Fast Deep Gaussian Process Modeling and Design for Large Complex Computer Experiments

by

Faezeh Yazdi

M.Sc., Brock University, 2016
M.Sc. (Math), Isfahan University of Technology, 2010

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
Department of Statistics and Actuarial Science
Faculty of Science

SIMON FRASER UNIVERSITY
Spring 2022

Declaration of Committee

Name: **Faezeh Yazdi**

Degree: **Doctor of Philosophy**

Thesis title: **Fast Deep Gaussian Process Modeling and Design for Large Complex Computer Experiments**

Committee: **Chair:** Liangliang Wang
Associate Professor, Statistics and Actuarial Science

Derek Bingham
Supervisor
Professor, Statistics and Actuarial Science

David A. Campbell
Committee Member
Professor, Mathematics and Statistics
Carleton University

David Stenning
Committee Member
Assistant Professor, Statistics and Actuarial Science

Richard Lockhart
Examiner
Professor, Statistics and Actuarial Science

David M. Higdon
External Examiner
Professor, Statistics
Virginia Tech University

Abstract

Computer models, or simulators, are widely used as a way to explore complex physical systems, but can be computationally expensive to evaluate or are not readily available to the broad scientific community. In either case, an emulator is used as a surrogate. Stationary Gaussian process emulators are often used to stand in for the computer models. In many cases, the computer model response surface does not resemble a realization of a stationary Gaussian process. Deep Gaussian processes have been shown to be capable of capturing non-stationary behaviors and abrupt regime changes in the response surface. In this thesis, we explore some of the properties of two common deep Gaussian process models for computer model emulation. We propose new methodology for one of the models so that it can serve as a computer model emulator. We introduce a new parameter that controls the amount of smoothness in the deep Gaussian process layers. We also adapt a stochastic variational approach to our deep Gaussian process model which allows for prior specification and posterior exploration of the smoothness of the response surface, thereby giving good predictive performance. Our approach can be applied to a large class of complex computer models, and scales to arbitrarily large simulation designs. The proposed methodology was motivated by the need to emulate an astrophysical model of the formation of binary black holes. Lastly, we propose a sequential design approach by combining the non-stationary deep Gaussian process model with an expected improvement based criterion. An adaptation in the deep Gaussian process prediction method facilitates the proposed sequential design approach. Our methods are illustrated in a series of synthetic examples and the real-world application.

Keywords: Computer Experiments; Surrogate Model; Deep Gaussian Processes; Uncertainty Quantification; Stochastic Variational Inference; Sequential Design; Local Kriging; Integrated Mean-squared Prediction Error

Dedication

To the memory of my loving father, who inspired me to always believe in myself.

Acknowledgements

Firstly, I would like to thank my supervisor, Prof. Derek Bingham, for his invaluable guidance, endless support and patience during my studies. I feel very privileged to have worked under his supervision. Besides my advisor, I would like to thank the rest of my dissertation committee: Profs. David M. Higdon, Richard Lockhart, David A. Campbell, David Stenning, and Liangliang Wang for their insightful comments and advice.

I wish to thank Prof. Daniel Williamson at the university of Exeter, for whose help in my research, and the time he dedicated to regular productive Zoom meetings within the last two years of my PhD. I would like to thank the Department of Statistics and Actuarial Science that enabled me to pursue my studies in a unique and friendly environment.

Finally, on a personal note, many thanks to my family, for sending their love and support from miles away, and for being proud of me, and to my amazing husband, whose support and care drives me constantly.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

Computer models, or simulators, are widely used as a way to explore complex physical systems. Frequently, a simulator is computationally expensive and only a limited number of model evaluations are available. In other settings, the computational model is relatively fast to evaluate, but is not readily available to the broad scientific community (e.g. Kaufman et al. [2011]). In either case, an emulator of the computer model is required.

Stationary Gaussian processes (GP) have become the conventional approach for deterministic computer model emulation (Sacks et al. [1989b], Jones et al. [1998]). In many cases, the simulator response surface does not resemble a realization of a stationary GP and innovations to adapt to the non-stationary behaviour have been developed (e.g., Gramacy and Apley [2015]). In recent years, deep Gaussian process (DGP) models have been proposed for non-parametric regression (Damianou and Lawrence [2013], Dunlop et al. [2018]), and more recently the approach of Damianou and Lawrence [2013] has been adapted to address computer model emulation (Monterrubio-Gomez et al. [2020], Radaideh and Kozlowski [2020], Rajaram et al. [2020], Sauer et al. [2020], Ming et al. [2021]).

In this thesis, our first aim is to lay out some of the properties of DGPs for computer model emulation. In particular, we consider two common DGP models (Damianou and Lawrence [2013], Dunlop et al. [2018]) and show how they can be written as Bayesian hierarchical models. We adapt the approach of Dunlop et al. [2018] so that it can serve as an emulator of complex computer models. The proposed approach allows for prior specification and posterior exploration of the smoothness of the response surface. The proposed methodology is motivated by the application of emulation of a complicated astrophysical model, namely the Compact Object Mergers: Population Astrophysics and Statistics (COMPAS) that simulates the formation of binary black holes (BBHs) (Stevenson et al. [2017], Barrett et al. [2018], Vigna-Gomez et al. [2018]).

A practical problem of interest for simulation experiments is that of experimental design. For DGPs improvement of the estimate of the response surface requires design points where the surface changes rapidly. Sequential experimental design can help this goal by spending relatively more effort sampling in regions where the response surface is more complex. Choosing new runs sequentially has been formalized using expected improvement (EI) criteria for various goals such as optimization (Jones et al. [1998]) and contour estimation (Ranjan et al. [2008]). In this thesis, we focus on sequential design approaches where additional design points are chosen by optimizing a criterion based on the predictive variance (e.g. Sacks et al. [1989a]). To improve performance of our DGP emulator and exploration of the input space for the selection of future runs, a sequential design strategy is introduced. We combine the non-stationary DGP model with an EI-based sequential design criterion to deviate from usual space-filling designs. An adaptation in the DGP prediction method facilitates our proposed sequential design approach. The proposed approach is illustrated on a 2-d toy model as well as the COMPAS model.

1.2 Thesis Outline

The rest of the thesis is organized as follows: in Chapter 2, we provide background on computer model emulation, design and variational inference (VI) as key ingredients for the new developments in the next chapters. The chapter specifically is started by an overview of stationary GP emulation and describes existing work in non-stationary GP modelling. We finish off this chapter with a review of VI approaches that are relevant for our proposed methodology.

The work in Chapter 3 is motivated by emulation of the COMPAS model with the type of discontinuities that are not easily captured by conventional methods. This chapter is started by discussing the challenges encountered emulating the COMPAS model. We generalize the notation of two broad forms of the DGP in order to highlight their differences and properties. We propose our DGP emulator by modifying one of the forms through introducing a new parameter (or parameters) that allows us to control the smoothness of the DGP layers. We theoretically illustrate and numerically visualise the impact of this proposed parameter on the degree of smoothness of in the layers of the DGP. We develop a VI approach to fit our DGP emulator. We demonstrate the importance of estimating this new parameter on the performance of our DGP emulator by two 1-d toy models. We finish off this chapter by demonstrating the proposed approach on a 2-d toy model, as well as emulation of the COMPAS model.

In Chapter 4, we propose a sequential design approach that aims to reduce predictive variance of our DGP emulator along with improving exploration of the input space for guiding future simulations. To proceed with this method, we adapt our prediction method. To speed up our sequential design algorithm, we propose to use nearest neighbor (NN) predictions using our DGP. We utilize our localized prediction method with the sequential design strategy. We investigate the impact of refitting model in batches of added design points on the prediction performance of the DGP and the ability of our localized design criterion in exploring the input space. A comparison of resulting sequential designs with and without refitting model in the 2-d toy model is presented. The chapter is finished off by demonstrating the proposed approach on the COMPAS data. Chapter 5 summarizes the key contributions of the thesis and discusses potential areas for future work.

Chapter 2

Background

A computer model, or a simulator, is a mathematical model of a process that has been translated into computer code. Computer models have been used as a way to explore real systems in many areas of scientific research such as oil reservoir modelling (Tavassoli et al. [2004]), Galaxy formation (Vernon et al. [2010]), climate and environmental sciences (Challenor [2004], Lynch [2008], Edwards et al. [2011]), industrial design and engineering (Ankenman et al. [2010]), and medical applications such as HIV transmission modelling (Andrianakis et al. [2015]). Usually physical experimentation is too costly, sometimes impossible, and hence computer models may reduce the cost of exploring a system.

The inputs to a computer model can generally be placed into one of two groups: (i) control variables that are observable or adjustable in the physical system; and (ii) calibration parameters that are needed to run the computer model, but whose values in the physical system are unknown and must be estimated from observations. In settings where the calibration parameters have no physical meaning, they are sometimes instead called tuning parameters (Higdon et al. [2004]). Ideally, for appropriate choice of the calibration parameters, a computer model simulates the mean of the physical system. Outputs of computer models can be scalars, functional, time series, or spatial fields, for example. The output of a simulator can be deterministic or stochastic, though this thesis considers only deterministic computer models.

In the literature, there exist statistical techniques relevant to the analysis of computer models such as emulation (Sacks et al. [1989b], Santner et al. [2003]), calibration (Kennedy and O'Hagan [2001], Higdon et al. [2008], Chang and Guillas [2018]), experimental design (Johnson et al. [1990], Tang and Wu [1997], Cheng et al. [1998], Bingham et al. [2014]), and uncertainty and sensitivity analysis (Oakley and O'Hagan [2002, 2004], Saltelli et al. [2008]). In this chapter, we briefly review some of the methods applied in this thesis.

2.1 Computer Model Emulation

Computer models are often computationally demanding to evaluate because each simulation is the solution of complex mathematical equations. In other cases, they are fast to evaluate but are not readily available to all scientists. As a result, a statistical surrogate or emulator is used in place of the computer model to make predictions of the model output at unsampled input values with estimates of uncertainty. An emulator can help provide insight into the functional form of the computer model response and the importance of inputs (Oakley and O'Hagan [2004]). The traditional approach to computer model emulation is to use a GP. This section presents a brief description of GP emulators. Additionally, non-stationary GP models are briefly discussed. This serves as background for Chapter 3, where we introduce our version of the DGP for emulating computer models.

2.1.1 Stationary Gaussian Process (GP) Emulators

Gaussian processes have become standard tools for analysis of computer models. This includes uncertainty propagation (Oakley [2004], Lockwood and Anitescu [2012]), model calibration (Kennedy and O'Hagan [2001], Higdon et al. [2008]), design of experiments (Sacks et al. [1989b], Pronzato and Muller [2012]), optimisation (Jones et al. [1998], Brochu et al. [2010]) and sensitivity analysis (Oakley and O'Hagan [2004], Iooss and Lemaitre [2015]). This section focuses on emulation of computer models using stationary GPs.

A stationary GP is the most common choice for an emulator of computer models (Sacks et al. [1989b], O'Hagan et al. [1999], Santner et al. [2003]). The GP is a random process whose evaluation at any finite collection of locations follows a multivariate Gaussian distribution (Stein [1999]). The computer model response surface is viewed as a realization of a GP (Sacks et al. [1989b]). For example, let $y^S = \eta(\mathbf{x})$ denote the scalar output of a deterministic computer model, $\eta(.)$, at input $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$. The inputs are typically scaled so that $\mathcal{X}$ is the d-dimensional unit cube. There are n_S inputs given by the rows of the $n_S \times d$ design matrix $\mathbf{X}$ and corresponding outputs $\mathbf{y}^S = (y_1^S, \ldots, y_{n_S}^S)^T$ i.e., $y_i^S = \eta(\mathbf{x}_i)$ for $i = 1, 2, \ldots, n_S$.

In computer model settings, the GP is often specified with a constant mean and a stationary covariance function. The computer model output is modeled as

$$y^S(\mathbf{x}) = \mu + z(\mathbf{x}),$$

where μ is a constant, $z(\mathbf{x})$ is a stationary, mean zero GP and

$$\mathrm{Cov}(y^S(\mathbf{x}), y^S(\mathbf{x}')) = k(\mathbf{x}, \mathbf{x}'; \sigma^2, \boldsymbol{\phi}) = k(\|\mathbf{x} - \mathbf{x}'\|_2; \sigma^2, \boldsymbol{\phi}), \tag{2.1}$$

where $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, σ^2 is the marginal variance of the process and $\boldsymbol{\phi}$ is a vector of parameters governing the correlation. Using a GP with a constant mean can be viewed as a way of forcing the covariance structure of the GP to model all the signal in the output.

Typically, the computer model output is centred and scaled so that a mean-zero GP can be used.

Conditioning on the hyperparameters $\{\sigma^2, \boldsymbol{\phi}\}$, the design, $\mathbf{X}$, and outpus, $\mathbf{y}^S$, the predictive distribution at new input $\mathbf{x}^*$ is conditionally Gaussian with mean and variance, respectively,

$$\begin{aligned} &k(\mathbf{X}, \mathbf{x}^*)^T \mathbf{K}^{-1} \mathbf{y}^S, \\ &k(\mathbf{X}, \mathbf{x}^*)^T \mathbf{K}^{-1} k(\mathbf{X}, \mathbf{x}^*), \end{aligned} \tag{2.2}$$

where $\mathbf{K} = k(\mathbf{X}, \mathbf{X})$ is the covariance matrix for the simulations, with $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j; \sigma^2, \boldsymbol{\phi})$, and $k(\mathbf{X}, \mathbf{x}^*)$ is the $n_S \times 1$ vector of correlations between a response at $\mathbf{x}^*$ and those at the inputs in the design, $\mathbf{X}$. In practice, the hyperparameters $\{\sigma^2, \boldsymbol{\phi}\}$ are unknown and must be estimated (Currin et al. [1991], Higdon et al. [2008], Irvine et al. [2007], Kaufman and Sain [2010]).

The main reasons for the use of a GP emulator lie in its success as a non-parametric regression method, its ability to interpolate the known outputs, and also to provide a foundation for uncertainty quantification in a deterministic setting (Sacks et al. [1989b], Jones et al. [1998]). With that said, there are many situations where the simulator outputs are not well represented by a stationary model. This can occur, for example, when there are rapid changes in the behaviour of the response surface or discontinuities (e.g., the astronomy application in Chapter 3). Figure 2.1 shows a simple, illustrative model given by the indicator function $\eta(x) = \mathbb{1}_{(0.3,0.7)}$ for $x \in (0, 1)$. This model has appeared in Dunlop et al. [2018] as an example of a function where a stationary GP has difficulty modeling the behaviour in the response. This sort of computer model is challenging for the stationary GP, because it will favour different correlation lengths in different parts of the input space, i.e., near the discontinuities small correlation lengths will be more desirable, while the flat regions will favour relatively long correlation lengths. A more appropriate emulator of this model will have to locally adapt the correlation lengths to the response surface. There are a number of approaches that adapt GPs to model non-stationary responses. We discuss these methods in the next section.

2.1.2 Non-stationary GP Models

Most approaches to non-stationary GP modeling can be broadly classified as space-warping or covariance-modeling (sometimes called space-partitioning). These two classes inform the types of DGPs that we are going to introduce in Chapter 3. In the former, the input space is warped so that the observations can be modeled as a stationary GP. Examples of space-warping include work by Sampson and Guttorp [1992], Schmidt and O'Hagan [2003], Bornn et al. [2012] and Marmin et al. [2018]. For example, in Sampson and Guttorp [1992] multidimensional scaling and thin-plate splines are employed to reach the smooth mapping

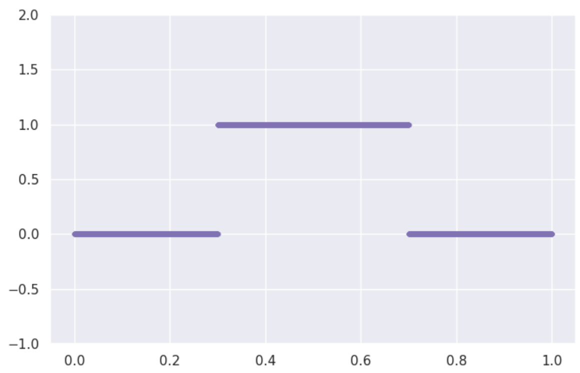


Figure 2.1: A simple, illustrative computer model with regions of discontinuities

from original input space to a warped space where a stationary GP can be used to model the outputs. Schmidt and O'Hagan [2003] propose using a latent GP prior of the inputs to provide a mapping from the original input space to a transformed space where the outputs can be modeled as a stationary GP. In particular, a warping function $\mathbf{d}(.)$ is defined from geographical space G to the warped space (latent space) D. The spatial covariance function at two input points $\mathbf{x}, \mathbf{x}' \in G$ is defined as

$$\operatorname{Cov}(Y(\mathbf{x},t), Y(\mathbf{x}',t) = \sqrt{\operatorname{var}(Y(\mathbf{x},t))\operatorname{var}(Y(\mathbf{x}',t))}\ g(\|\mathbf{d}(\mathbf{x}) - \mathbf{d}(\mathbf{x}')\|), \tag{2.3}$$

where $Y(\mathbf{x}, t)$ is a spatiotemporal process defined for $\mathbf{x} \in G$ and arbitrary time t, and g(.) is a monotone function. Schmidt and O'Hagan [2003] adopted a fully Bayesian approach specifying a GP prior distribution for the mapping function $\mathbf{d}(.)$. In their work, they claim that the GP prior on the deformation process $\mathbf{d}(.)$ tends to eliminate non-injective mappings that can be observed in the approach of Sampson and Guttorp [1992]. In Bornn et al. [2012], the original field is embedded in a space of higher dimension where it can be more straightforwardly described and modelled. More specifically, the dimensionality of the problem is shifted from 2 or 3 dimensions to 4, 5, or more in order to recover stationarity in the process. Marmin et al. [2018] combine dimensional reduction with a multiple index model and a non-linear mapping in the input space to achieve a non-stationary GP.

Alternatively, there are a number of approaches that vary the correlation function over the input space to deal with non-stationarity. In spatial statistics, the weighted sum of locally defined kernels is widely used to model non-stationarity in the spatial process (e.g. Fuentes [2001], Fuentes and Smith [2003], Banerjee et al. [2004]). In Higdon [1998] and Higdon et al. [1999], a non-stationary covariance function is obtained by convolving the spatially varying kernel functions $K_{\mathbf{x}}(\mathbf{u})$ centred on $\mathbf{x}$

$$k(\mathbf{x}, \mathbf{x}') = \int_{\mathcal{X}} K_{\mathbf{x}}(\mathbf{u}) K_{\mathbf{x}'}(\mathbf{u})\ d\mathbf{u},$$

where $\mathbf{x}$, $\mathbf{x}'$ and $\mathbf{u}$ are locations in $\mathcal{X}$. Higdon et al. [1999] used the Gaussian kernel density estimator and derived a non-stationary version of the stationary squared exponential covariance function,

$$k(\mathbf{x},\mathbf{x}') = \sigma^2 \frac{|\Sigma(\mathbf{x})|^{1/4}|\Sigma(\mathbf{x}')|^{1/4}}{|(\Sigma(\mathbf{x})+\Sigma(\mathbf{x}'))/2|^{1/2}} \exp(-Q(\mathbf{x},\mathbf{x}')), \tag{2.4}$$

with

$$Q(\mathbf{x},\mathbf{x}') = (\mathbf{x}-\mathbf{x}')^T \left(\frac{\Sigma(\mathbf{x})+\Sigma(\mathbf{x}')}{2}\right)^{-1} (\mathbf{x}-\mathbf{x}'), \tag{2.5}$$

where $\Sigma(\mathbf{x})$ is a covariance matrix of a Gaussian kernel centred at $\mathbf{x}$, and $|.|$ denotes the determinant of a matrix. If kernel matrices $\Sigma(.)$ are constant, the special case of the squared exponential correlation based on Mahalanobis distance is recovered. If they are not constant with respect to $\mathbf{x}$, the evolution of the kernel covariance matrices in space produces non-stationary covariance. Paciorek and Schervish [2004] generalized the covariance function (2.4) for any stationary correlation function R as following

$$k(\mathbf{x},\mathbf{x}') = \sigma^2 \frac{|\Sigma(\mathbf{x})|^{1/4}|\Sigma(\mathbf{x}')|^{1/4}}{|(\Sigma(\mathbf{x})+\Sigma(\mathbf{x}'))/2|^{1/2}} \; R(\sqrt{Q(\mathbf{x},\mathbf{x}')}), \tag{2.6}$$

to produce a class of non-stationary covariance functions that provide more flexibility than the special case (2.4). More directly, the approach of Paciorek and Schervish [2004] allows the correlation between observations to vary smoothly as an unknown function of their spatial location, and will be used in constructing the DGP in Chapter 3.

Other approaches in the computer experiments literature include local models such as treed GPs (Gramacy and Lee [2008]) and local GPs (Gramacy and Apley [2015]), to name just a few. These methods aim to provide non-stationary modeling features and reduce the computational burden for large computer experiments at the same time.

In recent years, deep learning approaches (e.g. Damianou and Lawrence [2013], Dunlop et al. [2018]) have been developed to accommodate the non-stationary structures in complex response surfaces for non-parametric regression problems. Recently, the first approach (Damianou and Lawrence [2013]) has been adapted in computer model emulation (Monterrubio-Gomez et al. [2020], Radaideh and Kozlowski [2020], Rajaram et al. [2020], Sauer et al. [2020], Ming et al. [2021]). In Chapter 3, we lay out forms of both models defined in Damianou and Lawrence [2013] and Dunlop et al. [2018] as generalizations of space-warping and covariance-modeling methods of Schmidt and O'Hagan [2003] and Paciorek and Schervish [2004], respectively, and a computer model emulation is proposed through modifying version defined in Dunlop et al. [2018].

2.2 Design of Computer Experiments

If we wish to make predictions using an emulator, the selection of computer model trials is important. The process of running a computer model at a variety of different input values is described as a computer experiment. This section presents a literature review of work which has been done in the design of computer experiments, and this serves background for Chapter 4.

Experimental designs for computer models typically begin with space filling type designs (McKay et al. [1979], Johnson et al. [1990]). These include Latin hypercube designs (McKay et al. [1979]) and variations thereof (e.g. Morris and Mitchell [1995], Cheng et al. [1998], Tang [1998]). An important feature of computer experiments is that they can frequently be performed sequentially (e.g. Currin et al. [1988], Sacks et al. [1989b], Welch et al. [1992], Santner et al. [2003]). A sequential design strategy involves adding design points to an existing design in stages according to a specified criterion. Information obtained about the response surface from previous stages is used to select the inputs at the next stage. Each stage can consist of adding one or more observations. Depending on the goal of the experiment, different criteria can be derived using the concept of expected improvement (EI) to update designs sequentially. Analysts might be interested in obtaining additional design points that could help reduce prediction uncertainty. In this case, additional design points are chosen sequentially by optimizing a criterion based on prediction errors (Sacks et al. [1989a]) or entropy (Mitchell and Scott [1987], Shewry and Wynn [1987], Currin et al. [1991]).

EI algorithms are also developed to solve problems of constrained optimization (Schonlau et al. [1998], Gramacy et al. [2016]). Particularly, in Gramacy et al. [2016], they propose an algorithm for constrained optimization of complex computer models motivated by the augmented Lagrangian numerical optimization framework of Notz [2015]. Improvement strategies have been implemented for problems of contour estimation (Ranjan et al. [2008], Bingham et al. [2014]) and percentile estimation (Roy and Notz [2014]). Specifically, in Ranjan et al. [2008], a sequential design approach is presented for estimating a contour of a complex computer model, where a contour identifies a boundary that distinguishes good and bad performance. Their strategy is to sequentially choose design points that are on or near the estimate of the contour.

In recent years, the DGP defined in Damianou and Lawrence [2013] has been used in sequential design of computer experiments. In Dutordoir et al. [2017], this DGP is used to fit to sequentially collected data, but acquisition criteria are not based on the DGP fits. In Rajaram et al. [2020], a strategy based on maximum variance criterion of MacKay [1992] is applied with this DGP. Hebbal et al. [2021] applied the DGP of Damianou and Lawrence [2013] to Bayesian optimization via the EI of Jones et al. [1998]. In Sauer et al. [2020], they construct a sequential design using the integrated mean-squared prediction error (IMSPE) and an active learning strategy in Cohn [1994] with this DGP. In Chapter 4, we propose a sequential design approach using IMSPE with the modified DGP of Dunlop et al. [2018].

2.3 Variational Inference (VI)

Variational Inference (VI) is a method for approximating probability densities. There are several good reviews to VI available in the machine learning and statistics literature, for instance Jordan et al. [1999], Wainwright and Jordan [2008] and Blei et al. [2017]. Estimating parameters is typically done in a Bayesian context, via Markov chain Monte Carlo (MCMC) or related methods (Metropolis et al. [1953], Hastings [1970], Geman and Geman [1984], Gelfand and Smith [1990]). However, when datasets are large, doing inference for GPs requires inversion of large covariance matrices and this is difficult to impossible. VI provides an approach to parameter estimation for large datasets through approximating samples from the posterior distribution of the parameter. This section presents basics of VI and emphasizes other VI ideas in the literature that are most relevant for Chapter 3.

The goal of VI is to approximate a conditional density of latent variables given observed variables by specifying a family Q of densities over the latent variables. The main idea is to find the best candidate in Q, the one that minimizes the Kullback-Leibler (KL) divergence to the exact conditional distribution. The KL divergence is a measure of proximity between two densities and equals to zero when the two densities are the same (Kullback and Leibler [1951]). The family Q is chosen to be flexible enough to capture a density close to the target conditional density, but simple enough for efficient optimization.

Let $\mathbf{y}$ and $\mathbf{u}$ be a set of observed and latent variables, respectively. Inference in a Bayesian model amounts to conditioning on data and computing the posterior distribution $\mathbb{P}(\mathbf{u}|\mathbf{y})$. VI seeks an approximate posterior distribution $q(\mathbf{u})$ that is made as close as possible to the true posterior distribution $\mathbb{P}(\mathbf{u}|\mathbf{y})$, where the closeness is measured by the Kullback-Leibler divergence,

$$\begin{aligned}
\mathrm{KL}\Big(q(\mathbf{u})\|\mathbb{P}(\mathbf{u}|\mathbf{y})\Big) &= \mathbb{E}_q\Big[\log q(\mathbf{u})\Big] - \mathbb{E}_q\Big[\log\mathbb{P}(\mathbf{u}|\mathbf{y})\Big] \\
&= \mathbb{E}_q\Big[\log q(\mathbf{u})\Big] - \mathbb{E}_q\Big[\log\mathbb{P}(\mathbf{u},\mathbf{y})\Big] + \log\mathbb{P}(\mathbf{y}) \\
&= -\mathbb{E}_q\Big[\frac{\log\mathbb{P}(\mathbf{u},\mathbf{y})}{\log q(\mathbf{u})}\Big] + \log\mathbb{P}(\mathbf{y}) \\
&= -\mathcal{L}_q + \log\mathbb{P}(\mathbf{y}).
\end{aligned} \tag{2.7}$$

$\mathbb{E}_q$ indicates that the expectation is taken with respect to $q(\mathbf{u})$. For many models, the marginal density of the observations $\mathbb{P}(\mathbf{y})$, also called the evidence, is unavailable in closed form or requires exponential time to compute. Therefore, the KL is not computable. On the other hand, since the KL divergence is non-negative, it follows that $\mathcal{L}_q$ is a lower bound on $\log\mathbb{P}(\mathbf{y})$. Hence we can maximize the alternative objective $\mathcal{L}_q$, called the evidence lower bound (ELBO), that is equivalent to minimizing the KL. The lower bound $\mathcal{L}_q$ can also be written as

$$\begin{aligned}\mathcal{L}_q &= \mathbb{E}_q\Big[\frac{\log\mathbb{P}(\mathbf{u},\mathbf{y})}{\log q(\mathbf{u})}\Big] \\ &= \mathbb{E}_q\Big[\log\mathbb{P}(\mathbf{y}|\mathbf{u})\Big] + \mathbb{E}_q\Big[\log\mathbb{P}(\mathbf{u})\Big] - \mathbb{E}_q\Big[\log q(\mathbf{u})\Big] \\ &= \mathbb{E}_q\Big[\log\mathbb{P}(\mathbf{y}|\mathbf{u})\Big] - \mathrm{KL}\Big(q(\mathbf{u})\|\mathbb{P}(\mathbf{u})\Big),\end{aligned} \tag{2.8}$$

which is a sum of the expected log-likelihood of the data and the KL divergence between the prior $\mathbb{P}(\mathbf{u})$ and $q(\mathbf{u})$. The first term encourages densities $q(.) \in Q$ that place their mass on configurations of the latent variables that explain the observed data. The second term pushes densities $q(.) \in Q$ close to the prior. Thus, the usual balance between likelihood and prior is reflected.

General VI algorithms have been developed for a variety of classes of models, including shrinkage models (e.g. Armagan and Dunson [2011]), general time-series models (e.g. Roberts et al. [2004], Barber and Chiappa [2006]), robust models (e.g. Tipping and Lawrence [2005]), and GP models (e.g. Titsias [2009], Titsias and Lawrence [2010], Hensman et al. [2013]).

