University of Exeter

Department of Mathematics

Uncertainty Quantification for complex computer models with nonstationary output. Bayesian optimal design for iterative refocussing

Victoria Volodina

September 2019

Supervised by

Daniel Williamson

Submitted by Victoria Volodina, to the University of Exeter as a thesis for the degree of Doctor of Philosophy in Mathematics, September 2019.

I certify that all material in this thesis which is not my own work has been identified and that no material has previously been submitted and approved for the award of a degree by this or any other University.

Signature: .....

Abstract

In this thesis, we provide the Uncertainty Quantification (UQ) tools to assist automatic and robust calibration of complex computer models. Our tools allow users to construct a cheap (statistical) surrogate, a Gaussian process (GP) emulator, based on a small number of climate model runs. History matching (HM), the calibration process of removing parameter space for which computer model outputs are inconsistent with the observations, is combined with an emulator. The remaining subset of parameter space is termed the Not Ruled Out Yet (NROY).

A weakly stationary GP with a covariance function that depends on the distance between two input points is the principal tool in UQ. However, the stationarity assumption is inadequate when we operate with a heterogeneous model response. In this thesis, we develop diagnostic-led nonstationary GP emulators with a kernel mixture. We employ diagnostics from a stationary GP fit to identify input regions with distinct model behaviour and obtain mixing functions for a kernel mixture. The result is a continuous emulator in parameter space that adapts to changes in model response behaviour.

History matching has proven to be more effective when performed in waves. At each wave of HM, a new ensemble is obtained to update an emulator before finding an NROY space. In this thesis, we propose a Bayesian experimental design with a loss function that compares the volume of the NROY space obtained with an updated emulator to the volume of the "true" NROY space obtained using a "perfect" emulator. We combine Bayesian Design Criterion with our proposed nonstationary GP emulator to perform calibration of climate model.

Acknowlegments

I want to thank my supervisor Daniel Williamson for his support and guidance throughout my whole PhD. There is a lot more that I am grateful for, but first and foremost thank you for encouraging me to do and actually finish this PhD. I would also like to thank Peter Challenor for acting as my second advisor and his invaluable advice on my research in the moments when I was stuck.

Special mention goes to my collaborators from HIGH-TUNE project who provided the application for my work. In particular, I would like to thank Frédéric Hourdin for being helpful and patient with all my struggles to run and install a Single Column Climate model. I would also like to thank Fleur Couvreux and Romain Roehrig.

A huge thank you goes to my family. I will always be grateful to my mother, who has been my rock and my role model. Thank you for believing in me in times when I didn't believe in myself. Although, your constant praise made me a little bit big-headed. I would also like to thank my father for supporting me through all these years.

I would like to thank my colleagues and friends from EuroClim project: Stephen, Phil, Ruth, Jo and Mike for being awesome office mates. Special mention goes to my two friends Penny and Blanca, who have never failed to give valuable professional as well as personal advice. Thank you for all the coffee chats in Exploding Bakery and fun times in Madrid.

I must also thank people from Laver 601. To Louise, who has been a huge support and cheerleader in my last two years of PhD, I will always be grateful. Your cakes and cat videos never failed to cheer me up! A special mention goes to Wenzhe, who shared all my struggles with climate model and surviving some of the academic

conferences. I would also like to thank Evan for our countless morning chats in the office and coffee breaks, that I miss very much. And finally, thank you Heba for your baking sessions and movie nights.

Finally thank you to my amazing friends outside math circle. I would like to thank Elena for being an awesome housemate in my last two years in Exeter. Thank you to my best friend Yana, who despite being extremely busy at her work, was always there to listen and welcome me for a weekend in Paris.

Contents

List of Tables

List of Figures

Publications

The majority of the results of Chapter 4 have appeared in the following:

Volodina, Victoria, and Daniel Williamson. "Diagnostic-Driven Nonstationary Emulators Using Kernel Mixtures." SIAM/ASA Journal on Uncertainty Quantification 8.1 (2020): 1-26.

Chapter 1

Introduction

Computer models are used across a wide range of fields to study physical (real-life) systems. Based on these models, we perform inference about the past, current and future states of the system. For example, Galform, the state-of-the-art model in cosmology, is used to study the creation and evolution of one million galaxies from the beginning of the Universe until the current day (Vernon et al., 2010). In the climate community, large-scale models such as the General circulation model (GCM) are employed to generate predictions about the state of the climate. Due to the complexity of this type of model; it is computationally expensive to run a simulation.

A possible solution to this problem is to construct a surrogate based on a limited number of computer model runs. Gaussian Process (GP) emulator is a type of surrogate that generates a prediction for a computer model together with uncertainty on the prediction. A standard class of GP emulators, i.e. with stationary kernels, fails to provide accurate predictions and a fair assessment of uncertainty to the nonstationary (heterogeneous) model response. In this thesis, we present a nonstationary GP emulator with kernel mixtures. We use diagnostics from stationary GP fits to partition the input space into different regions with distinct model response behaviour. We then proceed to fit a single GP emulator based on diagnostics results by varying the correlated residual behaviour across the input space, which allows users to operate within the original input space and retain interpretability of model response behaviour across the input space.

After assessing the adequacy of a proposed GP emulator to represent a model response, it is mainly used to carry out computer model related tasks, such as calibration, sensitivity analysis and uncertainty analysis.

One of the important types of calibration that we consider in detail in this thesis is history matching. History matching is performed by ruling out regions of input parameter space that are considered unlikely to give a model output consistent with real-world observation. On the contrary, the retained regions of input space, the not ruled out yet (NROY) space, correspond to acceptable matches between observations and computer model outputs. History matching is most effective when it is performed in waves, i.e. refocussing steps. At each step, a new design is required within the current NROY space to update an emulator and perform the cutting down of the input space. In this thesis, we present a Bayesian Design Criterion with a loss function that compares the volumes of the NROY space obtained at the next iteration of history matching and the 'true' NROY space. The 'true' NROY space is obtained by assuming the existence of 'perfect' emulator. The Bayesian Optimal Design is found by minimizing the Bayesian Design Criterion. The uniqueness of the proposed approach is that the Bayesian Design Criterion takes into account not only the knowledge about the size and shape of the NROY space but also incorporates the measures of uncertainty provided by our GP emulator. The last feature is particularly useful in situations when the computer model response behaviour is nonstationary and therefore we might be interested to improve our knowledge about model behaviour in a more complex, high-variability input region.

The final contribution of this thesis is the ExeterUQ library for modelling and quantifying uncertainties in complex computer simulators. In particular, the core components of ExeterUQ are GP emulation and history matching. Prior specification for GP hyperparameters allows modellers, non-experts in statistics, to obtain stable and robust predictions produced by a GP emulator. We demonstrate the functionality of ExeterUQ in application to a Single Column Model (SCM) of a GCM and Large Eddies simulations (LES).

1.1 Thesis Outline

In Chapter 2, we provide a review of the current approaches adopted in Uncertainty Quantification (UQ). We mainly focus on emulation, the diagnostics used to validate the adequacy of proposed GP emulator to model simulator output, nonstationary GP models and history matching.

Chapter 3 presents ExeterUQ software for modelling and quantifying uncertainties in complex computer simulators. We discuss the characteristic features of our developed software, such as GP emulation and history matching, and how it is different from other available and open source software and packages.

In Chapter 4, we present our nonstationary GP emulator with kernel mixtures. At the beginning of this Chapter, we carefully review the partitioning approaches such as Treed Gaussian Process (TGP) (Gramacy and Lee, 2008) and Voronoi Tessellation GP (Pope et al., 2018) as well as Composite Gaussian Process (CGP) (Ba and Joseph, 2012). We present a diagnostic-led approach to fitting nonstationary GP emulators by specifying finite mixtures of region-specific covariance kernels. Our method first fits a stationary GP and, if traditional diagnostics exhibit nonstationarity, those diagnostics are used to fit appropriate mixing functions for a covariance kernel mixture designed to capture the nonstationarity, ensuring an emulator that is continuous in parameter space and readily interpretable. We compare our approach to the principal nonstationary GP models, i.e. TGP and CGP, on a number of idealised test cases as well as the nonstationary response produced by a climate model. We finish off this Chapter by demonstrating the importance of nonstationary GP emulators for history matching when we are dealing with nonstationary model response.

In Chapter 5, we present the Bayesian Design Criterion (BDC) for history matching. We start this chapter by discussing the challenges encountered in the process of generating an ensemble in the not ruled out space, and the current approaches adopted. We proceed to introduce our Bayesian Design Criterion, as well as each individual component of the proposed criterion. We conduct a simulation study on a simple 2D toy model to compare the effect of a space-filling design over the current

not ruled out space to a Bayesian Optimal Design on Wave 2 NROY space size and shape.

In Chapter 6, we adapt our proposed Bayesian Design Criterion to the situation of attempting to perform history matching with a nonstationary computer model output. We state a number of assumptions for our proposed nonstationary GP model to ensure that we achieve an optimal design by fixing a number of input regions and the form of mixing functions throughout the whole process of history matching. We derive that the BDC employed with our proposed nonstationary GP model takes into account the decomposition of the current NROY input space, in particular the relative volume of regions of input space with distinct model behaviour. We perform two simulation studies using climate model output.

Chapter 7 offers concluding remarks and highlights a number of potential areas for further work.

Chapter 2

Literature Review

2.1 Computer simulators

A computer model (simulator) is a coded representation of a true process, and is usually treated as a mathematical function f, that takes varying values of input parameters denoted by a vector $\mathbf{x} = (x_1, \ldots, x_p) \in \mathbb{R}^p$, and returns output $f(\mathbf{x})$. Computer simulations have been used in oil reservoir modelling (Tavassoli et al., 2004), analysis of Galaxy formation (Vernon et al., 2010), climate and environmental sciences (Challenor, 2004; Lynch, 2008; Edwards, 2010; Lee et al., 2011), industrial design and engineering (Ankenman et al., 2010; Rogers et al., 2003), as well as in medical applications such as HIV transmission modelling (Andrianakis et al., 2017) and neural mass models (Ferrat et al., 2018).

Computer models are widely used for prediction and forecasting. For instance in weather forecasting, a General circulation model (GCM) of the atmosphere is used to predict density, pressure and air velocity (wind) in the near future, usually 15 days (Gettelman and Rood, 2016). Computer models can be used to study the process of interest under different scenarios. For instance, climate modellers use four representative concentration pathways (RCPs) scenarios, that represent radiative forcing values from 2.6 to 8.5 W/m^2 in 2100. These scenarios contain a range of variables (parameters) that reflects the socio-economic change, technological change, energy use, and emissions of greenhouse gases and air pollutants (van Vuuren et al., 2011). These variables are used as inputs to a climate model and their impact on

the climate through the year 2100 is considered to understand climate change.

Computer models can produce many forms of output such as a single scalar, a vector of scalars, time series, a spatial field or a combination of the output variables mentioned above. For example, the output of the IC fault oil reservoir model is a time series of the oil production rate, the water injection rate, and the water cut rate (Tavassoli et al., 2004, 2005). General circulation models (GCMs), or global climate models, such as LMDZ (Laboratoire de Météorologie Dynamique) (Hourdin et al., 2006), use discretised Navier-Stokes dynamical equations on the sphere and produce different output fields e.g. potential temperature, pressure and precipitation over a horizontal and vertical grid. The output of a computer model can be deterministic or stochastic. Deterministic models produce the same output under the same model conditions, whilst stochastic models output possess inherent randomness. For instance, the Susceptible-Infected-Recovered (SIR) epidemic model has a stochastic version to predict number of infected from communicable diseases, such as influenza or dengue (Binois et al., 2018). The inherent randomness in model output comes from the process of solving stochastic differential equations as part of the model.

The input parameters of a computer model could be observable and directly linked to the process of interest, or could be used to account for inadequate physics in the simulator (Higdon et al., 2008).

The process that we are trying to describe could be very complex, leading to a large and expensive computer model, in terms of a single run time, that suffers from a number of uncertainty sources (Kennedy and O'Hagan, 2001). Prior to using a computer simulator to make informed decisions or studies about the true process of interest, we should attempt to quantify and take into account different sources of uncertainty.

2.2 Uncertainty quantification

Uncertainty quantification (UQ) is a field that focuses around quantifying and taking account of uncertainties for mathematical and computer models that attempt to

describe real-world processes.

Kennedy and O'Hagan (2001) classify various sources of uncertainty in computer models. We encounter parameter uncertainty when we are dealing with a computer model that contains a number of unknown input parameters. In particular, the inverse problem is a vast area of UQ that attempts to learn about model input parameters from observations of the process that the model describes. Calibration is the process of finding the input parameter values that allow the computer model to be a trustworthy representation of the physical process; and is commonly used to solve the inverse problem (Kennedy and O'Hagan, 2001; Higdon et al., 2008; Chang and Guillas, 2018). For example, the input parameters space of a Canadian Atmospheric Global Climate Model (CanAM4) has been explored by comparing vertical air temperature produced by the model to observed temperature (Salter et al., 2018). For a calibration exercise, we usually use actual observations of the physical process of interest. The accuracy of observations is usually affected by human error and/or limitations of instruments and this effect can be expressed via the observation error term.

Another source of uncertainty arises from the notion that the model is not a perfect representation of the true process due to the process complexity and the lack of computational resources. For instance, climate models solve coupled PDEs discretized over the vertical and horizontal grids over the sphere numerically to study the effect of input variables of interest on the climate (Hourdin et al., 2017). For GCMs, grids with cell sizes of the order of 100 to 300 km horizontally are typically used, which leads to the inability to explicitly calculate the effect of cloud processes (Diallo et al., 2017). These processes which are too small are considered as "sub-grid scale processes", and the effect of these processes is approximated inside the model. As a result, model discrepancy, the difference between the process and the model representation, will appear.

Computer models are computationally expensive to run at as many input parameter settings as we usually require. We treat them as "black box" models since we only manage to learn their outputs if we run them for given inputs' values (Kennedy

and O'Hagan, 2001). Cheap surrogates (emulators) such as neural networks, splines and polynomial chaos could be used to approximate computer model behaviour across the input space (Jin et al., 2000; Chen et al., 2006; Owen et al., 2017; Mohammadi et al., 2018). Gaussian Process (GP) emulators, a non-parametric class of surrogate models, have become increasingly popular due to their flexibility, as no assumptions about the form of the simulator response are required (Mohammadi et al., 2018). Unlike other surrogate models, GP emulators provide a measure of uncertainty about the predicted model output.

Apart from calibration, there are other techniques in UQ used to deal with different sources of uncertainty in detail, such as uncertainty analysis and sensitivity analysis. Uncertainty analysis deals with the parameter uncertainty by looking at the distribution of the computer model output induced by a probability distribution on input parameters (Oakley and O'Hagan, 2002).

Sensitivity analysis looks at identifying how model input parameters affect the model outputs (Oakley and O'Hagan, 2004). There are two commonly used measures of sensitivity of computer model output to an individual parameter x_i, the main effect index and the total effect index. The main effect index is based on considering the expected reduction in the uncertainty in the computer model output after we learn the true value of x_i (Saltelli et al., 2000). The total effect index is based on quantifying the remaining uncertainty in the computer model output after we have learnt everything except x_i (Homma and Saltelli, 1996).

2.3 Emulation

An emulator is a cheap statistical representation of a computer model (Currin et al., 1988, 1991; Sacks et al., 1989; Santner et al., 2003). Emulators are used to produce various inferences about a computer model for uncertainty quantification. For example, model calibration, prediction, sensitivity and uncertainty analyses (discussed in subsection 2.2) use emulators to avoid generating more expensive computer model runs.

We start by defining a statistical model for a scalar output $f(\mathbf{x})$. An emulator

is typically considered as a sum of two processes (Williamson, 2015)

$$f(\mathbf{x}) = h(\mathbf{x})^T\boldsymbol{\beta} + \epsilon(\mathbf{x}), \tag{2.1}$$

where $h(\mathbf{x})^T\boldsymbol{\beta}$ represents a global response surface behaviour, $\epsilon(\mathbf{x})$ is a correlated residual process capturing local input dependent deviation from the global response surface.

Gaussian processes are the principal tools for both building emulators and representing our uncertainty about the simulator output (Santner et al., 2003; Bastos and O'Hagan, 2009).

2.3.1 Gaussian processes

A Gaussian process is a stochastic process defined for a collection of random variables such that every finite collection of those random variables has a multivariate normal distribution (Rasmussen and Williams, 2004; Santner et al., 2003). We specify that $f(\mathbf{x})$ is Gaussian process if, for any $n \geq 1$, $n \in \mathbb{N}$ and $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathcal{X}$, the vector $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n))$ has a multivariate normal distribution (Santner et al., 2003).

We model the residual term, $\epsilon(\mathbf{x})$, introduced in equation (2.1), as a zero-mean Gaussian Process with covariance function $k(\mathbf{x}, \mathbf{x}'; \sigma^2, \boldsymbol{\delta}) = \sigma^2 r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\delta})$. The covariance function depends on the vector of correlation length parameters, $\boldsymbol{\delta}$, defined inside the correlation function, $r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\delta})$, and variance parameter, σ^2. A careful and detailed review of covariance function structure is presented in subsection 2.3.2. The first term of equation (2.1) consists of $h(\mathbf{x})$, a vector of regression functions evaluated at $\mathbf{x}$, such as linear terms, powers of inputs, and interactions between inputs, whilst $\boldsymbol{\beta}$ is a vector of regression parameters.

We derive a Gaussian process distribution for $f(\mathbf{x})$ determined by the mean function

$$E[f(\mathbf{x})] = h(\mathbf{x})^T\boldsymbol{\beta} = \sum_{i=1}^{p} \boldsymbol{\beta}_i h_i(\mathbf{x}) \tag{2.2}$$

and the covariance function

$$Cov[f(\mathbf{x}), f(\mathbf{x}^{'})] = k(\mathbf{x}, \mathbf{x}'; \sigma^2, \boldsymbol{\delta}) = \sigma^2 r(\mathbf{x}, \mathbf{x}', \boldsymbol{\delta}), \tag{2.3}$$

where $r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\delta})$ is a user-specified correlation function. Using probabilistic notation, the probability distribution for $f(\mathbf{x})$ conditioned on the statistical model parameters $\{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}\}$ is

$$f(\mathbf{x})|\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta} \sim GP\Big(h(\mathbf{x})^T\boldsymbol{\beta}, k(\cdot, \cdot; \sigma^2, \boldsymbol{\delta})\Big) \tag{2.4}$$

(Currin et al., 1991). Suppose we observe n computer model realisations $\mathrm{F} = \big(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)\big)$ at design points $\mathrm{X} = \big(\mathbf{x}_1, \ldots, \mathbf{x}_n\big)$. According to equation (2.4), the distribution of outputs is multivariate normal,

$$\mathrm{F}|\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta} \sim MVN\Big(H\boldsymbol{\beta}, K\Big),$$

where $H = \big[h(\mathbf{x}_1), \ldots, h(\mathbf{x}_n)\big]^T$ and K is an $n \times n$ covariance matrix with entries $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Using standard techniques for conditioning in multivariate normal distributions, the distribution for f at a new input $\mathbf{x}$ given ensemble $\{\mathrm{X}, \mathrm{F}\}$ and parameters $\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}$ is

$$f(\mathbf{x})|\{\mathrm{X}, \mathrm{F}\}, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta} \sim GP(m^*(\mathbf{x}), C^*(\mathbf{x}, \mathbf{x}')) \tag{2.5}$$

with mean

$$m^*(\mathbf{x}) = E[f(\mathbf{x})|\{\mathrm{X}, \mathrm{F}\}, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}] = h(\mathbf{x})^T\boldsymbol{\beta} + k(\mathbf{x}, \boldsymbol{X})K^{-1}\big(\boldsymbol{F} - H\boldsymbol{\beta}\big) \tag{2.6}$$

and variance

$$C^*(\mathbf{x}, \mathbf{x}') = Cov[f(\mathbf{x}), f(\mathbf{x}')|\{\mathrm{X}, \mathrm{F}\}, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}] = k(\mathbf{x}, \mathbf{x}') - k(\mathbf{x}, \boldsymbol{X})K^{-1}k(\boldsymbol{X}, \mathbf{x}) \tag{2.7}$$

where $k(\mathbf{x}, \boldsymbol{X})$ is n-vector whose ith component is $k(\mathbf{x}, \mathbf{x}_i), i = 1, \cdots, n$, the covariance between the point of interest $\mathbf{x}$ and the design point $\mathbf{x}_i$.

2.3.2 Covariance functions

The covariance function, or kernel, is a crucial part of a Gaussian process emulator, and represents the similarity between the model output evaluated at two points $\mathbf{x}$ and $\mathbf{x}'$ in the input space $\mathcal{X}$. It also depends on the definition of the correlation function. There are several commonly used families of correlation functions in the computer experiments literature (Rasmussen and Williams, 2004). The power exponential correlation function is defined as

$$r(\mathbf{x}, \mathbf{x}') = \exp\left\{ -\sum_{j=1}^{p} \left(\frac{x_j - x'_j}{\delta_j} \right)^{\phi_j} \right\} \tag{2.8}$$

where $\delta_j > 0$ and $0 < \phi_j \leq 2$. In this thesis we mainly use a squared exponential correlation function with $\phi_j = 2, j = 1, \ldots, p$, which is an infinitely differentiable function. Gaussian Processes with this covariance function have mean square derivatives of all orders, and thus produce smooth sample paths (Rasmussen and Williams, 2004). The notion of mean squared differentiability of the process is directly connected to the differentiability of the correlation function. A Gaussian Process, $f(\cdot)$, has mean square partial derivative at $\mathbf{x}^*$, if and only if $\frac{\partial^2 r(\mathbf{x},\mathbf{x}')}{\partial \mathbf{x}_j, \mathbf{x}'_j}$ exists and is finite at $(\mathbf{x}^*, \mathbf{x}^*)$ (Adler, 1981, p. 27). If the derivative exists, the correlation function of the mean square partial derivative process is the partial derivative of the original correlation function (Paciorek, 2003). Stein (2012) advocates against using squared exponential correlation functions to model physical processes, and proves his point with a numerical study: the predictions from a GP model with a squared exponential correlation function were generated for a sample path from a GP with once mean square differentiable correlation function, for which $1 \leq \phi_j < 2$. The performance measures used for this study demonstrated the unsatisfactory performance of the GP model with a squared exponential correlation function. One possible solution to this problem could be to specify $\phi = 1.9$ which leads to a "rougher" Gaussian process representation (Bayarri et al., 2007). Williamson and Blaker (2014) adopted this form of the correlation function for their GP emulator to model the AMOC time series. However, we didn't find any practical issues with employing a squared

exponential correlation function for our GP specification.

Another set of important parameters of power exponential correlation functions are the correlation length parameters, $\boldsymbol{\delta}$. In each input dimension i, the distance between two input points is in the ϕ_ith power and rescaled by the corresponding entry of the correlation length vector in the ϕ_ith power. Larger values of δ_i leads to a stronger correlation for a pair of input points in the ith direction, while the complete opposite is true for small values of δ_i.

The Matérn correlation function (Paciorek and Schervish, 2006; Golchi et al., 2015) is defined as

$$r(\mathbf{x}, \mathbf{x}') = \frac{2^{1-\alpha}}{\Gamma(\alpha)} \left(\frac{\sqrt{2\alpha}|\mathbf{x} - \mathbf{x}'|}{\delta} \right)^{\alpha} \mathcal{K}_{\alpha} \left(\frac{\sqrt{2\alpha}|\mathbf{x} - \mathbf{x}'|}{\delta} \right)$$

where α, δ are positive parameters, $|\mathbf{x} - \mathbf{x}'|$ is a distance between two points in the input space, $\Gamma(\cdot)$ is a gamma function, $\mathcal{K}_{\alpha}$ is a modified Bessel function of the second kind. Matérn correlation functions produce Gaussian process sample paths that are $\alpha - 1$ differentiable and, as $\alpha \to \infty$, the Matérn correlation converges to the squared exponential (Rasmussen and Williams, 2004; Santner et al., 2003). Rasmussen and Williams (2004) stated that two choices, $\alpha = 3/2$ and $\alpha = 5/2$, are commonly used within the computer science community. These two choices are in between $\alpha = 1/2$, which leads to GP paths with a large number of local peaks, and $\alpha = 7/2$, which leads to overly smooth GP paths.

Let us return to the specification of the covariance function. In general, an arbitrary function of two inputs $\mathbf{x}$ and $\mathbf{x}'$ will not be a valid covariance function for Gaussian Process, unless it is a positive semidefinite (Rasmussen and Williams, 2004). A covariance function is said to be a positive semidefinite if it gives rise to a positive semidefinite (PSD) $n \times n$ matrix K with entries $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j; \sigma^2, \boldsymbol{\delta})$, where $\mathbf{x}_i$, $\mathbf{x}_j, i, j \in 1, \cdots, n$ are two design points in X. This condition is crucial as it allows us to compute the inverse of the covariance matrix K, which is necessary for fitting a Gaussian process emulator.

There are properties that we could adopt for the covariance function to reduce the computational costs related to a covariance matrix and its inverse. A stationary

covariance function depends only on the distance between two input points, $|\mathbf{x}-\mathbf{x}'|$, and thus is invariant to the translation in the input space (Bastos and O'Hagan, 2009; Santner et al., 2003; Rasmussen and Williams, 2004). An isotropic covariance function models the correlation between the pair of the input points the same way in each ith direction (Rasmussen and Williams, 2004). The isotropic property is achieved by deriving a single value of δ for each $i = 1, \ldots, p$, and therefore allowing users to estimate a single correlation length parameter. However, the stationarity and isotropy assumptions could be too strong for many models whose response varies across the input space, i.e. nonstationary response (see subsection 2.5). In this case, applying GP models with stationary and isotropic covariance functions could lead to unsatisfactory performance (see subsection 2.4).

A new form of covariance function could be generated from the combination and/or modification of existing covariance functions (Rasmussen and Williams, 2004). For example, the sum of two kernels is a kernel

$$k(\mathbf{x}, \mathbf{x}'; \sigma^2, \boldsymbol{\delta}) = k_1(\mathbf{x}, \mathbf{x}'; \sigma^2, \boldsymbol{\delta}_1) + k_2(\mathbf{x}, \mathbf{x}'; \sigma^2, \boldsymbol{\delta}_2).$$

In spatial statistics, the weighted sum of locally defined kernels is widely used to model nonstationarity in the spatial process and will be discussed in detail in Section 2.5 and Chapter 4 (Fuentes, 2001; Fuentes and Smith, 2001; Banerjee et al., 2004).

A nonstationary covariance function could be obtained by convolving the spatially-varying kernel functions (kernel density estimators) $K_{\mathbf{x}}(\boldsymbol{u})$ centred on $\mathbf{x}$ (Higdon et al., 1999; Paciorek and Schervish, 2006)

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

where $\mathbf{x}, \mathbf{x}'$ and $\boldsymbol{u}$ are locations (inputs) in $\mathcal{X}$. Higdon et al. (1999) used the Gaussian kernel density estimator

$$K_{\mathbf{x}}(\boldsymbol{u}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}^{1/2}|} \exp\Big[-\frac{1}{2}(\mathbf{x}-\boldsymbol{u})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{u}) \Big]$$

where $\boldsymbol{\Sigma}$ is the covariance matrix centred at $\mathbf{x}$. Integration over the input space (do-

main) allows us to evolve the covariance function spatially (Paciorek and Schervish, 2006) and this covariance function could be used to target a nonstationary model response (see section 2.5).

The product of kernels and the tensor product of kernels both produce valid kernels (Rasmussen and Williams, 2004)

$$k(\mathbf{x}, \mathbf{x}'; \sigma^2, \boldsymbol{\delta}) = \sigma^2 \times \Pi_{i=1}^{p} r(x_i, x_i'; \delta_i).$$

In this thesis we use separable squared exponential kernel and define

$$r(x_i, x_i', \delta_i) = \exp\left\{ -\left(\frac{x_i - x_i'}{\delta_i} \right)^2 \right\}.$$

2.3.3 The nugget parameter

During the process of fitting an emulator via equations (2.6) and (2.7), or the variants of these equations demonstrated in section 2.3.4, we could encounter numerical problems with the inversion of a covariance matrix, K (Neal, 1997). A nugget parameter is added to an emulator to alleviate the singularity of the covariance matrix (Ranjan et al., 2011; Andrianakis and Challenor, 2012).

The emulator definition in equation (2.1) from section 2.3 is modified by including an independent nugget term, $\nu(\mathbf{x})$,

$$f(\mathbf{x}) = h(\mathbf{x})^T \boldsymbol{\beta} + \epsilon(\mathbf{x}) + \nu(\mathbf{x}). \tag{2.9}$$

A nugget term is a zero mean normally distributed random quantity with variance τ^2 at all $\mathbf{x}$, and it is assumed to be uncorrelated with itself at different inputs (Andrianakis and Challenor, 2012; Gramacy and Lee, 2012; Neal, 1997; Ranjan et al., 2011), precisely

$$Cov[\nu(\mathbf{x}), \nu(\mathbf{x}')] = \begin{cases} \tau^2, & \mathbf{x} = \mathbf{x}', \\ 0, & \text{otherwise.} \end{cases}$$

The inclusion of a nugget term modifies our probabilistic specification for $f(\mathbf{x})$ in the equation (2.4)

$$f(\mathbf{x})|\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}, \tau^2 \sim GP\Big(h(\mathbf{x})^T\boldsymbol{\beta}, k(\cdot,\cdot;\sigma^2,\boldsymbol{\delta},\tau^2)\Big) \tag{2.10}$$

with a newly defined covariance function

$$Cov[f(\mathbf{x}), f(\mathbf{x}')] = k(\mathbf{x},\mathbf{x}';\sigma^2,\boldsymbol{\delta},\tau^2) = \sigma^2 r(\mathbf{x},\mathbf{x}';\boldsymbol{\delta}) + \tau^2 \mathbb{1}\{\mathbf{x}=\mathbf{x}'\},$$

where the indicator function is defined as

$$\mathbb{1}\{\mathbf{x}=\mathbf{x}'\} = \begin{cases} 1, & \mathbf{x}=\mathbf{x}', \\ 0, & \text{otherwise.} \end{cases}$$

However, this parameterization of the covariance function leads to an analytically intractable marginalisation of σ^2, implying that σ^2 would have to be estimated jointly with $\boldsymbol{\delta}$ or even marginalised numerically (Andrianakis and Challenor, 2012). An alternative way of adding a nugget parameter to a covariance function, adopted by Andrianakis and Challenor (2012); Gramacy and Lee (2012); Montagna and Tokdar (2016), is

$$Cov[f(\mathbf{x}), f(\mathbf{x}')] = k(\mathbf{x},\mathbf{x}';\sigma^2,\boldsymbol{\delta},\tau^2) = \sigma^2 \times \big[r(\mathbf{x},\mathbf{x}';\boldsymbol{\delta}) + \tau^2 \mathbb{1}\{\mathbf{x}=\mathbf{x}'\}\big],$$

where $\sigma^2\tau^2$ represents the variability that is not captured by the correlated part of the process.

The presence of a nugget results in GP models that do not interpolate the design points, X, and instead attaching a non-zero uncertainty band around them. In cases when we are aiming to obtain a numerical stability, the nugget process term is set to as small as possible value, so that its effect on the Gaussian process is negligible (Ranjan et al., 2011).

There are other possible reasons to include a nugget term in an emulator. The variation in response could be driven by a small number of inputs, known as active

inputs, $\mathbf{x}^A$, (Goldstein and Rougier, 2009; Craig et al., 2001). The global response surface and the residual process terms are modelled in terms of $\mathbf{x}^A$, whilst the nugget process term is used to account for any remaining variation in the model (Cumming and Goldstein, 2009). An emulator representation for $f(\mathbf{x})$ becomes

$$f(\mathbf{x}) = h(\mathbf{x}^A)^T \boldsymbol{\beta} + \epsilon(\mathbf{x}^A) + \nu(\mathbf{x}).$$

This model could provide significant computational savings by reducing dimensions of the residual process term. In spatial statistics, it is common to account for the measurement error or noise in spatial data through a nugget term (Cressie, 1993; Banerjee et al., 2004).

Gramacy and Lee (2012) demonstrated that the inclusion of a nugget allows a GP model to smooth the simulator output, acting like an extrapolator, and perform better than a GP model with zero nugget for a number of situations. For instance, in situations where we are unable to explore the simulator behaviour across the input space, the inclusion of a nugget term in the emulator could be useful. The interpolating emulator tends to produce predictions and confidence bands outside of the actual simulator response range. On the other hand, the emulator with a nugget manages to capture the global behaviour of the model and produces reasonable confidence bands by increasing variance at the points of interest.

2.3.4 Fitting a Gaussian process emulator

The simulator output, $f(\mathbf{x})$, is modelled as a Gaussian Process, as defined in equation (2.4). We observe the computer model runs, $\mathrm{F} = \big(f(\mathbf{x}_1), \cdots, f(\mathbf{x}_n)\big)$, at design points, $\mathrm{X} = \big(\mathbf{x}_1, \cdots, \mathbf{x}_n\big)$, and using the identities for multivariate Gaussian distributions, the distribution of $f(\mathbf{x})$ conditioned on GP parameters and ensemble, $\{\mathrm{X}, \mathrm{F}\}$, takes the form of equation (2.5).

In general, the GP parameters are unknown and have to be estimated or marginalised. Currin et al. (1991) proposed to fix the unknown hyperparameters at the values found by maximum likelihood. The expression for log-likelihood is derived, and the maximisation over σ^2 and $\boldsymbol{\beta}$ yields formulas dependent on $\boldsymbol{\delta}$. The $\hat{\boldsymbol{\delta}}$ is usually found

by a constrained iterative search, which gets increasingly computationally expensive as the number of input dimensions increases (Currin et al., 1991). There is a possible danger to get stuck in a local optima using numerical optimisation (Warnes and Ripley, 1987; Ripley, 1991).

Haylock and O'Hagan (1996) suggest specifying the non-informative prior for $\boldsymbol{\beta}$ and σ^2 in the form

$$\pi(\boldsymbol{\beta}, \sigma^2) \propto 1/\sigma^2. \tag{2.11}$$

The benefit of this prior is that the posterior analysis is tractable: it is possible to write down the posterior distribution conditioned on the correlation length parameter.

We start by integrating $\boldsymbol{\beta}$ out and obtaining the distribution of f conditioned on ensemble, $\{\mathrm{X}, \mathrm{F}\}$, and parameters σ^2 and $\boldsymbol{\delta}$. This is Gaussian as before with mean

$$m^{**}(\mathbf{x}) = E[f(\mathbf{x})|\{\mathrm{X}, \mathrm{F}\}, \sigma^2, \boldsymbol{\delta}] = h(\mathbf{x})^T\hat{\boldsymbol{\beta}} + k^T(\mathbf{x}, \mathrm{X})K^{-1}\left(\mathrm{F} - H\hat{\boldsymbol{\beta}}\right) \tag{2.12}$$

and variance

$$\begin{aligned} C^{**}(\mathbf{x}, \mathbf{x}') &= Cov[f(\mathbf{x}), f(\mathbf{x}')|\{\mathrm{X}, \mathrm{F}\}, \sigma^2, \boldsymbol{\delta}] = k(\mathbf{x}, \mathbf{x}') - k^T(\mathbf{x}, \mathrm{X})K^{-1}k(\mathbf{x}', \mathrm{X}) \\ &+ (h(\mathbf{x}) - k^T(\mathbf{x}, \mathrm{X})K^{-1}H)(H^TK^{-1}H)^{-1} \\ &\times (h(\mathbf{x}') - k^T(\mathbf{x}', \mathrm{X})K^{-1}H)^T, \end{aligned} \tag{2.13}$$

where $\hat{\boldsymbol{\beta}} = (H^TK^{-1}H)^{-1}H^TK^{-1}\mathrm{F}$. The next step is to integrate σ^2 and we derive that f conditional on $\{\mathrm{X}, \mathrm{F}\}$ and $\boldsymbol{\delta}$ is Student Process with $n-p$ degrees of freedom, with the same form of mean, $m^{**}(\mathbf{x})$, and the covariance defined as

$$C^{***}(\mathbf{x}, \mathbf{x}') = Cov[f(\mathbf{x}), f(\mathbf{x}')|\{\mathrm{X}, \mathrm{F}\}, \boldsymbol{\delta}] = \hat{\sigma}^2C^{**}(\mathbf{x}, \mathbf{x}'), \tag{2.14}$$

where $\hat{\sigma}^2 = \frac{\mathrm{F}^T(K^{-1} - K^{-1}H(H^TK^{-1}H)^{-1}H^TK^{-1})\mathrm{F}}{n-p-2}$. The final step is to derive estimates of $\boldsymbol{\delta}$, and then use these estimates as if they were the true values of $\boldsymbol{\delta}$ (Kennedy and O'Hagan, 2001). Integrating out the hyperparameter $\boldsymbol{\delta}$ leads to an intractable posterior distribution for $f(\mathbf{x})$.

The addition of a nugget term to the covariance function in the following way

$$\sigma^2 \times \left[r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\delta}) + \tau^2 \mathbb{1}\{\mathbf{x} = \mathbf{x}'\}\right]$$

does not affect the posterior predictive equations for $f(\mathbf{x})$ with the non-informative prior specification for $\boldsymbol{\beta}$ and σ^2 (De Oliveira, 2007).

For mathematical tractability, Oakley and O'Hagan (2002) suggest using the conjugate prior for $\boldsymbol{\beta}$ and σ^2 in place of (2.11)

$$\pi(\boldsymbol{\beta}, \sigma^2) \propto \sigma^{-(p+r+2)/2} \exp\Big[- \{(\boldsymbol{\beta} - \boldsymbol{z})^T V^{-1} (\boldsymbol{\beta} - \boldsymbol{z}) + a\}/2\sigma^2 \Big], \tag{2.15}$$

where $r, \boldsymbol{z}, V$ and a are the parameters of this distribution and have to be user-specified. This allows users to incorporate prior knowledge about f into the model, which could be done via the elicitation of priors (Oakley, 2002). The inclusion of these parameters changes the form of the posterior distribution for f to

$$\frac{f(\mathbf{x}) - m^{**}(\mathbf{x})}{\hat{\sigma}\sqrt{C^{**}(\mathbf{x}, \mathbf{x})}} \sim t_{p+n}, \tag{2.16}$$

where $m^{**}(\mathbf{x})$ and $C^{**}(\mathbf{x}, \mathbf{x}')$ were previously defined in equations (2.12) and (2.13) respectively with

$$\begin{aligned}
V^* &= (V^{-1} + H^T K^{-1} H)^{-1}, \\
\hat{\boldsymbol{\beta}} &= V^*(V^{-1}\boldsymbol{z} + H^T K^{-1} \mathrm{F}), \\
\sigma^2 &= \frac{a + \boldsymbol{z}^T V^{-1} \boldsymbol{z} + \mathrm{F}^T K^{-1} \mathrm{F} - \hat{\boldsymbol{\beta}}^T (V^*)^{-1} \hat{\boldsymbol{\beta}}}{n + p - 2}.
\end{aligned}$$

Full Bayesian MCMC methods have been used (Higdon et al., 2008; Kaufman and Sain, 2010; Williamson and Blaker, 2014). For these methods, priors for model parameters are specified, and MCMC is used to obtain samples from the posterior distribution for these parameters. The samples from the posterior are drawn and substituted into (2.6) and (2.7) to derive the posterior predictive distribution for f conditional on $\{\mathrm{X}, \mathrm{F}\}$. Despite the method being more computationally expensive

than previously considered approaches to constructing GP emulators, it allows users to incorporate prior knowledge about f into their statistical models.

A number of modifications have been used to reduce the computational costs associated with fitting GP emulators. For instance, approaches such as Local Gaussian Process approximation (LAGP) (Gramacy and Apley, 2015) and Nearest Neighbour Gaussian Process (NNGP) (Datta et al., 2016) generate predictive equations for f locally by deriving smaller design set, $\tilde{\mathrm{X}} \subset \mathrm{X}$, around a point of interest, $\mathbf{x}$, (see section 2.5 for more details), which leads to the reduction in costs of computing covariance matrices of design points. Snelson and Ghahramani (2006) proposed a method for generating a reduced pseudo-input set, $\bar{\mathrm{X}} = (\bar{\mathbf{x}}_1, \ldots, \bar{\mathbf{x}}_M)^T$, where M is smaller than the size of original design set X.

2.3.5 Priors for parameters

We demonstrated in subsection 2.3.4 that prior specification of GP parameters is crucial since it affects the form of the posterior distribution for $f(\mathbf{x})$. Berger et al. (2001) and Paulo et al. (2005) reviewed different types of objective priors such as Jeffreys-rule, independence Jeffreys and reference priors for Gaussian Process models that lead to a proper posterior distribution for parameters under a number of assumptions. In particular, they specify a general form of prior on parameters $\Theta = \{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}\}$

$$\pi(\Theta) \propto \frac{\pi(\boldsymbol{\delta})}{(\sigma^2)^a}$$

for different a and $\pi(\boldsymbol{\delta})$. By specifying different values of a and $\pi(\boldsymbol{\delta})$, they derive Jeffreys-rule, independence Jeffreys and reference priors for Θ. Objective priors are mainly used as default priors, in situations when we are dealing with little or no prior information about the model parameters (Paulo et al., 2005).

However, objective priors should be used in the GP model with caution. Gelman et al. (2017) demonstrated how default priors (uniform priors, Jeffreys priors, reference priors) could produce a poor predictive posterior performance for a number of examples. They encourage users to specify prior distributions for model parameters in the context of the problem and data. In Chapter 4, we will demonstrate the ef-

fect of prior distributions on the inference and predictions generated by a statistical model.

One of the approaches in specifying subjective priors is via elicitation. Elicitation of expert knowledge is the process of constructing a distribution from the finite number of expert statements about the process of interest (O'Hagan, 1998; Oakley and O'Hagan, 2007; Gosling et al., 2007). The expert knowledge elicitation involves two agents, i.e. an expert/experts, and a facilitator (decision maker) who attempts to extract relevant knowledge from experts about the quantity of interest. In particular, Gaussian Process models are used for nonparametric elicitation. The central assumption is that the analyst's prior belief about the process under consideration, $f(\mathbf{x})$, is represented by a Gaussian Process, and the elicitation process is viewed as an inference problem (Oakley and O'Hagan, 2007; Gosling et al., 2007). In this case, the process of elicitation is to ask experts for probability statements about the observed quantity $f(\mathbf{x})$ at $\mathbf{x}$ and then update the facilitator's prior beliefs about the form of expert's probability distribution in the light of these expert's judgements. The prior specification for $f(\cdot)$ depends on the smoothness parameter δ, variance parameter σ^2 and the form of underlying density $g(\cdot)$. It is important to note that the facilitator is required to specify GP model hyperparameters before anything is elicited from the expert. For instance, the facilitator's prior beliefs about both the smoothness of $f(\cdot)$ and how far $f(\cdot)$ is expected to deviate from $g(\cdot)$ could guide the prior specification for δ and σ^2 (Gosling et al., 2007).

Gosling et al. (2007) have performed thorough studies to derive the proper priors for δ and σ^2 by considering the effect of different values for σ^2 and δ on the facilitator's distribution $f(\cdot)$. In particular, for each pair selected from (δ, σ^2)-plane five hundred functions are simulated from the prior distribution for $f(\cdot)$ with parameters of $g(\cdot)$ fixed at arbitrary values. These simulated functions are effectively used to discover the values of σ^2 and δ that generate the simulated functions consistent with the facilitator's prior beliefs about $f(\cdot)$. Such beliefs could be, for instance, about the sign of $f(\cdot)$ and/or about the similarity of $f(\cdot)$ to an underlying density $g(\cdot)$. This approach has resemblance to the studies of generative priors considered

by Gelman et al. (2017); Gabry et al. (2019) and performed in Chapter 4.

Subjective priors on correlation function parameters have been used in UQ (Higdon et al., 2008; Williamson and Blaker, 2014; Kaufman and Sain, 2010). The parameters of the correlation function, such as correlation length parameters, $\boldsymbol{\delta}$, affect the amount of information being learned from the neighbouring points in the input space by varying the smoothness properties of GP paths being produced (Santner et al., 2003). Higdon et al. (2008) specified a correlation function for a GP model

$$r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\rho}) = \prod_{i=1}^{p} \rho_i^{4(x_i - x_i')},$$

where parameter ρ_i is the correlation between outputs evaluated at inputs that vary in only the ith dimension by their half domain. They proceed with specifying independent $Beta(a_i, b_i)$ priors for ρ_i. In particular, only a subset of inputs is expected to influence the simulator response, and the prior specification for ρ_i is used to reflect this expectation. Under the model definition, input i is inactive if $\rho_i = 1$ and as a result, it is advised to choose beta prior with $a_i = 1$ and $0 < b_i < 1$, which will produce a density with substantial mass near 1. Williamson and Blaker (2014) also considered the separation of input parameters into active and less active. Less active input parameters are parameters for which only linear terms were included in the mean function, $h(\mathbf{x})^T \boldsymbol{\beta}$.

The separable exponential correlation function for the Gaussian Process was chosen

$$r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\kappa}) = \prod_{i=1}^{p} \exp\big\{ -\kappa_i |x_i - x_i'|^{1.9} \big\},$$

where instead of correlation length parameter vector, $\boldsymbol{\delta}$, the roughness parameter vector, $\boldsymbol{\kappa}$, is used. A half length correlation, defined as $s_j = \exp\big\{-\kappa_j / 2^{1.9}\big\}$, is used to specify the prior distribution over the roughness parameter. This transformation allowed Williamson and Blaker (2014) to operate within a reduced range $[0, 1]$. Williamson and Blaker (2014) proposed to use two forms of prior for s_j. In particular, $Beta(1, 1.9)$ is chosen for half length correlation that corresponds to active input parameter, leading to more weight being placed towards shorter roughness param-

eters and increasing the correlation between points in the input space. For input variables that were considered to be relatively inactive in the model, $Beta(1, 4.5)$ prior was chosen, leading to a negatively skewed distribution, concentrated on small values. As a result, more weight was given to longer roughness parameters. Figure 2.1 demonstrates the prior distributions on the half length correlations as well as roughness length parameters as histograms.

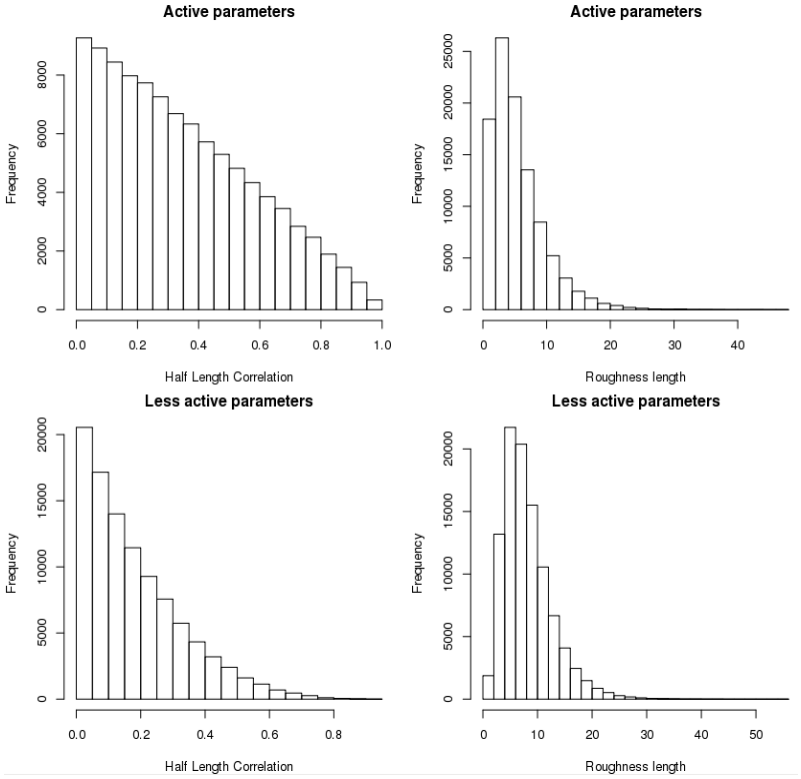

Figure 2.1: Histograms for the half length distributions and the corresponding roughness length distributions specified by $Beta(1, 1.9)$ (*top*) and $Beta(1, 4.5)$ (*bottom*).

The subjective prior specification for correlation function parameters described above are mainly used to express our beliefs about the role of input parameters in the model. As a result, we are required to identify which input parameters are active and less active before specifying our priors. Also, the prior specification is largely dependent on the form of parameters that we are using in our correlation function. However, we are dealing with complex statistical models, for which priors will have a significant effect on inference, and it is worth the extra effort to carefully consider our priors' specification for parameters (Gelman et al., 2017).

2.4 Diagnostics for Gaussian process emulator

Before using a Gaussian process emulator for inferential tasks such as prediction, calibration, uncertainty analysis or sensitivity analysis, the emulator performance should be validated in terms of its accuracy of representing the computer model response of interest. Model validation methods such as cross-validation (CV) are primarily used to check the predictive performance of a statistical model by dividing the original sample into a design set, used to train (fit) a statistical model, and a validation set used to evaluate the statistical model's performance. For K-fold CV, the original sample is divided into K equal size subsamples. A single subsample is retained as a validation set to evaluate the performance of the statistical model, and the remaining K-1 subsamples together make up the design set. The Leave-One-Out cross-validation (LOO-CV) is a special case of K-fold CV, where K equals to the size of the sample.

We define a validation data set, $\mathrm{X}^* = (\mathbf{x}_1^*, \cdots, \mathbf{x}_m^*)$, which is not used when training the GP model, and produce runs of the computer model for the validation data set, $\mathrm{F}^* = (f(\mathbf{x}_1^*), \cdots, f(\mathbf{x}_m^*))$. We also define $\mathrm{X}_{-i} = (\mathbf{x}_1, \cdots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_n)$ and $\mathrm{F}_{-i} = (f(\mathbf{x}_1), \cdots, f(\mathbf{x}_{i-1}), f(\mathbf{x}_{i+1}), \cdots, f(\mathbf{x}_n))$ for LOO-CV.

Residuals and standardized residuals are commonly used in diagnostic calculations, as well as graphical displays, to investigate the validity of adopted assumptions and the accuracy of emulator predictions (Bastos and O'Hagan, 2009). At the diagnostic stage we assume that we have constructed a GP emulator and derived posterior predictive distribution for $f(\cdot)$ conditioned on ensemble, $\{\mathrm{X}, \mathrm{F}\}$ with mean function, $E[f(\cdot)|\{\mathrm{X}, \mathrm{F}\}]$, and variance function, $Var[f(\cdot)|\{\mathrm{X}, \mathrm{F}\}]$. We proceed to compute the individual prediction errors (residuals) for the validation data set as, $\big(f(\mathbf{x}_{\boldsymbol{i}}^*) - E[f(\mathbf{x}_i^*)|\{\mathrm{X}, \mathrm{F}\}]\big)$ for $i = 1, \ldots, m$, as well as the standardized prediction errors (standardized residuals), i.e.

$$\frac{f(\mathbf{x}_i^*) - E[f(\mathbf{x}_i^*)|\{\mathrm{X}, \mathrm{F}\}]}{\sqrt{Var[f(\mathbf{x}_i^*)|\{\mathrm{X}, \mathrm{F}\}]}}. \tag{2.17}$$

We perform similar computations for X_{-i} with $i = 1, \ldots, n$ using virtual LOO

formulas (Ripley, 2005; Bachoc, 2013) for expectation and variance, i.e.

$$E[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}] = f(\mathbf{x}_i) - \frac{\left[K^{-1}\left(\mathrm{F} - H\boldsymbol{\beta}\right)\right]_i}{(K^{-1})_{i,i}} \tag{2.18}$$

$$Var[f(\mathbf{x}_i)|\{\mathrm{X}_{-1}, \mathrm{F}_{-i}\}] = \frac{1}{\left(K^{-1}\right)_{i,i}}. \tag{2.19}$$

Bachoc (2013) explicitly used these formulae to compare the performance of zero-mean GP models, with parameters σ^2 and $\boldsymbol{\delta}$ being estimated via maximum likelihood and cross-validation approaches. However, we do not find any arguments against using posterior samples for GP hyperparameters $\Theta = \{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}\}$ drawn from $\pi\big(\Theta|\{\mathrm{X}, \mathrm{F}\}\big)$ to compute equations (2.18) and (2.19). Throughout this thesis we used RStan to perform Bayesian emulation (Stan Development Team, 2017) (see Chapter 3 for more information). We propose to sample from the obtained set of posterior simulations of GP hyperparameters, $\Theta_j = \{\boldsymbol{\beta}_j, \sigma_j^2, \boldsymbol{\delta}_j, \tau_j^2\}, j = 1, \ldots, M$ and compute at point $\mathbf{x}_i$, the expectation $E[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}, \Theta_j]$ and variance $Var[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}, \Theta_j]$. We proceed to simulate $f^{(j)}(\mathbf{x}_i), j = 1, \ldots, M$ from a normal distribution with computed above expectation and variance to obtain the distribution of simulated values and compute the sample mean and sample standard deviation. In particular, we use a sample mean and sample variance in place of $E[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}]$ and $Var[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}]$.

Similar to residuals from validation set, we could derive individual prediction errors (residuals) for X_{-i} as $\big(f(\mathbf{x}_i) - E[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}]\big)$ and standardized prediction errors (standardized residuals) for X_{-i},

$$\frac{f(\mathbf{x}_i) - E[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}]}{\sqrt{Var[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}]}}. \tag{2.20}$$

Bastos and O'Hagan (2009) described a number of possible problems encountered with Gaussian Process emulators, such as poor estimates of statistical model parameters and inappropriate choices of mean and covariance structures. Gaussian Process models could be considered as a special case of a linear model with dependent errors imposed by a correlated residual term $\epsilon(\mathbf{x})$. Bastos and O'Hagan (2009) employed diagnostics used for linear models, such as quantile-quantile plots,

to validate the performance of GP models. The Mahalanobis distance given by

$$D_{MD}(\mathrm{F}^*) = \big(\mathrm{F}^* - E[f(\mathrm{X}^*)|\{\mathrm{X},\mathrm{F}\}]\big)^T \times Var[f(\mathrm{X}^*)|\{\mathrm{X},\mathrm{F}\}]^{-1}\big(\mathrm{F}^* - E[f(\mathrm{X}^*)|\{\mathrm{X},\mathrm{F}\}]\big) \tag{2.21}$$

is used to validate an emulator's performance, allowing users to include the correlation between outputs by using the predictive covariance matrix $Var[f(\mathrm{X}^*)|\{\mathrm{X},\mathrm{F}\}]$, where $Var[f(\mathrm{X}^*)|\{\mathrm{X},\mathrm{F}\}]_{ij} = Cov[f(\mathbf{x}_i^*), f(\mathbf{x}_j^*)|\{\mathrm{X},\mathrm{F}\}]$ with $1 \leq i,j \leq n$, instead of a diagonal matrix of predictive variance for $f(\mathrm{X}^*)$ (Bastos and O'Hagan, 2009).

Extremely large or small values of this diagnostic indicate a conflict between the emulator and simulator. The distribution of $D_{MD}(\mathrm{F}^*)$ is a chi-squared distribution with m degrees of freedom, where m is a number of points in the validation data set. Therefore the observed Mahalanobis distance value could be compared to values from its theoretical distribution (Bastos and O'Hagan, 2009).

Particular attention should be paid to covariance function misspecification for GP models, which in practice could occur because of bad estimates of the covariance function hyperparameters, or because of strong assumptions imposed by the covariance function choice such as stationarity or isotropcy (see subsection 2.3.2 for more details) (Stein, 1988, 1990b,a). An example of a misspecified covariance function is the stationary covariance function, described in subsection 2.3.2, defined for a GP emulator to model a nonstationary or discontinuous response. We consider a nonstationary response as the one for which its behaviour changes significantly across the input space and cannot be depicted by GP emulator with constant or linear mean and stationary covariance structure such as in the example demonstrated in Figure 2.2.

Cressie (1993) introduced a criterion to validate covariance function specification for a GP model

$$C_{LOO} = \frac{1}{n}\sum_{i=1}^{n}\frac{(f(\mathbf{x}_i) - E[f(\mathbf{x}_i)|\{\mathrm{X}_{-i},\mathrm{F}_{-i}\}])^2}{Var[f(\mathbf{x}_i)|\{\mathrm{X}_{-i},\mathrm{F}_{-i}\}]}, \tag{2.22}$$

which should be close to 1. To derive C_{LOO} we effectively compute the sum of squares

of n standardized residuals (errors), that are independent and normally distributed with mean 0 and standard deviation 1. As a result, the C_{LOO} statistic is chi-squared distributed with n degrees of freedom. Scaling the chi-squared distribution by its degrees of freedom results in a modified chi-squared distribution with the expected value 1, which is why we favour C_{LOO} close to 1.

Graphical methods are commonly used to evaluate the performance of emulators. Plots such as those showing residuals against the emulator's predictions, quantile-quantile plots and plots of residuals against the inputs are typically used (Bastos and O'Hagan, 2009; Montagna and Tokdar, 2016). In particular, plots of the predictive posterior mean together with prediction intervals, e.g. two standard deviation prediction intervals, for both the validation data set, X^*, and the design data set, X, could be produced together with standardized residual plots against the inputs to validate the covariance structure specified for a GP model.

Figure 2.2 demonstrates the failure of a GP emulator with a stationary covariance structure to model $f(x)$ with a sharp localised feature at $x = 0$, as considered by Montagna and Tokdar (2016). From the left panel of Figure 2.2, we observe narrow prediction intervals being produced by the emulator around the peak (the emulator is over-confident). On the contrary, prediction intervals are large in the region far from the peak and where f is well-behaved (the emulator is under-confident). The information from the right panel of Figure 2.2 is used to produce a plot of individual standardized errors of emulator predictions against x using equation (2.20). To pass the validation check, we would expect to observe a horizontal band between -2 and 2, due to the standard normality of individual standardized errors. However, the right panel of Figure 2.2 demonstrates an increase in the error variability around the region when f has a tall peak. Therefore, we conclude that a GP model with linear mean and stationary covariance should not be used to analyse and perform inference on model in this example.

We proceed to discuss in detail in Section 2.5 a range of methods for constructing GPs if stationary GPs fail some of the diagnostics mentioned above.

