Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

DEEP GAUSSIAN PROCESS METAMODELING OF SEQUENTIALLY SAMPLED NON-STATIONARY RESPONSE SURFACES

Vincent Dutordoir
Nicolas Knudde
Joachim van der Herten
Ivo Couckuyt
Tom Dhaene

Ghent University - imec
IDLab
iGent Tower - Department of Information Technology
Technologiepark-Zwijnaarde 15
B-9052 Ghent, BELGIUM

ABSTRACT

Simulations are often used for the design of complex systems as they allow to explore the design space without the need to build several prototypes. Over the years, the simulation accuracy, as well as the associated computational cost has increased significantly, limiting the overall number of simulations during the design process. Therefore, metamodeling aims to approximate the simulation response with a cheap-to-evaluate mathematical approximation, learned from a limited set of simulator evaluations. Kernel-based methods using stationary kernels are nowadays wildly used. However, using stationary kernels for non-stationary responses can be inappropriate and result in poor models when combined with sequential design. We present the application of a novel kernel-based technique, known as Deep Gaussian Processes, which is better able to cope with these difficulties. We evaluate the method for non-stationary regression on a series of real-world problems, showing that it outperforms the standard Gaussian Processes with stationary kernels.

1 INTRODUCTION

During the design and analysis of complex systems, computer simulations are often used to limit the number of expensive prototypes or real-life measurements. While adding more flexibility to study phenomena under controlled conditions, high-accuracy computer simulations often require a substantial investment of computational resources and time. A single simulation may easily take several hours or even days to complete. This is especially problematic for routine tasks such as optimization, sensitivity analysis and design space exploration. As a result, over the past decade, a lot of research has focused on alleviating these computational issues. An accomplished technique is the construction of a mathematical equation which approximates the behavior of the simulation as closely as possible, while being computationally cheap(er) to evaluate. If the approximation is based on a (limited) set of simulator evaluations, this is approximation is known as a metamodel (Kleijnen 2015, Gorissen et al. 2010). The metamodel may then be used instead of the computationally expensive simulator to derive characteristics of the problem at hand or to search for optima. More formally, we sample the expensive simulation N times at different locations $\boldsymbol{x}_i$ in order to get the simulation response $f_i \triangleq f(\boldsymbol{x}_i)$ and then collect these into a (training) dataset $\{\boldsymbol{x}_i, f_i\}_{i=1}^N$. Here we assume the simulation is deterministic and noise-free (with the exception of quantization errors due to the finite representations of real numbers). We subsequently use this dataset to construct a metamodel aiming to approximate the real simulation response surface as close as possible, so it can replace the simulation for prediction of new points $\boldsymbol{x}_*$ in the design space.

Many different supervised machine learning techniques can be used to obtain the metamodel. Among the most popular ones are Neural Networks (Fonseca et al. 2003), Kriging (Santner and Notz 2013, Forrester and Keane 2008), Gaussian Processes (GP) (Rasmussen and Williams 2006, Gorissen et al. 2010) and Least Squares - Support Vector Machines (LS-SVM) (Suykens and Vandewalle 1999). The latter three metamodels belong to the class of kernel-based regression methods, for which the prediction is a linear combination of kernel evaluations between the test and data points. This can be written as

$$f(\boldsymbol{x}_*) = \sum_{i=1}^{N} w_i \, k(\boldsymbol{x}_*, \boldsymbol{x}_i),$$

where k is the kernel used and N is the number of training data points. In general, there exists two broad classes of kernels: stationary and non-stationary kernels. A stationary kernel is one that only depends on the relative position of its two inputs, and not on their absolute location. A well-known and often used stationary kernel is the squared exponential (SE)

$$k_{SE}(\boldsymbol{r}) = \exp\left(-\sum_{d=1}^{D} \frac{r_d^2}{2\,\ell_d^2}\right),$$

where $\boldsymbol{r} = \boldsymbol{x}_* - \boldsymbol{x}_i$, D the input dimension and kernel hyperparameters ℓ_d define the characteristic lengthscale in each dimension.

GPs and Kriging models have a number of advantages which make them particularly attractive to use in metamodeling scenarios. The learning process involves optimization of a cost function which automatically guards against overfitting. This typically results in good generalization behavior. The methods have comparatively few hyperparameters (in most cases only a couple of hyperparameters which parametrize the kernel) which makes them favorable to use when the dataset is limited in size. It is also possible to get a full confidence estimate on the predictions, i.e. for every prediction we get the accompanying uncertainty of the model. This greatly supports the design of sequentially constructed datasets, discussed in Section 2.1.

One of the main difficulties of kernel-based methods is the selection of a suitable kernel, which should represent the particular properties of the response under study. Unfortunately, most of the properties of the response surface such as the smoothness, linearity or non-stationarity, are usually unknown at the start of the metamodeling process. If sufficient data is available and uniformly distributed over the input space this problem is solved by using a stationary kernel, such as the squared exponential, as they are universally applicable under these conditions. However, requiring a dataset to be sufficiently large and uniformly distributed is often not possible in metamodeling. In the context of metamodeling, constructing a large dataset is challenging as it involves numerous computationally expensive simulation evaluations. Furthermore, efficient metamodeling procedures frequently make use of sequential sampling methodologies which may result in non-uniform distributions (Crombecq et al. 2011). As illustrated in this work, the use of stationary kernels under these conditions can result in poor models as it becomes impossible to approximate the response surface using a single set of lengthscales.

In this paper we use a recent regression technique, known as Deep Gaussian Processes (DGPs) (Damianou and Lawrence 2013, Hensman and Lawrence 2014, Bui et al. 2016), to address this problem. A DGP is a cascade of multiple GPs, where each node acts as output of the layer above and as input for the layer below. Due to this hierarchical composition of GPs, DGPs are able to model highly non-linear and non-stationary response surfaces, even when only using a stationary kernel in each node. We will use the DGP to metamodel non-stationary response surfaces and show that it is an attractive method for metamodeling combined with sequential design.