Inference in GP models is intractable if the size of data is large. The variational approach of Titsias [2009] using inducing points has been highly influential in the area of scalable GP approximations. The inducing variables are latent function values evaluated at some inputs which can be a subset of the training inputs or pseudo-points (Snelson and Ghahramani [2006]). Inducing variables were first introduced in the context of sparse GPs by Snelson and Ghahramani [2006] to overcome the inversion of the covariance matrix of the whole dataset. This idea is used by Titsias [2009] in a variational approach, where the inducing inputs are defined to be variational parameters which are selected by minimizing the KL divergence between the variational distribution and the exact posterior distribution.

An important advance in the use of variational methods is their combination with stochastic gradient descent (Hoffman et al. [2013]). The variational inducing point framework has been combined with such methods in Hensman et al. [2013, 2015]. The approach has also been successfully used to perform scalable inference in more complex models such as the GP latent variable models (Titsias and Lawrence [2010], Damianou et al. [2016]) and the DGP models (Damianou and Lawrence [2013], Hensman and Lawrence [2014], Dai et al. [2016], Salimbeni and Deisenroth [2017]). Specifically, Salimbeni and Deisenroth [2017] introduce the doubly stochastic variational inference (DVSI) method for inference with a DGP model of Damianou and Lawrence [2013], which allows scalability to large datasets with an effective performance. The state-of-the-art DSVI has been recently implemented in GPflux (Dutordoir et al. [2021]), an actively maintained open-source library dedicated to the DGP. In Chapter 3, we derive an inference procedure based on DSVI method of Salimbeni and Deisenroth [2017] for fitting the modified DGP of Dunlop et al. [2018].

Chapter 3

Deep Gaussian Processes Emulation

3.1 Introduction

Complex phenomena are often explored by means of computer models that simulate their behaviour. In some cases, the computer models are computationally demanding. In other cases, they are fast to evaluate but run only on supercomputers or must be run by specialists, thereby limiting their availability to all scientists. In either case, a statistical emulator may be used in place of the computer model.

The common approach of emulating computer model responses is using a GP. In many cases, the computer model response surface does not resemble a realization of a stationary GP (e.g. Figure 2.1). Computer models whose response surface is not well represented by a stationary model display different behaviour in terms of complexity in different regions of the input space. This arises, for example, when the variability in the shape of the response surface changes in the inputs, or the model response exhibits sharp localized features, e.g. discontinuities or high peaks. Stationary GP emulators fail to capture unusual behaviours in this class of complex computer model (Dunlop et al. [2018], Volodina and Williamson [2020]). In this chapter, we introduce a non-stationary DGP emulator which could be applied to a large class of complex computer models, and scales to arbitrarily large simulation designs.

The application that motivated the proposed methodology was emulation of an astrophysical model (i.e., the COMPAS model). COMPAS is a binary population synthesis model that simulates the formation of BBHs through the evolution of pairs of massive stars (Stevenson et al. [2017], Vigna-Gomez et al. [2018], Barrett et al. [2018], Neijssel et al. [2019]). A BBH is a system consisting of two black holes in close orbit around each other (Figure 3.1). A gravitational-wave signal is emitted during the merger of two black holes, and can be measured by ground-based gravitational-wave detectors (Mandel and Farmer [2018]). Also as two black holes merge, the *chirp mass* is expelled. In the COMPAS model, inputs describe the initial conditions of a binary star system (e.g., the mass of the most massive star) and parameters that govern physical processes in the system. The output upon which we focus is the chirp mass of the formed BBH. Given the high dimensionality of the input and complexity of binary stellar evolution, in practice many billions of binaries need to be simulated to perform an experiment that is sufficiently large to make scientific inferences. This can amount to computing times of years, and, as a consequence, a fast statistical emulator for the COMPAS model is required.

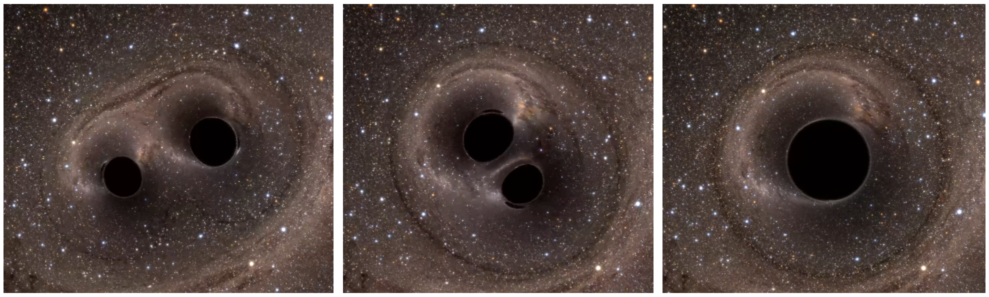


Figure 3.1: Computer simulation of the black hole binary system GW150914. Credits: SXS (Simulating eXtreme Spacetimes) project

The first challenge for computer model emulation in this setting is the presence of regions of discontinuities in the response surface of the chirp mass. That is, BBHs only form in some regions of the input space. If a BBH system does not form, then no chirp mass is observed. Unfortunately, the portions of input space that result in non-zero outputs are unknown, though it is assumed that a chirp mass is observed in a union of compact regions in the input space. As a result, there are series of disconnected chirp mass response surfaces. The second challenge in population synthesis emulation is that in a simulation suite of two million of COMPAS trials, roughly 24% results in a BBH merger (e.g. Belczynski et al. [2002], Kruckow et al. [2018], Taylor and Gerosa [2018], Broekgaarden et al. [2019]), so the vast majority of computational time is spent on simulations that do not produce an outcome. If millions of BBHs are wanted, a large number of simulations needs to be performed. So, an emulator of the COMPAS model will have to be adapted to the type of discontinuities that we have in this setting with a required large number of simulation runs.

In this chapter, we propose a new methodology for building a non-stationary emulator which can be used in a wide range of applications - including the COMPAS model. We build a DGP emulator that aims to capture non-standard features of the COMPAS model, and scales to the large number of simulation runs. We introduce a new parameter (or parameters) that allows us to control the smoothness of the DGP layers. We also adapt a stochastic variational inference approach to our DGP model which allows us to specify prior distribution and explore posterior distribution for the smoothness parameter(s) of the response surface, thereby giving a good predictive performance.

This chapter has the following structure. In Section 3.2, the notation of two broad forms of the DGP are generalized to emphasize their differences and properties. In Section

3.3, our DGP emulator is proposed by modifying one of the forms through introducing a new parameter (or parameters) that controls smoothness of the DGP layers, followed by illustration of this property both theoretically and numerically. Section 3.4 details the adapted variational inference approach to our DGP emulator. In Section 3.5, our proposed method is illustrated in a series of synthetic examples as well as emulation of the COMPAS model. Section 3.6 concludes with discussion and future work.

3.2 Deep Gaussian Processes (DGPs)

As discussed in Section 2.1, most non-stationary GP modeling approaches can be broadly classified as space-warping and covariance-modeling (or space-partitioning). Deep Gaussian processes, as a way to accommodate the complex structures in some response surfaces, belong to one of these two classes too. In the literature, there are two broad forms of a DGP introduced in Damianou and Lawrence [2013] and Dunlop et al. [2018], which are generalizations of space-warping and covariance-modeling methods of Schmidt and O'Hagan [2003] and Paciorek and Schervish [2004], respectively. In this section, we lay out the notation of these two forms in such a way that allows us to highlight their differences and properties.

3.2.1 DGP Formulations

The first common form introduced in Damianou and Lawrence [2013] is a GP directly composed with another GP recursively, leading to what is referred to as a DGP. A DGP with N hidden layers in this form is defined by composition of functions $u_1 : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}^{d'_1}$ and $u_n : \mathbb{R}^{d'_{n-1}} \to \mathbb{R}^{d'_n}$ that are conditionally Gaussian,

$$u_{1,l}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k_{1,l}(\mathbf{x}; \boldsymbol{\phi}_1)) \quad , \qquad l = 1, \dots, d'_1 \tag{3.1}$$

$$u_{n,l}(\mathbf{x})|u_{n-1}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k_{n,l}(u_{n-1}(\mathbf{x}); \boldsymbol{\phi}_n)) \quad , \qquad l = 1, \dots, d'_n, \tag{3.2}$$

for $n = 2, \dots, N+1$, $d'_{N+1} = 1$ and $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$, where $u_{n,l}(\mathbf{x})$ represents the l^{th} component of $u_n(\mathbf{x}) \in \mathbb{R}^{d'_n}$. Here $k_{1,l} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $k_{n,l} : \mathbb{R}^{d'_{n-1}} \times \mathbb{R}^{d'_{n-1}} \to \mathbb{R}$ are stationary covariance functions. Vector parameters $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_n$ are parameters such as the variance and correlation lengths. Typically the covariance functions are chosen to be the same (e.g. squared exponential). Since the covariance function at layer n is a function of the outputs from the previous layer, this approach amounts to warping the input space. Under this specification $\mathbf{u}_{N+1}$ is the observation vector $\mathbf{y}^S$, and the previous layers are unobserved latent variables that warp the input space. Specifically, $\mathbf{y}^S = \mathbf{u}_{N+1}$ where $\mathbf{u}_{N+1} = u_{N+1}(u_N(...(u_1(\mathbf{X})))) \in \mathbb{R}^{n_S}$, and u_{N+1} refers to a DGP with N hidden layers.

An alternative form of the DGP was introduced in Dunlop et al. [2018]. In their specification, the covariance parameters of each hidden layer are a function of the output of a previous hidden layer. Specifically, a DGP with N hidden layers in this form is defined by sequences of functions $u_n : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}$ that are conditionally Gaussian,

$$u_1(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k_1(\mathbf{x}; \boldsymbol{\phi}_1)), \tag{3.3}$$

$$u_n(\mathbf{x}) | u_{n-1}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k_n(\mathbf{x}; \boldsymbol{\phi}_n(u_{n-1}(\mathbf{x})))), \tag{3.4}$$

where $n = 2, \ldots, N+1$ and $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$. Similar to the previous approach, a stationary covariance function $k_1 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, with vector parameter $\boldsymbol{\phi}_1$, is used in the base layer. In this form, $k_n : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a non-stationary covariance function which operates on the original input space $\mathcal{X}$ and $\boldsymbol{\phi}_n(u_{n-1}(\mathbf{x}))$ denotes a vector of all covariance parameters that are a function of the previous layer u_{n-1}. Explicitly, the covariance function for any layer other than the base layer $(n > 1)$ is $k_n(\mathbf{x}, \mathbf{x}'; \boldsymbol{\phi}_n(u_{n-1}(\mathbf{x}), u_{n-1}(\mathbf{x}')))$ where the covariance function $k_n(.,.)$ is always expressed between the original input locations $\mathbf{x}$ and $\mathbf{x}'$, but the covariance parameters depend on the processes $u_{n-1}(\mathbf{x})$ and $u_{n-1}(\mathbf{x}')$ at those locations. Hence, handling non-stationarity in this form of the DGP can be considered as a covariance-modelling method. Under this specification $\mathbf{u}_{N+1}$ is the observation vector $\mathbf{y}^S$, i.e. $\mathbf{y}^S = \mathbf{u}_{N+1}$ where $\mathbf{u}_{N+1} = u_{N+1}(\mathbf{X}) \in \mathbb{R}^{ns}$. Here u_{N+1} refers to a DGP with N hidden layers and the unobserved random vectors $(\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N)$, where $\mathbf{u}_n = u_n(\mathbf{X}) \in \mathbb{R}^{ns}$ for $n = 1, \ldots, N$ are considered as hidden layers which are used to discover the covariance among the given simulation outputs.

Remark 1. *As seen in equations (3.2) and (3.4), the two DGP formulations look similar, however there are important differences. In equation (3.4) the input of each layer is always* $\mathbf{x}$ *in a d-dimensional space and is mapped to a* 1*-dimensional output. The outputs of each hidden layer are considered as parameters and are used to model covariance parameters in the next layer. In equation (3.2) the input of layer* u_n $(n > 1)$ *is the output of the previous layer* u_{n-1} *in* d'_{n-1}*-dimensional space and is mapped to a* d'_n*-dimensional output where* $d'_n \geq d$ *and* d'_n *is not necessarily one (except* $d'_{N+1} = 1$*). In this form, the outputs in each layer are considered as an warped domain space for the covariance function of the next layer.*

The DGP formulations can deal more complex response surfaces than standard GPs. In the first form by warping the input space through hidden Gaussian layers effectively move some input points closer and others farther apart to achieve non-stationarity. In the second form, adapting the covariance parameters (e.g length scales) across the input space through hidden Gaussian layers achieves non-stationarity.

One might ask what the DGP would look like if the data were actually a realization of a stationary GP. The following propositions aim to address exactly these circumstances.

Proposition 1. *Under the DGP model in (3.2),* $u_{n,l}(\mathbf{x}) | u_{n-1}(\mathbf{x})$ *is a stationary GP for* $l = 1, \ldots, d'_n$ *if and only if* $u_{n-1}(\mathbf{x}) = \mathbf{c} \odot \mathbf{x}$ *for any* $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$*, where* $\mathbf{c} \in \mathbb{R}^d$ *is a vector of constants and* $\odot$ *represents an element-wise product.*

Proof. $\Rightarrow$) Let $u_{n,l}(\mathbf{x})|u_{n-1}(\mathbf{x})$ be a stationary GP. This implies that $k_{n,l}(u_{n-1}(\mathbf{x}); \boldsymbol{\phi}_n)$ is a stationary covariace function, where the input space is not warped through $u_{n-1}(.)$. Hence $u_{n-1}(\mathbf{x})$ must be a linear transformation of $\mathbf{x}$. i.e there exists a vector of constants $\mathbf{c} \in \mathbb{R}^d$ such that $u_{n-1}(\mathbf{x}) = \mathbf{c} \odot \mathbf{x}$ for any $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$.

$\Leftarrow$) Let $u_{n-1}(\mathbf{x}) = \mathbf{c} \odot \mathbf{x}$ for a vector of constants $\mathbf{c} \in \mathbb{R}^d$. This implies $k_{n,l}(\mathbf{c} \odot \mathbf{x}; \boldsymbol{\phi}_n)$ where $\mathbf{c}$ scales the length scale vector in $\boldsymbol{\phi}_n$. As a result the covariance function $k_{n,l}(.,.)$ operates on the input space. So, $u_{n,l}(\mathbf{x})|u_{n-1}(\mathbf{x})$ is a stationary GP. □

Proposition 2. *Under the DGP model in (3.4),* $u_n(\mathbf{x})|u_{n-1}(\mathbf{x})$ *is a stationary GP if and only if* $u_{n-1}(\mathbf{x})$ *is a constant for any* $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$.

Proof. $\Rightarrow$) Let $u_n(\mathbf{x})|u_{n-1}(\mathbf{x})$ be a stationary GP. This implies that $k_n(\mathbf{x}; \boldsymbol{\phi}_n(u_{n-1}(\mathbf{x})))$ is a stationary covariace function. Hence covariance parameter $\boldsymbol{\phi}_n(u_{n-1}(\mathbf{x})))$ must be a constant and would not change respect to any space location $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$. To have this, $u_{n-1}(\mathbf{x})$ must be a constant for any $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$.

$\Leftarrow$) Let $u_{n-1}(\mathbf{x})$ be a constant. This implies that covariance parameter $\boldsymbol{\phi}_n(u_{n-1}(\mathbf{x})))$ in $k_n(\mathbf{x}; \boldsymbol{\phi}_n(u_{n-1}(\mathbf{x})))$ is a constant too and would not change respect to the space location $\mathbf{x}$. As a result $k_n(.,.)$ would be a stationary covariance function and $u_n(\mathbf{x})|u_{n-1}(\mathbf{x})$ is a stationary GP.

□

Remark 2. *Proposition 2 has practical implications that are helpful when one wants to do preliminary exploration on a dataset. Specifically, to diagnose if a stationary GP is a good enough solution for the the given data, one can fit a one hidden layer DGP to the data. If the first hidden layer is (almost) constant, a staionary GP will be an adequate solution. An illustration for this will be presented in Section 3.5.*

There are some advantages of the DGP of Dunlop et al. [2018] that lead us to consider it in our framework. As mentioned before, in the DGP model (3.4) covariance functions operate on the original input space rather than the warped input space. Retaining data locations in this form of the DGP allows us to explore the model, where the correlation as a function of the layer changes through the original input space. Specifically, this exploration can be done by visualizing the layers versus the original input space even in many dimensions in this form of the DGP. While getting these features in the compositional form (3.2) would be hard and a bit tricky as the input space is warped so much. In the next section, we detail how the DPG formulation in (3.4) can be considered as a Bayesian hierarchical model (BHM).

3.2.2 DGP as a Bayesian Hierarchical Model (BHM)

The notation used for the DGP models above is common in the computer science literature. On the other hand, it is not the conventional way of writing out a model among statisticians (i.e., first the likelihood is specified followed by prior distributions and hyper-prior distributions). The DGP formulation defined in (3.4) is now written as a Bayesian hierarchical model.

Let y^S be an output of deterministic simulator $\eta(.)$ at design location $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ and $u_{N+1}(\mathbf{x})$ is a DGP with N hidden layers. So we have

$$y^S = u_{N+1}(\mathbf{x}), \tag{3.5}$$

$$u_{N+1}(\mathbf{x})|u_N(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k_{N+1}(\mathbf{x}; \boldsymbol{\phi}_{N+1}(u_N(\mathbf{x}), \boldsymbol{\omega}_{N+1}), \boldsymbol{\psi}_{N+1})), \tag{3.6}$$

$$\vdots$$

$$u_1(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k_1(\mathbf{x}; \boldsymbol{\phi}_1)), \tag{3.7}$$

$$\boldsymbol{\phi}_1 \sim \mathbb{P}(\boldsymbol{\phi}_1) \quad , \quad \boldsymbol{\omega}_n \sim \mathbb{P}(\boldsymbol{\omega}_n) \quad , \quad \boldsymbol{\psi}_n \sim \mathbb{P}(\boldsymbol{\psi}_n), \tag{3.8}$$

for $n = 2, \ldots, N+1$ and $\mathbb{P}(.)$ is used to generically specify the prior distributions. As seen in (3.6), kernel parameters are separated into (i) parameters that are function of previous layer u_N and a vector of parameters $\boldsymbol{\omega}_{N+1}$ through $\boldsymbol{\phi}_{N+1}$, and (ii) parameters $\boldsymbol{\psi}_{N+1}$ in the kernel such as variance and smoothness parameters which do not depend on previous layer u_N. When there is no hidden layer ($N = 0$), $y^S = u_1(\mathbf{x})$ is a stationary GP in (3.7), where $\boldsymbol{\phi}_1$ is a vector of standard parameters in stationary covariance function $k_1(.,.)$. In (3.8) which is the last level of our hierarchical model, we introduce prior distributions for hyperparameters $\boldsymbol{\phi}_1$, $\{\boldsymbol{\omega}_n\}_{n=2}^{N+1}$ and $\{\boldsymbol{\psi}_n\}_{n=2}^{N+1}$ that need to be estimated. Hence the DGP form (3.4) with a data level as in (3.5), process levels as in (3.6) and (3.7) and a prior level as in (3.8) can be viewed as a BHM. For the observation vector $\mathbf{y}^S$, the likelihood is as

$$\mathbb{P}(\mathbf{y}^S|\ \mathbf{u}_N, \boldsymbol{\omega}_{N+1}, \boldsymbol{\psi}_{N+1}) \propto |\mathbf{K}|^{-\frac{1}{2}} \exp\Big\{ -\frac{1}{2}\ (\mathbf{y}^{S^T}\mathbf{K}^{-1}\mathbf{y}^S)\Big\}, \tag{3.9}$$

where $\mathbf{K} = k_{N+1}(\mathbf{X}, \mathbf{X}; \boldsymbol{\phi}_{N+1}(\mathbf{u}_N, \boldsymbol{\omega}_{N+1}), \boldsymbol{\psi}_{N+1})$ is an $n_S \times n_S$ covariance matrix obtained from the last hidden layer, $\mathbf{u}_N = u_N(\mathbf{X}) \in \mathbb{R}^{n_S}$. The hyperparameters $\boldsymbol{\phi}_1$, $\{\boldsymbol{\omega}_n\}_{n=2}^{N+1}$, $\{\boldsymbol{\psi}_n\}_{n=2}^{N+1}$ and unobserved random vectors $(\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N)$ have to be estimated.

Inference can be done in several ways for BHMs - for example via Markov Chain Monte Carlo (MCMC) (Metropolis et al. [1953], Hastings [1970]). However, accurate sampling methods such as MCMC for large designs are computationally infeasible due to evaluation of the likelihood which requires inverse of covariance matrix $\mathbf{K}$ at each step (Dunlop et al. [2018]). Specifically, in implementation of Dunlop et al. [2018], it is required to construct Cholesky decompositions of covariance matrices for each layer at every step through the MCMC as well. As a result, the computational burden is compounded when simulators are nested inside larger frameworks.

The next section describes how we build a DGP emulator by modifying the DGP form of Dunlop et al. [2018], followed by adapting a variational inference approach in Section 3.4 to overcome the computational issue encountered in large designs.

3.3 DGP as a Surrogate Model

The DGP of Damianou and Lawrence [2013] has been adapted as a statistical surrogate in computer model emulation (Monterrubio-Gomez et al. [2020], Radaideh and Kozlowski [2020], Rajaram et al. [2020], Sauer et al. [2020], Ming et al. [2021]). Methodology as surrogates using the DGP defined by Dunlop et al. [2018] for simulation experiments has not been developed. Also as discussed in previous section, this DGP form has features that makes it suitable in our application. These two reasons lead us to develop new methodology using the DGP of Dunlop et al. [2018] for our application.

3.3.1 Non-stationary Covariance Functions

Covariance functions determine important properties of a realization of a GP such as its variation and smoothness. So it is crucial to select covariance functions for the hierarchy of conditional GPs in (3.3) and (3.4) such that the DGP emulator can appropriately represent the simulator output.

Let $\rho(.)$ be a stationary correlation function, where correlation between any two outputs at locations $\mathbf{x}$ and $\mathbf{x}'$ depends on the Euclidean distance $\| \mathbf{x} - \mathbf{x}' \|_2$. Let the covariance function $k_1(.,.)$ in equation (3.3) be defined by $k_1(\mathbf{x}, \mathbf{x}'; \sigma_1^2) = \sigma_1^2 \rho(\| \mathbf{x} - \mathbf{x}' \|_2)$. Dunlop et al. [2018] apply the approach of Paciorek and Schervish [2004] discussed in 2.1.2 to construct non-stationary covariance functions $k_n(.,.)$ in equation (3.4) for $n > 1$ using

$$k_n(\mathbf{x}, \mathbf{x}'; \Sigma(\mathbf{x}), \Sigma(\mathbf{x}')) = \sigma_n^2 \frac{|\Sigma(\mathbf{x})|^{1/4} |\Sigma(\mathbf{x}')|^{1/4}}{|(\Sigma(\mathbf{x}) + \Sigma(\mathbf{x}'))/2|^{1/2}} \; \rho(\sqrt{Q(\mathbf{x}, \mathbf{x}')}), \tag{3.10}$$

where $\Sigma : \mathbb{R}^d \to \mathbb{R}^{d \times d}$ is a $d \times d$ matrix. The quadratic form

$$Q(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T \left(\frac{\Sigma(\mathbf{x}) + \Sigma(\mathbf{x}')}{2} \right)^{-1} (\mathbf{x} - \mathbf{x}'), \qquad \mathbf{x}, \mathbf{x}' \in \mathcal{X} \subseteq \mathbb{R}^d \tag{3.11}$$

averages the kernel matrices for the two locations when computing the distance between $\mathbf{x}$ and $\mathbf{x}'$. Proposition 2 in Dunlop et al. [2018] shows that positive definiteness of covariance function $k_n(.,.)$ is satisfied if ρ is continuous with $\lim_{r \to \infty} \rho(r) = 0$.

In this work, we follow Dunlop et al. [2018] and choose kernel matrix $\Sigma(\mathbf{x})$ depending on the process $u_{n-1}(\mathbf{x})$ as

$$\Sigma(\mathbf{x}) = F(u_{n-1}(\mathbf{x})) \mathbf{I}_d, \tag{3.12}$$

where $F : \mathbb{R} \rightarrow \mathbb{R} \geq 0$ is a non-negative function, called the length scale function, and $\mathbf{I}_d$ is an identity matrix of the order d. Using (3.12), the non-stationary covariance function defined in (3.10) can be written as

$$k_n(\mathbf{x}, \mathbf{x}'; F(u_{n-1}(\mathbf{x})), F(u_{n-1}(\mathbf{x}'))) = \sigma_n^2 \frac{2^{d/2}[F(u_{n-1}(\mathbf{x}))]^{d/4}[F(u_{n-1}(\mathbf{x}'))]^{d/4}}{[F(u_{n-1}(\mathbf{x})) + F(u_{n-1}(\mathbf{x}'))]^{d/2}} \rho(\sqrt{Q(\mathbf{x}, \mathbf{x}')}), \tag{3.13}$$

where

$$\sqrt{Q(\mathbf{x}, \mathbf{x}')} = \frac{\parallel \mathbf{x} - \mathbf{x}' \parallel_2}{\sqrt{F(u_{n-1}(\mathbf{x})) + F(u_{n-1}(\mathbf{x}'))/2}}. \tag{3.14}$$

In (3.14), the distance between $\mathbf{x}$ and $\mathbf{x}'$ is scaled by square root of average of length scales at those locations. Hence, observations with the same distance in their inputs in the input space can have different correlations and as a result non-stationarity is produced in the process. Therefore, the hierarchy of conditional GPs in the DGP defined in (3.3) and (3.4) are constructed using non-stationary covariance functions $k_n(.,.)$ with kernel matrices $\Sigma(\mathbf{x})$ which are derived from the previous GP or layer, $u_{n-1}(\mathbf{x})$. In other words, each layer, through the length scale function, $F(.)$, can be interpreted as the length-scale of the following layer.

Remark 3. *Proposition 2 holds for the DGP constructed using the non-stationary covariance function defined in (3.13). Hence, if $u_{n-1}(\mathbf{x}) = c$ for a constant c at all $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$, then the covariance function $k_n(.,.)$ in (3.13) is stationary. The reason is that in this case the prefator will equal one and the kernel becomes*

$$k_n(\mathbf{x}, \mathbf{x}'; a) = \sigma_n^2 \ \rho(\frac{\parallel \mathbf{x} - \mathbf{x}' \parallel_2}{\sqrt{F(c)}}). \tag{3.15}$$

Equation (3.15) implies that for fixed distance between $\mathbf{x}$ and $\mathbf{x}'$, the bigger $F(c)$ is, the higher correlation between $\mathbf{x}$ and $\mathbf{x}'$ should be. The reason is that the distance is scaled by $\sqrt{F(c)}$ and this makes points to get closer together.

In this work, the stationary correlation function $\rho(.)$ is chosen from the stationary Matern family,

$$\rho(\mathbf{x}, \mathbf{x}'; \lambda, \nu) = \frac{1}{\Gamma(\nu)2^{\nu-1}} \Big(\frac{\parallel \mathbf{x} - \mathbf{x}' \parallel_2}{\lambda}\Big)^{\nu} K_{\nu}\Big(\frac{\parallel \mathbf{x} - \mathbf{x}' \parallel_2}{\lambda}\Big), \tag{3.16}$$

where $\Gamma(.)$ is the gamma-function, $\nu > 0$ is the smoothness parameter, $\lambda > 0$ is the length scale, and $K_{\nu}(.)$ denotes the modified Bessel function of the second kind of the order ν. Hence, the stationary Matern covariance function is defined as

$$k_1(\mathbf{x}, \mathbf{x}'; \sigma_1^2, \lambda, \nu) = \sigma_1^2 \rho(\mathbf{x}, \mathbf{x}'; \lambda, \nu), \tag{3.17}$$

where σ_1^2 is a variance parameter. For $n > 1$, $k_n(\mathbf{x}, \mathbf{x}'; F(u_{n-1}(\mathbf{x})), F(u_{n-1}(\mathbf{x}')))$ is a non-stationary version of the Matern covariance function as

$$\frac{\sigma_n^2 \; 2^{d/2}[F(u_{n-1}(\mathbf{x}))]^{d/4}[F(u_{n-1}(\mathbf{x}'))]^{d/4}}{\Gamma(\nu)2^{\nu-1}\big[F(u_{n-1}(\mathbf{x})) + F(u_{n-1}(\mathbf{x}'))\big]^{d/2}} \; \Big(2\sqrt{\nu Q(\mathbf{x}, \mathbf{x}')}\Big)^{\nu} K_{\nu}\Big(2\sqrt{\nu Q(\mathbf{x}, \mathbf{x}')}\Big), \tag{3.18}$$

where σ_n^2 is the variance parameter, and $\sqrt{Q(\mathbf{x}, \mathbf{x}')}$ is defined as (3.14).