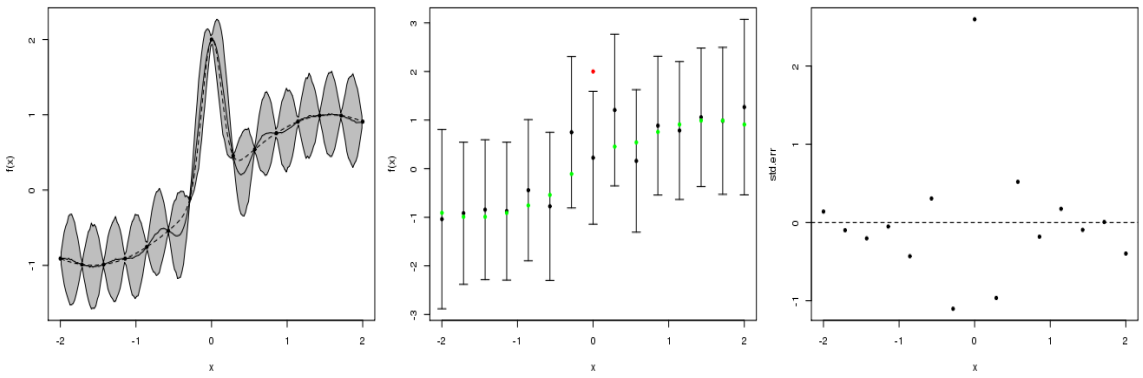

Figure 2.2: *Left panel*: plot of function $f(x) = \sin(x) + 2\exp(-30x^2), x \in [-2, 2]$ (dashed line). The black points, 15 equally spaced values of x, are the design points used to train the GP model. The solid line is the posterior predictive mean of f obtained from a GP emulator with stationary covariance function. The shaded area denotes two standard deviation prediction intervals. *Central panel*: Leave One Out diagnostic plot against x. The predictions and two standard deviation prediction intervals are in black. The true function values are in green if they lie within two standard deviation prediction intervals, or red otherwise. *Right panel*: Individual standardized errors of emulator predictions against x.

2.5 Nonstationary Gaussian Process Models

It is common to encounter heterogeneous and discontinuous responses in spatial model outputs and computer model experiments. For example, Paciorek and Schervish (2006) attempted to model climatological data, in particular the annual precipitation for the state of Colorado in the United States, for which Figure 2.3 is directly adapted from Paciorek and Schervish (2006). The left panel plot in Figure 2.3 demonstrates the topography of the state of Colorado, with a mountainous western region and flat eastern region. The right panel plot shows the values of log-transformed annual precipitation observations over the state in 1981. The blue (red) coloured points correspond to the lower (larger) values of observed precipitation on a log scale. From the right panel plot in Figure 2.3, we observe that the variability in the log-transformed annual precipitation increases in the mountainous region in comparison with the flat plain region.

Another famous example in the computer experiments literature is a computational fluid dynamics simulator of the Langley glide-back booster (LGBB), a proposed rocket booster at NASA (Gramacy and Lee, 2008; Montagna and Tokdar, 2016; Marmin et al., 2018). In particular, there is an interest to examine the lift

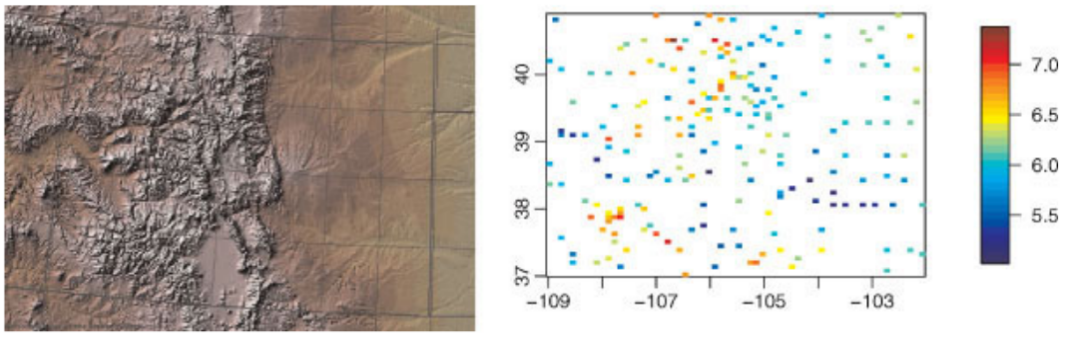

Figure 2.3: Reprinted from Paciorek and Schervish (2006)[p.489]. Copyright (2006) John Wiley & Sons, Ltd. *Left panel*: Topography of Colorado, with thicker line indicating state boundary, and *right panel*: image plot of log-transformed annual precipitation observations in 1981, with the color of the box indicating the magnitude of the observation.

force modelled as a function of the speed of the vehicle at reentry (measured by Mach number), the angle of attack (the alpha angle) and the sideslip angle (the beta angle). The beta angle is quantized and run only at 6 particular levels. Montagna and Tokdar (2016) were interested in examining the lift response as a function of Mach and alpha with the beta angle fixed at zero. Figure 2.4, directly adopted from Montagna and Tokdar (2016), demonstrates that the lift variable is relatively constant for large values of Mach. However, we observe heterogeneity in response near the Mach=1. Interestingly, the ridge in response at Mach=1 has a physical meaning, separating subsonic flows and supersonic flows (Montagna and Tokdar, 2016).

GP models with simple, i.e. constant or linear mean functions, and stationary covariance functions fail to capture the unusual behaviour in the model response for these two examples (see subsection 2.4). There are a number of approaches that adapt GPs to model nonstationary responses both in spatial statistics and computer experiments.

Spatial deformation approaches include Sampson and Guttorp (1992), Schmidt and O'Hagan (2003), Montagna and Tokdar (2016) and Marmin et al. (2018). The first approaches to model nonstationarity via spatial deformation appeared in geostatistics. The main idea is to apply a non-linear transformation from the original, "geographic" space, denoted by G, into a "transformed" space, denoted by D, where

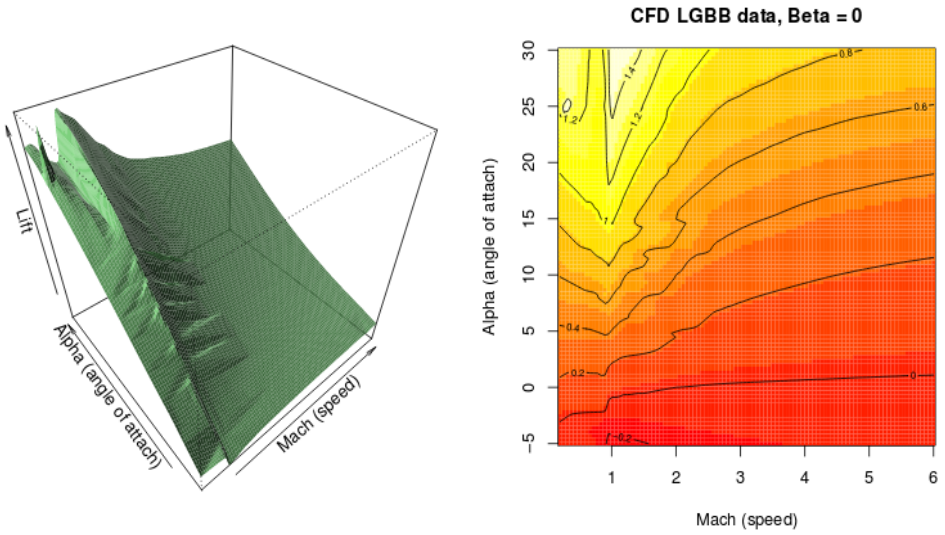

Figure 2.4: Reprinted from Montagna and Tokdar (2016)[Appendix]. Copyright by SIAM and ASA. Interpolated lift surface plotted as a function of Mach (speed) and Alpha (angle of attack) with Beta (side-slip angle) fixed at zero.

the spatial structure is stationary and isotropic. Sampson and Guttorp (1992) proposed to combine a GP model with warping function. In particular, a warping function $T(\cdot)$ is a function from G to D applied to $\mathbf{x} \in G$. The covariance of f at two input points $\mathbf{x}, \mathbf{x}' \in G$ is defined as

$$Cov[f(\mathbf{x}), f(\mathbf{x}')] = k\big(T(\mathbf{x}), T(\mathbf{x}')\big),$$

where $k(\cdot, \cdot)$ is a stationary covariance function. Sampson and Guttorp (1992) proposed to use a non-linear $T(\cdot)$ such as multidimensional scaling (MDS) to accommodate the nonstationary structure in the model. A number of issues have been identified with this approach, such as the fact that the mapping $T(\cdot)$ does not guarantee the prediction of points of interest to be bijective, i.e. the mapping could produce a surface that folds back on itself (Schmidt and O'Hagan, 2003). Schmidt and O'Hagan (2003) adopted a Bayesian approach by specifying a Gaussian process prior distribution for the mapping. This approach handles the mapping of both the measured and unmeasured sites in a single step and takes into account uncertainty in the mapping during the predictive inference. However, it is computationally more expensive, as a prior specification is required at each location, and Markov chain

Monte Carlo (MCMC) methods are used to obtain samples from the posterior distributions. The spatial deformation approach to model nonstationarity has been recently used in the computer experiment literature. Montagna and Tokdar (2016) proposed to include a latent input Z into the residual term $\epsilon([\mathbf{x}, Z])$ in equation (2.1), which is a zero-mean GP with covariance function:

$$k[(\mathbf{x}, Z), (\mathbf{x}', Z'); \sigma^2, \boldsymbol{\phi}] = \sigma^2 \exp\Big\{ - \sum_{i=1}^{p} \phi_i (x_i - x_i')^2 - \phi_{p+1}(Z - Z')^2 \Big\}.$$

Interestingly, though the extended process $f(\mathbf{x}, Z)$ is modelled to be stationary, the marginal process $f(\mathbf{x})$ is nonstationary. Similar to the Schmidt and O'Hagan (2003) approach, the latent input, Z, is modelled as a continuous function of the inputs, $Z = g(\mathbf{x})$, using a stationary GP:

$$g|\tilde{\boldsymbol{\phi}} \sim GP(\mathbf{0}, \tilde{K}),$$

where $\tilde{K}_{ij} = \exp\Big\{ - \sum_{i=1}^{p} \tilde{\phi}_i (x_i - x_j)^2 \Big\}$, and the scale parameter of covariance function, σ^2, is fixed at 1. Montagna and Tokdar (2016) demonstrated satisfactory performance of a modified GP emulator of this type for computer models that are characterized by sharp local features. The latent input Z is used to stretch the distance between two points at and about the localized feature, and as a result weakens the correlation between these two points. Marmin et al. (2018) presented Warped Gaussian Processes with a WaMI kernel (Warped Multiple Index) of the form:

$$k(\mathbf{x}, \mathbf{x}') = k(T(A\mathbf{x}), T(A\mathbf{x}'); \boldsymbol{\theta}),$$

where A is a $q \times p$ matrix, with $q < p$ and $k(\cdot, \cdot; \boldsymbol{\theta})$ is a covariance kernel on $\mathbb{R}^q$ parameterized by $\boldsymbol{\theta}$. The multiplication by A leads to dimension reduction, and the warping functions $T(\boldsymbol{u}) = \Big(T_i(u_i; \boldsymbol{\phi}_i)\Big)\Big)_{1 \le i \le q}$ are applied to the points in the modified space. The warping function allows users to capture the nonstationarity in the response along the noncanonical directions. Despite the satisfactory performance of the method on a number of examples, it is not precisely clear how to choose A and the form of $T(\boldsymbol{u})$. Also, the parameters of the covariance function have

been estimated by maximum likelihood. Marmin et al. (2018) pointed out that the Bayesian approach or more efficient algorithms could be used instead.

Several approaches aim to model large computer experiments and provide the flexibility to model nonstationarity in response as well. For instance, Snelson and Ghahramani (2006) introduced Sparse Pseudo-input Gaussian Processes (SPGPs) by defining pseudo-inputs (pseudo-design), denoted by $\bar{\mathrm{X}} = (\bar{\mathbf{x}}_1, \ldots, \bar{\mathbf{x}}_M)^T$ with $M < n$, where n is the size of the original design set, X. Snelson and Ghahramani (2006) were interested in obtaining a reduced set of pseudo-inputs, $\bar{\mathrm{X}}$, and then redefining the predictive distribution for $f(\mathbf{x})$ parameterized by $\bar{\mathrm{X}}$. They started by specifying a model for the pseudo targets, $\bar{\boldsymbol{f}} = (\bar{f}_1, \ldots, \bar{f}_M)$, as zero-mean Gaussian with $M \times M$ covariance matrix, $K_M = k(\bar{\mathrm{X}}, \bar{\mathrm{X}})$. They obtained a modified predictive distribution for $f(\mathbf{x})$ given the original ensemble $\{\mathrm{X}, \mathrm{F}\}$, pseudo-inputs $\bar{\mathrm{X}}$ and parameters $\sigma^2, \boldsymbol{\delta}$, which is Normal with mean

$$E[f(\mathbf{x})|\{\mathrm{X}, \mathrm{F}\}, \bar{\mathrm{X}}, \sigma^2, \boldsymbol{\delta}] = k(\mathbf{x}, \bar{\mathrm{X}})\Big[K_M + k(\bar{\mathrm{X}}, \mathrm{X})\tilde{K}^{-1}k(\mathrm{X}, \bar{\mathrm{X}})\Big]^{-1}k(\bar{\mathrm{X}}, \mathrm{X})\tilde{K}^{-1}\mathrm{F}$$

and variance

$$Var[f(\mathbf{x})|\{\mathrm{X}, \mathrm{F}\}, \bar{\mathrm{X}}, \sigma^2, \boldsymbol{\delta}] = k(\mathbf{x}, \mathbf{x}) - k(\mathbf{x}, \bar{\mathrm{X}})\Big[K_M^{-1} - \big(K_M + k(\bar{\mathrm{X}}, \mathrm{X})\tilde{K}^{-1}k(\mathrm{X}, \bar{\mathrm{X}})\big)^{-1}\Big]^{-1} \times k(\bar{\mathrm{X}}, \mathbf{x}).$$

In the expressions for both the mean and variance, $\tilde{K}$ is $n \times n$ diagonal matrix with ith entry obtained as

$$\tilde{K}_i = k(\mathbf{x}_i, \mathbf{x}_i) - k(\mathbf{x}_i, \bar{\mathrm{X}})K_M^{-1}k(\bar{\mathrm{X}}, \mathbf{x}_i), \quad i = 1, \ldots, n.$$

Snelson and Ghahramani (2006) demonstrated that the mean could be computed in $\mathcal{O}(M)$, and variance calculated in $\mathcal{O}(M^2)$ from the transformed predictive distribution for $f(\mathbf{x})$. The pseudo-inputs are treated the same way as GP model parameters, and are estimated by maximising the marginal likelihood, $p(\mathrm{F}|\mathrm{X}, \bar{\mathrm{X}}, \sigma^2, \boldsymbol{\delta})$, with respect to $\big\{\bar{\mathrm{X}}, \sigma^2, \boldsymbol{\delta}\big\}$ by gradient descent. SPGP could be used to model nonsta-

tionarity by moving pseudo-inputs to the parts of input space where the computer model response exhibits localized features.

Another approach, Local Gaussian Process Approximation (LAGP) (Gramacy and Apley, 2015) considers a subdesign for predicting a particular point of interest, when the original design is too big for efficient inversion of a covariance matrix K, via greedy search algorithm by minimizing mean-squared predictive error. In particular, given $\mathbf{x}$ and $\mathrm{D}_j = \big\{\mathrm{X}_j = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_j), \mathrm{F}_j = (f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_j))\big\}$, where $j = n_0, \ldots, n$, the current sub-ensemble at step j, a new point $\mathbf{x}_{j+1}$ is added to the sub-design by considering its impact on the variance of $f(\mathbf{x})$ through mean-square prediction error (MSPE) criterion, i.e.

$$J(\mathbf{x}_{j+1}, \mathbf{x}) = E\big\{[f(\mathbf{x}) - \mu_{j+1}(\mathbf{x})]^2\big\},$$

which should be minimized. The posterior predictive mean at step $j+1$ has the following form: $\mu_{j+1}(\mathbf{x}) = E[f(\mathbf{x})|\mathrm{D}_j \cup (\mathbf{x}_{j+1}, f(\mathbf{x}_{j+1})), \boldsymbol{\beta}^{[j+1]}, (\sigma^2)^{[j+1]}, \boldsymbol{\delta}^{[j+1]}]$ and parameters $\boldsymbol{\beta}^{[j+1]}, (\sigma^2)^{[j+1]}, \boldsymbol{\delta}^{[j+1]}$ are found via MLE after $(\mathbf{x}_{j+1}, f(\mathbf{x}_{j+1}))$ have been added to the sub-ensemble. The final design ensemble size n is chosen to be as large as computational constraints allow. Interestingly, operating with design locally, relative to $\mathbf{x}$, allows users to deal with nonstationarity in the model response by focusing on local model behaviour.

A common approach in mitigating nonstationarity is to fit a complicated response surface, $h(\mathbf{x})^T\boldsymbol{\beta}$, that captures global nonstationarity, leaving a stationary process residual (Rougier et al., 2009; Vernon et al., 2010; Williamson et al., 2013). If the right functions can be found and added to $h(\mathbf{x})$, they should be used, but, in practice, this can be extremely difficult and can require too much manual fitting in choosing the terms in the regression model.

Ba and Joseph (2012) provided an alternative to fitting a complex mean function with the Composite Gaussian Process (CGP). CGP consists of two processes, precisely

$$f(\mathbf{x}) = Z_{global}(\mathbf{x}) + \sigma(\mathbf{x})Z_{local}(\mathbf{x}). \tag{2.23}$$

They proposed to specify $Z_{global}(\mathbf{x})$ as a GP with a constant mean μ and covariance $\tau^2 r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\phi}_g)$, and $Z_{local}(\mathbf{x})$ as a zero mean GP with covariance $r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\phi}_l)$. The first process is used to model the global trend, whilst the second process captures the local variability around this trend by including a variance model, $\sigma^2(\mathbf{x}) = \sigma^2 v(\mathbf{x})$, where σ^2 is a variance term, and $v(\mathbf{x})$ is a standardized volatility function, which fluctuates around 1. Both the form and the importance of standardized volatility function is discussed in subsection 4.2. Both $r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\phi}_g)$ and $r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\phi}_l)$ are Gaussian correlation functions:

$$r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\phi}_g) = \exp\Big(- \sum_{i=1}^{p} \phi_{gi}(x_i - x_i')^2 \Big), \quad r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\phi}_l) = \exp\Big(- \sum_{i=1}^{p} \phi_{li}(x_i - x_i')^2 \Big)$$

with their own vectors of correlation parameters, $\boldsymbol{\phi}_g = (\phi_{g1}, \ldots, \phi_{gp})$, and, $\boldsymbol{\phi}_l = (\phi_{l1}, \ldots, \phi_{lp})$, that satisfy $\mathbf{0} \leq \boldsymbol{\phi}_g \leq \boldsymbol{\alpha}$ and $\boldsymbol{\alpha} \leq \boldsymbol{\phi}_l$. The bounds, $\boldsymbol{\alpha}_l = (\alpha_{l1}, \ldots, \alpha_{lp})$, are set to be moderately large to ensure that the global component of the model is smoother than the local one. The final model specification is

$$f(\mathbf{x}) \sim GP\Big(\mu, \tau^2 r(\cdot, \cdot; \boldsymbol{\phi}_g) + \sigma^2 v(\mathbf{x}) r(\cdot, \cdot; \boldsymbol{\phi}_l)\Big),$$

which also could be considered as a GP model with a flexible covariance structure.

Another approach to modelling nonstationarity is through modifications in covariance function, which are common in spatial statistics (Higdon et al., 1998; Fuentes, 2001; Fuentes and Smith, 2001; Banerjee et al., 2004; Paciorek and Schervish, 2006). In subsection 2.3.2, we discussed nonstationary covariance functions that can be obtained by convolving spatially-varying kernel functions. Higdon et al. (1998) derived a nonstationary version of the squared exponential stationary covariance function,

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

with

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

where $\Sigma(\mathbf{x})$ is a covariance matrix of a Gaussian kernel centred at $\mathbf{x}$, and $|\cdot|$ denotes the determinant of a matrix. We observe that instead of using a spatially invariant matrix Σ, the kernel matrices for $\mathbf{x}$ and $\mathbf{x}'$ are averaged, which generates a spatially varying covariance function (Paciorek and Schervish, 2006). The extensions to other families of covariance functions were derived by Paciorek and Schervish (2006). For example, a nonstationary version of a Matérn covariance function is defined as

$$k(\mathbf{x},\mathbf{x}') = \sigma^2 \frac{2^{1-\alpha}}{\Gamma(\alpha)} |\Sigma(\mathbf{x})|^{\frac{1}{4}} |\Sigma(\mathbf{x}')|^{\frac{1}{4}} \left| \frac{\Sigma(\mathbf{x}) + \Sigma(\mathbf{x}')}{2} \right|^{-\frac{1}{2}} \left(2\sqrt{\alpha Q(\mathbf{x},\mathbf{x}')} \right)^{\alpha} \mathcal{K}_{\alpha} \left(2\sqrt{\alpha Q(\mathbf{x},\mathbf{x}')} \right),$$

and it is easy to derive an exponential covariance function by setting $\alpha = 0.5$. Fuentes (2001), Fuentes and Smith (2001) and Banerjee et al. (2004) developed a kernel mixture approach. They assumed the existence of L stationary processes in different regions of a 2D spatial field. They specified L centroids, i.e. $\mathbf{x}_l, l = 1, \cdots, L$, in that field by applying a rectangle-partitioning method (Banerjee et al., 2004). Together with a weight function determined by the distance between the point of interest, $\mathbf{x}$, and the centroids, $w(\mathbf{x}, \mathbf{x}_l)$, they specify a nonstationary kernel for the whole input space as the weighted sum of L stationary region-specific covariance kernels. The nonstationary kernel has the following form

$$k(\mathbf{x},\mathbf{x}';\boldsymbol{\sigma}^2,\boldsymbol{\delta}) = \sum_{l=1}^{L} w(\mathbf{x},\mathbf{x}_l) w(\mathbf{x}',\mathbf{x}_l) \sigma_l^2 r_l(\mathbf{x},\mathbf{x}';\boldsymbol{\delta}_l), \tag{2.24}$$

where $\boldsymbol{\delta}_l$ is a vector of length 2, as only spatial variability is included, and σ_l^2 corresponds to parameters of the region specific covariance kernel. This specification fits a single process with a covariance kernel that varies spatially. These methods rely on the response variability with respect to the location in two-dimensional space, and have not yet been implemented for computer experiments where we deal with $p > 2$ parameter inputs. For example, it is challenging to adapt the distance-based weight function, as the definition of distance changes as we increase the number of inputs, i.e. some inputs are less active than others and could be omitted from the calculation of the weight function.

Alternatively, several works in UQ literature use piecewise GPs to model non-

stationarity, and these methods are discussed in detail in Chapter 3. The basic idea behind piecewise GPs is to partition the input space into a number of different input regions and then fit individual and independent base models such as GPs to partitioned design. Gramacy and Lee (2008) developed a treed partition model, using a decision tree, to divide the input space, $\mathcal{X}$, by making binary splits on the value of single variables recursively. The Bayesian approach is applied to the partitioning process by specifying a prior through a tree-generating process, and the tree is averaged out by integrating over possible trees using Reversible Jump MCMC (RJ-MCMC). In each leaf of the tree, a stationary Gaussian Process is fitted. Partitioning does not guarantee continuity of the fitted function, because the posterior predictive surface conditional on a particular tree is discontinuous across the partition boundaries, which translates into higher posterior predictive uncertainty near region boundaries. However, Bayesian model averaging provides mean fitted functions that are relatively smooth in practice. Pope et al. (2018) provided an extension to classification trees for modelling discontinuities in the model response. In particular, Voronoi tessellation is applied to partition the input space and provides users with a flexibility to obtain non-convex and disconnected regions. These regions consist of a number of cells, and Pope et al. (2018) removed the requirement for the Voronoi cells to share a vertex to be in the same region and centres of cells to be part of design set X. These assumptions were adopted in spatial statistics by Kim et al. (2005). Similar to TGP, RJ-MCMC is used for implementation, which requires a larger number of MCMC chains and iterations per chain to ensure the convergence.

We have observed an increasing trend to employ "deep" approaches to model nonstationarity, that we are going to discuss briefly in this Section. Damianou and Lawrence (2013) proposed to specify a Gaussian process prior over inputs to another Gaussian process. They considered a sequence of conditionally Gaussian functions $\{f_q\}$ defined as

$$f_{q+1}|f_q \sim GP\big(m(\mathbf{x}; f_q), k(\mathbf{x}, \mathbf{x}'; f_q)\big).$$

The number of Gaussian functions in a sequence $\{f_q\}$ determines the number of

layers and Damianou and Lawrence (2013) demonstrated a flexibility of Deep GPs with 5 layers in modelling complex digit images. However, as the number of layers increases, this leads to an increase in the computational complexity of a model. Dunlop et al. (2018) derived that fewer layers of deep GPs are sufficient to model response with discontinuity and inhomogeneity in one- and two-dimensions. Roininen et al. (2018) presented a shallow-layer GPs, where a GP prior is specified for a precision parameter, the inverse of a correlation length parameter. An explicit link between GPs and Stochastic Partial Differential equations (SPDE) is also used to obtain computational efficiency. In particular, this connection allows users to employ numerical algorithms for sparse matrices, and as a result leads to the simplification of matrices inversion (Lindgren et al., 2011). However, these approaches require a large number of points in the design; for instance Roininen et al. (2018) used 81 design points to construct a shallow-layer deep GPs model for a one-dimensional example. In this thesis, we mainly operate with complex computer models such as climate models, for which a large training (design) set is unattainable. These models are typically high-dimensional, which means that training data will still be sparse in the input space.

2.6 Calibration

Previously, we have discussed the process of building a GP emulator for $f(\mathbf{x})$. GP emulators for computer model output can be used to perform inference about $f(\mathbf{x})$. In particular, we have discussed the importance of computer models in representing physical processes of interest in subsection 2.1. In some cases, a small number of physical observations could be available to ensure the consistency between the simulator output and the physical process of interest. In particular, input parameters of the computer model are adjusted to achieve consistency between the computer model and observations of the system represented by $f(\mathbf{x})$ (Kennedy and O'Hagan, 2001; Rougier, 2007; Jackson et al., 2008). The process of finding input parameter values for which the computer model is consistent with observations is called the inverse problem (calibration). It is crucial if we are planning on using a computer

model for predictions or any other inferential tasks. For example, hydrological models can be used to predict the level of discharge from streams after rainfall, and measurements of discharge of actual streams can be used to learn about the values of input parameters such as average effective transmissivity of the soil and the level of subsurface drainage (Romanowicz et al., 1994; Kennedy and O'Hagan, 2001).

Calibration has been widely used in a range of fields such as hydrology (Borsuk et al., 2004; Wagener and Gupta, 2005; Renard et al., 2010; Razavi et al., 2012), biology (Van Oijen et al., 2005; Hartig et al.) and climate sciences (Rougier, 2007; Edwards et al., 2011; Sexton et al., 2012; Bellprat et al., 2012; Salter et al., 2018; Chang and Guillas, 2018).

The statistical approach to calibration specifies a mathematical model that links together observations and the computer model response, and attempts to take account explicitly of all the sources of uncertainty arising in calibration and subsequent use of computer models, considered in subsection 2.2 (Kennedy and O'Hagan, 2001; Goldstein and Rougier, 2009).

Kennedy and O'Hagan (2001) distinguished between two groups of inputs, $\mathbf{x} = (\mathbf{x}^{cal}, \mathbf{x}^{con})$, to the computer model. The calibration inputs, $\mathbf{x}^{cal}$, are supposed to take fixed but unknown values for all observations used for calibration and are the inputs that we are interested to learn about. The variable (control) inputs, $\mathbf{x}^{con}$, are assumed to be known for the calibrated model. The output of the computer model is denoted by $f(\mathbf{x}^{con}, \mathbf{x}^{cal})$. Calibration adopts the "best input" approach by assuming the existence of a "best input" setting $\mathbf{x}^*$ for $\mathbf{x}^{cal}$ that represents observations, $\boldsymbol{z} = (z_1, \ldots, z_m)$, faithfully within a specified statistical model (Kennedy and O'Hagan, 2001; Rougier, 2007; Murphy et al., 2009). The calibration data is comprised of m observations $\boldsymbol{z} = (z_1, \ldots, z_m)$ at $\mathrm{X}_{\boldsymbol{z}} = (\mathbf{x}_1^{con}, \ldots, \mathbf{x}_m^{con})$, where z_i is the observation of $y(\mathbf{x}_i^{con})$ for the known variable input $\mathbf{x}_i^{con}$, and the aim of calibration task is to learn $\mathbf{x}^*$ for $\mathbf{x}^{cal}$.

A statistical model that describes the relationship between the computer model $f(\cdot, \cdot)$ and the true process $y(\cdot)$ is given by Kennedy and O'Hagan (2001) as

$$y(\mathbf{x}^{con}) = \rho f(\mathbf{x}^{con}, \mathbf{x}^*) \oplus \eta(\mathbf{x}^{con}), \tag{2.25}$$

where ρ is an unknown regression parameter, $\oplus$ indicates the addition of independent terms, and $\eta(\cdot)$ is a discrepancy term that accounts for the discrepancy between the computer model representation and the physical process of interest. For instance, if $f(\cdot,\cdot)$ is a climate model, then the discrepancy term $\eta(\cdot)$ could account for errors generated due to missing or poorly understood physics, or parameterization schemes in the model (Williamson et al., 2013).

Kennedy and O'Hagan (2001) model the relationship between observation, z_i, true process, $y(\cdot)$, and computer model output, $f(\cdot,\cdot)$, as

$$z_i = y(\mathbf{x}_i^{con}) \oplus e_i = \rho f(\mathbf{x}_i^{con}, \mathbf{x}^*) \oplus \eta(\mathbf{x}_i^{con}) \oplus e_i, \tag{2.26}$$

where e_i is the observation error term with $e_i \sim N(0, \lambda)$.

In addition, there is an assumption that n runs of the computer model, $f(\cdot,\cdot)$, could be generated at $\mathrm{X_F} = \big((\mathbf{x}_1^{'con}, \mathbf{x}_1^{cal}), \ldots, (\mathbf{x}_n^{'con}, \mathbf{x}_n^{cal})\big)$ in order to obtain $\mathrm{F} = (f(\mathbf{x}_1^{'con}, \mathbf{x}_1^{cal}), \cdots, f(\mathbf{x}_n^{'con}, \mathbf{x}_n^{cal}))^T$ with known values of variable and calibration inputs for each run. A set $\boldsymbol{d}^T = (\mathrm{F}^T, \boldsymbol{z}^T)$ is used to perform calibration. Kennedy and O'Hagan (2001) proposed to specify Gaussian prior distributions for $f(\cdot,\cdot)$ and $\eta(\cdot)$, i.e.

$$f(\cdot,\cdot) \sim N\Big(m_1(\cdot,\cdot), C_1\big\{(\cdot,\cdot),(\cdot,\cdot)\big\}\Big) \quad \eta(\cdot) \sim N\Big(m_2(\cdot), C_2(\cdot,\cdot)\Big).$$

They proceed to derive the posterior distribution for the calibration parameters, i.e. $\pi(\mathbf{x}^*|\boldsymbol{d})$, which takes into account of all major sources of uncertainty. The posterior distribution for $\pi(\mathbf{x}^*|\boldsymbol{d})$ is derived by firstly obtaining the posterior for $\pi(\mathbf{x}^*, \eta|\boldsymbol{d})$, and marginalising for $\mathbf{x}^*$. This posterior distribution is used in deriving the posterior distribution for $z(\mathbf{x}^{con})$ conditional on $\mathbf{x}^*$ and $\boldsymbol{d}$, which is used to perform inferences about the value of the system (Salter, 2017).

Rougier (2007) performed calibration for climate models, where only one set of historical observations of the climate were available. As there was only one possible setting of $\mathbf{x}^{con}$, the dependence of equation (2.26) on $\mathbf{x}^{con}$ could be removed to derive the following equation

$$z_i = \rho f(\mathbf{x}^*) \oplus \eta_i \oplus e_i, \quad i = 1, \ldots, m. \tag{2.27}$$

Special attention should be paid to the model discrepancy term $\eta(\cdot)$. Brynjarsdóttir and O'Hagan (2014) demonstrated the importance of the model discrepancy term. In particular, disregarding the model discrepancy term could lead to biased and over-confident calibration parameter estimates and predictions. However, strong and meaningful prior information is required in specifying distributions for $\mathbf{x}^*$ and η to avoid the identifiability issues for $\mathbf{x}^*$ and η, which is a challenging task (Kennedy and O'Hagan, 2001; Rougier, 2007; Brynjarsdóttir and O'Hagan, 2014; Salter et al., 2018).

Alternatively, Wong et al. (2017) presented a frequentist approach to computer model calibration with a non-parametric discrepancy function $\eta(\cdot)$. Contrary to the Bayesian approach, prior distributions on the surrogate model and discrepancy term are replaced with "smoothness" assumptions, and an emphasis is intuitively placed on the ability to predict the observation via cross-validation. Under the frequentist regime, $(\mathbf{x}^*, \eta)$ are treated as unknown quantities to be estimated, and $\mathbf{x}^*$ is chosen as the value that makes the computer model representation, $f(\cdot, \cdot)$, close to the physical process of interest, $y(\cdot)$, and the model discrepancy term, η, is used to account for the remainder. To quantify the major sources of uncertainty in calibration problem, a bootstrap sample of estimates for $\mathbf{x}^*$ and η is used to obtain a bootstrap confidence region for quantities of interest, such as a confidence interval for a prediction of physical process, $y(\cdot)$, at new input, $\mathbf{x}^{con}$.

2.6.1 History matching

History matching is a type of calibration that attempts to find input parameters values to achieve the consistency between observations and computer model representation. History matching was initially presented by Craig et al. (1996) to derive parameter settings for expensive oil well model output to match the observed history of an oil well. Contrary to calibration that tries to identify a probability density function (pdf) for the best input $\mathbf{x}^*$, the goal of history matching is to identify the regions of input space corresponding to acceptable matches, and this is performed by ruling out the implausible regions iteratively in waves. In particular, we are trying to

rule out regions in $\mathcal{X}$ that could not contain $\mathbf{x}^*$ given the uncertainty specification.

History matching has been successfully applied across a range of fields including galaxy formation (Bower et al., 2010; Vernon et al., 2010), oil reservoirs (Craig et al., 1996; Cumming and Goldstein, 2010), HIV transmission (Andrianakis et al., 2015, 2017) and climate (Edwards et al., 2011; McNeall et al., 2013; Williamson and Blaker, 2014; Williamson, 2015; Williamson et al., 2017; Salter et al., 2018)

As previously, the statistical model is specified to represent the relationship between the observation (empirical data) z_i and computer model output $f_i(\cdot)$ as

$$z_i = f_i(\mathbf{x}^*) \oplus \eta_i \oplus e_i,$$

where each term has a subscript i as it is possible to simultaneously history match using multiple model outputs (see subsection 2.6.2). Because we usually operate with computer models that are computationally expensive to run, history matching requires an emulator for $f_i(\mathbf{x})$ to be fitted so that, for any setting of the parameter $\mathbf{x}$, an expectation, $E[f_i(\mathbf{x})]$, and variance, $Var[f_i(\mathbf{x})]$, can be computed from the emulator. However, if a computer model is computationally fast to run, we do not require the emulator, and we could use $f_i(\mathbf{x})$ directly instead of $E[f_i(\mathbf{x})]$ and as a result specify $Var[f_i(\mathbf{x})] = 0$ (Gladstone et al., 2012). The implausibility function is defined as a measure of the distance between the output of a model at $\mathbf{x}$ and observation, z_i, and is used to rule out regions of input space, $\mathcal{X}$, that lead to model output values that are inconsistent with observations (Craig et al., 1996; Vernon et al., 2010; Williamson, 2015). The implausibility function has the following form

$$\mathcal{I}_i(\mathbf{x}) = \frac{|z_i - E[f_i(\mathbf{x})]|}{\sqrt{Var[z_i - E[f_i(\mathbf{x})]]}}. \tag{2.28}$$

For model output $f_i(\cdot)$ and observation z_i, large values of $\mathcal{I}_i(\mathbf{x})$ at any $\mathbf{x}$ imply that, relative to our uncertainty, the predicted output of computer model at $\mathbf{x}$ is very far from where we would expect is to be if $f_i(\mathbf{x})$ were consistent with z_i. A threshold, a, is chosen so that any value of $\mathcal{I}_i(\mathbf{x}) > a$ is deemed implausible. The remaining parameter space is termed as Not Ruled Out Yet (NROY) and defined

as (Williamson, 2015)

$$\mathcal{X}_{NROY} = \{\mathbf{x} \in \mathcal{X} : \mathcal{I}_i(\mathbf{x}) \leq a\}. \tag{2.29}$$

The value of a is often taken to be 3 following the 3 sigma rule (Pukelsheim, 1994), which states that for any unimodal continuous probability distribution, at least 95% of the probability mass is within 3 standard deviations of the mean.

Based on our statistical model, the form of $Var[z_i - E[f_i(\mathbf{x})]]$ could be rewritten as

$$\begin{aligned} Var[z_i - E[f_i(\mathbf{x})]] &= Var[z_i - y_i + y_i - E[f_i(\mathbf{x})]] \\ &= Var[e_i + y_i - f_i(\mathbf{x}) + f_i(\mathbf{x}) - E[f_i(\mathbf{x})]] \\ &= Var[e_i + \eta_i + f_i(\mathbf{x}) - E[f_i(\mathbf{x})]] \\ &= Var[e_i] + Var[\eta_i] + Var[f_i(\mathbf{x}) - E[f_i(\mathbf{x})]] \\ &= Var[e_i] + Var[\eta_i] + Var[f_i(\mathbf{x})], \end{aligned} \tag{2.30}$$

where $Var[e_i]$ is the variance of the observation error and $Var[\eta_i]$ is the model discrepancy variance.

If $\mathcal{I}_i(\mathbf{x})$ is large for some $\mathbf{x}$, we are confident that the model output is too far from the observations given all the uncertainties. However, small values of implausibility function can occur either if the model is close to our observations for $\mathbf{x}$ or when our emulator is extremely uncertain about the model, i.e. $Var[f_i(\mathbf{x})]$ is large.

2.6.2 Multi-dimensional implausibility

There may be more than one output of interest produced by a computer model. In the case of a multivariate simulator with l outputs, we could use a multivariate form of the implausibility function (Craig et al. 1997)

$$\mathcal{I}(\mathbf{x}) = (\boldsymbol{z} - E[\boldsymbol{f}(\mathbf{x})])^T (Var[\boldsymbol{z} - E[\boldsymbol{f}(\mathbf{x})]])^{-1} (\boldsymbol{z} - E[\boldsymbol{f}(\mathbf{x})]), \tag{2.31}$$

and expand the denominator of the function

$$Var[\boldsymbol{z} - E[\boldsymbol{f}(\mathbf{x})]] = Var[\boldsymbol{f}(\mathbf{x})] + Var[\boldsymbol{e}] + Var[\boldsymbol{\eta}],$$

where $\boldsymbol{z}$ and $E[\boldsymbol{f}(\mathbf{x})]$ are vectors of length l, and $Var[\boldsymbol{f}(\mathbf{x})]$, $Var[\boldsymbol{e}]$ and $Var[\boldsymbol{\eta}]$ are covariance matrices of dimension $l \times l$. The multivariate form of the implausibility function is constructed by emulating different components of $\boldsymbol{f}(\mathbf{x})$ jointly to obtain $E[\boldsymbol{f}(\mathbf{x})]$ and $Var[\boldsymbol{f}(\mathbf{x})]$. Covariance matrices for the model discrepancy, $Var[\boldsymbol{\eta}]$, and observation error, $Var[\boldsymbol{e}]$, also must be specified. This new measure takes account of covariance structures for the quantities in the denominator (Craig et al., 1997; Vernon et al., 2010), and could be particularly useful when $\boldsymbol{z}$ is a spatial field (Williamson et al., 2017). In this case, $\mathcal{I}(\mathbf{x})$ is compared to a Chi-squared distribution with l degrees of freedom as a cut-off value. For instance, a high percentile such as 95% or 99% of Chi-squared distribution with l degrees of freedom could be used as a cut-off value, a (Vernon and Goldstein, 2009; Vernon et al., 2010). The NROY space using this measure is defined as

$$\mathcal{X}_{NROY} = \left\{\mathbf{x} \in \mathcal{X} : \mathcal{I}(\mathbf{x}) < \chi^2_{l,0.995}\right\}. \tag{2.32}$$

The intuition to compare $\mathcal{I}(\mathbf{x})$ to a Chi-square distribution with l degrees of freedom comes from the definition of discrepancy calculation in Bayes linear methods (Goldstein and Wooff, 2007):

$$D_{MD}(\boldsymbol{d}) = (\boldsymbol{d} - E[\boldsymbol{D}])^T Var[\boldsymbol{D}]^{-1}(\boldsymbol{d} - E[\boldsymbol{D}]),$$

where $\boldsymbol{d}$ is an observed value of $\boldsymbol{D}$, an l-dimensional random variable, with prior expectation $E[\boldsymbol{D}]$ and prior variance $Var[\boldsymbol{D}]$. The random quantity $\boldsymbol{D}$ is approximately multivariate normal, implying that $D_{MD}(\boldsymbol{d})$ has approximately a Chi-squared distribution with l degrees of freedom. This measure is the same as the Mahalanobis distance used to validate the performance of an emulator against the computer model output, and is discussed in more detail in section 2.4.

The use of a multivariate form of the implausibility function $\mathcal{I}(\mathbf{x})$ in equation

(2.31) has a number of limitations. Firstly, this form of implausibility function does not allow for different cut-off values when assessing a match between the observations and computer model responses. Secondly, this measure also assumes that it is important to match all the outputs of $\boldsymbol{f}(\mathbf{x})$, and as a result a single poorly matching component of $\boldsymbol{f}(\mathbf{x})$ could have a large influence on implausibility measure (Craig et al., 1997).

A simplification to the multivariate form of implausibility function, $\mathcal{I}(\mathbf{x})$, was proposed by Craig et al. (1997) to construct implausibility per each output, $\mathcal{I}_j(\mathbf{x}), j = 1, \ldots, l$, separately, by assuming that outputs are uncorrelated. Removing the covariance structure from the implausibility function calculation significantly reduces the computational costs (Craig et al., 1997). This approach also provides with the flexibility to perform history matching using either all model outputs or a collection of them (Williamson et al., 2017). For instance, Andrianakis et al. (2015) performed history matching in 9 waves (see subsection 2.6.3) for the study on HIV transmission in Uganda. During the study, 18 outputs such as HIV prevalence and population male size were considered. For the first wave, emulators for two outputs did not validate well, and these two outputs were left out of the first wave analysis and were added later when their behaviour became more regular.

The maximum implausibility is defined as

$$\mathcal{I}_M(\mathbf{x}) = \arg\max_j \mathcal{I}_j(\mathbf{x}) \tag{2.33}$$

and it could be used to rule out input parameter settings. This measure is considered to be conservative since it imposes the requirement that all outputs of a computer model to be consistent with the observation to be part of NROY space (Craig et al., 1997). A less sensitive measure, j^{th} maximum implausibility, could be used to rule out input parameter space (Craig et al., 1997) and defined as

$$\mathcal{I}_{jM}(\mathbf{x}) = \max_i \Big(\big\{ \mathcal{I}_i(\mathbf{x}) \backslash \mathcal{I}_M(\mathbf{x}), \mathcal{I}_{2M}(\mathbf{x}), \cdots, \mathcal{I}_{(j-1)M}(\mathbf{x}) \big\} \Big). \tag{2.34}$$

By specifying $j > 1$ for the measure defined above, we would rule out $\mathbf{x}$ for which at

least j computer model outputs of $\boldsymbol{f}(\mathbf{x})$ are not consistent with the observation $\boldsymbol{z}$. It is common to consider the second $\mathcal{I}_{2M}(\mathbf{x})$ and the third $\mathcal{I}_{3M}(\mathbf{x})$ highest implausibility for each $\mathbf{x}$ (Vernon and Goldstein, 2009; Vernon et al., 2010; Andrianakis et al., 2015, 2017; Williamson et al., 2017). The NROY space using this measure is defined as

$$\mathcal{X}_{NROY} = \big\{\mathbf{x} \in \mathcal{X} : \mathcal{I}_{jM}(\mathbf{x}) \leq a\big\}. \tag{2.35}$$

For instance, Williamson et al. (2017) used the $\mathcal{I}_{3M}$ measure to rule out the implausible input space for temperature and salinity profiles in the NEMO ocean model. In particular, the implausibility function for each of the thirty depth levels of ocean mean temperature was computed and the third largest $\mathcal{I}_j(\mathbf{x})$, for $j = 1\ldots, 30$, was defined as a measure for deriving NROY space. Effectively, if 3 or more model outputs are greater than the threshold for input point $\mathbf{x}$, this point is ruled out.

2.6.3 Refocussing

Calibration has proven to be challenging in situations where we are dealing with simulators with a large number of inputs and outputs, as we are required to simultaneously match a large number of inputs. For instance, for the HIV transmission model considered by Andrianakis et al. (2015) and Andrianakis et al. (2017) with 96 inputs and 50 outputs, it is computationally infeasible to perform calibration. Instead, we could perform history matching iteratively, i.e. use history matching to identify regions of input space where metrics of interest from the model output, predicted by an emulator, are not consistent with observations, and discard them in iterations known as waves (refocussing). Refocussing has been used previously to analyse the hydrocarbon model (Craig et al., 1996), Galaxy formation model (Vernon et al., 2010), HIV transmission model (Andrianakis et al., 2015), and climate models (Edwards et al., 2011; Williamson et al., 2017; Salter et al., 2018).

We describe the process of refocussing below. First we are required to choose an initial ensemble design $\mathrm{X}_{[1]} \in \mathcal{X}$, defined as

$$\mathrm{X}_{[1]} = (\mathbf{x}_{1,1}, \cdots, \mathbf{x}_{1,n_1})^T,$$

and produce the computer model runs at the initial design $\mathrm{F}_{[1]}$

$$\mathrm{F}_{[1]} = (f(\mathbf{x}_{1,1}), \cdots, f(\mathbf{x}_{1,n_1}))^T.$$

Based on the generated ensemble, we construct an emulator to predict key metrics (outputs) from the computer model and use the implausibility function to define a subset of $\mathcal{X}$ that is not ruled out yet (NROY) to be the subset for which $\mathcal{I}(\mathbf{x}; \mathrm{F}_{[1]})$ is less than or equal a pre-specified threshold value a, i.e.

$$\mathcal{X}^1 = \Big\{\mathbf{x} \in \mathcal{X} : \mathcal{I}(\mathbf{x}; \mathrm{F}_{[1]}) \leq a\Big\}.$$

The whole process of deriving NROY space $\mathcal{X}^1$ is called wave 1. Refocussing is the process of repeating this multiple times, each time, in wave k, beginning with the parameter space $\mathcal{X}^{k-1}$. Mathematically, NROY space in wave k is defined as

$$\mathcal{X}^k = \Big\{\mathbf{x} \in \mathcal{X}^{k-1} : \mathcal{I}(\mathbf{x}; \mathrm{F}_{[k]}) \leq a\Big\},$$

where $\mathcal{I}(\mathbf{x}; \mathrm{F}_{[k]})$ can be evaluated by using an emulator for $f(\mathbf{x})$ defined inside $\mathcal{X}^{k-1}$ and constructed based on the design

$$\mathrm{X}_{[k]} = (\mathbf{x}_{k,1}, \cdots, \mathbf{x}_{k,n_k})^T \in \mathcal{X}^{k-1}$$

and running the computer model at the design to generate

$$\mathrm{F}_{[k]} = (f(\mathbf{x}_{k,1}), \cdots, f(\mathbf{x}_{k,n_k}))^T.$$

It is important to note that $\mathbf{x}$ is nonimplausible at wave k only if it is nonimplausible for all the waves that precede it.

The process of refocussing provides a number of benefits (Williamson et al., 2017). Firstly, we have a freedom to use a different form of implausibility function, $\mathcal{I}(\mathbf{x})$, at every wave, i.e. a different set of metrics could be used such as we could include complex metrics, spatial fields or time series, after the very non-physical re-

gions of parameter space have been removed in earlier waves. For instance, Vernon et al. (2010) and Salter et al. (2018) demonstrated the use of both multidimensional implausibility metrics, as described in subsection 2.6.2. In particular, for the first few waves, the second highest implausibility, $\mathcal{I}_{2M}(\mathbf{x})$, and the third highest implausibility, $\mathcal{I}_{3M}(\mathbf{x})$, were used to derive the NROY space. For later waves, the multivariate implausibility was used, because it is easier to specify the covariance structure in the reduced input space for $Var[\boldsymbol{e}]$ and $Var[\boldsymbol{\eta}]$. In contrast, calibration requires all model outputs to be considered at once.

Secondly, as later waves are reached, we manage to run the computer model at the reduced parameter space, and, as a result, we increase the density of our ensemble (Williamson et al., 2017). The emulator for $f(\cdot)$ needs to be accurate only over the NROY space at the current wave, which allows us to produce a better proxy for $f(\cdot)$ than at wave 1. After reducing the size of parameter space and having an accurate emulator, calibration may be used to find a more accurate probability distribution for $\mathbf{x}^*$ than a calibration performed over $\mathcal{X}$ (Edwards et al., 2011).

Thirdly, the result of Bayesian calibration is always the probability distribution for $\mathbf{x}^*$ over the input space $\mathcal{X}$. If $f(\cdot)$ cannot represent the system, then this result is meaningless, and the calibration distribution is unfit to be used in any further applications, e.g. forecasting. In contrast, history matching would rule out the entire parameter space, which indicates that the computer model is not representative of the true physical process.

The stopping rule, i.e. how many iterations (waves) to perform during iterative refocussing, is determined by a number of factors addressed by Williamson et al. (2015). Firstly, the computational budget and time could limit the number of waves of refocussing (Williamson et al., 2017). For instance, history matching has been used for GCM (General circulation model) class models by Williamson et al. (2013) and Williamson et al. (2015) and only one wave of history matching was performed due to the inability to generate ensembles in NROY space to perform the next wave. Another stopping criterion is the emulator variance, $Var[\boldsymbol{f}(\mathbf{x})]$: once it is smaller than the denominator in the implausibility calculation, then it is unlikely that further

ensembles for history matching will significantly change the implausibility, and as a result the shape of NROY space (Williamson et al., 2015). Finally, we have to stop the iterative refocussing process in the extreme case when the whole input space is ruled out using a certain metric, and as a result, structural error, determined by discrepancy variance, $Var[\boldsymbol{\eta}]$, has been located (Williamson et al., 2017). For instance, during climate model tuning, the model discrepancy variance, $Var[\boldsymbol{\eta}]$, could be considered as the tolerance to structural error (model error), i.e. missing an accurate representation of certain metrics. History matching allows users to explore different regions of NROY space by varying tolerances to the structural error through model discrepancy variance (Williamson et al., 2015).

In this thesis, we are interested in using uncertainty quantification methods to calibrate (history match) complex computer models (climate models) with nonstationary response across the parameter space.

Chapter 3

Automatic and robust uncertainty quantification (UQ) software

ExeterUQ

3.1 Introduction

The increasing importance of the Uncertainty Quantification (UQ) field, discussed in section 2.2, has led to the development of libraries for modelling and quantifying uncertainties in complex computer simulators. In particular, there are a number of open-source software platforms for engineering and commercial applications written in MATLAB, Python and C++ such as OpenTURNS (Baudin et al., 2017), UQLab (Marelli and Sudret, 2014), DAKOTA (Adams et al., 2009) and COSSAN (Patelli, 2017). These tools follow a general-purpose framework, which means that a reasonably wide range of engineering and scientific problems can be treated by a single software (Patelli, 2017). As a result, special attention is paid to the interaction of the software with external code, i.e. producing simulations (runs) of a complex computer model to assist Uncertainty Quantification tasks. Also, due to commercial and engineering applications, some of these tools have powerful and interactive interfaces to assist inexperienced users in modelling and quantifying uncertainties. The core components, methodological components, allow users to undertake such tasks as sensitivity analysis, uncertainty propagation and meta-modelling (emulation). The

emulation tools available in the majority of software libraries are polynomial chaos, artificial neural networks and kriging.

Interestingly, we have not found any comprehensive packages for uncertainty quantification in the R programming language. Most of the R packages are focused around a specific aspect of uncertainty quantification. For instance, there are packages that could be used for emulation, in particular GP emulation, such as `DiceKriging` (Roustant et al., 2012b), `tgp` (Gramacy et al., 2007), `CGP` (Ba and Joseph, 2018), `GPfit` (MacDonald et al., 2015), `mlegp` (Dancik and Dorman, 2008) (maximum likelihood estimates of Gaussian Processes) and `RobustGaSP` (Gu et al., 2018b). The `sensitivity` package (Iooss et al., 2018) could be used to perform the sensitivity analysis.

In this chapter, we present our own, in-house developed R and Stan-based software for UQ (Stan Development Team, 2017), namely `ExeterUQ`. `ExeterUQ` possesses a number of key and unique features. In particular, Bayesian emulation is performed using RStan (Stan Development Team, 2017), the Stan interface for R. Our software also allows our users to employ the constructed GP emulators for multi-wave calibration, history matching. We demonstrate the performance of `ExeterUQ` on a climate model application in the tuning process of internal parameters of boundary-layer clouds parameterization scheme. The tuning targeting the boundary-layer clouds parameterization scheme is crucial for the climate community for a number of reasons. Firstly, Hourdin et al. (2015) found that deficiencies in clouds parameterization contributed to persistent and systematic biases in sea surface temperature observed in the global climate model. Secondly, boundary-layer clouds have been shown to largely account for the spread in climate change predictions, limiting the accuracy of forecasts produced by climate models (Bony and Dufresne, 2005; Randall et al., 1996; Williams and Webb, 2009; Vial et al., 2013).

The chapter has the following structure. In section 3.2, we introduce the application for `ExeterUQ`, i.e. tuning the boundary layer clouds. In section 3.3, we consider in detail a number of R packages used to construct a GP emulator as alternatives to our approach implemented in `ExeterUQ` software. Section 3.5 demonstrates how

to build a GP emulator using our tools on one of the scenarios in HIGH-TUNE. In section 3.6, we describe the approaches we use to validate the performance of our surrogate models, and in section 3.7, we perform tuning using our tools. Section 3.8 contains the discussion and the future planned development of `ExeterUQ` software as well as some methodological questions in UQ raised by discussions with climate modellers.

3.2 Climate model application

`ExeterUQ` software has been developed as part of the HIGH-TUNE project in collaboration with climate modellers and scientists from Laboratoire de Météorologie Dynamique (LMD), Centre National de Recherches Météorologiques (CNRM) and Laboratoire Plasma et Conversion d'Energie (LAPLACE). The main objective of this project was to improve the representation of the boundary-layer clouds in the global climate models. These types of clouds play a crucial role in the water and energy cycles of the atmosphere and impact surface temperatures at various scales (Bony and Dufresne, 2005). However, the boundary layer clouds are much smaller than a grid cell of a climate model, and therefore the collective behaviour and the effect on the large scale model outputs of an ensemble of boundary-layer clouds is parameterized. The parameterization schemes depend on a variety of "free" parameters and calibrating (tuning) these parameters is crucial to avoid biases in the global climate model (Hourdin et al., 2017).

The tuning of cloud parameters could be addressed in two ways, namely a traditional global model tuning and a process-based tuning. Global model tuning is based on the process of tuning of climate model by considering a specific climate performance metric such as statistics on surface precipitation and temperature (Hourdin et al., 2017). However, Hourdin et al. (2017) stated that this type of tuning could lead to over-fitting or over-tuning because a good performance in those metrics could be achieved due to compensating errors.

An alternative approach adopted in HIGH-TUNE project is process-based tuning. Process-based tuning is centred around the idea of using process-oriented met-

rics for tuning, such as compositing cloud or precipitation characteristics by dynamical regimes (Bony and Dufresne, 2005). Process-oriented metrics could be helpful in relating large-scale biases observed in a global climate model to the deficiencies in the parameterization scheme (Hourdin et al., 2017).

We rarely have process-based data, since it is extremely challenging to collect data for the process-oriented metrics. Therefore, the process-based tuning in HIGH-TUNE project is based on the comparison of single-column versions of the global models (SCM) with explicit 3D high-resolution Large Eddy simulations (LES) of the same boundary layer clouds. LES are the numerical simulations of clouds produced with a resolution of a few tens of meters that are able to resolve the turbulent eddies, which constitute the root of the cumulus and the cloud dynamics. In fact, LES have been extensively used up until now to study different cloud properties within different clouds regimes (Couvreux et al., 2005; Heus and Jonker, 2008; Wang and Feingold, 2009).

During HIGH-TUNE project, the metrics that correspond to the radiative effect of clouds as well as the key characteristics of clouds were computed on a series of LES simulations and treated as observations. Our collaborators were interested in finding a subset of input parameter values of boundary-layer clouds parameterization scheme, at which SCM's output closely matches LES on a number of cases. To find these acceptable matches the UQ tools such as GP emulators and history matching (iterative refocussing) were of interest to employ.

Climate modellers studied different cloud regimes that correspond to twelve cases that sample various conditions over the globe. As part of demonstration of `ExeterUQ` functionality, we consider a selection of cases, in particular

- ARMCU/REF is the case of boundary layer cumulus cloud, continental shallow over land (USA), which is based on Large-Eddy simulations performed by Brown et al. (2002). In general, cumulus clouds have flat bases and appear to be puffy in appearance. Cumulus clouds are low-level clouds and indicators of a fair weather.

- BOMEX/REF is the case of boundary layer cumulus cloud, oceanic shallow

over the Caribbean region, which is based on Large-Eddy simulations performed by Siebesma et al. (2003). The cumulus clouds produced over the ocean has different characteristics than the cumulus clouds produced over the land. In particular, shallow cumulus convection over land have cloud forcing that is both stronger and time-varying than over the ocean.

- SANDU/REF is the case that considers the transition from cumulus to stratocumulus. Stratocumulus clouds are low-level patches of clouds and present in all types of the weather.

For each case of clouds' behaviour, a selection of metrics provided in Table 3.1 were considered.

Metric	Description	Units of measure
`theta500`	the potential temperature at 500 metres	K
`qv500`	water vapour at 500 metres	grams per kilogram
`zhneb`	effective height of clouds	km
`nebmax`	maximum nebulosity	
`nebzmin`	base of clouds	(metres/feet/hectopascal)
`nebzmax`	top of clouds	(metres/feet/hectopascal)

Table 3.1: Model metrics (outputs of interest) description considered in HIGHTUNE project.

