and J-P Martens. On the initialization and optimization of multilayer perceptrons. *IEEE Transactions on Neural Networks*, 5(5):738–751, 1994.

[18] 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.

[19] Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes. *arXiv preprint arXiv:1705.08933*, 2017.

[20] DEGO. DEGO: The python implementation of efficient global optimization with deep gaussian processes. `https://github.com/M2CI-ONERA/M2CI-ONERA.github.io/tree/Deep-Gaussian-Process-EGO`, since 2018.

[21] Nikolaus Hansen. The cma evolution strategy: a comparing review. In *Towards a new evolutionary computation*, pages 75–102. Springer, 2006.

[22] PyDeepGP. PyDeepGP: The python implementation of deep gaussian processes. `https://github.com/SheffieldML/PyDeepGP`, since 2016.