In the next section, a new parameter is introduced in the non-stationary covariance function (3.13) that it controls the amount of smoothness in the DGP layers.

3.3.2 Controlling Smoothness of DGP Layers

The choice of length scale function $F(.) : \mathbb{R} \to \mathbb{R} \geq 0$ in the kernel matrix $\Sigma(.)$ in (3.12) allows us to propose a new parameter that can impact on the smoothness of the DGP layers. In Dunlop et al. [2018], typical choices for $F(u)$ are u^2 and $\exp(u)$. In this work, we propose $F(u) = \exp(\alpha u)$, where α is a new parameter that controls the level of smoothness in the DGP layers. We illustrate our observations analytically and numerically below.

Using the proposed length scale function $F(.)$, the kernel matrix $\Sigma(\mathbf{x})$ in (3.12) can be written as

$$\Sigma(\mathbf{x}) = \begin{bmatrix} \exp(\alpha u(\mathbf{x})) & 0 & \cdots & 0 \\ 0 & \exp(\alpha u(\mathbf{x})) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \exp(\alpha u(\mathbf{x})) \end{bmatrix}_{d\times d} . \tag{3.19}$$

For $\mathbf{x}, \mathbf{x}' \in \mathcal{X} \subseteq \mathbb{R}^d$ the quadratic form $Q(\mathbf{x}, \mathbf{x}')$ and the prefactor of the non-stationary covariance function in (3.13) are then written as

$$Q(\mathbf{x}, \mathbf{x}') = \frac{(x_1 - x_1')^2 + \cdots + (x_d - x_d')^2}{\Big[\frac{\exp(\alpha u(\mathbf{x})) + \exp(\alpha u(\mathbf{x}'))}{2}\Big]}, \tag{3.20}$$

and

$$\frac{\Big[\exp(\alpha u(\mathbf{x}))\Big]^{\frac{d}{4}}\Big[\exp(\alpha u(\mathbf{x}'))\Big]^{\frac{d}{4}}}{\Big[\frac{\exp(\alpha u(\mathbf{x})) + \exp(\alpha u(\mathbf{x}'))}{2}\Big]^{\frac{d}{2}}}, \tag{3.21}$$

respectively. Equations (3.20) and (3.21) show where the parameter α exactly impacts on the non-stationary covariance functions in the DGP. The following proposition aims to explain exactly what is happening in the non-stationary covariance function as the proposed parameter α changes from zero to infinity.

Proposition 3. *Under the DGP model in (3.4), let the non-stationary covariance function $k_n(.,.)$ be defined in (3.13) with the length scale function $F(u_{n-1}(\mathbf{x})) = \exp(\alpha u_{n-1}(\mathbf{x}))$ for*

$n > 1$ and $\alpha \geq 0$. Then increasing α from zero to ∞ decreases degree of smoothness of each layer.

Proof. Let $\alpha = 0$. Then the length scale function $F(u_{n-1}(\mathbf{x})) = \exp(\alpha u_{n-1}(\mathbf{x})) = 1$ for all $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$, i.e. length scale function does not change through the input space $\mathcal{X}$ in each layer. It follows that the prefactor in (3.21) equals 1 and the square root of the quadratic form in (3.20) becomes $\sqrt{Q(\mathbf{x}, \mathbf{x}')} = \| \mathbf{x} - \mathbf{x}' \|_2$. As a result, for $n > 1$ $k_n(\mathbf{x}, \mathbf{x}') = \sigma_n^2 \, \rho(\| \mathbf{x} - \mathbf{x}' \|_2)$, a stationary covariance function. This is where we reach stationarity in the DGP. As α increases, the degree of the smoothness of the DGP layers get smaller. To show this, we investigate correlation between any two outputs at $\mathbf{x}, \mathbf{x}' \in \mathcal{X} \subseteq \mathbb{R}^d$ in three situations.

Let $\alpha \to \infty$. (i) If $u_{n-1}(\mathbf{x}), u_{n-1}(\mathbf{x}') > 0$, then $F(u_{n-1}(\mathbf{x})) \to \infty$ and $F(u_{n-1}(\mathbf{x}')) \to \infty$ and the quadratic form $Q(\mathbf{x}, \mathbf{x}')$ in (3.20) goes to zero. Since $\rho(0) = 1$, then $\rho(\sqrt{Q(\mathbf{x}, \mathbf{x}')}) \to 1$. Furthermore, the perfactor in (3.21) can be written as

$$2^{d/2} \left[\frac{\exp(\alpha u(\mathbf{x})) \, \exp(\alpha u(\mathbf{x}'))}{(\exp(\alpha u(\mathbf{x})) + \exp(\alpha u(\mathbf{x}')))^2} \right]^{d/4}. \tag{3.22}$$

Since $\exp(.)$ is a non-negative function, the denominator in (3.22) goes to infinity faster than the numerator and the prefactor goes to zero. This implies that $k_n(\mathbf{x}, \mathbf{x}') \to 0$ at any inputs $\mathbf{x}$ and $\mathbf{x}'$, thereby having less smooth realizations. (ii) If $u_{n-1}(\mathbf{x}) > 0, u_{n-1}(\mathbf{x}') < 0$, similar to the case (i), $k_n(\mathbf{x}, \mathbf{x}') \to 0$. (iii) If $u_{n-1}(\mathbf{x}), u_{n-1}(\mathbf{x}') < 0$, then $F(u_{n-1}(\mathbf{x})) \to 0$ and $F(u_{n-1}(\mathbf{x}')) \to 0$ and as a result the quadratic form $Q(\mathbf{x}, \mathbf{x}')$ in (3.20) goes to ∞. Since $\lim_{r \to \infty} \rho(r) = 0$, then $\rho(\sqrt{Q(\mathbf{x}, \mathbf{x}')}) \to 0$. Hence $k_n(\mathbf{x}, \mathbf{x}') \to 0$ at any inputs $\mathbf{x}$ and $\mathbf{x}'$, thereby having rougher realizations again. □

Remark 4. *Proposition 3 holds as α increases from zero to $-\infty$. In our work, for identifiability reasons for α and the u's, we specify that α is a non-negative scalar parameter which is the same in all the DGP layers having the non-stationary covariance function.*

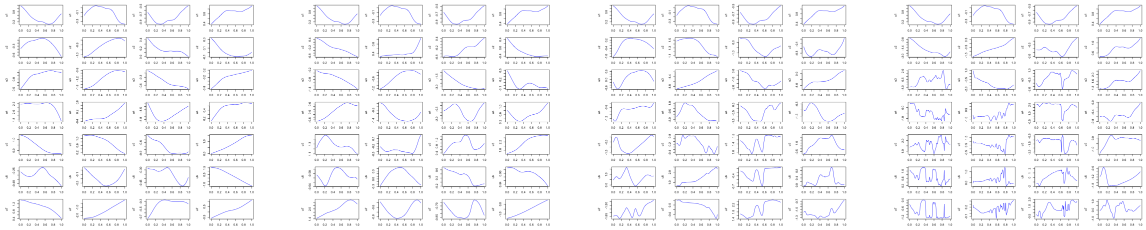
Remark 5. *If $u_{n-1}(\mathbf{x}) = c$ for a constant c at any $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$, then with the choice of $F(u_{n-1}) = \exp(\alpha u_{n-1})$, equation (3.15) can be written as*

$$k_n(\mathbf{x}, \mathbf{x}') = \sigma_n^2 \, \rho(\frac{\| \mathbf{x} - \mathbf{x}' \|_2}{\sqrt{\exp(\alpha c)}}).$$

If c is a positive constant, as α increases then $\exp(\alpha c)$ increases. As a result, the correlation $k_n(\mathbf{x}, \mathbf{x}')$ becomes larger and $u_n(.)$ gets smoother. If c is a negative constant, as α increases then $\exp(\alpha c)$ decreases. As a result correlation $k_n(\mathbf{x}, \mathbf{x}')$ becomes smaller and $u_n(.)$ becomes rougher.

To numerically illustrate the impact of the parameter α on the level of smoothness in the the DGP layers, we investigate realizations of the DGP generated using the non-stationary covariance functions defined in (3.13). We choose the stationary Matern covariance function in (3.17) with $\sigma_1^2 = 1$, $\lambda = 0.5$ and $\nu = 2.5$ for $k_1(.,.)$, and the non-stationary Matern covariance function in (3.18) with $\sigma_n^2 = 1$ for $k_n(.,.)$, $n = 2, \ldots, 7$ with the choice of $F(u) = \exp(\alpha u)$.

Panels (a), (b), (c) and (d) in Figure 3.2 show four independent realizations of the first seven layers $u_1, \ldots, u_7$ from the proposed DGP with $\alpha = 0.1, 1, 2, 3$, respectively. It can be seen that as α increases from 0.1 to 3, the layers become more wiggly from panel (a) to (d). Also, in panels (c) and (d) where $\alpha = 2, 3$, we can see more rougher samples through layer u_1 to layer u_7. These panels clearly illustrate the features outlined in Proposition 3.



(a) $F(u) = exp(0.1u)$ (b) $F(u) = exp(u)$ (c) $F(u) = exp(2u)$ (d) $F(u) = exp(3u)$

Figure 3.2: Four independent realizations of a DGP constructed by the stationary and non-stationary Matern covariance function

To measure the smoothness in the final response surface, the average of sums of absolute second derivatives of the last hidden layer for 500 samples is computed. As seen in each row of the Table 3.1, the scores become larger by increasing α from 0.1 to 3. This is expected since the less smooth the layer is, the bigger score is. Also the most significant increase occurs when α increases from 2 to 3 where the layers are most rough.

Table 3.1: Average of sums of $\mid d^2u_6/dx^2 \mid$

λ	$\alpha = 0.1$	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$
0.1	$2,327.4$	$3,183.7$	$18,616.8$	$226,127.6$
0.5	$2,342.7$	$3,100.3$	$16,246.2$	$217,751.7$
1	$2,323.2$	$2,975.7$	$17,352.1$	$209,352.8$
2	$2,332.6$	$3,167.3$	$16,189.5$	$220,885.9$

Remark 6. *Under the DGP model in (3.4), let the non-stationary covariance function $k_n(.,.)$ be defined in (3.13) with the length scale function $F(u_{n-1}(\mathbf{x})) = \exp(\alpha u_{n-1}(\mathbf{x}))$ for $n > 1$ and $\alpha \geq 0$. Then $\boldsymbol{\phi}_n(.,.)$ in (3.6) plays the same role as the non-negative function $F(.)$ and as a result $\boldsymbol{\omega}_n = \alpha$ and $\boldsymbol{\psi}_n = \sigma_n^2$ for $n = 2, \ldots, N+1$.*

It will be illustrated at the end of this chapter that estimating the proposed parameter α impacts the performance of our DGP emulator. On the other hand, as discussed in Subsection 3.2.2, doing inference on the DGP emulator using MCMC is computationally infeasible in large designs. In the next section, we adapt a variational inference approach that aims to not only estimate posterior distribution of smoothness parameter α but also overcome this computational issue.

3.3.3 Other Possible Innovations

There are other possible innovations to the DGP model in (3.4) through (i) specifying proposed parameter α in different ways and (ii) changing dimensionality layers, which we only mention some of them here as future work.

(i) The smoothness parameter α can be specified in the non-stationary covariance function $k_n(.,.)$ in different ways. One is that α can be different in all non-stationary layers. In this case, for $n = 2, \ldots, N+1$, $k_n(.,.)$ has its own smoothness parameter which has to be estimated in each layer. Therefore, the smoothness level of the layers is controlled differently and separately. Another way is that at each coordinate dimension of the input space $\mathcal{X} \subseteq \mathbb{R}^d$, the smoothness parameter α can be specified differently, i.e. $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_d)$ where d is dimension of the input space. Hence, for every input $\mathbf{x} \in \mathcal{X}$ kernel matrix $\Sigma(\mathbf{x})$ in (3.12) is written as

$$\Sigma(\mathbf{x}) = \begin{bmatrix} \exp(\alpha_1 u(\mathbf{x})) & 0 & \cdots & 0 \\ 0 & \exp(\alpha_2 u(\mathbf{x})) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \exp(\alpha_d u(\mathbf{x})) \end{bmatrix}_{d\times d}. \tag{3.23}$$

Then the quadratic form $Q(\mathbf{x}, \mathbf{x}')$ and the prefactor of the non-stationary covariance function in (3.13) are written as

$$Q(\mathbf{x}, \mathbf{x}') = \frac{(x_1 - x_1')^2}{\left[\frac{\exp(\alpha_1 u(\mathbf{x})) + \exp(\alpha_1 u(\mathbf{x}'))}{2}\right]} + \cdots + \frac{(x_d - x_d')^2}{\left[\frac{\exp(\alpha_d u(\mathbf{x})) + \exp(\alpha_d u(\mathbf{x}'))}{2}\right]}, \tag{3.24}$$

$$\prod_{l=1}^{d} \frac{\Big[\exp(\alpha_l u(\mathbf{x}))\Big]^{\frac{1}{4}} \Big[\exp(\alpha_l u(\mathbf{x}'))\Big]^{\frac{1}{4}}}{\left[\frac{\exp(\alpha_l u(\mathbf{x})) + \exp(\alpha_l u(\mathbf{x}'))}{2}\right]^{\frac{1}{2}}}, \tag{3.25}$$

respectively. In quadratic form (3.24) the distance at each coordinate dimension is scaled differently by its corresponding smoothness parameter compared with (3.20). Also the prefactor in (3.25) is a product of prefactors in each dimension. Although specifying α in above methods and their combination may have some advantages, more complexity is added to an already complex statistical model and more parameters need to be estimated.

(ii) In the DGP model in (3.4), the output of each layer is in one dimensional. We propose a variant of this DGP form with d-mensional layers, where d is dimension of the input space. Specifically, a DGP with N hidden layers in this new form is defined by sequences of functions $u_n : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}^d$ that are conditionally Gaussian,

$$u_{1,l}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k_{1,l}(\mathbf{x}; \boldsymbol{\phi}_1)) \quad , \qquad l = 1, \dots, d \tag{3.26}$$

$$u_{n,l}(\mathbf{x})|u_{n-1}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k_{n,l}(\mathbf{x}; \boldsymbol{\phi}_n(u_{n-1}(\mathbf{x})))) \quad , \qquad l = 1, \dots, d \tag{3.27}$$

where $u_{n,l}(\mathbf{x})$ represents the l^{th} component of $u_n(\mathbf{x}) \in \mathbb{R}^d$. Similar to the original form (3.4), $k_{1,l}(.,.)$ and $k_{n,l}(.,.)$ are specified as a stationary and non-stationary covariance function, respectively. The only difference in non-stationay covariance function $k_{n,l}(.,.)$ for $n > 1$ is that its kernel matrix, $\Sigma(\mathbf{x})$, in (3.12) is proposed as

$$\Sigma(\mathbf{x}) = \begin{bmatrix} F(u_{n-1,1}(\mathbf{x})) & 0 & \cdots & 0 \\ 0 & F(u_{n-1,2}(\mathbf{x})) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & F(u_{n-1,d}(\mathbf{x})) \end{bmatrix}_{d \times d} \quad , \tag{3.28}$$

where non-negative function $F : \mathbb{R} \to \mathbb{R} \geq 0$ is the length scale function. If $F(u) = exp(\alpha u)$ for a single smoothness parameter α, then the quadratic form $Q(\mathbf{x}, \mathbf{x}')$ and the prefactor of the non-stationary covariance function in (3.13) are written as

$$Q(\mathbf{x}, \mathbf{x}') = \frac{(x_1 - x_1')^2}{\left[\frac{\exp(\alpha u_{n-1,1}(\mathbf{x})) + \exp(\alpha u_{n-1,1}(\mathbf{x}'))}{2}\right]} + \cdots + \frac{(x_d - x_d')^2}{\left[\frac{\exp(\alpha u_{n-1,d}(\mathbf{x})) + \exp(\alpha u_{n-1,d}(\mathbf{x}'))}{2}\right]}, \tag{3.29}$$

$$\prod_{l=1}^{d} \frac{\Big[\exp(\alpha u_{n-1,l}(\mathbf{x}))\Big]^{\frac{1}{4}} \Big[\exp(\alpha u_{n-1,l}(\mathbf{x}'))\Big]^{\frac{1}{4}}}{\left[\frac{\exp(\alpha u_{n-1,l}(\mathbf{x})) + \exp(\alpha u_{n-1,l}(\mathbf{x}'))}{2}\right]^{\frac{1}{2}}}, \tag{3.30}$$

respectively. In comparison with (3.20), the quadratic form (3.29) scales the distance at each coordinate dimension by its corresponding component of the layer output. Also the prefactor in (3.30) is a product of prefactors in each dimension. Although, this new variant of the DGP benefits from having multidimensional layers in the DGP form (3.2) and non-stationary covariace functions in the DGP model (3.4), the model is more complex and has many more parameters. We defer a more thorough investigation of these new variants to future work.

3.4 Inference

In previous section, we proposed a DGP emulator that aims to capture non-standard features of complex computer models, such as our motivating application, the COMPAS model with large number of simulation runs. The additional computation required to fit both DGP models in (3.2) and (3.4) poses a challenge for larger sample sizes (Dunlop et al. [2018], Sauer et al. [2020]). Specifically implementation of Dunlop et al. [2018] using non-centerd MCMC algorithm of Chen et al. [2018] is computationally expensive when data is abundant due to the form of the likelihood in (3.9). Advances in variational inference provide solutions to deal with the computational burden in this setting. This section is started by a review of VI methods applied in the DGP form (3.2).

3.4.1 Related Work

Variational inference has been successfully used to perform inference for the DGP in (3.2) (Damianou and Lawrence [2013], Hensman and Lawrence [2014], Dai et al. [2016], Salimbeni and Deisenroth [2017]). In the approach of Damianou and Lawrence [2013], extending the seminal work on variational sparse GPs by Titsias [2009], the number of variational parameters increases linearly with the number of training data which hinders the use of this method for large scale datasets. An extension of this work is proposed in Dai et al. [2016]. A nested variational scheme is introduced by Hensman and Lawrence [2014] that only requires a variational distribution over the inducing outputs, removing the parameter scaling problem of Damianou and Lawrence [2013]. However, both approaches of Hensman and Lawrence [2014] and Dai et al. [2016] have not been fully evaluated on medium to large scale datasets.

Salimbeni and Deisenroth [2017] introduce the doubly stochastic variational inference (DVSI) method for inference with the DGP model of Damianou and Lawrence [2013], which allows scalability to large datasets with an effective performance. They employed a sparse inducing point variational framework (Matthews et al. [2016], Matthews [2017]) and two sources of stochasticity in the evaluation of the ELBO to achieve scalability to arbitrarily large data. Also, their implementation has been integrated with GPflow (Matthews et al. [2017]), an open-source GP framework built on top of Tensorflow (Abadi et al. [2015]).

In the next subsection, we adapt the DSVI (i) to be suitable for the DGP model of Dunlop et al. [2018] modified by our proposed smoothness parameter α and (ii) to our framework, emulation of deterministic computer models which has not been done yet in this form of the DGP. Our adapted approach allows us to explore posterior distribution of the smoothness of the model response surface and is demonstrated to preserve accuracy with uncertainty measures for arbitrary large designs.

3.4.2 Fitting the DGP Emulator

Recall that $y^S = \eta(\mathbf{x})$, the scalar output of the deterministic computer model $\eta(.)$ at input $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$. The inputs are typically scaled so that $\mathcal{X}$ is the d-dimensional unit cube. Let $\mathbf{X} = \left[\mathbf{x}_1, \ldots, \mathbf{x}_{n_S}\right]^T$ be the $n_S \times d$ design matrix and $\mathbf{y}^S = (y_1^S, \ldots, y_{n_S}^S)^T$ be the corresponding outputs of the simulator. Now we build our DGP emulator described in previous section, with N hidden layers $(\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N)$ for the computer model $\eta(.)$, where

$\mathbf{u}_n = u_n(\mathbf{X}) \in \mathbb{R}^{n_S}$, $n = 1, \ldots, N$ are unobserved random vectors which are used to discover the covariance among the given simulation outputs.

Following Salimbeni and Deisenroth [2017], in each layer we define an additional set of m inducing locations where $m \ll n_S$, i.e. $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_N$ such that $\mathbf{Z}_n \in \mathbb{R}^{m \times d}$, $n = 1, \ldots, N$. Inducing variables $(\tilde{\mathbf{u}}_1, \tilde{\mathbf{u}}_2, \ldots, \tilde{\mathbf{u}}_N)$ are unobserved at their corresponding inducing locations, i.e. $\tilde{\mathbf{u}}_{\mathbf{n}} = \eta(\mathbf{Z}_n)$ such that $\tilde{\mathbf{u}}_n \in \mathbb{R}^m$ has the same prior as the $\mathbf{u}_n$. We choose a GP prior with a zero mean function in each layer (our first adjustment in DSVI). Explicitly, in the first layer the joint GP prior is factorised as

$$\mathbb{P}(\mathbf{u}_1, \tilde{\mathbf{u}}_1; \mathbf{X}, \mathbf{Z}_1) = \mathbb{P}(\mathbf{u}_1 | \tilde{\mathbf{u}}_1; \mathbf{X}, \mathbf{Z}_1) \mathbb{P}(\tilde{\mathbf{u}}_1; \mathbf{Z}_1), \tag{3.31}$$

where $\mathbb{P}(\mathbf{u}_1 | \tilde{\mathbf{u}}_1; \mathbf{X}, \mathbf{Z}_1) = \mathcal{N}(\mathbf{u}_1 | \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $\mathbb{P}(\tilde{\mathbf{u}}_1; \mathbf{Z}_1) = \mathcal{N}(\tilde{\mathbf{u}}_1 | \mathbf{0}, k_1(\mathbf{Z}_1, \mathbf{Z}_1))$, $k_1(.,.)$ is a stationary covariance function and for $i, j = 1, \ldots, n_S$

$$[\boldsymbol{\mu}_1]_i = \boldsymbol{\Gamma}_1(\mathbf{x}_i)^T \tilde{\mathbf{u}}_1,$$

$$[\boldsymbol{\Sigma}_1]_{ij} = k_1(\mathbf{x}_i, \mathbf{x}_j) - \boldsymbol{\Gamma}_1(\mathbf{x}_i)^T k_1(\mathbf{Z}_1, \mathbf{Z}_1) \boldsymbol{\Gamma}_1(\mathbf{x}_j),$$

and $\boldsymbol{\Gamma}_1(\mathbf{x}_i) = k_1(\mathbf{Z}_1, \mathbf{Z}_1)^{-1} k_1(\mathbf{Z}_1, \mathbf{x}_i)$. Our next modifications in DSVI appear in the upcoming layers, where non-stationanry covariance functions $k_n(.,.)$ in (3.13) are used with the proposed length scale function, i.e. $F(u) = \exp(\alpha u)$. In this setting, for $n = 2, \ldots, N$ the joint GP prior is factorized as

$$\mathbb{P}(\mathbf{u}_n, \tilde{\mathbf{u}}_n; \mathbf{u}_{n-1}, \mathbf{X}, \mathbf{Z}_n, \alpha) = \mathbb{P}(\mathbf{u}_n | \tilde{\mathbf{u}}_n; \mathbf{u}_{n-1}, \mathbf{X}, \mathbf{Z}_n, \alpha) \mathbb{P}(\tilde{\mathbf{u}}_n | \mathbf{Z}_n, \alpha) \mathbb{P}(\alpha), \tag{3.32}$$

where $\mathbb{P}(\alpha)$ is the prior of the new parameter α, $\mathbb{P}(\mathbf{u}_n | \tilde{\mathbf{u}}_n; \mathbf{u}_{n-1}, \mathbf{X}, \mathbf{Z}_n, \alpha) = \mathcal{N}(\mathbf{u}_n | \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$ and $\mathbb{P}(\tilde{\mathbf{u}}_n | \mathbf{Z}_n, \alpha) = \mathcal{N}(\tilde{\mathbf{u}}_n | \mathbf{0}, k_n(\mathbf{Z}_n, \mathbf{Z}_n; \boldsymbol{\delta}_{\mathbf{Z}_n}, \alpha))$. For $i, j = 1, \ldots, n_S$

$$[\boldsymbol{\mu}_n]_i = \boldsymbol{\Gamma}_n(\mathbf{x}_i, \mathbf{u}_{n-1}^i)^T \tilde{\mathbf{u}}_n,$$

$$[\boldsymbol{\Sigma}_n]_{ij} = k_n(\mathbf{x}_i, \mathbf{x}_j; \mathbf{u}_{n-1}^i, \mathbf{u}_{n-1}^j, \alpha) - \boldsymbol{\Gamma}_n(\mathbf{x}_i, \mathbf{u}_{n-1}^i)^T k_n(\mathbf{Z}_n, \mathbf{Z}_n; \boldsymbol{\delta}_{\mathbf{Z}_n}, \alpha) \boldsymbol{\Gamma}_n(\mathbf{x}_j, \mathbf{u}_{n-1}^j),$$

and $\boldsymbol{\Gamma}_n(\mathbf{x}_i, \mathbf{u}_{n-1}^i) = k_n(\mathbf{Z}_n, \mathbf{Z}_n; \boldsymbol{\delta}_{\mathbf{Z}_n}, \alpha)^{-1} k_n(\mathbf{Z}_n, \mathbf{x}_i; \boldsymbol{\delta}_{\mathbf{Z}_n}, \mathbf{u}_{n-1}^i, \alpha)$ for $\mathbf{u}_n^i := (\mathbf{u}_n)_i = u_n(\mathbf{x}_i)$. The non-stationary covariance functions operate on $\mathbf{X}$ and inducing locations $\mathbf{Z}_n$. Also outputs of the previous layers are used for modeling length scales at input locations $\mathbf{X}$. We specify $\boldsymbol{\delta}_{\mathbf{Z}_n} \in \mathbb{R}^m$ as unknown vector parameters representing length scale values at inducing locations. Therefore using (3.31) and (3.32) the joint density of the outputs and parameters to be estimated is

$$\mathbb{P}(\mathbf{y}^S, \{\mathbf{u}_n, \tilde{\mathbf{u}}_n\}_{n=1}^N, \alpha) = \mathbb{P}(\mathbf{y}^S | \mathbf{u}_N) \mathbb{P}(\mathbf{u}_1, \tilde{\mathbf{u}}_1; \mathbf{X}, \mathbf{Z}_1) \prod_{n=2}^{N} \mathbb{P}(\mathbf{u}_n, \tilde{\mathbf{u}}_n; \mathbf{u}_{n-1}, \mathbf{X}, \mathbf{Z}_n, \alpha) \tag{3.33}$$

$$= \mathbb{P}(\mathbf{y}^S|\mathbf{u}_N)\mathbb{P}(\mathbf{u}_1|\tilde{\mathbf{u}}_1;\mathbf{X},\mathbf{Z}_1)\mathbb{P}(\tilde{\mathbf{u}}_1;\mathbf{Z}_1)\Big(\prod_{n=2}^{N}\mathbb{P}(\mathbf{u}_n|\tilde{\mathbf{u}}_n;\mathbf{u}_{n-1},\mathbf{X},\mathbf{Z}_n,\alpha)\mathbb{P}(\tilde{\mathbf{u}}_n|\mathbf{Z}_n,\alpha)\Big)\mathbb{P}(\alpha).$$