3.3 Review of Uncertainty Quantification (UQ) software

In this section, we provide a short review of available Uncertainty Quantification (UQ) software, which is a natural motivation for developing `ExeterUQ`. In particular, we are interested in considering software, which allows users to construct an emulator and perform calibration.

Open-source UQ software such as OpenTURNS (Baudin et al., 2017), UQLab (Marelli and Sudret, 2014), DAKOTA (Adams et al., 2009) and COSSAN (Patelli, 2017) enable users to construct a wide range of emulators such as neural networks, polynomial chaos, polynomial approximations and kriging. However, unlike other

surrogates, GP emulators (kriging) provide uncertainty about the prediction, code uncertainty, and taking into account code uncertainty when performing calibration is crucial (Kennedy and O'Hagan, 2001). Based on this reasoning, we had to employ one of the available software or packages or develop our own code for constructing a GP emulator. A range of packages that are available for building GP emulators varies based on different methods used to estimate GP emulator hyperparameters, precisely $\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}, \tau^2$. The estimation of GP emulator hyperparameters is crucial since it largely affects the predictive power of GP emulator and any statistical inference. The values of regression coefficients, $\boldsymbol{\beta}$, and scale parameter, σ^2, could be easily obtained by performing marginalisation, described in subsection 2.3.4. However, the estimation of correlation length parameters, $\boldsymbol{\delta}$, has proven to be challenging (Kennedy and O'Hagan, 2001)

One approach to the estimation of correlation length parameters, $\boldsymbol{\delta}$, is via maximum likelihood estimation, i.e. fixing at the MLE $\hat{\boldsymbol{\delta}}_{MLE}$. For instance, Roustant et al. (2012a) provides an overview of packages and libraries available for kriging, and in particular, `DiceKriging` is widely used to construct a kriging model. It is suitable for applications in higher dimensions and provides a wide range of covariance functions such as Gaussian, Matérn with $\nu = \frac{5}{2}$ and $\nu = \frac{3}{2}$, exponential and power-exponential, discussed in subsection 2.3.2. `DiceKriging` uses equations (2.12) and (2.13) to produce predictions and derives the values of $\boldsymbol{\delta}$ by likelihood maximization. A quasi-Newton Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (function `optim`) and the hybrid algorithm `genoud` from package `rgenoud` (Mebane and Sekhon, 2011) are implemented in `DiceKriging` package to derive the MLEs, $\hat{\boldsymbol{\delta}}_{MLE}$.

However, a number of problems such as numerical issues and poor predictive performance of GP emulator arise by employing maximum likelihood estimation for correlation length parameters. In particular, Gu et al. (2018b) identified two main cases. The first case is when the estimated correlation length parameters leads to the generation of covariance matrix evaluated at design points, K, close to singular. However, this numerical issue could be easily resolved by adding a nugget described

in detail in subsection 2.3.3. It is crucial to note that the behaviour of GP emulator would significantly change with the addition of a nugget parameter (Andrianakis and Challenor, 2012). Rasmussen and Williams (2004) demonstrated on a simple numerical example that the marginal likelihood defined as a function of $\boldsymbol{\delta}$ and τ^2 exhibits two local modes. At the first local mode, we observe a nugget value, τ^2, close to 1 and the independence of marginal likelihood function from $\boldsymbol{\delta}$. In this case, the GP emulator models everything as part of the noise. At the second mode, we observe a small value of a nugget parameter, τ^2, and a short correlation length scale. In this case, the marginal likelihood is close to being independent of the nugget process, and GP emulator behaves like an interpolator.

The second major case considered by Gu et al. (2018b) is when a covariance matrix of design points, K, is near-diagonal. It is common to employ a separable covariance function in the GP emulator specification such as we could redefine the covariance function $k(\mathbf{x}, \mathbf{x}')$ as

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \prod_{i=1}^{p} r(x_i, x_i'; \delta_i),$$

with $r(\cdot, \cdot; \delta_i)$ being a one-dimensional correlation function for the i^{th} coordinate of the input vector. In fact, such pre-defined covariance structure leads to the decomposition of the covariance matrix into a product of the correlation matrix of these inputs, i.e.

$$K = \sigma^2 \times R_1 \circ R_2 \circ \cdots \circ R_p,$$

where each R_i is the correlation matrix for the i^{th} input and $\circ$ is the Hadamard product. If one of R_i for $i = 1, \ldots, p$ is being near I, an $n \times n$ diagonal matrix, which is caused by $\hat{\delta}_i$ being close to zero, then K has a form close to diagonal. $\hat{\delta}_i$ close to zero is obtained in the situations when the marginal likelihood is very flat in the tails. As a result, the predictive mean function $m^{**}(\mathbf{x})$, given in equation (2.12), behaves as an impulse function at the design points, X (Gu et al., 2018b).

To avoid the unstable results described above, `DiceKriging` and `DiceOptim` employ expected improvement criterion as well as bounds for correlation length

parameters (Roustant et al., 2012a). However, the trade-off with numerical stability is larger predictive errors (Gu et al., 2018b).

Recently, a new and more robust method for estimating GP emulator hyperparameters, a marginal posterior mode estimator, has been implemented inside `RobustGaSP` package (Gu et al., 2018a). In particular, one of the objective priors (reference prior), $\pi(\boldsymbol{\delta})$, discussed in subsection 2.3.5, is employed inside the package, and the values of the correlation length parameters are estimated by the modes of the marginal posterior distribution defined as

$$(\hat{\delta}_1, \ldots, \hat{\delta}_p) = \arg\max_{\delta_1,\ldots,\delta_p} p\big(\mathrm{F}|\delta_1, \ldots, \delta_p\big)\pi(\delta_1, \ldots, \delta_p).$$

These new estimates are demonstrated to provide stable results for GP emulator with lower predictive errors.

As an alternative to the approaches to hyperparameter estimation adopted above, `tgp` constructs Treed Gaussian process (TGP) models and offers a Bayesian treatment of covariance parameters, relying on Markov chain Monte Carlo techniques (Gramacy et al., 2007). Despite being calculation intensive, `tgp` is a highly efficient and fast package. The main drawback for users is that they could not explore prior options for GP model hyperparameters other than the ones provided by Gramacy and Lee (2008).

The simulator could be calibrated to the observation by performing Bayesian calibration or history matching. The `R` package `BACCO` (Hankin, 2005) is developed to perform Bayesian calibration, and this package has been used to calibrate C-GOLDSTEIN's yearly global mean air temperature to the observational data taken from SGEN dataset (Marsh et al., 2002). C-GOLDSTEIN is an efficient, intermediate complexity climate model with highly simplified physics, but with 3-D ocean dynamics. However, we were interested in performing calibration via history matching to consider a wide range of metrics (model outputs) at the same time which is computationally challenging to achieve by employing the Bayesian calibration presented by Kennedy and O'Hagan (2001).

To address all the confounding with GP hyperparameters and calibration, we

decided to develop our own ExeterUQ software, which allows users to construct GP emulators that provide robust predictions and fully acknowledge hyperparameters' uncertainty as well as to perform multi-wave calibration (history matching) for a range of metrics simultaneously. We were interested in constructing GP emulators with flexible priors for GP hyperparameters. For this purpose we use Stan (Carpenter et al., 2017), in particular, RStan, the Stan interface to R. Stan allows its users to perform full Bayesian statistical inference by adopting the Hamiltonian Monte Carlo (HMC) and no-U-turn samplers (NUTS) (Carpenter et al., 2017), which is more efficient and robust than Gibbs sampling used by BUGS (Bayesian inference Using Gibbs Sampling) (Spiegelhalter et al., 1996) and JAGS (Just Another Gibbs Sampler) (Plummer et al., 2003).

3.4 Priors for model hyperparameters

We stated in section 3.3 that we are interested in constructing GP emulators with a flexible prior specification for model hyperparameters that allows users to obtain robust predictions. Stan provides users with an option to specify priors using the probability functions implemented in Stan or defined by users (Stan Development Team, 2017). ExeterUQ software is primarily targeting non-experts in statistics and, as a result, we had to propose "out of the box" choices for priors for the GP model hyperparameters, that provide users with stable and robust results.

We start by employing a statistical model for a scalar output (metric of interest), $f(\mathbf{x})$, defined in equation (2.4). We have to pre-specify the form of the regression function, $h(\mathbf{x})$, (more details on how we derive the form of regression functions in Appendix B.3.1) and correlation function, $r(\cdot, \cdot)$. We choose the squared exponential as our correlation function, i.e.

$$r(\mathbf{x}, \mathbf{x}', \boldsymbol{\delta}) = \exp\left\{-\sum_{i=1}^{p}\left(\frac{x_j - x'_j}{\delta_j}\right)^2\right\}.$$

We adopt the squared exponential correlation function as $r(\mathbf{x}, \mathbf{x}', \boldsymbol{\delta})$ as it is the only correlation function option provided by Stan math library (Stan Development

Team, 2017). In particular, we used the `cov_exp_quad` and its variants to compute covariance matrices in a fast way, avoiding usage of expensive for loops and other inefficient programming practices (Stan Development Team, 2017). We acknowledge the fact that it would be useful to provide users of our tools with a broader choice of correlation functions described in subsection 2.3.2 and implemented, for instance, in R package DiceKriging (Roustant et al., 2012a). These correlation functions could be specified in a special function-definition block in Stan file (Stan Development Team, 2017).

Prior specification for GP emulator hyperparameters, $\Theta = \left\{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}, \tau^2\right\}$, requires special attention. We specify a weakly informative prior, i.e. $\boldsymbol{\beta}_{2:m} \sim N(0, 10)$, for the regression coefficients of the mean function. We employ a weakly informative prior so that it rules out unreasonable parameter values but is not so strong to rule out values that might make sense. Operating with weakly informative prior, we effectively allow the likelihood to dominate the posterior influence. For the intercept parameter β_1, we do not explicitly specify prior distribution, which in Stan corresponds to default prior $(-\infty, +\infty)$. Instead, we use information from the linear model to define the initial values for each chain.

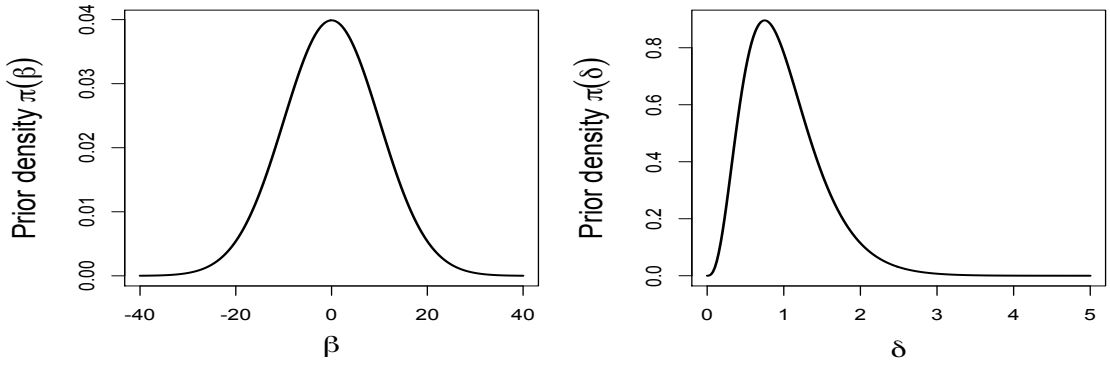

Figure 3.1: Density plots of Normal(0, 10) (*left*) and Gamma(4, 4) (*right*) priors specified for GP hyperparameters.

The weakly informative prior, $\delta_k \sim \text{Gamma}(4, 4), k = 1, \ldots, p$, is used for correlation length parameters to avoid the issues with correlation length parameter estimation discussed in detail in section 3.3. From Figure 3.1, we observe that this form of prior restricts extremely small correlation length values as well as constrains

large correlation length values. Small correlation length values would lead to instability issues with the GP emulator (see section 3.3 for more details). On the contrary, a large correlation length value obtained for one of the inputs, say i, would lead to off-diagonal entries of R_i close to 1, i.e. $r(x_i, x_i'; \delta_i) \to 1$. As a result, we expect to observe no major contribution of R_i to the overall covariance matrix, and we could deduce that the likelihood and, hence, the posterior becomes independent from this particular input. Therefore there is no effect from this input on the final estimated GP model (Stan Development Team, 2017). To explain these effects of correlation length parameters estimates on the GP model, we consider the log-likelihood for F, defined as

$$\log\big(p(\mathrm{F}|\mathrm{X}, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta})\big) = -\frac{1}{2}\big(\mathrm{F} - H\boldsymbol{\beta}\big)^T K^{-1}\big(\mathrm{F} - H\boldsymbol{\beta}\big) - \frac{1}{2}\log(det(K)) - \frac{n}{2}\log 2\pi.$$

The three terms have interpretable roles: $-\frac{1}{2}\big(\mathrm{F} - H\boldsymbol{\beta}\big)^T K^{-1}\big(\mathrm{F} - H\boldsymbol{\beta}\big)$ is the data-fit term, as it is the only term that involves full ensemble $\big\{\mathrm{X}, \mathrm{F}\big\}$; $\log(det(K))/2$ is a complexity penalty and $n\log(2\pi)/2$ is a normalization constant (Rasmussen and Williams, 2004).

We attempt to perform numerical studies similar to Rasmussen and Williams (2004). In particular, we consider the effect of the correlation length parameter on the likelihood function by setting $(\beta_0, \beta_1, \sigma^2, \tau) = (0, 0, 1, 0.1)$ and varying values of the correlation length parameter, δ. Figure 3.2 (left panel) shows an F of size thirty, drawn from a zero-mean Gaussian process with squared exponential correlation function and parameters $(\delta, \sigma^2, \tau) = (1, 1, 0.1)$. From Figure 3.2 (central panel), we observe that the negative complexity penalty (red dashed line) increases with the correlation length parameter value since the GP model becomes less complex. The data-fit term (green dashed line) rapidly decreases with the increase in correlation length parameter values since the GP model becomes less flexible. We also observe an increase in the value of the complexity penalty term together with no change in the data-fit term for smaller values of the correlation length parameter. We encountered overfitting, i.e. the model complexity has increased, but at the same time, the model representation of F has not improved.

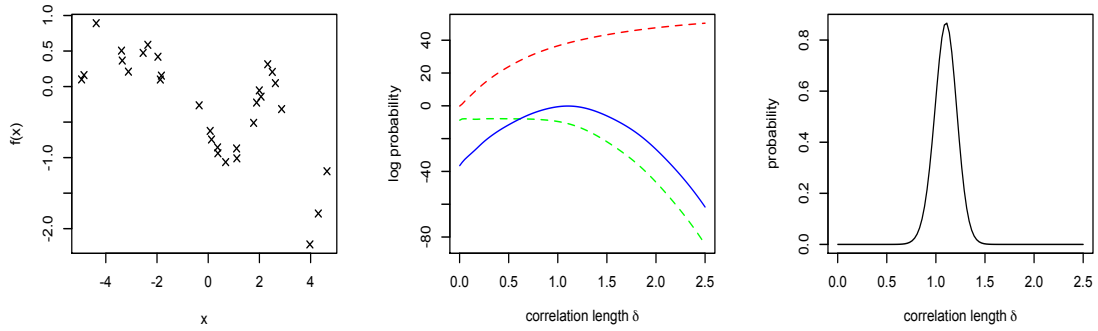

Figure 3.2: *Left panel*: F is generated from a zero-mean GP with hyperparameters $(\delta, \sigma, \tau) = (1, 1, 0.1)$, as shown by the + symbols. *Central panel*: decomposition of log likelihood (*blue line*) into data-fit term (*green dashed line*) and minus complexity penalty (*ref dashed line*), as a function of correlation length parameter. *Right panel*: Likelihood for ensemble $\{\mathrm{X}, \mathrm{F}\}$ is derived as a function of correlation length parameter.

For parameter σ, we use a lower truncated normal prior distribution, i.e. truncated from 0, with location α_σ as the residual standard error from the Ordinary Least Squared (OLS) fit with the mean function specified in $h(\mathbf{x})$. The OLS is employed in the process of obtaining the mean function (for more details see Williamson et al. (2015)). The squared scale β_σ is the squared difference of the standard deviation of the ensemble runs F and α_σ. Effectively, this form of prior represents our attempt to model the variability in the response unexplained by the mean function. In regards to the nugget parameter τ^2, we fix it at an arbitrarily small value, 0.0001, to facilitate stable matrix inversion as suggested by Andrianakis and Challenor (2012).

3.5 Example: constructing an emulator with `ExeterUQ`

In this section, we demonstrate how to implement a GP emulator with `ExeterUQ` tools to introduce plots and ideas that will be seen throughout the thesis. We start with showing how to construct a GP emulator for a single output (single metric of interest). We then proceed to construct q independent univariate GP emulators for q metrics of interest based on the emulation of a single output.

To demonstrate the functionality of `ExeterUQ`, we have chosen the ARMCU/REF

case of HIGH-TUNE, described in section 3.2. We start by generating a 3-extended LHC of size 90 following the methodology presented by Williamson (2015). The 90 member LHC is composed of three, 30 member LHCs, each added sequentially, ensuring that the composite design is orthogonal and space-filling at each extension. Secondly, we evaluate the simulator (SCM) at the generated design. We start by considering a single metric of interest `theta500`. The first two LHCs, a 60 member LHC, are used as a design for our GP emulator, while the last extension is needed to validate the performance of GP emulator. The process of defining an ensemble as well as the validation set is described in the Appendix B.3.

Figure 3.3 demonstrates the scatter plots of response variable `theta500` against five input parameters. We observe that there is a strong negative signal in the input variable `thermals_fact_epsilon` with a slightly weaker signal in `thermals_ed_dz`. Judging from scatter plots, we do not observe any obvious effect from input variables `cld_lc_lsc`, `rad_chaud1` and `z0min` on the response variable, `theta500`.

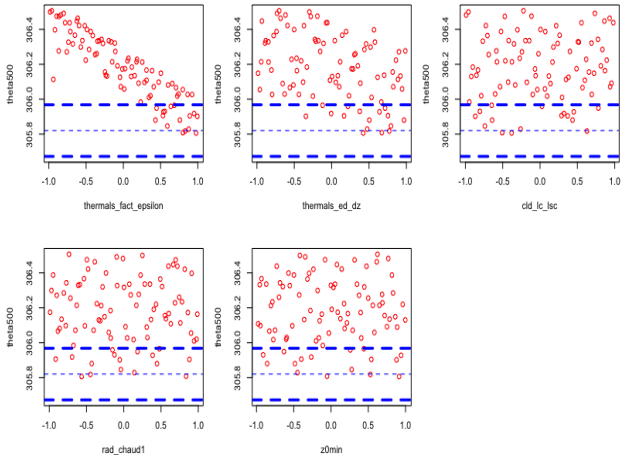

Figure 3.3: `theta500` response against five input parameters on the standardized scale. The blue dashed lines correspond to z plus and minus $2(Var[e] + Var[\eta])^{1/2}$, where $Var[e]$ is the variance of the observation error and $Var[\eta]$ is the model discrepancy error.

We proceed further to derive a form of regression function, $h(\mathbf{x})$, for a GP model. In particular, we are interested in fitting a complex and sophisticated form of prior mean function, which could include linear, quadratic and cubic terms in each of the parameters with Fourier transformations and interactions between all input parameters. In subsection 2.5, we mentioned that fitting a complicated response surface

could be used in mitigating nonstationarity (Rougier et al., 2009; Vernon et al., 2010; Williamson et al., 2013). Introducing a highly structured mean function into the model could also lead to significant computational savings, since it allows users to use a less complex form for the residual term (Craig et al., 1996; Cumming and Goldstein, 2009). The approach to obtaining a mean function for GP emulator is described in detail in the Appendix B.3.1.

After obtaining the form of a regression function, $h(\mathbf{x})$, we use it in our prior mean function specification for a full GP model and proceed to construct a full GP emulator (for more details on the use of R and Stan in obtaining a full GP emulator see Appendix B.3.2).

By following the steps described above, we obtain an object `StanEmulator`, which is further used for validation, prediction and inference in `ExeterUQ` software presented later in this Chapter. A detailed description of the content of an object is provided in the Appendix B.3.3.

3.5.1 Emulating multivariate output of computer model

Our collaborators from `HIGH-TUNE` are interested in emulating a collection of independent and uncorrelated metrics of interest generated by the SCM. As a result, we construct emulators for these metrics independently. This approach is useful for history matching, as discussed in subsection 3.7.2. In particular, we can evaluate implausibilities for each metric separately and then rule out any parameter settings that fail to meet either all of these targets or most.

We use the same 90 member LHC described in section 3.5 to obtain the metrics of interest. The responses in two variables, i.e. `nebzmin` and `nebzmax`, are not continuous, and both variables act as factor variables so that we cannot use GPs. As a result, we proceed to emulate only the first four metrics of interest (see Appendix B.3.4 for more details). Figure 3.4 contains the scatter plots for all four metrics by column, and we observe that two input parameters, i.e. `thermals_fact_epsilon` and `thermals_ed_dz`, mainly affect the responses in all four metrics of interest.

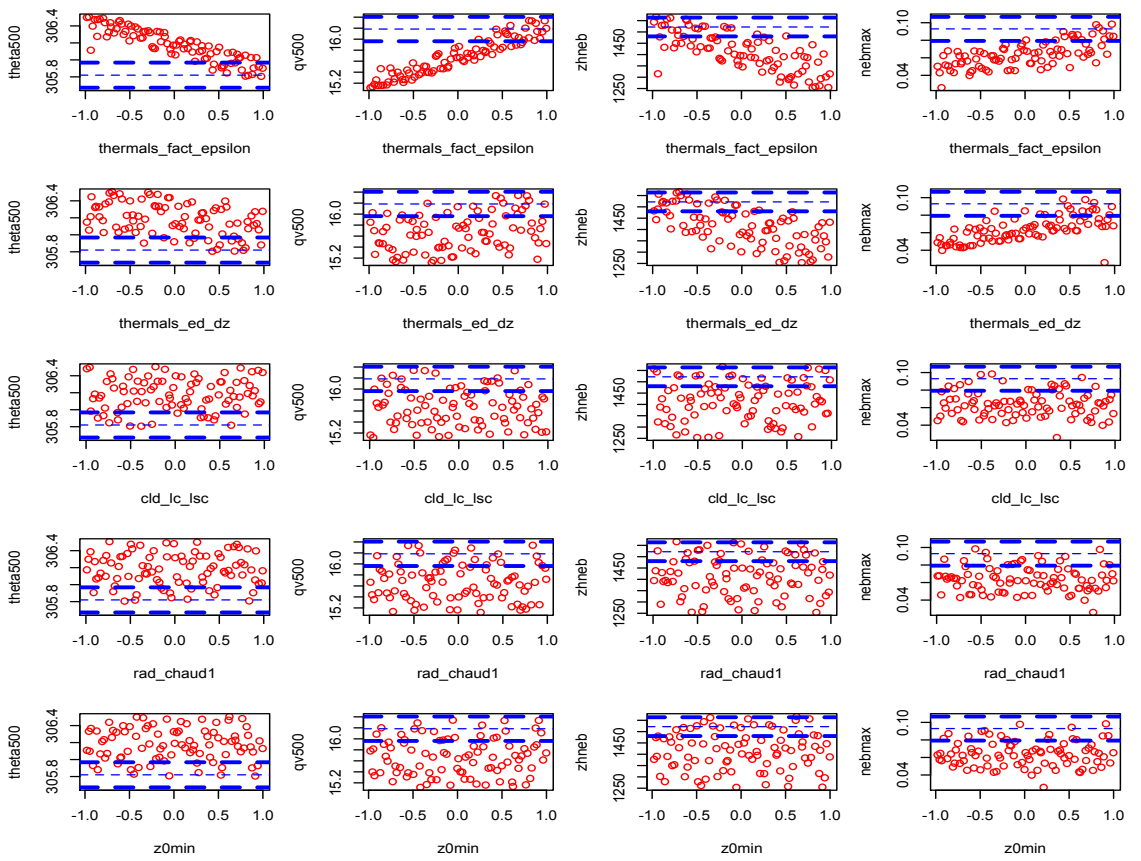

Figure 3.4: Scatter plots for `theta500`, `qv500`, `zhneb`, `nebmax`, by column against five input parameters on the standardized scale. The blue dashed lines correspond to z_i plus and minus $2(Var[e_i] + Var[\eta_i])^{1/2}$, where $Var[e_i]$ is the variance of the observation error and $Var[\eta_i]$ is the model discrepancy error.

3.6 Diagnostics for GP emulators

In the previous section, we described how to construct a GP emulator inside `ExeterUQ` software. In this section, we demonstrate how to perform diagnostics to assess the performance of a generated GP emulator before employing it for inferences such as prediction or history matching.

3.6.1 Leave-One-Out (LOO) diagnostics for GP emulators

The users of our software could be interested in producing a Leave-One-Out (LOO) diagnostics plot presented in Figure 3.5. The black dots and error bars show predictions together with ± 2 standard deviation from the leave-one-out emulators, while the green/red points are the true model output coloured by whether or not the truth lies within the error bars. From the LOO diagnostics plot in Figure 3.5, we

can deduce that the obtained GP emulator is a good representation of the metric of interest, `theta500`, as the emulator predictions are close to the actual model values with small error bars.

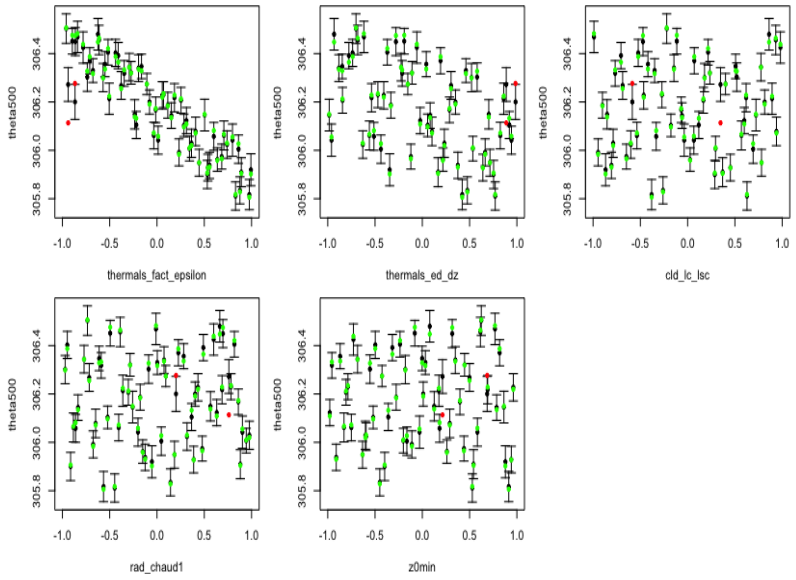

Figure 3.5: LOO diagnostics plot against each of the parameters. The predictions and two standard deviation prediction intervals are in black. The true function values are in green if they lie within two standard deviation prediction intervals, or red otherwise.

To obtain leave-one-out predictions, we perform the following steps. To produce predictions for input point $\mathbf{x}_j, j = 1, \ldots, n$ from design matrix, X, we have to compute equation (2.18) and (2.19) introduced in subsection 2.4. In practice the samples from the set of posterior simulations of the parameters, $(\boldsymbol{\beta}_i, \sigma_i, \boldsymbol{\delta}_i, \tau_i^2), i = 1, \ldots, 2000$ are used to compute

$$E\big[f(\mathbf{x}_j)|\big\{\mathrm{X}_{-j}, \mathrm{F}_{-j}\big\}, \boldsymbol{\beta}_i, \sigma_i, \boldsymbol{\delta}_i, \tau_i^2\big] \quad \text{and} \quad Var\big[f(\mathbf{x}_j)|\big\{\mathrm{X}_{-j}, \mathrm{F}_{-j}\big\}, \boldsymbol{\beta}_i, \sigma_i, \boldsymbol{\delta}_i, \tau_i^2\big].$$

We simulate $f^{(i)}(\mathbf{x}_j)$ from a Normal distribution with the previously computed expectation and variance with $i = 1, \ldots, M$ to obtain the distribution of points and compute the $E\big[f(\mathbf{x}_j)|\big\{\mathrm{X}_{-j}, \mathrm{F}_{-j}\big\}\big]$ and $Var\big[f(\mathbf{x}_j)|\big\{\mathrm{X}_{-j}, \mathrm{F}_{-j}\big\}\big]$. The function used to obtain the LOO diagnostics is provided in the Appendix B.4.1.

We could also compute standardized prediction errors (standardized residuals)

based on LOO diagnostics (see Appendix B.4.1 for more details)

$$\frac{f(\mathbf{x}_j) - E\big[f(\mathbf{x}_j)|\{\mathrm{X}_{-j}, \mathrm{F}_{-j}\}\big]}{\sqrt{Var\big[f(\mathbf{x}_j)|\{\mathrm{X}_{-j}, \mathrm{F}_{-j}\}\big]}}, j = 1, \ldots, n.$$

In particular, plots of residuals against inputs are commonly used to evaluate the performance of emulators. We discussed different diagnostics used to validate GP emulator performance in subsection 2.4. We expect to see a horizontal band containing the errors.

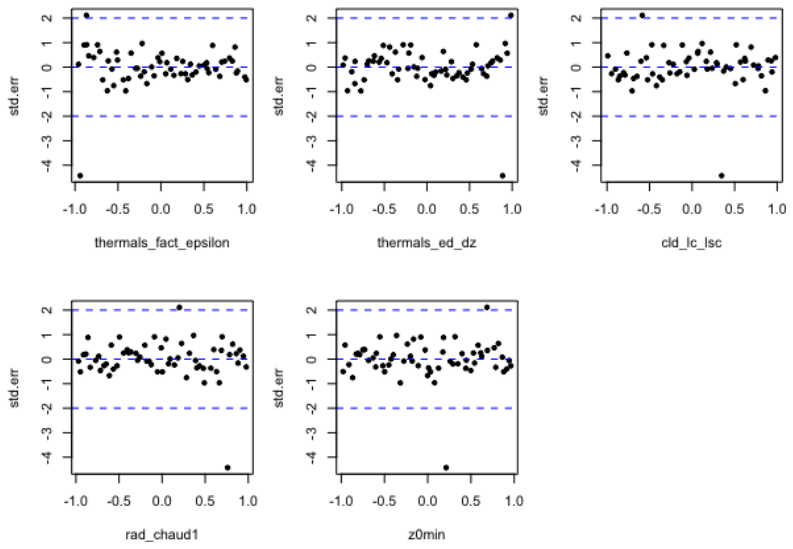

Figure 3.6: Individual standardized errors obtained for the design against each of the parameter.

From Figure 3.6, we do not observe any strong relationship between the standardized error values and the model inputs. The majority of errors are within horizontal band between -2 and 2. There is one large individual error, with an absolute value greater than 3. However, an isolated error of this kind might be ignored (Bastos and O'Hagan, 2009). Based on LOO diagnostics, we can conclude that there is no conflict between emulator and climate model representation.

3.6.2 Validation plots for GP emulators

It is a useful practice to validate the adequacy of a GP emulator in representing model response by comparing the model outputs with the GP emulator predictions

based on an unseen validation set (Bastos and O'Hagan, 2009).

Figure 3.7 demonstrates the diagnostics plot similar to the ones obtained in subsection 3.6.1 for a validation set (see Appendix B.4.2 for more details). As before, black dots and error bars show predictions together with ±2 standard deviation prediction interval, while green/red dots are the true model output coloured by whether or not the true value lies within the error bar. From Figure 3.7, we could observe one failure (red dot) which is consistent with our uncertainty specification. In general, the emulator's predictions are close to true values with small error bars.

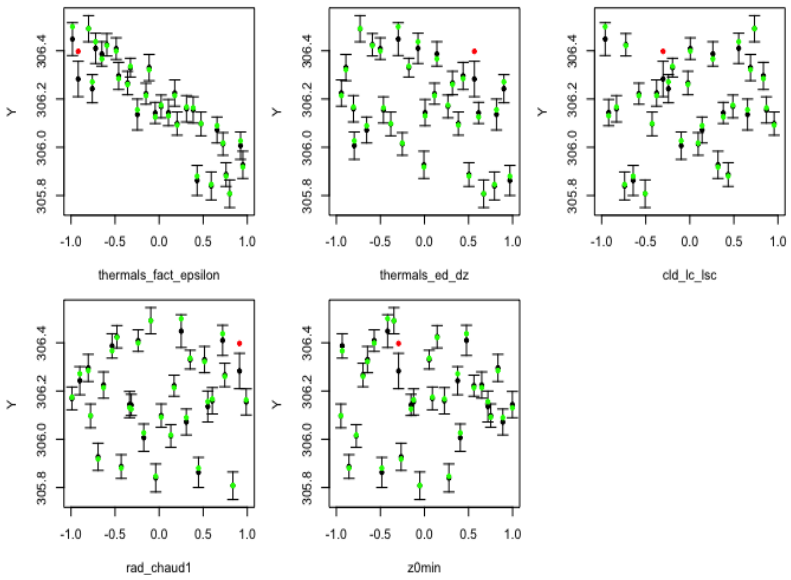

Figure 3.7: Validation plot against each of the parameters. The predictions and two standard deviation prediction intervals are in black. The true function values are in green if they lie within two standard deviation prediction intervals, or red otherwise.

3.6.3 Validation Summary Statistics

Summary statistics are commonly used to assess the performance of a GP emulator, in particular, if we are interested in performing a comparative study (see Chapter 4) (Gramacy and Lee, 2008; Ba and Joseph, 2012; Montagna and Tokdar, 2016).

The performance of GP emulators could be assessed by considering the Root Mean Squared Error (RMSE) and the Interval Score for $(1-\alpha)\times 100\%$ prediction interval (Gneiting and Raftery, 2007). The interval score with lower l and upper u

endpoints at level $\frac{\alpha}{2}$ and $1-\frac{\alpha}{2}$ quantiles generated by GP emulator is found via

$$S_{\alpha}^{int}(l,u;f(\mathbf{x})) = (u-l)+\frac{2}{\alpha}(l-f(\mathbf{x}))\mathbb{1}\{f(\mathbf{x})<l\}+\frac{2}{\alpha}(f(\mathbf{x})-u)\mathbb{1}\{f(\mathbf{x})>u\}. \quad (3.1)$$

We are interested in low values of the Interval Score by being rewarded for the narrow prediction interval and being penalized by the distance between $f(\mathbf{x})$ (true model value) and u or l if the true value is outside the prediction interval generated by the emulator. The size of the penalty depends on α. We specify $\alpha = 0.05$, as two standard deviation prediction intervals are approximately 95%.

We obtained RMSE and IS as 0.0262 and 0.162 respectively (see Appendix B.4.3 for more details on how to compute these values). Both of these values are very low and close to zero and based on these summary diagnostics together with the diagnostics plots, we conclude that the GP emulator performs satisfactorily in representing `theta500`.

3.7 Calibration with `ExeterUQ` software

As part of `ExeterUQ` software, we introduced a set of tools to perform history matching. We start by considering history matching for a single metric of interest, `theta500`. Then we proceed to present the form of multi-dimensional implausibility adopted in `ExeterUQ` and consider four metrics such as `theta500`, `qv500`, `zhneb` and `nebmax`. The method is most powerful when refocussing steps are taken (Williamson et al., 2017), and we will describe how to perform multi-wave history matching employing `ExeterUQ` tools.

3.7.1 History Matching

We have presented history matching in section 2.6.1. The general workflow of the process of history matching is based on obtaining ensemble, $\{\mathrm{X}, \mathrm{F}\}$, using them to train statistical emulator that predicts the key metric of interest with uncertainty on the prediction, and then using the emulator to rule out regions of parameter space that are "too far" from observation guided by implausibility function, $\mathcal{I}(\mathbf{x})$.

In section 3.5, we demonstrated how to construct a GP emulator, we can now use it to search for values of the model parameters that lead to "close enough" models (as defined by our uncertainties). In order to rule out regions of parameter space, we adopt the implausibility function presented in subsection 2.6.1:

$$\mathcal{I}(\mathbf{x}) = \frac{|z - E[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var[f(\mathbf{x})]}},$$

where z corresponds to observation, an expectation $E[f(\mathbf{x})]$ and variance $Var[f(\mathbf{x})]$, could be computed from the emulator, $Var[e]$ and $Var[\eta]$ are the variance of the observation error and the model discrepancy respectively.

We proceed to obtaining the Not Ruled Out Yet (NROY) space, which is the subset of parameter space, $\mathcal{X}$, for which $\mathcal{I}(\mathbf{x}) \leq a$, where a is a pre-specified threshold. Mathematically, NROY space is

$$\mathcal{X}^1 = \big\{\mathbf{x} \in \mathcal{X} : \mathcal{I}(\mathbf{x}) \leq a\big\}.$$

To obtain an NROY space, $\mathcal{X}^1$, in practice, we start by generating a 10,000 random LHC to represent the parameter space $\mathcal{X}$, at which we compute the implausibility function specified above. For this demonstration, our collaborators from HIGH-TUNE provided the observation value z set at 305.82, the observation error $Var[e]$ and model discrepancy error $Var[\eta]$ are 0.0218 and 0 respectively. We managed to produce the NROY density and minimum implausibility plots for 2D projections of the parameters, shown in Figure 3.8. These plots are similar to the ones presented by Williamson et al. (2015, 2017); Salter et al. (2018). Each panel on the upper triangle shows the proportion of parameter settings behind each pixel that are NROY. Grey regions are completely ruled out. From NROY density plots, we could observe a strong relationship between two parameters `thermals_fact_epsilon` and `thermals_ed_dz`.

The lower triangle shows minimum implausibility plots. We plot the value of the smallest implausibility found in each pixel. For comparative purposes, the plots have the same orientation as those on the upper triangle. Green and yellow areas of this

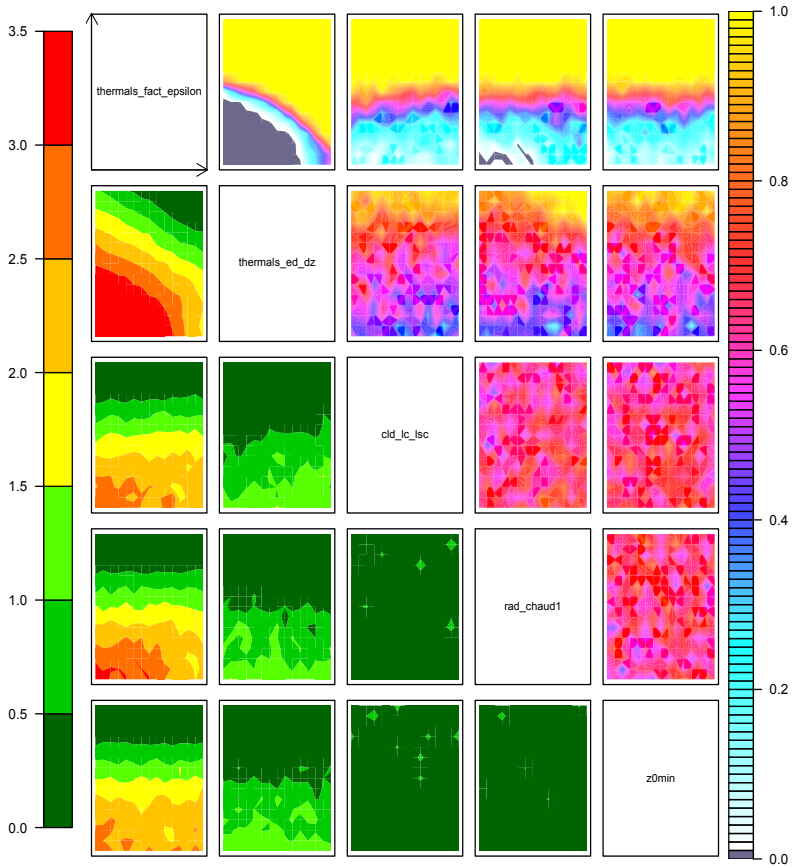

Figure 3.8: NROY density plots (*upper triangle*) and minimum implausibility plots (*lower triangle*) for `Xp` of NROY space. Each panel plots either NROY density or minimum implausibility for a pair of parameters. NROY densities, for each pixel on any panel in the *upper triangle*, represent the proportion of points in `Xp` behind that pixel that are NROY and are indicated by the colour whose scale is indicated on the right. Grey coloured regions are completely ruled out. Minimum implausibilities, for each pixel on any panel on the *lower triangle* of the picture, represent the smallest implausibility found in `Xp`. These plots are oriented the same way as those on the *upper triangle*, for the ease of visual comparison.

plot would indicate the location of potentially "good" settings of model parameters. We would look to further explore the green and yellow areas of these plots as these areas correspond to NROY space after wave1. By performing a single wave, we have managed to cut out parameter space and achieve an NROY space of size 64.01% of original input space, $\mathcal{X}$. The relationship between input parameters depicted in the parameter plots were expected and in line with modellers' knowledge and beliefs about the model behaviour.

3.7.2 Multi-dimensional implausibility

It is common for modellers to consider a collection of metrics for tuning a model or a model sub-component (Williamson et al., 2017). In subsection 2.6.1, we discussed different forms of multivariate implausibility function. In this section, we adopt a simplified form of the multivariate implausibility function proposed by Craig et al. (1997) to construct implausibility per each output, i.e. $\mathcal{I}_j(\mathbf{x}), j = 1, \ldots, l$, separately. We use this simplified form of multivariate implausibility function since we are unable to specify the correlation structure between a collection of metrics. This approach provides our users with the flexibility to rule out any parameter settings that fail to meet either all of the targets or most. We start by generating a 10,000 random LHC in the parameter space $\mathcal{X}$, at which we evaluate the implausibility function per output. In particular, we return to a collection of metrics that we emulated in subsection 3.5.1, i.e. `theta500`, `qv500`, `zhneb` and `nebmax`. The information on the values of observation, observation and model errors are provided in Table 3.2.

metric	z	$Var[e]$	$Var[\eta]$
`theta500`	305.82	0.00218	0
`qv500`	16.18	0.0492	0
`zhneb`	1522	1729	0
`nebmax`	0.0930	0.00019	0

Table 3.2: HIGH-TUNE model information necessary for history matching.

Modellers have the flexibility to decide which form of implausibility measure they want to choose to obtain the NROY space. For instance, our users could employ the maximum implausibility defined as

$$\mathcal{I}_M = \arg\max_j \mathcal{I}_j(\mathbf{x}), j = 1, \ldots, 4$$

This measure is considered to be conservative since it requires all outputs of the computer model to be consistent with the observation to be part of NROY space (Craig et al., 1997).

We could obtain the NROY space using $\mathcal{I}_M$ measure, which is defined as

$$\mathcal{X}_{NROY} = \left\{\mathbf{x} \in \mathcal{X} : \mathcal{I}_M(\mathbf{x}) \leq a\right\}$$

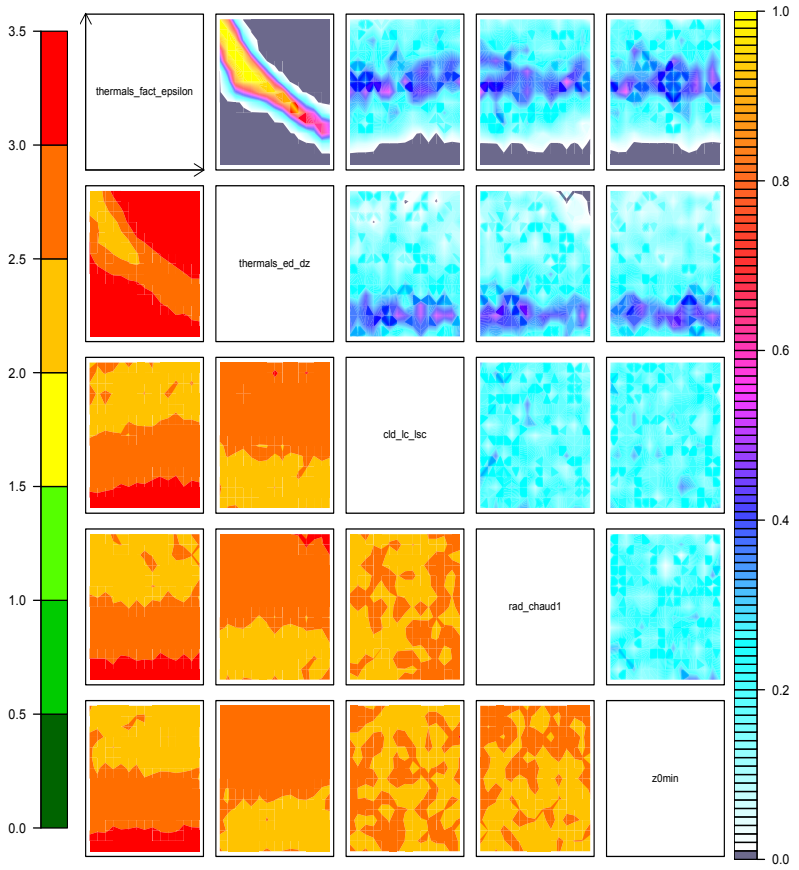

Figure 3.9: NROY density plots (*upper triangle*) and minimum implausibility plots (*lower triangle*) for `Xp` of NROY space with maximum implausibility, $\mathcal{I}_M$. Each panel plots either NROY density or minimum implausibility for a pair of parameters. NROY densities, for each pixel on any panel in the *upper triangle*, represent the proportion of points in `Xp` behind that pixel that are NROY and are indicated by the colour whose scale is indicated on the right. Grey coloured regions are completely ruled out. Minimum implausibilities, for each pixel on any panel on the *lower triangle* of the picture, represent the smallest implausibility found in `Xp`. These plots are oriented the same way as those on the *upper triangle*, for the ease of visual comparison.

The NROY density and minimum implausibility plots for 2D projections of the parameters, shown in Figure 3.9, are produced in a similar way to the one that we discussed in subsection 3.7.1. We observe a strong linear relationship between two input parameters `thermals_fact_epsilon` and `thermals_ed_dz`, which also indicates the importance of these two parameters for the tuning. By performing a single wave with $\mathcal{I}_M$ measure, we have managed to cut out parameter space and achieve an NROY space of size 19.5% of the original input space. The details of computing maximum implausibility measure and obtaining NROY density plots and minimum implausibility plots in Figure 3.9 using `ExeterUQ` are provided in the Appendix

B.5.2.

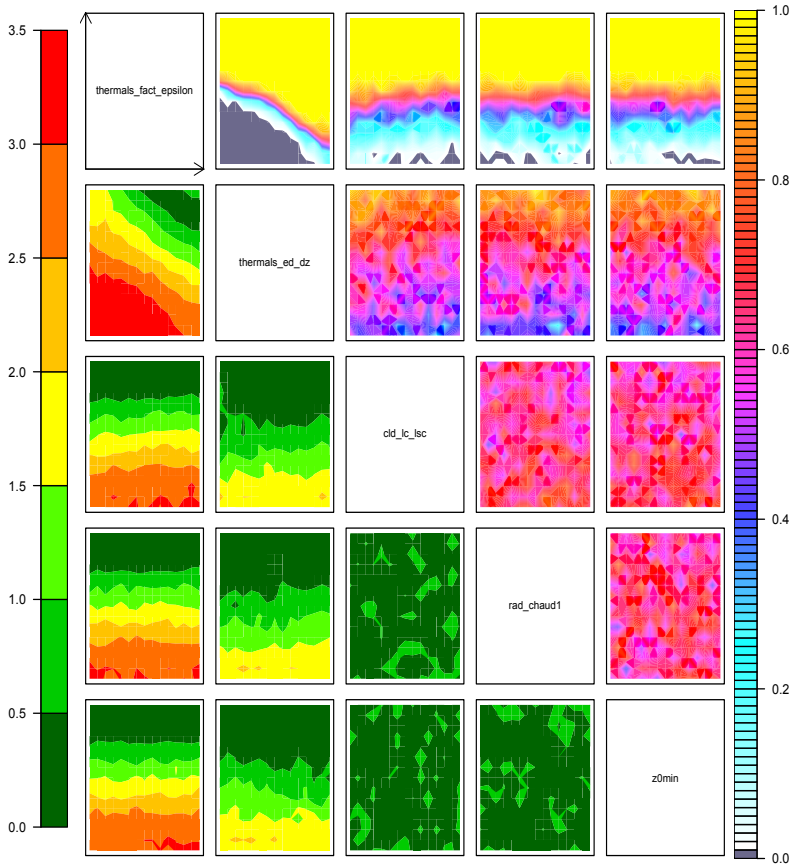

Figure 3.10: NROY density plots (*upper triangle*) and minimum implausibility plots (*lower triangle*) for `Xp` of NROY space with second-highest implausibility, $\mathcal{I}_{2M}$. Each panel plots either NROY density or minimum implausibility for a pair of parameters. NROY densities, for each pixel on any panel in the *upper triangle*, represent the proportion of points in `Xp` behind that pixel that are NROY and are indicated by the colour whose scale is indicated on the right. Grey coloured regions are completely ruled out. Minimum implausibilities, for each pixel on any panel on the *lower triangle* of the picture, represent the smallest implausibility found in `Xp`. These plots are oriented the same way as those on the *upper triangle*, for the ease of visual comparison.

A less sensitive measure, the j^{th} maximum implausibility, could be used to rule out input parameter space, that was presented and discussed in detail in subsection 2.6.2. We consider the second-highest implausibility for each $\mathbf{x}$, $\mathcal{I}_{2M}(\mathbf{x})$, so that if two or more metrics are more than a standard deviations away from the observations for some parameter choice $\mathbf{x}$, the choice is ruled out. In this case, we define an NROY space as

$$\mathcal{X}_{NROY} = \{\mathbf{x} \in \mathcal{X} : \mathcal{I}_{2M} \leq a\}.$$

We observe from Figure 3.10 that less input space has been cut off using the $\mathcal{I}_{2M}$ implausibility measure.

We could deduct from NROY density plots from Figure 3.10 that we have retained the positive right-hand corner of parameter settings for `thermals_fact_epsilon` and `thermals_ed_dz` contrary to NROY density plots from Figure 3.9. The NROY space using second maximum implausibility measure is 63.25% of original input space.

3.7.3 Refocussing

We have described the process of deriving NROY space $\mathcal{X}^1$, i.e. performing wave 1, with `ExeterUQ` tools in subsection 3.7.1. However, the method is most powerful when refocussing steps are taken, within NROY space, a new ensemble is run, and the procedure is repeated (Williamson et al., 2017). In subsection 2.6.3, we provided a number of reasons why refocusing is powerful such as an increase in the density of ensemble at later waves leading to the improvement in the performance of our statistical emulators and narrowing the search for potentially good models.

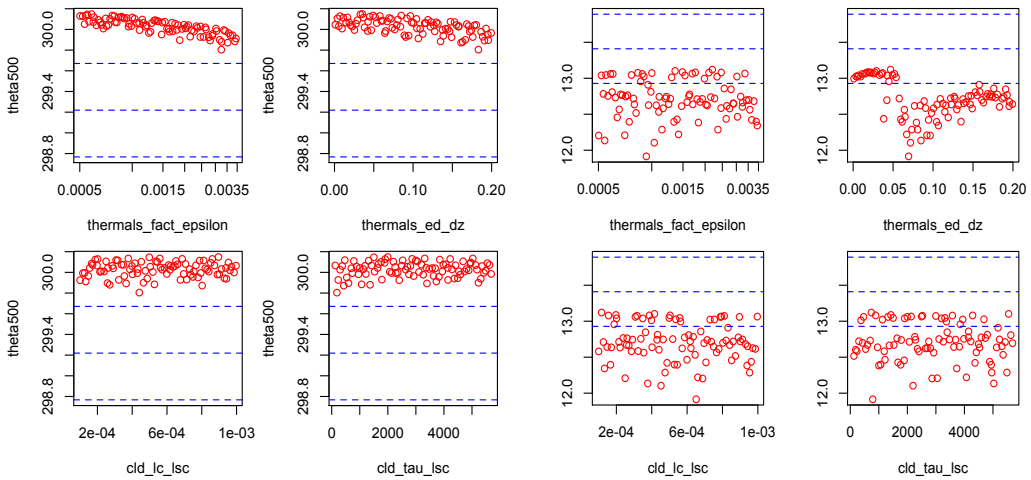

Figure 3.11: *Left*: `theta500` response from BOMEX/REF case against four inputs on the original scale. *Right*: `qv500` response from SANDU/REF case against four inputs on the original scale. The blue dashed lines correspond to z plus and minus $2(Var[e] + Var[\eta])^{1/2}$, where $Var[e]$ is the variance of the observation error and $Var[\eta]$ is the model discrepancy error.

The process of performing iterative refocusing using `ExeterUQ` software is explicitly described in the Appendix B.5.3. An important remark is that at the moment

the design points for wave k of history matching are selected randomly from $\mathcal{X}^{k-1}$. Obtaining the design for iterative refocusing is an open research question, and we present our design approach in Chapter 5 and Chapter 6.

As well as considering multi-metrics for a single case at a time, our collaborators from HIGH-TUNE are interested in exploring a combination of metrics from different cases together. For a pedagogical purpose, we consider `theta500` from BOMEX/REF case and `qv500` from SANDU/REF for refocussing. Both cases, BOMEX/REF and SANDU/REF are part of 12 selected 1D cases that cover the main boundary layer clouds regimes in real-time and described in section 3.2.

We have performed three waves of history matching, and at each iteration of refocusing, we specified cutoff value, a, at three and used maximum implausibility measure, $\mathcal{I}_M$, to derive the NROY space. We have also set the model discrepancy value for both metrics of interest at 0.05.

After performing wave 1 of history matching, we have managed to cut down space to 4.12% of original input space, $\mathcal{X}$. The NROY space after wave 2 is 2.72% of the original input space. Performing wave 3 does not provide us with a significant reduction in NROY space, i.e. the NROY space after wave 3 is 2.53%. We could have decreased the value of model discrepancy to observe a further reduction in the NROY space.

From Figure 3.12 we observe that after performing three consecutive waves of history matching we have managed to cut the original input space, $\mathcal{X}$, down to a positive corner of values of parameters `thermals_fact_epsilon` and `thermals_ed_dz`. This result is supported by scatter plots from Figure 3.11, i.e. most of the variability in `theta500` is explained by parameters `thermals_fact_epsilon` and `thermals_ed_dz`, while the variability in response `qv500` is explained by `thermals_ed_dz`.

It is important to mention that we had some difficulties in constructing a GP emulator to represent `qv500` using our `ExeterUQ` software tools. In particular, we have performed LOO diagnostics to check the performance of emulator for `qv500`. The LOO diagnostics plot in Figure 3.13 demonstrates that our "out of the box" GP emulator fails to model the variability of the response against inputs, especially

against `thermals_ed_dz`. Our GP emulator is under-confident for `thermals_ed_dz` $<$ -0.5 and `thermals_ed_dz` $>$ 0.25, while it is overconfident in the input region $-0.5 <$ `thermals_ed_dz` < 0.25.

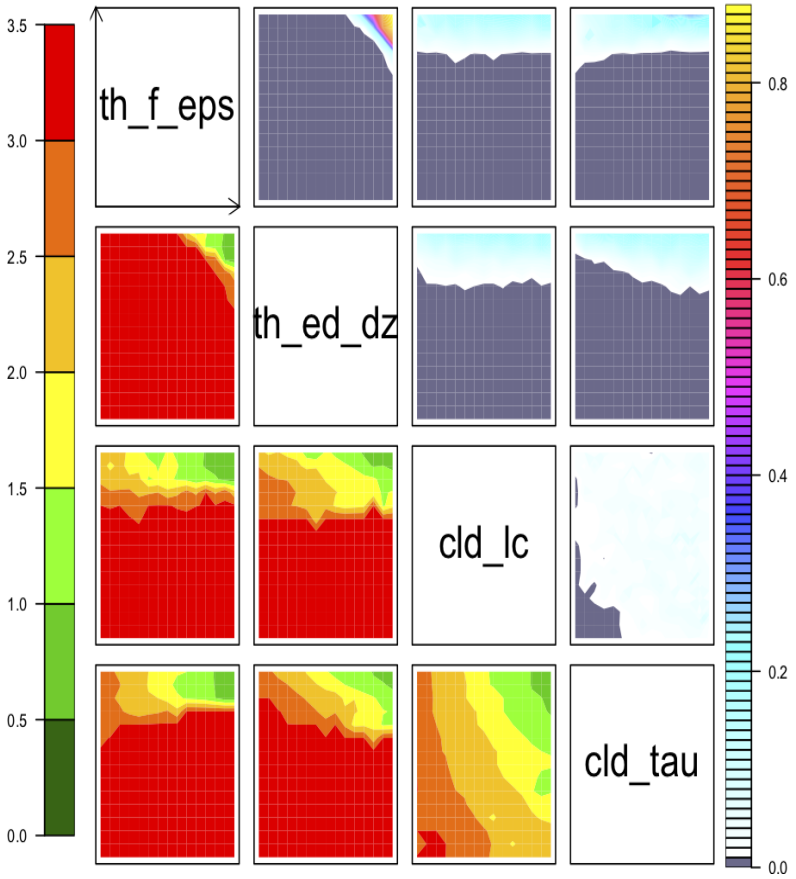

Figure 3.12: NROY density plots (*upper triangle*) and minimum implausibility plots (*lower triangle*) of the wave 3 NROY space for all four parameters. Each panel plots either NROY density or minimum implausibility for a pair of parameters. NROY densities, for each pixel on any panel in the *upper triangle*, represent the proportion of points in `Xp` behind that pixel that are NROY and are indicated by the colour whose scale is indicated on the right. Grey coloured regions are completely ruled out. Minimum implausibilities, for each pixel on any panel on the *lower triangle* of the picture, represent the smallest implausibility found in `Xp`. These plots are oriented the same way as those on the *upper triangle*, for the ease of visual comparison.

Based on LOO diagnostics, we generated the standardized errors plots. Figure 3.14 demonstrates the unusual behaviour of standardized errors, i.e. heteroscedasticity of residuals against `thermals_ed_dz`. Such unusual behaviour is often observed with SCM outputs.