The remainder of the paper is organized as follows. In Section 2 we introduce metamodeling with sequential design. Subsequently, we focus on Gaussian Processes, sparse Gaussian Processes and Deep Gaussian Processes. We discuss the models' fundamentals, their applicability in metamodeling and we identify their limitations. In Section 3 and 4 we empirically show the benefits of using Deep Gaussian Processes over other metamodeling techniques by conducting a series of experiments. We conclude the paper in Section 5.

2 METAMODELING: BACKGROUND

2.1 Sequential Design

A crucial step in the metamodeling process is the selection of the data points, i.e. inputs and corresponding simulation evaluations, to build the metamodel. During the selection procedure, there are two counter-working forces. On the one side, we try to keep the dataset as small as possible in order to limit the number of expensive simulation evaluations. On the other side, it is of paramount importance that the dataset is large and informative enough in order to build an accurate model that covers the complete design space. In traditional Design of Experiments (DoE) the dataset is selected upfront through a one-shot approach such as factorial designs (Lehmensiek et al. 2002) and optimized Latin hypercubes (Park 1994).

As these selection procedures do not take into account any prior knowledge about the simulation response surface at hand they make an arbitrary choice in the number of samples required. This can lead to a metamodel constructed with too few samples (undersampling) which has a displeasing performance. On the contrary, selecting too many samples (oversampling) results in wasting computational resources and increases the learning time. As a solution, sequential design of the dataset is nowadays often used to improve the metamodeling efficiency and accuracy. In sequential designs the one-shot approach is turned into an iterative process where the data acquired from previous iterations is analyzed in order to intelligently select locations for new data points. In addition to preventing under/over sampling, the iterative procedure can be used to halt the construction of the metamodel once the stopping criteria are met, avoiding unnecessary simulator evaluations. The complete procedure is as follows. The process starts by selecting a small initial dataset through a one-shot approach. Then, the metamodel is constructed and the stopping criteria (e.g., Is the metamodel sufficiently accurate according to a quality criterion? Has the time limit or total number of allowed evaluations been exceeded?) are evaluated. When the criteria are not met, new points are intelligently selected in demanding regions and the procedure restarts. This is illustrated in Figure 1.

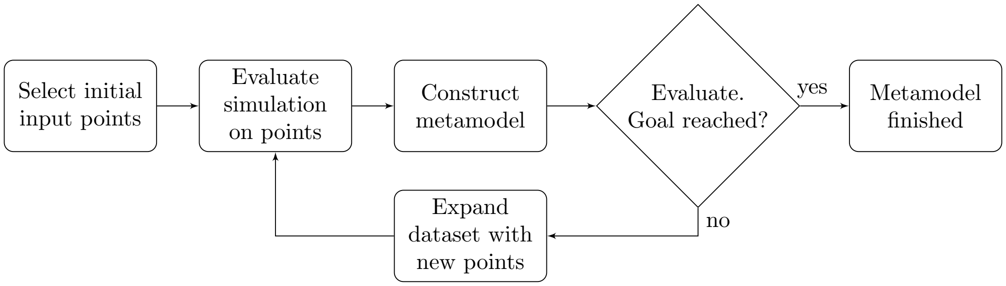

Select initial input points
Evaluate simulation on points
Construct metamodel
Evaluate. Goal reached?
yes
Metamodel finished
no
Expand dataset with new points

Figure 1: The sequential design metamodeling paradigm.

FLOLA-Voronoi (van der Herten et al. 2015) is a frequently used algorithm for the sequential selection of datapoints. It identifies highly irregular and non-linear regions in the response surface and will accordingly increase the number of samples in these regions. By doing this, it assists the metamodel in approximating the more difficult regions while limiting the number of expensive simulator evaluations needed. The algorithm will be used in the reported experiments.

2.2 Gaussian Processes

Gaussian Processes are a well-established class of metamodels, which are popular due to their analytical tractability, a training procedure which automatically guards against overfitting and their ability to provide a full predictive distribution over the outputs. A GP is formally defined by Rasmussen and Williams (2006) as any distribution over functions such that any finite set of function values have a joint multivariate Gaussian distribution. A GP is fully specified by its mean function $m(\boldsymbol{x})$ and its covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$. The covariance or kernel function k is chosen upfront and defines the correlation between the input vectors by assuming some properties such as smoothness, periodicity, trends or bias. It is usually parametrized by a limited set of kernel hyperparameters $\boldsymbol{\theta}_k$. Constructing the GP metamodel (as in the 3$^{\text{rd}}$ step of Figure 1) consists of finding the optimal set of hyperparameters $\boldsymbol{\theta}_k$. This is done by optimizing the likelihood function $p(\boldsymbol{y}|X, \boldsymbol{\theta}_k)$ in order to "fit" the data. Denoting the training input data as $X = \{\boldsymbol{x}_i\}_{i=1}^N$ and collecting the noisy simulation evaluations in $\boldsymbol{y}$, i.e. $y_i = f_i + \epsilon_i$ with $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, this can be written as the pdf of a multivariate normal distribution

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}_k) = (2\pi)^{-\frac{N}{2}} |\boldsymbol{K_{ff}} + \sigma^2 \boldsymbol{I}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} \boldsymbol{f}^\top (\boldsymbol{K_{ff}} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{f}\right), \tag{1}$$

where $\boldsymbol{\theta}_k$ is added to denote the dependence of the kernel parameters and where $\boldsymbol{K_{ff}}$ is a covariance matrix storing the correlation between every pair of training datapoints $(\boldsymbol{K_{ff}})_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Although σ^2 is typically chosen to be zero in metamodeling (as we assume that there is no noise on the simulation evaluations) we introduced it as it will be convenient in later models.