We follow Salimbeni and Deisenroth [2017] and choose our DGP variational posterior as follows

$$q(\{\mathbf{u}_n,\tilde{\mathbf{u}}_n\}_{n=1}^N,\alpha) = \mathbb{P}(\mathbf{u}_1|\tilde{\mathbf{u}}_1;\mathbf{X},\mathbf{Z}_1)q(\tilde{\mathbf{u}}_1)\Big(\prod_{n=2}^{N}\mathbb{P}(\mathbf{u}_n|\tilde{\mathbf{u}}_n;\mathbf{u}_{n-1},\mathbf{X},\mathbf{Z}_n,\alpha)q(\tilde{\mathbf{u}}_n)\Big)q(\alpha), \tag{3.34}$$

where $q(\tilde{\mathbf{u}}_n)$ is chosen to be $\mathcal{N}(\mathbf{m}_n,\mathbf{s}_n)$ such that $\mathbf{m}_n \in \mathbb{R}^m$ and $\mathbf{s}_n \in \mathbb{R}^{m\times m}$ for $n = 1,\ldots,N$. Also we specify the variational posterior of α, $q(\alpha) = \mathcal{N}(m_\alpha, s_\alpha)$ for scalar parameters m_α and s_α. With this specification of $q(\tilde{\mathbf{u}}_n)$, the inducing variables can be marginalized from each layer analytically as

$$\begin{aligned}
q(\mathbf{u}_1|\mathbf{m}_1,\mathbf{s}_1,\mathbf{X},\mathbf{Z}_1) &= \int \mathbb{P}(\mathbf{u}_1|\tilde{\mathbf{u}}_1;\mathbf{X},\mathbf{Z}_1)q(\tilde{\mathbf{u}}_1)\ d\tilde{\mathbf{u}}_\mathbf{1} = \mathcal{N}(\mathbf{u}_1|\tilde{\boldsymbol{\mu}}_1,\tilde{\boldsymbol{\Sigma}}_1),\\
q(\mathbf{u}_n|\mathbf{m}_n,\mathbf{s}_n,\mathbf{u}_{n-1},\mathbf{X},\mathbf{Z}_n,\alpha) &= \int \mathbb{P}(\mathbf{u}_n|\tilde{\mathbf{u}}_n;\mathbf{u}_{n-1},\mathbf{X},\mathbf{Z}_n,\alpha)q(\tilde{\mathbf{u}}_n)q(\alpha)\ d\tilde{\mathbf{u}}_\mathbf{n}\\
&= \mathcal{N}(\mathbf{u}_n|\tilde{\boldsymbol{\mu}}_n,\tilde{\boldsymbol{\Sigma}}_n)q(\alpha),
\end{aligned} \tag{3.35}$$

where for $i,j = 1,\ldots,n_S$

$$\begin{aligned}
[\tilde{\boldsymbol{\mu}}_1]_i &:= \mu_{\mathbf{m}_1,\mathbf{Z}_1}(\mathbf{x}_i) = \boldsymbol{\Gamma}_1(\mathbf{x}_i)^T\mathbf{m}_1,\\
[\tilde{\boldsymbol{\mu}}_n]_i &:= \mu_{\mathbf{m}_n,\mathbf{Z}_n,\alpha}(\mathbf{x}_i,\mathbf{u}_{n-1}^i) = \boldsymbol{\Gamma}_n(\mathbf{x}_i,\mathbf{u}_{n-1}^i)^T\mathbf{m}_n,\\
[\tilde{\boldsymbol{\Sigma}}_1]_{ij} &:= \Sigma_{\mathbf{s}_1,\mathbf{Z}_1}(\mathbf{x}_i,\mathbf{x}_j) = k_1(\mathbf{x}_i,\mathbf{x}_j) - \boldsymbol{\Gamma}_1(\mathbf{x}_i)^T\Big[k_1(\mathbf{Z}_1,\mathbf{Z}_1)-\mathbf{s}_1\Big]\boldsymbol{\Gamma}_1(\mathbf{x}_j),\\
[\tilde{\boldsymbol{\Sigma}}_n]_{ij} &= k_n(\mathbf{x}_i,\mathbf{x}_j;\mathbf{u}_{n-1}^i,\mathbf{u}_{n-1}^j,\alpha) - \boldsymbol{\Gamma}_n(\mathbf{x}_i,\mathbf{u}_{n-1}^i)^T\Big[k_n(\mathbf{Z}_n,\mathbf{Z}_n;\boldsymbol{\delta}_{\mathbf{Z}_n},\alpha)-\mathbf{s}_n\Big]\boldsymbol{\Gamma}_n(\mathbf{x}_j,\mathbf{u}_{n-1}^j),
\end{aligned} \tag{3.36}$$

which is $[\tilde{\boldsymbol{\Sigma}}_n]_{ij} := \Sigma_{\mathbf{s}_n,\mathbf{Z}_n,\alpha}(\mathbf{x}_i,\mathbf{x}_j;\mathbf{u}_{n-1}^i,\mathbf{u}_{n-1}^j)$. The derivation for (3.35) is given in the Appendix A. Therefore, using (3.35) we obtain

$$q(\{\mathbf{u}_n\}_{n=1}^N,\alpha) = \Big(\prod_{n=1}^{N}\mathcal{N}(\mathbf{u}_n|\tilde{\boldsymbol{\mu}}_n,\tilde{\boldsymbol{\Sigma}}_n)\Big)q(\alpha). \tag{3.37}$$

Substituting (3.33), (3.34) and (3.37) into the general form for the ELBO (2.8), the evidence lower bound of our DGP, $\mathcal{L}_{DGP}$, is obtained as following

$$\mathcal{L}_{DGP} = \mathbb{E}_{q(\{\mathbf{u}_n,\tilde{\mathbf{u}}_n\}_{n=1}^N,\alpha)}\log\left(\frac{\mathbb{P}(\mathbf{y}^S,\{\mathbf{u}_n,\tilde{\mathbf{u}}_n\}_{n=1}^N,\alpha)}{q(\{\mathbf{u}_n,\tilde{\mathbf{u}}_n\}_{n=1}^N,\alpha)}\right)$$

$$= \int\cdots\int q(\{\mathbf{u}_n,\tilde{\mathbf{u}}_n\}_{n=1}^N,\alpha)\ \log\left(\mathbb{P}(\mathbf{y}^S|\mathbf{u}_N)\times\frac{\mathbb{P}(\tilde{\mathbf{u}}_1;\mathbf{Z}_1)}{q(\tilde{\mathbf{u}}_1)}\times\frac{\prod_{n=2}^N\mathbb{P}(\tilde{\mathbf{u}}_n|\mathbf{Z}_n,\alpha)}{\prod_{n=2}^N q(\tilde{\mathbf{u}}_n)}\times\frac{\mathbb{P}(\alpha)}{q(\alpha)}\right)$$

$d\{\mathbf{u}_n, \tilde{\mathbf{u}}_n\}_{n=1}^N d\alpha$

$$= \int \cdots \int q(\{\mathbf{u}_n\}_{n=1}^N, \alpha) \log\Big(\prod_{i=1}^{n_S} \mathbb{P}(y_i^S|\mathbf{u}_N^i) \Big)\ d\{\mathbf{u}_n\}_{n=1}^N d\alpha + \int q(\tilde{\mathbf{u}}_1) \log\Big(\frac{\mathbb{P}(\tilde{\mathbf{u}}_1; \mathbf{Z}_1)}{q(\tilde{\mathbf{u}}_1)} \Big) d\tilde{\mathbf{u}}_1$$

$$+ \int \cdots \int q(\{\tilde{\mathbf{u}}_n\}_{n=2}^N, \alpha) \log\left(\frac{\prod_{n=2}^N \mathbb{P}(\tilde{\mathbf{u}}_n|\mathbf{Z}_n, \alpha)}{\prod_{n=2}^N q(\tilde{\mathbf{u}}_n)} \right)\ d\{\tilde{\mathbf{u}}_n\}_{n=2}^N\ d\alpha + \int q(\alpha) \log\Big(\frac{\mathbb{P}(\alpha)}{q(\alpha)} \Big) d\alpha.$$

Therefore, the ELBO of the DGP can be formed as

$$\mathcal{L}_{DGP} = \sum_{i=1}^{n_S} \mathbb{E}_{q(\{\mathbf{u}_n\}_{n=1}^N, \alpha)} \Big(\ \log\ \mathbb{P}(y_i^S|\mathbf{u}_N^i)\ \Big) - \mathrm{KL}\Big(q(\tilde{\mathbf{u}}_1) \parallel \mathbb{P}(\tilde{\mathbf{u}}_1; \mathbf{Z}_1) \Big)$$

$$- \mathbb{E}_{q(\alpha)} \Big[\sum_{n=2}^N \mathrm{KL}\Big(q(\tilde{\mathbf{u}}_n) \parallel \mathbb{P}(\tilde{\mathbf{u}}_n; \mathbf{Z}_n, \alpha) \Big) \Big] - \mathrm{KL}\Big(q(\alpha) \parallel \mathbb{P}(\alpha) \Big). \tag{3.38}$$

To evaluate the ELBO, it is required to compute the first expectation term at each design point $\mathbf{x}_i$ for $i = 1, \ldots, n_S$. That is,

$$E_i := \mathbb{E}_{q(\{\mathbf{u}_n\}_{n=1}^N, \alpha)} \Big(\ \log\ \mathbb{P}(y_i^S|\mathbf{u}_N^i)\ \Big) = \mathbb{E}_{q(\{\mathbf{u}_n\}_{n=1}^N, \alpha)} \Big(\ \log\ \mathbb{P}(y_i^S|u_N(\mathbf{x}_i))\ \Big).$$

The expectation term E_i is approximated with a Monte Carlo (MC) sample from the variational posterior in (3.37) and is performed using univariate Gaussians through the re-parameterization trick (Kingma et al. [2015], Rezende et al. [2014]). Specifically, we first sample $\alpha_t \sim \mathcal{N}(m_\alpha, s_\alpha) = q(\alpha)$ and $(\epsilon_n^i)_t \sim \mathcal{N}(0, 1)$, where $t = 1, \ldots, T$ and $n = 1, \ldots, N$ represent indices of MC sampling iterations and number of DGP layers, respectively. Then, recursively the sampled variables $(\hat{\mathbf{u}}_1^i)_t \sim q(\mathbf{u}_1^i|\mathbf{m}_1, \mathbf{s}_1, \mathbf{x}_i, \mathbf{Z}_1)$ and $(\hat{\mathbf{u}}_n^i)_t \sim q(\mathbf{u}_n^i|\mathbf{m}_n, \mathbf{s}_n, (\hat{\mathbf{u}}_{n-1}^i)_t, \mathbf{x}_i, \mathbf{Z}_n, \alpha_t)$, $n = 2, \ldots, N$ are drawn as

$$(\hat{\mathbf{u}}_1^i)_t = \mu_{\mathbf{m}_1, \mathbf{Z}_1}(\mathbf{x}_i) + (\epsilon_1^i)_t \sqrt{\Sigma_{\mathbf{s}_1, \mathbf{Z}_1}(\mathbf{x}_i, \mathbf{x}_i)}, \tag{3.39}$$

$$(\hat{\mathbf{u}}_n^i)_t = \mu_{\mathbf{m}_n, \mathbf{Z}_n, \alpha_t}(\mathbf{x}_i, (\hat{\mathbf{u}}_{n-1}^i)_t) + (\epsilon_n^i)_t \sqrt{\Sigma_{\mathbf{s}_n, \mathbf{Z}_n, \alpha_t}(\mathbf{x}_i, \mathbf{x}_i; (\hat{\mathbf{u}}_{n-1}^i)_t)}, \tag{3.40}$$

where $\hat{\mathbf{u}}_n^i = \hat{u}_n(\mathbf{x}^i) \in \mathbb{R}$. At each input $\mathbf{x}_i$, this procedure is repeated T times to obtain an unbiased estimate by taking a Monte Carlo estimate, i.e.

$$E_i \approx \frac{1}{T} \sum_{t=1}^T \log\ \mathbb{P}\Big(y_i^S \mid (\hat{\mathbf{u}}_N^i)_t \Big),$$

where $(\hat{\mathbf{u}}_N^i)_t = \mu_{\mathbf{m}_N, \mathbf{Z}_N, \alpha_t}(\mathbf{x}_i, (\hat{\mathbf{u}}_{N-1}^i)_t) + (\epsilon_N^i)_t \sqrt{\Sigma_{\mathbf{s}_N, \mathbf{Z}_N, \alpha_t}(\mathbf{x}_i, \mathbf{x}_i; (\hat{\mathbf{u}}_{N-1}^i)_t)}$. Also all the KL terms in the ELBO can be computed analytically, and $\mathbb{E}_{q(\alpha)} \Big[\sum_{n=2}^N \mathrm{KL}\Big(q(\tilde{\mathbf{u}}_n) \parallel \mathbb{P}(\tilde{\mathbf{u}}_n; \mathbf{Z}_n, \alpha) \Big) \Big]$ is estimated by sampling $\alpha \sim \mathcal{N}(m_\alpha, s_\alpha)$. To achieve scalability as the data is large, the sum over E_i can be estimated using data sub-sampling. That is,

$$\mathcal{L}_{DGP} \approx \frac{n_S}{|\mathcal{B}|}\sum_{i\in|\mathcal{B}|} E_i - \mathrm{KL}\Big(q(\tilde{\mathbf{u}}_1) \parallel \mathbb{P}(\tilde{\mathbf{u}}_1;\mathbf{Z}_1)\Big) - \mathbb{E}_{q(\alpha)}\Big[\sum_{n=2}^{N}\mathrm{KL}\Big(q(\tilde{\mathbf{u}}_n) \parallel \mathbb{P}(\tilde{\mathbf{u}}_n;\mathbf{Z}_n,\alpha)\Big)\Big] \\ - \mathrm{KL}\Big(q(\alpha) \parallel \mathbb{P}(\alpha)\Big), \tag{3.41}$$

where $\mathcal{B}$ represents a batch or sub-sample of data. The reason for using data sub-sampling is that it decreases the time needed to evaluate the ELBO and thus perform optimization. This is an important aspect of the DSVI that it is highly scalable and is preserved in our modified version of the ELBO in (3.41).

With that said, to approximate the ELBO in our settings, three sources of stochasticity are used: (i) in estimating E_i through Monte Carlo sampling, (ii) sub-sampling the data, and (iii) approximating expectation of KL terms with sampling from $q(\alpha)$. Stochasticity sources of (i) and (ii) exist in the DSVI approach (Salimbeni and Deisenroth [2017]), although in our case we have an extra step in (i) which is a sampling from $q(\alpha)$. Since our inference approach aims to explore the posterior distribution of the new parameter α, the extra source of stochasticity (iii) is taken into account for evaluating the ELBO of our DGP.

The bound is maximized with respect to the variational parameters $\mathbf{m}_n$, $\mathbf{s}_n$, inducing related parameters $\mathbf{Z}_n$, $\boldsymbol{\delta}_{\mathbf{Z}_n}$ and model parameters (e.g. covariance function parameters) in each layer. Also m_α and s_α are found by maximizing the ELBO and as a result an estimate of the variational posterior of α is obtained. We perform the optimization of the ELBO using a loop procedure consisting of an optimization step with the natural gradient to perform the optimization with respect to the variational parameters of the last layer, then an optimization step using the momentum optimizer ADAM (Kingma and Ba [2014]) to perform the optimization for the other parameters in all layers. This optimization procedure has been adopted in DSVI and has shown better results than using only the Adam optimizer for all the layers and parameters (Chapter 3, Salimbeni [2020]).

Remark 7. *Estimating the proposed parameter α effectively impacts the performance of the DGP emulator. This will be illustrated in Section 3.5 where we compare our emulation approach in three cases: (i) α is estimated, (ii) α is optimized and (iii) $\alpha = 1$. Case (i) is the main purpose of our inference approach presented in this section. As mentioned before, in Dunlop et al. [2018], typical choices for the length scale function are $F(u) = u^2$ and $F(u) = \exp(u)$. Hence case (iii) is equivalent to the DGP in Dunlop et al. [2018] with the choice of $F(u) = \exp(u)$. In this case, the ELBO of the DGP is*

$$\mathcal{L}_{DGP} = \sum_{i=1}^{n_S}\mathbb{E}_{q(\{\mathbf{u}_n\}_{n=1}^{N})}\Big(\ log\ \mathbb{P}(y_i^S|\mathbf{u}_N^i)\ \Big) - \sum_{n=1}^{N} KL\Big(q(\tilde{\mathbf{u}}_n) \parallel \mathbb{P}(\tilde{\mathbf{u}}_n;\mathbf{Z}_n)\Big), \tag{3.42}$$

where

$$q(\{\mathbf{u}_n\}_{n=1}^N) = \prod_{n=1}^N \mathcal{N}(\mathbf{u}_n|\tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n).$$

Also in case (ii), α is just optimized along with other covariance function parameters through maximizing the ELBO in 3.42.

3.4.3 Prediction

After fitting the DGP emulator on the simulation runs $\mathbf{y}^S$, the next goal is that to make a prediction at a new input $\mathbf{x}^*$. The predictive distribution of $y^S(.)$ at the new input $\mathbf{x}^*$ is approximated through sampling from the variational posterior (3.37). To do this, first we sample $(\epsilon_n^*)_r \sim \mathcal{N}(0,1)$ and $\hat{\alpha}_r \sim \mathcal{N}(\hat{m}_\alpha, \hat{s}_\alpha)$, where $r = 1, \ldots, R$ and $n = 1, \ldots, N$ represent indices of sampling iterations and number of DGP layers, respectively. Then conditioned on the sampled estimated parameter $\hat{\alpha}_r$ and estimated variational parameters $\hat{\mathbf{Z}}_n$, $\hat{\mathbf{m}}_n$, $\hat{\mathbf{s}}_n$, $\hat{\boldsymbol{\delta}}_{\mathbf{Z}_n}$ obtained through optimizing the ELBO, the sampled variables $(\hat{u}_1^*)_r \sim q(\mathbf{u}_1|\hat{\mathbf{m}}_1, \hat{\mathbf{s}}_1, \mathbf{x}^*, \hat{\mathbf{Z}}_1)$ and $(\hat{u}_n^*)_r \sim q(\mathbf{u}_n|\hat{\mathbf{m}}_n, \hat{\mathbf{s}}_n, (\hat{u}_{n-1}^*)_r, \mathbf{x}^*, \hat{\mathbf{Z}}_n, \hat{\alpha}_r)$ for $n = 2, \ldots, N$ are drawn at $\mathbf{x}^*$ recursively as

$$(\hat{u}_1^*)_r = \mu_{\hat{\mathbf{m}}_1, \hat{\mathbf{Z}}_1}(\mathbf{x}^*) + (\epsilon_1^*)_r \sqrt{\Sigma_{\hat{\mathbf{s}}_1, \hat{\mathbf{Z}}_1}(\mathbf{x}^*, \mathbf{x}^*)}, \tag{3.43}$$

$$(\hat{u}_n^*)_r = \mu_{\hat{\mathbf{m}}_n, \hat{\mathbf{Z}}_n, \hat{\alpha}_r}(\mathbf{x}^*, (\hat{u}_{n-1}^*)_r) + (\epsilon_n^*)_r \sqrt{\Sigma_{\hat{\mathbf{s}}_n, \hat{\mathbf{Z}}_n, \hat{\alpha}_r}(\mathbf{x}^*, \mathbf{x}^*; (\hat{u}_{n-1}^*)_r)}, \tag{3.44}$$

where $\hat{u}_n^* = \hat{u}_n(\mathbf{x}^*) \in \mathbb{R}$ and

$$\begin{gathered}
\mu_{\hat{\mathbf{m}}_1, \hat{\mathbf{Z}}_1}(\mathbf{x}^*) = \boldsymbol{\Gamma}_1(\mathbf{x}^*)^T \hat{\mathbf{m}}_1, \\
\mu_{\hat{\mathbf{m}}_n, \hat{\mathbf{Z}}_n, \hat{\alpha}_r}(\mathbf{x}^*, (\hat{u}_{n-1}^*)_r) = \boldsymbol{\Gamma}_n(\mathbf{x}^*, (\hat{u}_{n-1}^*)_r)^T \hat{\mathbf{m}}_n, \qquad (3.45) \\
\Sigma_{\hat{\mathbf{s}}_1, \hat{\mathbf{Z}}_1}(\mathbf{x}^*, \mathbf{x}^*) = k_1(\mathbf{x}^*, \mathbf{x}^*) - \boldsymbol{\Gamma}_1(\mathbf{x}^*)^T \Big[k_1(\hat{\mathbf{Z}}_1, \hat{\mathbf{Z}}_1) - \hat{\mathbf{s}}_1\Big] \boldsymbol{\Gamma}_1(\mathbf{x}^*), \\
\Sigma_{\hat{\mathbf{s}}_n, \hat{\mathbf{Z}}_n, \hat{\alpha}_r}(\mathbf{x}^*, \mathbf{x}^*; (\hat{u}_{n-1}^*)_r) = k_n(\mathbf{x}^*, \mathbf{x}^*; (\hat{u}_{n-1}^*)_r, \hat{\alpha}_r) - \boldsymbol{\Gamma}_n(\mathbf{x}^*, (\hat{u}_{n-1}^*)_r)^T \Big[k_n(\hat{\mathbf{Z}}_n, \hat{\mathbf{Z}}_n; \hat{\boldsymbol{\delta}}_{\mathbf{Z}_n}, \hat{\alpha}_r) - \hat{\mathbf{s}}_n\Big] \\
\boldsymbol{\Gamma}_n(\mathbf{x}^*, (\hat{u}_{n-1}^*)_r),
\end{gathered}$$

such that

$$\boldsymbol{\Gamma}_1(\mathbf{x}^*) = k_1(\hat{\mathbf{Z}}_1, \hat{\mathbf{Z}}_1)^{-1} k_1(\hat{\mathbf{Z}}_1, \mathbf{x}^*),$$

and

$$\boldsymbol{\Gamma}_n(\mathbf{x}^*, (\hat{u}_{n-1}^*)_r) = k_n(\hat{\mathbf{Z}}_n, \hat{\mathbf{Z}}_n; \hat{\boldsymbol{\delta}}_{\mathbf{Z}_n}, \hat{\alpha}_r)^{-1} k_n(\hat{\mathbf{Z}}_n, \mathbf{x}^*; \hat{\boldsymbol{\delta}}_{\mathbf{Z}_n}, (\hat{u}_{n-1}^*)_r, \hat{\alpha}_r).$$

As seen in the posterior predictive means and variances defined in (3.45), the optimized inducing locations $\hat{\mathbf{Z}}_n$ play an analogous role of the design $\mathbf{X}$ in usual kriging formula (2.2), which scale up computations in prediction as well. This sampling procedure is repeated R times to obtain the posterior sample of of $y^S(.)$ at $\mathbf{x}^*$ along with incorporating all sources of uncertainty from α and hidden layers in the predictive distribution. If we have $\mathbf{X}^* = [\mathbf{x}_1^*, \ldots, \mathbf{x}_p^*]^T$ as a prediction set, this procedure can be also proceed with computing full covariance matrices at all new inputs in this set, where each entry of the matrices is obtained with $\Sigma_{\hat{\mathbf{s}}_1,\hat{\mathbf{Z}}_1}(\mathbf{x}_i^*,\mathbf{x}_j^*)$ and $\Sigma_{\hat{\mathbf{s}}_n,\hat{\mathbf{Z}}_n,\hat{\alpha}_r}(\mathbf{x}_i^*,\mathbf{x}_j^*;.)$ in (3.45), for $i,j=1,\dots,p$.

3.5 Illustration

In this section, three synthetic examples are considered to illustrate the proposed approach. To do this, we compare our emulation methodology in three cases: (i) α is estimated, (ii) α is optimized and (iii) $\alpha=1$ and is fixed. In all examples, a DGP with two hidden layers is fitted, where the DGP is constructed by the stationary and non-stationary Matern covariance functions formulated as (3.17) and (3.18) with $\nu=2.5$, respectively. For estimating α in case (i), which is the main goal of our inference approach in Subsection 3.4.2, specification of the prior distribution for α is needed. It is also necessary to specify initial values for the parameters of the variational distributions. Also, for doing inference in cases (ii) and (iii), the ELBO specified in (3.42) is used, so there is no sampling step for α. To compare the performance of the DGP in these three cases, the following criteria are calculated
(1) Nash–Sutcliffe Efficiency (NSE)

$$\mathrm{NSE}=1-\frac{\mathrm{MSPE}}{\mathrm{Var}}, \tag{3.46}$$

where Var is the variance of true values $y^S(.)$ at prediction inputs $\mathbf{x}_1^*,\dots,\mathbf{x}_p^*$ and MSPE represents the mean square prediction error formulated as

$$\mathrm{MSPE}=\frac{1}{p}\sum_{i=1}^{p}\Big(\hat{y}^S(\mathbf{x}_i^*)-y^S(\mathbf{x}_i^*)\Big)^2,$$

where $\hat{y}^S(\mathbf{x}_i^*)$ is the posterior predictive mean at the new input $\mathbf{x}_i^*$. Similar to the coefficient of determination, the NSE (Nash and Sutcliffe [1970]) attempts to measure the proportion of variation that can be explained by a predictive model. NSE values close to 1 indicate that the emulator has performed well in terms of prediction accuracy.
(2) Average Relative Width of 95% Credible Intervals (ARW)

$$\mathrm{ARW}=\frac{1}{p}\sum_{i=1}^{p}\frac{\Big(Q_{0.975}^i-Q_{0.025}^i\Big)}{\Big|y^S(\mathbf{x}_i^*)\Big|}, \tag{3.47}$$

where Q_c^i is the c-th quantile of the posterior predictive samples at $\mathbf{x}_i^*$. ARW scales the interval by the magnitude of the response value, allowing for direct comparison of the interval length over different test functions and multiple test sets.

(3) 95% Coverage Probability (CP)

$$\mathrm{CP} = \frac{1}{p}\sum_{i=1}^{p} \mathbb{1}_{\{y^S(\mathbf{x}_i^*)\in(Q^i_{0.025}, Q^i_{0.975})\}}, \tag{3.48}$$

where $\mathbb{1}(.)$ denotes an indicator function. The 95% CP is the proportion of the true values $y^S(.)$ at prediction inputs that contain inside their 95% credible interval. This section is finished off by demonstrating the performance of our proposed methodology in the real-world application that motivated this work, which is the emulation of the COMPAS model with millions of simulation runs.

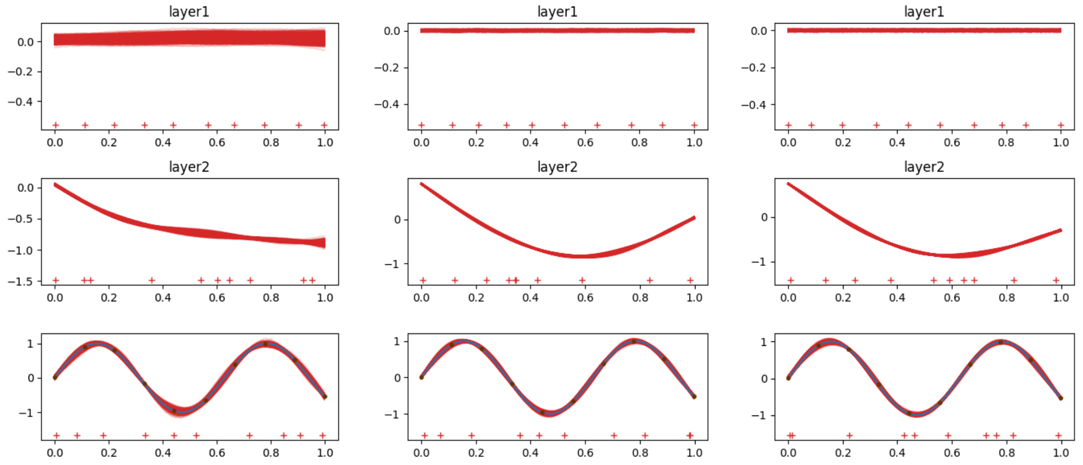
3.5.1 1-d Toy Models

In this section, two 1-d examples are presented, where two data sets with size 200 are generated from the following numerical models

$$f_1(x) = \sin(10x) \quad , \quad f_2(x) = \begin{cases} 1.35\ \cos(12\pi x) & x \in [0, 0.33] \\ 1.35 & x \in [0.33, 0.66] \\ 1.35\ \cos(6\pi x) & x \in [0.66, 1] \end{cases}, \tag{3.49}$$

evaluated on equally spaced inputs in $\mathcal{X} = [0, 1]$. Each data set is split into a training set with size n_S and a prediction set with size $p = 200 - n_S$. We fit a two hidden layer DGP with the training data using $m = n_S$ inducing points in each layer and emulate the value of the response over the prediction set with 1000 replications. In fitting, inducing locations are initialized at input locations and are optimized along with other variational parameters and model parameters through maximizing the ELBO.

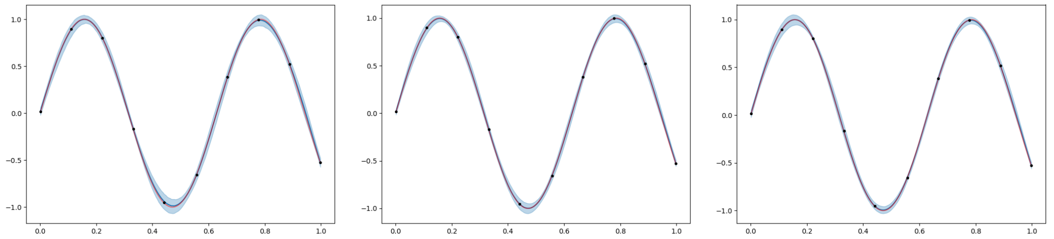
For the first 1-d example, the simple simulator $f_1(.)$ in (3.49) will be used to demonstrate the practical property of the DGP form (3.4) proved in Proposition 2, as the smoothness of the response surface of $f_1(.)$ is very well suited to a stationary GP emulator. Panels (a), (b) and (c) in Figure 3.3 show 1000 samples (red curves) from the posterior predictive distribution of the DGP emulator with two hidden layers fitted over a training set with 10 observed simulation data (brown dots) in three cases (i) α is estimated, (ii) α is optimized and (iii) $\alpha = 1$, respectively. For estimating α in case (i) the normal distribution $\mathcal{N}(2, 1)$ is chosen for $\mathbb{P}(\alpha)$ and $q(\alpha)$ is initialized with $\mathcal{N}(m_{\alpha\mathrm{ini}}, s_{\alpha\mathrm{ini}})$ where $m_{\alpha\mathrm{ini}} = 1$ and $s_{\alpha\mathrm{ini}} = 0.5$. Red plus dots show the optimized inducing locations and the true function (blue curve) is shown in the last row plots. As it is expected, the samples of the first hidden layer shown in the first row of the Figure 3.3 are almost constant in all cases (i), (ii) and (iii). Therefore, a stationary GP is likely an adequate solution for the given data and a complex DGP emulator with two hidden layers is not needed in this regime.