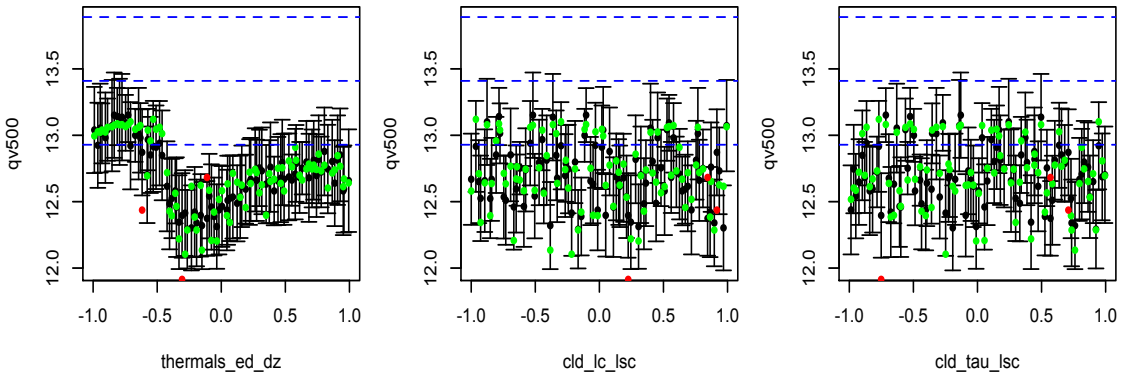

Figure 3.13: LOO diagnostics plot against each of the parameters. The predictions and two standard deviation prediction intervals are in black. The true function values are in green if they lie within two standard deviation prediction intervals, or red otherwise. The blue dashed lines correspond to z plus and minus $2(Var[e] + Var[\eta])^{1/2}$, where $Var[e]$ is the variance of the observation error and $Var[\eta]$ is the model discrepancy variance.

3.8 Future planned development. Conclusion

We have presented the application and use of `ExeterUQ` in the context of climate modelling. Currently, `ExeterUQ` is actively used as part of HIGH-TUNE project to assist the development of new parameterization schemes for global climate model. However, our developed software is not limited to this single application and could be employed by a broader modelling community. The focus of this Chapter was mainly to present the functionality of `ExeterUQ`. It is easily noticeable that the variance of model discrepancy was set at zero in all of the examples considered. However, since the process of history matching is very fast and automatic thanks to the `ExeterUQ` software, modellers have a power and flexibility to experiment scientifically with their model discrepancy specification.

There is a wide range of improvements planned by the UQ research group. Firstly, a number of UQ tools such as sensitivity analysis and uncertainty analysis are still under development. Secondly, it might be interesting to define a new class, for instance, `gpsurrogate` or `gpemulator`, that corresponds to an obtained GP emulator. This will ensure that we are dealing with objects of a certain structure. We also could adapt generic functions such as `plot` to our new class type. Class definition is commonly used in many base and stats packages available in `CRAN`. For

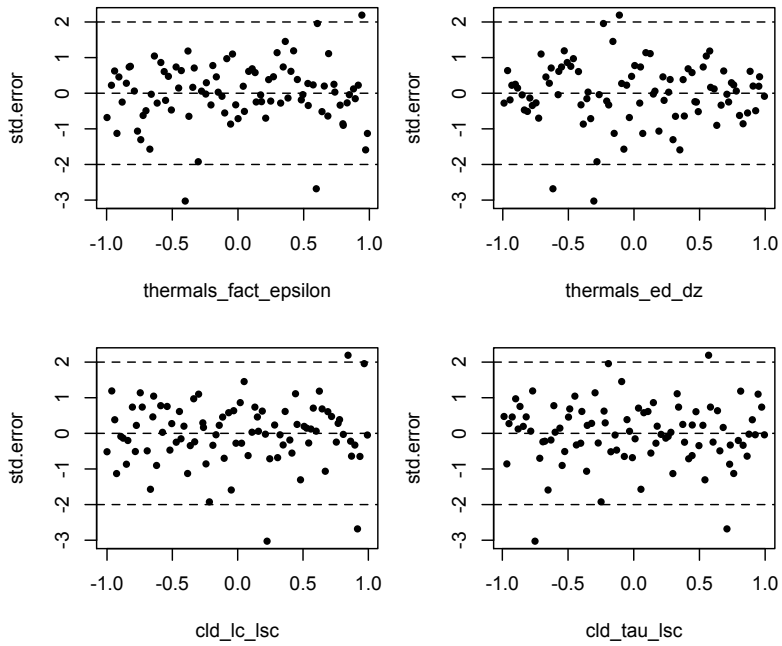

Figure 3.14: Individual standardized errors obtained for the design against each of the parameter.

instance, `bgp`, the `R` function used to construct a GP model in `tgp` package, returns an object of class "tgp" (Gramacy et al., 2007).

Two very important methodological questions have been raised during our collaboration with climate modellers. Firstly, we have observed that response generated by some of the metrics of interest exhibit nonstationary behaviour across the input space, as demonstrated in subsection 3.7.3. Interestingly, the LOO diagnostics plots indicated the failure of stationary GP emulator as well as pictured the unusual relationship between standardized errors based on LOO diagnostics and the input parameters. In Chapter 4, we propose a new diagnostic-led approach to fitting nonstationary GP emulators to model nonstationary model response. Secondly, we recognise that generating a design for wave k based on the random sampling in NROY space is not efficient. In Chapter 5, we present a Bayesian Design Criterion for obtaining a new ensemble for refocusing.

Chapter 4

Nonstationary Gaussian Process Emulators with Kernel Mixtures

4.1 Introduction

In Chapter 3, we finished with an example of how a standard GP emulator constructed with our software failed to capture the nonstationary behaviour of a climate model. This observation has led to the development of a new method for incorporating nonstationary features in Gaussian Process (GP) emulators that could be applied to a large class of complex computer models, including climate models.

We start by defining a nonstationary complex computer model, $f(\cdot)$, as one for which the model response varies significantly across the input space. For example, when the variability in the model response changes with the changes in input values, or the model response exhibits sharp localized features, e.g. a discontinuity or tall peaks (Montagna and Tokdar, 2016). Examples of such models were mentioned in subsection 2.5. In this Chapter, as part of our comparative study, we are going to consider a nonstationary function, the "wavy" function, presented by Ba and Joseph (2012). The "wavy" function is defined as

$$f(x_1, x_2) = \sin\big(1/(0.7x_1 + 0.3)(0.7x_2 + 0.3)\big)$$

with $x_1, x_2 \in [0, 1]$. It fluctuates rapidly when x_1 and x_2 is small, but gradually

becomes smooth as x_1 and x_2 increases towards 1 (see Figure 4.1).

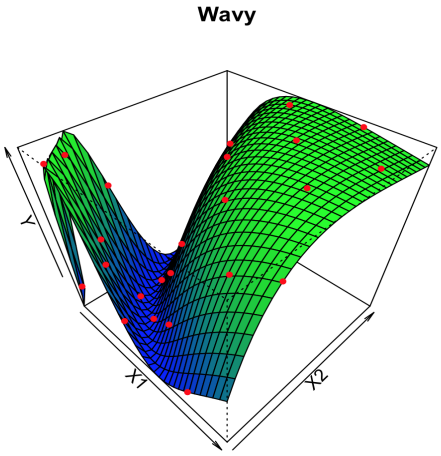

Figure 4.1: True function for the two-dimensional numerical example and 24-run maximin distance LHD red points used as design.

Research on computer emulation has largely focused on stationary GP models, where the distance between $\mathbf{x}$ and $\mathbf{x}'$ mainly determines the similarity between $f(\mathbf{x})$ and $f(\mathbf{x}')$. Applying this class of GP emulators to model nonstationary responses leads to the poor inference. When the nonstationarity is not adequately captured, prediction intervals produced by emulators can be too narrow in the region of a high variability of f (the emulator is over-confident). On the contrary, the emulator is under-confident in the input space where f is "well-behaved" by producing too large prediction intervals (Bastos and O'Hagan, 2009; Oughton and Craig, 2016).

Figure 4.2 demonstrates the performance of a stationary GP emulator for the "wavy" function described above. The top left panel displays the predictions generated by our ExeterUQ software (see Appendix B.4.2) onto a 30×30 grid over x_1 and x_2 ($x_1 \in [0, 1]$ and $x_2 \in [0, 1]$). From the top left panel, we observe that the mean surface is too rough in the highly smooth area for x_1 and x_2 large. The top right panel shows predictions and two standard deviation prediction intervals for the line $x_1 = x_2$. From this plot, we observe overconfidence in the region where f fluctuates rapidly, for small values of x_1 and x_2, and underconfidence in the region where the function is relatively smooth, for large values of x_1 and x_2. The diagnostic plots, presented in section 2.4 and shown in the lower left and right panels of Figure 4.2, are indicative of problems we commonly encounter when building emulators

in practice, in particular we observe heteroskedasticity in the plot of standardized errors (bottom right). The lower left panel demonstrates the Leave One Out (LOO) diagnostics plot against input x_1.

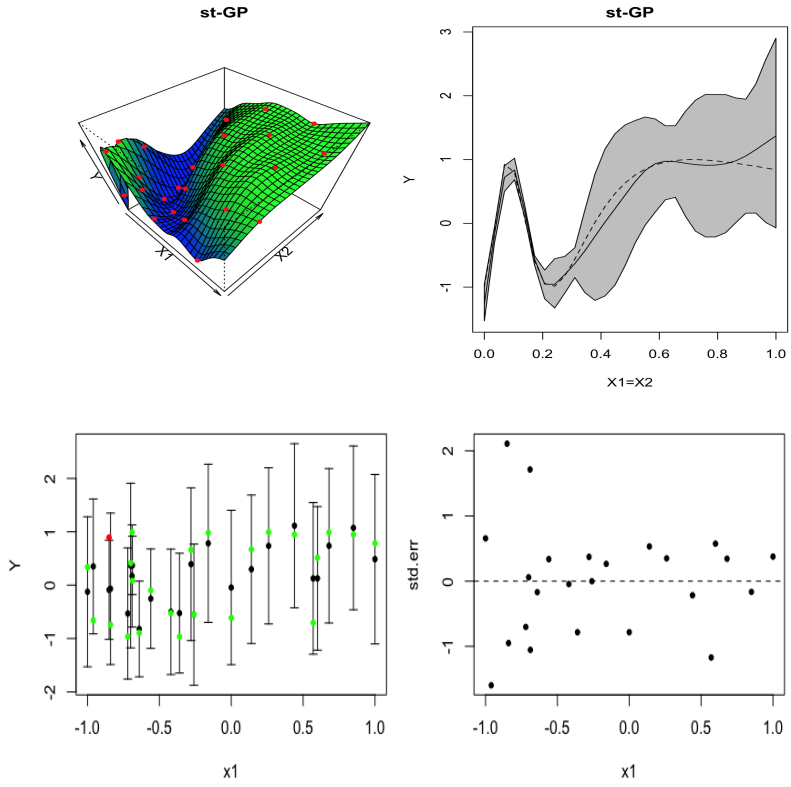

Figure 4.2: Failure of stationary GP emulator. *Top left*: Posterior mean predictive surface produced by stationary GP emulator with 24 design points in red. *Top right*: Emulator performance for the cross section $x_1 = x_2$. The dashed line is the true function value, the solid black line is the posterior mean predictive curve, and the grey areas denote two standard deviation prediction intervals. *Bottom left*: Leave One Out diagnostic plot against x_1. The predictions and two standard deviation prediction intervals are in black. The true function values are in green if they lie within two standard deviation prediction intervals, or red otherwise. *Bottom right*: Individual standardized errors of emulator predictions against x_1.

Figure 4.2 demonstrates that the stationary GP emulator fails to provide a fair assessment of uncertainty for a nonstationary response. Interestingly, we could also observe from the lower right panel of Figure 4.2 that there are two distinct, identifiable regions of standardized errors' behaviour. For $x_1, x_2 < -0.5$, the standardized errors exhibit large variability, i.e. ranging from -1.5 to 2, while for input values greater than -0.5 the standardized errors' variability significantly decreases. We observed that standardized Leave One Out (LOO) diagnostics for a stationary GP emulator fitted to the training data could be informative of the distinct regions

of model response behaviour in the input space in the case where the model response is nonstationary. This finding has led to the development of a new method of modelling nonstationarity in response that will be presented in this Chapter. We partition the input space using diagnostics from initial stationary GP fits to develop a single nonstationary GP emulator with a flexible mixture kernel obtained via the partition. Firstly, we perform standardized Leave One Out (LOO) diagnostics for a stationary GP emulator fitted to our training data. We specify a finite mixture model for the standardized LOO errors to identify L distinct regions of behaviour in the input space. We then assign a Gaussian process for f whose covariance kernel is a mixture of L stationary covariance kernels, each belonging to the L previously identified regions. This approach allows us to fit a single Gaussian process and operate within the original input space.

A number of methods that aim to model a nonstationary response with modified GP models have appeared in the UQ literature. One approach is to partition the input space and fit separate GPs for each partition. For instance, Treed Gaussian Process (TGP) (Gramacy and Lee, 2008) employs treed partitioning and makes splits on the value of a single variable. Several works aim to deal with the nonstationarity through separating the model response in terms of a global, large-scale behaviour, and a locally stationary process. The composite Gaussian Process (CGP) (Ba and Joseph, 2012) uses GP with longer length scale to capture the global trend throughout the input space. A flexible variance model is introduced to capture varying volatility across the input space. We compare the performance of our proposed nonstationary GP emulator against TGP and CGP as we consider these two models to be the representations of these two approaches adopted in UQ literature to deal with nonstationarity in response.

The Chapter has the following structure. Section 4.2 demonstrates in detail the approaches to model nonstationarity that explicitly attempt to identify separate regions of response behaviour in input space using partition model (meta-model) or diagnostics. Section 4.3 introduces our nonstationary GP emulator. Section 4.4 examines the performance of our nonstationary GP emulator for a number of

idealised numerical examples designed to demonstrate performance in different types of scenarios. In section 4.5, we discuss prior choices for GP hyperparameters. In section 4.6, we apply our methodology to the boundary layer of the single column ARPEGE-Climat model as part of HIGH-TUNE. ARPEGE-Climat is developed at the Centre National de Recherches Météorologuques (CNRM) and is the atmospheric component of the CNRM climate model. This is an updated version compared to the one described in Voldoire et al. (2013) (see also Abdel-Lathif et al. (2018) for further details). Section 4.7 demonstrates the importance of nonstationary GP emulators for history matching performed on one of the idealised numerical examples. Section 4.8 contains a conclusion.

4.2 Approaches in modelling nonstationarity in the literature

An explicit review of nonstationary GP models is provided in section 2.5. In this Chapter, we consider in detail methods that explicitly attempt to identify and model the nonstationarity in a model response by either using a partition model (meta-model) or a diagnostic measure. We view our proposed GP model as a hybrid of these approaches. In particular, our model borrows the strength of input space partitioning similar to TGP and Voronoi tessellation GP and kernel choice to provide a single GP model that adapts to different behaviour similar to CGP.

Gramacy and Lee (2008) used treed partitioning to divide the input space prior to fitting different base models, such as linear or GP models, to data, independently in each region. In simple terms, treed partitioning divides up the input space by making binary splits on the value of a single input variable. Gramacy and Lee (2008) define $\mathcal{T}$ as a decision tree and $\eta \in \mathcal{T}$ as a leaf node, representing the region of the input space with characteristic response behaviour. A split occurs with probability $a(1+q_\eta)^{-b}$ where q_η is the depth of $\eta \in \mathcal{T}$, the path from tree root to the leaf η, a and b are user-defined parameters to control the size and spread of the distribution of trees. As the depth of the leaf node increases, the probability of

splitting decreases, penalizing large decision trees for modelling data. The splitting dimension, u, is sampled from a discrete uniform distribution and the split location, s, is chosen uniformly in the u^{th} dimension. The decision tree is constructed with the following traditional operations: grow, prune, change and swap (Chipman et al., 1998). Gramacy and Lee (2008) added a rotate operation to encourage better mixing of MCMC methods and escape local minima in the marginal posterior of $\mathcal{T}$. Treed partitioning is a computationally fast approach and works well for modelling certain discontinuities in model responses. However, it is restricted to axis-aligned partitions and introduces discontinuities across the partition boundaries of $\mathcal{T}$.

As an alternative to treed partitioning, Voronoi tessellation could be used to partition the input space. A Voronoi tessellation Gaussian process uses the Euclidean distance from a set of centres to create Voronoi cells and construct an individual Gaussian process model within each cell (Kim et al., 2005). Pope et al. (2018) extended the method by allowing non-convex and disconnected input regions to be considered. In this case, Voronoi cells are not required to share a vertex to be in the same region, implying that the same GP model could be specified for disconnected cells. The input space $\mathcal{X}$ is partitioned into r disjoint regions $\boldsymbol{\mathcal{R}} = \left\{\mathcal{R}_1, \ldots, \mathcal{R}_r\right\}$ with $\mathcal{R}_i \subseteq \mathcal{X}$ for $\forall i \in \left\{1, \ldots, r\right\}$ and $\cup_{i=1}^{r} \mathcal{R}_i = \mathcal{X}$. A set of centres $\mathbf{x}_T^* = \left\{\mathbf{x}_{T_1}^*, \ldots, \mathbf{x}_{T_k}^*\right\}$ is defined for k cells of a Voronoi tessellation. A point $\mathbf{x} \in \mathcal{X}$ is contained in the cell of the ith centre $\mathbf{x}_{T_i}^*$ if

$$||\mathbf{x} - \mathbf{x}_{T_i}^*|| < ||\mathbf{x} - \mathbf{x}_{T_j}^*|| \quad \forall j \in \left\{1, \ldots, k\right\} \backslash i,$$

where $||\mathbf{x} - \mathbf{x}_{T_j}^*||$ is the Euclidean distance between $\mathbf{x}$ and $\mathbf{x}_{T_j}^*$. Pope et al. (2018) denote $\boldsymbol{c}_i$ as the indices of all cell centres in R_i. The collection of tessellation parameters is defined as $\boldsymbol{t} = \left\{\mathbf{x}_T^*, k, r, \boldsymbol{c}\right\}$ with the prior specification. A proper prior specification is crucial to resolve the potential identifiability issues since a region in one model which consists of multiple cells joined together can be equivalent to a region in another model consisting of a single cell. The Voronoi tessellation is derived with the birth, death and move operations to update the set of centres and change operation to update the relationship between centres. The acceptance ratio is used

to proceed with one of the moves. Voronoi tessellation provides users with significant flexibility by producing input space partitions of complex shapes. However, it also introduces the extra complexity due to the use of RJ-MCMC, which requires a larger number of MCMC chains and iterations per chain to ensure convergence.

Partitioning approaches can be very effective as, when fitting diagnostics, we often "see" regions of different behaviour, as we could observe from the standardized error plot in Figure 4.2, and it can be intuitive for the modellers to think of the model having different characteristics in different input regions. However, we do not believe the implied boundary discontinuities that these models specify.

The Composite Gaussian Process (CGP) model was introduced in section 2.5. CGP consists of two processes, in particular

$$f(\mathbf{x}) = Z_{global}(\mathbf{x}) + \sigma^2(\mathbf{x}) Z_{local}(\mathbf{x}), \tag{4.1}$$

with $Z_{global}(\mathbf{x}) \sim GP\big(\mu, \tau^2 r(\cdot,\cdot;\boldsymbol{\phi}_g)\big)$, $Z_{local}(\mathbf{x}) \sim GP\big(0, r(\cdot,\cdot;\boldsymbol{\phi}_l)\big)$ and variance model $\sigma^2(\mathbf{x}) = \sigma^2 v(\mathbf{x})$. Both $r(\mathbf{x},\mathbf{x};\boldsymbol{\phi}_g)$ and $r(\mathbf{x},\mathbf{x}';\boldsymbol{\phi}_l)$ are Gaussian correlation functions described in detail in section 2.5.

Interestingly, Ba and Joseph (2012) advocate the use of squared residuals to model the volatility in response, i.e. $\sigma^2(\mathbf{x})$. Firstly, the expectation of the conditional distribution of f at a new input point, $\mathbf{x}$, given the ensemble, $\{\mathrm{X}, \mathrm{F}\}$, and model parameters is separated into global and local components, i.e.

$$E[f(\mathbf{x})|\{\mathrm{X},\mathrm{F}\},\lambda,\boldsymbol{\phi}_g,\boldsymbol{\phi}_l,b] = m^*_{global}(\mathbf{x}) + m^*_{local}(\mathbf{x})$$

$$m^*_{global}(\mathbf{x}) = \hat{\mu} + r(\mathbf{x},\mathrm{X};\boldsymbol{\phi}_g)\Big(G + \lambda\Sigma^{1/2}L\Sigma^{1/2}\Big)^{-1}(\mathrm{F} - \hat{\mu}\mathbf{1})$$

$$m^*_{local}(\mathbf{x}) = \lambda v^{1/2}(\mathbf{x}) r(\mathbf{x},\mathrm{X};\boldsymbol{\phi}_l)\Sigma^{1/2}\Big(G + \lambda\Sigma^{1/2}L\Sigma^{1/2}\Big)^{-1}(\mathrm{F} - \hat{\mu}\mathbf{1}),$$

where $\lambda = \sigma^2/\tau^2$ is the ratio of global and local variances respectively, $\mathbf{1}$ denotes a $n \times 1$ vector where each entry is equal to one, $r(\mathbf{x},\mathrm{X};\boldsymbol{\phi}_g)$ is $1 \times n$ vector whose ith entry is $r(\mathbf{x},\mathbf{x}_i;\boldsymbol{\phi}_g)$, and G is $n \times n$ matrix with entries $G_{ij} = r(\mathbf{x}_i,\mathbf{x}_j;\boldsymbol{\phi}_g)$. These components correspond to the global part of the statistical model. The components

that correspond to the local adjustment part of the model are $r(\mathbf{x}, \mathrm{X}; \boldsymbol{\phi}_l)$, a $1 \times n$ vector whose ith entry is $r(\mathbf{x}, \mathbf{x}_i; \boldsymbol{\phi}_l)$, and L, a $n \times n$ matrix with entries $L_{ij} = r(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\phi}_l)$. The remaining terms $\hat{\mu}$ and Σ are defined as

$$\hat{\mu} = \left(\mathbf{1}^T (G + \lambda \Sigma^{1/2} L \Sigma^{1/2})^{-1} \mathbf{1}\right)^{-1} \mathbf{1}^T (G + \lambda \Sigma^{1/2} L \Sigma^{1/2})^{-1} \mathrm{F},$$
$$\Sigma = diag\big\{v(\mathbf{x}_1), \ldots, v(\mathbf{x}_n)\big\}.$$

The volatility in the response is primarily modelled through a standardized volatility function $v(\mathbf{x})$. To model a standardized volatility function, $v(\mathbf{x})$, for a given global trend, the squared residuals are obtained, $s_i^2 = (f(\mathbf{x}_i) - m^*_{global}(\mathbf{x}_i))^2$. Based on squared residuals, the volatility function is defined as

$$v(\mathbf{x}) = \frac{\sum_{i=1}^n r(\mathbf{x}, \mathbf{x}_i; \boldsymbol{\phi}_g, b) \times s_i^2}{\sum_{i=1}^n r(\mathbf{x}, \mathbf{x}_i; \boldsymbol{\phi}_g, b)}$$

where $r(\mathbf{x}, \mathbf{x}'; \boldsymbol{\phi}_g, b) = \exp\big\{ - b \sum_{i=1}^p \phi_{gi} (x_i - x_i')^2 \big\}$. The correlation parameters, $\boldsymbol{\phi}_g$, were used in the global trend specification, and $b \in [0, 1]$ is an extra bandwidth parameter. By increasing b we strengthen the global trend correlation between the points used in the specification of $r(\cdot, \cdot; \boldsymbol{\phi}_g, b)$ and as a result place more weight on the local part, $m^*_{local}(\mathbf{x})$, in the model. Also greater effect of the local adjustment part is achieved in the input space where the global trend model fails, which is identified through the increase in the computed squared residuals for design points X.

We do view approaches to fit complex "global" mean functions to be appropriate as they offer interpretability, but still, often find more complex nonstationarity in the residual so that methods like CGP are not as effective. In particular, the local adjustment part of the CGP fails to capture variability in the response, as demonstrated in section 4.4.

4.3 Nonstationary GP through mixtures of stationary processes

When fitting an emulator in practice, we would typically begin by fitting a stationary Gaussian Process and examining diagnostics to assess whether the emulator was sufficient. Possible failure of the stationary assumption can then be checked from the plots of standardized errors against the model inputs (Bastos and O'Hagan, 2009). We may notice, as we do in the plot of standardized errors in Figure 4.2, that the model is "well-behaved" in some regions of the input space but not in the others. For example, the standardized errors are close to zero when x_1 and x_2 close to 1, yet the model changes rapidly, and the standardized errors are large for small values of x_1 and x_2. Approaches such as TGP explicitly model these as regions of distinct behaviour by axis-partitioning of the input space and fitting distinct GP models to each region. Our approach captures the distinct regional behaviours we see in stationary diagnostics, yet uses input-dependent mixing functions to ensure a continuous covariance kernel. We proceed to develop this approach below.

4.3.1 Model definition

Suppose, upon examining the diagnostics of a stationary GP emulator, we identify L input regions of distinct model behaviour, $\mathcal{X}_l, l = 1, \ldots, L$ (see subsection 4.3.2 for our method for identifying these regions and the optimal number of these regions L) such that $\cup_{l=1}^{L} \mathcal{X}_l = \mathcal{X}$. We define $f(\mathbf{x})$ as:

$$f(\mathbf{x}) = h(\mathbf{x})^T \boldsymbol{\beta} + \sum_{l=1}^{L} \lambda_l(\mathbf{x}) \epsilon_l(\mathbf{x}) + \sum_{l=1}^{L} z_l(\mathbf{x}) \nu_l(\mathbf{x}), \tag{4.2}$$

where $\lambda_1(\mathbf{x}), \ldots, \lambda_L(\mathbf{x})$ are input-dependent mixture components on the unit simplex, i.e. $\sum_{l=1}^{L} \lambda_l(\mathbf{x}) = 1$. Here $\epsilon_l(\mathbf{x})$ are independent, mean zero, Gaussian processes with covariance kernel, $k_l(\cdot, \cdot; \sigma_l^2, \boldsymbol{\delta}_l)$, and region-specific parameters σ_l^2 and $\boldsymbol{\delta}_l = (\delta_{1l}, \ldots, \delta_{pl})$, so that

$$\epsilon_l(\mathbf{x}) | \sigma_l^2, \boldsymbol{\delta}_l \sim GP\Big(0, k_l(\cdot, \cdot; \sigma_l^2, \boldsymbol{\delta}_l)\Big).$$

The final term of equation (4.2) is a nugget process term. We define a region specific nugget process term for each $\mathcal{X}_l, l = 1, \ldots, L$

$$\nu_l(\mathbf{x}) \sim N(0, \tau_l^2), \quad z_l(\mathbf{x}) = \mathbb{1}\big(\lambda_l(\mathbf{x}) = \max_k \big\{\lambda_k(\mathbf{x})\big\}\big).$$

We define $z_l(\mathbf{x})$ as an indicator function of the form

$$z_l(\mathbf{x}) = \begin{cases} 1, & \lambda_l(\mathbf{x}) = \max_k \big\{\lambda_k(\mathbf{x})\big\}, \\ 0, & \text{otherwise.} \end{cases}$$

In particular, we adopt a single nugget process term from one of the regions $l = 1, \ldots, L$ in equation (4.2) by finding l that corresponds to the maximum value of the mixture components evaluated at the point of interest, $\mathbf{x}$. This specification allows the nugget process to be region specific, but it does not vary in the same way as the residual process. In subsection 2.3.3, we mentioned that the nugget term could account for noise in the simulator output or the effect of inactive inputs in the residual term, and we consider it as an unstructured term in our model. Given $\boldsymbol{\beta}$, $\boldsymbol{\lambda}(\mathbf{x}) = (\lambda_1(\mathbf{x}), \cdots, \lambda_l(\mathbf{x}))$, $\Delta = (\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_L)^T$, $\boldsymbol{\sigma}^2 = (\sigma_1^2, \ldots, \sigma_L^2)$ and $\boldsymbol{\tau}^2 = (\tau_1^2, \ldots, \tau_L^2)$, our nonstationary GP is

$$f(\mathbf{x})|\boldsymbol{\beta}, \boldsymbol{\lambda}(\mathbf{x}), \boldsymbol{\sigma}^2, \Delta, \boldsymbol{\tau}^2 \sim GP\Big(h(\mathbf{x})^T\boldsymbol{\beta}, k(\cdot, \cdot; \boldsymbol{\sigma}^2, \Delta, \boldsymbol{\tau}^2)\Big) \tag{4.3}$$

with covariance

$$k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\sigma}^2, \Delta, \boldsymbol{\tau}^2) = \sum_{l=1}^{L} \lambda_l(\mathbf{x})\lambda_l(\mathbf{x}')k_l(\mathbf{x}, \mathbf{x}'; \sigma_l^2, \boldsymbol{\delta}_l) + \mathbb{1}\big\{\mathbf{x} = \mathbf{x}'\big\} \sum_{l=1}^{L} z_l(\mathbf{x})z_l(\mathbf{x}')\tau_l^2,$$

so that the covariance kernel for our nonstationary GP is a mixture of stationary covariance kernels. This formulation allows us to specify a different type of process behaviour in L regions similar to TGP and the Voronoi Gaussian Process model; however, we avoid introducing boundary discontinuities between regions. Therefore by mixing GPs in this way we can have a non-zero covariance between the points from different regions. This type of GP aims to model a response, whose behaviour

changes continuously across the input space. We observed this type of response behaviour multiple times from working with climate modellers.

It is important to mention that our approach has similarities to the method from spatial statistics literature. Fuentes (2001) and Banerjee et al. (2004) assumed the existence of L stationary processes in different regions of a 2D spatial field and specified L centroids (centres of mass, knots), $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_L$, in that field by applying rectangle-partitioning. In this approach, the input space, $\mathcal{X}$, is enclosed in a rectangle and $\boldsymbol{t}_1$ is obtained as the centroid of $L = 1$. For $L = 2$, the rectangle is divided from right to left, and the centroids for the resulting two rectangles are defined as $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$. The process is continued in a similar way with defining more centroids, and BIC (Schwarz, 1978) is used to select the optimal L. The $f(\mathbf{x})$ is defined in the same way as in equation (4.2) and with the mixture component definition, i.e. $\lambda_l(\mathbf{x}) = \alpha(\mathbf{x}, \boldsymbol{t}_l)$, that depends on the centroids via a distance function.

Banerjee et al. (2004) specified one of the possible forms for $\alpha(\mathbf{x}, \boldsymbol{t}_l)$ as

$$\alpha(\mathbf{x}, \boldsymbol{t}_l) = \frac{\gamma(\mathbf{x}, \boldsymbol{t}_l)}{\sqrt{\sum_{l'=1}^{L} \gamma^2(\mathbf{x}, \boldsymbol{t}_{l'})}},$$

where $\gamma(\mathbf{x}, \boldsymbol{t})$ is a decreasing function of the distance between $\mathbf{x}$ and $\boldsymbol{t}$. The indicator function, $z_l(\mathbf{x})$, in the last term of equation (4.2) also depends on the distance between $\mathbf{x}$ and $\boldsymbol{t}_l$ and is defined as

$$z_l(\mathbf{x}) = \begin{cases} 1, & ||\boldsymbol{t}_l - \mathbf{x}|| = \min_k ||\boldsymbol{t}_k - \mathbf{x}||, \\ 0, & \text{otherwise}, \end{cases}$$

where $||\boldsymbol{t} - \mathbf{x}||$ is the Euclidean distance between $\mathbf{x}$ and $\boldsymbol{t}$. This proposed model has been applied to a 2D spatial field, and in an illustrative study of modelling house prices in Stockton, California, 632 data points (design points) were used to fit a nonstationary GP model and up to $L = 16$ centroids were considered for fitting (Banerjee et al., 2004). In the field of computer experiments, such large data sets are rare, and the rectangular partitioning and Euclidean distance-dependent weight functions are unlikely to work for problems with $p > 2$. Our approach is more

scalable to larger dimensions and, as we argue below, more natural for modellers to implement.

4.3.2 Mixture components definition based on diagnostics

We propose to partition the input space using diagnostics from the initial stationary GP fits. The diagnostics, both numerical summaries as well as graphical methods, based on prediction errors are commonly used to check the validity of a GP model to represent a response of interest (see section 2.4 for more details). We consider the LOO standardized errors, e_i, defined as

$$e_i = \frac{f(\mathbf{x}_i) - E[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}]}{\sqrt{Var[f(\mathbf{x}_i)|\{\mathrm{X}_{-i}, \mathrm{F}_{-i}\}]}}.$$

In spatial statistics, Cressie (1993) used diagnostics based on LOO standardized errors to evaluate the validity of a fitted variogram, covariance function, of a GP model. In particular, he proposed to check the assumptions adopted for covariance function, such as stationarity, of GP model by considering $C_{LOO} = \frac{1}{n}\sum_{i=1}^{n} e_i^2$. This criterion should be close to 1 to confirm that there is no conflict between the GP model representation and the response (Bachoc, 2013). Samper and Neuman (1989) proposed the assumption for their model that the LOO standardized errors, e_i, are standard normal $N(0, 1)$ if a specified GP model is an appropriate representation of a response. They realised that the independence of residual is a strong hypothesis and performed a number of statistical tests to verify this hypothesis and to show that in many cases the correlation between these errors is weak. In computer experiments literature, Bastos and O'Hagan (2009) suggested to use the plots of residuals against inputs to check the validity of the stationarity assumption of the covariance function. In particular, we should expect to see a horizontal band containing the errors and centred around zero.

Based on the diagnostics methods considered above, we concluded that the LOO standardized residuals, e_i, obtained from stationary GP fits could be informative in partitioning the input space and modelling the mixture components. We propose

to consider the $\boldsymbol{\lambda}(\mathbf{x}) = (\lambda_1(\mathbf{x}), \ldots, \lambda_L(\mathbf{x}))$ as a vector of probabilities indicating the dominant local behaviour around $\mathbf{x}$ as described by $\epsilon_l(\mathbf{x})$ and $\nu_l(\mathbf{x})$.

We define a latent indicator process $s(\mathbf{x})$

$$s(\mathbf{x}) \sim \text{Multinomial}(\lambda_1(\mathbf{x}), \ldots, \lambda_L(\mathbf{x})),$$

and model the LOO standardized cross-validation residuals, e_i, via

$$e_i | s(\mathbf{x}_i) = l \sim \text{Normal}(0, \zeta_l),$$

where ζ_l is the standard deviation for the distribution of standardized errors in region l. The patterns of small and large errors in different input regions indicate the failure of the stationarity assumption used for emulator construction. Then, we can fit the $\lambda_l(\mathbf{x})$ via, for example, a categorical regression by specifying a linear function, $g(\mathbf{x}) = (\mathrm{x}_1, \ldots, \mathrm{x}_p)$, i.e.

$$\lambda_l(\mathbf{x}) = \frac{\exp(g(\mathbf{x})^T \boldsymbol{\alpha}_l)}{\sum_{l'=1}^{L} \exp(g(\mathbf{x})^T \boldsymbol{\alpha}_{l'})}, \tag{4.4}$$

with parameters $A = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_L)^T$ and $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_L)$ and a suitable prior $\pi(A, \boldsymbol{\zeta})$.

The $\boldsymbol{\lambda}(\mathbf{x})$ are computed for M posterior samples (after warm-up) and fixed at the mean value over the posterior samples denoted by $\hat{\boldsymbol{\lambda}}(\mathbf{x}) = (\hat{\lambda}_1(\mathbf{x}), \cdots, \hat{\lambda}_L(\mathbf{x}))$:

$$\hat{\boldsymbol{\lambda}}(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{\lambda}(\mathbf{x}; A_m).$$

We modify the nonstationary GP model specification introduced in subsection 4.3.1 by removing the conditioning on $\boldsymbol{\lambda}(\mathbf{x})$ by fixing $\boldsymbol{\lambda}(\mathbf{x}) = \hat{\boldsymbol{\lambda}}(\mathbf{x})$. Given region-specific parameters $\Delta = (\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_L)^T, \boldsymbol{\sigma}^2 = (\sigma_1^2, \ldots, \sigma_L^2)$ and $\boldsymbol{\tau}^2 = (\tau_1^2, \ldots, \tau_L^2)$, our nonstationary GP is therefore

$$f(\mathbf{x}) | \boldsymbol{\beta}, \boldsymbol{\sigma}^2, \Delta, \boldsymbol{\tau}^2 \sim GP\Big(h(\mathbf{x})^T \boldsymbol{\beta}, k(\cdot, \cdot; \boldsymbol{\sigma}^2, \Delta, \boldsymbol{\tau}^2)\Big) \tag{4.5}$$

with covariance function:

$$k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\sigma}^2, \Delta, \boldsymbol{\tau}^2) = \sum_{l=1}^{L} \hat{\lambda}_l(\mathbf{x}) \hat{\lambda}_l(\mathbf{x}') k_l(\mathbf{x}, \mathbf{x}'; \sigma_l^2, \boldsymbol{\delta}_l) + \mathbb{1}\{\mathbf{x} = \mathbf{x}'\} \sum_{l=1}^{L} z_l(\mathbf{x}) z_l(\mathbf{x}') \tau_l^2,$$

where

$$z_l(\mathbf{x}) = \begin{cases} 1, & \hat{\lambda}_l(\mathbf{x}) = \max_k \{\hat{\lambda}_k(\mathbf{x})\}, \\ 0, & \text{otherwise.} \end{cases}$$

We fix $\boldsymbol{\lambda}(\mathbf{x})$ at $\hat{\boldsymbol{\lambda}}(\mathbf{x})$ to avoid double counting. Fixing $\boldsymbol{\lambda}(\mathbf{x})$ at $\hat{\boldsymbol{\lambda}}(\mathbf{x})$ in this way resembles the common and effective Cross-Validation (CV) approach to estimate the parameters of the statistical model, i.e. the model parameter values that provide the user with the smallest LOO error are chosen. For example, Bachoc (2013) uses the LOO-CV to estimate σ^2 and $\boldsymbol{\delta}$ for a GP model and numerically proves that the Cross-Validation (CV) estimation is more robust than the Maximum Likelihood estimation (ML) of parameters to the model misspecification, the case when the true underlying covariance function does not belong to the family of covariance functions that is considered. Our approach resembles Cross-Validation (CV) as we attempt to fix $\boldsymbol{\lambda}(\mathbf{x})$ at values that are consistent with the ensemble, $\{\mathrm{X}, \mathrm{F}\}$.

We propose to choose the number of regions, L, by fitting separate mixture models for $L = 1, \ldots, 4$ and comparing the fit of the models using a penalized measure of model fit. It is common to believe that the optimal value of L is small since we usually operate with small n, number of design points (Almond, 2014). For our mixture model, we adapted the Akaike information criterion (AIC) with mixture model parameters fixed at the posterior mean, $\bar{A} = (\bar{\boldsymbol{\alpha}}_1, \ldots, \bar{\boldsymbol{\alpha}}_L)$ and $\bar{\boldsymbol{\zeta}} = (\bar{\zeta}_1, \ldots, \bar{\zeta}_l)$. In particular, we define the modified AIC criterion, AIC_{mod}, as

$$AIC_{mod} = -D(\bar{A}, \bar{\boldsymbol{\zeta}}) + 2 \times L(p+1) + L(p+2).$$

The first term of the modified AIC criterion corresponds to the deviance, which is twice the negative log-likelihood, where our mixture model likelihood with parame-

ters fixed at the posterior mean takes the form:

$$\pi\big(\boldsymbol{e}|\boldsymbol{X},\bar{A},\bar{\boldsymbol{\zeta}}\big)=\prod_{i=1}^{n}\sum_{l=1}^{L}\frac{\exp(g(\mathbf{x}_i)^T\bar{\boldsymbol{\alpha}}_l)}{\sum_{l'=1}^{L}\exp(g(\mathbf{x}_i)^T\bar{\boldsymbol{\alpha}}_{l'})}\phi\Big(\frac{e_i}{\bar{\zeta}_l}\Big),$$

where $\phi(\cdot)$ is the unit normal density. The second term of AIC_{mod} is equal to twice the number of parameters in the mixture model. We also propose to add the penalty that corresponds to the number of region-specific parameters of full nonstationary GP model, i.e. $\boldsymbol{\sigma}^2, \Delta$ and $\boldsymbol{\tau}^2$, as the number of regions, L, will directly affect the complexity of full nonstationary GP model. We choose L with the lowest AIC_{mod} score.

As a result, we propose a two-stage approach for constructing our nonstationary GP model. The first step is to derive the number of input regions, L, and mixture components, $\hat{\boldsymbol{\lambda}}(\mathbf{x})$, by considering the LOO standardized residuals from stationary fit. The second step is to construct a nonstationary GP model with fixed mixture components and conditioned on region-specific parameters.

We can draw similarities between our proposed nonstationary GP model and CGP reviewed in section 4.2. In particular, we could consider a two-stage volatility approach in CGP as a mixture of two covariance kernels from global and local components of the model. The added effect of a local component inside the covariance kernel of the overall model is controlled by the volatility function, $v(\mathbf{x})$, which depends on standardized residuals. We believe that our proposed nonstationary GP model provides users with greater flexibility since it attempts to model the response behaviour across $L > 2$ input regions of distinct model behaviour. In numerical studies in section 4.4 as well as in section 4.6, we observe that the added effect from a local component in CGP is not strong enough to capture the high-variability in model response.

4.3.3 Priors for nonstationary GP model hyperparameters

For the mixture model we specify priors for parameters, $\zeta_l \sim \log N(-1,1)$ and $\boldsymbol{\alpha}_l \sim \text{Normal}(0,5)$, with $l = 1,\ldots,L$ and $\boldsymbol{\alpha}_l = (\alpha_{1l},\ldots,\alpha_{pl})$, probability dis-

tributions with wide support representing weak prior information (Almond, 2014; Stan Development Team, 2017), which allows us to rely on the standardized errors, $\boldsymbol{e} = (e_1, \ldots, e_n)$, in obtaining $\hat{\lambda}_l(\mathbf{x}), l = 1, \ldots, L$. The two prior distributions on parameters are shown as density plots in Figure 4.3. We also constrain the prior standard deviation parameter to follow the ordering, $\zeta_1 \leq \zeta_2 \leq \cdots \leq \zeta_L$, solving the problem of having multiple modes in the posterior distribution for mixture models, and ensuring good mixing of our Markov chains (Almond, 2014).

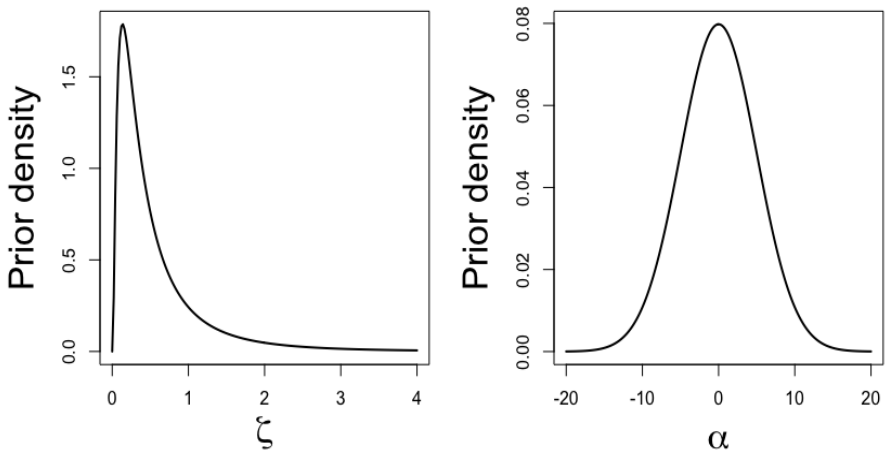

Figure 4.3: Lognormal(-1, 1) (*left*) and Normal(0, 5) (*right*) priors for parameters.

We use Stan (Stan Development Team, 2017) for our inference, introduced in Chapter 3. However, Stan does not provide sampling for discrete parameters (Stan Development Team, 2017), therefore, the posterior of the discrete group allocation indices, $s = (s(\mathbf{x}_1), \ldots, s(\mathbf{x}_n))$, cannot be sampled directly and so we integrate s out in the likelihood. Mixture components are computed a posteriori. The regression coefficient parameters are sampled from the joint posterior, integrating over s,

$$p(A|\boldsymbol{e}, \boldsymbol{\zeta}) \propto \int p(\boldsymbol{e}|s, \boldsymbol{\zeta}) p(s|A) p(A) p(\boldsymbol{\zeta}) ds.$$

Sinse s is discrete, this is equivalent to

$$p(A|\boldsymbol{e}, \boldsymbol{\zeta}) \propto \prod_{i=1}^{n} \left(\sum_{l=1}^{L} Pr(s(\mathbf{x}_i) = l|A) p(e_i|\zeta_l) \right) p(A) p(\boldsymbol{\zeta}),$$

where $A = \{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_L\}$ and $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_L)$.

As a result, we obtain posterior simulations of the parameters, $(A_i, \boldsymbol{\zeta}_i), i = 1, \ldots, M$, that are used to compute $\hat{\lambda}_l(\mathbf{x}), l = 1, \ldots, L$.

The ordering in the prior specification of the standard deviation parameter for residuals persist in nonstationary GP model. In particular, $l = 1$ region corresponds to the region with the smallest variability in residuals from stationary GP fit, ζ_1. Therefore we should expect to observe comparatively smaller values of σ_1 and longer correlation length values, $\boldsymbol{\delta}_1$, produced by our proposed nonstationary GP emulator, as we are in the "well-behaved" region of input space. The complete opposite is true for $l = L$.

The workflow of our proposed nonstationary GP model consists of the following steps:

1. Construct a GP emulator with a stationary covariance function to approximate the computer model output (metric of interest).

2. Use LOO diagnostics to validate the performance of stationary GP emulator. In particular, we are interested in examining the individual standardized residuals plots against inputs. If the emulator can adequately represent the simulator, we should expect to observe a horizontal band between -2 and 2 containing the errors. An isolated outlier, a residual with an absolute value greater than 2, could be ignored (Bastos and O'Hagan, 2009). However, if we identify any relationship between residuals and inputs such as heteroscedasticity, we could use our proposed method.

3. Consider the mixture model for LOO standardized residuals, $\boldsymbol{e} = (e_1, \ldots, e_n)$, presented in section 4.3.2 with $L = 1, \ldots, 4$. Since we generally operate with far fewer design points, considering $L > 4$ would potentially introduce issues with the convergence and mixing of Markov chains for mixture parameters (Almond, 2014). Choose the mixture model that provides the lowest AIC_{mod} score. Compute $\hat{\boldsymbol{\lambda}}(\mathbf{x}_i)$ for design set, $\mathrm{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, by using a set of posterior samples generated for mixture model parameters $(A_j, \boldsymbol{\zeta}_j), j = 1, \ldots, M$.

4. Specify the same form of linear model and priors for nonstationary GP hy-

perparameters as for the stationary GP model and use fixed $\hat{\boldsymbol{\lambda}}(\mathbf{x})$ in place of $\boldsymbol{\lambda}(\mathbf{x})$ for the design set. Construct nonstationary GP emulator.

4.4 Numerical studies

We consider the performance of stationary and our proposed nonstationary GP emulators on a set of test functions and also demonstrate the performance of Bayesian TGP (Gramacy et al., 2007) and composite GP (CGP) (Ba and Joseph, 2012). To perform a fair comparative study, we modified the prior specification used in `ExeterUQ` software presented in Chapter 3.

Prior to fitting our emulators, we perform a transformation on ensemble, $\{\mathrm{X}, \mathrm{F}\}$, i.e. we transform X on $[-1, 1]$ scale and centre response around zero by subtracting mean of response and scaling by the standard deviation of response. This step allows us to specify the same form of proper prior for regression coefficients, $\boldsymbol{\beta}_{1:(p+1)} \sim N(0, 10)$, and perform prior studies in section 4.5.

We specify a linear structure for the prior mean, $h(\mathbf{x}) = (1, x_1, \ldots, x_p)^T$. We adopt the standard choice regarding the regressors with $h(\cdot)$ that includes a constant and linear terms in each component of $\mathbf{x}$ (Kennedy and O'Hagan, 2001), contrary to approaches that attempt to use a complex mean function (see subsection 2.5). An important remark is that `ExeterUQ` software allows users to use a complex form of $h(\cdot)$ and we were concerned that adopting a complex form of the regression function, $h(\cdot)$ will provide us with an unfair advantage over `TGP` and `CGP` in our comparative study.

We have also modified the form of prior for scale parameter, σ. In particular, we define a weakly informative prior, i.e. $\sigma^2 \sim IG(2, 1)$, commonly used in UQ literature (Gramacy and Lee, 2008; Montagna and Tokdar, 2016). We specify the same forms of the prior distribution for model parameters, $\boldsymbol{\beta}$ and $\left\{\sigma_l^2, \boldsymbol{\delta}_l\right\}_{l=1:L}$, for our nonstationary GP emulator. In practice, different priors could be employed for region-specific parameters. However, as we are aiming to assess and compare the performance of our proposed nonstationary GP emulator against other GP models, we stick to the prior specification used for stationary GP. In the rest of the Chap-

ter, we refer to our stationary and nonstationary emulators with prior specification introduced above as "out of the box".

To implement TGP for comparative study, we used R package TGP with default settings; in particular, we used linear mean function specification and separable power exponential correlation function. The form of covariance function used in the package TGP is written as

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \left(\exp\Big\{ -\sum_{i=1}^{p} \frac{|x_i - x_i'|^{\phi}}{\delta_i} \Big\} + \mathbb{1}\{\mathbf{x} = \mathbf{x}'\}\tau^2 \right)$$

where τ^2 is a nugget term. We fixed the nugget parameter at 0.01 since the nugget term in TGP (Gramacy et al., 2007) has a different interpretation of our definition of a nugget term. We used CGP with default settings (Ba and Joseph, 2018). Both R packages are available from CRAN.

We assess the performance of the mentioned above GP emulators by considering the Root Mean Squared Error (RMSE) and the Interval Score for $(1-\alpha) \times 100\%$ with $\alpha = 0.05$ prediction interval (Gneiting and Raftery, 2007). Both scores were introduced and described in subsection 3.6.3.

4.4.1 The 2D "wavy" function

We propose to consider and compare our nonstationary GP model on a toy function, the 2D "wavy" function, introduced in section 4.1. We use a 24-run maximin distance Latin Hypercube (LHC) (Morris and Mitchell, 1995) to train our emulators (provided in CGP manual by Ba and Joseph (2012)). Firstly, we construct a stationary GP emulator and consider the standardized errors by fitting the mixture model with $L = 1, \ldots, 4$.

Models	Deviance	AIC_{mod}
$L = 1$	58.51	67.51
$L = 2$	48.83	66.83
$L = 3$	48.43	75.43
$L = 4$	48.87	84.87

Table 4.1: Mixture Model Comparison for $L = 1, 2, 3, 4$.

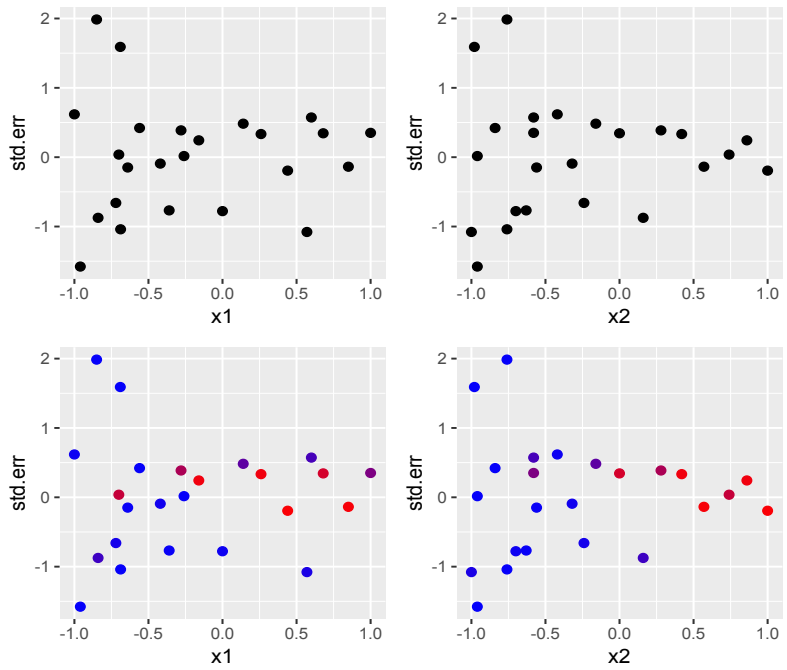

Figure 4.4: *Top row*: e_i against x_1 and x_2. *Bottom row*: coloured e_i: the deep blue colour corresponds to the higher probability of a point being allocated to region 2, while the deep red colour corresponds to the higher probability of a point being allocated to region 1.

Table 4.1 demonstrates that the mixture model with $L = 2$ has the lowest AIC_{mod} measure and based on this measure we conclude that $L = 2$ is the optimal number of the input regions for our nonstationary GP model.

Figure 4.4 demonstrates the behaviour of standardized errors against input parameters. The colour scale shows the probability of allocation of design point to one of the regions, i.e. the deep blue colour corresponds to the probability of a point being allocated to the high variability in response region close to 1. In contrast, the deep red colour corresponds to the probability of a point being allocated to the low variability in response region close to 1. Figure 4.4 confirms that $L = 2$ is a good choice for the mixture model, i.e. the variability of the errors depends on both inputs, and there are two distinct, identifiable regions of standardized error behaviour. For $x_1, x_2 < -0.5$, the standardized errors exhibit large variability, i.e. standardized errors ranging from -1.5 to 2, while for input values greater than -0.5 the variability of standardized errors significantly decreases. We use the model described in section 4.3.2 to derive mixing functions.

Figure 4.5 demonstrates the performance of all four emulators for the wavy

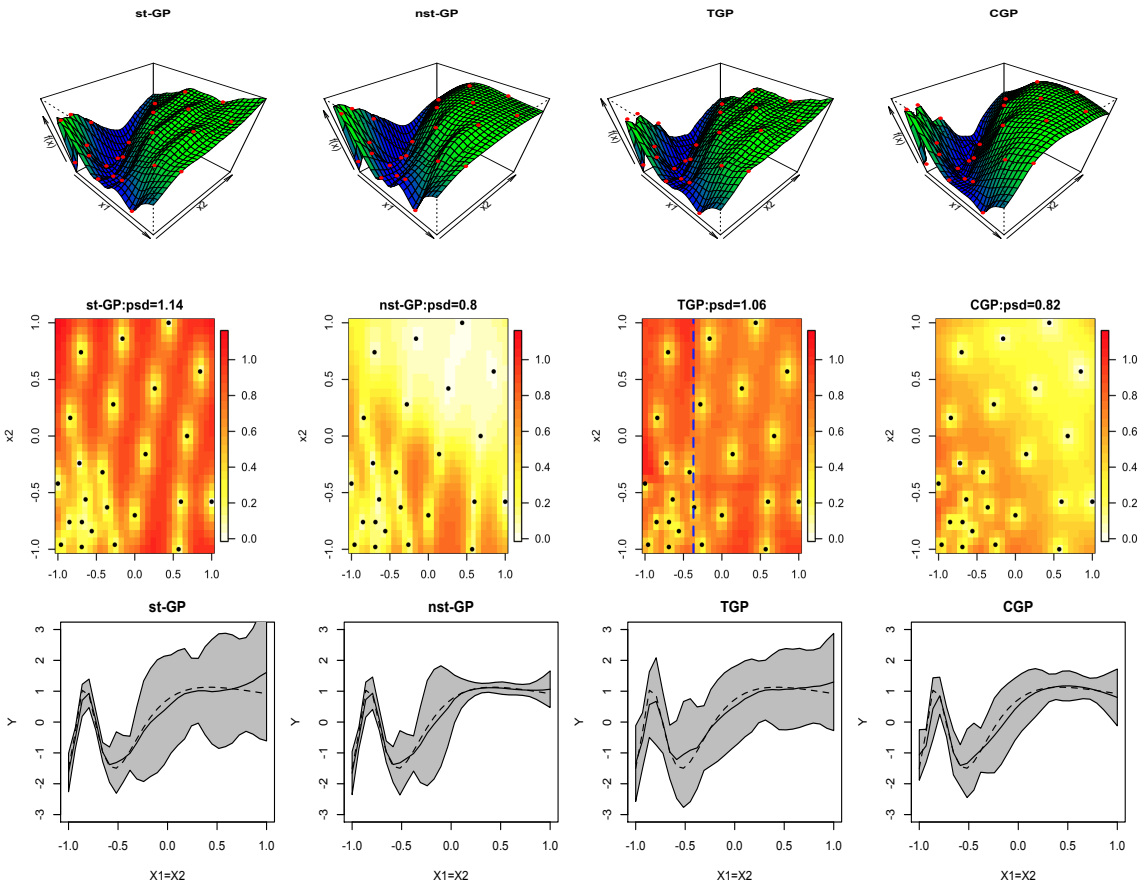

Figure 4.5: Comparison between stationary GP (*st-GP*), our nonstationary GP (*nst-GP*), TGP, and CGP. *First row*: posterior predictive mean surface and root mean squared error (rmse). *Second row*: posterior predictive standard deviation and maximum posterior predictive standard deviation (max psd). *Third row*: cross-section performance at $x_1 = x_2$. The dashed line is the true function, the solid black line is the posterior mean predictive curve, and the grey areas denote two standard deviation prediction intervals.

function. The predictive mean surface, $m^*(\mathbf{x})$, (top row) produced by our "out of the box" nonstationary emulator and by CGP resemble the image of the true function surface, while the stationary GP emulator and TGP emulator produce a number of ridges across the input space.

From the second row of Figure 4.5, we observe that the stationary GP emulator produces the lowest values of predictive standard deviation around the design points, but due to the small values of the correlation length parameters, the information is quickly "dying" away from the design points. Our nonstationary GP emulator and CGP manage to learn and explore the function behaviour for high values of x_1 and x_2 based on a few design points due to the stronger correlation structure in this region. TGP partitions the input space at $x_1 = -0.38$ (blue dashed line) and we

observe the highest predictive standard deviation around the partition.

We consider in detail the tree structure obtained for this particular example with TGP package (Gramacy et al., 2007). In the process of fitting a TGP, we automatically store the MAP tree $\hat{\mathcal{T}}$, a tree with the maximum log-posterior value. The left plot in Figure 4.6 demonstrates that the MAP tree, $\hat{\mathcal{T}}$, produces a split at $x_1 = -0.37931$. As a result, we expect to observe a greater variability in response in the input region for $x_1 < -0.37931$ according to the value of estimated scale parameter, $\hat{\sigma}^2 = 0.0884$, than in the region for $x_1 > -0.37931$ with $\hat{\sigma}^2 = 0.0449$. Separate GP models are fitted to each input partition based on a subset of original ensemble, $\{\mathrm{X}, \mathrm{F}\}$, i.e. D_1 and D_2, which contain 11 and 13 elements respectively.