Once the optimal hyperparameters for the dataset are determined, the model can be used to predict the response at a new point in the design space $\boldsymbol{x}_*$. In fact, GPs are able to provide a full predictive distribution for the simulation evaluation f_*, which is again characterized by a Gaussian distribution

$$f_* | \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}_* \sim \mathcal{N}(\mu_*, \sigma_*^2), \tag{2}$$

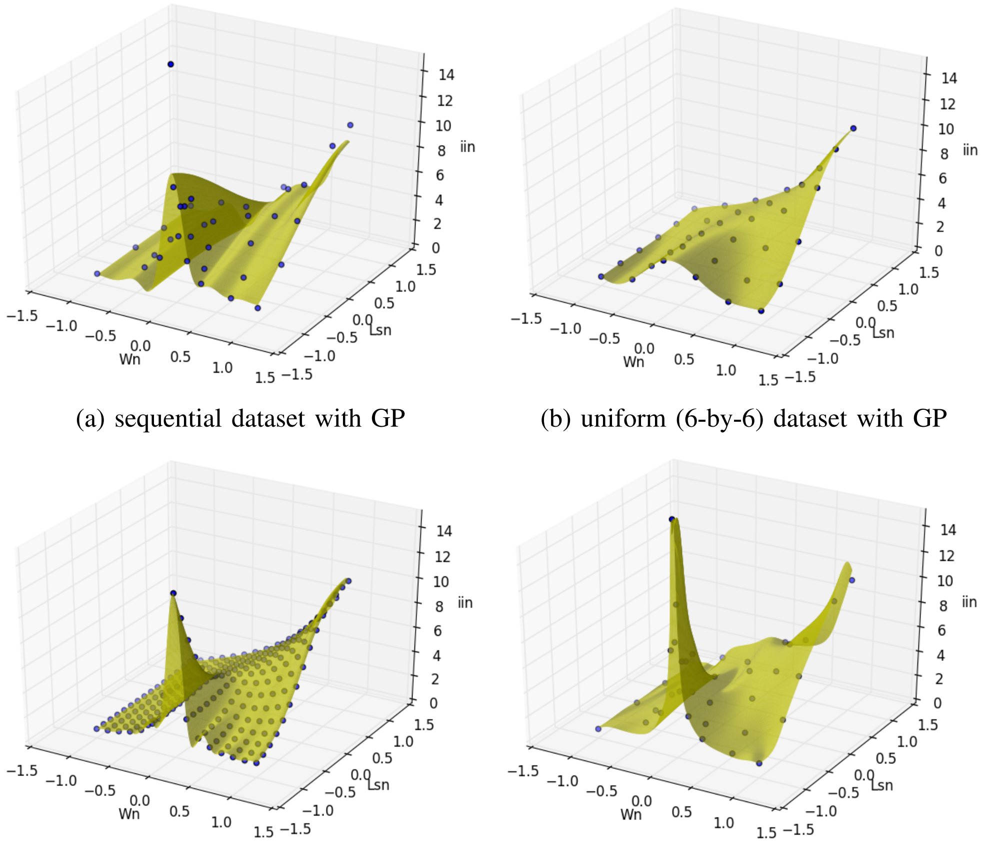
with

$$\begin{aligned} \mu_* &= \boldsymbol{k}_*^\top (\boldsymbol{K_{ff}} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{f}, \\ \sigma_*^2 &= k_{**} - \boldsymbol{k}_*^\top (\boldsymbol{K_{ff}} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}_*, \end{aligned}$$

where $k_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$ and $\boldsymbol{k}_* = k(\boldsymbol{X}, \boldsymbol{x}_*)$ stands for the vector $(N \times 1)$ of covariance evaluations between the single input point $\boldsymbol{x}_*$ and all the training input points.

The advantage of obtaining a complete predictive distribution for new test points is that the mean μ_* can conveniently be used as a point estimation of the simulation response and that the variance σ_* can be used to quantify the uncertainty of the model at that point. The downside, however, is that the calculation of the likelihood (1), as well as the predictive distribution (2) are of the order $\mathcal{O}(N^3)$ due to the need to compute the Cholesky decomposition for inversion of $\boldsymbol{K_{ff}}$. This makes the usage of GPs problematic when the dataset becomes very large.

The choice of kernel is crucial as it encodes the properties of the response surface, which is usually unknown a priori in metamodeling. Frequently, stationary kernels such as the popular SE kernel are used successfully. However, in this work it is shown how this type of kernels applied to non-stationary problems can be troublesome in combination with sequential design. Consider, for example, Figure 2a, showing a sequentially designed dataset of a Low Noise Amplifier (LNA) (Gorissen et al. 2009) and the resulting optimized standard GP model, using a stationary SE kernel. The sequential dataset has a much higher sampling density in the irregular regions of the response surface (e.g. near the peak) than in the flatter regions. Unfortunately, the approximating GP has oscillating behavior in undersampled regions and ignores the sample on the peak. The problem is mainly due to the fact that GPs, using stationary kernels, are only able to detect a single lengthscale ℓ_d for each dimension. However, for the given data on this non-stationary response surface short lengthscales are required near the peak whereas longer lengthscales are needed for the smooth surroundings. Here, the likelihood optimization resulted in a trade-off. For the peak the lengthscale is still too broad causing the sample on the peak to be ignored. For the surrounding regions on the other hand, the lengthscales are too short and the GP reverts to predicting the mean between the data points, resulting in fluctuations.