(a) (b) (c)

Figure 3.3: 1000 samples (red curve) from the posterior predictive distribution of the DGP emulator with two hidden layers fitted over a training set with 10 observed simulation data (brown dots) in case of (a) α is estimated, (b) α is optimized , (c) $\alpha = 1$.

To assess the prediction and uncertainty performance of the DGP model, 95% credible intervals are constructed with the resulting predictive posterior samples in cases (i), (ii) and (iii) and are highlighted with light blue color in panels (a), (b) and (c) of Figure 3.4, respectively. The predictive posterior mean (blue curve), the true function (red curve) and observed simulation data (balck dots) are also shown in each panel.



(a) NSE = 99.97% (b) NSE = 99.99% (c) NSE = 99.98%

Figure 3.4: 95% credible intervals highlighted with light blue color are constructed with the resulting predictive posterior samples in case of (a) α is estimated, (b) α is optimized, (c) $\alpha = 1$. The predictive posterior mean (blue line), the true function (red line) and the observed simulation data (black dots) are also shown in each panel.

As seen in Figure 3.4, the uncertainty captured by the credible intervals in panel (a) includes the posterior uncertainty of the parameter α, where α is sampled from $\hat{q}(\alpha) = \mathcal{N}(\hat{m}_\alpha, \hat{s}_\alpha)$ for $\hat{m}_\alpha = 1.2297$ and $\hat{s}_\alpha = 0.2968$ obtained through maximizing the ELBO in (3.38). In contrast, the 95% credible intervals in the two other panels are narrower and do not incorporate this uncertainty, as α is optimized ($\alpha_{opt} = 0.8838$) in panel (b) and is fixed

($\alpha = 1$) in panel (c). In spite of that, in all three cases the predictive model explains nearly the same proportion of variation. To further compare the methods, performance criteria including CP and ARW are computed for the DGP emulator with two hidden layers fitted on different number of observed simulation data in three cases (i), (ii) and (iii) and displayed in Table 3.2. As expected, ARW of the 95% credible intervals in case of estimating parameter α is more than in the other two cases, when $n_S = 10, 15, 20, 25$, although the 95% CPs are all the same.

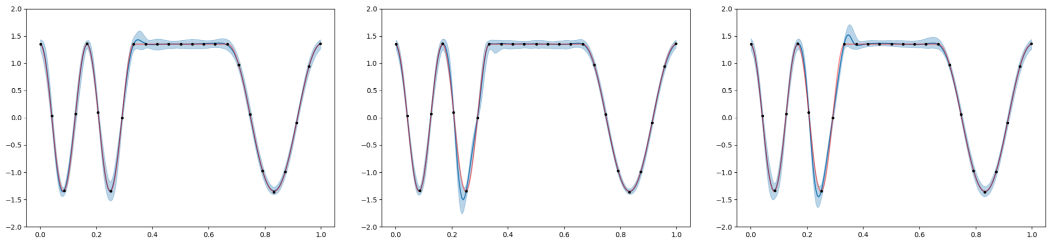
Table 3.2: Prediction accuracy of the DGP for three different methods and four sample sizes

	$n_S = 10$			$n_S = 15$		
	α_{est}	α_{opt}	$\alpha = 1$	α_{est}	α_{opt}	$\alpha = 1$
CP	100%	100%	100%	100%	100%	100%
ARW	0.6369	0.4845	0.4571	0.5331	0.2075	0.2836

	$n_S = 20$			$n_S = 25$		
CP	100%	100%	100%	100%	100%	100%
ARW	0.3114	0.1636	0.1872	0.3393	0.1321	0.1750

For the second 1-d example, we will use the example model $f_2(.)$ in (3.49), presented by Sauer et al. [2020], to demonstrate the impact of estimating parameter α on the performance of the DGP model. The reason that we choose $f_2(.)$ is that the model response varies significantly across the input space and is not well represented by a stationary GP emulator. Here, the DGP emulator with two hidden layers is fitted over a training set with size of 25 in three cases (i) α is estimated, (ii) α is optimized and (iii) $\alpha = 1$. In case (i), the normal distribution $\mathcal{N}(3, 1.2)$ is chosen for $\mathbb{P}(\alpha)$ and $q(\alpha)$ is initialized with $\mathcal{N}(m_{\alpha\text{ini}}, s_{\alpha\text{ini}})$ where $m_{\alpha\text{ini}} = 2.5$ and $s_{\alpha\text{ini}} = 1$.

The 95% credible intervals are constructed with the resulting predictive posterior samples (1000 samples) in cases (i), (ii) and (iii) and are highlighted with light blue color in panels (a), (b) and (c) of Figure 3.5, respectively. As seen in Figure 3.5, the predictive posterior mean (blue curve) in panel (a) compared to the other ones in panels (b) and (c) contains the true function (red curve), particularly in the area that $f_2(.)$ changes rapidly. Also, the predictive model explains higher proportion of the variation of the observed response (NSE=99.93%) in panel (a), where α is sampled from $\hat{q}(\alpha) = \mathcal{N}(\hat{m}_\alpha, \hat{s}_\alpha)$ for $\hat{m}_\alpha = 3.1075$ and $\hat{s}_\alpha = 0.2635$ obtained through maximizing the ELBO in (3.38). This illustrates that estimating parameter α has effectively impacted the prediction performance in this example compared to the other two cases, where α is just optimized ($\alpha_{opt} = 2.7535$) in panel (b) and is fixed ($\alpha = 1$) in panel (c).



(a) NSE=99.93% (b) NSE=99.50% (c) NSE=99.51%

Figure 3.5: 95% credible intervals highlighted with light blue color are constructed with the resulting predictive posterior samples in case of (a) α is estimated, (b) α is optimized, (c) $\alpha = 1$. The predictive posterior mean (blue line), the true function (red line) and the observed simulation data (black dots) are also shown in each panel.

Table 3.3 displays performance criteria including CP, ARW and NSE computed for the DGP emulator with two hidden layers fitted on different numbers of observed simulation data in three cases (i), (ii) and (iii). As shown in the table, ARW of the 95% credible intervals in case of estimating parameter α is larger than the two other cases, when $n_S = 25, 35, 45$. Also, the 95% CP and NSE tend to be high for each of the three inference procedures.

Table 3.3: Prediction accuracy of the DGP for three different methods and sample sizes

	$n_S = 25$			$n_S = 35$		
	α_{est}	α_{opt}	$\alpha = 1$	α_{est}	α_{opt}	$\alpha = 1$
CP	100%	95.43%	91.43%	100%	95.15%	95.76%
ARW	0.3835	0.3407	0.3511	0.7467	0.5157	0.4494
NSE	99.93%	99.50%	99.51%	99.88%	99.84%	99.83%

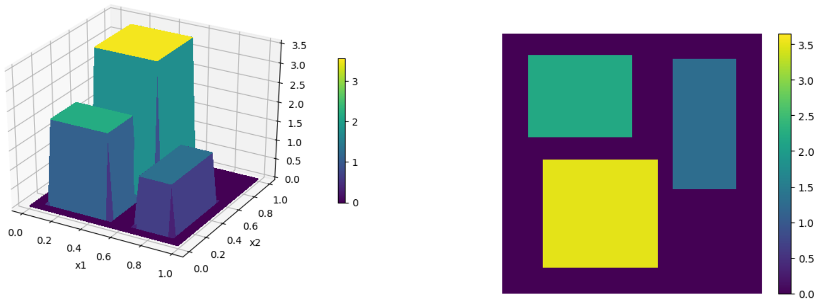
	$n_S = 45$		
	α_{est}	α_{opt}	$\alpha = 1$
CP	100%	94.84%	95.71%
ARW	0.4990	0.4764	0.3753
NSE	99.97%	99.92%	99.91%

3.5.2 2-d Toy Model

To further illustrate the performance of the proposed methodology, a 2-d example is conducted in this section. we consider the following 2-d piece-wise computer model

$$g(x_1, x_2) = \begin{cases} 1.3 & x_1 \in [0.66, 0.91] \text{ and } x_2 \in [0.4, 0.91] \\ 2.2 & x_1 \in [0.1, 0.5] \text{ and } x_2 \in [0.6, 0.92] \\ 3.5 & x_1 \in [0.15, 0.6] \text{ and } x_2 \in [0.1, 0.52] \\ 0 & \text{o.w.} \end{cases} \quad , \quad x_1, x_2 \in [0, 1] \tag{3.50}$$

shown in panel (a) of Figure 3.6, which illustrates the type of discontinuity we expect in the COMPAS model. Similar to the 1-d examples, we fit a two hidden layer DGP with the training data evaluated on a 25 by 25 grid in $\mathcal{X} = [0, 1]^2$, using $m = 200$ inducing points in each layer in three cases (i) α is estimated, (ii) α is optimized and (iii) $\alpha = 1$. In case (i), a normal distribution $\mathcal{N}(3.5, 1)$ is chosen for $\mathbb{P}(\alpha)$ and $q(\alpha)$ is initialized with $\mathcal{N}(m_{\alpha\text{ini}}, s_{\alpha\text{ini}})$ where $m_{\alpha\text{ini}} = 3$ and $s_{\alpha\text{ini}} = 1$. The response is emulated over the prediction set, a 70 by 70 grid in $\mathcal{X} = [0, 1]^2$, by sampling from the resulting predictive posterior (5000 samples) in all cases (i), (ii) and (iii). The true response values at prediction points is shown in a heatmap plot in panel (b) of Figure 3.6.

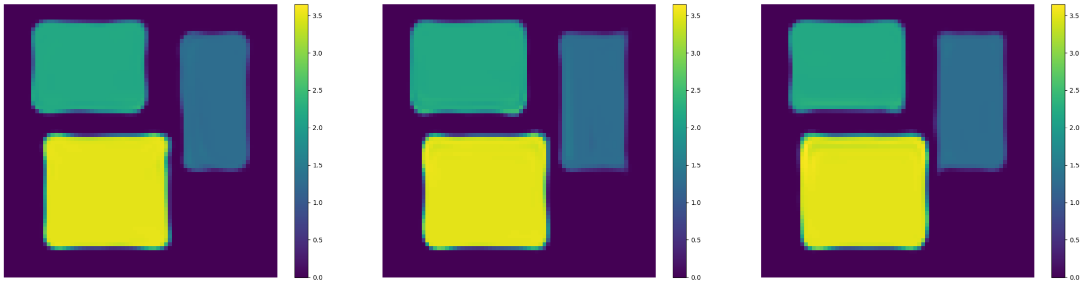


(a)

(b)

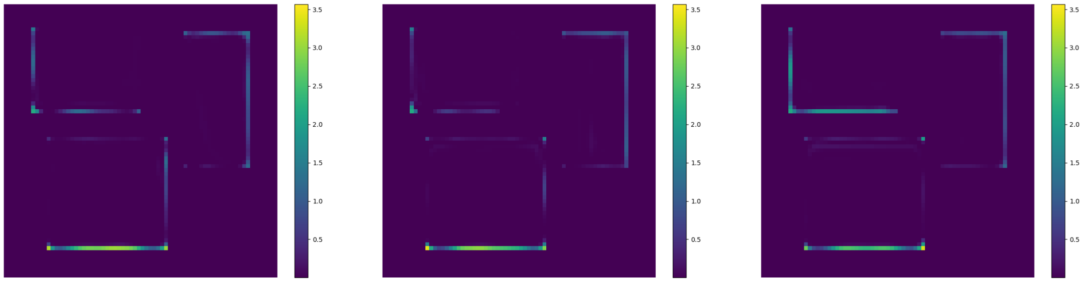
Figure 3.6: (a) 2-d illustrative computer model with regions of discontinuities (b) Heatmap of true function outputs at prediction points

The predictions and absolute prediction errors are compared in panels (a), (b) and (c) of Figures 3.7 and 3.8 in cases (i), (ii) and (iii), respectively. In the heatmaps, brighter color corresponds to larger predicted value and larger absolute prediction error. As seen in Figures 3.8, in all three cases, larger errors are observed around the boundaries of the three regions, where values of the model response change between zero and positive outputs. In panels (a) and (b) where that parameter α is estimated and optimized, it is evident that the performance of the DGP around the boundaries is greatly improved.



(a) NSE = 91.64% (b) NSE = 91.41% (c) NSE = 88.28%

Figure 3.7: Heatmap of the predictions in case of (a) α is estimated, (b) α is optimized, (c) $\alpha = 1$. Plots share the same color bar as given in the left side of each, where brighter colors indicate greater predicted values



(a) (b) (c)

Figure 3.8: Heatmap of absolute prediction errors in case of (a) α is estimated, (b) α is optimized, (c) $\alpha = 1$. Plots share the same color bar as given in the left side of each, where brighter colors indicate larger errors.

To numerically assess the prediction and uncertainty performance of the DGP model in all cases, 95% CP and NSE are computed and displayed in Table 3.4. As shown in the table the largest 95% CP in the predictive DGP model is reached in the case of estimating parameter α. This is due to including the extra uncertainty from estimating α. Also, the DGP model where the posterior distribution for α is estimated ($\hat{q}(\alpha) = \mathcal{N}(\hat{m}_\alpha, \hat{s}_\alpha)$ for $\hat{m}_\alpha = 3.2452$ and $\hat{s}_\alpha = 0.0308$) explains the largest amount of the response variability, although a similar result is achieved by optimizing α ($\alpha_{opt} = 3.3005$).

Table 3.4: Prediction accuracy of the DGP for three different methods

	α_{est}	α_{opt}	$\alpha = 1$
CP	93.82%	92.69%	90.27%
NSE	91.64%	91.41%	88.28%

3.5.3 COMPAS Model

Our proposed methodology is motivated by the need to emulate the COMPAS model. The input and the output of the COMPAS model are displayed in Table 3.5. There are two groups of input in the COMPAS model, (i) initial conditions of a binary star system, denoted by $\mathbf{x}$, which provide the state of the binary at formation; and (ii) the set of population parameters, denoted by $\mathbf{t}$, which is shared between all binaries in a population. The initial conditions of a binary system follow constrained distributions specified in the third column of the table. The true values of the parameters $\mathbf{t}$ are unknown and will be inferred (not in this thesis) by comparing the BBH properties predicted by simulations with different choices of $\mathbf{t}$ with field observations (Mandel and Farmer [2017]). The output of the COMPAS which we focus is the chirp mass of the formed BBH, a combination of the masses that are typically best measured from the gravitational-wave signal (Peters and Mathews [1963]). If no BBH is

Table 3.5: Input and output of COMPAS model

Input	**Range**	**Distribution**
Initial conditions: x		
m_1 : the mass of the initially more massive star	$[8,150]M_\odot$	Power law(-2.35)
m_2 : the mass of the initially less massive star	$(0.1\ M_\odot,\ m_1]$	Uniform
a: the initial orbital separation	[0.01,1000]AU	Power law(-1)
v$_i$: supernova natal kick vector for supernova i for $i = 1, 2$ including :		
v_i - magnitude of the supernova natal kick (km s^{-1})	$[0,\infty)$	Maxwellian
θ_i - polar angle defining the direction of the natal kick	$[0,\pi]$	Uniform
ϕ_i - azimuthal angle defining the direction of the natal kick	$[0,2\pi]$	Uniform
ω_i - mean anomaly	$[0,2\pi]$	Uniform
Population parameters: t		
Z: the metallicity	[0.0001, 0.03]	
α: the common envelope efficiency parameter	[0,10]	
σ: 1D root-mean-square value representing a typical supernova kick	[0,1000] km s^{-1}	
flbv: multiplication factor for the mass loss rate during the luminous blue variable (LBV) phase	[0,10]	
Output		
$\mathcal{M}_c$: chirp mass of BBH	$(0,150)M_\odot$ or NA	

observed, the output would be "NA", although in our work, the chirp mass is considered to be zero in this case.

As discussed in Section 3.1, the challenges in emulation of COMPAS model are the presence of regions of discontinuities in the response surface of the chirp mass and a large number of simulation runs with very low success rate for BBH formation. We apply our proposed emulation method to two million computer model runs where roughly 24% of the simulations resulted in a chirp mass output. In the remaining simulations, no BBH was formed and thus no chirp mass is computed. For these simulations, the parameters $\mathbf{t}$ were held constant at $Z = 0.001, \alpha = 1, \sigma = 265$ km/s, flbv $= 1.5$. Hence, in our example the input dimension is the dimensionality of initial conditions $\mathbf{x}$ (i.e. 11), as $\mathbf{t}$ is fixed.

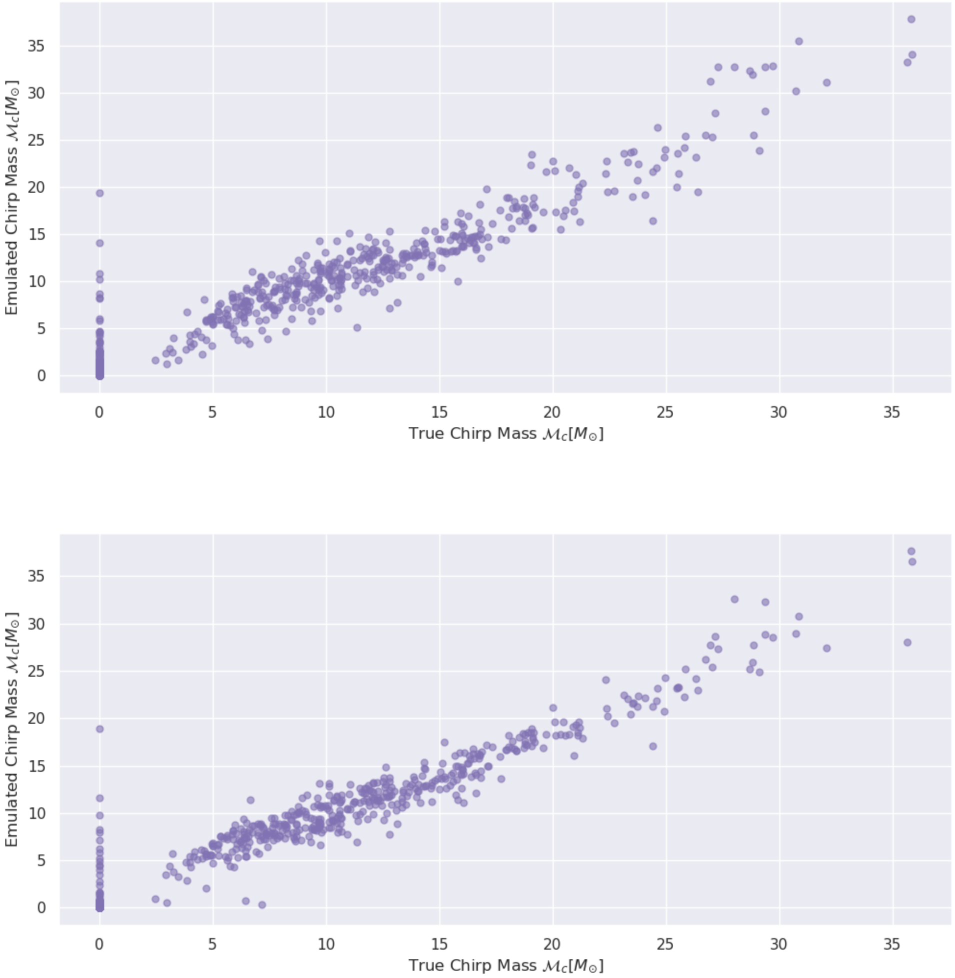


Figure 3.9: Emulated chirp mass against the true chirp mass for the DGP with two hidden layers (upper) and the DGP with three hidden layers (lower).

We standardized input variables of the simulation data to the 11-dimensional unit hypercube $[0,1]^{11}$. A prediction set with size of 1000 including 450 successful simulations (active points) was held out from the data to evaluate performance of our DGP emulator. We fit the DGP with two and three layers with the training data using 100 inducing points in each layer and emulate the response over the prediction set with 5000 samples from the resulting predictive posterior distribution. The DGPs are constructed using the stationary and non-stationary Matern covariance functions (3.17) and (3.18) with $\nu = 2.5$, respectively. For training, we approximated the ELBO (3.41) with a batch size of $10,000$ to achieve scalability. m_α and s_α were found by maximizing the ELBO to obtain an estimate of the variational posterior distribution of α, i.e. $\hat{q}(\alpha) = \mathcal{N}(\hat{m}_\alpha, \hat{s}_\alpha)$.

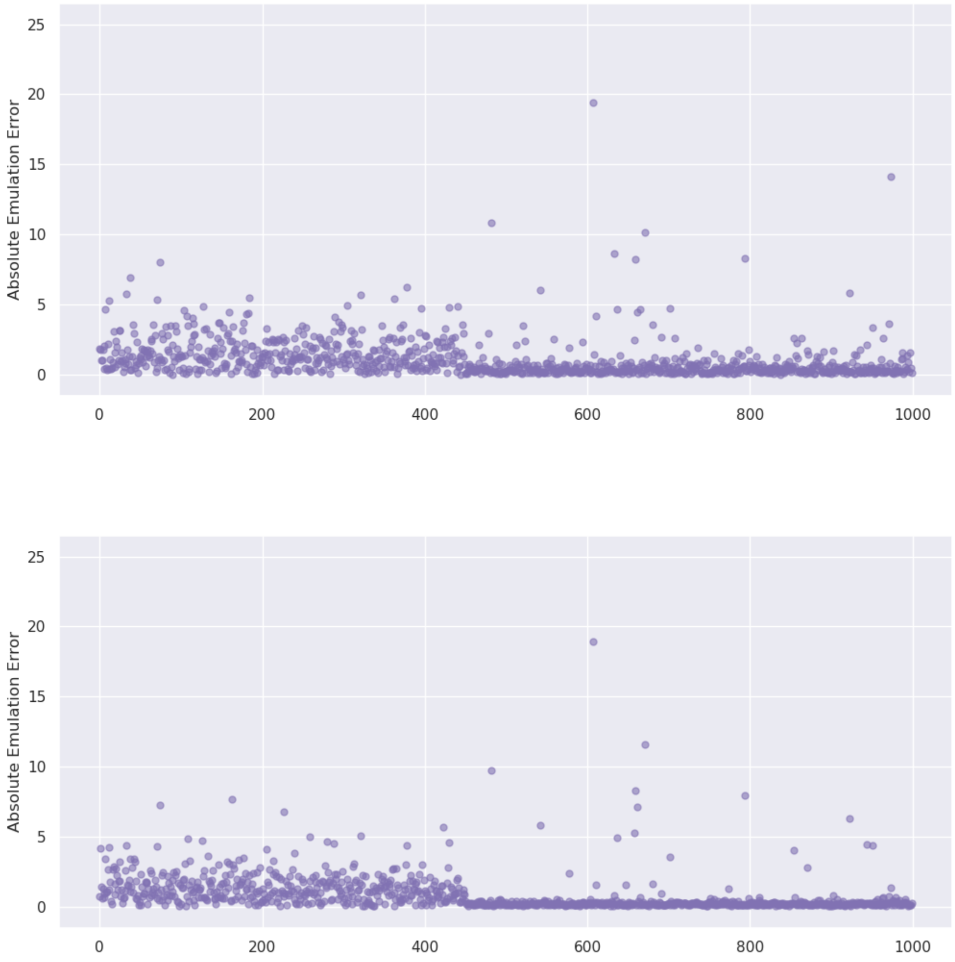


Figure 3.10: Absolute emulation errors for the DGP with two hidden layers (upper) and the DGP with three hidden layers (lower).

Figure 3.9 shows the plot of the emulated chirp mass against the true chirp mass at 1000 prediction points using the DGP with two hidden layers on the top and three hidden layers in the bottom. As seen in the figure, most of the active and non-active points lie at or near the 45 degree line, meaning that the proposed method appears successful at emulating the response. The largest errors appear where the true response was zero, but the emulator predicts a non-zero value and vice versa. Comparing two and three hidden layer DGPs in Figure 3.9, we see that points are more tightly centred around the 45 degree line for the three hidden layer model (lower plot).

Absolute emulation errors are plotted in Figure 3.10. As seen in this figure, most of the active and non-active points have small errors close to zero, although there are large errors in a few points due to the similar reason explained for previous figure. Also clearly we can see here that points are closer to the zero in the lower plot than the ones in the upper plot, meaning that the three layer DGP efficiently reduces the emulation error compared with the two layer DGP.

Table 3.6 displays the performance criteria including NSE and 95% CP computed for the DGP emulator with two and three hidden layers. As it is expected, the predictive model explains a larger proportion of the response variability (NSE=95.76%) in the three layer DGP, although its 95% CP is less than the two layer DGP. The estimated parameters $\hat{m}_\alpha$ and $\hat{s}_\alpha$ displayed in the table show that smoothness parameter α is estimated around 0.7. We attribute it to the size of the data and the higher dimensional input space of the COMPAS model. The training time of this large data set shown in the table illustrate how our emulation method can be computationally efficient. Also it shows that increasing number of DGP layers from two to three increases the computational time.

Table 3.6: COMPAS emulation results using two and three hidden layer DGP

# of Layer	$\hat{m}_\alpha$	$\hat{s}_\alpha$	NSE	CP	Training Time (1 iter)
2	0.7484	0.0003	94.31%	91.7	0.31 s
3	0.7695	0.0002	95.76%	90.4	0.35 s

3.6 Summary and Discussion

In this chapter, the DGP of Dunlop et al. [2018] was investigated and modified. This work was motivated by the application of emulation of COMPAS model, a binary population synthesis model that simulates the formation of binary black holes (BBHs). We proposed a non-stationary DGP emulator which can be adapted to the type of discontinuities of COMPAS response surface, and scales to the large number of simulation runs. Our approach can be applied to a large class of complex computer models, and scales to arbitrarily large simulation designs as well.

We generalized the notation of two broad forms of DGP (Damianou and Lawrence [2013], Dunlop et al. [2018]) to emphasize their differences and properties. Specifically, we addressed exactly what the DGP forms would look like if the data were actually a realization of a stationary GP in two propositions. Moreover, we showed how the DGP form of Dunlop et al. [2018] can be written as Bayesian hierarchical models. Some advantages of the DGP of Dunlop et al. [2018] such as operating covariance functions on the original input space rather than the warped input space, led us to consider it in our framework. Our DGP emulator was proposed by modifying DGP form of (Dunlop et al. [2018]) through introducing a new parameter (or parameters) that controls smoothness of the DGP layers. The impact of the proposed parameter(s) on the level of smoothness of layers were theoretically illustrated and numerically visualised in DGP realizations. Particularly, in Proposition 3, we proved how increasing new parameter α from zero to ∞ decreases degree of smoothness of DGP layers.

Doing inference on the DGP emulator using sampling methods such as MCMC is computationally infeasible in large designs. Hence, we employed a variational inference approach to overcome the computational issues and adapted it to be able to estimate posterior distribution of smoothness parameter α. Specifically, we adapted the doubly stochastic variational inference (DVSI) method introduced by Salimbeni and Deisenroth [2017] (i) to be suitable for the DGP model of Dunlop et al. [2018] modified by our proposed parameter α and (ii) to our framework, emulation of deterministic computer models. Our modified approach allowed us to explore posterior distribution of the smoothness of the model response surface and was demonstrated to preserve accuracy with uncertainty measures for arbitrary large designs.

Some potential avenues for future work were detailed in Subsection 3.3.3. Our possible innovations to the DGP model of Dunlop et al. [2018] could be through (i) specifying proposed parameter α in different ways and (ii) changing dimensionality layers. By combining methods (i) and (ii) new different variants of the DGP can be proposed. Although, these new DGP variants get the benefit of having multidimensional layers in the DGP form (3.2) and non-stationary covariace functions in the DGP model (3.4), more complexities are introduced into the original model (3.2) by adding extra unobserved latent variables in each dimension and adding extra parameters to be estimated. We defer a more thorough investigation of these new variants to future work.

Chapter 4

Sequential Experiment Design using DGP Emulator

4.1 Introduction

A practical problem of interest is the selection of computer model trials to improve the emulator performance. In other words, if we wish to make good predictions about the output of the computer model using the emulator, we need to consider what choice of inputs will lead to the best predictions. In Chapter 3, we introduced our DGP emulator to accommodate the non-stationary structures in complex computer models. Since we are interested in exploring regions of the input space that are more complicated in the response, such as a region of the input space with high variability in the response, stationary design strategies such as uniform and space-filling designs are unsuitable (Gramacy and Lee [2009], Volodina and Williamson [2020]). In this chapter, we propose to combine the non-stationary DGP model with a variance-based criterion to deviate from usual space-filling designs and guide the selection of future runs in more complex regions of the input space along with improving predictive accuracy of our DGP emulator.