Figure 4.6: A MAP tree, $\hat{\mathcal{T}}$, with one split, resulting in two regions, shown in a diagram (*left*) and pictorially (*right*). The split occurs at $x_1 = -0.37931$. The ensemble, $\{\mathrm{X}, \mathrm{F}\}$, is divided into subsets D_1 and D_2, that contain 11 and 13 elements of ensemble respectively. The numerical value at each leave corresponds to $\hat{\sigma}^2$.

The behaviour of the wavy function changes along the line where $x_1 = x_2$. Figure 4.5 demonstrates the cross-section plot for the stationary model and shows the limitations of the stationary emulator, i.e. it is overconfident in the region where the function behaviour changes rapidly. Both the TGP and CGP models are under-confident across the cross-section, while our nonstationary GP emulator produces the lowest prediction intervals among all of the methods, through is perhaps under-confident during the transition to smoother behaviour $x_1, x_2 \geq -0.5$.

Table 4.2 demonstrates that our "out of the box" nonstationary GP emulator produces the lowest Interval Score for the validation ensemble followed by CGP. However, CGP and TGP demonstrate lower RMSE than our proposed nonstationary

Models	Interval Score	RMSE
st-GP	2.76	0.298
nst-GP	1.52	0.278
TGP	2.67	0.267
CGP	1.61	0.233

Table 4.2: Interval Score and RMSE for the 2D "wavy" function

GP emulator.

In section 2.5, we briefly mentioned that "deep " approaches, in particular deep GP (DGP), could be used to model nonstationarity. We are interested in studying the performance of DGP for 2D "wavy" function.

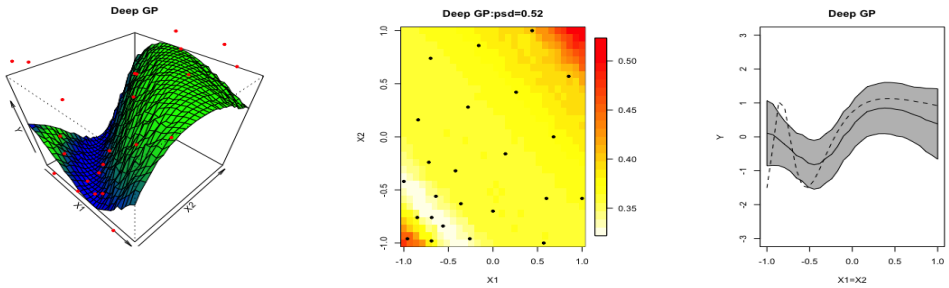

Figure 4.7: Performance of DGP for "wavy" function. *Left panel*: posterior predictive mean surface. *Central panel*: posterior predictive standard deviation and maximum posterior predictive standard deviation (max psd). *Right panel*: cross-section performance $x_1 = x_2$. The dashed line is the true function, the solid black line is the posterior mean predictive curve, and the grey area denotes two standard deviation prediction intervals.

To construct DGP and produce results presented in Figure 4.7, we utilize the readily available GPyTorch library (GPyTorch, 2019). GPyTorch library allows users to build a GP model for large data sets by employing variational techniques. The training and predictions in GPyTorch are performed by using sparse variational inference to simplify the correlation within layers, but at the same time maintain the correlation between layers (Salimbeni and Deisenroth, 2017). The parameters of the Gaussian process and the parameters of the neural network are learnt using empirical Bayes, i.e. by optimizing the variational evidence lower bound stochastically.

For our specific example, we used a simple "out of the box" two-layer deep GP. We specified linear and constant mean functions for the first (hidden) and second layers respectively, while we specified RBF kernel for both layers.

From the left panel plot in Figure 4.7, we observe that DGP underestimates the values of model response in high and low variability region. The highest values of predictive uncertainty are obtained in these two regions. The cross-section plot shows that the width of two standard deviation prediction interval does not change. We conclude that DGP attempts to capture a general model behaviour, i.e. how the model behaves on average across the input space. We have also computed the Interval Score and RMSE, which are 3.81 and 0.423 respectively. A possible explanation of the poor performance of DGP is that a training ensemble is too small for a complex DGP framework.

4.4.2 Nonstationarity in 5 dimensions

We now examine the performance of our method in higher dimensions, using the 5D test function

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 \sin(\beta_6 x_5),$$

with β_5, β_6 changing in each of 5 different intervals in x_5, as shown in Figure 4.8. Intervals in white correspond to five separate function behaviours, while the intervals in blue are a mixture of functions from two neighbouring regions to ensure continuity. Our choices of β_5, β_6 impose significant variability in smoothness with changes in x_5. We choose to vary the stationarity properties along one axis to favour TGP, which partitions the input space along one input.

We generate a 4-extended LHC of size 100 following the methodology presented by Williamson (2015). The 100 member LHC is composed of four, 25 member LHCs, each added sequentially, ensuring that the composite design is orthogonal and space-filling at each stage of extension. We will compare the performance of all methods using Leave One Latin Hypercube Out (LOLHO) diagnostics, i.e. each row in Figure 4.10 represents the predictions generated for a left out from the design LHC by refitted emulators. LOLHO diagnostics offer a sterner test for an emulator than LOO diagnostics and allow us to assess which areas of the input space do not validate well (Williamson, 2015).

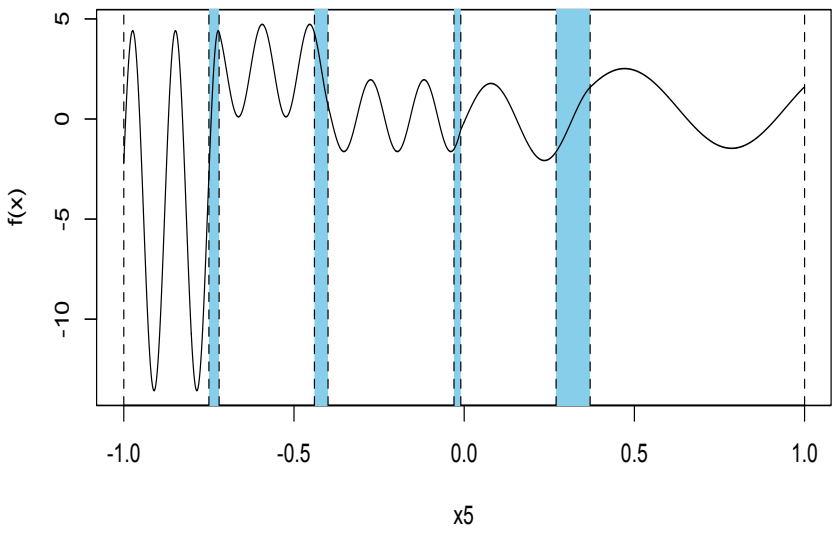

Figure 4.8: Our 5D function $f(\mathbf{x})$ plotted against x_5. All the other inputs, i.e. $x_1, \ldots, x_4$, are fixed at 0.7.

We consider four ensembles separately and proceed by constructing a stationary GP emulator and considering the standardized errors by fitting the mixture model with $L = 1, \ldots, 4$.

Models	Ens 1	Ens 2	Ens 3	Ens 4
$L = 1$	205.07	214.02	203.80	188.46
$L = 2$	**165.65**	**213.35**	**198.09**	**180.41**
$L = 3$	181.69	225.39	215.89	191.99
$L = 4$	199.48	242.85	233.46	213.47

Table 4.3: Mixture Model Comparison for $L = 1, 2, 3, 4$ for Ensembles using AIC_{mod} score. The best AIC_{mod} score for each ensemble is in bold.

Table 4.3 demonstrates that the mixture model with $L = 2$ obtains the lowest AIC_{mod} score for all ensembles, although the true function has $L = 5$ in practice. Considering $L = 5$ would lead to issues with the convergence and mixing of Markov chains for mixture model parameters (see subsection 4.3.2 for more details). Figure 4.9 demonstrates the performance of the mixture model with $L = 2$ for all four ensembles.

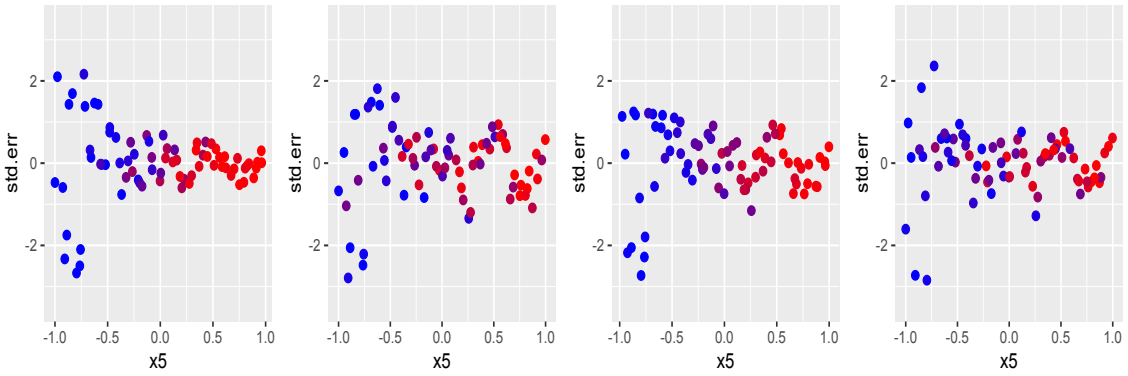

Figure 4.9: e_i against x_5 for four ensembles (sub-designs). The deep blue colour corresponds to the higher probability of a point being allocated to region 1 (high response variability region), while the deep red colour corresponds to the higher probability of a point being allocated to region 2 (low response variability region).

Models	Ens 1	Ens 2	Ens 3	Ens 4
st-GP	13.68	26.09	22.69	30.51
nst-GP	9.94	21.72	**16.57**	**23.83**
TGP	**8.99**	**9.62**	23.84	58.66
CGP	9.92	22.91	21.69	35.80

Table 4.4: Interval Score for 5D example. The best score is in bold for each ensemble.

Models	Ens 1	Ens 2	Ens 3	Ens 4
st-GP	2.315	3.725	3.557	**3.550**
nst-GP	2.085	3.531	**3.296**	3.709
TGP	**2.068**	**3.229**	4.574	6.511
CGP	2.606	3.728	3.307	4.227

Table 4.5: RMSE for 5D example. The best score is in bold for each ensemble.

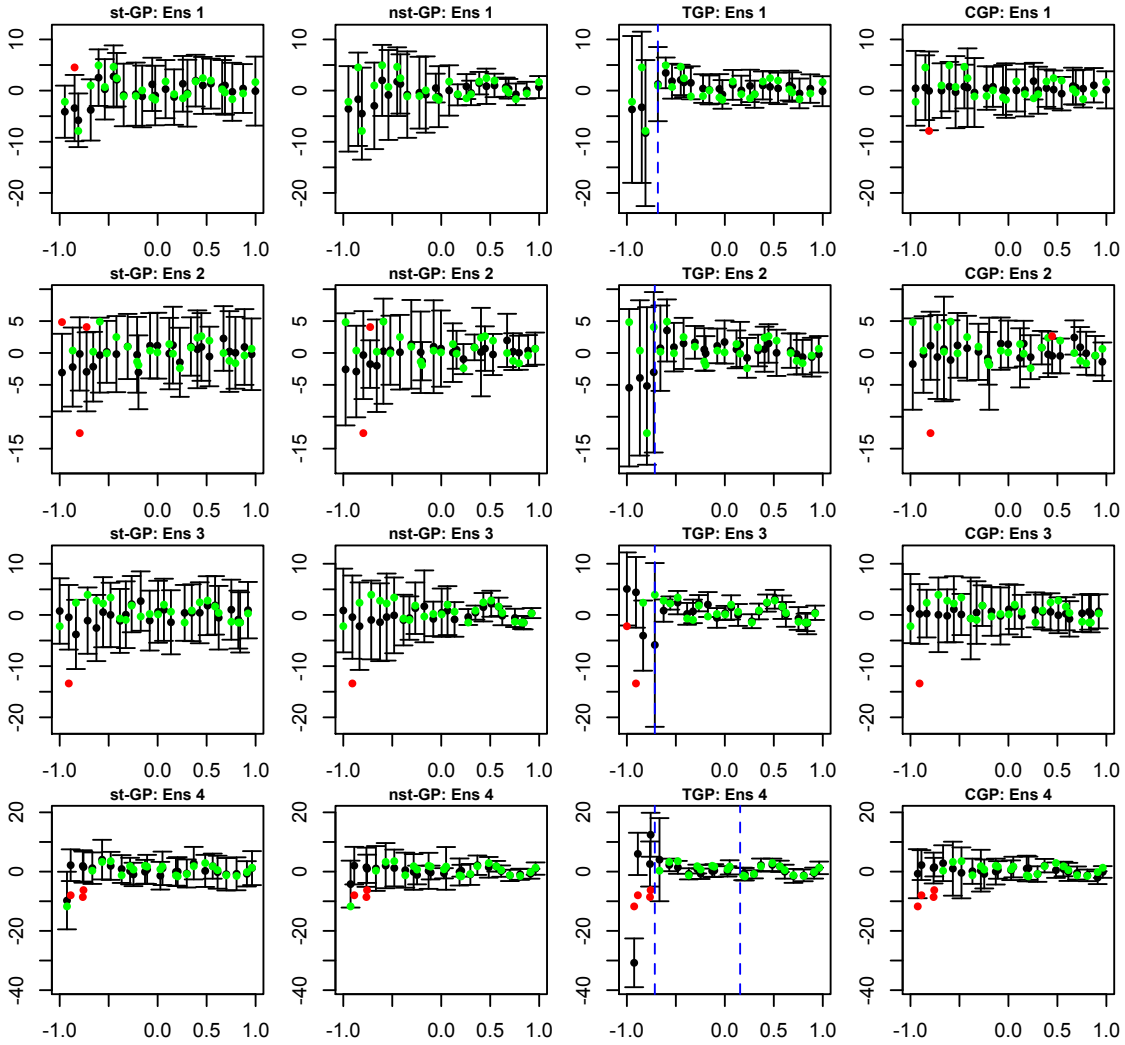

Figure 4.10: Comparison between stationary GP (*st-GP*), nonstationary GP (*nst-GP*), TGP, and CGP for 5D toy example. Blue dashed lines correspond to the partitions produced by TGP. Each row is constructed by leaving one LHC out. The posterior mean and two standard deviation prediction intervals produced by emulators are in black. The green and red points are the model values, coloured "green" if they lie within two standard deviation prediction intervals and "red" if they lie outside.

Figure 4.10 demonstrates the main issue with the stationary GP emulator, i.e. the same length of prediction intervals across the whole range of x_5 values, which indicates that it fails to recognise the changes in the function response variability. Similarly to our mixture model approach TGP partitions the input space into two regions for ensembles 1, 2, 3 and performs well in the region where the function is "well-behaved". However, due to hard partitioning and the fact that TGP fits GP models to each partition independently, we observe a significant increase in the length of prediction intervals in the high variability region. CGP does recognise the variability of the function response. However, it is overconfident for all four ensembles. Our nonstationary GP emulator performs relatively well and produces the second-lowest RMSE score (see Table 4.5) for all four ensembles and the lowest Interval Score for ensemble 3 and 4 (see Table 4.4). It performs relatively well across ensembles by recognising the change in function variability. Interestingly, all four GP approaches fails to model function response in the high variability region, $x_5 < -0.715$, for ensemble 4, which could be due to design used to fit GP models being not representative of the function behaviour. In subsection 4.5.2, we will show how informative priors help us produce much better nonstationary emulators for this function.

4.4.3 Nugget predictor example

In this subsection, we analyse the nugget predictor example considered by Gramacy and Lee (2012) to demonstrate the superior performance of adding a nugget process term into the GP model in the situations when the model assumptions are violated, or data points are sparse. In particular, we consider a simulated example that appeared in Gramacy and Lee (2012) and Ba and Joseph (2012), where test function $f(x) = \sin(10\pi x)/(2x) + (x-1)^4$ is evaluated at 20 unequally spaced locations as shown in Figure 4.11. With this example, we are interested in demonstrating how does the varying nugget process affect the inference of our proposed nonstationary GP emulator.

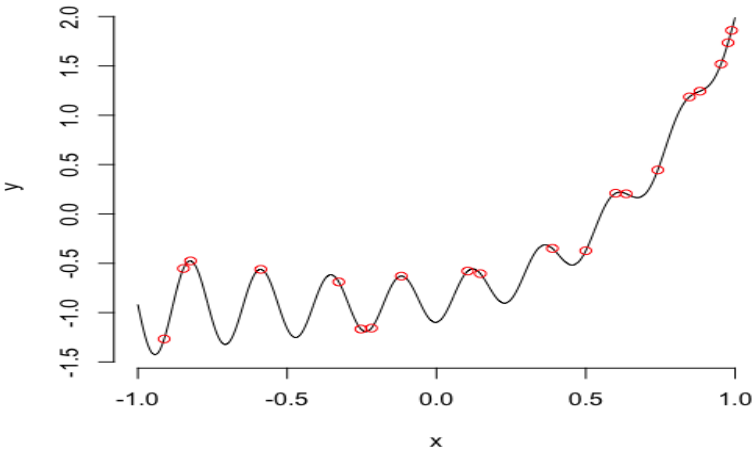

Figure 4.11: Plot of (true) function $f(x) = \sin(10\pi x)/(2x) + (x-1)^4$. The red dots represent observed data at 20 unequally spaced locations.

As in all of the previous simulation examples, we firstly construct a stationary GP emulator and consider the LOO diagnostics together with the standardized errors.

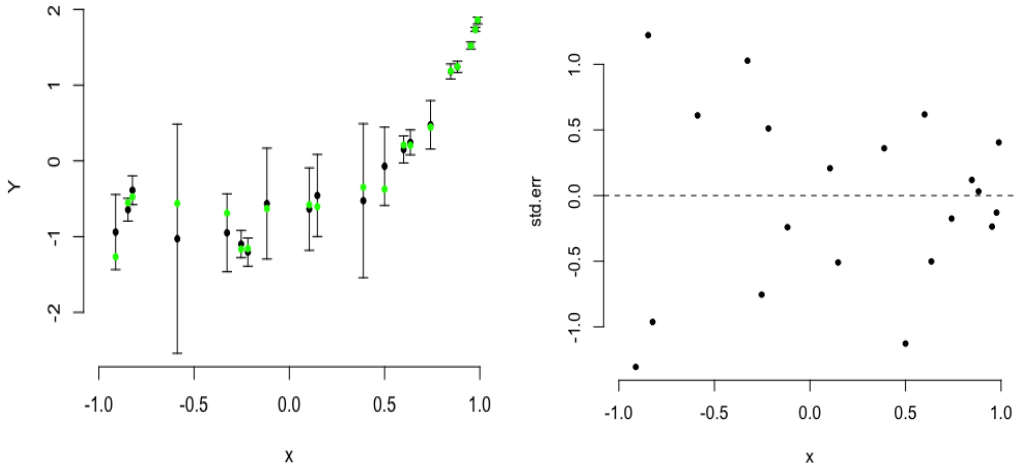

Figure 4.12: *Left panel*: Leave-One-Out diagnostic plot produced for stationary GP emulator. The posterior mean and two standard deviation prediction intervals produced by emulator are in black. The true function values are in green if they lie within two standard deviation prediction intervals, or red otherwise. *Right panel*: e_i (standardized errors) against x.

From Figure 4.12, we observe the heteroscedasticity in standardized errors against x. In particular, we could derive an increase in standardized error values in the region where design points are sparse. We proceed to fit the mixture model with $L = 1, \ldots, 4$ to identify the number of input regions with characteristic model behaviour. Table 4.6 demonstrates that the mixture model with $L = 1$ has the lowest

AIC_{mod}. However, based on the conclusions drawn from LOO diagnostics plot from Figure 4.12 and the fact that the mixture model with $L = 2$ has the second-lowest AIC_{mod} score, we have decided to fit a nonstationary GP model with $L = 2$. Figure 4.13 confirms that $L = 2$ is a good choice for a mixture model.

Models	Deviance	AIC_{mod}
$L = 1$	41.23	**48.23**
$L = 2$	37.02	51.02
$L = 3$	36.24	57.24
$L = 4$	36.69	64.69

Table 4.6: Mixture Model Comparison for $L = 1, 2, 3, 4$ for a nugget predictor example using AIC_{mod} score. The best score value is in bold.

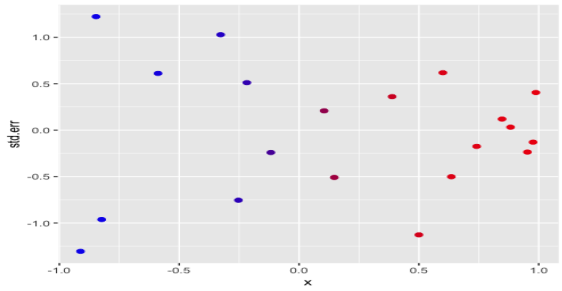

Figure 4.13: e_i against x. The deep blue color corresponds to the higher probability of a point being allocated to region 2, while the deep red color corresponds to the higher probability of a point being allocated to region 1.

In subsection 4.3.1, we defined the nugget process dependence in region l on the hyperparameter τ_l^2, and we could model the varying effect of the nugget process through the prior specification for τ_l^2 for $l = 1, \ldots, L$. We have specified a non-informative prior, $\tau_l^2 \sim \text{IG}(3, 0.1)$ for $l = 1, \cdots, L$. This distribution is negatively skewed with a long positive tail. Obtaining large values for τ_l^2 would lead the emulator to act as a smoother in the input region l, the region where we observe sparsity in the design points allocation.

From Figure 4.14 and Table 4.7, we conclude that we have managed to derive different nugget process parameter behaviour in two regions. In particular, we observe that the mean of posterior samples for τ_2^2 (Region 2) is twice as big as the mean of posterior samples for τ_1^2 (Region 1). For the $l = 1$ region, where we observe that most of the design points are spaced close to each other, we tend to obtain small

values for τ_1^2 from the MCMC sample, and in this case, the emulator will act more like an interpolator. For the $l = 2$ region, where we observe the sparse allocation of design points, we obtained relatively large values for τ_2^2 from the MCMC samples, and in this case, the emulator will act as a smoother.

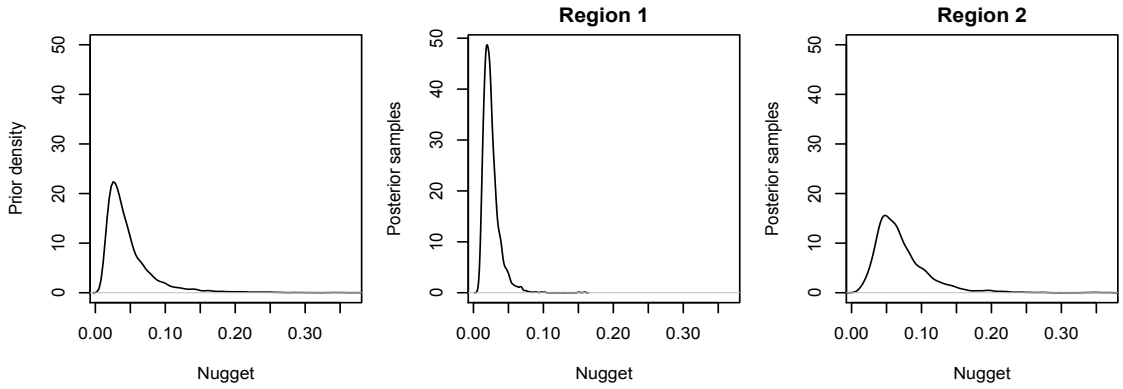

Figure 4.14: Density plot of IG(3, 0.1) (*left*), posterior samples of τ_1^2 (*center*) and posterior samples of τ_2^2 (*right*).

	Parameter	Min	1st Qu.	Median	Mean	3rd Qu.	Max.
Region 1	τ^2	0.0090	0.0176	0.0229	0.0254	0.0300	0.1453
Region 2	τ^2	0.0091	0.0445	0.0622	0.0718	0.0860	0.5420
Region 1	σ^2	0.1208	0.6182	0.8172	0.8439	1.0611	2.0163
Region 2	σ^2	0.01537	0.5162	0.7374	0.7673	0.9885	2.0570
Region 1	δ	0.2234	0.5839	0.7067	0.7479	0.8703	2.8466
Region 2	δ	0.0719	0.4357	0.5815	0.6762	0.8406	2.8200

Table 4.7: Summary statistics of posterior samples of region specific parameters of nonstationary GP model for a nugget predictor example.

Interestingly, we also observe from Table 4.7 and Figure 4.15 that region-specific parameters σ_l^2 and $\delta_l, l = 1, 2$ do not vary between regions and effectively we could conclude that $\sigma_1^2 = \sigma_2^2$ and $\delta_1 = \delta_2$. As a result, the nonstationarity in the response of the function is modelled via varying the nugget process representation. Since we have an indicator function in front of a nugget process term instead of a smoothly varying mixing function, this could explain the abrupt change in the prediction intervals produced by our nonstationary GP emulator between two regions in Figure 4.16.

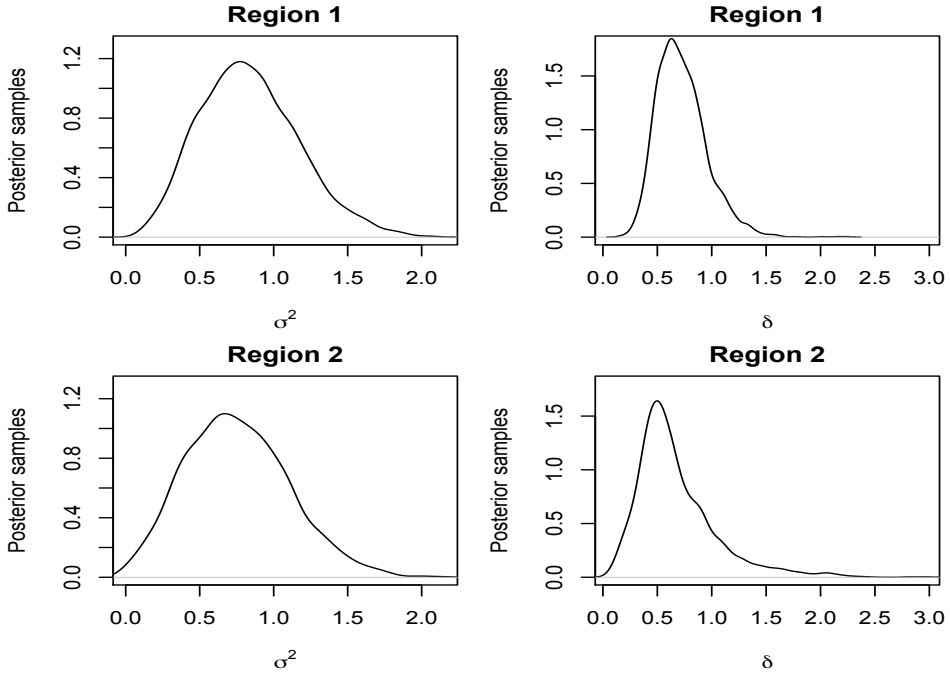

Figure 4.15: Posterior samples of region specific parameters.

Figure 4.16 demonstrates the performance of all four emulators for the nugget predictor example. For this particular example, we do not fix a nugget process term for TGP. We observe that the stationary GP emulator produces predictions outside of the range of the test function, which leads to an unfavourably high Interval Score (1.366). CGP uses a nugget term in regions away from design points and acts as an interpolator at design points. Hence CGP obtains the lowest Interval Score (0.679). Both our nonstationary GP emulator and TGP produce noninterpolating predictions, capturing the general behaviour of the test function. The size of prediction intervals produced by our nonstationary GP emulator is significantly smaller for the region where the design points are dense. This is due to the varying nature of the nugget process, and such a model feature could be useful for real data applications, where model assumptions and design density varies across the input space.

We note that an interpolator predictor is not appropriate for many real data applications. For instance, a nugget term in an emulator could be used to model the internal variability in climate models, for which slight perturbations to the input parameter settings, $\mathbf{x}$, and initial conditions may lead to a large difference in model output (Hawkins and Sutton, 2009; Williamson and Blaker, 2014). Table 4.8

demonstrates that our nonstationary GP emulator manages to achieve the second lowest Interval Score (1.039) and RMSE (0.246) after the CGP.

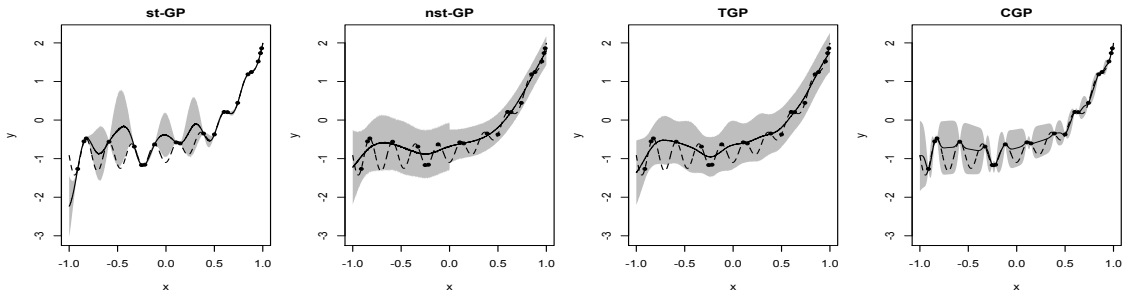

Figure 4.16: Comparison between stationary GP (*first panel*), nonstationary GP (*second panel*), TGP (*third panel*), CGP (*fourth panel*). The dashed line corresponds to the true function, the solid black line is the posterior mean predictive curve, and the grey areas denote two standard deviation prediction intervals. Black points correspond to the design points used to train our GP models. Estimates are obtained at 100 equally spaced test points.

Models	Interval Score	RMSE
st-GP	1.366	0.361
nst-GP	1.039	0.246
TGP	1.259	0.264
CGP	**0.679**	**0.197**

Table 4.8: Interval Score and RMSE for nugget predictor example. The best score value is in bold.

4.5 Generative priors

At the beginning of section 4.4, we defined the default "out of the box" choice of priors for our GP model hyperparameters, and our stationary and nonstationary emulators were constructed using the default prior specification for hyperparameters in the simulation studies in section 4.4 to offer a fair comparison. However, in general, we build statistical models to represent complex simulator responses, and a good prior for hyperparameters used to model one simulator response, might not serve as a good prior to model another simulator response (Gelman et al., 2017). Every effort should be made to specify priors for GP hyperparameters that would facilitate inference about simulator response that produce good and accurate predictions. We are aware of the notion that for problems with a large number of

data generated, the posterior distribution is going to be dominated by likelihood, and we expect to observe less effect from the prior distribution. However, this point was disproved (Gelman et al., 2017; Gabry et al., 2019) for cases when we are operating with complex statistical models with many parameters. For instance, Gabry et al. (2019) considered the statistical model for predicting $PM_{2.5}$, particulate matter measuring less than 2.5 microns in diameter, from the estimates of $PM_{2.5}$ produced by a high-resolution satellite data, defined as

$$y_{ij} \sim \mathrm{N}(\beta_0 + \beta_{0j} + (\beta_1 + \beta_{1j})x_{ij}, \sigma^2),$$
$$\beta_{0j} \sim \mathrm{N}(0, \tau_0^2), \beta_{1j} \sim \mathrm{N}(0, \tau_1^2),$$

where y_{ij} is the logarithm of the observed $PM_{2.5}$, x_{ij} is the logarithm of the estimate from the satellite model, i corresponds to observations in each pre-specified region, j ranges over the pre-specified regions. The prior distributions for model hyperparameters, $\sigma, \tau_0, \tau_1, \beta_0$ and β_1, are required. Two prior choices for parameters were considered. Firstly, vague priors have been used, i.e. $\beta_0 \sim \mathrm{N}(0, 100)$, $\beta_1 \sim \mathrm{N}(0, 100)$ and $\tau^2 \sim \mathrm{IG}(1, 100)$. The prior predictive distribution using these priors produced predictions well outside the range of observed $PM_{2.5}$ data. As an alternative, a tighter prior specification was introduced, i.e. $\beta_0 \sim \mathrm{N}(0, 1)$, $\beta_1 \sim (1, 1)$ and $\tau^2 \sim \mathrm{N}_+(0, 1)$, where N_+ is the half-normal distribution. In this case, the predictive prior distribution produced a range of values that covers the observed values, completely avoiding implausible values generated by the previous model.

We are interested in investigating the properties of prior predictive distributions (prior marginal distributions for the data) such as mean and variance for two competing prior specifications applied to a specific problem. In particular, we would like to know if our prior could generate the model response that we expect to see (Gelman et al., 2017). Considering generative priors is an elegant way to investigate how parameters jointly affect inference about a model response (Gelman et al., 2017) instead of considering the parameter by parameter effect on a model.

To derive a generative prior distribution, we have to specify proper priors for our parameters, priors whose probability distribution integrates to 1 (Gabry et al.,

2019). In our case, we specify $\pi(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}) = \pi(\boldsymbol{\beta})\pi(\sigma^2)\pi(\boldsymbol{\delta})$, as we do not assume any dependence between hyperparameters in our prior specification. We also have to specify the data-generating distribution, or in our case sampling distribution for $f(\mathbf{x})$, which was defined in subsection 2.3.1 as

$$f(\mathbf{x})|\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta} \sim GP\Big(h(\mathbf{x})^T\boldsymbol{\beta}, k(\cdot, \cdot; \sigma^2, \boldsymbol{\delta})\Big).$$

These two components are used to derive the prior predictive distribution for $f(\mathbf{x})$, which effectively attempts to predict $f(\mathbf{x})$ at $\mathbf{x}$ before any data, ensemble $\{\mathrm{X}, \mathrm{F}\}$, is obtained. The prior predictive distribution is defined as

$$\begin{aligned} \pi(f(\mathbf{x})) &= \int \pi(f(\mathbf{x})|\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta})\pi(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}) d\boldsymbol{\beta} d\sigma^2 d\boldsymbol{\delta} \\ &= \int \pi(f(\mathbf{x}), \boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}) d\boldsymbol{\beta} d\sigma^2 d\boldsymbol{\delta}. \end{aligned} \tag{4.6}$$

Gabry et al. (2019) proposed to use generative priors to investigate prior choices for a specific problem; in particular, there is an attempt to include diagnostics of priors in the Bayesian workflow. The prior predictive distribution for $\mathrm{F} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n))$ is obtained and samples from $\pi(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n))$ are compared to the observed values of F using scatter plots. The prior predictive distribution with mass on completely implausible values for F is considered to indicate that the prior specification for parameters leads to a statistical model that is inconsistent with the process that we are trying to address (Gelman et al., 2017; Gabry et al., 2019). For instance, climate modellers typically know response ranges for models as they correspond to the real-world quantities and models tend to prohibit certain extreme values.

In practice the samples from prior predictive distribution for F is obtained by firstly simulating hyperparameters values, $(\boldsymbol{\beta}_i, \sigma_i^2, \boldsymbol{\delta}_i), i = 1, \ldots, M$, from prior distribution, $\pi(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}) = \pi(\boldsymbol{\beta})\pi(\sigma^2)\pi(\boldsymbol{\delta})$, in a naive way, i.e. we randomly simulate values of the hyperparameters from the corresponding distributions with acceptance ratio fixed at 1. We proceed to using these generated values to obtain samples, $\mathrm{F}_i, i = 1, \ldots, M$, from $\pi(\mathrm{F}|\boldsymbol{\beta}_i, \sigma_i^2, \boldsymbol{\delta}_i)$.

4.5.1 Prior specification for 5D toy model

Let us return to the toy model from subsection 4.4.2. Prior to constructing our emulators, we decided to produce a scatter plot of the function response against the inputs. In Figure 4.17, we observe that the variability in the function response, $f(\mathbf{x})$, is mainly driven by x_5. The input x_5 could be considered as an "active input", and the global trend together with the residual term are modelled in terms of the "active inputs" with the nugget term to account for the remaining variation (Craig et al. 2001, Goldstein and Rougier 2009, Cumming and Goldstein 2009). However, operating within a Bayesian framework and with flexible priors, we could specify a stronger prior information for δ_5 (Higdon et al., 2008; Williamson and Blaker, 2014) for our stationary and nonstationary GP emulators. We achieve this by keeping the same Gamma$(4, 4)$ prior for δ_5, but specifying a smoother prior in the other four dimensions via $\delta_1, \ldots, \delta_4 \sim$ Gamma$(42, 9)$ (this distribution was chosen by using the MATCH elicitation tool (Morris et al., 2014) to capture a reasonable distribution giving more weight to longer correlation lengths). In subsection **??**, we discussed that a large correlation length value would lead to a flat posterior with respect to a particular input corresponding to the correlation length parameter under consideration, i.e. limited effect from this input on the final estimated GP model.

Prior Specification 1	**Prior Specification 2**
$\beta_i \sim \mathrm{N}(0, 10), i = 1 \ldots, 6$	$\beta_i \sim \mathrm{N}(0, 10), i = 1, \ldots, 6$
$\sigma^2 \sim \mathrm{IG}(2, 1)$	$\sigma^2 \sim \mathrm{IG}(2, 1)$
$\delta_i \sim \mathrm{Gamma}(4, 4), i = 1, \ldots, 5$	$\delta_i \sim \mathrm{Gamma}(42, 9), i = 1, \ldots, 4$, $\delta_5 \sim \mathrm{Gamma}(4, 4)$.

Table 4.9: The details of two prior specifications for 5D toy model.

Before constructing our stationary GP emulator, we would like to test our new prior specification against the old one by considering scatter plots of samples from the prior predictive distribution for F against F (Gabry et al., 2019).

The prior distribution on parameters are shown as density plots in Figure 4.18.

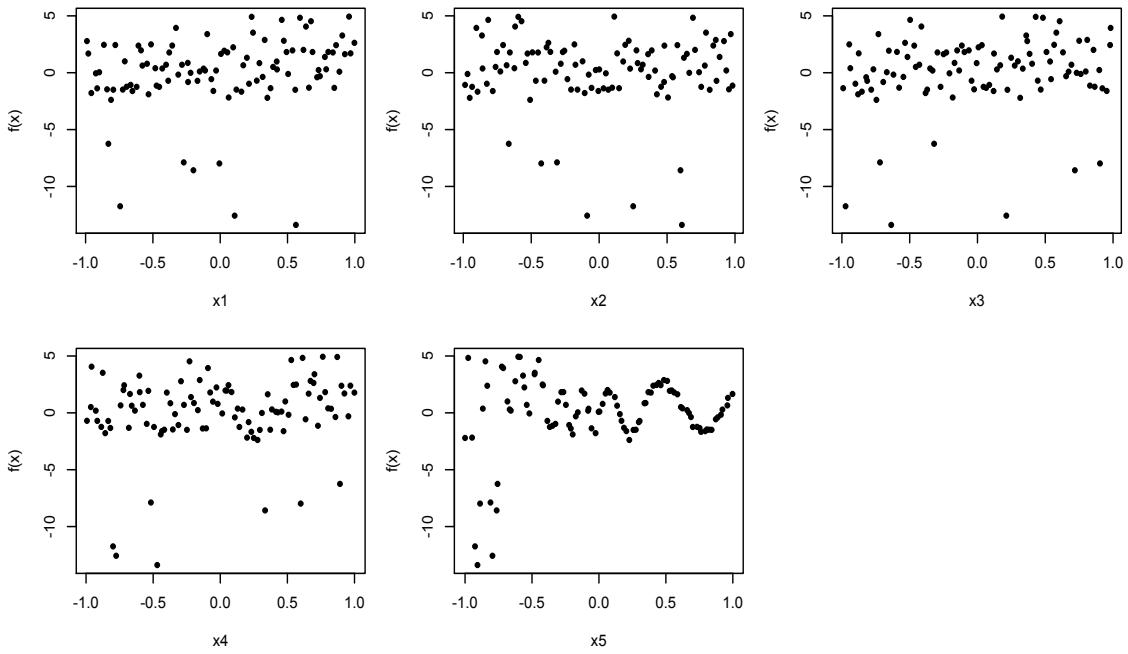

Figure 4.17: The response of 5D function $f(\mathbf{x})$ against all five inputs.

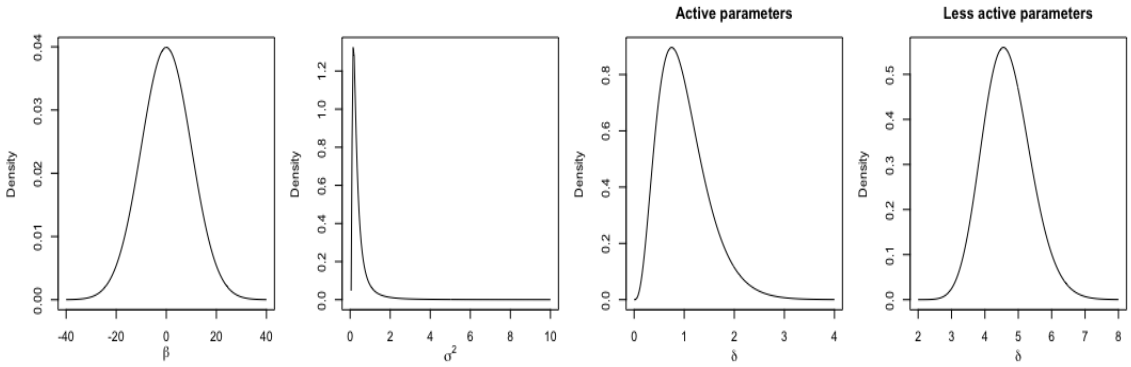

Figure 4.18: Density plots of N(0, 10) (*first panel*), IG(2, 1) (*second panel*), Gamma(4, 4) (*third panel*) and Gamma(42, 9) (*fourth panel*) priors specified for GP hyperparameters.

We produce samples from the prior predictive distribution, $\pi(\mathrm{F})$, described in subsection 4.5 for two prior specifications and visualize them against F in Figure 4.19. From Figure 4.19, we observe that the second prior choice leads to a data generating distribution that could represent the full range of the function response. Figure 4.20 demonstrates that the prior predictive distribution corresponding to the second prior choice is a weakly informative joint prior data generating process since we still observe some mass around extreme but plausible values of the function

response (Gabry et al., 2019). On the contrary, the prior predictive distribution with the first prior choice generates a range of values that are much wider than the function response that we expect to observe. From Figure 4.20, we observe that there is some mass generated by the prior predictive distribution around extreme and implausible values of the function response. We proceed with repeating our analysis in subsection 4.4.2 with the second prior choice as our prior specification for our stationary and nonstationary GP emulators.

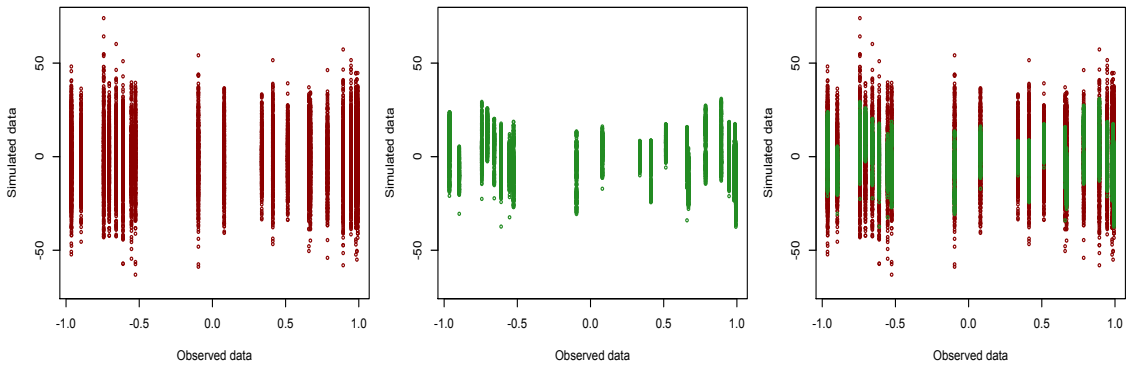

Figure 4.19: Visualizing the prior predictive distribution. *Left panel* and *central panel* show realizations from the prior predictive distribution using prior specification 1 and 2 respectively, defined in Table 4.9. The simulated data is plotted on y-axis and observed data on the x-axis. *Right panel* demonstrates the difference in the simulations by showing the red points from *left panel* and the green points from *central panel*.

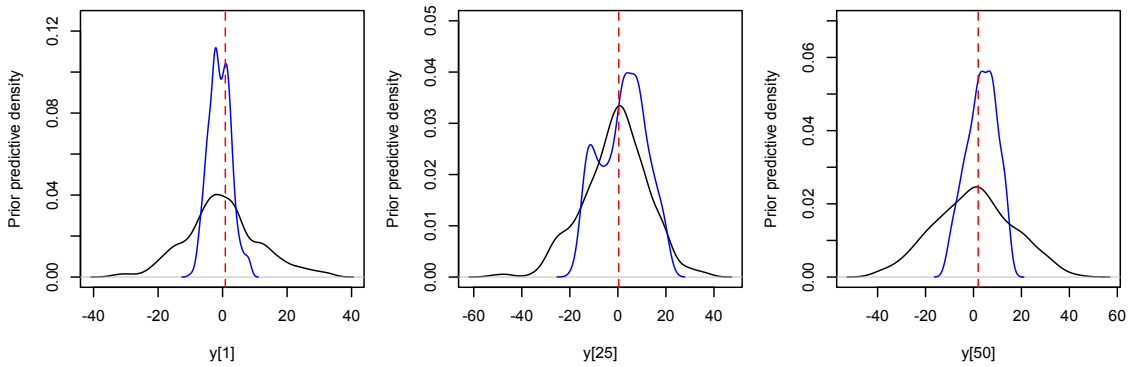

Figure 4.20: Probability densities produced by prior predictive distributions for a selection of function outputs for the first prior choice (black) and the second prior choice (blue). The true function output is given by the dashed red line

4.5.2 Results for 5D toy model

As in subsection 4.4.2, to construct our nonstationary GP emulator, we consider the mixture model for standardized errors for each LOLHO derived from a stationary fit with $L = 1, \ldots, 4$. We observe from Table 4.10 that the mixture model with $L = 2$ offers the lowest AIC_{mod} score for all four ensembles, even though the true function has $L = 5$ in practice. Figure 4.21 demonstrates the performance of the mixture model with $L = 2$ for all four ensembles.

Models	Ens 1	Ens 2	Ens 3	Ens 4
$L = 1$	173.18	175.11	157.97	178.57
$L = 2$	**105.95**	**145.79**	**126.20**	**119.21**
$L = 3$	120.48	162.29	142.87	125.96
$L = 4$	137.31	181.11	158.79	144.28

Table 4.10: Mixture Model Comparison for $L = 1, 2, 3, 4$ for Ensembles using AIC_{mod} score. The best score is in bold for all ensembles.

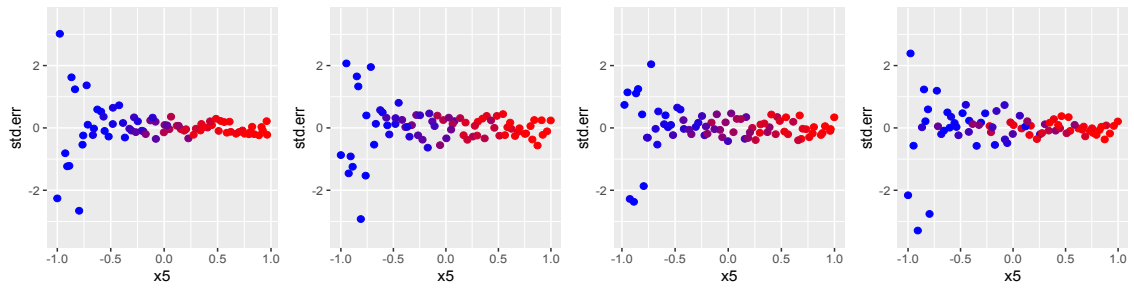

Figure 4.21: e_i against the x_5 for four sub-designs (ensembles). The deep blue colour corresponds to the higher probability of a point being allocated to region 1 (high response variability region), while the deep red colour corresponds to the higher probability of a point being allocated to region 2 (low response variability region).

Figure 4.22 shows the LOLHO diagnostic plots for each of our four emulators (columns) for each of the sub-designs considered (rows). We do observe a significant improvement in the performance of stationary and our proposed nonstationary GP emulators for all four ensembles. Low values of Interval Score and RMSE in Table 4.11 and Table 4.12 also confirm this conclusion. However, we still do observe the failure of the stationary GP emulator to recognise the changes in the function response variability by producing the same length of prediction intervals across the whole range of x_5. Our nonstationary GP emulator with $L = 2$ performs well

and produces smaller prediction intervals in the region where the function is "well-behaved", i.e. for x_5 close to 1. Interestingly, in subsection 4.4.2 we observed that all four GP approaches fail to model function response in the high variability region, $x_5 < -0.715$, for ensemble 4, which led to the conclusion that there might be an issue with the design, i.e. the design used to fit GP models being unrepresentative of the function response. However, we observe from Figure 4.22 that special treatment of "active input" x_5 resolved the issue.

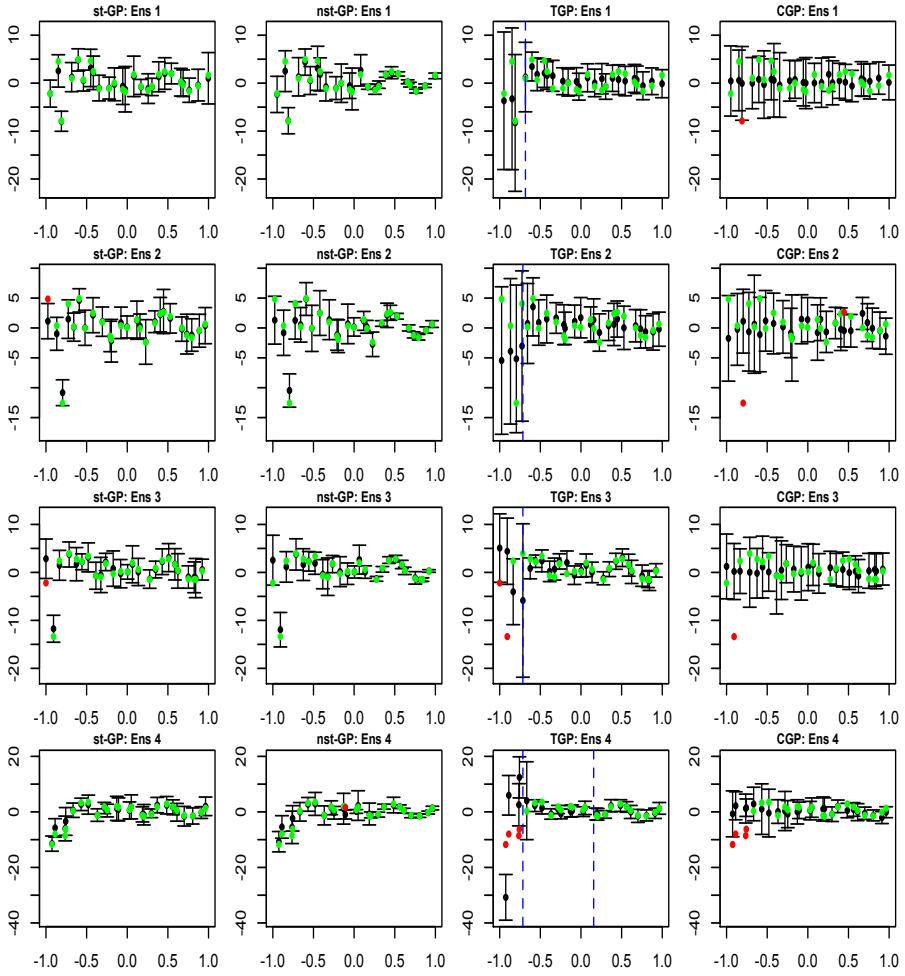

Figure 4.22: Leave One Latin Hypercube Out (LOLHO) plots for stationary GP (*st-GP*), our non-stationary GP (*nst-GP*), TGP and CGP for 5D toy example. Blue dashed lines correspond to the partitions produced by TGP. Each row is constructed by leaving one LHC out. The posterior mean and two standard deviation prediction intervals produced by emulators are in black. The true function values are in green if they lie within two standard deviation prediction intervals, or red otherwise.

Models	Ens 1	Ens 2	Ens 3	Ens 4
st-GP	6.10	6.63	7.25	6.22
nst-GP	**4.40**	**4.16**	**3.92**	**6.64**
TGP	8.99	9.62	23.84	58.66
CGP	9.92	22.91	21.69	35.80

Table 4.11: Interval Score for 5D example. The best score is in bold for all ensembles.

Models	Ens 1	Ens 2	Ens 3	Ens 4
st-GP	**0.580**	**1.028**	1.152	**0.993**
nst-GP	0.619	1.042	**1.117**	1.312
TGP	2.068	3.229	4.574	6.511
CGP	2.606	3.728	3.307	4.227

Table 4.12: RMSE for 5D example.The best score is in bold for all ensembles.

4.6 Experiments with ARPEGE-Climat model

In Chapter 3 we discussed that emulators for the SCM's would be required whenever new parameterizations are developed and tested as part of HIGH-TUNE project. In our work with the HIGH-TUNE team, we have found using routine stationary GP's insufficient to model the nonstationary response of metric of interest, and so we present the performance of our nonstationary method within this application. We will consider the average potential temperature generated by the SCM by varying nine input parameters of interest, all associated with the parameterization of convection. In the present study, we use ARPEGE-Climate, developed at the Centre National de Recherches Météorologiques (CNRM) and is the atmospheric component of the CNRM climate model. The simulations with the version 6.3 of ARPEGE-Climat are used. This is an updated version compared to the one described in Voldoire et al. (2013) (see also Abdel-Lathif et al. (2018) for further details).

Firstly we identified the physically plausible ranges of the input model parameters with the HIGH-TUNE team and standardized these to the range [-1, 1], which is the usual practice in constructing emulators for computer experiments. We generated a 160 member LHC composed of four, 40 member LHCs (Williamson, 2015), similar to the design for the 5D example in subsection 4.4.2. This type of design will also allow us to validate the performance of our emulators using LOLHO diagnostics

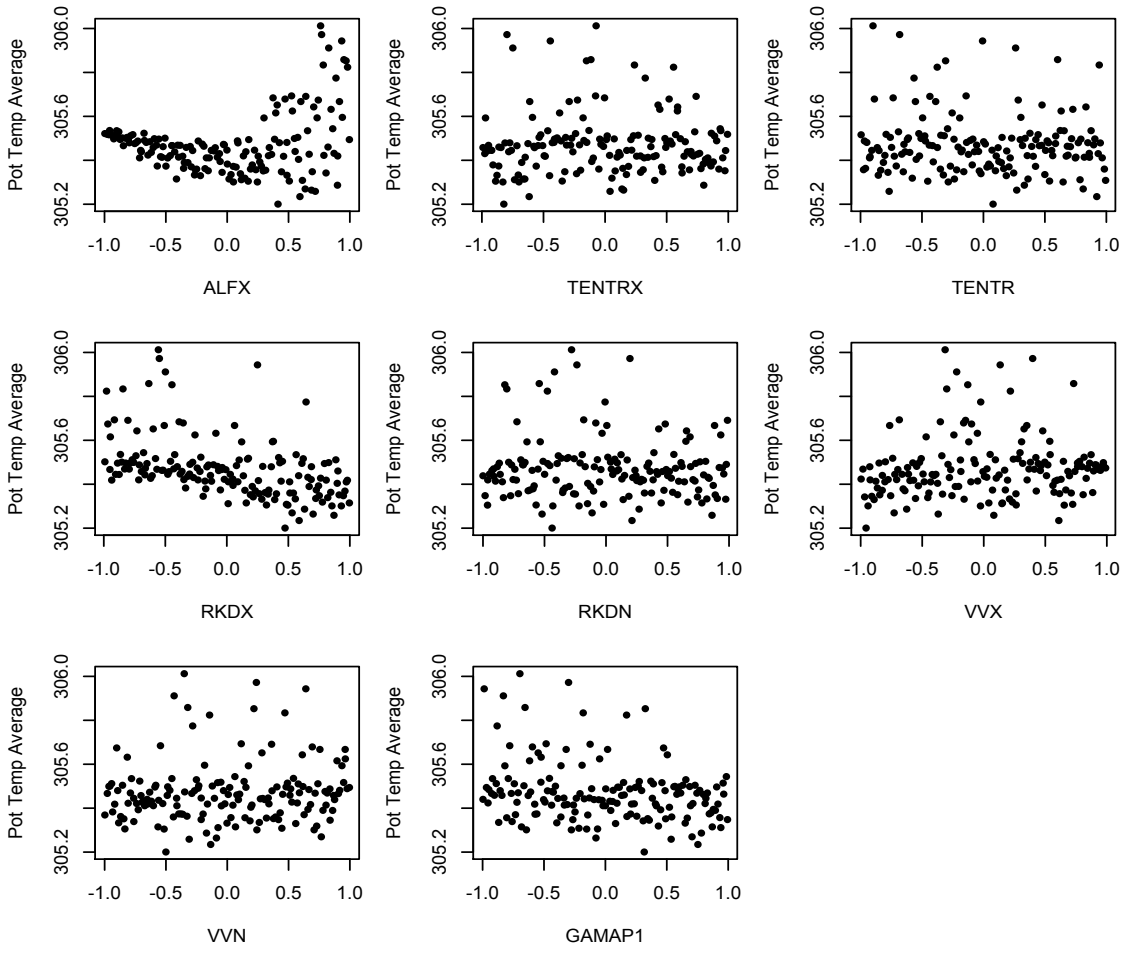

Figure 4.23: Average potential temperature against nine standardized inputs to SCM (Single Column Model).

(Williamson, 2015).

Figure 4.23 plots the average potential temperature against each input. We observe that the average potential temperature varies most with the input ALFX, i.e. the variability in the response of the average potential temperature increases as ALFX increases. From Table 4.13, we observe that the lowest AIC_{mod} score is obtained by the mixture model for standardized errors with $L = 2$. We compare the performance of our nonstationary GP emulator to the stationary GP emulator, TGP and CGP.

From Figure 4.25 we observe that the stationary GP emulator fails to recognise the variability of the model response in relation to ALFX, i.e. the length of two standard deviation prediction intervals is the same across the whole range of ALFX. TGP demonstrates satisfactory performance for all four validation en-

Models	Ens 1	Ens 2	Ens 3	Ens 4
$L = 1$	318.60	328.76	323.71	331.95
$L = 2$	**300.93**	**320.78**	**291.50**	**309.20**
$L = 3$	320.98	347.31	325.40	336.96
$L = 4$	350.55	377.52	351.21	362.07

Table 4.13: Mixture Model Comparison for $L = 1, 2, 3, 4$ for ensembles using AIC_{mod} score. The best score is in bold for all ensembles.

sembles; however, due to the hard partitioning mentioned in section 2.5, the two standard deviation prediction intervals increase significantly. CGP performs well in the input region where the model is well-behaved, however, is over-confident in the region where the model response varies the most, especially for sub-designs 2 and 4. Our nonstationary GP emulator demonstrates a gradual increase in the length of prediction intervals with increasing ALFX. From Table 4.14 we observe that our nonstationary GP model obtains the lowest Interval Score for ensembles 2 and 4 indicating good coverage, while Table 4.15 demonstrates that our nonstationary GP emulator achieves the lowest RMSE for ensembles 1, 3 and 4.