(a) sequential dataset with GP

(b) uniform (6-by-6) dataset with GP

(c) uniform (15-by-15) dataset with GP

(d) sequential dataset with DGP

Figure 2: Illustration of the impact of different sampling strategies on the resulting GP metamodel of a Low Noise Amplifier (LNA) dataset (Gorissen et al. 2009).

On the contrary, when using uniform sampling schemes (e.g. Latin Hypercubes) the fitting of the non-stationary response surface is satisfactory, even when the model uses a stationary kernel. Under this setting, a single set of lengthscales is enough and the GP acts as a local interpolation. However, in order to represent the highly irregular response surface with a uniform grid, a dense training set is required, ensuring us that even sharp non-linear behavior is represented. This is illustrated in Figure 2b and 2c, where two different uniform grids and the resulting GP models are shown: the model of 2b is constructed using a 6-by-6 undersampled grid and the model of 2c is constructed using a 15-by-15 oversampled grid. The first dataset is clearly less representative as the high peak at $(0, 0)$ is only slightly present. Although this problem can be solved partially using a denser grid, as is done in 2c, gathering such a grid is significantly more expensive in the context of metamodeling as it involves many additional computationally expensive simulation evaluations, and the top of the peak is still missed. The problem becomes even more troublesome for high dimensional systems due to the fact that in order to keep the sampling density at the same level, we need to increase the number of samples exponentially as a consequence of the "curse of dimensionality". Even more, increasing the number of samples in the dataset does not only increase the time required to construct the dataset, it also increases the time required to optimize the model hyperparameters, which scales $\mathcal{O}(N^3)$, with N the number of training samples. However, this problem can be circumvented by using sparse GP approximations instead of standard GPs as they retain the favorable properties of GPs but at a lower computational cost.

2.2.1 Sparse Approximations

The main line of work in the literature attempting to overcome the computational bottleneck of standard GPs, namely $\mathcal{O}(N^3)$, is related to sparse approximations. In Sparse Gaussian Processes (SGPs) the focus lies on augmenting the GP model by introducing an extra set of pseudo output variables $\boldsymbol{u}$ and their corresponding pseudo inputs $\boldsymbol{Z}$ (Csató and Opper 2002, Seeger et al. 2003, Snelson and Ghahramani 2006, Titsias 2009). The pseudo inputs and outputs are assumed to be jointly Gaussian with the original function $\boldsymbol{f}$, which allows us to write

$$p(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{Z}, \boldsymbol{X}) = \mathcal{N}\left(\begin{bmatrix}\boldsymbol{f}\\ \boldsymbol{u}\end{bmatrix}\middle| \boldsymbol{0}, \begin{bmatrix}\boldsymbol{K_{ff}} & \boldsymbol{K_{fu}}\\ \boldsymbol{K_{uf}} & \boldsymbol{K_{uu}}\end{bmatrix}\right).$$

Introducing independent Gaussian noise on the simulation evaluations again, we can write the joint density for the data and the augmented latent variables as

$$p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{y}|\boldsymbol{f}).$$

Titsias (2009) shows that this equation can be further derived, through variational compression (VC), to lower bound the log marginal likelihood as

$$\log p(\boldsymbol{y}|\boldsymbol{u}, \boldsymbol{Z}, \boldsymbol{X}) \geq \mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{Q_{ff}} + \sigma^2\boldsymbol{I}) - \frac{1}{2\sigma^2}\operatorname{tr}\left(\boldsymbol{K_{ff}} - \boldsymbol{Q_{ff}}\right), \tag{3}$$

where $\boldsymbol{Q_{ff}} = \boldsymbol{K_{fu}}\boldsymbol{K_{uu}^{-1}}\boldsymbol{K_{uf}}$. The chief attribute of sparse approximations is thus to approximate the marginal likelihood $p(\boldsymbol{y}|\boldsymbol{u}, \boldsymbol{Z}, \boldsymbol{X})$ by replacing the $N \times N$ covariance matrix $\boldsymbol{K_{ff}}$ in (1) by a low-rank matrix $\boldsymbol{Q_{ff}}$. As the approximation is based on a set of M pseudo input - output pairs, with $M \ll N$, the computational complexity of the model is lowered. However, the quality of the sparse model now highly depends on the representational capacity of the pseudo dataset. As we will see in the experiments, the SGPs are able to handle large datasets more easily by being computationally very efficient but have to comprise on accuracy in doing so. Nonetheless, they are a fundamental building block for the Deep Gaussian Processes, used in this work.

2.3 Deep Gaussian Processes

Deep Gaussian Processes (DGPs) (Damianou and Lawrence 2013, Hensman and Lawrence 2014, Bui et al. 2016) are a probabilistic model based on the composition of processes. Rather than assuming that a simulation evaluation $\boldsymbol{y}$ is drawn directly from a single Gaussian Process, we assume that it is drawn from a functional composition of GPs which is corrupted by noise ϵ, this is written as follows

$$\boldsymbol{y} = \boldsymbol{f}_L(\boldsymbol{f}_{L-1}(\dots\boldsymbol{f}_2(\boldsymbol{f}_1(\boldsymbol{x}))) + \boldsymbol{\epsilon},$$

where every function $\boldsymbol{f}_\ell$ is itself drawn from a Gaussian Process and $\boldsymbol{\epsilon}$ is an error term which holds the propagation of white Gaussian noise terms, $\boldsymbol{\epsilon_\ell} \sim \mathcal{N}(0, \sigma_\ell^2\boldsymbol{I})$, added to every layer. Denoting $\boldsymbol{h}_\ell = \boldsymbol{f}_\ell(\boldsymbol{h}_{\ell-1}) + \boldsymbol{\epsilon}_\ell$ it is possible to write the complete joint probability

$$p(\boldsymbol{y}, \{\boldsymbol{h_\ell}\}_{\ell=1}^{L-2}|\boldsymbol{X}) = p(\boldsymbol{h}_1|\boldsymbol{X})\left(\prod_{\ell=2}^{L-2} p(\boldsymbol{h}_\ell|\boldsymbol{h}_{\ell-1})\right)p(\boldsymbol{y}|\boldsymbol{h}_{L-2}). \tag{4}$$