In recent years, the DGP defined in Damianou and Lawrence [2013] has been used as a surrogate in sequential design of computer experiments (Dutordoir et al. [2017], Rajaram et al. [2020], Sauer et al. [2020], Hebbal et al. [2021]). Particularly, in Hebbal et al. [2021], expected improvement (EI) of Jones et al. [1998] has been implemented for the problem of Bayesian optimization using this DGP. In Rajaram et al. [2020], a strategy based on maximum variance criterion of MacKay [1992], known as active learning MacKay (ALM), is applied using this form of DGP. In Sauer et al. [2020], a sequential design is constructed using integrated mean-squared prediction error (IMSPE) and active learning strategy of Cohn [1994](ALC) with this DGP form.

In this chapter, we propose a sequential design approach using IMSPE with our DGP emulator, a modified version of the DGP in Dunlop et al. [2018]. In order to proceed with our method, we adapt the prediction method described in Section 3.4.3 in a way that nearest neighbor designs can be used. The variance-based criterion is able to effectively recognize that more data is needed where there is high variability in the response than where it is not. We also investigate the impact of refitting the model in batches of added design points in improving the emulator.

This chapter is organized as follows: In Section 4.2, a general scheme for constructing sequential designs is presented, followed by a brief review of the improvement function and expected improvement (EI). In Section 4.3, we introduce a new sequential design strategy for complex computer models using the DGP emulator. The performance of our proposed approach is demonstrated on the 2D toy example and our motivating application, the COMPAS model, in Section 4.4. We finish off with some discussion and avenues for further research in Section 4.5.

4.2 Sequential Design of Computer Experiments

The process of running a computer model at a variety of different input values is described as a computer experiment. A computer experiment may have objectives similar to those of a physical experiment, while often it may be more time and cost effective than running a physical experiment or collecting data directly. In this setting, sequential designs are useful particularly when the objective is to estimate pre-specified process features such as global minimum and maximum, local optima, change points, contours, percentiles, confidence intervals, and overall surface fit. Depending on the goal of the experiment different algorithms for obtaining optimal designs can be derived. In the next section, we briefly review a general scheme of constructing sequential designs for computer experiments.

4.2.1 Sequential Design Scheme

Recall that $y^S = \eta(\mathbf{x})$ represents the scalar output of the deterministic computer model $\eta(.)$ at input $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$, where $\mathcal{X}$ is the d-dimensional unit cube. Assume the computer model is expensive to evaluate, and there is a fixed budget, W, of trials to be performed. In general, the sequential design of computer experiments proceeds as follows:

1. Choose initial design $\mathbf{X}_{t_0} = \left[\mathbf{x}_1, \ldots, \mathbf{x}_{t_0}\right]^T$ where $t_0 < W$.
2. Evaluate the computer model $\eta(.)$ at $\mathbf{X}_{t_0}$ and obtain $\mathbf{y}^S_{t_0} = (y^S_1, \ldots, y^S_{t_0})^T$.
3. Obtain data $\mathbf{D}_{t_0} = (\mathbf{X}_{t_0}, \mathbf{y}^S_{t_0})$ and set $t = t_0$ (indexing iterations of sequential design)
4. Fit a statistical surrogate model using D_t.
5. Choose a new trial location $\mathbf{x}_{\text{new}}$ from $\mathcal{X}$.
6. Update $\mathbf{D}_{t+1} = \mathbf{D}_t \cup (\mathbf{x}_{\text{new}}, \mathbf{y}^S_{\text{new}})$ where $\mathbf{y}^S_{\text{new}} = \eta(\mathbf{x}_{\text{new}})$, and set $t = t + 1$.
7. Repeat this procedure from step 4 if $t < W$ or a stopping criterion is achieved.

8. Return $\mathbf{D}_t$ if $t = W$ along with surrogate fit.

In step 1 it is important to know how we choose the initial design $\mathbf{X}_{t_0}$ and what the right choice of t_0 is. For example if the objective is understanding the overall surface, then the popular choices are space-filling designs (maximin, uniform, D-optimal designs, etc.). Also the choice of initial design depends on the complexity of the computer model and it should not be too small or too big. In step 4, the choice of surrogate model is important. The sequential design scheme is not restricted to only stationary GP models (e.g. Gramacy and Lee [2009], Rajaram et al. [2020], Sauer et al. [2020], Volodina and Williamson [2020]). Particularly when the response of a computer model varies significantly across the input space, appropriate choices for surrogate models are non-stationary processes that can concentrate exploration in regions of the input space with more complicated response.

In step 5 of the sequential design scheme, the way of selecting new trial locations is important. One can choose to select new design $\mathbf{x}_{\text{new}}$ from the input space $\mathcal{X}$ randomly, which is not very efficient. Most sequential design strategies obtain new design points based on a specific criterion. The expected Improvement (EI) is a popular criterion that has been adopted in many sequential design strategies. Depending on the goal of the experiment (e.g. overall surface fit, optimization, estimating contours, etc.), different criteria can be derived using the concept of EI. In this stage, choosing design points sequentially can be done one at a time (Ranjan et al. [2008]) or in a batch of a pre-specified number of trials. Often a batch sequential design is preferred due to the associated cost constraints and experimental settings (Loeppky et al. [2010]) and most of EI criteria can be modified to choose a batch of a pre-specified number of trials in $\mathcal{X}$. In the next section we briefly review definitions of improvement functions and EI criterion in general.

4.2.2 Expected Improvement Criterion

Improvement functions, denoted by $I(\mathbf{x})$, are defined for any $\mathbf{x}$ in the input space $\mathcal{X}$ and are helpful to identify optimal computer trails. Depending on the scientific objective, different forms of the improvement can be defined. In general, the improvement function is formulated to efficiently estimate a pre-specified feature of the computer model output (e.g. global minimum and maximum, contours, percentiles).

The expected improvement criterion, EI, is given by the expectation of a given improvement function $I(\mathbf{x})$ over the predictive distribution $y^S(\mathbf{x})$ conditional on all runs $\mathbf{D}_t = (\mathbf{X}_t, \mathbf{y}_t^S)$ so far, as follows

$$\mathrm{E}\{I(\mathbf{x})\} = \int I(\mathbf{x}) \; \mathbb{P}(y^S(\mathbf{x}) \mid \mathbf{D}_t) \; d\mathbf{x}. \tag{4.1}$$

The EI based criteria are specifically very efficient, as the expectation over the prediction distribution facilitates a balance between global (exploration) vs local (exploitation) search. $\mathrm{E}\{I(\mathbf{x})\}$ is evaluated over the entire input space $\mathcal{X}$ using the information given by the fitted surrogate model. The choice of new trial location $\mathbf{x}_{\text{new}}$ in step 5 of the sequential design scheme is the maximizer (or minimizer) of the expected improvement criterion $\mathrm{E}\{I(\mathbf{x})\}$ given by

$$\mathbf{x}_{\text{new}} = \underset{\mathbf{x}\in\mathcal{X}}{\arg\max}\ \mathrm{E}\{I(\mathbf{x})\}. \tag{4.2}$$

The new optimal design $\mathbf{x}_{\text{new}}$ is added to the current design of experiment in order to improve the current estimate of the feature of interest. To solve the criterion (4.2), optimization techniques can be employed. For example, $\mathrm{E}\{I(\mathbf{x})\}$ can be evaluated on a candidate set of inputs in $\mathcal{X}$ in a discrete or randomized search. In Bingham et al. [2014], a review of EI criteria has been provided.

4.3 Sequential Design for Complex Computer Models

In complex computer models, a practical problem of interest is making a design for the experiment to learn about where the variability of the response is highest or where uncertainty is largest, and spend relatively more effort sampling in these areas. In this work, we are interested in improving predictions made using our DGP emulator for outputs of complex computer models throughout the entire input space $\mathcal{X}$. Our proposed design strategy puts together the non-stationary DGP model and an integrated sequential design strategy, resulting in a more efficient emulator and exploration of the input space. To proceed, an adaptation is needed in our prediction method with the DGP emulator which is described in the next section.

4.3.1 Localized Prediction using DGP

Recall that $\mathbf{y}^S = (y_1^S, \ldots, y_{n_S}^S)^T$ be the computer model runs observed at $n_S \times d$ design matrix $\mathbf{X} = \big[\mathbf{x}_1, \ldots, \mathbf{x}_{n_s}\big]^T$. Let $\mathbf{D} = (\mathbf{X}, \mathbf{y}^S)$ represent the full simulation data. In Chapter 3, we described our DGP emulator with N hidden layers fitted on the data using a modified version of DSVI (Subsection 3.4.2). We approximated the predictive distribution of $y^S(.)$ at the new input $\mathbf{x}^*$ conditioned on the estimated parameter $\hat{\alpha}_r \sim \mathcal{N}(\hat{m}_\alpha, \hat{s}_\alpha)$ and estimated variational parameters $\hat{\mathbf{Z}}_n$, $\hat{\mathbf{m}}_n$, $\hat{\mathbf{s}}_n$, $\hat{\boldsymbol{\delta}}_{\mathbf{Z}_n}$ by drawing sampled variables $(\hat{u}_1^*)_r \sim q(\mathbf{u}_1|\hat{\mathbf{m}}_1, \hat{\mathbf{s}}_1, \mathbf{x}^*, \hat{\mathbf{Z}}_1)$ and $(\hat{u}_n^*)_r \sim q(\mathbf{u}_n|\hat{\mathbf{m}}_n, \hat{\mathbf{s}}_n, (\hat{u}_{n-1}^*)_r, \mathbf{x}^*, \hat{\mathbf{Z}}_n, \hat{\alpha}_r)$ recursively as (3.43) and (3.44), where $\hat{u}_n^* = \hat{u}_n(\mathbf{x}^*)$ for $n = 1, \ldots, N$ and $r = 1, \ldots, R$ represents index of sampling iterations. Note that $\hat{\mathbf{Z}}_n$ in predictive equations (3.45) play an analogous role of the design $\mathbf{X}$ in usual kriging formula (2.2). On the other hand, our goal is to improve prediction performance of our DGP emulator through a variance based sequential design criterion. One way to do this to treat $\hat{\mathbf{Z}}_N$ (inducing locations at the last layer) as experimental design. As $\hat{\mathbf{Z}}_N$ is optimized for the existing training set without the presence of the new candidate design point, this is required to refit the model each time. Although inducing locations when $m \ll n_S$ scale up the computations in predictive equations (3.45), searching for a new candidate design by considering its impact on the predictive variance is computationally intensive in this regime. As a result, we adapt our prediction approach to tackle the sequential design problem for complex computer models using our DGP emulator with an efficient computational cost.

Conditioning on the data, $\mathbf{D}$, posterior mean estimate $\hat{\alpha}$, $\hat{u}_N^* = \hat{u}_N(\mathbf{x}^*) \in \mathbb{R}$, and $\hat{\mathbf{u}}_N = \hat{u}_N(\mathbf{X}) \in \mathbb{R}^{n_S}$, the posterior predictive distribution of $y^S(.)$ at $\mathbf{x}^*$ is conditionally Gaussian with mean and variance,

$$\begin{aligned}&k(\mathbf{X}, \mathbf{x}^*; \hat{\mathbf{u}}_N, \hat{u}_N^*, \hat{\alpha})^T \ \mathbf{K}^{-1} \ \mathbf{y}^S, \\ &k(\mathbf{X}, \mathbf{x}^*; \hat{\mathbf{u}}_N, \hat{u}_N^*, \hat{\alpha})^T \ \mathbf{K}^{-1} \ k(\mathbf{X}, \mathbf{x}^*; \hat{\mathbf{u}}_N, \hat{u}_N^*, \hat{\alpha}),\end{aligned} \tag{4.3}$$

respectively, where $k(.,.)$ is a non-stationary covariance function defined in (3.13). $\mathbf{K} = k(\mathbf{X}, \mathbf{X}, \hat{\mathbf{u}}_N, \hat{\alpha})$ is the covariance matrix for the simulations, with $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j; \hat{\mathbf{u}}_N^i, \hat{\mathbf{u}}_N^j, \hat{\alpha})$, where $\hat{\mathbf{u}}_N^i := (\hat{\mathbf{u}}_N)_i = u_N(\mathbf{x}_i)$ for $i, j = 1, \ldots, n_S$. $k(\mathbf{X}, \mathbf{x}^*, \hat{\mathbf{u}}_N, \hat{u}_N^*, \hat{\alpha})$ is the $n_S \times 1$ vector of covariances between a response at $\mathbf{x}^*$ and those at the inputs in the design, $\mathbf{X}$. Both $\hat{\mathbf{u}}_N$ and $\hat{u}_N^*$ are estimated through the sampling procedure in (3.43) and (3.44) and used to compute length scales at the design $\mathbf{X}$ and the new input $\mathbf{x}^*$, respectively. Finding an optimal design with this solution and our DGP emulator can be computationally intensive, especially when the size of the data, n_S, is large, because the sequential design algorithm involves repeated inversion of the large covariance matrix $\mathbf{K}$.

To address this, we propose to use local points in the neighborhood of the unsampled input, $\mathbf{x}^*$, to alleviate the computational burden. This idea comes from local kriging in the spatial statistics literature (Cressie [1993], pp. 131–134) and local GPs in computer experiments literature (Gramacy and Apley [2015]). Here, we propose to use nearest neighbors (NN) to $\mathbf{x}^*$ for making prediction using the proposed DGP emulator. First, we find the n_b NNs to the new input $\mathbf{x}^*$ based on the Euclidean distance and create the sub-design $\mathbf{X}^b \subseteq \mathbf{X}$ with their inputs, and $\mathbf{y}^b$ for their outputs. Then we apply the predictive mean and variance equations (4.3) with local data set $D^b(\mathbf{x}^*) = (\mathbf{X}^b, \mathbf{y}^b)$ as follows

$$\begin{aligned}&k(\mathbf{X}^b, \mathbf{x}^*; \hat{\mathbf{u}}_N^b, \hat{u}_N^*, \hat{\alpha})^T \ (\mathbf{K}^b)^{-1} \ \mathbf{y}^b, \\ &k(\mathbf{X}^b, \mathbf{x}^*; \hat{\mathbf{u}}_N^b, \hat{u}_N^*, \hat{\alpha})^T \ (\mathbf{K}^b)^{-1} \ k(\mathbf{X}^b, \mathbf{x}^*; \hat{\mathbf{u}}_N^b, \hat{u}_N^*, \hat{\alpha}),\end{aligned} \tag{4.4}$$

respectively, where $\hat{\mathbf{u}}_N^b = \hat{u}_N(\mathbf{X}^b) \in \mathbb{R}^{n_b}$. Here, $\mathbf{K}^b$ is a $n_b \times n_b$ covariance matrix for n_b NNs. The number of NNs, n_b, is chosen as large as computational restrictions allow. For $n_b \ll n_S$ the computation for emulating $y^S(\mathbf{x}^*)$ can be reduced from $O({n_S}^3)$ to $O({n_b}^3)$. It is clear that as $n_b \to n_S$, then predictive equations (4.4) converges to equations (4.3). In other words, the larger NN design is, the more information is provided for emulating at $\mathbf{x}^*$. Hence, there exist trade-offs between computational efficiency and prediction accuracy with an appropriate choice of n_b. The next section details how this adjusted prediction approach facilitates planning future computer simulations through an EI based criterion for the sequential design.

4.3.2 Localized Design Criterion

In this work, we are interested in improving predictions made using our DGP emulator for outputs of complex computer models throughout the entire input space $\mathcal{X}$. Our aim is to adopt an EI-based criterion guiding the selection of future runs to improve not only predictive accuracy of our DGP emulator but also the exploration of the input space.

The predictive variance is a measure of uncertainty about the model behaviour and is widely used as part of the design criterion. An intuitive sequential design procedure is to minimize predictive variance by assigning new simulations where the variability is the largest. From step 1 of the sequential design scheme, recall that $\mathbf{D}_t = (\mathbf{X}_t, \mathbf{y}_t^S)$ denote an initial data set of size $t = t_0$. After fitting a DGP with N hidden layers to $\mathbf{D}_t$, the posterior predictive variance (or MSPE) at new location $\mathbf{x}^*$ denoted by $\sigma_t^2(\mathbf{x}^*)$ is calculated from initial design $\mathbf{X}_t$ via equation (4.3). Let $\widetilde{\mathbf{x}}$ be a candidate design point added to the initial design $\mathbf{X}_t$. To determine whether or not $\widetilde{\mathbf{x}}$ should be selected for simulation next, its impact on the posterior predictive variance of $y^S(.)$ is assessed via the Integrated Mean Squared Prediction Error (IMSPE) criterion. The IMSPE is the predictive variance averaged over the input space, and is given by

$$\text{IMSPE}(\widetilde{\mathbf{x}}) = \int_{\mathbf{x}\in\mathcal{X}} \sigma_{t+1}^2(\mathbf{x})\ d\mathbf{x}, \tag{4.5}$$

where $\sigma_{t+1}^2(\mathbf{x})$ denote the deduced posterior predictive variance at location $\mathbf{x} \in \mathcal{X}$ calculated from $\mathbf{X}_{t+1} = \mathbf{X}_t \cup \{\widetilde{\mathbf{x}}\}$ via equation (4.3). The integration in (4.5) is with respect to the uniform distribution over the input space $\mathcal{X}$, hence the IMSPE can be viewed as an EI based criterion. At any candidate design point, $\widetilde{\mathbf{x}} \in \mathcal{X}$, the IMSPE$(\widetilde{\mathbf{x}})$ can be approximated over a prediction set denoted by $\mathbf{X}^*$ in the input space $\mathcal{X}$ with size H as follows

$$\text{IMSPE}(\widetilde{\mathbf{x}}) \approx \frac{1}{H}\sum_{i=1}^{H} \sigma_{t+1}^2(\mathbf{x}_i^*), \tag{4.6}$$

where $\mathbf{x}_i^* \in \mathbf{X}^*$. The choice of new trial location $\mathbf{x}_{\text{new}}$ in step 5 of the sequential design scheme is the minimizer of the IMSPE given by

$$\mathbf{x}_{\text{new}} = \underset{\mathbf{x}\in\mathcal{X}}{\arg\min}\ \ \text{IMSPE}(\mathbf{x}). \tag{4.7}$$

In this work, we solve this acquisition through a randomized search over a candidate set $\widetilde{\mathbf{X}}$ in $\mathcal{X}$. When the data size is large, the procedure of finding even one new design point is computationally expensive as it is required to calculate the posterior predictive variance at all prediction points via equation (4.3) to obtain the IMSPE (4.6) at each candidate point.

Although solving equation (4.7) can be parallelized at each candidate point, we address this issue using the proposed localized predictive variance equation in (4.4), which is our emphasis in this work.

To utilize our localized prediction idea in the sequential design strategy, first we find n_b NNs to each prediction point $\mathbf{x}_i^* \in \mathbf{X}^*$ and construct sub-designs $\mathbf{X}_{t,i}^b \subseteq \mathbf{X}_t$ for $i = 1, \ldots, H$. Then posterior predictive variance at each prediction point $\mathbf{x}_i^*$ is calculated from the initial NN design $\mathbf{X}_{t,i}^b$ via equation (4.4) as

$$\check{\sigma}_t^2(\mathbf{x}_i^*) = k(\mathbf{X}_{t,i}^b, \mathbf{x}_i^*; \hat{\mathbf{u}}_{N,t,i}^b, \hat{u}_N^*, \hat{\alpha})^T \ (\mathbf{K}_{t,i}^b)^{-1} \ k(\mathbf{X}_{t,i}^b, \mathbf{x}_i^*; \hat{\mathbf{u}}_{N,t,i}^b, \hat{u}_N^*, \hat{\alpha}), \tag{4.8}$$

where $\hat{\mathbf{u}}_{N,t,i}^b = \hat{u}_N(\mathbf{X}_{t,i}^b) \in \mathbb{R}^{n_b}$ and $\mathbf{K}_{t,i}^b$ is a $n_b \times n_b$ covariance matrix for NN design $\mathbf{X}_{t,i}^b$. Let $\mathbf{v}_t = (\check{\sigma}_t^2(\mathbf{x}_1^*), \ldots, \check{\sigma}_t^2(\mathbf{x}_H^*))^T$ be a $H \times 1$ vector of all calculated posterior predictive variances from NN designs. With localized predictive variances, the IMSPE criterion at candidate design point $\widetilde{\mathbf{x}}$ is defined and approximated as

$$\text{IM}\check{\text{S}}\text{PE}(\widetilde{\mathbf{x}}) = \int_{\mathbf{x} \in \mathcal{X}} \check{\sigma}_{t+1}^2(\mathbf{x}) \ d\mathbf{x} \approx \frac{1}{H} \sum_{i=1}^{H} \check{\sigma}_{t+1}^2(\mathbf{x}_i^*), \tag{4.9}$$

where $\check{\sigma}_{t+1}^2(\mathbf{x}_i^*)$ denote the deduced posterior predictive variance at location $\mathbf{x}_i^* \in \mathbf{X}^*$ calculated from NN design $\mathbf{X}_{t+1,i}^b \subseteq \mathbf{X}_{t+1} = \mathbf{X}_t \cup \{\widetilde{\mathbf{x}}\}$ with size n_b as

$$\check{\sigma}_{t+1}^2(\mathbf{x}_i^*) = k(\mathbf{X}_{t+1,i}^b, \mathbf{x}_i^*; \hat{\mathbf{u}}_{N,t+1,i}^b, \hat{u}_N^*, \hat{\alpha})^T \ (\mathbf{K}_{t+1,i}^b)^{-1} \ k(\mathbf{X}_{t+1,i}^b, \mathbf{x}_i^*; \hat{\mathbf{u}}_{N,t+1,i}^b, \hat{u}_N^*, \hat{\alpha}), \tag{4.10}$$

where $\hat{\mathbf{u}}_{N,t+1,i}^b = \hat{u}_N(\mathbf{X}_{t+1,i}^b) \in \mathbb{R}^{n_b}$ and $\mathbf{K}_{t+1,i}^b$ is a $n_b \times n_b$ covariance matrix for NN design $\mathbf{X}_{t+1,i}^b$. Since the candidate design point $\widetilde{\mathbf{x}}$ can not be in all of the NN designs $\mathbf{X}_{t+1,i}^b$ for $i = 1, \ldots, H$, changes in the IMSPE through adding the candidate design point to the initial design are due to changes in posterior predictive variances calculated from NN sub-designs include the candidate design. This allows us to reduce unnecessary calculations such that if $\widetilde{\mathbf{x}} \in \mathbf{X}_{t+1,i}^b$ then $\check{\sigma}_{t+1}^2(\mathbf{x}_i^*)$ is calculated, otherwise its value is replaced with already calculated value of $(\mathbf{v}_t)_i = \check{\sigma}_t^2(\mathbf{x}_i^*)$. It is clear that a larger choice of n_b makes more number of calculations, as more NN designs include the candidate design $\widetilde{\mathbf{x}}$. The new design $\mathbf{x}_{\text{new}}$ is obtained via solving acquisition

$$\mathbf{x}_{\text{new}} = \underset{\mathbf{x} \in \mathcal{X}}{\arg\min} \ \text{IM}\check{\text{S}}\text{PE}(\mathbf{x}), \tag{4.11}$$

through a randomized search over the candidate set $\widetilde{\mathbf{X}}$ in $\mathcal{X}$. After obtaining the first new design, we update the data as $\mathbf{D}_{t+1} = \mathbf{D}_t \cup (\mathbf{x}_{\text{new}}, \mathbf{y}_{\text{new}}^S)$ where $\mathbf{y}_{\text{new}}^S = \eta(\mathbf{x}_{\text{new}})$. It is a computational choice that one can decide not to back to the step 4 of the sequential design scheme to refit the model and instead backs to the step of solving the acquisition (4.11) to find the next new design. This procedure is repeated if $t < W$ where W is the final size of the design. Refitting model in batches of added design points is another choice. In particular, after sequentially adding a specified number of new design points, the DGP model is refitted to the updated data. In this work, we also investigate how this choice can impact the prediction performance of the DGP and the ability of our design criterion in exploring the input space. A comparison of resulting sequential designs with and without refitting model in the 2-d toy model is presented in the next section.

4.4 Illustration

In this section, the localized sequential design strategy proposed in the previous section is validated on the 2-d toy model and the real-world application, the COMPAS model. These two examples were used for illustrating the proposed methodology for the DGP emulation in Subsections 3.5.2 and 3.5.3, respectively. A comparison of two design strategies with refitting the model in batches of added new design points and without refitting the model is conducted through evaluating Nash–Sutcliffe Efficiency (NSE) and IMSPE values in the 2-d example. In the COMPAS model, two sequential designs are constructed with refitting model using two different number of NNs to illustrate our localized design strategy.

4.4.1 2-d Toy Model

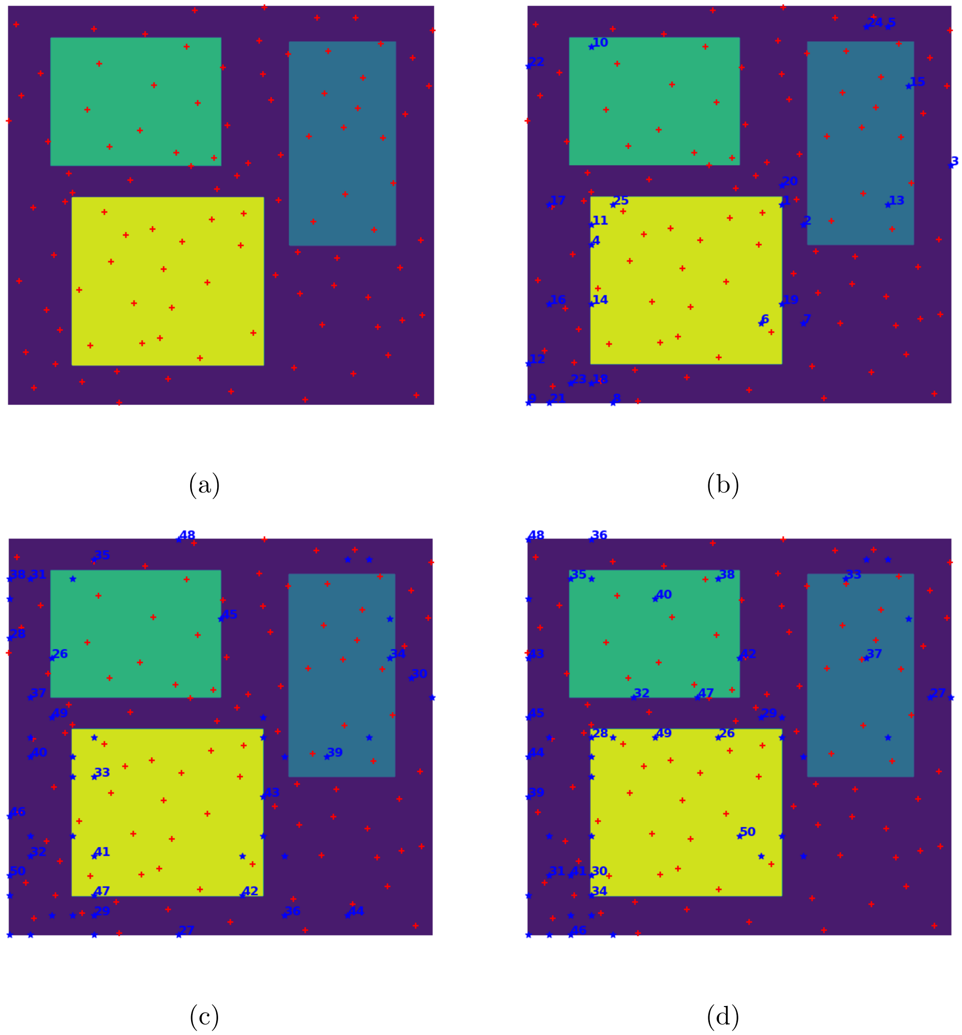
We use the following model $g(x_1, x_2)$ defined in

$$g(x_1,x_2) = \begin{cases} 1.3 & x_1 \in [0.66, 0.91] \text{ and } x_2 \in [0.4, 0.91] \\ 2.2 & x_1 \in [0.1, 0.5] \text{ and } x_2 \in [0.6, 0.92] \\ 3.5 & x_1 \in [0.15, 0.6] \text{ and } x_2 \in [0.1, 0.52] \\ 0 & \text{o.w.} \end{cases}, \quad x_1, x_2 \in [0,1] \tag{4.12}$$

to illustrate how the proposed design criterion improves the DGP emulator by adding additional trials to the design. We generate a Maximin Latin hypercube design (LHD) of size 100 as an initial design for the computer model (panel (a) in Figure 4.1) to conduct an initial set of simulations. Then a DGP with two hidden layers is fitted on this initial design and corresponding outputs using $m = 80$ inducing points in each layer. The DGP is constructed using the Matern covariance functions formulated as (3.17) and (3.18) with $\nu = 2.5$, respectively. For estimating α, the normal distribution $\mathcal{N}(3.5, 1)$ is chosen for $\mathbb{P}(\alpha)$ and $q(\alpha)$ is initialized with $\mathcal{N}(m_{\alpha\text{ini}}, s_{\alpha\text{ini}})$ where $m_{\alpha\text{ini}} = 3$ and $s_{\alpha\text{ini}} = 1$.