Models	Ens 1	Ens 2	Ens 3	Ens 4
st-GP	2.52	3.23	2.79	3.13
nst-GP	1.99	**2.09**	2.57	**1.97**
TGP	2.03	2.91	**2.11**	2.30
CGP	**1.91**	2.38	2.32	3.73

Table 4.14: Interval Score for ARPEGE-Climat model. The best score is in bold for all ensembles.

Models	Ens 1	Ens 2	Ens 3	Ens 4
st-GP	1.30	1.70	1.30	0.56
nst-GP	**1.28**	1.66	**1.26**	**0.54**
TGP	1.36	1.76	1.30	0.65
CGP	1.38	**1.65**	**1.26**	0.60

Table 4.15: RMSE for ARPEGE-Climat model. The best score is in bold for all ensembles.

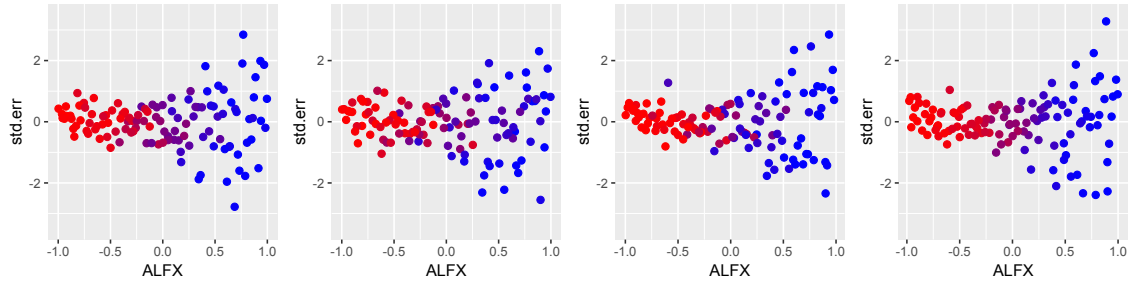

Figure 4.24: Coloured $\boldsymbol{e}_i$ against ALFX for four sub-designs. The deep blue colour corresponds to the higher probability of a point being allocated to region 1, while the deep red colour corresponds to the higher probability of a point being allocated to region 2.

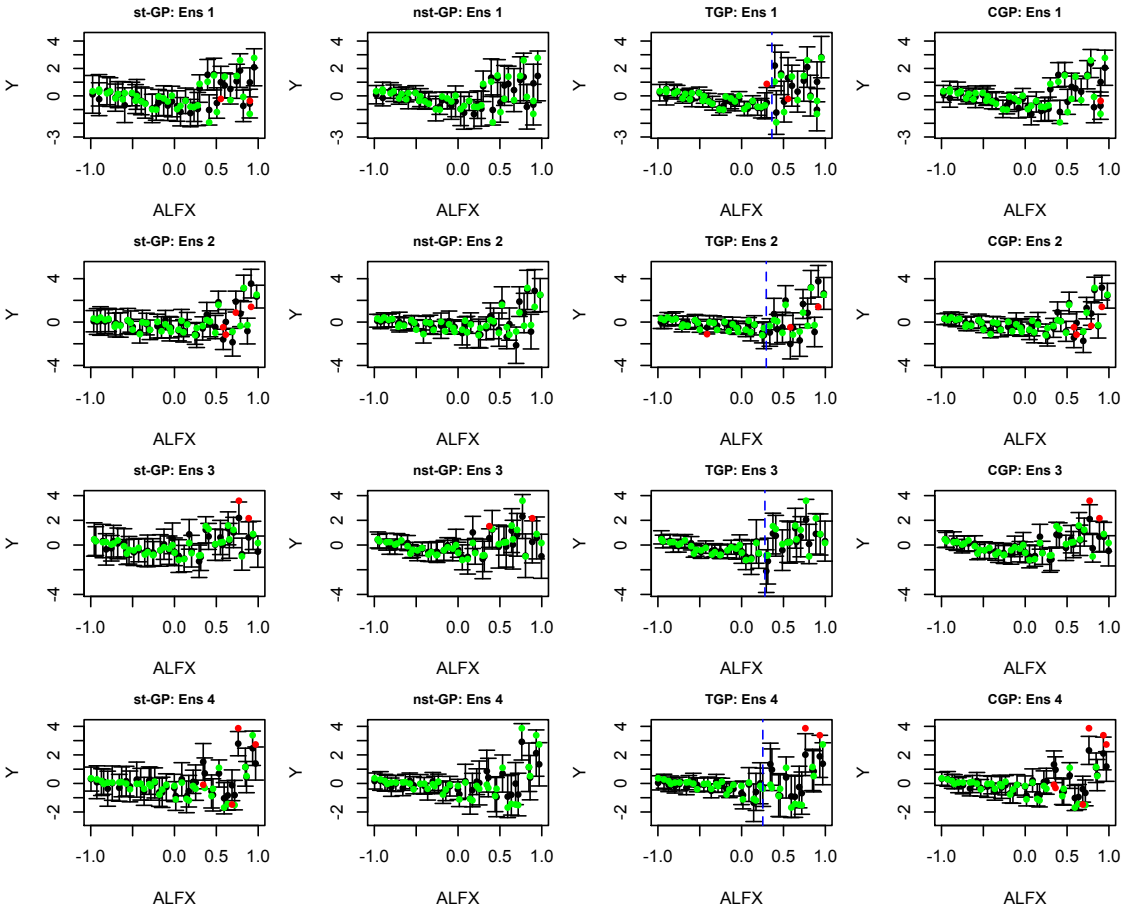

Figure 4.25: Comparison between stationary GP (*st-GP*), nonstationary GP (*nst-GP*), TGP, and CGP on modelling average potential temperature on four validation designs. Blue dashed lines correspond to partitions produced by TGP. Each row is constructed by leaving one LHC out. The posterior mean and two standard deviation prediction intervals produced by emulators are in black. The green and red points are the model values, coloured green if they lie within two standard deviation prediction intervals and red if they lie outside.

4.7 History Matching with nonstationary GP emulator

Previously, we have introduced and compared our nonstationary GP model against other approaches in modelling nonstationary simulator response. In this section, we are interested in investigating the effect of using nonstationary GP emulators on history matching results, described in subsection 2.6.1, for nonstationary simulator output. To illustrate the potential impact of fitting a nonstationary GP emulator instead of a stationary GP emulator on history matching, we refer back to the 2D "wavy" function considered in subsection 4.4.1. We compare the history matching performance of two GP models by considering the diagnostics described by Salter and Williamson (2016).

Figure 4.26 demonstrates the prediction from each of these emulators, with the associated two standard deviation prediction intervals, along a line in 2D space defined between design points $\mathbf{x}_1$ and $\mathbf{x}_2$. We have chosen $\mathbf{x}_1 = (-0.56, -0.84)$, the point allocated in the input space where the toy function behaviour is "hectic", and $\mathbf{x}_2 = (0.68, 0)$, the point allocated in the input space where the toy function is "well-behaved". The solid black line represents an assumed observed or "true" value for the function, and the dotted lines around this value depicts the observation error. We are interested in understanding how the results from history matching depend on the type of emulator that we are using as well as the location of the observed or "true" value in the input space. Therefore, apart from comparing our stationary and nonstationary GP emulators, we consider the following two cases: in the first case we specify $z = -0.6$ in the region where the toy function is highly nonstationary, while in the second case we assign $z = 1.7$ in the region where the toy function is relatively stable and "well-behaved".

Range	z	$Var[e]$	$Var[\eta]$	NROY size	a
$[-1, 1]$	-0.6	0.05^2	0	0.078%	3
$[-1, 1]$	1.7	0.05^2	0	0%	3

Table 4.16: Information about the toy function for history matching. Range denotes the spread of possible outputs for the function, and NROY size denotes the theoretical size of NROY space, given the error structure.

In Table 4.16, we denote "NROY size" as the percentage of $\mathcal{X}$, the volume of the input space, that cannot be ruled out given this setting for the observation and these variances. In particular, we assume that the function is known perfectly, and the emulator is not required (Salter and Williamson, 2016). We estimate this volume by generating a 10,000 member Latin Hypercube sample from $\mathcal{X}$ and calculating the percentage of the points that are not ruled out, which is effectively "true" NROY space.

The left panel plots in Figure 4.26 show the predictions from stationary and nonstationary GP emulators, along with two standard deviation prediction intervals, on a line through 2D space between two design points. We observe that the stationary GP emulator is underconfident in the region for $\lambda > 0.7$, producing larger uncertainty bands. Furthermore, we specify -0.6 and 1.7 as our observations (with dashed black lines representing the observation uncertainty and discrepancy). The right panel plots in Figure 4.26 demonstrate how implausibility changes when we use $a = 3$. We observe that the nonstationary GP emulator managed to rule out more space than the stationary GP emulator, shown in grey shaded region. In particular, for $\lambda > 0.7$ the implausibility function with nonstationary GP emulator produces larger values than the implausibility function with stationary GP emulator, which is due to stationary emulator being underconfident in this particular input region (more detailed explanation in subsection 4.4.1).

For both cases, we decided to perform a single wave of history matching and compare the results. For technical details regarding history matching, refer to subsection 2.6.1. For the first case, using the stationary GP emulator, we reduced the input space down to 83.8% of the original space. In contrast, with nonstationary GP emulator, we have managed to reduce the input space down to 46.6% of the original space, which is twice as much reduction. For the second case, employing the stationary GP emulator, we reduced the input space down to 64.5% of the original space, whereas with the nonstationary GP emulator, we have managed to reduce the input space down to 20.5%. We conclude that applying nonstationary GP emulator could allow us to perform fewer waves of iterative refocussing (history matching)

and produce simulator runs required for each step of iterative refocussing, leading to significant computational savings.

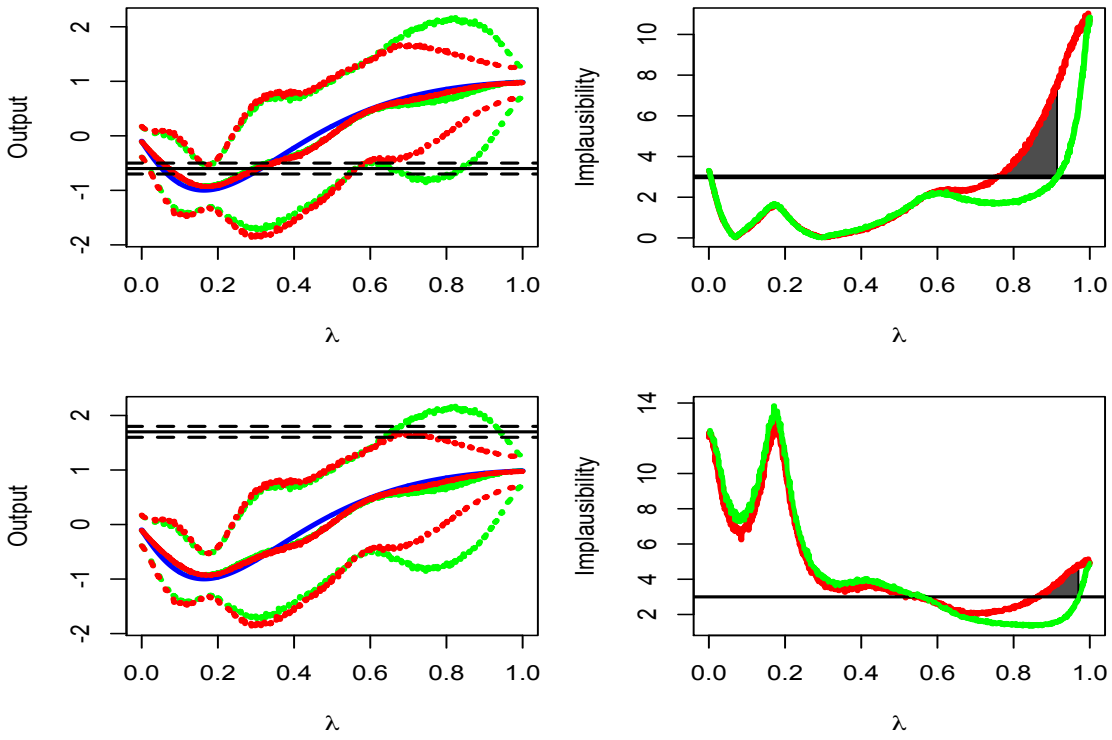

Figure 4.26: *Left*: Predictions and two standard deviation prediction intervals for stationary (green) and nonstationary (red) GP emulators for a line between two design points $\mathbf{x}_1$ and $\mathbf{x}_2$ in 2D space, with this line given by $\lambda\mathbf{x}_2 + (1-\lambda)\mathbf{x}_1$. The blue line shows the toy function, red line shows the predictions together with the two standard deviation prediction intervals produced by nonstationary emulator and green lines shows the stationary emulator performance. The observation is in black, observed with an observation error given by the dotted black lines. *Right*: The implausibility $\mathcal{I}(\mathbf{x})$ for the two emulators. The black line is the threshold set at 3 for ruling out points. Grey shaded region correspond to the part of region that is ruled out by nonstationary GP emulator and is not ruled out by stationary GP emulator.

For case 1, we could consider the composition of NROY space after Wave 1 since we could reproduce the observation value with our simulator. This allows us to check if our emulators are not incorrectly ruling out points close to z or are leaving regions of space that give output far from z (Salter and Williamson, 2016). We produce density plots of the function outputs weighted by $e^{-\mathcal{I}(\mathbf{x})}$ at samples from NROY space. Salter and Williamson (2016) pointed out that for a uniform prior on the best input, $\mathbf{x}^*$, and with normality assumptions given in Kennedy and O'Hagan (2001), the likelihood of the observation z is proportional to $e^{-\mathcal{I}(\mathbf{x})}$ in NROY space.

As a result, we could consider our sample from NROY space weighted by $e^{-\mathcal{I}(\mathbf{x})}$ to be a sample from the posterior distribution $p(\mathbf{x}^*|\mathrm{F}_{[1]})$, assuming zero likelihood at points ruled out at Wave 1.

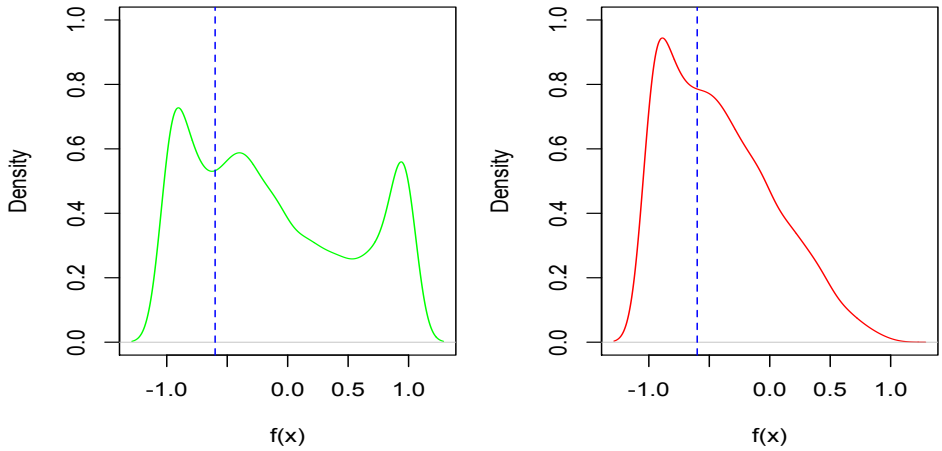

Figure 4.27: The weighted densities for the function output at points in NROY space after Wave 1 for the first case for stationary GP emulator (green) and nonstationary GP emulator (red). The observation is given by the blue dashed line.

Figure 4.27 shows the weighted densities for the NROY space defined after Wave 1. We observe that the stationary GP density is bimodal with the second mode centred around 1, which is far from the observation. Both densities exhibit bias; however, we notice that nonstationary GP assigns more weight closer to z.

The parameter plots for points in the Wave 1 NROY space where stationary and nonstationary GP emulators have been used for case 1 are shown in Figure 4.28. The size of the resulting NROY space is not the only important result, and we add the runs in the "true" NROY space overlaid in green. The "true" NROY space was found by computing implausibility function with function value in place of emulator expectation and setting emulator variance at 0. We observe from the parameter plots for case 1 in Figure 4.28 that we have managed to rule out only the input space close to the design points for high values of x_1 and x_2 due to the weak correlation structure in this region discussed in 4.4.1. On the contrary, using the nonstationary GP emulator, we have managed to rule out this region of input space successfully. Figure 4.29 shows the equivalent plots for case 2. We do not have "true" NROY this time, as the observation is too far from the function values. Regarding case

2, we observe that using the stationary GP emulator we are struggling to rule out the region of input space with $x_1, x_2 > 0$ due to higher uncertainty of the emulator about this region.

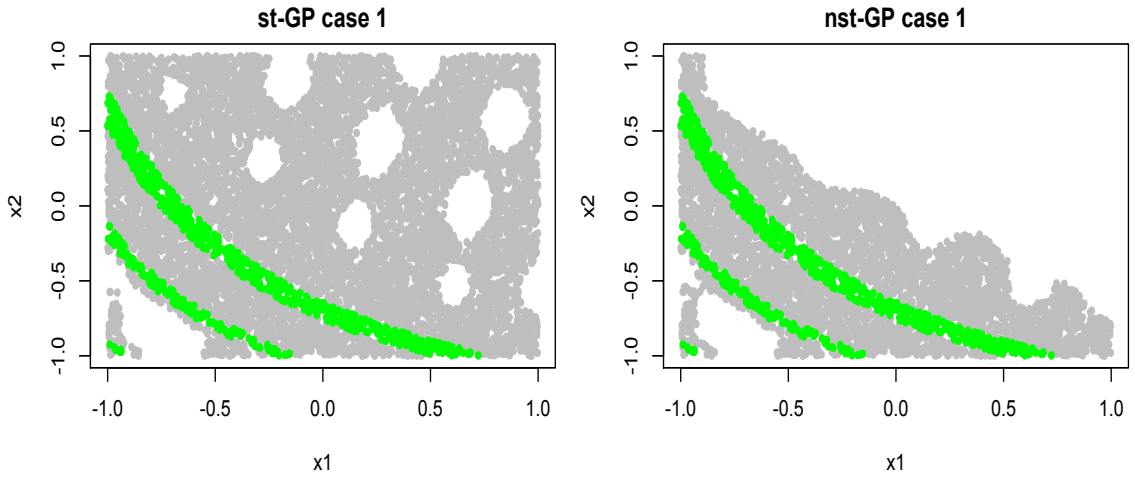

Figure 4.28: Parameter plots showing the "true" NROY space (green) and points classified as being NROY space obtained with stationary and nonstationary GP emulators after a single wave of history matching (grey) for case 1.

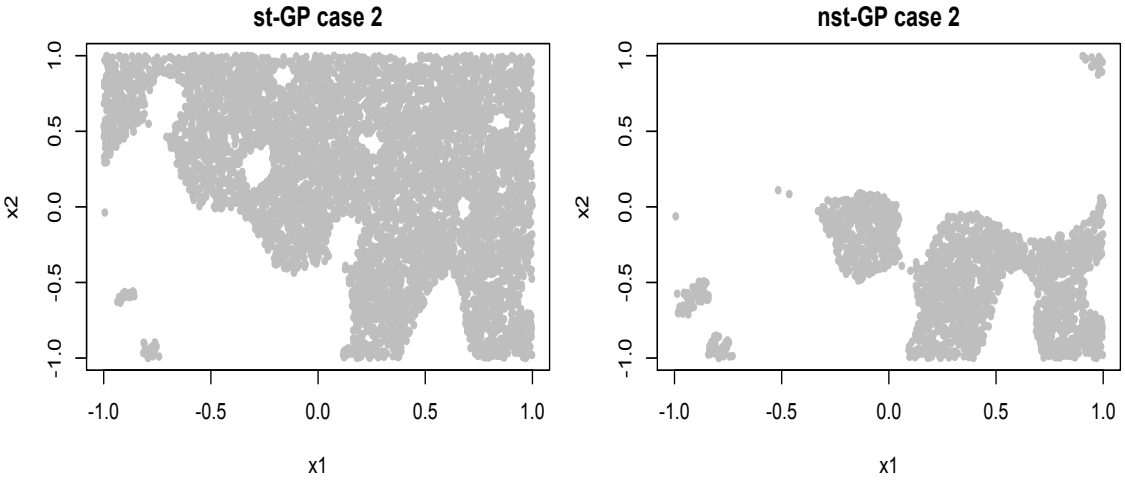

Figure 4.29: Parameter plots showing points classified as being NROY space obtained with stationary and nonstationary GP emulators after a single wave of history matching (grey) for case 2.

4.8 Conclusion

In this Chapter, we present our nonstationary GP emulation approach via kernel mixture. Working closely with climate modellers, we were interested in developing

an emulation strategy that is easy to understand and use, i.e. a practical tool that could be incorporated as part of `ExeterUQ` software presented in Chapter 3. We propose to employ standard emulator diagnostics (cross-validation) to fit nonstationary GP emulator, in particular to identify input regions of distinct model behaviour as well as estimate mixing functions, in the absence of expert knowledge regarding potential model input space regions and different properties within them. The process of constructing our nonstationary GP emulator consists of two major steps. Firstly, we check if the stationary GP emulator performs well by considering plots of individual standardized errors against inputs to identify any signs of nonstationarity/heteroskedasticity (Bastos and O'Hagan, 2009). We then fit a mixture model to the standardized errors to produce a mixture function prescribing the covariance kernel mixture for a nonstationary GP. We specify region-specific stationary covariance kernels, then establish the covariance kernel for our nonstationary GP as a mixture of these. The numerical examples, together with the real-data application, demonstrated the competitive performance of our method compared to the main nonstationary methods implemented in software, TGP and CGP. We have also considered the importance of using nonstationary GP emulators for history matching when we are dealing with nonstationary computer model response.

There is a number of possible extensions to our developed methodology. Firstly, we may remove the two-stage approach altogether by operating with the joint prior distribution, $\pi(\boldsymbol{\beta}, \boldsymbol{\sigma}_L^2, \boldsymbol{\tau}_L^2, \boldsymbol{\delta}_L, \boldsymbol{\lambda}(\mathbf{x})_L, L)$, using reversible jump MCMC (Green, 1995; Kim et al., 2005) to attempt full Bayesian inference in a similar manner to Voronoi tessellation GP model (Pope et al., 2018). However, this approach will not be diagnostic-driven as well as it could become more time-consuming and operationally expensive due to implementing reversible jump MCMC Green (1995); Pope et al. (2018). We are also cautious that this model specification could raise identifiability issues for $\mathbf{x}$, in particular to which input region $\mathbf{x}$ should be allocated, affecting the robustness and stability of the inferences produced by nonstationary GP model.

In Chapter 6, we employ our nonstationary GP emulator in refocusing as part of a climate model application. In particular, we are interested in exploring how the

Bayesian Design Criterion, introduced in Chapter 5, modifies when it is combined with our proposed nonstationary GP model.

Chapter 5

Bayesian Optimal Design for multi-wave computer experiments

5.1 Introduction

History matching is a type of calibration that rules out input parameter settings that are inconsistent between observations and computer model output according to a distance-based implausibility function and uncertainty specifications (Craig et al., 1996; Williamson and Vernon, 2013; Williamson et al., 2013). History matching has proven to be more effective when it is performed in several waves by iteratively cutting down the input space of a computer model (Vernon et al., 2010; Salter and Williamson, 2016). Performing history matching in waves, termed iterative refocusing, leads to emulator performance improvement since the density of the ensemble increases in the reduced space (Williamson et al., 2017).

We provide a short summary of the process of iterative refocusing: for more details, see subsection 2.6.3. During each step, m, of iterative refocussing, we are required to obtain a design, $\mathrm{X}_{[m]} \in \mathcal{X}^{m-1}$, and generate computer model runs corresponding to the design $\mathrm{F}_{[m]}$. A new ensemble, $\left\{\mathrm{X}_{[m]}, \mathrm{F}_{[m]}\right\}$, is used to update an emulator for $f(\mathbf{x})$, and to derive the NROY space denoted as $\mathcal{X}^m \subseteq \mathcal{X}^{m-1}$ at wave m.

To generate a design for wave 1, one of many approaches of space-filling designs such as Uniform designs, Sobol sequences (Fang et al., 2005; Challenor, 2011) and

Latin Hypercubes (LHCs) (McKay et al., 1979), could be adopted since we are operating within an original input space $\mathcal{X}$. However, generating good designs for wave $m > 1$ is still an open research question, since we are required to obtain a design over the NROY space of varying geometric shapes and sizes, and the usual space-filling designs mentioned above are not applicable (Williamson et al., 2013; Gong et al., 2016; Williamson et al., 2017).

In general, the proposed design approaches for iterative refocusing are focused on generating space-filling designs over NROY space. For instance, Andrianakis et al. (2017) proposed to generate a maximin design over NROY space, whist Williamson et al. (2013) and Gong et al. (2016) employed the partitioning of input space to obtain a uniform design over NROY space.

In this Chapter, we propose a new approach to obtaining a design, $\mathrm{X}_{[m]}$, at step m of iterative refocusing, based on a decision-theoretic approach (Chaloner and Verdinelli, 1995). We employ Bayesian experimental design where the Bayesian optimal design is found by minimizing the expected loss function over the design space with respect to future computer model runs and the "truth", a state of nature that we attempt to learn. We specify a loss function that compares the volume of the NROY space, $\mathcal{X}^m$, obtained using the updated emulator for $f(\mathbf{x})$ with a candidate design, to the volume of the "true" NROY space obtained using a "perfect" emulator. In a decision-theoretic context, we treat the volume of "true" NROY space as the "truth". The intuition behind the proposed loss function corresponds to the aim of history matching. In particular, by performing history matching, we want to ensure that we are not going to incorrectly rule out points that are in fact close to observations ,or leave regions of space that give output far from the observations (Salter and Williamson, 2016).

The Chapter has the following structure. In section 5.2, we discuss various approaches adopted in the literature for generating designs for multi-wave computer experiments as well as follow-up designs. We introduce our Bayesian Design Criterion for multi-wave history matching and consider each term of the design criterion in detail in section 5.3. In section 5.4, we describe the process of computing our

Bayesian Design Criterion. The performance of our proposed approach is demonstrated on a simple 2D toy model in section 5.5. We finish off with a discussion, and propose some future developments and extensions in section 5.6.

5.2 Follow-up designs for computer models

In this section, we review some of the approaches used to generate a follow-up design for computer models in the context of calibration and history matching. We consider the design for wave m, $\mathrm{X}_{[m]}$, as a follow-up design, since we do have some sort of preliminary information. This preliminary information includes a collection of designs generated at previous iterations of history matching, $\langle \mathrm{X} \rangle_{[m-1]} = \left\{ \mathrm{X}_{[1]}, \ldots, \mathrm{X}_{[m-1]} \right\}$, and the corresponding computer model runs, $\langle \mathrm{F} \rangle_{[m-1]} = \left\{ \mathrm{F}_{[1]}, \ldots, \mathrm{F}_{[m-1]} \right\}$, as well as the not ruled out space (NROY) obtained at the previous iterations of history matching, $\mathcal{X}^{m-1}$. We start by considering sampling approaches over the NROY space that are used to obtain a design for wave m (Vernon et al., 2010; Williamson and Vernon, 2013; Gong et al., 2016; Andrianakis et al., 2017). These approaches are mainly focused on obtaining a design that is well-spaced across the NROY space from a previous iteration of history matching. This is particularly problematic to achieve if the NROY space has a very small size in relation to the original input space. Another issue arises if the NROY space is characterized by a non-regular geometric shape, making it challenging to ensure the specific space-filling property of design is satisfied. We then move on to sequential design approaches used for calibration and history matching (Craig et al., 1996; Ranjan et al., 2008, 2011). This type of design is obtained by optimizing a pre-defined design criterion. This design criterion could take into account both exploration, i.e. sampling areas of high uncertainty about computer model output, and exploitation, i.e. sampling areas likely to obtain computer model outputs close to observations (Brochu et al., 2010).

5.2.1 Non-Implausible Sampling

A number of approaches for generating a design, $\mathrm{X}_{[m]}$, for wave m are focused around the idea of exploring the NROY space, $\mathcal{X}^{m-1}$, obtained at the previous iteration of

history matching.

Vernon et al. (2010) propose to use a rejection sampling to generate a design $X_{[m]}$. Firstly, a Latin Hypercube over the original input space, $\mathcal{X}$, is produced and only those points, not ruled out by history matching, are retained. The process is repeated until a desired number of design points is obtained. This approach is very easy to implement and follow; however, it is not clear what the properties of the obtained design over the NROY space are, i.e. this type of design might not offer a good coverage of the NROY space. For example, this type of design is not suitable when we are dealing with the NROY space of a small volume relative to the original input space, since it could be difficult to retain any points in NROY space generated from a Latin Hypercube defined over the whole input space. For instance, Williamson and Vernon (2013) considered an NROY of a galaxy simulation model called GALFORM, following four previous waves of history matching, which was 0.001% the size of the volume of the original input space.

Andrianakis et al. (2017) are interested in obtaining a space-filling design, $X_{[m]}$, over the NROY space, $\mathcal{X}^{m-1}$, as this type of design is expected to be informed about computer model output behaviour across the NROY space. Firstly, a large number of points distributed uniformly across the NROY, $\mathcal{X}^{m-1}$, is produced. The first design point is chosen at random, and the second one is chosen as the point which is furthest away from the first, by considering Euclidean distances. The rest of the points are obtained iteratively by maximizing the minimum Euclidean distance between design points. This fast approach is believed to generate a well-spread design, $X_{[m]}$, with good coverage over the NROY space. However, it is arguable that by employing this method, i.e. considering Euclidean distance, we could actually be able to produce a maximin design over the NROY space since we are not operating in Euclidean space, as $\mathcal{X}^{m-1}$ usually possesses unusual geometric shapes, or could consist of disjoint regions.

More sophisticated approaches have been employed in order to obtain uniform designs for multi-wave computer experiments. Williamson and Vernon (2013) presented the Implausibility Driven Evolutionary Monte Carlo algorithm (IDEMC) to

obtain a uniform design for history matching. To obtain a design for step m of history matching, they propose to start by generating an initial set of starting points, $\mathrm{X}^{(0)} = \left\{\mathbf{x}_0^{(0)}, \mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_n^{(0)}\right\}$, in the current NROY space, $\mathcal{X}^{m-1}$, with $\mathbf{x}_i^{(0)} \in \mathcal{X}_i^{m-1}$ and $\mathcal{X}_i^{m-1} = \left\{\mathbf{x} \in \mathcal{X}^{m-1} : \mathcal{I}(\mathbf{x}_i) \leq b_i\right\}$ for $i > 0$ and $b_1, \ldots, b_n$, where $\mathcal{X}_i^{m-1}$ with $i = 1, \ldots, n$ is a well-chosen implausibility ladder. Points $\mathbf{x}_i, i = 1, \ldots, n$ are defined as choromosomes. Each individual level of the implausibility ladder, $\mathcal{X}_i^{m-1}, i = 1, \ldots, n$, is further partitioned, i.e. $\mathcal{X}_{i1}^{m-1}, \ldots, \mathcal{X}_{ir_i}^{m-1}$, and an appropriate variance matrix, V_{ij}, for sampling $\mathbf{x}_i$ from $\mathcal{X}_{ij}^{m-1}$ is specified. This approach is employed to generate candidate points for the Evolutionary Monte Carlo algorithm. The t iterations of Evolutionary Monte Carlo are performed with three types of update move to obtain a uniform design: mutation, crossover, and exchange. Williamson and Vernon (2013) demonstrated that employing this algorithm leads to a uniform design with good coverage over the current NROY space, in particular when the volume of NROY space is small in relation to the original input space, $\mathcal{X}$. However, the efficiency of the proposed IDEMC algorithm is determined by a number of factors, such as the number and setting of the implausibility ladder. In particular, by increasing the number of levels for the implausibility ladder, better spaced chromosomes sample spaces could be achieved, which leads to mixing improvement. However, the trade-off is an increase in the computational cost of the algorithm. Further, defining an initial set of starting points, $\mathrm{X}^{(0)}$, could be challenging when we are dealing with an NROY space of a very small size.

Another method for obtaining a uniform design over the NROY space has been developed by Gong et al. (2016). They propose to use Subset Simulation (SuS), a widely used technique in engineering reliability computations, to sample from NROY space. They propose to generate a sequence of subsets, where the current NROY space $\mathcal{X}^{m-1}$ is defined as $F = \left\{\mathbf{x} : \mathcal{I}(\mathbf{x}) \leq a\right\} = F_d \subset F_{d_1} \subset \cdots \subset F_1$. The subset F is treated as a rare event and the probability of a rare event F can be represented as a product of larger probabilities, i.e.

$$P(F) = P(F_d) = P(F_1) \times P(F_2|F_1) \times \cdots \times P(F_d|F_{d-1}).$$

The algorithm begins by sampling n design points in the whole input space, $(\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_n^{(0)})$ and computing the implausibility function at these points. The values of the corresponding implausibilities are ordered, $(\mathcal{I}_1^{(0)}, \ldots, \mathcal{I}_n^{(0)})$. Let $p \in (0, 1)$ represent a user-specified probability parameter. The first intermediate failure domain is defined as

$$F_1 = \left\{ \mathbf{x} : \mathcal{I}(\mathbf{x}) < \frac{\mathcal{I}_{np}^{(0)} + \mathcal{I}_{np+1}^{(0)}}{2} \right\},$$

and the next step is to generate samples from F_1, on which the subsequent levels are conditioned. More intermediate events are added following the described procedure. This process is easy to follow, however the user should be careful in specifying n, the number of samples for each level, as it could affect the computational speed of the MCMC algorithm. It is also not clear how to specify the probability parameter, p, for the proposed algorithm. After obtaining samples from the NROY space by employing SUS, a small subset of points is chosen by maximizing a minimum Euclidean distance between points in order to generate a space-filling design, which is similar to the approach adopted by Andrianakis et al. (2017). The partitioning of the input space has some similarities to the Implausibility Driven Evolutionary Monte Carlo algorithm proposed by Williamson and Vernon (2013).

5.2.2 Sequential Experiment Design

Instead of sampling uniformly over the NROY space by employing one of the methods discussed in subsection 5.2.1, analysts might be interested in obtaining additional design points that could help them to capture and reduce prediction uncertainty. In this case, sequential design approaches can be adopted, i.e. additional design points are chosen by optimizing a criterion based on prediction errors (Sacks et al., 1989) or entropy (Shewry and Wynn, 1987; Currin et al., 1991). In this subsection, we are interested in reviewing sequential design approaches used for inverse problems, where the objective is to determine the input parameter settings that produce a specific value of computer model output. We consider calibration and in particular history matching as different solutions to an inverse problem.

Sequential approaches are widely used to guide users as to where in the input

space to evaluate a complex computer model (Santner et al., 2003; Lam, 2008; Beck and Guillas, 2016). Ranjan et al. (2008) present a sequential design approach for estimating a contour of a complex computer model, where a contour identifies a boundary that distinguishes "good" and "bad" performance. The contour problem is defined as

$$S(a) = \big\{\mathbf{x} \in \mathcal{X} : f(\mathbf{x}) = a\big\},$$

where a is a value of response surface for which the contour is estimated. The strategy is to perform a relatively small experimental design, and sequentially choose design points that are on or near the estimate of the contour. In particular, they are interested in choosing new design points to evaluate the computer model at the points where $E[f(\mathbf{x})]$ belongs to a neighbourhood, $(a-\epsilon, a+\epsilon)$, of the current contour estimate with $\epsilon(\mathbf{x}) = \alpha\sqrt{Var[f(\mathbf{x})]}$ for some constant α. A new design point, $\mathbf{x}$, is chosen for which the expected improvement function is maximized, where the improvement function is defined as

$$I(\mathbf{x}) = \epsilon^2(\mathbf{x}) - \min\big\{(f(\mathbf{x}) - a)^2, \epsilon^2(\mathbf{x})\big\}.$$

The term $\epsilon(\mathbf{x})$ is used to define a neighbourhood around the contour and is a function of $\sqrt{Var[f(\mathbf{x})]}$. As a result, we expect to see an increase in the improvement function value where $Var[f(\mathbf{x})]$ is high. The attractive feature of the proposed design criterion, $E[I(\mathbf{x})]$, is that it has easily interpretable terms that ensure sampling in the input region close to the contour, as well as in the regions where the uncertainty of the prediction is high. We conclude that this design criterion encourages both exploitation and exploration. Choosing design points sequentially, one at a time, works well for network queuing problems such as those considered by Ranjan et al. (2008). However, due to the associated cost constraints and experimental settings, it is often preferred to choose a follow-up design in a batch of a pre-specified number of trials (Loeppky et al., 2010).

Ranjan et al. (2011) adopt an approach to obtaining sequential batch designs for the calibration problem discussed in detail in section 2.6. Given the existing

responses $\boldsymbol{d}^T = (\mathrm{F}^T, \boldsymbol{z}^T)$ and designs $\mathrm{X_F}$ and X_z, the mean squared error (MSE) for predicting the true physical process at an unobserved location, $\mathbf{x}$, is given by

$$MSE[z(\mathbf{x})|\Omega] = E\Big[\big(E(z(\mathbf{x})|\boldsymbol{d}, \Omega) - z(\mathbf{x})\big)^2|\Omega\Big],$$

where $\Omega = \left\{\mathbf{x}^*, \boldsymbol{\delta}_f, \boldsymbol{\delta}_\eta, \sigma_f^2, \sigma_\eta^2, \lambda, \boldsymbol{\beta}_f, \boldsymbol{\beta}_\eta\right\}$ with $\left\{\boldsymbol{\delta}_f, \sigma_f^2, \boldsymbol{\beta}_f\right\}$ being the parameters used in Gaussian Process specification for $f(\cdot, \cdot)$ and $\left\{\boldsymbol{\delta}_\eta, \sigma_\eta^2, \boldsymbol{\beta}_\eta\right\}$ being the parameters used in Gaussian Process specification for the model discrepancy term $\eta(\cdot)$, and $\mathbf{x}^*$ is a vector of calibration parameters given in equation (2.26). The plug-in estimate (the posterior mean) of Ω is used to evaluate $MSE[z(\mathbf{x})|\Omega]$ in order to reduce the computational cost. Ranjan et al. (2011) are interested in improving predictions throughout the entire input space, $\mathcal{X}$. Therefore, a batch of designs ξ of size $m_z + m_f$, where m_z and m_f are the number of new observations and computer simulations respectively, is chosen according to the integrated mean squared error (IMSE) criterion,

$$IMSE(\xi) = \int_{\mathcal{X}} MSE[z(\mathbf{x})|\xi] d\mathbf{x},$$

with an optimal design ξ^* found by minimizing IMSE criterion, i.e.

$$\xi^* = \arg\min IMSE(\xi).$$

The input space $\mathcal{X}$ (assumed to be a $[0,1]^q$ for the observations and $[0,1]^{p+q}$ for the computer model) leads to a $m_z q + m_f(p+q)$-dimensional optimization problem, which is computationally expensive. Interestingly, the cost of the proposed methodology could be reduced by aligning the proportion of candidate design points at which we obtain new observations with the existing computer model simulations and/or observations, i.e. reducing the dimensionality of the optimization problem, since m_z is decreased. The alignment of new observations with computer model simulations could be useful in updating our beliefs about the model discrepancy term, while replication of observations could be informative about the observation error term. The simulation study presented by Ranjan et al. (2011) demonstrates a significant effect on IMSE reduction when a combination of observation and computer

simulations (either aligned or IMSE optimal) is considered. However, there is no theoretical guidance on how many runs should be retained for alignment/replication when using IMSE design criteria.

Craig et al. (1996) proposed a sequential design for history matching, and applied this method for matching hydrocarbon reservoir history. We note that Craig et al. (1996) employed Bayes linear methods to model their beliefs about computer model response behaviour. At step m of history matching the prior knowledge about f is specified via a Gaussian process distribution with mean function $E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]$ and variance function $Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]$. Using probabilistic notation, the probability distribution for $f(\mathbf{x})$ conditioned on the statistical parameters $\{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}, \tau^2\}$ is

$$f(\mathbf{x})|\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}, \tau^2 \sim GP\Big(E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})], Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]\Big). \tag{5.1}$$

At step m of history matching, a new ensemble, $\{\mathrm{X}_{[m]}, \mathrm{F}_{[m]}\}$, is obtained to update the distribution for f at a new input $\mathbf{x}$ given a new ensemble, $\{\mathrm{X}_{[m]}, \mathrm{F}_{[m]}\}$, and parameters $\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}, \tau^2$, which is defined as

$$f(\mathbf{x})|\{\mathrm{X}_{[m]}, \mathrm{F}_{[m]}\}, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}, \tau^2 \sim GP\Big(E_{\langle \mathrm{F} \rangle_{[m]}}[f(\mathbf{x})], Var_{\langle \mathrm{F} \rangle_{[m]}}[f(\mathbf{x})]\Big), \tag{5.2}$$

with mean function

$$\begin{aligned} &E_{\langle \mathrm{F} \rangle_{[m]}}[f(\mathbf{x})] = E_{\{\langle \mathrm{F} \rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})] \qquad (5.3)\\ &= E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})] + Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x}), \mathrm{F}_{[m]}]\big(Var_{\langle \mathrm{F} \rangle_{[m-1]}}[\mathrm{F}_{[m]}]\big)^{-1}\big(\mathrm{F}_{[m]} - E_{\langle \mathrm{F} \rangle_{[m-1]}}[\mathrm{F}_{[m]}]\big) \end{aligned}$$

and variance function

$$\begin{aligned} &Var_{\langle \mathrm{F} \rangle_{[m]}}[f(\mathbf{x})] = Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})] \qquad (5.4)\\ &= Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})] - Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x}), \mathrm{F}_{[m]}]\big(Var_{\langle \mathrm{F} \rangle_{[m-1]}}[\mathrm{F}_{[m]}]\big)^{-1} Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[\mathrm{F}_{[m]}, f(\mathbf{x})]. \end{aligned}$$

Craig et al. (1996) are interested in choosing the design at step m, $\xi = \mathrm{X}_{[m]}$, at which to obtain computer model runs, $f(\xi) = \mathrm{F}_{[m]}$, that leads to the reduction of uncertainty over the input space. They propose to define a collection of computer runs,

$\langle \mathrm{F} \rangle_{[m]} = \{\mathrm{F}_{[1]}, \ldots, \mathrm{F}_{[m]}\}$, corresponding to input designs, $\langle \mathrm{X} \rangle_{[m]} = \{\mathrm{X}_{[1]}, \ldots, \mathrm{X}_{[m]}\}$, obtained up to step m. The expected reduction in variance of $f(\mathbf{x})$ at $\mathbf{x}$ by evaluating the computer model at $\mathrm{X}_{[m]}$ is measured by

$$\Big(Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]\Big)^{-1} RVar_{\mathrm{F}_{[m]}/\langle \mathrm{F} \rangle_{[m-1]}}(f(\mathbf{x})), \tag{5.5}$$

where $Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]$ corresponds to the variance function of the GP distribution for f at new input $\mathbf{x}$, given in equation (5.4).

The second term is a partial resolved variance of $f(\mathbf{x})$ by a new ensemble $\mathrm{F}_{[m]}$ given $\langle \mathrm{F} \rangle_{[m-1]}$, defined as

$$\begin{aligned} RVar_{\mathrm{F}_{[m]}/\langle \mathrm{F} \rangle_{[m-1]}}(f(\mathbf{x})) = Cov_{\langle \mathrm{F} \rangle_{[m-1]}}(f(\mathbf{x}), \mathrm{F}_{[m]}) [Var_{\langle \mathrm{F} \rangle_{[m-1]}}(\mathrm{F}_{[m]})]^{-1} \\ \times Cov_{\langle \mathrm{F} \rangle_{[m-1]}}(\mathrm{F}_{[m]}, f(\mathbf{x})). \end{aligned}$$

In Bayes Linear theory this term is used to measure the effect on beliefs about $f(\mathbf{x})$ by observing $\mathrm{F}_{[m]}$, given a collection of previously obtained computer model runs $\langle \mathrm{F} \rangle_{[m-1]}$ (Goldstein and Wooff, 2007).

Finally, the overall weighted reduction in variance of $f(\mathbf{x})$ by obtaining simulation runs at the candidate design ξ is given by

$$U(\xi) = \int w(\mathbf{x}) \Big(Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]\Big)^{-1} RVar_{\mathrm{F}_{[m]}/\langle \mathrm{F} \rangle_{[m-1]}}(f(\mathbf{x})) d\mathbf{x}, \tag{5.6}$$

where the weight $w(\mathbf{x})$ is a function of the current plausibility evaluation of $\mathbf{x}$ based on the values for the adjusted expectation and variance. Craig et al. (1996) propose to define $w(\cdot)$ as a decreasing function of $c(\cdot)$, where

$$c(\mathbf{x}) = \frac{\big(E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]\big)^2}{Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]}.$$

The proposed function for plausibility calculations does not contain any information about observation z, the variance of the observation error $Var[e]$, nor the model discrepancy variance $Var[\eta]$.

Craig et al. (1996) propose to choose ξ^* by maximizing $U(\cdot)$, i.e.

$$\xi^* = \arg\max U(\xi),$$

where the next design points are chosen as the ones that lead to the overall reduction in uncertainty about the surface.

Interestingly, Craig et al. (1996) extended the proposed design to refocusing when considering q outputs of simulator, the multi-dimensional implausibility, i.e.

$$U(\xi) = \sum_{i=1}^{q} l_i U_i(\xi),$$

where the weights l_i reflect the relative importance of matching the various outputs $f_i(\mathbf{x})$, and $U_i(\mathrm{X}_{[m]})$ is defined as

$$U_i(\mathrm{X}_{[m]}) = \int w(\mathbf{x}) \Big(Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f_i(\mathbf{x})] \Big)^{-1} RVar_{\mathrm{F}_{[m]}/\langle \mathrm{F} \rangle_{[m-1]}}(f_i(\mathbf{x})) d\mathbf{x} \quad i = 1, \ldots, q.$$

The function $c(\cdot)$ is transformed to

$$c(\mathbf{x}) = \max_i \frac{\big(E_{\langle \mathrm{F} \rangle_{[m-1]}}[f_i(\mathbf{x})] \big)^2}{Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f_i(\mathbf{x})]}.$$

The values of weights l_i are suggested to be elicited from experts, however the method (framework) of deriving these weights is not provided. Craig et al. (1996) recognises that using this design criterion could be computationally challenging, especially when operating within a high-dimensional input space. Significant computational savings are achieved by obtaining a collection of active inputs, and modelling the global and the residual components of $f(\mathbf{x})$ in terms of active inputs, effectively adopting a dimensional reduction approach. As a result, a $p \times n_m$ optimisation problem is reduced down to $p^A \times n_m$, with $p^A < p$, where p is the number of input parameters and p^A is the number of active input parameters. Although the design criterion definition provided by equation (5.6) implies that we are considering the whole input space $\mathcal{X}$, Craig et al. (1996) demonstrated the application of design criteria on a smaller region for which the plausibility weights fulfil $w(\mathbf{x}) > 2$,

mimicking the idea of refocusing in which the computer model response behaviour is analysed only over the NROY space.

5.3 Bayesian Optimal Design for iterative refocussing

Bayesian methodologies for obtaining optimal experimental design allow users to incorporate prior information and uncertainties regarding the statistical model together with a utility/loss function that explicitly describes the inferential aim (Ryan et al., 2016). We are interested in adopting Bayesian Optimal design for refocussing with a loss function that incorporates our goal to obtain the NROY space that contains all the points close to observations with fewer iterations of history matching.

We start by defining a candidate design for wave m of history matching as $\xi = \mathrm{X}_{[m]}$ at which to obtain computer model runs, $f(\xi) = \mathrm{F}_{[m]}$, using a Bayesian Design Criterion (BDC), where the Bayesian Optimal design (BOD), ξ^*, is found by minimizing the BDC (expected loss).

We define the expected loss as

$$\Psi(\xi) = \int\int L(V_{\mathcal{X}^m}, V_{\mathcal{X}_T}; \xi)\pi(V_{\mathcal{X}^m}, V_{\mathcal{X}_T}; \xi)dV_{\mathcal{X}^m}dV_{\mathcal{X}_T}, \tag{5.7}$$

and follow the formulation of the Bayesian design problem as a decision problem introduced by Chaloner and Verdinelli (1995) with the following basic elements:

- the "truth" (the true nature of the state), the volume of the "true" NROY space, denoted by $V_{\mathcal{X}_T}$ and defined as

$$V_{\mathcal{X}_T} = \int_{\mathcal{X}} \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\}d\mathbf{x} = \int_{\mathcal{X}} \mathbb{1}\left\{\frac{|z - f(\mathbf{x})|}{\sqrt{Var[e] + Var[\eta]}} \leq a\right\}d\mathbf{x}, \tag{5.8}$$

  where we adopt the assumption of a "perfect" emulator, i.e. we use a complex computer model directly instead of an emulator, and as a result, we have no uncertainty about computer model outputs.

- a decision, the volume of the NROY space at wave m, denoted by $V_{\mathcal{X}^m}$ and determined by a candidate design, ξ. We define $V_{\mathcal{X}^m}$ mathematically as

$$\begin{aligned} V_{\mathcal{X}^m} &= \int_{\mathcal{X}^{m-1}} \mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} d\mathbf{x} \\ &= \int_{\mathcal{X}^{m-1}} \mathbb{1}\left\{ \frac{|z - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \right\} d\mathbf{x}, \end{aligned} \tag{5.9}$$

  where values of $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$ and $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$ are obtained by computing equations (5.3) and (5.4) respectively.

- a loss function, $L(V_{\mathcal{X}^m}, V_{\mathcal{X}_T}, \xi)$, that quantifies our loss from choosing a candidate design, $\xi \in \mathcal{X}^{m-1}$, to obtain the volume of the NROY space, $V_{\mathcal{X}^m}$, when the volume of the "true" NROY space is $V_{\mathcal{X}_T}$. We define our loss function as

$$\begin{aligned} L(V_{\mathcal{X}^m}, V_{\mathcal{X}_T}; \xi) &= \int_{\mathcal{X}} \left(\mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} - \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \right)^2 d\mathbf{x} \\ &= \int_{\mathcal{X}} \mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} d\mathbf{x} - 2 \int_{\mathcal{X}} \mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} \times \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} d\mathbf{x} + \int_{\mathcal{X}} \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} d\mathbf{x}. \end{aligned} \tag{5.10}$$

  The first and the last terms of our loss function correspond to equation (5.9) and equation (5.8) respectively, while the second term, $\int_{\mathcal{X}} \mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} \times \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} d\mathbf{x}$, is used to measure the volume of the input region that is in both the NROY obtained at wave m and the "true" NROY.

Finally, the BOD, ξ^*, is obtained by minimizing the expected loss function, $\Psi(\xi)$, i.e.

$$\xi^* = \arg\min \Psi(\xi).$$

5.3.1 Expected Loss Function

We proceed to consider in detail our proposed loss function, given in equation (5.10). Since we do not know the volume of the "true" NROY space, we have to operate with the expected loss, and we require a joint density function, $\pi(V_{\mathcal{X}^m}, V_{\mathcal{X}_T}|\xi)$, to compute it.

Based on equations (5.9) and (5.8), we derive the following form of the expected

loss

$$\Psi(\xi) = \int\int \left[\int_{\mathcal{X}} \Big(\mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} - \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \Big)^2 d\mathbf{x} \right] \pi(V_{\mathcal{X}^m}, V_{\mathcal{X}_T}; \xi) dV_{\mathcal{X}^m} dV_{\mathcal{X}_T}$$
$$= \int\int \left[\int_{\mathcal{X}} \Big(\mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} - \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \Big)^2 d\mathbf{x} \right] \pi(V_{\mathcal{X}_T} | V_{\mathcal{X}^m}; \xi) \pi(V_{\mathcal{X}^m}; \xi) dV_{\mathcal{X}^m} dV_{\mathcal{X}_T}.$$

Since the variability in $V_{\mathcal{X}_T}$ is determined by $f(\mathbf{x})$ and the variability in $V_{\mathcal{X}^m}$ is determined by $f(\xi)$, we could re-write the above equation as

$$\begin{aligned}
\Psi(\xi) &= \int \int_{\mathcal{X}} \int \Big[\mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} - \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \Big]^2 \pi(f(\mathbf{x}) | f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \\
&\times \pi(f(\xi) | \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} \pi(f(\mathbf{x}) | f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \pi(f(\xi) | \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&- 2 \int \int_{\mathcal{X}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \pi(f(\mathbf{x}) | f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \\
&\times \pi(f(\xi) | \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&+ \int \int_{\mathcal{X}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \pi(f(\mathbf{x}) | f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \pi(f(\xi) | \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \Psi_1(\xi) - 2\Psi_2(\xi) + \Psi_3(\xi)
\end{aligned} \tag{5.11}$$

where individual terms have their own interpretation and we proceed further to consider these terms in detail. Note that the probability distributions for $f(\mathbf{x})$ and $f(\xi)$ are both conditioned on the collection of ensembles obtained at previous iterations of history matching, i.e. $\big\{ \langle \mathrm{X} \rangle_{[m-1]}, \langle \mathrm{F} \rangle_{[m-1]} \big\}$, which corresponds to the process of updating the posterior distribution for $f(\mathbf{x})$ from the previous wave of HM. We briefly mentioned the process of updating the posterior distribution for $f(\mathbf{x})$ in our discussion of design approach proposed by Craig et al. (1996) in subsection 5.2.2.

A full Bayesian approach could be implemented by integrating $\Psi(\xi)$ over the posterior distribution of GP hyperparameters, $\Theta = \{\beta, \sigma^2, \boldsymbol{\delta}, \tau^2\}$, i.e.

$$\begin{aligned}
\Psi(\xi) = \int \int \int_{\mathcal{X}} \int \Big[\mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} - \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \Big]^2 \pi(f(\mathbf{x}) | f(\xi), \langle \mathrm{F} \rangle_{[m-1]}, \Theta) \\
\times \pi(\Theta | f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \pi(f(\xi) | \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} d\Theta df(\xi),
\end{aligned} \tag{5.12}$$

where $\pi(\Theta|f(\xi), \langle \mathrm{F} \rangle_{[m-1]})$ is the posterior distribution of Θ and $\pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) = \int \pi(f(\xi)|\Theta, \langle \mathrm{F} \rangle_{[m-1]})\pi(\Theta|\langle \mathrm{F} \rangle_{[m-1]})d\Theta$ is the marginal distribution of data over the prior distribution of Θ for wave m. A full Bayesian approach allows us to take into account the effect of candidate design on the GP hyperparameters' values, as well as the uncertainty about these values. However, it is computationally expensive since for a single realisation of $f(\xi)$ we are required to obtain a posterior distribution $\pi(\Theta|f(\xi), \langle \mathrm{F} \rangle_{[m-1]})$ (Ryan et al., 2016).

To reduce computational costs, we propose fix the values of the GP hyperparameters at maximum a posteriori (MAP) estimates obtained during the first wave of history matching. In cases where we are interested in employing BDC to generate the initial design, we have to perform a full Bayesian analysis since we do not have these values.

To analyse individual terms of BDC, we return to the definition of expected loss provided in equation (5.11).

5.3.2 Derivation and interpretation of $\Psi_1(\xi)$

The first term of the expected loss function, $\Psi_1(\xi)$, corresponds to the expected volume of NROY space at wave m. We are interested in minimizing $\Psi_1(\xi)$ as part of the proposed Bayesian Design Criterion. This corresponds to our aim to minimize the NROY space volume to achieve a tighter calibration.

To derive the volume of NROY space at wave m, we do not need to compute integration over the whole input space, $\mathcal{X}$, since $\mathbf{x}$ is nonimplausible at wave m only if it is nonimplausible for all waves that precede it. We propose to compute the integration over $\mathcal{X}^{m-1}$, the NROY space obtained during the $m-1$th iteration of history matching for computational ease. We expand the expressions in the equation (5.9) as

$$
\begin{aligned}
\Psi_1(\xi) &= \int \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\} \pi(f(\mathbf{x})|f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \\
&\times \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\left\{ \frac{|z - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \right\} \\
&\times \pi(f(\mathbf{x})|f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}^{m-1}} \mathbb{1}\left\{ \frac{|z - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \right\} \\
&\times \left[\int \pi(f(\mathbf{x})|f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) \right] \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}^{m-1}} \mathbb{1}\left\{ \frac{|z - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \right\} \\
&\times \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) d\mathbf{x} df(\xi), \qquad (5.13)
\end{aligned}
$$

since $\int \pi(f(\mathbf{x})|f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) = 1$ (integrating over the full support of a probability density function). The integrand of $\Psi_1(\xi)$ is mainly affected by the values of expectation, $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, and variance, $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$ at input point $\mathbf{x}$ from the distribution for f. In particular, if we expect the model output to be close to z at $\mathbf{x} \in \mathcal{X}^{m-1}$, this will affect the value of an integrand of $\Psi_1(\mathbf{x})$. This argument is discussed in detail in subsection 5.3.4. More interesting is the inclusion of the variance of $f(\mathbf{x})$ in our computation of $\Psi_1(\xi)$, since this term encourages the exploration of design criterion, i.e. we are interested in obtaining computer model simulations $f(\xi)$ that would allow us to reduce the predictive uncertainty of $f(\mathbf{x})$ over the NROY space $\mathcal{X}^{m-1}$ expressed through $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$.

We could further simplify equation (5.13) (see Appendix A.1) to derive the following expression for $\Psi_1(\xi)$,

$$
\Psi_1(\xi) = \int_{\mathcal{X}^{m-1}} \Big[\Phi\big(s_2\big) - \Phi\big(s_1\big) \Big] d\mathbf{x} \qquad (5.14)
$$

where

$$s_1 = \frac{z - a\sqrt{Var[e] + Var[\eta] + Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]} - E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]}{\sqrt{Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x}), f(\xi)]\Big(Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]\Big)^{-1} Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi), f(\mathbf{x})]}}$$

$$s_2 = \frac{z + a\sqrt{Var[e] + Var[\eta] + Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]} - E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]}{\sqrt{Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x}), f(\xi)]\Big(Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]\Big)^{-1} Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi), f(\mathbf{x})]}},$$

and $\Phi(\cdot)$ is the standard normal cumulative distribution function (cdf).