The graphical model of a DGP is shown in Figure 3. After renaming $\boldsymbol{X}$ and $\boldsymbol{Y}$ as $\boldsymbol{H}_0$ and $\boldsymbol{H}_{L-1}$, respectively, every probability in (4) can be written as a single Gaussian Process mapping with mean zero and a covariance matrix $\boldsymbol{K}_{\boldsymbol{h}_\ell \boldsymbol{h}_\ell}$

$$p(\boldsymbol{h}_\ell | \boldsymbol{h}_{\ell-1}) = \mathcal{N}(\boldsymbol{h}_\ell | \boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{h}_\ell \boldsymbol{h}_\ell}).$$

As was the case for standard GPs, we want to obtain a marginal likelihood to estimate the hyperparameters. Subsequently, once the model is optimized we want to infer a posterior distribution for new test points. Due to the non-linear dependency between the layers in DGPs inference becomes analytically intractable and approximating techniques are required. In this derivation we use the nested variational approach, proposed by Hensman and Lawrence (2014) as it is computationally efficient and scales appropriately to large datasets. The approach starts with augmenting each layer with inducing inputs $\boldsymbol{Z}$ and outputs $\boldsymbol{U}$ and then uses variational compression, like in sparse GPs (Section 2.2.1). The extra datasets, $\boldsymbol{Z}$ and $\boldsymbol{U}$, are also shown in the graphical model. Now, it is possible to apply variational compression by substituting the result of (3) in every mapping of (4). The last step is to introduce an additional variational distribution over the output distributions of the inducing points and to marginalize out the hidden layers $\boldsymbol{h}$. We refer the interested reader to Damianou and Lawrence (2013) and Hensman and Lawrence (2014) for a rigorous derivation. The final bound is given by:

$$\begin{aligned}
\log p(\boldsymbol{y}|\boldsymbol{X}) \geq & -\frac{1}{2\sigma_1^2}\mathrm{tr}\left(\boldsymbol{K}_{\boldsymbol{h}_1\boldsymbol{h}_1} - \boldsymbol{K}_{\boldsymbol{h}_1\boldsymbol{u}_1}\boldsymbol{K}^{-1}_{\boldsymbol{u}_1\boldsymbol{u}_1}\boldsymbol{K}_{\boldsymbol{u}_1\boldsymbol{h}_1}\right) - \sum_{\ell=1}^{L-1}\mathrm{KL}\left(q(\boldsymbol{u}_\ell)\|p(\boldsymbol{u}_\ell)\right) \\
& - \sum_{\ell=2}^{L}\frac{1}{2\sigma_\ell^2}\left(\psi_\ell - \mathrm{tr}\left(\boldsymbol{\Phi}_\ell \boldsymbol{K}^{-1}_{\boldsymbol{u}_\ell\boldsymbol{u}_\ell}\right)\right) \\
& - \sum_{\ell=2}^{L-1}\frac{1}{2\sigma_\ell^2}\mathrm{tr}\left((\boldsymbol{\Phi}_\ell - \boldsymbol{\Psi}_\ell^\top\boldsymbol{\Psi}_\ell)\boldsymbol{K}^{-1}_{\boldsymbol{u}_\ell\boldsymbol{u}_\ell}\langle \boldsymbol{u}_\ell\boldsymbol{u}_\ell^\top\rangle_{q(\boldsymbol{u}_\ell)}\boldsymbol{K}^{-1}_{\boldsymbol{u}_\ell\boldsymbol{u}_\ell}\right) \\
& + \log\mathcal{N}(\boldsymbol{y}|\boldsymbol{\Psi}_{L-1}\boldsymbol{K}^{-1}_{\boldsymbol{u}_{L-1}\boldsymbol{u}_{L-1}}\boldsymbol{m}_{L-1}, \sigma_{L-1}^2\boldsymbol{I}),
\end{aligned}$$

where ψ_ℓ, Ψ_ℓ and $\boldsymbol{\Phi}_\ell$ are expectations under the variational distribution $q(\boldsymbol{u}_{\ell-1})$ of the covariance matrices $\boldsymbol{K}_{\boldsymbol{h}_\ell\boldsymbol{h}_\ell}$ and $\boldsymbol{K}_{\boldsymbol{u}_\ell\boldsymbol{h}_\ell}$, which can be computed analytically for some kernels, such as the SE and the linear one.

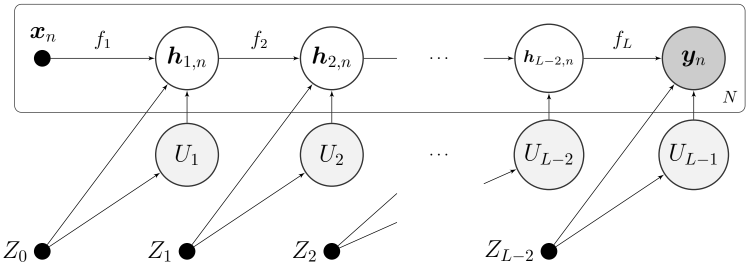


Figure 3: Graphical model of a generic DGP with L layers: one input layer ($\boldsymbol{x}$), $L-2$ hidden layers ($\boldsymbol{h}$'s) and one observed layer ($\boldsymbol{y}$).