Our goal is to add 50 new simulation runs sequentially to the initial design through the localized design criterion in two cases (i) without refitting the model (ii) with refitting model in batch of added new design points. In both cases, a 21 by 21 grid on $\mathcal{X}$ is used as a candidate set. We evaluate the design criterion IMŠPE defined in (4.9) at each candidate point $\tilde{\mathbf{x}}_j$ for $j = 1, \ldots, 441$ over a 10 by 10 grid on $\mathcal{X}$ as a prediction set using 80 NNs, and find the best candidate by solving acquisition (4.11) to perform the next simulation. The search for new design points continues in two cases (i) and (ii) until a total run size of 150 is obtained.

Panel (b) of Figure 4.1 shows the design updated by adding the first 25 new design points (in blue $*$) labeled according to the order in which they are added. Panel (c) of Figure 4.1 shows the updated design by adding the second 25 new design points without refitting the model, where previously added points (in blue $*$) do not have ordered labels. To see the impact of refitting model, the model is refitted using the updated design in panel (b) and then the second 25 points are added sequentially by the same procedure (panel (d) in figure 4.1).



(a)

(b)

(c)

(d)

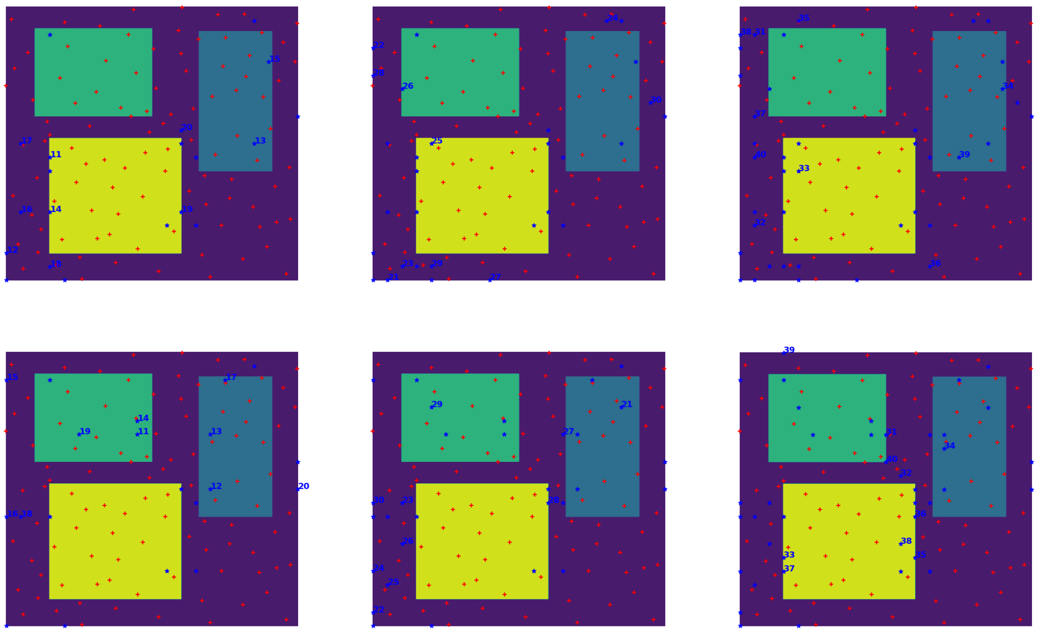
Figure 4.1: Sequential design construction. (a) Initial design (b) 25 points added (c) 50 points added (d) 50 points added after refitting the model. Red dots represent the initial design, blue $*$'s labeled with numbers represents new design points according to the order in which they are added, blue $*$'s without ordered labels represent previously added points.

As seen in panels (b), (c) and (d), in both cases, (i) and (ii), our sequential design criterion is able to effectively place design points near the boundary of the three regions where there is high variability and uncertainty about the computer model output. The criterion is also able to jump to sparsely sampled regions (e.g., points 3, 6 and 10 in panel (b)) and within the regions (e.g. points 50, 40 and 37 in panel (d)) for choosing design points. The combination of the design criterion with our non-stationary DGP emulator enable our design strategy to distinguish these distinct regimes and favor acquisitions in an explore-and-exploit manner. Comparing the updated designs in panels (c) and (d) shows how refitting the model can improve the ability of the model and the design criterion to place design points in the regions with highest variability.

Particularly, panel (d) shows that after refitting the model, the criterion is able to effectively recognize that more design points are needed to be placed in the bottom of the green region and the top of the yellow region (e.g. points 32, 47 and 49, 26 in panel (d)). The updated design shown in panel (c) is constructed from the DGP model fitted on the initial design shown in panel (a) with size 100, whereas the design shown in panel (d) is constructed from the DGP model refitted on the design shown in panel (b) with size 125 after adding the first 25 design points. We investigated the predictive variances at prediction points located in the bottom of the green region, the top of the yellow region and the area of between them after fitting designs in panel (a) and (b). We realized that the predictive variances at these points became higher after refitting the model on design (b). That's one reason why the criterion places points 32, 47 and 49, 26 in these parts in panel (d). Another reason that we realized is that refitting after adding 25 points changed the correlation structure of the model. To reach this explanation, we compared the correlations between pairs of points, where one is located in the the bottom of the green region and across of that, another one is located in the top of the yellow region. We observed that the correlation between these two points went down after refitting the model on design (b). In other words, after refitting the model, the criterion was able to distinguish the yellow region from the green region.

We also explore the choice of the batch size for adding new design points in case of refitting the model. It is clear that if the batch size is small, it is needed to refit the model multiple times and as a result the computational time of constructing the sequential design goes up. For this example, we choose to refit the model after adding every 10 points to see how much it improves the ability of the design criterion in exploring the input space. After adding the first 10 points shown in panel (b) of Figure 4.1, the model is refitted on the updated design and then the second 10 points are added. This procedure is repeated until a total run size of 150 is obtained. Bottom row panels from (a) to (c) in Figure 4.2 show the updated design with refitting the model after adding every 10 points, with the final design plotted in the panel (b) of Figure 4.3 . For comparison purposes, the final design constructed without refitting the model plotted in panel (c) of Figure 4.1 is shown for every 10 iterations in the top row panels from (a) to (c) in Figure 4.2, with the final design plotted in the panel

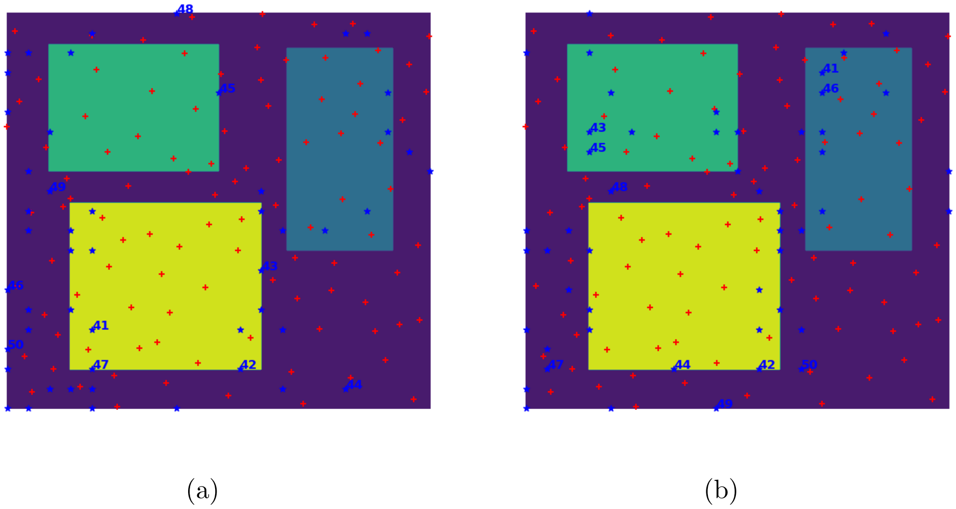
(a) of Figure 4.3. The first 10 added points in both final designs shown in panels (a) and (b) of Figure 4.3 are the same. Comparing final designs in panel (a) and (b) of Figure 4.3 shows refitting the model every 10 points makes a slight improvement in the ability of the design criterion in exploring the input space. Similar to the updated designs without refitting the model, new design points are effectively placed where the variability is higher, i.e. near the boundary of the three regions and sparsely sampled regions in bottom row updated designs in Figure 4.2 and the final design in panel (b) of Figure 4.3.



(a) (b) (c)

Figure 4.2: Sequential design construction every 10 iterations. Top row: without refitting the model, Bottom row: with refitting the model after adding every 10 points (a) The second 10 points added (b) The third 10 points added (c) The fourth 10 points added. Red dots represent the initial design, blue ∗'s labeled with numbers represents new design points according to the order in which they are added, blue ∗'s without ordered labels represent previously added points.

IMSPEs are plotted and compared in three different sequential design constructions: (i) without refitting the model (ii) with refitting model after adding every 10 points and (iii) with refitting model after adding every 25 points in panels (a), (b) and (c) of Figure 4.4, respectively. A significant effect on IMSPE reduction is demonstrated in three panels by adding the first 10 new design points which are the same in all three designs. As it is expected, IMSPE jumps up exactly after refitting the model in every 10 and 25 iterations in panels (b) and (c), respectively, although IMSPE come down afterwards. In panel (b), IMSPE reductions after adding the second to the fourth 10 points are not as significant as



(a)

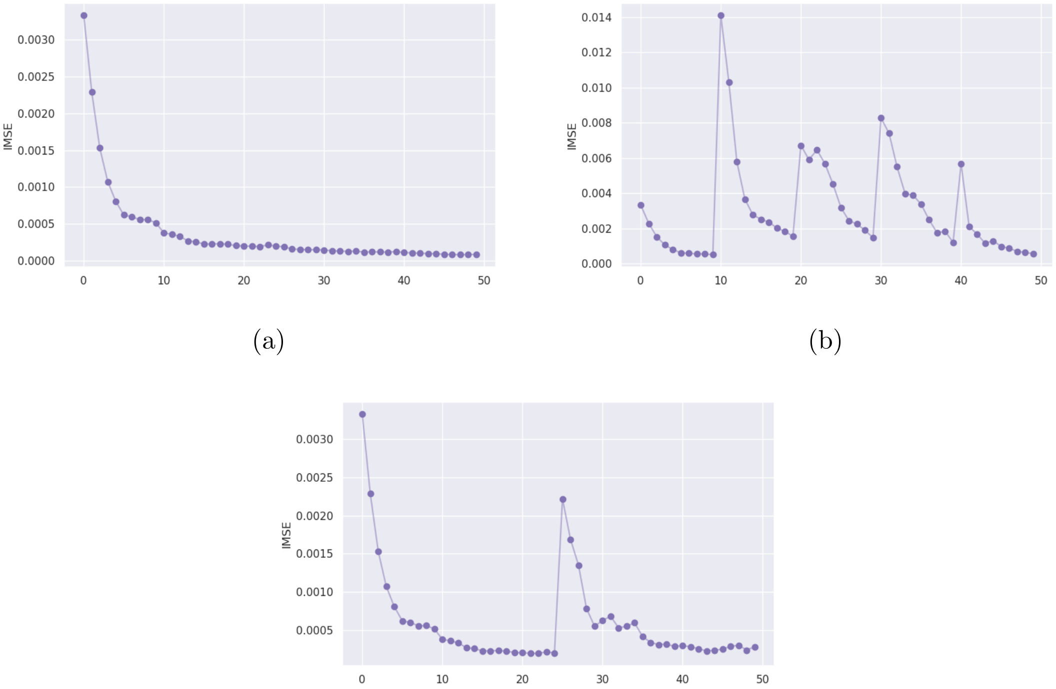
(b)

Figure 4.3: Sequential design construction every 10 iterations. Final designs after adding the fifth 10 points (a) without refitting the model (b) with refitting the model after adding every 10 points. Red dots represent the initial design, blue $*$'s labeled with numbers represents new design points according to the order in which they are added, blue $*$'s without ordered labels represent previously added points.

IMSPE reductions gained by adding the last 25 points in panel (c). Hence, an appropriate choice of the batch size should be examined carefully to gain more improvement in the IMSPE with reducing computational costs.

Three different sets of simulation data from final sequential designs constructed through our localized sequential design criterion are used to emulate the computer model $g(.,.)$ on a 20×20 grid in $\mathcal{X}$. To do this, first a DGP with two hidden layers is fit on these three simulation data sets using 80 inducing points and the same initializations. For making predictions, our localized prediction method is applied with 80 NNs. The DGP is also fit on simulation data from a maximin LHD with the same size as sequential designs and then is used to make predictions at the same prediction set (20×20 grid in $\mathcal{X}$).

The prediction accuracy of the DGP model (NSE) in all the cases are computed and displayed in Table 4.1. As seen in the table, all the sequential designs are able to significantly improve prediction performance of the DGP emulator compared with the LHD. The sequential designs constructed from refitting the model after every 10 and 25 points have the larger NSE than the sequential designs constructed without refitting the model. Depending on the cost restrictions and availability of computational resources, refitting the model may be worth the effort with an appropriate choice of the batch size for adding new design points. Also using NN designs in making predictions and the localized sequential design criterion is also a computational choice and as a result there are trade-offs between the computational efficiency and prediction accuracy with an appropriate choice of size of the NN designs.



(a)

(b)

(c)

Figure 4.4: IMSPEs of three different sequential design constructions: (a) without refitting the model (b) with refitting the model after adding every 10 points (c) with refitting the model after adding 25 points. In all three panels, IMSPES are the same for the first 10 points.

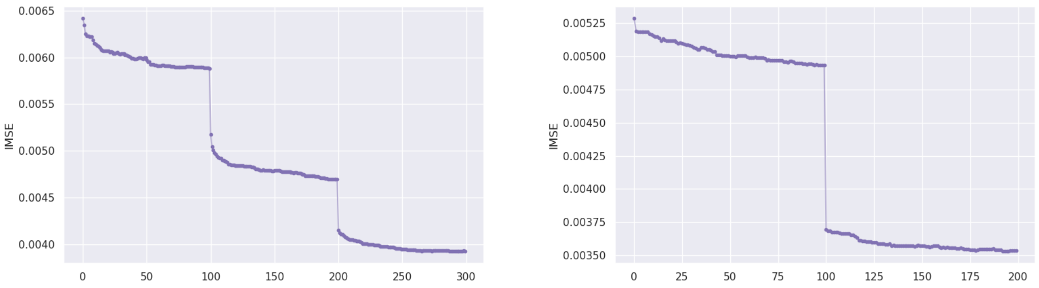
Table 4.1: Prediction performance of the DGP using four different designs

	LHD	SeqD (without refit)	SeqD (refit after 10)	SeqD (refit after 25)
NSE	72.48%	80.03%	80.80%	82.07%

4.4.2 COMPAS Model

We now return to the COMPAS model and apply the proposed sequential design approach to the simulations runs used in 3.5.3. Our aim is to demonstrate how additional trials may be added to improve the model based on the design criterion introduced in 4.3.2. We use the two million simulation runs with input variables standardized to the 11-dimensional unit cube $[0,1]^{11}$. More specifically, we choose the same prediction set with size of 1000 (including 450 active points) which is held out from the data. To construct an initial set of simulations, $100,000$ simulations are randomly selected from the rest of the data with 40% success rate for BBH formation. The remaining data is considered as the candidate set.

A DGP with three hidden layers is fit using $m = 100$ inducing points in each layer. The DGP is constructed using the stationary and non-stationary Matern covariance functions formulated as (3.17) and (3.18) with $\nu = 2.5$, respectively. For training, we approximated the ELBO (3.41) with a batch size of 1000 to achieve scalability. Two sequential designs are constructed through the localized design criterion: (i) 300 new simulation runs are added sequentially to the initial design using 300 NNs and (ii) 200 new simulation runs are added sequentially to the initial design using 500 NNs, with refitting model after adding every 100 new design points. In both cases, we evaluate the localized design criterion defined in (4.9) at each candidate point over the prediction set, and identify the best design point from the candidate set using a randomized search with size of 500 (including 200 active points) over the candidate set to perform the next simulation.



(a)

(b)

Figure 4.5: IMSPEs of two different sequential design constructions with refitting model after adding every 100 points (a) using 300 NNs and (b) using 500 NNs

Panel (a) and (b) of Figure 4.5 shows a comparison of IMSPE of two sequential designs constructed through (i) and (ii), respectively. In both panels, IMSPEs are improved (jump down) efficiently by refitting the model after adding every 100 points. Also most of the new design points were selected from the active points, illustrating that our localized sequential design criterion successfully diagnosed more complex regions of the chirp mass input space. Comparing the maximum and minimum of IMSPEs displayed in these two plots, shows that using more NNs in evaluating the design criterion in case (ii) can greatly improve the performance of the DGP emulator even with adding 200 new design points compared with the other case which adding 300 new design points using 300 NNs.

The resulting sequential designs in cases (i) and (ii) were also compared with the initial design in the prediction accuracy of the DGP model (NSE). After fitting a DGP with three hidden layers on these three simulation data sets, we emulated the COMPAS model at 1000 points in the prediction set. The prediction accuracy of the DGP model (NSE) using the initial design and the resulting sequential designs in cases (i) and (ii) were computed and displayed in Table 4.2. As seen in the table, the sequential designs have a larger NSE than the initial design, showing that adding new design points to the initial design sequentially in both cases (i) and (ii) improved the prediction performance of the DGP emulator. Also, NSE of the sequential design constructed using 500 NNs is larger than NSE of the other sequential design using 300 NNs, as more information is provided in evaluating IMSPE through larger sub-designs. Although, the computational time of adding one new design using 500 NNs (293.6 s) is more than the computational time of adding one new design using 300 NNs (169.5 s), more improvement in the prediction performance of the DGP emulator is achieved by adding less number of design points (200 new design points added) to the initial design compared with the other sequential design (300 new design points added). This shows how an appropriate choice of size of the NN designs in our localized design criterion can impact on the computational efficiency and prediction accuracy of the DGP emulator.

Table 4.2: Prediction performance of the DGP using the initial design and two different sequential designs

	Initial Design	SeqD (300 NNs)	SeqD (500 NNs)
NSE	70.01%	78.61%	80.12%

4.5 Summary and Discussion

This chapter presents new methodology for selecting design points to improve the DGP emulator. In this chapter, we proposed a sequential design approach to improve performance of our DGP emulator and exploration of the input space for guiding future simulations. We combined the non-stationary DGP model with an EI based sequential design criterion to deviate from usual space-filling designs and guiding the selection of future runs in more complex regions of the input space.

In order to proceed our method, we had to tackle a conflict between the DGP prediction method introduced in Subsection 3.4.3 and our variance based design criterion. Particularly, the issue was that the posterior predictive variances defined in (3.45) depend on the optimized inducing locations instead of the design $\mathbf{X}$. We addressed this problem by an adaptation in the DGP prediction method, where we incorporate $\mathbf{X}$ in posterior predictive equations as (4.3) conditioning on the estimated last layer used to find the correlation among observations and the new run. We defined our design criterion, IMSPE, at each candidate design point through the new posterior predictive variance formula in (4.3) and approximated it over a prediction set in the input space as (4.6). To find the new design point, we minimized the IMSPE through a randomized search over a candidate set.

When the size of the data is large or there are some restrictions in availability of computational resources, finding an optimal design with this solution and our DGP emulator could be computationally intensive, since the sequential design algorithm would involve repeated inversion of a large covariance matrix with the size of the data in evaluating the predictive variances. Specifically, it would be required to calculate the posterior predictive variance at all prediction points via equation (4.3) to obtain IMSPE (4.6) at each candidate point. In this setting, we proposed to construct a smaller, local design using n_b nearest neighbors (NN) to the unsampled input to make a prediction using our DGP emulator via predictive equations (4.4). We utilized our localized prediction method with the sequential design strategy through defining the design criterion IMŠPE, the localized predictive variance averaged over the input space as (4.9). To find the new design point, we minimized IMŠPE through solving acquisition (4.11) by a randomized search over a candidate set. To evaluate IMŠPE at each candidate design point, we calculated localized posterior predictive variances from only NN sub-designs include the candidate design point. This allowed us to reduce unnecessary calculations for solving (4.11). We also investigated impact of refitting model in batches of added new design points on the prediction performance of the DGP and the ability of our design criterion in exploring the input space.

Using NN designs in making predictions and the localized sequential design criterion is a computational choice. Also the larger NN design is, the more information is provided for emulation at a new input. In fact, there are trade-offs between the computational efficiency and prediction accuracy with an appropriate choice for the size of the NN designs. Hence, to reach the desired accuracy and computational efficiency, it is recommended to perform a preliminary analysis to find a suitable NN design size. Also depending on the computational restrictions, refitting the model may be worth with an appropriate choice of the batch size for adding new design points. This choice of the batch size should also be examined carefully to gain more prediction accuracy along with reducing computational costs.

In future work, we aim to investigate how to make an optimal design using inducing locations to improve our DGP emulator. In other words, we are curious to know if placing new inducing locations where the variability of the response model is highest or where uncertainty is largest could increase the prediction accuracy and what criterion should be defined in this regime. We hope to proceed this idea by utilizing our DGP prediction method introduced in Subsection 3.4.3 without any adaptation such as using NN designs, as inducing locations play an analogous role of the design in posterior predictive equations in (3.45).

Chapter 5

Conclusion

In this thesis, methodologies were developed in computer model emulation and experimental design. In Chapter 3, new methodology for emulation of complex computer models was proposed. A non-stationary DGP emulator was presented that could apply to a large class of complex computer models, and scale to arbitrarily large simulation designs. We introduced a new parameter that allows us to control smoothness of the DGP layers. The impact of the proposed parameter on the level of smoothness of layers was theoretically illustrated and numerically visualised in DGP realizations. We also adapted a stochastic variational inference approach to be suitable for the DGP model in our framework. Our modified inference approach allowed for prior specification and posterior exploration of the smoothness of the response surface and was demonstrated to preserve accuracy with uncertainty measures for arbitrary large designs. The impact of estimating the proposed parameter on the performance of the DGP emulator was illustrated in three synthetic examples. The proposed methodology was applied to the emulation of a complicated astrophysical model efficiently through the data sub-sampling with measuring uncertainties. Additionally, variants of the DGP were proposed which specify our proposed new parameter in different ways. We defer a more thorough investigation of these new variants to future work.

In Chapter 4, a sequential design strategy for complex computer models was proposed. The proposed method aimed to improve performance of the resulting DGP emulator introduced in the previous chapter and exploration of the input space for guiding future simulations. The non-stationary DGP model and an EI based sequential design criterion were combined to deviate from usual space-filling designs and guide the selection of future runs in regions of the input space that are more complicated in the response along with improving prediction accuracy of our DGP emulator. We chose the IMSPE, the predictive variance averaged over the input space, as a sequential design criterion. In order to proceed with our design strategy, it was required to adapt our prediction method. For large simulation designs, we proposed to use nearest neighbour (NN) predictions using our DGP. The design criterion becomes the localized predictive variance averaged over the input space. Our localized sequential design strategy was illustrated in the 2-d toy model as well as the COMPAS model. The combination of our localized sequential design criterion with the non-stationary DGP emulator enabled our design strategy to effectively place design points where there was high variability and uncertainty about the computer model output and favored acquisitions in an explore-and-exploit manner. Also refitting model in batches of added new design points improved the prediction performance of the DGP (NSE and IMSPE) and the ability of our design criterion to encourage exploration of the input space. Using more NNs in evaluating the design criterion greatly improved prediction accuracy in emulation of the COMPAS. In future work, we aim to investigate how to make an optimal design using inducing locations to improve our DGP emulator without using NN designs.

Bibliography

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, L. Kaiser, M. Kudlur, J. Levenberg, D. Man, R. Monga, S. Moore, D. Murray, J. Shlens, B. Steiner, I. Sutskever, P. Tucker, V. Vanhoucke, V. Vasudevan, O. Vinyals, P. Warden, M. Wicke, Y. Yu, and X. Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. 2015. doi: 1603.04467.

I. Andrianakis, I.R. Vernon, N. McCreesh, T.J. McKinley, J.E. Oakley, R.N. Nsubuga, M. Goldstein, and R.G. White. Bayesian history matching of complex infectious disease models using emulation: A tutorial and a case study on hiv in uganda. *PLoS Computational Biology*, 11(1):e1003968, 2015.

B. Ankenman, B.L. Nelson, and J. Staum. Stochastic kriging for simulation meta modeling. *Operations Research*, 58(2):371–382, 2010.

A. Armagan and D. Dunson. Sparse variational analysis of linear mixed models for large data sets. *Statistics and Probability Letters*, 81:1056–1062, 2011.

S. Banerjee, A.E. Gelfand, J.R. Knight, and C.F. Sirmans. Spatial modeling of house prices using normalized distance-weighted sums of stationary processes. *Journal of Business and Economic Statistics*, 22(2):206–213, 2004.

D. Barber and S. Chiappa. Unified inference for variational bayesian linear gaussian state-space models. *in Neural Information Processing Systems*, page 81–88, 2006.

J.W. Barrett, S.M. Gaebel, C.J. Neijssel, A. VignaGomez, S. Stevenson, C.P.L. Berry, W.M. Farr, and I. Mandel. Accuracy of inference on the physics of binary evolution from gravitational-wave observations. *MNRAS*, 477(4):4685–4695, 2018.

K. Belczynski, V. Kalogera, and T. Bulik. A comprehensive study of binary compact objects as gravitational wave sources: Evolutionary channels, rates, and physical properties. *The Astrophysical Journal*, 572:407–431, 2002.

D. Bingham, P. Ranjan, and W.J. Welch. Design of computer experiments for optimization, estimation of function contours, and related objectives. *Statistics in Action: A Canadian Outlook 109*, 109, 2014.

D.M. Blei, A. Kucukelbir, and J.D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112:859–877, 2017.

L. Bornn, G. Shaddick, and J. Zidek. Modelling non-stationary processes through dimension expansion. *Journal of the American Statistical Association*, 497(107):281–289, 2012.

E. Brochu, V.M. Cora, and N. Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Computer Science, Mathematics*, 2010. doi: 1012.2599.

F.S. Broekgaarden, S. Justham, S.E. DeMinK, J. Gair, I. Mandel, S. Stevenson, J.W. Barrett, A. VignaGomez, and C.J. Neijssel. Stroop-wafel: Simulating rare outcomes from astrophysical populations, with application to gravitational-wave sources. *Monthly Notices of the Royal Astronomical Society*, 490:5228–5248, 2019.

P. Challenor. The probability of rapid climate change. *Significance*, 1(4):155–158, 2004.

K.L. Chang and S. Guillas. Computer model calibration with large nonstationary spatial outputs: Application to the calibration of a climate model. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 68(1):51–78, 2018.

V. Chen, M.M. Dunlop, O. Papaspiliopoulos, and A.M. Stuart. Robust mcmc sampling with non-gaussian and hierarchical priors in high dimensions. *Mathematics*, 2018. doi: 1803.03344.

C.S. Cheng, R.J. Martin, and B. Tang. Two-level factorial designs with extreme numbers of level changes. *Annals of Statistics*, 26(4):1522–1539, 1998.

D. Cohn. Neural network exploration using optimal experiment design. *In Advances in Neural Information Processing Systems*, page 679–686, 1994.

N.A. Cressie. Statistics for spatial data. *Revised edition. John Wiley & Sons*, 1993. doi: 10.1002/9781119115151.

C. Currin, T. Mitchell, M. Morris, and D. Ylvisaker. A bayesian approach to the design and analysis of computer experiments. *(Technical Report 6498). Oak Ridge National Laboratory*, 1988.

C. Currin, T. Mitchell, M. Morris, and D. Ylvisaker. Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. *Journal of the American Statistical Association*, 86:953–963, 1991.

Z. Dai, A. Damianou, J. González, and N.D. Lawrence. Variational auto-encoded deep gaussian processes. *International Conference on Learning Representations*, 3, 2016.

A. Damianou and N. Lawrence. Deep gaussian processes. *In Artificial Intelligence and Statistics, PMLR*, 31:207–215, 2013.

A.C. Damianou, M.K. Titsias, and N.D. Lawrence. Variational inference for latent variables and uncertain inputs in gaussian processes. *Journal of Machine Learning Research, 17(42):1-62*, 17(42):1–62, 2016.

M.M. Dunlop, M.A. Girolami, A.M. Stuart, and A.L. Teckentrup. How deep are deep gaussian processes? *Journal of Machine Learning Research*, 19:1–46, 2018.

V. Dutordoir, N. Knudde, J. Vander Herten, I. Couckuyt, and T Dhaene. Deep gaussian process metamodeling of sequentially sampled non-stationary response surfaces. *In 2017 Winter Simulation Conference (WSC)*, page 1728–1739, 2017.

V. Dutordoir, H. Salimbeni, E. Hambro, J. McLeod, F. Leibfried, A. Artemev, M. Vander Wilk, M.P. Deisenroth, J. Hensman, and S. John. Gpflux: A library for deep gaussian processes. 2021. doi: 2104.05674.

N.R. Edwards, D. Cameron, and J. Rougier. Precalibrating an intermediate complexity climate model. *Climate Dynamics*, 37(7-8):1469–1482, 2011.

M. Fuentes. A high frequency kriging approach for nonstationary environmental processes. *Environmetrics*, 12(5):469–483, 2001.