5.3.3 Derivation and interpretation of $\Psi_3(\xi)$

The third term of the expected loss function, $\Psi_3(\xi)$, corresponds to the expected volume of the "true" NROY space. We are interested in expressing this term as an integral over $\mathcal{X}^{m-1}$, so that we could combine it together with $\Psi_1(\xi)$ in our proposed Bayesian Design Criterion computation. We specify $\Psi_3(\xi)$ as

$$\begin{aligned}
\Psi_3(\xi) &= \int \int_{\mathcal{X}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \pi(f(\mathbf{x}) | f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \pi(f(\xi) | \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \pi(f(\mathbf{x}) | f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \pi(f(\xi) | \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&+ \sum_{i=0}^{m-2} \left(\int_{\mathcal{X}^i / \mathcal{X}^{i+1}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \pi(f(\mathbf{x}) | \langle \mathrm{F} \rangle_{[i+1]}) df(\mathbf{x}) d\mathbf{x} \right) \\
&= \int \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \pi(f(\mathbf{x}) | f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \pi(f(\xi) | \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&+ \text{constant}, \qquad (5.15)
\end{aligned}$$

where input space notation $\mathcal{X}^i / \mathcal{X}^{i+1}$ corresponds to the NROY space at step i, which has been ruled out at step $i+1$ starting with the original input space $\mathcal{X}^0 = \mathcal{X}$. We treat the second term of equation (5.15) as a constant since it doesn't depend on ξ and $f(\xi)$. We can draw a connection between the decomposition of $\Psi_3(\xi)$ and the iterative refocussing process. At wave i we obtain a design, $\mathrm{X}_{[i]} \in \mathcal{X}^{i-1}$, and a collection of computer model runs, $\mathrm{F}_{[i]}$, to update a GP emulator for $f(\mathbf{x})$ generated at the previous waves of history matching using an ensemble, $\big\{\langle \mathrm{X} \rangle_{[i-1]}, \langle \mathrm{F} \rangle_{[i-1]}\big\}$. As a result, an updated GP emulator for $f(\mathbf{x})$ is used to obtain an NROY space $\mathcal{X}^i$.

We further reduce a number of integrals (see Appendix A.2) to obtain the following expression for $\Psi_3(\xi)$

$$\Psi_3(\xi) = \int\int_{\mathcal{X}^{m-1}} \left[\Phi\left(\frac{z + a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \right) \right.$$
$$\left. - \Phi\left(\frac{z - a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \right) \right]$$
$$\times \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) d\mathbf{x} df(\xi). \tag{5.16}$$

We consider in detail the effect of expectation, $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, on the integrand function conditioned on variance term, $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, with a short demonstration in Figure 5.1. Each panel plot in Figure 5.1 depicts the standard normal cumulative distribution function (CDF) against values of a random variable between -4 and 4. The blue dashed lines correspond to the CDF values obtained at two random variable realizations s_1 and s_2, with $s_2 > s_1$. The blue solid line represents the difference in CDF computed at s_2 and s_1, i.e. $\Phi(s_2) - \Phi(s_1)$. From Figure 5.1 we deduct that in case where $s_2 > 0$ and $s_1 < 0$ we obtain a higher value of $\Psi(s_2) - \Psi(s_1)$ (central panel plot) than in two other cases, i.e. $s_2, s_1 < 0$ (left panel plot) and $s_2, s_1 > 0$ (right panel plot).

For simplification, we define the arguments of two CDF functions in equation (5.16) as

$$s_1 = \frac{z + a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}},$$
$$s_2 = \frac{z - a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}}.$$

For $\mathbf{x} \in \mathcal{X}^{m-1}$, if $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$ lies inside the interval

$$\left[z - a\sqrt{Var[e] + Var[\eta]}, z + a\sqrt{Var[e] + Var[\eta]} \right], \tag{5.17}$$

we deduce that $s_2 > 0$ and $s_1 < 0$ and we operate in a steep region of the standard normal cumulative distribution function, demonstrated in the central panel

plot in Figure 5.1, leading to the larger value of an integrand function and therefore greater contribution towards $\Psi_3(\xi)$. On the contrary, if the expectation term, $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, lies outside the interval given in equation (5.17), for instance $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})] < z - a\sqrt{Var[e] + Var[\eta]}$, which corresponds to both s_1 and s_2 being positive, we observe lower values of $\Phi(s_2) - \Phi(s_1)$, demonstrated in the right panel plot in Figure 5.1. Similar observations are drawn for $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})] > z + a\sqrt{Var[e] + Var[\eta]}$, which corresponds to both s_1 and s_2 being negative, confirmed by the left panel plot in Figure 5.1.

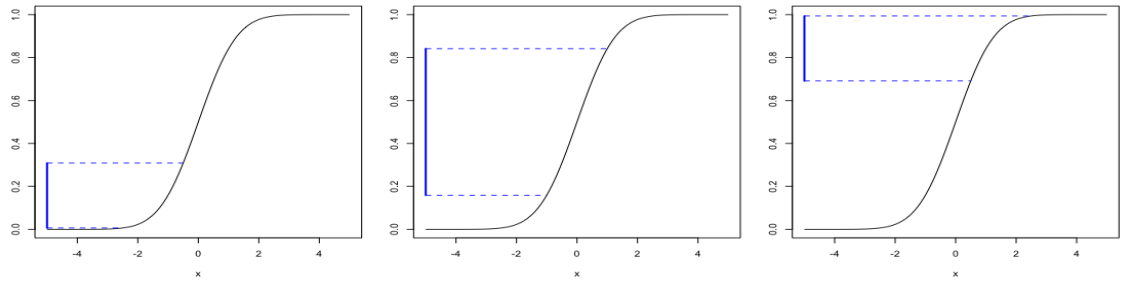

Figure 5.1: The plots of standard normal cumulative distribution function (CDF) against the varying values of a random variable x. The blue dashed lines correspond to CDF values computed at two values of a random variable x. The blue solid lines correspond to the differences between the CDF values at these two values of a random variable. *Left panel*: demonstrates the difference between the CDF evaluated at two negative values of a random variable. *Central panel*: demonstrates the difference between the CDF evaluated at positive and negative values of a random variable. *Right panel*: demonstrates the difference between the CDF evaluated at two positive values of a random variable.

We are interested to demonstrate the effect of the variance term, $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, conditioned on the expectation term, $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, being inside the interval in equation (5.17), on the integrand function of $\Psi_3(\xi)$. If the expectation, $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, lies inside the interval, then we observe $s_2 > 0$ and $s_1 < 0$. Figure 5.2 demonstrates the plots of standard normal cumulative distribution function (CDF) against a range of values of a random variable. Similarly to Figure 5.1, the blue dashed lines correspond to the CDF values obtained at s_2 and s_1, while the blue solid line represents the difference in CDF computed at s_2 and s_1, i.e. $\Phi(s_2) - \Phi(s_1)$. If we experience high uncertainty, expressed via $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, about the model behaviour at an arbitrary input point $\mathbf{x} \in \mathcal{X}^{m-1}$, this will lead to the downscaling of the CDF arguments s_2 and s_1, and as a result to a lower value

of $\Phi(s_2) - \Phi(s_1)$, demonstrated in the right panel plot of Figure 5.2. Therefore, we would expect to observe lower contribution from this input point in the calculation of $\Psi_3(\xi)$. On the contrary, if we observe the lower value of variance term $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$ at an arbitrary input point $\mathbf{x} \in \mathcal{X}^{m-1}$, we will obtain a larger difference between the CDF computed at s_2 and s_1, and therefore expect a larger contribution from this input point in the calculation of $\Psi_3(\xi)$. This argument is confirmed by the left panel plot in Figure 5.2.

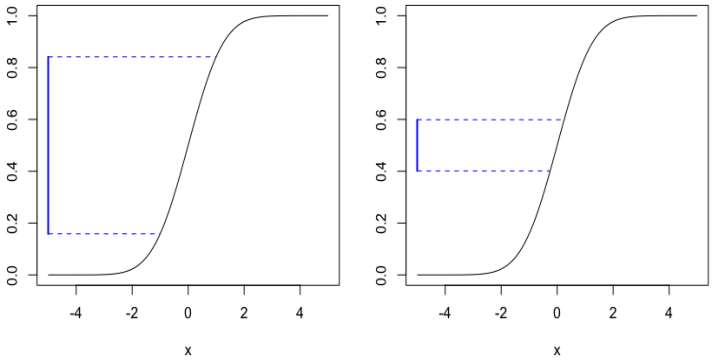

Figure 5.2: The plots of standard normal cumulative distribution function (CDF) against the varying values of a random variable x. The blue dashed lines correspond to CDF values computed at two values of a random variable x. The blue solid lines correspond to the differences between the CDF values at these two values of a random variable. *Left panel*: demonstrates the difference in the CDF evaluated at $x = 1$ and $x = -1$ *Right panel*: demonstrates the difference in the CDF evaluated at $x = 0.5$ and $x = -0.5$.

Based on the analysis above, we conclude that the largest contribution from $\mathbf{x} \in \mathcal{X}^{m-1}$ to the value of $\Psi_3(\xi)$ is obtained from points with expectation, $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, inside the interval in equation (5.17) and low values of variance, $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$.

5.3.4 Derivation and interpretation of $\Psi_2(\xi)$

Finally, we proceed to consider in detail the second term of the expected loss function, $\Psi_2(\xi)$, that corresponds to the expected volume of the input region that is in both the NROY obtained at wave m and the "true" NROY. A simplification (see

Appendix A.3) leads to the following form for $\Psi_2(\xi)$,

$$\Psi_2(\xi) = \int\int_{\mathcal{X}^{m-1}} \Bigg[\Phi\Bigg(\frac{z + a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \Bigg) \\ - \Phi\Bigg(\frac{z - a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \Bigg) \Bigg] \\ \times \mathbb{1}\Bigg\{ \frac{|z - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \Bigg\} \\ \times \pi(f(\xi) | \langle \mathrm{F} \rangle_{[m-1]}) d\mathbf{x} df(\xi). \tag{5.18}$$

The indicator function inside $\Psi_2(\xi)$ corresponds to the membership function from equation (5.13) that accounts for the points in the NROY space $\mathcal{X}^{m-1}$ that are expected to constitute the NROY space $\mathcal{X}^m$ after performing wave m of history matching with a candidate design, ξ, and the corresponding runs of the computer model, $f(\xi)$. As part of $\Psi_2(\xi)$, this membership function is now weighted by the difference of two standard normal CDFs from equation (5.16).

The membership function from equation (5.13) contains the implausibility function described in subsection 2.6.1. Low values of implausibility function can be obtained at an arbitrary input point $\mathbf{x} \in \mathcal{X}^{m-1}$ with a large value of $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, which will lead to the membership function being equal to 1. However, in subsection 5.3.3 we demonstrated that large values of $Var_{\{\mathrm{F}_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$ leads to the lower values of a difference between two CDFs. Therefore, $\mathbf{x} \in \mathcal{X}^{m-1}$ at which we obtain a high predictive variance, $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, will have a small contribution towards the final value of $\Psi_2(\xi)$. By contrast, we expect to observe a larger contribution from input points $\mathbf{x} \in \mathcal{X}^{m-1}$ with predictions, $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, that are close to observation z within the pre-specified error structure, i.e. $Var[e]$ and $Var[\eta]$, and low uncertainty on the predictions, $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$.

5.4 Implementation details

In this section, we discuss a number of computational details that are necessary for the computation of the BDC. We describe the process of performing m waves of

history matching.

1. Start with $m = 1$ and generate $\mathrm{X}_{[1]}$, employing one of the space-filling approaches such as uniform design, Sobol sequences (Fang et al., 2005; Challenor, 2011) or Latin Hypercubes (LHCs) (McKay et al., 1979) and obtain computer model runs $\mathrm{F}_{[1]}$. We start by defining a prior distribution on $f(\mathbf{x})$, denoted by $\pi\big(f(\mathbf{x})\big)$, which is a Gaussian process distribution with mean function

$$E[f(\mathbf{x})] = h(\mathbf{x})^T \boldsymbol{\beta}$$

and covariance function

$$Cov[f(\mathbf{x}), f(\mathbf{x}')] = k(\mathbf{x}, \mathbf{x}'; \sigma^2, \boldsymbol{\delta}, \tau^2).$$

We proceed to fit a GP emulator using ensemble, $\big\{\mathrm{X}_{[1]}, \mathrm{F}_{[1]}\big\}$, and obtain the maximum a posteriori (MAP) values of the GP's hyperparameters, in particular $\Theta_{MAP} = \big\{\boldsymbol{\beta}_{MAP}, \sigma^2_{MAP}, \boldsymbol{\delta}_{MAP}, \tau^2_{MAP}\big\}$. We are planning to fix the GP hyperparameters at MAP values throughout the whole Bayesian updating process to reduce the amount of computational effort. We update our prior distribution of $f(\mathbf{x})$ to its posterior, i.e. $\pi_1\big(f(\mathbf{x})\big) = \pi\big(f(\mathbf{x})|\mathrm{F}_{[1]}\big)$, which is a GP with expectation, $E_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$, and covariance,$Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$.

We use $E_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ and $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ to obtain the NROY space at wave 1, $\mathcal{X}^1$, which is defined as

$$\mathcal{X}^1 = \left\{\mathbf{x} \in \mathcal{X} : \frac{|z - E_{\mathrm{F}_{[1]}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]}} \leq a\right\}.$$

2. At wave $m > 1$, we start with a prior distribution $\pi_{m-1}\big(f(\mathbf{x})\big)$ for $f(\mathbf{x})$ as

$$f(\mathbf{x})|\Theta_{MAP} \sim GP\big(E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})], Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]\big)$$

and proceed to obtain a BOD by minimizing $\Psi(\xi)$ using an optimization algorithm, where the size of ξ is pre-specified. We denote the derived BOD as

$\mathrm{X}_{[m]} = \xi$.

3. We generate computer model runs, $\mathrm{F}_{[m]}$, at the design, $\mathrm{X}_{[m]}$, and update the probability distribution for $f(\mathbf{x})$ to derive $\pi_m(f(\mathbf{x}))$, which is given as

$$f(\mathbf{x})|\{\mathrm{X}_{[m]}, \mathrm{F}_{[m]}\}, \Theta_{MAP} \sim GP\Big(E_{\{\langle \mathrm{F} \rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})], Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]\Big)$$

4. We employ $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]$ and $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]$ to obtain the NROY space at wave m, $\mathcal{X}^m$, which is defined as

$$\mathcal{X}^m = \left\{\mathbf{x} \in \mathcal{X}^{m-1} : \frac{|z - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]}} \leq a\right\}.$$

Repeat Steps 2-4 until, for instance, the experimental budget has been exhausted or no change in the NROY space size is observed.

To use the optimization algorithm, we are required to evaluate an objective function, $\Psi(\xi)$, that contains two intractable integrals. We employ Monte Carlo integration to derive the numerical approximation to the integrals in $\Psi(\xi)$. In particular, we simulate a sample of $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}$ from the NROY space $\mathcal{X}^{m-1}$ and $f(\xi)^{(1)}, \ldots, f(\xi)^{(N)}$ from the distribution $MVN\big(E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)], Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)]\big)$. Then we approximate $\Psi(\xi)$ as

$$\begin{aligned}\hat{\Psi}(\xi) &= \frac{1}{N}\sum_{i=1}^{N} \Psi\big(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}\big) \\ &= \frac{1}{N}\sum_{i=1}^{N}\Big[\Psi_1(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}) - 2 \times \Psi_2(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}) + \Psi_3(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)})\Big],\end{aligned}$$

where

$$\Psi_1(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}) = \mathbb{1}\Big\{\frac{|z - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}} \leq a\Big\}, \tag{5.19}$$

$$\Psi_2(\xi;\mathbf{x}^{(i)},f(\xi)^{(i)}) = \Bigg[\Phi\Bigg(\frac{z+a\sqrt{Var[e]+Var[\eta]}-E_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}{\sqrt{Var_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}}\Bigg) - \Phi\Bigg(\frac{z-a\sqrt{Var[e]+Var[\eta]}-E_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}{\sqrt{Var_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}}\Bigg)\Bigg] \times \mathbb{1}\Big\{\frac{|z-E_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]|}{\sqrt{Var[e]+Var[\eta]+Var_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}} \leq a\Big\} \tag{5.20}$$

and

$$\Psi_3(\xi;\mathbf{x}^{(i)},f(\xi)^{(i)}) = \Bigg[\Phi\Bigg(\frac{z+a\sqrt{Var[e]+Var[\eta]}-E_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}{\sqrt{Var_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}}\Bigg) - \Phi\Bigg(\frac{z-a\sqrt{Var[e]+Var[\eta]}-E_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}{\sqrt{Var_{\{\langle \mathrm{F}\rangle_{[m-1]},f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})]}}\Bigg)\Bigg]. \tag{5.21}$$

Since we still have two intractable integrals inside $\Psi_2(\xi)$ and $\Psi_3(\xi)$, we do not use the reduced form of the expression with one integral derived for $\Psi_1(\xi)$ presented in subsection 5.3.2, as it does not provide us with any computational savings.

To evaluate the accuracy of $\hat{\Psi}(\xi)$, we calculate the standard error of the obtained approximation. We compute the unbiased estimate of σ^2_Ψ, which is defined as

$$s^2_\Psi = \frac{1}{N-1}\sum_{i=1}^{N}\Big(\Psi(\xi,\mathbf{x}^{(i)},f(\xi)^{(i)}) - \hat{\Psi}(\xi)\Big)^2,$$

and use this unbiased estimate in standard error computation, i.e.

$$\frac{s_\Psi}{\sqrt{N}}.$$

5.4.1 Implementation details to obtain the initial design

To obtain the initial design, i.e. a design to perform wave 1 of history matching, we are required to perform a full Bayesian analysis, described in subsection 5.3.1. Similar to the computation of BOD for wave m of history matching, we employ

Monte Carlo integration to derive the numerical approximation to integrals in $\Psi(\xi)$ in equation (5.12). Before describing the process of obtaining BOD at the first iteration of refocusing, we redefine the approximation $\hat{\Psi}(\xi)$ as

$$\begin{aligned}\hat{\Psi}(\xi) &= \frac{1}{N_1}\sum_{i=1}^{N_1}\Psi\big(\xi;\mathbf{x},f(\xi)^{(i)},\Theta_i^{MAP}\big)\\ &= \frac{1}{N_1}\sum_{i=1}^{N_1}\Big[\Psi_1\big(\xi;\mathbf{x},f(\xi)^{(i)},\Theta_i^{MAP}\big)-2\Psi_2\big(\xi;\mathbf{x},f(\xi)^{(i)},\Theta_i^{MAP}\big)+\Psi_3\big(\xi;\mathbf{x},f(\xi)^{(i)},\Theta_i^{MAP}\big)\Big],\end{aligned} \tag{5.22}$$

where

$$\Psi_1\big(\xi;\mathbf{x},f(\xi)^{(i)},\Theta_i^{MAP}\big)=\frac{1}{N_2}\sum_{k=1}^{N_2}\mathbb{1}\left\{\frac{|z-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]|}{\sqrt{Var[e]+Var[\eta]+Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\leq a\right\},$$

$$\begin{aligned}\Psi_2\big(\xi;\mathbf{x},f(\xi)^{(i)},\Theta_i^{MAP}\big)=&\frac{1}{N_2}\sum_{k=1}^{N_2}\left[\Phi\left(\frac{z+a\sqrt{Var[e]+Var[\eta]}-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}{\sqrt{Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\right)\right.\\ &\left.-\Phi\left(\frac{z-a\sqrt{Var[e]+Var[\eta]}-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}{\sqrt{Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\right)\right]\\ &\times\mathbb{1}\left\{\frac{|z-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]|}{\sqrt{Var[e]+Var[\eta]+Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\leq a\right\},\end{aligned}$$

$$\begin{aligned}\Psi_3\big(\xi;\mathbf{x},f(\xi)^{(i)}\Theta_i^{MAP}\big)=&\frac{1}{N_2}\sum_{k=1}^{N_2}\left[\Phi\left(\frac{z+a\sqrt{Var[e]+Var[\eta]}-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}{\sqrt{Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\right)\right.\\ &\left.-\Phi\left(\frac{z-a\sqrt{Var[e]+Var[\eta]}-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}{\sqrt{Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\right)\right]\end{aligned}$$

We proceed to describe the process of computing BDC for a fixed design setting, denoted as ξ:

1. Start by specifying a prior distribution on $f(\mathbf{x})$, denoted as $\pi\big(f(\mathbf{x})|\Theta\big)$, and prior beliefs about GP hyperparameters, $\Theta=\{\sigma^2,\boldsymbol{\delta},\tau^2,\boldsymbol{\beta}\}$, via $\pi\big(\Theta\big)$.

2. Generate N_1 samples of $f(\xi)$, i.e. $\big\{f(\xi)^{(i)}\big\}_{i=1}^{N_1}$, from the marginal distribution

of the data $\pi\big(f(\xi)|\xi\big)$.

3. For each $f(\xi)^{(i)}, i = 1, \ldots, N_1$, obtain maximum a posteriori (MAP) values of the GP hyperparameters, Θ_i^{MAP}.

4. Generate a uniform/space-filling sample of size N_2 of input points across the input space $\mathcal{X}$, i.e. $\{\mathbf{x}^{(k)}\}_{k=1}^{N_2}$

5. Compute $\hat{\Psi}(\xi)$ in equation (5.22).

In step 3 of the described algorithm we employ `mogp_emulator`, a Python package for fitting GP emulators and developed by the Research Engineering Group at the Alan Turing Institute (ATI). This package allows users to obtain MAP estimates in a fast way by optimizing the posterior distribution function with proper priors. Similar to BDC computation for wave m of history matching, we are interested in computing the standard error of obtained approximation. In this case the unbiased estimate of σ_Ψ^2 is defined as

$$s_\Psi^2 = \frac{1}{N_1 - 1} \sum_{i=1}^{N_1} \Big(\Psi\big(\xi; \mathbf{x}, f(\xi)^{(i)}, \Theta_i^{MAP}\big) - \hat{\Psi}(\xi) \Big)^2.$$

and we use this unbiased estimate in standard error computation, i.e.

$$\frac{s_\Psi}{\sqrt{N_1}}.$$

5.5 Simulation study

In our simulation study we consider a simple 2D toy function, a semi-sphere centred at 0 with radius 1. The toy function is defined as

$$f(\mathbf{x}) = f(x_1, x_2) = \sqrt{\big(1 - x_1^2 - x_2^2\big)}.$$

Figure 5.3 demonstrates the behaviour of the toy function response against x_1 and x_2. We assumed that we observe $z = 1.25$ with a measurement error variance of $Var[e] = 0.1^2$ and discrepancy error variance set at zero, i.e. $Var[\eta] = 0$. We also

specified the threshold value for history matching to be $a = 3$. Table 5.1 provides the values of the observation, together with the measurement error variance and the discrepancy error variance for the simulation study.

Range	z	$Var[e]$	$Var[\eta]$	Sample size	NROY size
[-2, 1]	1.25	0.1^2	0	10	5%

Table 5.1: 2D toy model information for history matching. Range denotes the spread of possible outputs for the function, and Not Ruled Out Yet (NROY) size denotes the theoretical size of NROY space, given this error structure, and assuming a "perfect" emulator.

To calculate the theoretical size of the NROY space in Table 5.1, we obtain function values onto a 150×150 grid over x_1 and x_2 and use these values in place of $E[f(\mathbf{x})]$. We also specify $Var[f(\mathbf{x})] = 0$ in our implausibility calculations with a "perfect" emulator.

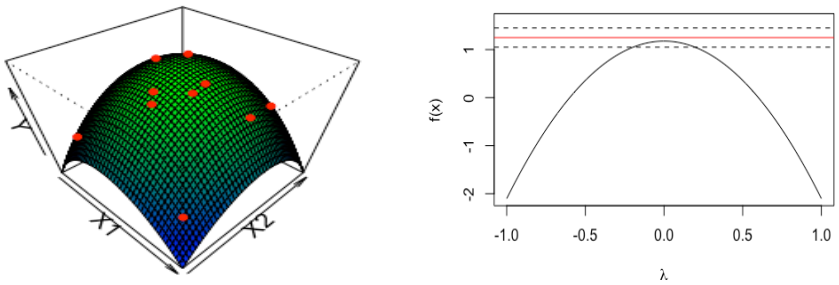

Figure 5.3: *Left*: True semi-sphere function for the two-dimensional numerical example and 10-run maximin distance LHD red points used as a design for wave 1 in subsection 5.5.2. *Right*: The cross-section plot of true semi-sphere function in black, which is represented by a line $\lambda\mathbf{x}_2 + (1-\lambda)\mathbf{x}_1$ against λ. The observed value is 1.25 (red line) with plus and minus two times observation error (dotted lines).

In order to produce the left panel plot in Figure 5.3, we generate values of the 2D toy function onto a 50×50 grid over x_1 and x_2 ($x_1 \in [-1, 1]$ and $x_2 \in [-1, 1]$). To produce the right panel plot in Figure 5.3, we generate a line, $\lambda\mathbf{x}_2 + (1-\lambda)\mathbf{x}_1$, between two design points $\mathbf{x}_1$ and $\mathbf{x}_2$ in two-dimensional space. The parameter λ describes how far along this line we are, and we observe how the function changes along this line as well as the actual function values in relation to the observation.

5.5.1 BOD for Wave 1 of history matching

In this subsection, we are interested in employing the proposed BDC to obtain a design for wave 1 of history matching, i.e. $\mathrm{X}_{[1]} = (\mathbf{x}_{11}, \ldots, \mathbf{x}_{1n_1})$ with $n_1 = 10$. We also consider the effect of the space-filling design, maximin LHC (Morris and Mitchell, 1995), on the history matching result.

To obtain a BOD for wave 1, we use the computational approach described in subsection 5.4.1. In particular, we use the optimization function genoud from R package rgenoud (Mebane and Sekhon, 2011). This function combines an evolutionary search algorithm and a genetic algorithm with derivative-based (Newton or quasi-Newton) methods to solve difficult optimization problems. We fixed the number of Monte Carlo samples at $N_1 = 250$ and generated a uniform sample of size $N_2 = 10,000$ to represent the input space, $\mathcal{X}$. At each iteration, we define 30 individuals (population size) with a hard limit on the maximum number of iterations equal to 30. The described set-up allows us to perform the high-dimensional optimization in a computationally cheap way.

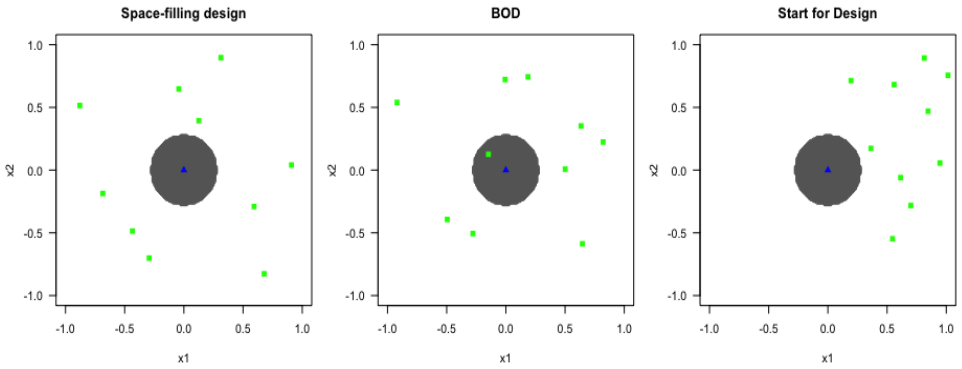

Figure 5.4: Each panel plot depicts the input space that demonstrates the "true" NROY space (dark grey) together with the candidate designs to perform wave 1 of history matching (bright green). The input point that correspond to the observation z is represented by a blue triangle.

Figure 5.4 demonstrates the allocation of three design candidates (bright green squares) across the input space in relation to the "true" NROY (dark grey points). Start for Design and BOD correspond to the starting points for the optimization algorithm and the optimal design choice respectively. We observe from the right panel plot in Figure 5.4 that the starting points are located in the region with

positive values of x_1, and the position of points in BOD is different to the starting design and is closer to Space-filling design representation.

Table 5.2 and Figure 5.5 show the BDC scores together with their MC errors for three candidate designs. We deduct that the lowest BDC scores are obtained for Space-filling design (0.0319) and BOD (0.0328); these two scores are very close to each other. Start for Design has the highest BDC score (0.117) as we expected, and we conclude that it is an unfavourable design choice to perform wave 1 of history matching.

Type	BDC	std. error	BDC-2×std.error	BDC+2×std.error
Space-filling design	0.0319	0.0026	0.0267	0.0370
BOD	0.0328	0.0031	0.0266	0.0390
Start for Design	0.117	0.0069	0.103	0.131

Table 5.2: Bayesian Design Criterion (BDC) computed at three design candidates. The second column corresponds to the BDC score, the third column is the Monte Carlo (MC) standard error on the BDC, the fourth and the fifth columns correspond to the BDC value plus and minus two standard errors.

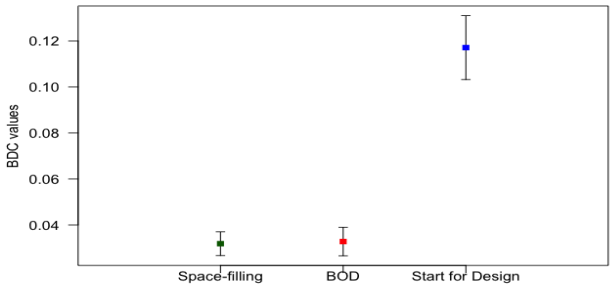

Figure 5.5: Plots of computed Bayesian Design Criterion (BDC), $\Psi(\xi)$, together with two Monte Carlo (MC) standard error bars for three candidate designs to perform wave 1 of history matching.

We proceed to consider the individual terms of BDC to understand the interpretation of scores better. Figure 5.6 shows the predictive variance computed over the whole input space, $\mathcal{X}$. To produce these plots we start by specifying a number of equally spaced intervals for x_1 and x_2, i.e. n_{res}, which determines the resolution, given by the number of pixels of obtained images. The value behind each pixel

represents the mean value of the predictive variance, i.e.

$$\frac{1}{N_1}\sum_{i=1}^{N_1}\frac{1}{N_{res}}\sum_{k=1}^{N_{res}} Var\big[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}\big], \quad x_1^l \leq x_1^{(k)} \leq x_1^{l+1}, l = 1,\ldots,n_{res}-1,$$
$$x_2^r \leq x_2^{(k)} \leq x_2^{r+1}, r = 1,\ldots,n_{res}-1,$$

where N_{res} corresponds to the number of points allocated to a pixel. Red and orange coloured regions correspond to the regions of input space with high predictive variance. On the contrary, regions in white correspond to low values of predictive variance. From Figure 5.6, we observe that both Space-filling design and BOD generate the lowest predictive variance values across the whole input space. These two design candidates are spaced out, allowing us to capture as much information as possible about the model response behaviour across $\mathcal{X}$. For Start for Design, we obtain the highest values of predictive variance in the region where the values of x_1 input are negative since we do not have any design points in this region.

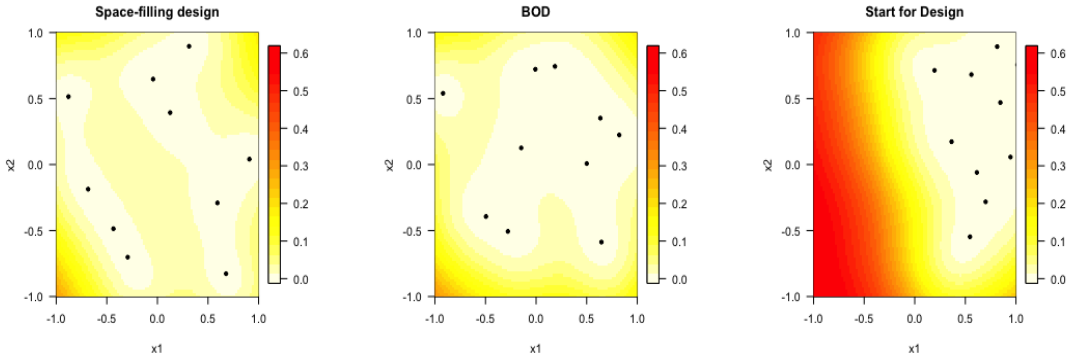

Figure 5.6: Predictive variance, $Var\big[f(\mathbf{x})|f(\xi)\big]$, computed over the input space, $\mathcal{X}$, as part of BDC computation for a range of design options. The colour corresponds to the mean value of the predictive variance behind each pixel. Different design options are depicted as black points.

We proceed to consider the effect of candidate designs on the Term 1, $\Psi_1(\xi)$, representation across the input space. Each panel plot demonstrates the proportion of points that are expected to be part of wave 1 NROY space behind each pixel and

computed as

$$\frac{1}{N_1}\sum_{i=1}^{N_1}\frac{1}{N_{res}}\sum_{k=1}^{N_{res}}\mathbb{1}\left\{\frac{|z-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]|}{\sqrt{Var[e]+Var[\eta]+Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\leq a\right\},$$

$$x_1^l \leq x_1^{(k)} \leq x_1^{l+1}, l = 1,\ldots,n_{res}-1,$$

$$x_2^r \leq x_2^{(k)} \leq x_2^{r+1}, r = 1,\ldots,n_{res}-1.$$

In Figure 5.7, red coloured regions correspond to the input regions with a high proportion of points expected to be part of wave 1 NROY space. The complete opposite is true for the yellow coloured regions. We observe the resembles of the plots in Figure 5.6 and Figure 5.7 and conclude that the predictive variance is important in the computation of $\Psi_1(\xi)$. In particular, we expect to rule out the input space close to the design points, since at $\mathbf{x}$ we obtain predictions far from the observation, z, together with low values of predictive variance. Therefore the implausibility function value computed at this arbitrary point is greater than the threshold value a. It is hard to draw meaningful conclusions about the effect of candidate designs on $\Psi_2(\xi)$ and $\Psi_3(\xi)$, and we provide comparative plots of these terms in the Appendix C.1.

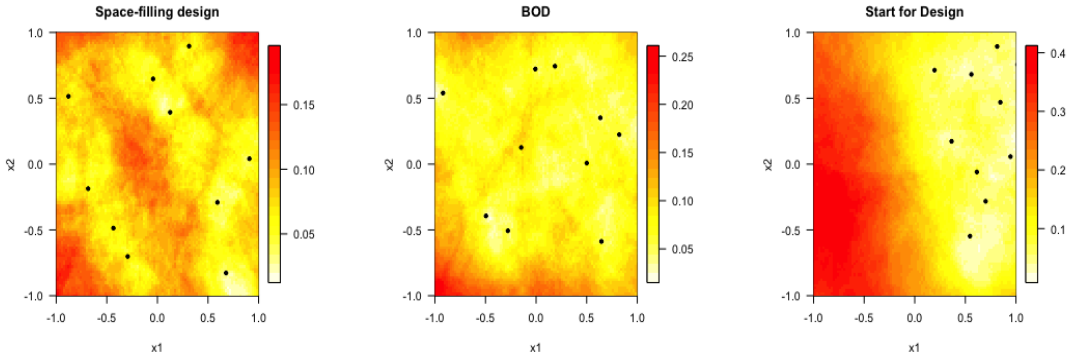

Figure 5.7: The integrand of Term 1, $\Psi_1(\xi)$, computed over the input space, $\mathcal{X}$. Each pixel of plots represents the mean value of the integrand of Term 1, $\Psi_1(\xi)$, computed at input settings behind each pixel. Different design options are depicted as black points.

We proceed to perform wave 1 history matching for three candidate designs. We start by obtaining the model runs, $\mathrm{F}_{[1]}$, at each design, $\mathrm{X}_{[1]}$, and construct GP emulators for $f(\mathbf{x})$. Figure 5.8 shows the results of wave 1 history matching for each

candidate design. The points in grey are retained as part of the NROY space, while points in green represent the "true" NROY space. Table 5.3 provides the summary results of history matching. The NROY space obtained with BOD is 6.43% of the original input space and is close in size to the "true" NROY space. However, this design choice has led to a small percentage of the "true" NROY space being ruled out. We deduce that Space-filling design is more conservative, with the size of the NROY being 9.60% of the original input space. Finally, by employing Start for Design, we generate an NROY space which is 45.02% of the original input space, which primarily contains the input region with negative values of x_1 due to the lack of design points in this region.

The history matching performance with Space-filling design and BOD are very similar, and we could conclude that running a highly complicated optimizer when we have no initial information about the function behaviour is not worthwhile. Instead, we suggest using one of the available space-filling initial designs mentioned in section 5.4 as an initial design.

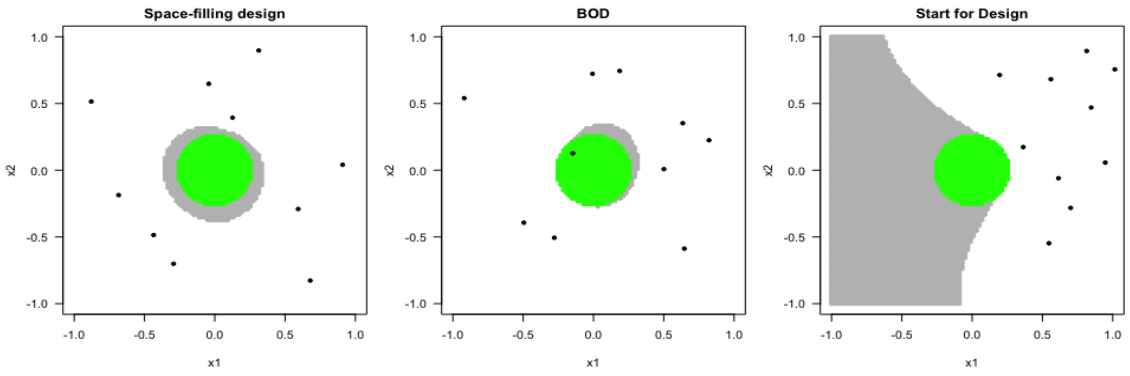

Figure 5.8: Each panel demonstrates a parameter plot showing points classified as being in NROY space after the first wave of history matching in grey when we use the design candidate (black dots) for constructing GP emulators. Points in green represents the input region identified as part of "true" NROY space.

Type	NROY size	Percentage of missing point
Space-filling design	9.60%	0%
BOD	6.43%	6.67%
Start for Design	45.02%	9.63%

Table 5.3: Summary of history matching results after wave 1 for candidate designs used to construct a GP emulator. Percentage of missing points corresponds to the percentage of points ruled out by performing wave 1 HM but that are part of "true" NROY space.

5.5.2 BOD for Wave 2 of history matching

In this subsection, we start by obtaining a 10-run maximin design to construct a GP emulator, $\pi_1\big(f(\mathbf{x})\big)$, and perform history matching. We compute the implausibility function over a $150x150$ grid in two-dimensional space to obtain a wave 1 NROY space which is 15.56% the size of the original input space, depicted in Figure 5.9.

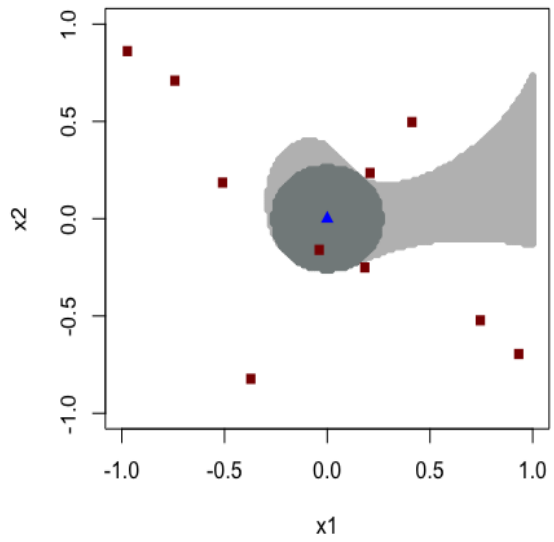

Figure 5.9: The input space plot demonstrates the "true" NROY space (dark grey) and those input points classified as part of NROY space after a single wave of HM (light grey) together with the design points used to construct a GP emulator for wave 1 of HM (brown squares)

To perform wave 2 of history matching, we need to obtain an ensemble, $\{\mathrm{X}_{[2]}, \mathrm{F}_{[2]}\}$, and update our GP emulator (see Step 3 in section 5.4). We demonstrate the application of employing a BDC in deriving a design for wave 2, $\mathrm{X}_{[2]} = (\mathbf{x}_{21}, \ldots, \mathbf{x}_{2n_2})$, with $n_2 = 5$. We also consider the performance of BOD against a design with good coverage over the current NROY space, $\mathcal{X}^1$, termed the "naive" design for history

matching.

The "naive" design is obtained by employing a simulated annealing algorithm. We consider the maximization of the minimum Euclidean distance between the design points, which is similar to a design approach proposed by Andrianakis et al. (2017) (see subsection 5.2.1 for more details). We consider all points that are not ruled out after performing wave 1 of history matching as candidate design points.

To obtain a BOD, we are interested in minimizing $\Psi(\xi)$ in equation (5.7). We obtain the values of $\Psi(\xi)$ using the computational method described in Section 5.4. We are required to specify N, the number of samples for Monte Carlo integration, and we have chosen $N = 2500$ based on empirical evidence, since this number provides us with a satisfactory Monte Carlo standard error and computational time.

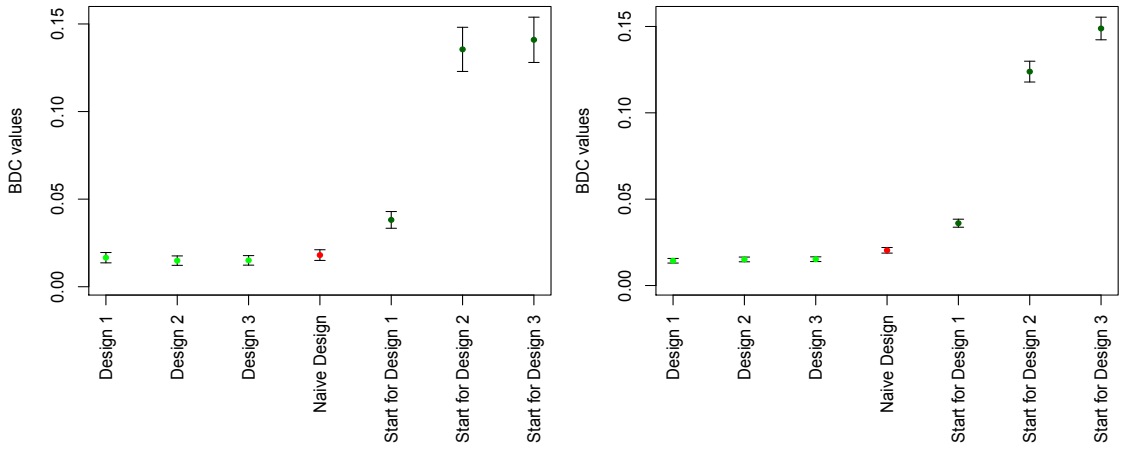

Figure 5.10: Plots of computed Bayesian Design Criterion (BDC), $\Psi(\xi)$, together with two Monte Carlo (MC) standard error bars for seven different candidate designs for wave 2 of history matching. *Left panel*: demonstrates BDC scores with $N = 2,500$ MC samples. *Right panel*: demonstrates BDC scores with $N = 10,000$ MC samples.

From Figure 5.10 we observe that the two standard error bars are smaller when we use $N = 10,000$ Monte Carlo samples for computing the BDC; however it leads to an increase in computational time for obtaining an optimal design. The ordering of designs according to the proposed BDC is the same for $N = 2,500$ and $N = 10,000$ and we conclude that $N = 2,500$ is a reasonable choice for the number of Monte Carlo samples.

Similarly to subsection 5.5.1, we use the optimization function genoud to obtain

the BOD. At each iteration, we define 100 individuals (population size) with a hard limit on the maximum number of iterations equal to 30. The genoud function allows users to specify the lower and upper bounds of the input space. However, in our case, it is challenging to derive upper and lower bounds of the input space since we are most unlikely to operate within a Euclidean space, i.e. the NROY space is defined by a membership rule based on implausibility function (see subsection 2.6.1). Therefore we propose to add a penalty term set to an arbitrarily large positive value inside the objective function so that if the optimization algorithm picks an individual (design candidate) in the ruled out space, we automatically encounter a penalty. We run the optimization algorithm in parallel from different starting points to investigate the effect of starting points on the final optimal design.

In Figure 5.11, each panel plot depicts the "true" NROY space and the points retained inside NROY space after wave 1 of history matching together with design $X_{[1]}$ (please see the description of Figure 5.9). The panel plots for Design 1, Design 2 and Design 3 demonstrate the allocation of starting points (dark green) used for the optimization of BDC to obtain the final designs (bright green). The panel plot for Naive Design in Figure 5.11 shows the generated "naive" space-filling design in bright green. We were interested in studying the effect of positioning the starting points on the final designs. In particular, we specified starting points for Design 1 clustered in the corner of $\mathcal{X}^1$ and away from the "true" NROY space. For Design 2, the starting points are spaced over the "true" NROY space, while for Design 3 the starting points are clustered around the observation z (blue triangle). From Figure 5.11, we conclude that the positioning of starting points specified for optimization algorithm across the reduced input space, $\mathcal{X}^1$, has a very limited effect on the final positioning of design points. In particular, we observe that a number of obtained design points are placed on the border of the "true" NROY space after employing the optimization algorithm.

We obtain BDC scores for the starting points as well their MC errors and include their values in Table 5.4, which provides a summary of BDC computations at different design candidates for wave 2 of HM.

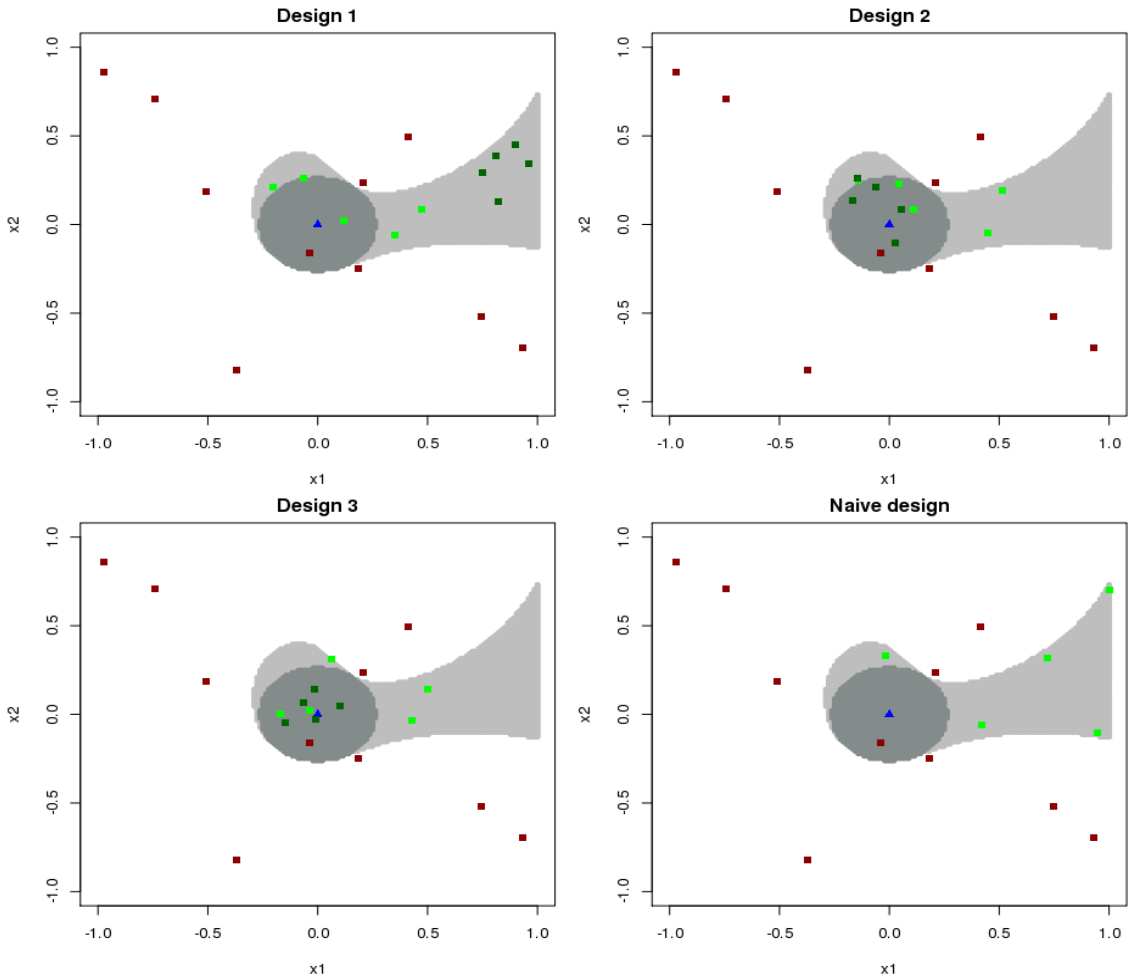

Figure 5.11: Each panel plot depicts the input space that demonstrates the "true" NROY space (dark grey) and those input points classified as part of NROY space after a single wave of HM (light grey) together with the design points used to construct a GP emulator for wave 1 of HM (brown squares). The panel plots labelled as Design 1, Design 2 and Design 3 demonstrate the positioning of the obtained designs (bright green) found by optimizing BDC with starting points specified for optimization (dark green). The panel plots labelled as Naive design demonstrates the positioning of "naive", space-filling design (bright green). The input point that correspond to the observation z is represented by a blue triangle.

Type	BDC	std. error	BDC-2×std.error	BDC+2×std.error
Design 1	0.0134	0.0013	0.0108	0.0159
Design 2	0.0138	0.0013	0.0113	0.0164
Design 3	0.0159	0.0014	0.0130	0.0188
Naive Design	0.0187	0.0015	0.0156	0.0218
Start for Design 1	0.0348	0.0023	0.0302	0.0394
Start for Design 2	0.1278	0.0061	0.1156	0.1401
Start for Design 3	0.1383	0.0063	0.1257	0.1509

Table 5.4: Bayesian Design Criterion (BDC) computed at seven design candidates. The second column corresponds to the BDC score, the third column is the Monte Carlo (MC) standard error on the BDC, the fourth and the fifth columns correspond to the BDC value plus and minus two standard errors.

From Table 5.4 we conclude that the lowest BDC scores are obtained for Design 1 (0.0134) and Design 2 (0.0138), which are both lower than the BDC score obtained for Naive Design (0.0187). However, we consider Naive Design as competitive given its Monte Carlo (MC) error. Interestingly, we observe that Start for Design 1 has the lowest BDC score (0.0348) amongst the other starting design options.

We proceed to consider in detail the composition of Bayesian Design Criterion for Design 2, that we refer to as BOD, Naive Design as well as Start for Design 1 and Start for Design 2.

Figure 5.12 and Figure 5.13 show the predictive variance computed over $\mathcal{X}^1$. The value behind each pixel corresponds to the mean value of the predictive variance, computed as

$$\frac{1}{N_{\text{res}}} \sum_{i=1}^{N_{\text{res}}} Var_{\{\text{F}_{[1]}, f(\xi)^{(i)}\}}[f(\mathbf{x}^{(i)})] \quad x_1^l \leq x_1^{(i)} \leq x_1^{l+1}, l = 1, \ldots, n_{\text{res}} - 1,$$

$$x_2^r \leq x_2^{(i)} \leq x_2^{r+1}, r = 1, \ldots, n_{\text{res}} - 1,$$

where N_{res} corresponds to the number of input points allocated to the pixel. Red and orange colour indicates the regions with high predictive uncertainty produced as part of BDC. On the contrary, the white colour corresponds to the region of low predictive variance produced as part of BDC. Grey regions correspond to the input regions that have been completely ruled out after performing wave 1 of history matching.

Figure 5.12 shows the predictive variance, $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, computed as part of BDC on a different scale to investigate in detail the effect of the proposed designs on the predictive variance over the $\mathcal{X}^1$. For instance, for Start for Design 1, we are most uncertain about the model response in the region close to the observation, since the points in this candidate design are clustered in the corner of $\mathcal{X}^1$ away from the observation. For Naive Design, we observe the largest values of predictive variance in the region around $x_1 = 1$ with the second highest values of predictive variance generated in the region close to the observation.

However, from Figure 5.13, we observe that these two design options produce the lowest predictive variance values over the $\mathcal{X}^1$ amongst a range of considered design options. This finding is anticipated since Start for Design 1 places all points in $\mathcal{X}^1$, where we do not have any points from $\mathrm{X}_{[1]}$, and Naive Design is considered as a space-filling design over $\mathcal{X}^1$ that aims to learn as much as possible about the model response across $\mathcal{X}^1$.

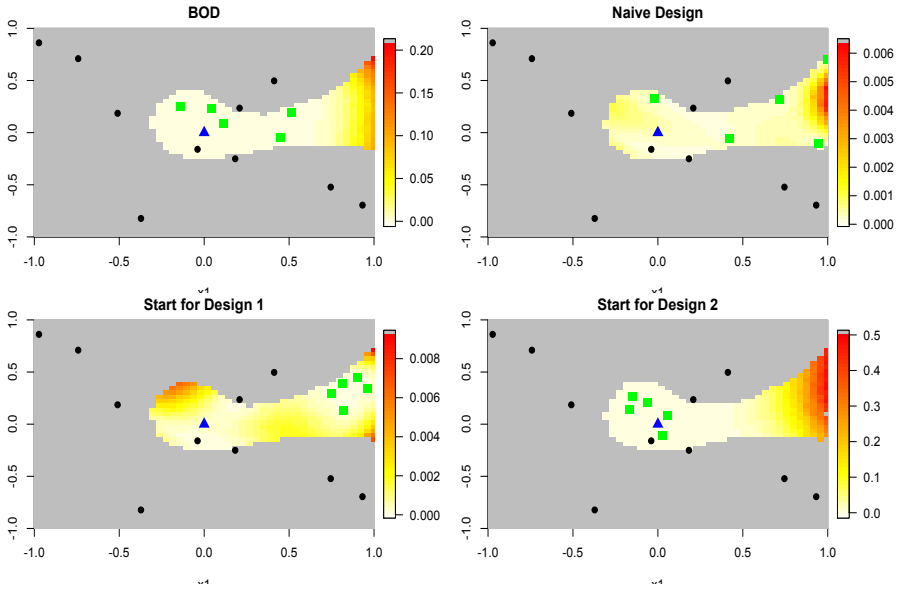

Figure 5.12: Predictive variance, $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, computed over wave 1 NROY space, $\mathcal{X}^1$, as part of BDC computation for a range of design options. The colour corresponds to the mean value of the predictive variance behind each pixel. Different design options are depicted as the green square points. Design for wave 1 are black points. The parameter setting for the observation z is the blue triangle. The ruled out input space is in grey.

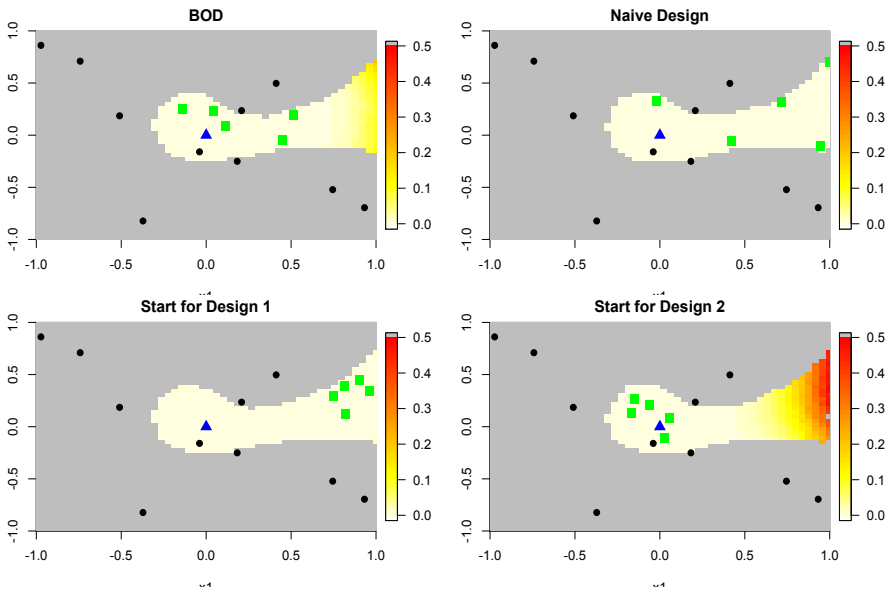

Figure 5.13: Predictive variance, $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, computed over wave 1 NROY space, $\mathcal{X}^1$, as part of BDC computation for a range of design options. The colour corresponds to the mean value of the predictive variance behind each pixel on the same scale. Different design options are depicted as the green square points. Design for wave 1 are black points. The parameter setting for the observation z is the blue triangle. The ruled out input space is in grey.

We observe the highest values of predictive variance in the region around $x_1 = 1$ for Start for Design 2 as the majority of design points are clustered around $x_1 = 0$ away from a high uncertainty region. The predictive variance obtained with BOD also exhibits high values in this region.

We are interested in considering the effect of different candidate designs on the final value of Term 1, $\Psi_1(\xi)$, given in equation (5.13). In particular, we are interested in decomposing the contributions to the final value of $\Psi_1(\xi)$ over the NROY space, for which we generate the plots shown in Figure 5.14. Each panel plot shows the proportions of the points that are expected to be part of wave 2 NROY space behind each pixel computed as

$$\frac{1}{N_{\text{res}}} \sum_{i=1}^{N_{\text{res}}} \Psi_1\big(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}\big) \quad x_1^l \leq x_1^{(i)} \leq x_1^{l+1}, l = 1, \ldots, n_{\text{res}} - 1$$

$$x_2^r \leq x_2^{(i)} \leq x_2^{r+1}, r = 1, \ldots, n_{\text{res}} - 1,$$

where $\Psi_1\big(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}\big)$ is given in equation (5.19). In this case, red areas of these plots correspond to the input regions with a higher proportion of points expected to

be retained as part of the NROY space after wave 2 as part of BDC computation, i.e. the largest contributions towards the value of $\Psi_1(\xi)$ are obtained at these input regions. On the contrary, white areas correspond to the input regions with a very low proportion of points expected to be retained as part of wave 2 NROY space, i.e. contributions towards $\Psi_1(\xi)$ are close to zero at these regions.