In Figure 2d we see the result of using a 5-layer DGP metamodel for the sequentially sampled, non-stationary response surface. Note that the DGP is using only the stationary SE kernel, like the other GP models, but is able to model the surface much more accurately. This is due to the composition of the multiple stacked GPs, each having their own kernel with appropriate lengthscales. Every layer of the model is able to learn a specific part of the pattern which results in a model that is able to accurately approximate the highly irregular region and the flatter regions around the peak simultaneously. Furthermore, we see that by combining DGPs with sequential sampling we are able to construct a regression model that can capture the highest value at the peak. In the uniform sampling case (Figure 2c), the training set did not contain this value so the model was not aware of the existence and consequently did not model it. On the contrary, in the case of Figure 2a, the sample at the top of the peak is present in the training dataset but the standard GP is not able to model it, as it would require different lengthscales to do so. Note that Figure 2a and Figure 2d are both based on the same sequentially sampled dataset.

3 EXPERIMENTAL SETUP

The GP models used to conduct the experiments are all constructed using GPflow (Matthews et al. 2016), a Gaussian Process library that uses TensorFlow for its core computations and Python for its front-end. We report the performance of four different models implemented in the package, a standard GP model (GP), a variational sparse GP (SGP), a scalable version of the variational sparse GP (SVGP) and an implementation of DGP. We configured the models with the same stationary SE kernel with Automatic Relevance Determination (ARD) lengthscales and the variational models all have the same number of inducing points. For the DGP model, the inducing point locations and the lengthscales of the first GP layer are chosen to correspond with the centers of k-means clustering and the median distance between the input points of the training dataset, as suggested by Bui et al. (2016). The DGP is trained in two stages. First, the locations of the inducing points are fixed and only the kernel hyperparameters are optimized. Then, all the hyperparameters, i.e. the locations of the inducing points and the kernel hyperparameters, are optimized jointly. This routine was found empirically to be much less prone to local minima as we assist the optimizer by limiting the number of optimizable variables in the first stage. The performance of the models is assessed using three metrics: Root-Mean-Square-Error (RMSE), Root-Relative-Square-Error (RRSE) and the Mean-Square-Error (MSE), which is the square of the RMSE (Li and Zhao 2006):

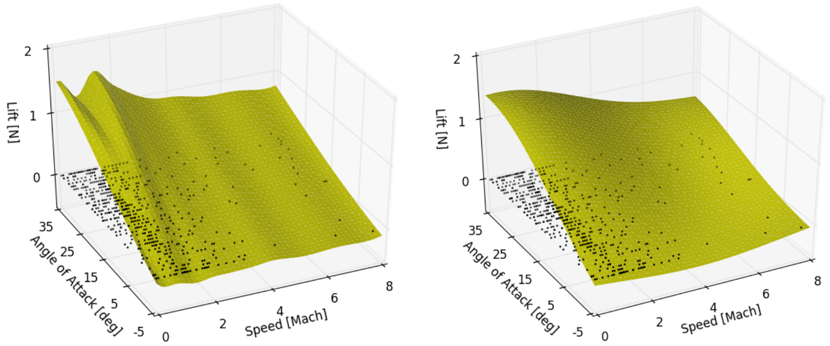
$$\text{RMSE}(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \tilde{y}_i)^2}{N}}, \quad \text{RRSE}(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \tilde{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}}$$

4 RESULTS

4.1 The Langley Glide-Back Booster Simulation

The Langley Glide-Back Booster (LGBB) (Gramacy and Lee 2008, Gorissen 2010) is a proposed rocket booster designed by NASA. A rocket booster is an instrument which assists the space shuttle during the launch process and then parachutes back to Earth once the fuel has been exhausted. The trajectory of a returning rocket booster is carefully planned so that they fall into the ocean, where they can be recovered and reused for subsequent missions. The design of such an LGBB heavily relies on simulators as wind tunnel experiments are still not fully realistic and quite expensive. More particularly, the re-entering of the rocket booster into the Earth's atmosphere is one of the most critical moments of the trajectory. Physically recreating such an environment is almost infeasible and hence simulations are turned to as they form an attractive and cheaper alternative. The particular simulator of a LGBB models the forces applied on the vehicle when it is re-entering the atmosphere. It is based on a computational fluid dynamics system and solves for approximately 1.5 million points the relevant Euler equations. One run of the simulator, for a single set of input parameters, can easily take between 5 to 20 hours on NASA's supercomputers. Therefore, NASA was interested in constructing a metamodeling of the simulation as it allows them to steer the design into promising directions and identify potential problems at an early stage. For this kind of critical applications Kriging and GP metamodels can be highly valuable as they are able to provide the uncertainty for each prediction.

The two most dominant input parameters of the simulator are the angle of attack (deg) and the speed at re-entry (Mach). The output of the simulator are the six relevant forces that can be applied on a free body in space: lift, drag, pitch, side-force, yaw, and roll. However, in the case of space-vehicle design, we only focus on the lift force, as that is the most important one for keeping a vehicle aloft. Figure 4a shows the lift response plotted as a function of speed and angle of attack with the other parameters fixed at a constant value. The large ridge at 1 Mach separates subsonic from supersonic cases. This transition from subsonic to supersonic is distinctly non-linear and non-continuous while the other parts of the surface are quite smooth. The scattered dots on the plot correspond to the input locations of the given dataset (i.e. the points where the simulation is evaluated) which should be used to construct the metamodels. We observe that this dataset is constructed using a sequential technique as the sampling density in non-stationary regions is significantly higher. Therefore, we expect (sparse) GPs with stationary kernels to have issues approximating this response surface.



(a) DGP

(b) GP

Figure 4: Two metamodels which are constructed using the same dataset (scattered dots) of the LGBB response: in (a) the DGP's predicted mean of the LGGB's lift response in function of the speed and the angle of attack and in (b) the GP's predicted mean. The scattered dots show the non-uniformly distributed, 2D input dataset.