M. Fuentes and R. Smith. A new class of nonstationary spatial models. *Journal of the American Statistical*, 2003.

A. Gelfand and A. Smith. Sampling based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85:398–409, 1990.

S. Geman and D. Geman. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

R.B. Gramacy and D.W. Apley. Local gaussian process approximation for large computer experiments. *Journal of Computational and Graphical Statistics*, 24(2):561–578, 2015.

R.B. Gramacy and H.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.

R.B. Gramacy and H.K.H. Lee. Adaptive design and analysis of supercomputer experiments. *Technometrics*, 51(2):130–145, 2009.

R.B. Gramacy, G.A. Gray, S. Le Digabel, H.K.H. Lee, P. Ranjan, G. Wells, and S.M. Wild. Modeling an augmented lagrangian for blackbox constrained optimization (with discussion). *Technometrics*, 58(1):1–29, 2016.

W. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109, 1970.

A. Hebbal, L. Brevault, M. Balesdent, E. Talbi, and N. Melab. Bayesian optimization using deep gaussian processes with applications to aerospace system design. *Optimization and Engineering*, 22:321–361, 2021.

J. Hensman and N.D. Lawrence. Nested variational compression in deep gaussian processes. 2014. doi: 1412.1370.

J. Hensman, N. Fusi, and N. Lawrence. Gaussian processes for big data. *Uncertainty in Artificial Intelligence*, page 282–290, 2013.

J. Hensman, A.G. Matthews, and Z. Ghahramani. Scalable variational gaussian process classification. *In 18th International Conference on Artificial Intelligence and Statistics*, page 351–360, 2015.

D. Higdon. A process-convolution approach to modeling temperatures in the north atlantic ocean. *Journal of Environmental and Ecological Statistics*, 5:173–190, 1998.

D. Higdon, J. Swall, and J. Kern. Non-stationary spatial modeling. *Bayesian Statistics*, 6 (1):761–768, 1999.

D. Higdon, M. Kennedy, J. Cavendish, J. Cafeo, and R. Ryne. Combining field data and computer simulations for calibration and prediction. *SIAM Journal on Scientific Computing*, 26(2):448–466, 2004.

D. Higdon, J. Gattiker, B. Williams, and M. Rightley. Computer model calibration using high-dimensional output. *Journal of the American Statistical Association*, 103:570–583, 2008.

M.D. Hoffman, D.M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.

B. Iooss and P. Lemaitre. A review on global sensitivity analysis methods. in uncertainty management in simulation-optimization of complex systems: Algorithms and applications. *Springer*, 2015.

K.M. Irvine, A.I. Gitelman, and J.A. Hoeting. Spatial designs and properties of spatial correlation: Effects on covariance estimation. *J. Agric. Biol. Environ. Stat. MR2405534*, 12:450–469, 2007.

M. Johnson, L. Moore, and D. Ylvisaker. Minimax and maximin distance designs. *Journal of Statistical Planning and Inference*, 26:131–148, 1990.

D.R. Jones, M. Schonlau, and W.J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998.

M.I. Jordan, Z. Ghahramani, T.S. Jaakkola, and L.K. Saul. An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2):183–233, 1999.

C.G. Kaufman and S.R. Sain. Bayesian functional ANOVA modeling using gaussian process prior distributions. *Bayesian Analysis*, 5:123–149, 2010.

C.G. Kaufman, D. Bingham, S. Habib, K. Heitmann, and J.A. Frieman. Efficient emulators of computer experiments using compactly supported correlation functions, with an application to cosmology. *The Annals of Applied Statistics*, 4(5):2470–2492, 2011.

M.C. Kennedy and A. O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

D.P. Kingma and J.L. Ba. Adam: A method for stochastic optimization. 2014. doi: 1412.6980.

D.P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. *Advances in Neural Information Processing Systems*, 28, 2015. doi: 1506.02557.

M.U. Kruckow, T.M. Tauris, N. Langer, M. Kramer, and R.G. Izzard. Progenitors of gravitational wave mergers: Binary evolution with the stellar grid-based code combine. *Monthly Notices of the Royal Astronomical Society*, 481:1908–1949, 2018.

S. Kullback and R. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

B.A. Lockwood and M. Anitescu. Gradient-enhanced universal kriging for uncertainty propagation. *Nuclear Science and Engineering*, 170(2):168–195, 2012.

J.L. Loeppky, L.M. Moore, and B.J. Williams. Batch sequential designs for computer experiments. *Journal of Statistical Planning and Inference*, 140(6):1452–1464, 2010.

P. Lynch. The origins of computer weather prediction and climate modeling. *Journal of Computational Physics*, 227(7):3431–3444, 2008.

D. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

I. Mandel and A. Farmer. Gravitational waves: Stellar palaeontology. *Nature*, 547:284–285, 2017.

I. Mandel and A. Farmer. Merging stellar-mass binary black holes. 2018. doi: 1806.05820.

S. Marmin, D. Ginsbourger, J. Baccou, and J. Liandrat. Warped gaussian processes and derivative-based sequential design for functions with heterogeneous variations. *SIAM/ASA Journal on Uncertainty Quantification*, 2018. doi: 10.1137/17M1129179.

A.G. Matthews. Scalable gaussian process inference using variational methods. *PhD thesis, University of Cambridge*, 2017.

A.G. Matthews, J. Hensman, R.E. Turner, and Z. Ghahramani. On sparse variational methods and the kullback-leibler divergence between stochastic processes. *Artificial Intelligence and Statistics*, 2016.

A.G. Matthews, M. Van Der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman. Gpflow: A gaussian process library using tensorflow. *Journal of Machine Learning Research*, 2017.

M.D. McKay, W.J. Conover, and R.J. Beckman. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, (21):239–245, 1979.

N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, and A.H. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

D. Ming, D. Williamson, and S. Guillas. Deep gaussian process emulation using stochastic imputation. 2021. doi: 2107.01590.

T.J. Mitchell and D.S. Scott. A computer program for the design of group testing experiments. *Commun Stat Theory Methods*, 16:2943–2955, 1987.

K. Monterrubio-Gomez, L. Roninen, S. Wade, T. Damoulas, and M. Girolami. Posterior inference for sparse hierarchical non-stationary models. *Computational Statistics & Data Analysis, Elsevier, ISSN: 106954*, (148):0167–9473, 2020.

M.D. Morris and T.J. Mitchell. Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43:381–402, 1995.

J.E. Nash and J.V. Sutcliffe. River flow forecasting through conceptual models part i - a discussion of principles. *Journal of Hydrology*, 10:282–290, 1970.

C.J. Neijssel, A. Vigna-Gomez, S. Stevenson, J.W. Barrett, S.M. Gaebel, F. Broekgaarden, De Mink S.E., D. Szecsi, S. Vinciguerra, and I. Mandel. The effect of the metallicity-specific star formation history on double compact object mergers. *MNRAS*, 490:3740–3759, 2019.

W.I. Notz. Expected improvement designs. *In: Bingham D, Dean AM, Morris M, Stufken J (eds) Handbook of design and analysis of experiments. Chapman and Hall*, page 675–716, 2015.

J. Oakley and A. O'Hagan. Bayesian inference for the uncertainty distribution of computer model outputs. *Biometrika*, 89(4):769–784, 2002.

J.E. Oakley. Estimating percentiles of uncertain computer code outputs. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 53(1):83–93, 2004.

J.E. Oakley and A. O'Hagan. Probabilistic sensitivity analysis of complex models: a bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66 (3):751–769, 2004.

A. O'Hagan, M.C. Kennedy, and J.E. Oakley. Uncertainty analysis and other inference tools for complex computer codes. *Oxford University Press, In Bayesian Statistics 6, eds. J. M. Bernardo, J. O. Berger, A. Dawid, and A. Smith*, page 503–524, 1999.

C. Paciorek and M. Schervish. Non-stationary covariance functions for gaussian process regression. *Advances in Neural Information Processing Systems*, 16:273–280, 2004.

P. C. Peters and J. Mathews. Gravitational radiation from point masses in a keplerian orbit. *Physical Review*, 131:435–440, 1963.

L. Pronzato and W.G. Muller. Design of computer experiments: Space filling and beyond. *Statistics and Computing*, 22(3):681–701, 2012.

M.I. Radaideh and T. Kozlowski. Surrogate modeling of advanced computer simulations using deep gaussian processes. *Reliability Engineering and System Safety*, 195:106731, 2020.

D. Rajaram, T.G. Puranik, S. Ashwin Renganathan, W. Sung, O.P. Fischer, D.N. Mavris, and A. Ramamurthy. Empirical assessment of deep gaussian process surrogate models for engineering problems. *Journal of Aircraft*, page 1–15, 2020.

P. Ranjan, D. Bingham, and G. Michailidis. Sequential experiment design for contour estimation from complex computer codes. *Technometrics*, 50(4):527–541, 2008.

D.J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *International Conference on Machine Learning*, 2014.

S. Roberts, T. Guilford, I. Rezek, and D. Biro. Positional entropy during pigeon homing i: Application of bayesian latent state modelling. *Journal of Theoretical Biology*, 227:39–50, 2004.

S. Roy and W.I. Notz. Estimating percentiles in computer experiments: a comparison of sequential-adaptive designs and fixed designs. *Stat Theory Practice*, 8:12–29, 2014.

J. Sacks, S.B. Schiller, and W.J. Welch. Designs for computer experiments. *Technometrics*, 31(1):41–47, 1989a.

J. Sacks, W.J. Welch, T.J. Mitchell, and H.P. Wynn. Design and analysis of computer experiments. *Statist. Sci.MR1041765*, 4:409–435, 1989b.

H. Salimbeni. Deep gaussian processes: Advances in models and inference. *PhD thesis, Imperial College London*, 2020.

H. Salimbeni and M. Deisenroth. Doubly stochastic variational inference for deep gaussian processes. *In Advances in Neural Information Processing Systems*, page 44588–4599, 2017.

A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola. Global sensitivity analysis. *John Wiley and Sons Ltd, ISBN 978-0-470-05997-5*, 2008.

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

T.J. Santner, B.J. Williams, and W.I. Notz. The design and analysis of computer experiments. *New York, NY: Springer-Verlag*, 2003.

A. Sauer, R.B. Gramacy, and D. Higdon. Active learning for deep gaussian process surrogates. 2020. doi: 2012.08015.

A.M. Schmidt and A. O'Hagan. Bayesian inference for nonstationary spatial covariance structure via spatial deformations. *Journal of the Royal Statistical Society, Series B*, 65: 745–758, 2003.

M. Schonlau, W.J. Welch, and D.R. Jones. Global versus local search in constrained optimization of computer models. *In New Developments and Applications in Experimental Design, Institute of Mathematical Statistics*, 34:11–25, 1998.

M.C. Shewry and H.P. Wynn. Maximum entropy sampling. *Journal of applied statistics, 14(2):165–170*, 14(2):165–170, 1987.

E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. *In Advances in neural information processing systems*, page 1257–1264, 2006.

M.L. Stein. Interpolation of spatial data. *New York, NY: Springer*, 1999.

S. Stevenson, A. VignaGomez, I. Mandel, J.W. Barrett, C.J. Neijssel, D. Perkins, and S.E. DeMink. Formation of the first three gravitational-wave observations through isolated binary evolution. 2017. doi: 1704.01352.

B. Tang. Selecting latin hypercubes using correlation criteria. *Statistica Sinica*, 8:965–977, 1998.

B. Tang and C.F.J. Wu. A method for constructing supersaturated designs and its es2 optimality. *Canadian Journal of Statistics*, pages 191–201, 1997.

Z. Tavassoli, J.N. Carter, and P.R. King. Errors in history matching. *SPE Journal*, 9(3): 352–361, 2004.

S.R. Taylor and D. Gerosa. Mining gravitational-wave catalogs to understand binary stellar evolution: A new hierarchical bayesian framework. *Physical Review Journal*, 98, 2018. doi: 1806.08365.

M. Tipping and N. Lawrence. Variational inference for student-t models: Robust bayesian interpolation and generalised component analysis. *Neurocomputing*, 69:123–141, 2005.

M. Titsias. Variational learning of inducing variables in sparse gaussian processes. *In Artificial Intelligence and Statistics*, page 567–574, 2009.

M. Titsias and N. Lawrence. Bayesian gaussian process latent variable model. *in Artificial Intelligence and Statistics*, page 844–851, 2010.

I. Vernon, M. Goldstein, and R.G. Bower. Galaxy formation: a bayesian uncertainty analysis. *Bayesian Analysis*, 5(4):619–669, 2010.

A. Vigna-Gomez, C.J. Neijssel, S. Stevenson, J.W. Barrett, K. Belczynski, S. Justham, S.E. DeMink, B. Muller, P. Podsiadlowski, M. Renzo, D. Szecsil, and I. Mandel. On the formation history of galactic double neutron stars. 2018. doi: 1805.07974.

V. Volodina and D.B. Williamson. Diagnostic-driven non-stationary emulators using kernel mixtures. 2020. doi: 1803.04906.

M.J. Wainwright and M.I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 2008.

W.J. Welch, R.J. Buck, J. Sacks, H.P. Wynn, T. Mitchell, and M.D. Morris. Screening, predicting, and computer experiment. *Technometrics*, 34:15–25, 1992.

Appendix A

Supplementary Material for Chapter 3

A derivation for marginalizing inducing variables in (3.35) as

$$\int \mathbb{P}(\mathbf{u}_n|\tilde{\mathbf{u}}_n; \mathbf{u}_{n-1}, \mathbf{X}, \mathbf{Z}_n) q(\tilde{\mathbf{u}}_n)\ d\tilde{\mathbf{u}}_{\mathbf{n}} = \mathcal{N}(\mathbf{u}_n|\tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n),$$

for $n = 2, \ldots, N$ where

$$[\tilde{\boldsymbol{\mu}}_n]_i = \boldsymbol{\Gamma}_n(\mathbf{x}_i, \mathbf{u}^i_{n-1})^T \mathbf{m}_n,$$

$$[\tilde{\boldsymbol{\Sigma}}_n]_{ij} = k_n(\mathbf{x}_i, \mathbf{x}_j; \mathbf{u}^i_{n-1}, \mathbf{u}^j_{n-1}) - \boldsymbol{\Gamma}_n(\mathbf{x}_i, \mathbf{u}^i_{n-1})^T \Big[k_n(\mathbf{Z}_n, \mathbf{Z}_n; \boldsymbol{\delta}_{\mathbf{Z}_n}) - \mathbf{s}_n\Big] \boldsymbol{\Gamma}_n(\mathbf{x}_j, \mathbf{u}^j_{n-1}),$$

$$\boldsymbol{\Gamma}_n(\mathbf{x}_i, \mathbf{u}^i_{n-1}) = k_n(\mathbf{Z}_n, \mathbf{Z}_n; \boldsymbol{\delta}_{\mathbf{Z}_n})^{-1} k_n(\mathbf{Z}_n, \mathbf{x}_i; \boldsymbol{\delta}_{\mathbf{Z}_n}, \mathbf{u}^i_{n-1}).$$

Proof. We have $\mathbb{P}(\mathbf{u}_n|\tilde{\mathbf{u}}_n; \mathbf{u}_{n-1}, \mathbf{X}, \mathbf{Z}_n) = \mathcal{N}(\mathbf{u}_n|\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$, where

$$[\boldsymbol{\mu}_n]_i = \boldsymbol{\Gamma}_n(\mathbf{x}_i, \mathbf{u}^i_{n-1})^T \tilde{\mathbf{u}}_n,$$

$$[\boldsymbol{\Sigma}_n]_{ij} = k_n(\mathbf{x}_i, \mathbf{x}_j; \mathbf{u}^i_{n-1}, \mathbf{u}^j_{n-1}) - \boldsymbol{\Gamma}_n(\mathbf{x}_i, \mathbf{u}^i_{n-1})^T k_n(\mathbf{Z}_n, \mathbf{Z}_n; \boldsymbol{\delta}_{\mathbf{Z}_n}) \boldsymbol{\Gamma}_n(\mathbf{x}_j, \mathbf{u}^j_{n-1}).$$

For simplifying the notations we assume that

$$\boldsymbol{\mu}_n = \boldsymbol{\Sigma}^T_{\mathbf{u}_n \tilde{\mathbf{u}}_n} \boldsymbol{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} \tilde{\mathbf{u}}_n,$$

$$\boldsymbol{\Sigma}_n = \boldsymbol{\Sigma}_{\mathbf{u}_n} - \boldsymbol{\Sigma}^T_{\mathbf{u}_n \tilde{\mathbf{u}}_n} \boldsymbol{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} \boldsymbol{\Sigma}_{\mathbf{u}_n \tilde{\mathbf{u}}_n}.$$

Hence, $\tilde{\boldsymbol{\Sigma}}_n$ and $\tilde{\boldsymbol{\mu}}_n$ can be simplified as

$$\tilde{\boldsymbol{\mu}}_n = \boldsymbol{\Sigma}^T_{\mathbf{u}_n \tilde{\mathbf{u}}_n} \boldsymbol{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} \mathbf{m}_n,$$

$$\tilde{\boldsymbol{\Sigma}}_n = \boldsymbol{\Sigma}_n + \boldsymbol{\Sigma}^T_{\mathbf{u}_n \tilde{\mathbf{u}}_n} \boldsymbol{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} \mathbf{s}_n \boldsymbol{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} \boldsymbol{\Sigma}_{\mathbf{u}_n \tilde{\mathbf{u}}_n}.$$

It is clear that the conditional covariance matrix $\mathbf{\Sigma}_n$ does not involve $\tilde{\mathbf{u}}_n$, whereas $\boldsymbol{\mu}_n$ is a linear function of $\tilde{\mathbf{u}}_n$. Since $q(\tilde{\mathbf{u}}_n) = \mathcal{N}(\mathbf{m}_n, \mathbf{s}_n)$, so we can write $q(\mathbf{u}_n)$ as

$$\begin{aligned} q(\mathbf{u}_n) &= \int \mathbb{P}(\mathbf{u}_n|\tilde{\mathbf{u}}_n; \mathbf{u}_{n-1}, \mathbf{X}, \mathbf{Z}_n) q(\tilde{\mathbf{u}}_n) d\tilde{\mathbf{u}}_\mathbf{n} \\ &= \int \frac{1}{2\pi^{(m+m_{ind})/2}|\mathbf{\Sigma}_n|^{1/2}|\mathbf{s}_n|^{1/2}}\ exp(-\frac{1}{2}Q)d\tilde{\mathbf{u}}_\mathbf{n} \\ &= \frac{1}{2\pi^{(m+m_{ind})/2}|\mathbf{\Sigma}_n|^{1/2}|\mathbf{s}_n|^{1/2}} \int exp(-\frac{1}{2}Q)d\tilde{\mathbf{u}}_\mathbf{n}, \end{aligned}$$

where

$$\begin{aligned} Q &= [(\mathbf{u}_n - \boldsymbol{\mu}_n)^T \mathbf{\Sigma}_n^{-1}(\mathbf{u}_n - \boldsymbol{\mu}_n)] + [(\tilde{\mathbf{u}}_n - \mathbf{m}_n)^T \mathbf{s}_n^{-1}(\tilde{\mathbf{u}}_n - \mathbf{m}_n)] \\ &= [(\mathbf{u}_n - \mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\tilde{\mathbf{u}}_n)^T \mathbf{\Sigma}_n^{-1}(\mathbf{u}_n - \mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\tilde{\mathbf{u}}_n)] + [(\tilde{\mathbf{u}}_n - \mathbf{m}_n)^T \mathbf{s}_n^{-1}(\tilde{\mathbf{u}}_n - \mathbf{m}_n)] \\ &= A + B, \end{aligned}$$

and

$$\begin{aligned} A &= \mathbf{u}_n^T \mathbf{\Sigma}_n^{-1}\mathbf{u}_n - 2\mathbf{u}_n^T\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\tilde{\mathbf{u}}_n + \tilde{\mathbf{u}}_n^T\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\tilde{\mathbf{u}}_n, \\ B &= \tilde{\mathbf{u}}_n^T\mathbf{s}_n^{-1}\tilde{\mathbf{u}}_n - 2\tilde{\mathbf{u}}_n^T\mathbf{s}_n^{-1}\mathbf{m}_n + \mathbf{m}_n^T\mathbf{s}_n^{-1}\mathbf{m}_n. \end{aligned}$$

Since the first term of A and the last term of B do not depend on $\tilde{\mathbf{u}}_n$, so $q(\mathbf{u}_n)$ can reach to this form

$$q(\mathbf{u}_n) = \frac{1}{2\pi^{(m+m_{ind})/2}|\mathbf{\Sigma}_n|^{1/2}|\mathbf{s}_n|^{1/2}} exp(-\frac{1}{2}[\mathbf{u}_n^T\mathbf{\Sigma}_n^{-1}\mathbf{u}_n + \mathbf{m}_n^T\mathbf{s}_n^{-1}\mathbf{m}_n]) \int exp(-\frac{1}{2}Q')d\tilde{\mathbf{u}},$$

where

$$\begin{aligned} Q' &= -2\mathbf{u}_n^T\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\tilde{\mathbf{u}}_n + \tilde{\mathbf{u}}_n^T\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\tilde{\mathbf{u}}_n + \tilde{\mathbf{u}}_n^T\mathbf{s}_n^{-1}\tilde{\mathbf{u}}_n - 2\tilde{\mathbf{u}}_n^T\mathbf{s}_n^{-1}\mathbf{m}_n \\ &= -2[\mathbf{u}_n^T\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} + \mathbf{m}_n^T\mathbf{s}_n^{-1}]\tilde{\mathbf{u}}_n + \tilde{\mathbf{u}}_n^T[\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} + \mathbf{s}_n^{-1}]\tilde{\mathbf{u}}_n. \end{aligned}$$

It follows that

$$\frac{1}{2}Q' = -[\mathbf{u}_n^T\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} + \mathbf{m}_n^T\mathbf{s}_n^{-1}]\tilde{\mathbf{u}}_n + \frac{1}{2}\tilde{\mathbf{u}}_n^T[\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} + \mathbf{s}_n^{-1}]\tilde{\mathbf{u}}_n.$$

The multivariate generalization of a mathematical trick known as "completion of squares" says that for a symmetric, non-singular matrix $\mathbf{A}$, the quadratic function can be written as

$$\frac{1}{2}\mathbf{Z}^T\mathbf{A}\mathbf{Z} - \mathbf{b}^T\mathbf{Z} + \mathbf{C} = \frac{1}{2}(\mathbf{Z} - \mathbf{A}^{-1}\mathbf{b})^T\mathbf{A}(\mathbf{Z} - \mathbf{A}^{-1}\mathbf{b}) - \frac{1}{2}\mathbf{b}^T\mathbf{A}^{-1}\mathbf{b} + \mathbf{C}.$$

Now, we can apply this trick in our situation by these assumptions

$$\mathbf{Z} := \tilde{\mathbf{u}}_n \quad , \quad \mathbf{C} := \mathbf{0},$$

$$\mathbf{b} := [\mathbf{u}_n^T\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} + \mathbf{m}_n^T\mathbf{s}_n^{-1}]^T,$$

$$\mathbf{A} := \mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}_n^{-1}\mathbf{\Sigma}^T_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\mathbf{\Sigma}^{-1}_{\tilde{\mathbf{u}}_n} + \mathbf{s}_n^{-1}.$$

By application of Aitken's integral and this fact that $\mathbf{b}$ in our setting does not involve $\tilde{\mathbf{u}}_n$, $q(\mathbf{u}_n)$ can be simplified as

$$
\begin{aligned}
q(\mathbf{u}_n) &= \frac{2\pi^{m_{ind}/2}|\mathbf{A}^{-1}|^{1/2}}{2\pi^{(m+m_{ind})/2}|\boldsymbol{\Sigma}_n|^{1/2}|\mathbf{s}_n|^{1/2}} exp(-\frac{1}{2}[\mathbf{u}_n^T\boldsymbol{\Sigma}_n^{-1}\mathbf{u}_n + \mathbf{m}_n^T\mathbf{s}_n^{-1}\mathbf{m}_n - \mathbf{b}^T\mathbf{A}^{-1}\mathbf{b}]) \\
&= \frac{|\mathbf{A}^{-1}|^{1/2}}{2\pi^{m/2}|\boldsymbol{\Sigma}_n|^{1/2}|\mathbf{s}_n|^{1/2}} exp(-\frac{1}{2}[\mathbf{u}_n^T\boldsymbol{\Sigma}_n^{-1}\mathbf{u}_n + \mathbf{m}_n^T\mathbf{s}_n^{-1}\mathbf{m}_n - \mathbf{b}^T\mathbf{A}^{-1}\mathbf{b}]).
\end{aligned}
$$

Now by showing

$$
[(\mathbf{u}_n - \tilde{\boldsymbol{\mu}}_n)^T\tilde{\boldsymbol{\Sigma}}_n^{-1}(\mathbf{u}_n - \tilde{\boldsymbol{\mu}}_n)] = [\mathbf{u}_n^T\boldsymbol{\Sigma}_n^{-1}\mathbf{u}_n + \mathbf{m}_n^T\mathbf{s}_n^{-1}\mathbf{m}_n - \mathbf{b}^T\mathbf{A}^{-1}\mathbf{b}],
$$

and

$$
|\tilde{\boldsymbol{\Sigma}}_n|^{-1/2} = \frac{|\mathbf{A}^{-1}|^{1/2}}{|\boldsymbol{\Sigma}_n|^{1/2}|\mathbf{s}_n|^{1/2}},
$$

the proof will be completed. To proof the first equation, we start from the left hand side (LHS) to reach the the right hand side (RHS). Using the Woodbury identity, $\tilde{\boldsymbol{\Sigma}}_n^{-1}$ can be writen as

$$
\begin{aligned}
\tilde{\boldsymbol{\Sigma}}_n^{-1} &= \boldsymbol{\Sigma}_n^{-1} - \boldsymbol{\Sigma}_n^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}^T\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1}(\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\boldsymbol{\Sigma}_n^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}^T\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1} + \mathbf{s}_n^{-1})^{-1}\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\boldsymbol{\Sigma}_n^{-1} \\
&= \boldsymbol{\Sigma}_n^{-1} - \boldsymbol{\Sigma}_n^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}^T\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1}\mathbf{A}^{-1}\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\boldsymbol{\Sigma}_n^{-1}.
\end{aligned}
$$

By plugging $\tilde{\boldsymbol{\mu}}_n$ and $\tilde{\boldsymbol{\Sigma}}_n^{-1}$ into the LHS and using equations

$$
\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\boldsymbol{\Sigma}_n^{-1}\mathbf{u}_n = \mathbf{b} - \mathbf{s}_n^{-1}\mathbf{m}_n,
$$

and

$$
\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\boldsymbol{\Sigma}_n^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}^T\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1} = \mathbf{A} - \mathbf{s}_n^{-1},
$$

we reach

$$
\begin{aligned}
LHS &= \mathbf{u}_n^T\boldsymbol{\Sigma}_n^{-1}\mathbf{u}_n - (\mathbf{b} - \mathbf{s}_n^{-1}\mathbf{m}_n)^T\mathbf{m}_n - (\mathbf{b} - \mathbf{s}_n^{-1}\mathbf{m}_n)^T\mathbf{A}^{-1}(\mathbf{b} - \mathbf{s}_n^{-1}\mathbf{m}_n) \\
&+ 2\mathbf{m}_n^T(\mathbf{A} - \mathbf{s}_n^{-1})\mathbf{A}^{-1}(\mathbf{b} - \mathbf{s}_n^{-1}\mathbf{m}_n) - \mathbf{m}_n^T(\mathbf{b} - \mathbf{s}_n^{-1}\mathbf{m}_n) + \mathbf{m}_n^T(\mathbf{A} - \mathbf{s}_n^{-1})\mathbf{m}_n \\
&- \mathbf{m}_n^T(\mathbf{A} - \mathbf{s}_n^{-1})\mathbf{A}^{-1}(\mathbf{A} - \mathbf{s}_n^{-1})\mathbf{m}_n \\
&= \mathbf{u}_n^T\boldsymbol{\Sigma}_n^{-1}\mathbf{u}_n + \mathbf{m}_n^T\mathbf{s}_n^{-1}\mathbf{m}_n - \mathbf{b}^T\mathbf{A}^{-1}\mathbf{b} \\
&= RHS.
\end{aligned}
$$

The next equation can be written as

$$
|\tilde{\boldsymbol{\Sigma}}_n^{-1}|^{1/2} = |\mathbf{A}^{-1}|^{1/2}|\boldsymbol{\Sigma}_n^{-1}|^{1/2}|\mathbf{s}_n^{-1}|^{1/2},
$$

using this fact that

$$
\tilde{\boldsymbol{\Sigma}}_n^{-1} = \boldsymbol{\Sigma}_n^{-1} - \boldsymbol{\Sigma}_n^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}^T\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1}\mathbf{A}^{-1}\boldsymbol{\Sigma}_{\tilde{\mathbf{u}}_n}^{-1}\boldsymbol{\Sigma}_{\mathbf{u}_n\tilde{\mathbf{u}}_n}\boldsymbol{\Sigma}_n^{-1}.
$$

□