Table 1 summarizes the test error metrics of the different optimized GP models. The DGP outperforms the other models as it is the only one able to accurately approximate both, the non-stationary ridge at the transition from subsonic to supersonic and the smoother regions (Figure 4a). Consulting the predicted mean of the GP model (Figure 4b), we observe that it ignores the ridge and models the entire surface as completely smooth. This is due to the fact that during the optimization, the GP has to choose between a configuration with large or small lengthscales. The latter one would result in highly oscillating behavior of the predicted mean surface. Although this configuration would be able to model the ridge correctly, it would fail to model the smoother regions. On the contrary, choosing larger lengthscales leads to a completely smooth approximation, which is not able to model the non-stationary ridge. This behavior is visible in the figure.

Table 1: Error metrices, calculated on 156 test points, for the different GP models on the LGBB dataset.

	RRSE	RMSE	MSE
GP	0.16	0.079	0.61E−2
SGP	0.22	0.11	1.11E−2
SVGP	0.16	0.077	0.59E−2
DGP	**0.053**	**0.025**	**0.06E−2**

4.2 Low Noise Amplifier

As a second application we visit the Low Noise Amplifier (LNA) modeling problem (Gorissen et al. 2009). An LNA is an electronic amplifier used to amplify very weak signals. It is usually located very close to the detection device and is frequently used in microwave systems like GPS. An LNA is typically the first stage of a receiver, having the main function of providing the gain needed to suppress the noise of subsequent stages, such as a mixer. It has to give negligible distortion to the signal while adding as little noise as possible. The performance figures of an LNA (gain, input impedance, noise figure and power consumption) can be determined by means of computer simulations where the underlying physics are accurately taken into account. Each simulation typically requires about a minute, which is too long for a circuit designer to work efficiently. Instead, a designer could use a very accurate metamodel to quickly explore how performance figures of the LNA scale with key circuit-design parameters, such as the dimensions of transistors, passive components, signal properties and bias conditions. The goal of the design process is to figure out one or more sets of design parameters resulting in a circuit which fulfills the specifications. For this example, we now consider the modeling input noise-current (i_{in}) in function of the normalized transistor width (Wn) and inductance (Lsn). The relation to the width and inductance quantities are given by $\mathrm{W} = 100\mathrm{E}{-6} \times 10^{\mathrm{Wn}}$ m and $\mathrm{Ls} = 0.1\mathrm{E}{-9} \times 10^{\mathrm{Lsn}}$ H. The simulator is sampled by using the state-of-the-art FLOLA-Voronoi algorithm (van der Herten et al. 2015). Its strengths are that it is able to automatically identify non-linear regions in the input domain and sample these more densely. In this way, we limit the number of computationally expensive simulations but introduce a non-uniform dataset. See Figure 2d for an example of a sequentially sampled dataset consisting of 36 points and the constructed DGP model.

If we study Table 2 it shows that the traditional GP models have again difficulty capturing the local non-linearity, while behaving smooth in the surroundings. This can also be seen in Figure 2a. Interestingly, we observe that when the number of training samples N increases (from $N = 30$ to $N = 50$), the performance of the traditional GP models lowers, while the performance of the DGP ameliorates. When N is increased for a sequential dataset, more and more samples will be located in the peak region. As a result, the GP model will shrink its lengthscales to models these samples more accurately. However, this causes heavier oscillating behavior in the smooth regions, which explains the degradation in performance. DGPs do not suffer from this and show better performance with increasing N.

Table 2: Test error metrices for the different GP models on the sequentially sampled LNA dataset.

	$N = 30$			$N = 50$		
	RRSE	RSME	MSE	RRSE	RSME	MSE
GP	0.38	0.84	0.70	0.73	1.63	2.65
SGP	0.38	0.84	0.70	0.73	1.62	2.63
SVGP	0.37	0.83	0.69	0.57	1.27	1.61
DGP	**0.25**	**0.55**	**0.30**	**0.19**	**0.41**	**0.17**

5 CONCLUSION

In this paper we proposed the use of a recent regression technique, known as Deep Gaussian Processes, for efficient sequential metamodeling of non-stationary response surfaces. Metamodeling of non-stationary surfaces can be troublesome for stationary GP and Kriging models as they require equidistant training samples. This is due to the restriction of having a constant lengthscale over the complete input space which leads to either models behaving too smooth (thus missing important features) or too non-linear (capturing the features but failing to fit the smooth parts). These problems can partially be remedied by using a uniformly distributed, dense grid of training samples, but this increases the computational complexity significantly.

Our empirical experiments show that DGPs are able to model the non-stationary response surfaces more accurately and do not require an uniformly distributed training dataset. Therefore, we believe that DGPs can be a significant contribution to metamodeling in combination with sequential design as they are able to leverage non-uniform sampling distributions to their advantage and have the appealing characteristic of providing the uncertainty over their predictions.

REFERENCES

Bui, T. D., J. M. Hernández-Lobato, D. Hernández-Lobato, Y. Li, and R. E. Turner. 2016. "Deep Gaussian Processes for Regression Using Approximate Expectation Propagation". In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, 1472–1481: JMLR.org.

Crombecq, K., D. Gorissen, D. Deschrijver, and T. Dhaene. 2011. "A novel hybrid sequential design strategy for global surrogate modeling of computer experiments". *SIAM Journal on Scientific Computing* 33 (4): 1948–1974.

Csató, L., and M. Opper. 2002, March. "Sparse On-line Gaussian Processes". *Neural Comput.* 14 (3): 641–668.

Damianou, A. C., and N. D. Lawrence. 2013. "Deep Gaussian Processes". In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2013, Scottsdale, AZ, USA, April 29 - May 1, 2013*, 207–215.

Fonseca, D., D. Navaresse, and G. Moynihan. 2003. "Simulation metamodeling through artificial neural networks". *Engineering Applications of Artificial Intelligence* 16 (3): 177 – 183.

Forrester, A., A. S., and A. Keane. 2008. *Engineering Design Via Surrogate Modelling: A Practical Guide*. Chichester: Wiley.

Gorissen, D. 2010. *Grid-enabled adaptive surrugate modeling for computer aided engineering*. Ph.D. thesis, Faculty of Engineering, Ghent University, Belgium.

Gorissen, D., I. Couckuyt, P. Demeester, T. Dhaene, and K. Crombecq. 2010, August. "A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design". *Journal of Machine Learning Research* 11:2051–2055.

Gorissen, D., L. De Tommasi, K. Crombecq, and T. Dhaene. 2009. "Sequential modeling of a low noise amplifier with neural networks and active learning". *Neural Computing and Applications* 18 (5): 485–494.

Gramacy, R. B., and H. K. H. Lee. 2008. "Bayesian Treed Gaussian Process Models With an Application to Computer Modeling". *Journal of the American Statistical Association* 103 (483): 1119–1130.

Hensman, J., and N. D. Lawrence. 2014. "Nested variational compression in deep Gaussian processes". *arXiv preprint arXiv:1412.1370*. visited on 12/03/2016, 2014.

Kleijnen, J. P. 2015. *Design and Analysis of Simulation Experiments*. 2 ed. Springer International Publishing.

Lehmensiek, R., P. Meyer, and M. Mller. 2002. "Adaptive sampling applied to multivariate, multiple output rational interpolation models with application to microwave circuits". *International Journal of RF and Microwave Computer-Aided Engineering* 12 (4): 332–340.

Li, X. R., and Z. Zhao. 2006, October. "Evaluation of estimation algorithms part I: incomprehensive measures of performance". *IEEE Transactions on Aerospace and Electronic Systems* 42 (4): 1340–1358.

Matthews, A. G. d. G., M. van der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman. 2016, October. "GPflow: A Gaussian process library using TensorFlow". *arXiv preprint 1610.08733.*

Park, J.-S. 1994. "Optimal Latin-hypercube designs for computer experiments". *Journal of Statistical Planning and Inference* 39 (1): 95 – 111.

Rasmussen, and Williams. 2006. *Gaussian Processes for Machine Learning*. MIT Press.

Santner, T., W. B., and W. Notz. 2013. *The Design and Analysis of Computer Experiments*. New York, NY, USA: Springer-Verlag.

Seeger, M., C. K. I. Williams, and N. D. Lawrence. 2003. “Fast Forward Selection to Speed Up Sparse Gaussian Process Regression”. In *Workshop on AI and Statistics 9*.

Snelson, E., and Z. Ghahramani. 2006. “Sparse Gaussian Processes using Pseudo-inputs”. In *Advances in Neural Information Processing Systems 18*, edited by Y. Weiss, B. Schölkopf, and J. C. Platt, 1257–1264. MIT Press.

Suykens, J. A. K., and J. Vandewalle. 1999, June. “Least Squares Support Vector Machine Classifiers”. *Neural Process. Lett.* 9 (3): 293–300.

Titsias, M. K. 2009. “Variational learning of inducing variables in sparse Gaussian processes”. In *In Artificial Intelligence and Statistics 12*, 567–574.

van der Herten, J., I. Couckuyt, D. Deschrijver, and T. Dhaene. 2015. “A fuzzy hybrid sequential design strategy for global surrogate modeling of high-dimensional computer experiments”. *SIAM Journal on Scientific Computing* 37 (2): 1020–1039.

AUTHOR BIOGRAPHIES

VINCENT DUTORDOIR holds a M.Sc. degree in computer science engineering from Ghent University. His research interests are in machine learning, artificial intelligence and Bayesian modeling. His email address is vincent.dutordoir@ugent.be.

NICOLAS KNUDDE received his M.Sc. degree in Engineering Physics from Ghent University in 2016. Starting from September 2016 he is active as a PhD student in the research group Internet Based Communication Networks and Services (IBCN) at Ghent University, working on the detection of human activity by using radar data. His email address is nicolas.knudde@ugent.be.

JOACHIM VAN DER HERTEN received his M.Sc. degree in Computer Science in 2013, from the University of Antwerp, Belgium. Starting from August 2013 he was active as a PhD student in the research group Internet Based Communication Networks and Services (IBCN) at Ghent University. In 2017 he obtained his PhD degree. His research interests include surrogate modeling methods with sequential design including surrogate based optimization, active learning, optimal learning and machine learning. His e-mail address is joachim.vanderherten@ugent.be.

IVO COUCKUYT is a postdoctoral research fellow at Ghent University - IDLab. He received the Master degree in Computer Science in 2007 from the University of Antwerp. Since then, he worked as a PhD student in the IBCN research group of the Department of Information Technology (INTEC) in the Faculty of Engineering at Ghent University, where he obtained the PhD degree in Science in 2013. His research is mainly focused on global and local surrogate modeling (metamodeling) and its application to solve real world problems, optimization of expensive functions, evolutionary computing and machine learning methods. His e-mail address is ivo.couckuyt@ugent.be.

TOM DHAENE is full professor at the Department of Information Technology (INTEC) of Ghent University. He received the PhD degree in electrical engineering from Ghent University, Belgium, in 1993. In September 2000, he joined the Department of Mathematics and Computer Science of the University of Antwerp as a Professor. Since October 2007, he has been a Full Professor with the Department of Information Technology, Ghent University. He is also affiliated with imec. As author or co-author, he has contributed to more than 350 peer-reviewed papers and abstracts in international conference proceedings, journals and books about computational science and numerical analysis and engineering. He is the holder of five U.S. patents. His e-mail address is tom.dhaene@ugent.be.