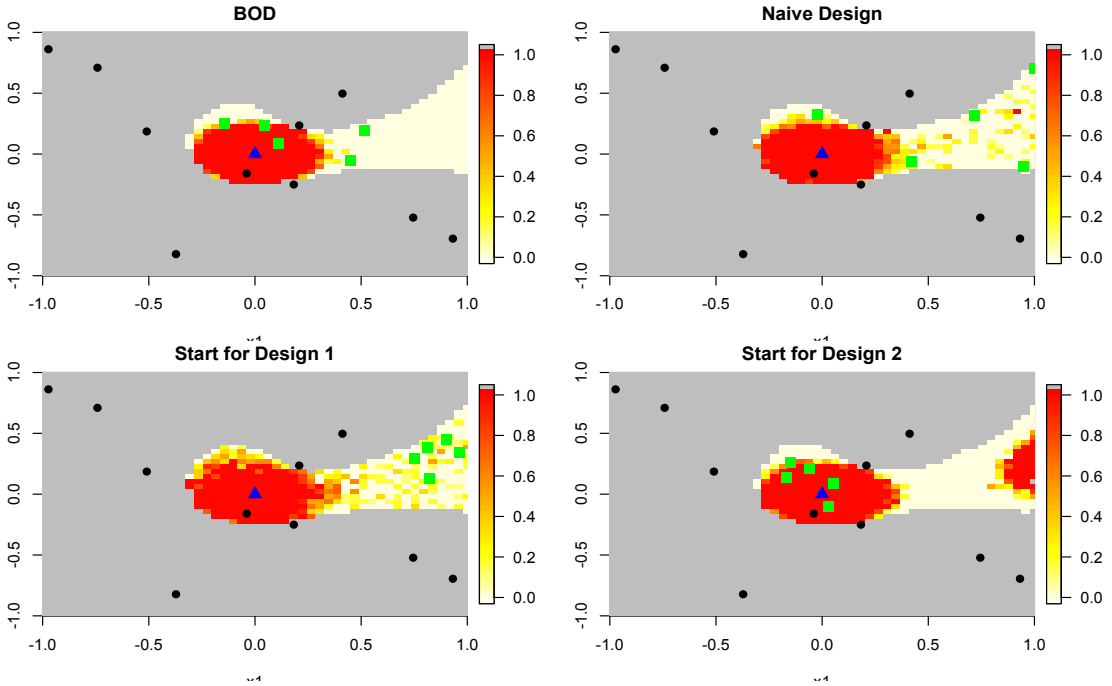

Figure 5.14: The integrand of Term 1, $\Psi_1(\xi)$, computed over wave 1 NROY space. Each pixel of plots represents the mean value of the integrand of Term 1, $\Psi_1(\xi)$, computed at input settings behind each pixel. For each panel plot we fix and consider the effect of different design options for wave 2 design, ξ, on the integrand value. Design candidates for wave 2 are green squares. Design for wave 1 are black points. The input parameter setting corresponding to the observation z is the blue triangle. The ruled out input space after wave 1 HM is in grey.

From Figure 5.14 we observe a large red region around $x_1 = 1$ for Start for Design 2, which corresponds to the region of high predictive variance depicted in Figure 5.13, i.e. a region of high uncertainty about the model response behaviour. We conclude that the updated GP emulator with this design is expected to be extremely uncertain about the model behaviour in this region. We observe orange shading on the borders of the "true" NROY space produced for Start for Design 1 as well as for Naive Design, which indicates that we are unsure if these regions are expected to be retained as part of the NROY space after wave 2, resulting in a higher value of $\Psi_1(\xi)$ in the overall BDC score. On the contrary, we observe a

distinct difference between the expected NROY space after wave 2 and the ruled out space for BOD, since the design points are mainly on the border of the "true" NROY space.

We proceed to analyse in detail Term 2 of BDC, $\Psi_2(\xi)$, given in equation (5.18). Similar to the analysis of Term 1 of BDC in Figure 5.14, we produce input space plots for each candidate design showing the contributions towards $\Psi_2(\xi)$ over the NROY space in Figure 5.15. The value behind each pixel is computed as

$$\frac{1}{N_{\text{res}}} \sum_{i=1}^{N_{\text{res}}} \Psi_2(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}) \quad x_1^l \leq x_1^{(i)} \leq x_1^{l+1}, l = 1, \ldots, n_{\text{res}} - 1$$
$$x_2^r \leq x_2^{(i)} \leq x_2^{r+1}, r = 1, \ldots, n_{\text{res}} - 1,$$

where $\Psi_2\big(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}\big)$ is given in equation (5.20). The value behind each pixel is the mean contribution towards the value of $\Psi_2(\xi)$ computed at the input points allocated to this pixel. Red coloured regions correspond to the input space where we have a high expectation that input points in this region are part of wave 2 NROY as well as the "true" NROY space. The complete opposite is true for the input points in white coloured regions of the input space.

Similarly to the plots in Figure 5.14, we observe orange shading on the borders of the "true" NROY space produced for Naive Design and Start for Design 1, which lead to the larger values of the integrand of $\Psi_2(\xi)$. This indicates that we are unsure whether this input region is contained in both the NROY space after wave 2 HM and the "true" NROY space as part of BDC computation. Interestingly, for both BOD and Start for Design 2 we observe a clearer representation of the overlap region, which could be due to the fact that a number of points in these two candidate designs are located close to the observation z, and on borders of the "true" NROY space.

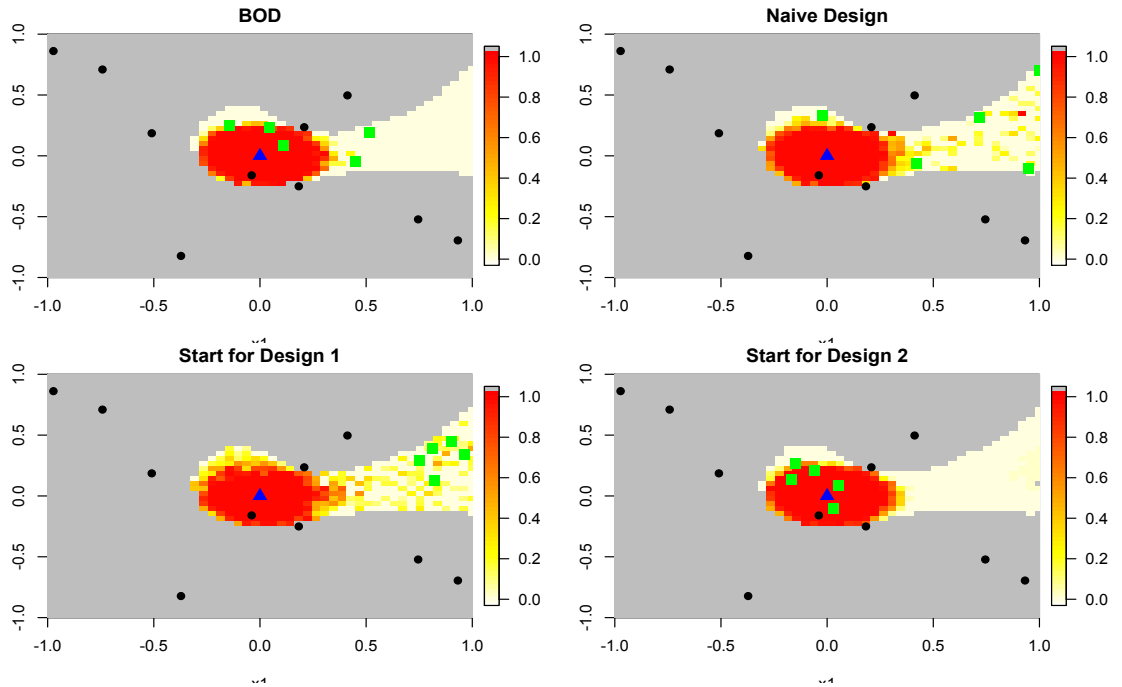

Figure 5.15: The integrand of Term 2, $\Psi_2(\xi)$, computed over wave 1 NROY space. Each pixel of plots represents the mean value of the integrand of Term 2, $\Psi_2(\xi)$ computed at input settings behind each pixel. For each panel plot we fix and consider the effect of different design options for wave 2 design, ξ, on the integrand value across the NROY space $\mathcal{X}^1$. Design options for wave 2 are depicted as green squares. Design for wave 1 are black points. The input parameter setting corresponding to the observation z is the blue triangle. The ruled out input space after wave 1 HM is in grey.

Finally, we proceed to analyse Term 3 of BDC, $\Psi_3(\xi)$, given in equation (5.16). We generate input space plots for each candidate design in Figure 5.16, similarly to those produced previously for predictive variance, Term 1 and Term 2. Each panel plot in Figure 5.16 gives the contributions towards $\Psi_3(\xi)$ over the NROY space for a candidate design. The value behind each pixel is computed as

$$\frac{1}{N_{\text{res}}}\sum_{i=1}^{N_{\text{res}}}\Psi_3(\xi;\mathbf{x}^{(i)},f(\xi)^{(i)}) \quad x_1^l \leq x_1^{(i)} \leq x_1^{l+1}, l=1,\ldots,n_{\text{res}}-1$$
$$x_2^r \leq x_2^{(i)} \leq x_2^{r+1}, r=1,\ldots,n_{\text{res}}-1,$$

where $\Psi_3\big(\xi;\mathbf{x}^{(i)},f(\xi)^{(i)}\big)$ is given in equation (5.21). The value behind each pixel represents the mean contribution towards the value of $\Psi_3(\xi)$ obtained at the input points allocated to this pixel.

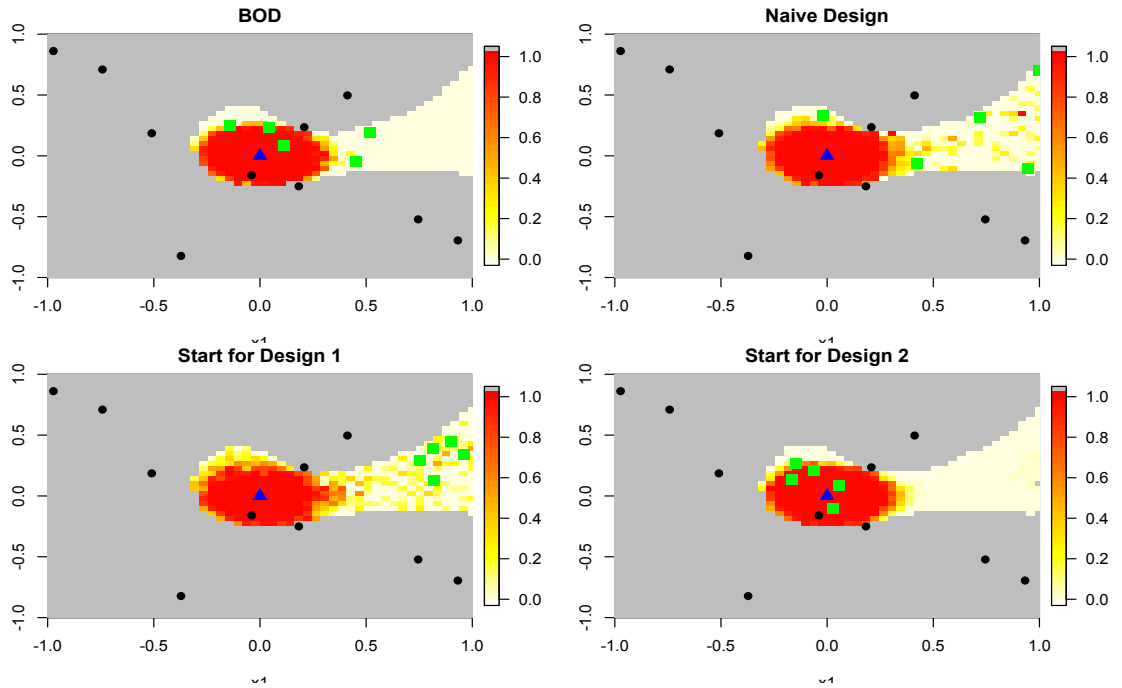

Figure 5.16: The integrand of Term 3, $\Psi_3(\xi)$, computed over wave 1 NROY space. Each pixel of plots represents the mean value of the integrand of Term 3, $\Psi_3(\xi)$, computed at input settings behind each pixel. For each panel plot we fix and consider the effect of different design options for wave 2 design, ξ, on the integrand value. Design options for wave 2 are depicted as green squares. Design for wave 1 are black points. The input parameter setting corresponding to the observation z is the blue triangle. The ruled out input space after wave 1 HM is in grey.

In Figure 5.16, red coloured regions correspond to the input regions where we have a higher expectation that input points in these regions are part of the "true" NROY space. The complete opposite conclusion is drawn about the white coloured input regions. We observe that the input space plots in Figure 5.16 resemble the input space plots in Figure 5.15, which could be explained by the fact that the difference of two standard normal CDFs (see subsection 5.3.3 for more details) is inside the integrand functions of $\Psi_2(\xi)$ and $\Psi_3(\xi)$.

We are interested in comparing wave 2 HM results for all four design candidates. We start by producing the model runs, $\mathrm{F}_{[2]}$, at each design, $\mathrm{X}_{[2]}$ and proceed to updating the probability distribution for $f(\mathbf{x})$ to derive $\pi\big(f(\mathbf{x})\big)$.

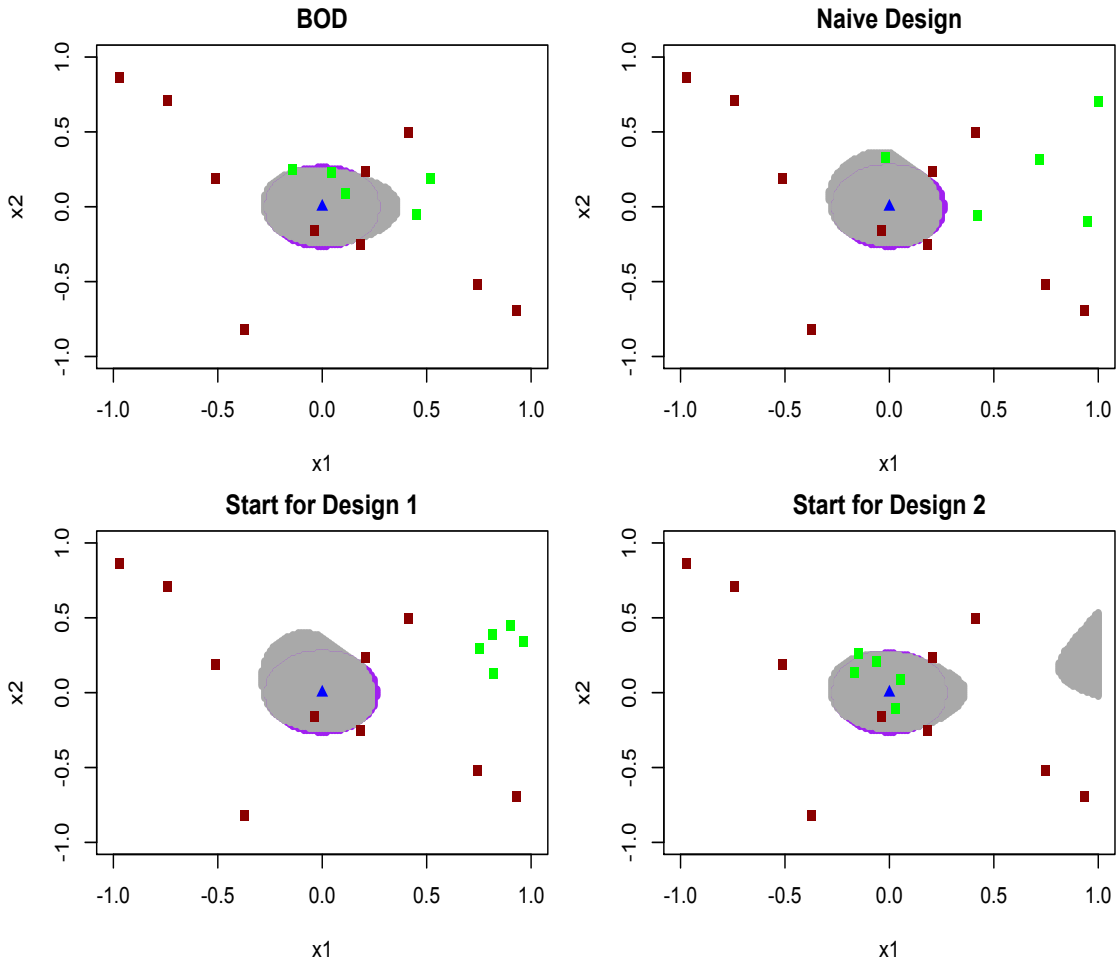

Figure 5.17: Each panel demonstrates a parameter plot showing points classified as being in NROY space after two waves of history matching in grey when we use the design in green for constructing Gaussian process emulators. Points in purple represents the portion of "true" NROY space that has been ruled out. The design used for wave 1 of HM are brown squares.

Figure 5.17 gives the results of wave 2 HM for each design candidate. In each input space plot in Figure 5.17, input points retained after a second iteration of history matching are shown in grey and the input points ruled out after two waves of HM, but that are part of the "true" NROY space, are in purple. We also add the candidate designs used to perform wave 2 HM represented by green squares together with the design $X_{[1]}$, which are represented by brown squares. Table 5.5 provides the summary results of history matching for each design candidate. Based on Figure 5.17 and Table 5.5, we derive that we obtain an NROY space for BOD and Naive Design which are of similar sizes, i.e. 6.10% and 6.15% of the original input space respectively. Both design options lead to the ruling out of a small percentage

of input space that is considered as part of the "true" NROY space. Interestingly, Start for Design 1 performs comparatively well, generating an NROY space which is 6.51% of the original input space. This finding could be explained by the fact that the design generated for wave 1 is sufficient to learn about the model response behaviour close to the observation, while Start for Design 1 is used to learn about the model response in high uncertainty regions. Start for Design 2 generates the NROY space of size 7.92%, and the shape of the wave 2 NROY space is very similar to the Term 1 decomposition over $\mathcal{X}^1$ depicted in Figure 5.14.

Type	NROY size	Percentage of missing point
BOD	6.10%	4.44%
Naive Design	6.15%	5.67%
Start for Design 1	6.51%	5.10%
Start for Design 2	7.92%	3.29%

Table 5.5: Summary of history matching results after wave 2 for candidate designs used to update Gaussian Process emulator to perform wave 2 HM. Percentage of missing points corresponds to the percentage of points ruled out by performing wave 2 HM but that are part of "true" NROY space.

Based on this simple simulation study we could conclude that our proposed Bayesian Design Criterion encourages the sampling of design points close to the observation z within the pre-specified error structure in order to improve the predictions, $E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, generated by an updated emulator for $f(\mathbf{x})$. Also, our proposed design criterion encourages exploration, since the predictive variance term, $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, plays an important part in the computation of each individual term of BDC. In particular, we observe the driving effect of high values of the predictive variance term on the Term 1 for Start for Design 1, which led to a higher BDC score for this design candidate. Our proposed criterion indicates that this particular design is an unfavourable choice to perform wave 2 of history matching among the discussed design candidates. On the contrary, BOD contained a number of points close to the observation z. As well as this, we observed a number of points placed further away from z and the design points in $\mathrm{X}_{[1]}$, which reduces our uncertainty about the model behaviour in the previously unexplored parameter space.

5.6 Conclusion

In this Chapter, we have presented a new method for generating an ensemble to perform the next iteration of history matching by employing a Bayesian experimental design, that has not been previously used for iterative refocussing. We specify a squared difference between the volume of the NROY space produced at the next wave and the volume of the "true" NROY space as our loss function that corresponds to our inferential aim to obtain an NROY space that contains all the points in the input space that are close to the observations. The optimal design is found by minimizing the expected loss. Our proposed design is easily decomposed into three individual and interpretable terms.

A full Bayesian approach has been implemented by integrating $\Psi(\xi)$ over the posterior distribution of GP hyperparameters to obtain an optimal design for wave 1 of history matching in subsection 5.5.1. In the process of obtaining BOD for wave 2 of history matching in subsection 5.5.2, we fixed the GP hyperparameters at their MAP estimates, turning our proposed criterion into pseudo-Bayesian. We recognise the importance of the full Bayesian design since it allows us to take into account the effect of candidate design on the GP hyperparameter values, as well as the uncertainty about these values, and we plan to adopt this approach for wave $m > 1$ design.

In our simulation study, we have chosen a 2D toy model to provide a simple and easy way to understand graphical representations of all of the major components of the proposed BDC. We employed the BDC to obtain optimal designs for two waves of history matching. We conclude that employing a computationally expensive BDC to generate the initial design is not worthwhile, since a space-filling design works well and is easy to obtain. On the contrary, employing BDC for wave $m > 1$ of history matching could be useful. In particular, by analysing the terms of BDC for wave 2 design individually, we conclude that our design criterion encourages both exploration, i.e. reduction of the predictive uncertainty, as well as exploitation, i.e. improving our knowledge about the model behaviour close to the observation.

In Chapter 6, we are interested in applying our nonstationary GP emulator via

kernel mixture from Chapter 4 to a proposed BDC. This type of design could be crucial when we are performing iterative refocussing by considering nonstationary simulator response. Contrary to the "naive" design approach, our proposed BDC contains the covariance structure from the GP emulator for $f(\mathbf{x})$, which will be informative during the iterative refocussing with a nonstationary computer model response. We have observed nonstationary model responses many times in our joint work with climate modellers, and we are interested in investigating the performance of our proposed BDC in these cases.

Chapter 6

Bayesian Optimal Design and the Nonstationary GP model

6.1 Introduction

We introduced our Bayesian Design Criterion (BDC) for multi-wave history matching in Chapter 5. In this Chapter, we are interested in providing an extension to our BDC by finding a Bayesian Optimal Design (BOD) for history matching when operating with a nonstationary model response.

In general, employing stationary design strategies such as uniform and space-filling designs are believed to be ill-suited when we are dealing with a nonstationary response, since we are interested in exploring more complicated regions of the input space, for instance, a region of the input space with high variability in response (Gramacy and Lee, 2009).

The majority of the nonstationary GP models, described in section 2.5, are adapted to the sequential design of computer experiments. Sequential design of computer experiments proceeds as follows: a new design point, $\mathbf{x}_{n+1}$, is chosen by optimizing a pre-specified design criterion, J_n, with respect to a current design set $\mathrm{X}_n = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$; the GP emulator is refitted conditional on the new pair $(\mathbf{x}_{n+1}, f(\mathbf{x}_{n+1}))$. Mathematically, $\mathbf{x}_{n+1}$ is added to a current design set as a point

that optimizes a design criterion J_n (Marmin et al., 2018)

$$\mathbf{x}_{n+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\, J_n(\mathbf{x}), \quad \text{or}$$

$$\mathbf{x}_{n+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmax}}\, J_n(\mathbf{x}).$$

Similar to standard GP models, to obtain a sequential design by using a nonstationary GP model, we are required to make the following two decisions: what design criterion, $J_n(\cdot)$, to specify and how to update the GP model conditioned on a new pair $(\mathbf{x}_{n+1}, f(\mathbf{x}_{n+1}))$. In general, the follow-up runs are produced to improve the modellers' knowledge, i.e. reduce uncertainty about the model behaviour. Therefore the design criteria that are commonly used are variance-based.

The process of updating emulator after each subsequent design iteration is another crucial decision. For instance, the Bayesian updating is not appropriate for partition models such as TGP (Gramacy and Lee, 2008, 2009) and Voronoi-tessellation GP (Kim et al., 2005; Pope et al., 2018), since the number and location of partitions in the input space could change with the addition of follow-up runs. Sequential Bayesian updating requires that the form of the model for $f(\mathbf{x})$ is known in advance.

After reviewing the application of nonstationary GP models to sequential design in section 6.2, the main focus of this Chapter is the adaptation of our proposed nonstationary GP emulator with kernel mixtures, introduced in Chapter 4, to the BDC for iterative refocusing. In section 6.3, we investigate how the values of predictive variance, $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, change in response to the varying model behaviour across wave 1 NROY space, $\mathcal{X}^1$, and how it affects individual components as well as the overall score of the BDC. We proceed to compare the effect of BOD found by employing a nonstationary GP model to a "naive", space-filling design over the wave 1 NROY space, on the size and shape of the wave 2 NROY space. The water vapour at 500 metres generated by a model described in section 3.2 is considered in our application studies. In particular, we decided to perform two application studies. In the first study in section 6.5, we use the values of observation and measurement

error variance provided by climate modellers. For the second study in section 6.6, we use synthetic values to study the effect on BDC. In section 6.7, we finish off with a discussion and future developments to the BDC with our proposed nonstationary GP emulator.

6.2 Sequential and adaptive designs for nonstationary computer models

Nonstationary GP emulators are employed to provide a trustworthy assessment of uncertainty via predictive variance when operating with a nonstationary model response. Predictive variance is considered as a measure of uncertainty about the model behaviour and therefore, is widely used as part of the design criterion. In subsection 6.2.1, we start by discussing a range of design criteria, J_n, employed for sequential design with a nonstationary GP emulator in UQ literature. In subsection 6.2.2, we proceed to consider different approaches adopted to updating nonstationary GP models for $f(\mathbf{x})$ conditioned on the new pair $(\mathbf{x}_{n+1}, f(\mathbf{x}_{n+1}))$.

6.2.1 Design Criteria

We start by considering two design criteria. We define the mean square error criterion as

$$J_n^{MSE}(\mathbf{x}) = C_n^*(\mathbf{x}, \mathbf{x}) = Cov[f(\mathbf{x}, \mathbf{x})|\{\boldsymbol{X}_n, \boldsymbol{F}_n\}]. \tag{6.1}$$

A new design point, $\mathbf{x}_{n+1}$, is added to a current design set as a point that maximizes the design criterion, i.e.

$$\mathbf{x}_{n+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmax}}\, J_n^{MSE}(\mathbf{x}).$$

In a similar way we specify the integrated mean square error criterion, i.e.

$$J_n^{IMSE}(\mathbf{x}) = \int_{\boldsymbol{u}} C_{n,\mathbf{x}}^*(\boldsymbol{u}, \boldsymbol{u})d\boldsymbol{u} = \int_{\boldsymbol{u}} Cov[f(\boldsymbol{u}), f(\boldsymbol{u})|\{\mathrm{X}_n, \mathrm{F}_n\} \cup \{\mathbf{x}, m_n^*(\mathbf{x})\}]d\boldsymbol{u},$$

$$\mathbf{x}_{n+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\, J_n^{IMSE}(\mathbf{x}) \tag{6.2}$$

where $C_n^*(\cdot,\cdot)$ and $C_{n,\mathbf{x}}^*(\cdot,\cdot)$ take the form of equation (2.7), and posterior mean $m_n^*(\cdot)$ takes the form of equation (2.6) introduced in subsection 2.3.1. In particular, these two criteria focus on sampling around areas in $\mathcal{X}$ with the highest predictive variance (Marmin et al., 2018). In these areas, we are most uncertain about the behaviour of a simulator. These two design criteria are very similar to two active learning approaches using the GP model (Gramacy and Lee, 2009). ALM (active learning-MacKay) has exactly the same form as the mean square error criterion given in equation (6.1), i.e. it selects an input point $\mathbf{x}_{n+1}$ from a set of candidates that has the greatest predictive standard deviation (MacKay, 1992).

ALC (active learning-Cohn) is similar to the integrated mean square error criterion given in equation (6.2) by considering a reduction in the predictive variance over the input space $\mathcal{X}$ and defined as

$$
\begin{aligned}
J_n^{ALC}(\mathbf{x}) &= \int_{\boldsymbol{u}} \Big(C_n^*(\boldsymbol{u},\boldsymbol{u}) - C_{n,\mathbf{x}}^*(\boldsymbol{u},\boldsymbol{u}) \Big) d\boldsymbol{u} \\
&= \int_{\boldsymbol{u}} \Big(Cov[f(\boldsymbol{u}), f(\boldsymbol{u}) | \{\mathrm{X}_n, \mathrm{F}_n\}] - Cov[f(\boldsymbol{u}), f(\boldsymbol{u}) | \{\mathrm{X}_n, \mathrm{F}_n\} \cup \{\mathbf{x}, m_n^*(\mathbf{x})\}] \Big) d\boldsymbol{u},
\end{aligned}
$$

and the ALC approach selects $\mathbf{x}_{n+1}$ that maximize the expected reduction in squared error over the input space $\mathcal{X}$ (Cohn, 1996), i.e.

$$
\mathbf{x}_{n+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmax}}\, J_n^{ALC}(\mathbf{x}).
$$

ALC criterion is computationally more intensive than ALM since it contains an integral. However, ALM tends to select $\mathbf{x}_{n+1}$ along the boundary of the input space since we tend to observe the highest values of predictive variance in these input regions (Montagna and Tokdar, 2016). Despite this argument, Montagna and Tokdar (2016) employed ALM approach with their proposed nonstationary GP model to obtain follow-up runs for their computer experiments, while Gramacy and Lee (2009) considered both criteria with TGP.

Pope et al. (2018) propose an adaptive sampling to estimate and better understand the discontinuities between regions of Voronoi tessellation (for more details about the proposed nonstationary GP model see section 4.2). They start by ob-

taining a MAP model, tessellation that corresponds to the posterior sample of tessellation parameters with the largest likelihood value. The sampling locations are chosen iteratively on the boundary of the region of interest and furthest away from the design points, X_n. This adaptive sampling is straightforward to understand and perform in practice. However, more intuition is required to choose from which region to sample.

Marmin et al. (2018) demonstrate the performance of variance-based criteria, precisely MSE in equation (6.1) and IMSE in equation (6.2), with warped multiple index GP model (WaMI-GP), described in detail in subsection 2.5. However, they conclude that variance-based design criteria depend on the geometry, i.e. location of design points across the input space $\mathcal{X}$ and not on the response values. They are interested in sampling additional design points in the region with high-variability in response. We will fail to identify these regions if we operate with a stationary GP model. Instead, Marmin et al. (2018) propose a new design criterion for detecting high variability in the model response based on the gradient of the GP.

Marmin et al. (2018) start by defining a vector valued GP $\bigtriangledown f(\mathbf{x}) = \left(\frac{\partial f}{\partial x_1}, \dots \frac{\partial f}{\partial x_p}\right)$, whose distribution conditional on ensemble, $\left\{\mathrm{X}, \mathrm{F}\right\}$, depends on the derivatives of $m^*(\cdot)$ and $C^*(\cdot, \cdot)$ defined in equation (2.6) and equation (2.7) respectively

$$
\begin{aligned}
E[\bigtriangledown f(\mathbf{x}) | \left\{\mathrm{X}, \mathrm{F}\right\}] &= \bigtriangledown m^*(\mathbf{x}) \\
Cov[\bigtriangledown f(\mathbf{x}), \bigtriangledown f(\mathbf{x}') | \left\{\mathrm{X}, \mathrm{F}\right\}] &= \left(\frac{\partial^2}{\partial t_i \partial t'_j} C^*(\boldsymbol{t}, \boldsymbol{t}') \Big|_{\boldsymbol{t}=\mathbf{x}, \boldsymbol{t}'=\mathbf{x}'} \right)_{1 \leq i,j \leq p}.
\end{aligned}
$$

The squared gradient norm process $Q(\mathbf{x})$ is defined by

$$
Q(\mathbf{x}) = \bigtriangledown f(\mathbf{x})^T \bigtriangledown f(\mathbf{x})
$$

and is used to identify the direction of the largest change in response variability. Deriving a full probability distribution for $Q(\mathbf{x})$ is challenging and instead moments of $Q(\mathbf{x})$ are used in sampling criteria definition. In particular, the gradient norm

variance (GNV) criterion is defined as

$$J_n^{GNV,\eta}(\mathbf{x}) = Var[Q^{\eta/2}(\mathbf{x})|\{\mathrm{X}_n, \mathrm{F}_n\}, \eta > 0],$$

and the integrated gradient norm variance criterion (IGNV) as

$$J_n^{IGNV,\eta}(\mathbf{x}) = \int_{\boldsymbol{u}} E\Big[Var[Q^{\eta/2}(\boldsymbol{u})|\{\mathrm{X}_n, \mathrm{F}_n\}, f(\mathbf{x})]|\{\mathrm{X}_n, \mathrm{F}_n\}\Big] d\boldsymbol{u}.$$

Marmin et al. (2018) are interested in sampling the next design point, $\mathbf{x}_{n+1}$, with a high gradient-based criterion value, since it indicates that this point is located in the high-variability region. To summarize they are interested in maximizing these gradient-based criteria in the process of obtaining sequential design, i.e.

$$\mathbf{x}_{n+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmax}}\, J_n^{GNV,\eta}(\mathbf{x}), \quad \mathbf{x}_{n+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmax}}\, J_n^{IGNV,\eta}(\mathbf{x}).$$

These new design criteria could be considered as improved alternatives to MSE and IMSE by including the gradient of the GP explicitly so that users are able to locate input regions where the model response variability increases (input regions with high slopes). In this case, the prior kernel of GP is arbitrarily defined. Marmin et al. (2018) demonstrated that the proposed design with stationary GP outperforms the variance-based designs on a number of examples. However, the forms of the proposed criteria are very sensitive to the choice of the η parameter, and it is not clear what are the guidelines for specifying the value of η parameter.

6.2.2 Approaches to updating a nonstationary GP emulator

We proceed to consider different approaches adopted to updating, or refitting nonstationary GP model after obtaining the follow-up computer runs found by optimizing one of the criterion introduced in subsection 6.2.1. The choice of the approach to updating largely depends on a number of factors such as

- the knowledge and assumption about the parametric form of $f(\mathbf{x})$
- how we treat and estimate parameters of GP model

- the computational costs of obtaining the follow-up computer model runs.

Montagna and Tokdar (2016) propose the following parametric form for $f(\mathbf{x})$, defined as

$$f(\mathbf{x}) = h(\mathbf{x})^T \boldsymbol{\beta} + \epsilon(\boldsymbol{x}, Z),$$

where a residual term, $\epsilon(\boldsymbol{x}, Z)$, is a zero-mean GP whose covariance is a function of input $\mathbf{x}$ and a latent input Z. Since the parametric form of $f(\mathbf{x})$ is known a priori, the design points $\mathbf{x}_{n+1}$ could be chosen and sampled sequentially and the nonstationary GP emulator updated conditioned on a new pair $(\mathbf{x}_{n+1}, f(\mathbf{x}_{n+1}))$. The updating of nonstationary GP model is performed by employing particle learning (PL) (Lopes and Tsay, 2011). The idea behind PL is to identify N particles $\left\{S_t^{(i)}\right\}_{i=1}^N$, which contain all the information about the uncertainties up to time t and used to approximate the posterior distribution, $\left\{S_t^{(i)}\right\}_{i=1}^N \sim \pi(S_t|\mathrm{X}_t, \mathrm{F}_t)$, where $\mathrm{X}_t = (\mathbf{x}_1, \ldots, \mathbf{x}_t)$ and $\mathrm{F}_t = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_t))$. In the case of nonstationary GP emulator approach, the particles, $\left\{S_t^{(i)}\right\}_{i=1}^N = \left\{(\boldsymbol{Z}_t, K_t, \tilde{K}_t)^{(i)}\right\}$, are introduced, with latent input vector, $\boldsymbol{Z}_t = (z_1, \ldots, z_t)$, and at each stage the parameters of models, i.e. correlation function parameters and latent input vector are updated to avoid particle depletion. Particle depletion is defined as a poor approximation of particles to posterior distribution due to lack of update of unknown quantities and model hyperparameters (see Gilks and Berzuini (2001) for more details). Particle learning could be efficiently used due to the conjugate prior specification for GP hyperparameters that allowed Montagna and Tokdar (2016) to derive the predictive posterior distribution for f conditioned on ensemble, $\left\{\mathrm{X}_t, \mathrm{F}_t\right\}$, a vector of latent input parameters, $\boldsymbol{Z}_t$, and matrices of correlations K_t and $\tilde{K}_t$. Since hyperparameters $\boldsymbol{\beta}$ and σ^2 are marginalised, only correlation function parameters, $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_p)$ and $\tilde{\boldsymbol{\phi}} = (\tilde{\phi}_1, \ldots, \tilde{\phi}_p)$, have to be updated. We think that it is possible to apply the proposed nonstationary GP model to a classical design criterion. However, it is not clear how the complexity of the updating changes with a different, more flexible GP hyperparameter prior specification for $\boldsymbol{\beta}$ and σ^2. Another question is whether PL could be used to choose a batch of design points.

Contrary to Montagna and Tokdar (2016), Gramacy and Lee (2009) do not

assume that the model for $f(\mathbf{x})$ is known a priori. Therefore to employ ALC together with TGP, Bayesian MCMC posterior inference on $\{\mathcal{T}, \Theta\}$ is performed described in Gramacy and Lee (2008), where $\mathcal{T}$ is a decision tree, and Θ is a collection of region-specific hyperparameters, and then samples from ALC are taken conditional on samples from $\{\mathcal{T}, \Theta\}$.

Gramacy and Lee (2009) argue that many complex computer experiment runs are produced in asynchronous distributed computer environment such as computing agents, or processors, which tend to start and finish simulations at different times. Sequential design is ill-suited for this type of computing environment since it requires the computing agents to operate in parallel and at each step refit, or update, the model, conditioned on newly obtained computer model runs. Instead, Gramacy and Lee (2009) propose a Bayesian Adaptive Sampling (BAS) which incorporates flexible design criterion (ALC and ALM) as well as copes with an asynchronous, agent-based computing environment.

Bayesian adaptive sampling (BAS) proceeds in trials. As before we assume that n design points and the corresponding responses have been gathered in previous trials, i.e. $\{\mathrm{X}_n, \mathrm{F}_n\}$. We proceed to describe the steps involved in performing a single trial of Bayesian adaptive sampling:

1. TGP model is estimated for $\{\mathrm{X}_n, \mathrm{F}_n\}$ (for more details see Gramacy and Lee (2008)).

2. Generate a space-filling candidate set of input points $\tilde{X} \in \mathcal{X}$ of size m. Proceed to obtaining samples of ALM or ALC conditional on $\{\mathcal{T}, \Theta\}$ at $\tilde{X}$.

3. A queue of ordered candidates according to ALM and ALC score is formed.

4. BAS gathers the finished and running input configurations and adds them to the design, X_{n+m}. Predictive expectation is used for running configurations until the computer model run is available.

Before proceeding to a next trial of Bayesian adaptive sampling, TGP model has to be updated. Instead of refitting TGP model for $\{\mathrm{X}_{n+m}, \mathrm{F}_{n+m}\}$, Gramacy and Lee (2009) propose to randomly prune back the tree $\mathcal{T}$ which is claimed to allow

short burn-in of the MCMC at the beginning of each trial and therefore provides computational savings.

Contrary to two different fully Bayesian approaches described above, Marmin et al. (2018) estimate nonstationary GP model parameters by likelihood maximization. In particular, after obtaining a pair $\big(\mathbf{x}_{n+1}, f(\mathbf{x}_{n+1})\big)$ a likelihood function is specified $p\big(\mathrm{F}_{n+1}|f(\mathbf{x})\big)$, and the nonstationary GP model hyperparameters are fixed at the maximum likelihood estimates.

6.3 Bayesian Design Criterion with nonstationary GP emulator with kernel mixtures

Since we are interested in obtaining an optimal design, it is crucial to keep in mind that the optimality is found with respect to the assumed parametric form for $f(\mathbf{x})$ (Gramacy and Lee, 2009). Therefore, when employing our proposed nonstationary GP model inside BDC, we have to fix the number of input regions L to ensure that the obtained design is optimal. The fixed L is also necessary for performing an update on posterior distribution for $f(\mathbf{x})$ after observing $\big\{\xi, f(\xi)\big\}$.

We employ the nonstationary GP model defined in equation (4.3) in subsection 4.3.1. We obtain a distribution $\pi_1\big(f(\mathbf{x})\big)$ that represents a posterior distribution for f at a new input point $\mathbf{x}$ given L and is used to perform wave 1 of history matching. In this Chapter, we only attempt to find Bayesian Optimal Design (BOD) to perform wave 2 of history matching.

We demonstrated in subsection 5.3.1 that the proposed design criterion is decomposed into three interpretable terms, precisely $\Psi_1(\xi)$, $\Psi_2(\xi)$ and $\Psi_3(\xi)$. By employing our proposed nonstationary GP model, we act under the assumption that there exist L input regions of distinct model behaviour $l = 1, \ldots, L$, such that $\cup_{l=1}^{L} \mathcal{X}_l = \mathcal{X}$, which will have a direct impact on the BDC and its individual terms. In subsection 4.3.3 we proposed to impose an ordering on standard deviation parameter for the distribution of standardized errors from stationary GP fit inside the mixture model, i.e. $\zeta_1 \leq \zeta_2 \leq \cdots \leq \zeta_L$. This condition is employed in to ensure a good mixing

of Markov chains for mixture model parameters. However, this condition also leads to the restrictions on our proposed nonstationary GP model, in particular, $\mathcal{X}_L$ corresponds to the input space with high variability in response since we expect to observe an increase in the range of standardized error' values expressed via ζ_L. On the contrary, $\mathcal{X}_1$ corresponds to the input space where the model response is "well-behaved" since we expect to obtain lower absolute values of standardized errors from stationary fit. We discussed in section 4.8 that a possible extension to the proposed nonstationary GP model is to remove a two-stage approach and dependence on the standardized errors from stationary fit and operate with a joint prior distribution $\pi\big(\boldsymbol{\beta}, \{\sigma_l^2, \boldsymbol{\delta}_l, \tau_l^2, \lambda_l(\mathbf{x})\}_{l=1}^L\big)$. However, to aid our presentation of a method, we still refer to $\mathcal{X}_L$ and $\mathcal{X}_1$ as input regions with high variability and low variability in model response respectively.

We assume that after performing wave 1 of history matching, we obtained an NROY space $\mathcal{X}^1$, defined as

$$\begin{aligned}
&\mathcal{X}^1 = \cup_{l=1}^L \mathcal{X}_l^1 \\
&\mathcal{X}_l^1 \subset \mathcal{X}^1, l = 1, \ldots, L, \\
&\text{s.t. } \mathbf{x} \in \mathcal{X}_l^1 \Leftrightarrow \lambda_l(\mathbf{x}) > \lambda_j(\mathbf{x}), j = \big\{1, \ldots, L\big\}/l.
\end{aligned}$$

Based on the decomposition of BDC, given in equation (5.11), together with the NROY space $\mathcal{X}^1$ decomposition, we rewrite the expression for the BDC. The first term of the expected loss function, $\Psi_1(\xi)$, corresponds to the expected volume of NROY space at wave 2 and is defined as

$$\begin{aligned}
\Psi_1(\xi) = \sum_{l=1}^{L} \Bigg(\int\int_{\mathcal{X}_l^1} \mathbb{1}\left\{ \frac{|z - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \right\} \\
\times \pi(f(\xi)|\mathrm{F}_{[1]}) d\mathbf{x} df(\xi) \Bigg) = \sum_{l=1}^{L} \Psi_{1l}(\xi). \qquad (6.3)
\end{aligned}$$

Similarly, we redefine $\Psi_2(\xi)$ and $\Psi_3(\xi)$ terms of BDC. We specify $\Psi_2(\xi)$ that corresponds to the expected volume of the input region that is in both NROY at

wave 2 and "true" NROY, as

$$\Psi_2(\xi) = \sum_{l=1}^{L} \Bigg(\int\int_{\mathcal{X}_l^1} \Bigg[\Phi\Bigg(\frac{z + a\sqrt{Var[e] + Var[\eta]} - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}} \Bigg) - \Phi\Bigg(\frac{z - a\sqrt{Var[e] + Var[\eta]} - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}} \Bigg) \Bigg] \times \mathbb{1}\Bigg\{ \frac{|z - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \Bigg\} \pi(f(\xi)|\mathrm{F}_{[1]}) d\mathbf{x} df(\xi) \Bigg)$$
$$= \sum_{l=1}^{L} \Psi_{2l}(\xi). \tag{6.4}$$

Finally, we re-define $\Psi_3(\xi)$, that corresponds to the expected volume of "true" NROY space, as

$$\Psi_3(\xi) = \sum_{l=1}^{L} \Bigg(\int\int_{\mathcal{X}_l^1} \Bigg[\Phi\Bigg(\frac{z + a\sqrt{Var[e] + Var[\eta]} - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}} \Bigg) - \Phi\Bigg(\frac{z - a\sqrt{Var[e] + Var[\eta]} - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}} \Bigg) \Bigg] \pi(f(\xi)|\mathrm{F}_{[1]}) d\mathbf{x} df(\xi) \Bigg)$$
$$= \sum_{l=1}^{L} \Psi_{3l}(\xi). \tag{6.5}$$

In section 5.3, we discussed the effect of expectation, $E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, on the integrand functions of the expected loss terms. In particular, we tend to observe greater contributions towards the values of $\Psi_1(\xi)$ and $\Psi_2(\xi)$ from the input points $\mathbf{x} \in \mathcal{X}^1$, at which we expect the model output to be close to z. We also tend to observe greater contribution towards the final value of $\Psi_3(\xi)$ and $\Psi_2(\xi)$ from the input points $\mathbf{x} \in \mathcal{X}^1$ at which $E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$ lies inside the interval

$$\left[z - a\sqrt{Var[e] + Var[\eta]}, z + a\sqrt{Var[e] + Var[\eta]} \right].$$

We propose to employ our nonstationary GP to model heterogeneous response. The modifications to the "out of the box", stationary GP emulator are implemented in a way that users obtain a fair assessment of uncertainty about the model response. In particular, we express our uncertainty about model behaviour f at input point $\mathbf{x}$ via

predictive variance, $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$. By employing our proposed nonstationary GP emulator, we obtain larger values of predictive variance at $\mathbf{x} \in \mathcal{X}_L^1$. On the contrary, we obtain lower values of predictive variance at $\mathbf{x} \in \mathcal{X}_1^1$. Therefore we expect to observe a variability in $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$ across $\mathcal{X}^1$, which would lead to the variability in the integrand function values of expected loss terms.

In Chapter 5, we mainly considered the effect of predictions, $E_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$, and predictive variance, $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$, obtained at an arbitrary input point $\mathbf{x}$ on the integrand functions of individual terms of BDC. However, by employing our proposed nonstationary GP model as part of BDC, we are required to take into account the relative volumes of input space regions with characteristic model behaviour, i.e. $\mathcal{X}_l^1, l = 1, \ldots, L$. For instance, if the volume of $\mathcal{X}_L^1$ is significantly smaller than the volumes of $\mathcal{X}_l^1, l = \{1, \ldots, L-1\}$, we would observe smaller contributions from $\Psi_{1L}(\xi)$, $\Psi_{2L}(\xi)$ and $\Psi_{3L}(\xi)$ towards the final values of $\Psi_1(\xi)$, $\Psi_2(\xi)$ and $\Psi_3(\xi)$ respectively.

In subsection 6.3.1, we consider in detail the behaviour of $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$, obtained as part of our proposed nonstationary GP model, across the input space $\mathcal{X}$. The predictive variance is considered for two main reasons. Firstly, this term is crucial in obtaining the reduced input space $\mathcal{X}^1$. Secondly, this term is part of $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$. In subsection 6.3.2, we consider the effect of placing ξ in different regions of input space on $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$ values. In subsection 6.3.3, we draw conclusions about terms of expected loss function of BDC employed with nonstationary GP model.

6.3.1 Covariance structure of a nonstationary GP model employed at Wave 1

We define the predictive variance, $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$, of distribution $\pi_1\big(f(\mathbf{x})\big)$ as

$$\sum_{l=1}^{L}\Big[\lambda_l(\mathbf{x})k_l(\mathbf{x},\mathbf{x})\lambda_l(\mathbf{x}) - \lambda_l(\mathbf{x})k_l(\mathbf{x},\mathrm{X}_{[1]})\Lambda_l(\mathrm{X}_{[1]})\big[K_{[1]}\big]^{-1}\Lambda_l(\mathrm{X}_{[1]})k_l(\mathrm{X}_{[1]},\mathbf{x})\lambda_l(\mathbf{x})\Big]$$
$$-\sum_{l=1}^{L}\lambda_l(\mathbf{x})k_l(\mathbf{x},\mathrm{X}_{[1]})\Lambda_l(\mathrm{X}_{[1]})\big[K_{[1]}\big]^{-1}\Big(\sum_{j\neq l}^{L}\Lambda_j(\mathrm{X}_{[1]})k_j(\mathrm{X}_{[1]},\mathbf{x})\lambda_j(\mathbf{x})\Big), \tag{6.6}$$

where $\Lambda_l(\mathrm{X}_{[1]})$ is $n_1 \times n_1$ diagonal matrix with ith diagonal entry $\lambda_l(\mathbf{x}_{i1})$. We also specify $n_1 \times n_1$ covariance matrix $K_{[1]}$, i.e.

$$K_{[1]} = \sum_{l=1}^{L} \Lambda_l(\mathrm{X}_{[1]}) k_l(\mathrm{X}_{[1]}, \mathrm{X}_{[1]}) \Lambda_l(\mathrm{X}_{[1]}).$$

We observe that the first part of the predictive variance in equation (6.6) corresponds to the sum of predictive variances computed using an individual region-specific kernel and weighted by the corresponding mixture component. We define a weighted precision term as $\Lambda_l(\mathrm{X}_{[1]})\left[K_{[1]}\right]^{-1}\Lambda_l(\mathrm{X}_{[1]})$ and investigate it in detail with the help of a simple example.

Assume that we obtain $\mathrm{X}_{[1]} = (\mathbf{x}_{11}, \mathbf{x}_{21}, \mathbf{x}_{31})$ and therefore $K_{[1]}$ is 3×3 covariance matrix computed at $\mathrm{X}_{[1]}$. We define a precision matrix $Q = \left[K_{[1]}\right]^{-1}$ with $Q_{im} = q_{im}$ and $i, m = 1, 2, 3$ and proceed to compute a weighted precision, i.e.

$$\Lambda_l(\mathrm{X}_{[1]}) Q \Lambda_l(\mathrm{X}_{[1]}) = \begin{pmatrix} \lambda_l(\mathbf{x}_{11})^2 q_{11} & \lambda_l(\mathbf{x}_{11})\lambda_l(\mathbf{x}_{21}) q_{12} & \lambda_l(\mathbf{x}_{11})\lambda_l(\mathbf{x}_{31}) q_{13} \\ \lambda_l(\mathbf{x}_{21})\lambda_l(\mathbf{x}_{11}) q_{21} & \lambda_l(\mathbf{x}_{21})^2 q_{22} & \lambda_l(\mathbf{x}_{21})\lambda_l(\mathbf{x}_{31}) q_{23} \\ \lambda_l(\mathbf{x}_{31})\lambda_l(\mathbf{x}_{11}) q_{31} & \lambda_l(\mathbf{x}_{31})\lambda_l(\mathbf{x}_{21}) q_{32} & \lambda_l(\mathbf{x}_{31})^2 q_{33} \end{pmatrix}.$$

For instance, if we observe $\lambda_l(\mathbf{x}_{31}) \to 0$, which corresponds to the low probability of point $\mathbf{x}_{31}$ being allocated to input region $\mathcal{X}_l$, this, in turn, will lead to entries on the third row and the third column to be close to zero, i.e.

$$\lambda_l(\mathbf{x}_{31})\lambda_l(\mathbf{x}_{i1}) q_{3i} = \lambda_l(\mathbf{x}_{i1})\lambda_l(\mathbf{x}_{31}) q_{i3} \to 0 \quad i = 1, 2, 3.$$

Based on this simple example, we conclude that in the process of computing predictive variance using a region-specific kernel and mixture component, say from region l, we reduce the effect from design points in $\mathrm{X}_{[1]}$ with a low probability of being allocated to $\mathcal{X}_l$ since we observe close to zero entries in weighted precision matrix at these points.

We draw a connection between our weighted precision matrix and a precision matrix computed as part of the Gaussian Markov random field (GMRF). GMRF is a discretely indexed Gaussian field, defined as $\mathrm{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$, with Markov

properties (Rue et al., 2009; Lindgren et al., 2011). In particular, the conditional distribution for $\mathbf{x}_i$, $\pi\big(\mathbf{x}_i|\mathrm{X}_{-i}\big)$, where $\mathrm{X}_{-i} = \big(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_n\big)$, depends only on a set of neighbours, $\mathcal{N}_i$, to a point $\mathbf{x}_i$ (Rue and Held, 2005). These Markov properties are embedded inside the precision matrix Q, for which the non-zero entries only correspond to the neighbours and diagonal elements, i.e.

$$Q_{im} = 0 \quad m \notin \big\{i, \mathcal{N}_i\big\}.$$

In our case by employing our proposed nonstationary GP model, in the first term of equation (6.6) we observe entries of weighted precision, $\Lambda_l(\mathrm{X}_{[1]})Q\Lambda_l(\mathrm{X}_{[1]})$, close to zero for points in $\mathrm{X}_{[1]}$ with a low probability of being allocated to input region l, i.e.

$$\lambda_l(\mathbf{x}_{i1})\lambda_l(\mathbf{x}_{m1})q_{im} \to 0 \quad \text{ if } \lambda_l(\mathbf{x}_{i1}) \to 0 \text{ or } \lambda_l(\mathbf{x}_{m1}) \to 0,$$

which indicates that we are mainly interested to include information about f locally from design points within $\mathcal{X}_l$.

The second component of equation (6.6) represents our beliefs that the variability in model response is continuous across the input space and we are interested in using information across boundaries of input regions with distinct model behaviour in our predictive variance computation. As part of the second component, we are required to compute weighted precisions of the form

$$\Lambda_l(\mathrm{X}_{[1]})\big[K_{[1]}\big]^{-1}\Lambda_j(\mathrm{X}_{[1]}) = \Lambda_l(\mathrm{X}_{[1]})Q\Lambda_j(\mathrm{X}_{[1]}).$$

We tend to observe the entries of a weighted precision matrix close to zero for input points with low probabilities of being allocated to $\mathcal{X}_l$ and $\mathcal{X}_j$, i.e.

$$\lambda_l(\mathbf{x}_{i1})q_{im}\lambda_j(\mathbf{x}_{m1}) \to 0 \quad \text{if } \lambda_l(\mathbf{x}_{i1}) \to 0 \text{ or } \lambda_j(\mathbf{x}_{m1}) \to 0,$$

which indicates that we are mainly interested in including information from design points within input regions $\mathcal{X}_l$ and $\mathcal{X}_j$.

We also propose to consider the extreme case, at input point $\mathbf{x} \in \mathcal{X}_l$ with $\lambda_l(\mathbf{x}) \gg$

$\lambda_j(\mathbf{x}), j = \{1, \ldots, L\}/l$ and since $\sum_{l=1}^{L} \lambda_l(\mathbf{x}) = 1$, we observe that $\lambda_l(\mathbf{x}) \to 1$ and $\lambda_j(\mathbf{x}) \to 0, j = \{1, \ldots, L\}/l$. For this particular extreme case, we obtain the following approximation to the predictive variance

$$Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})] \approx k_l(\mathbf{x}, \mathbf{x}) - k_l(\mathbf{x}, \mathrm{X}_{[1]})\Lambda_l(\mathrm{X}_{[1]})\left[K_{[1]}\right]^{-1}\Lambda_l(\mathrm{X}_{[1]})k_l(\mathrm{X}_{[1]}, \mathbf{x}). \tag{6.7}$$

We discussed in detail the effect of weighted precision, $\Lambda_l(\mathrm{X}_{[1]})\left[K_{[1]}\right]^{-1}\Lambda_l(\mathrm{X}_{[1]})$, in particular the reduced effect, or even the discard of information, from the design points in $\mathrm{X}_{[1]}$ with a low probability of being allocated to $\mathcal{X}_l$ and strengthening of the effect of information from the design points with a higher probability of being allocated to $\mathcal{X}_l$. We can draw the similarities between the predictive variance behaviour in this extreme case and the predictive variance obtained with partition-based nonstationary GP models such as TGP (Gramacy and Lee, 2008) and Voronoi tessellation GP model (Kim et al., 2005; Pope et al., 2018).

In general, we expect to obtain larger values of $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ computed at $\mathbf{x} \in \mathcal{X}_L$, in particular at $\mathbf{x} \in \mathcal{X}_L$ with $\lambda_L(\mathbf{x}) \gg \lambda_l(\mathbf{x}), l = \{1, \ldots, L-1\}$, than predictive variance values computed at $\mathbf{x} \in \mathcal{X}_1$. This argument is important for two reasons.

Firstly, $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ is employed in our computations of the implausibility measure, $\mathcal{I}(\mathbf{x})$, across the original input space, $\mathbf{x} \in \mathcal{X}$, and, therefore, has a direct effect on the shape and composition of wave 1 NROY space, $\mathcal{X}^1$. In particular in section 2.6.1, we mentioned that low values of implausibility function, $\mathcal{I}(\mathbf{x})$, could occur when our emulator is uncertain about the model behaviour, which is expressed via large values of $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$. In case when we are operating with a nonstationary GP emulator, this means that we could potentially retain input regions with high variability in model response as part of wave 1 NROY space.

Secondly, $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ is the first part of the updated variance term that we employ in BDC computation for wave 2. We consider in detail the effect of placing candidate design points ξ in $\mathcal{X}^1$ on $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, and as a result their potential effect on BDC, which is explicitly discussed in subsection 6.3.3.

6.3.2 Covariance structure of a nonstationary GP model employed in Wave 2

In this subsection, we operate under the assumption that the NROY space after performing wave 1 of history matching is defined as

$$\mathcal{X}^1 = \cup_{l=1}^{L} \mathcal{X}_l^1, \quad \mathcal{X}_l^1 = \emptyset, l = 1, \ldots, L.$$

To perform wave 2 of history matching, we update our knowledge about $f(\mathbf{x})$, $\pi_1\big(f(\mathbf{x})\big)$, and obtain a distribution $\pi_2\big(f(\mathbf{x})\big)$ for f at a new input point $\mathbf{x}$ given a candidate ensemble, $\big\{\xi, f(\xi)\big\}$, mixture components, $\boldsymbol{\lambda}(\mathbf{x}) = \big\{\lambda_l(\mathbf{x})\big\}_{l=1}^{L}$, and parameters $\boldsymbol{\beta}, \Delta, \boldsymbol{\sigma}^2, \boldsymbol{\tau}^2$, defined as

$$f(\mathbf{x})|\big\{\xi, f(\xi)\big\}, \boldsymbol{\lambda}(\mathbf{x}), \boldsymbol{\beta}, \boldsymbol{\sigma}^2, \Delta, \boldsymbol{\tau}^2 \sim GP\Big(E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})], Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]\Big),$$

where expectation, $E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, and variance, $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, computed via equation (5.3) and equation (5.4) respectively. At wave 1 of history matching, we employed $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ as a measure of our uncertainty about the model response behaviour across the original input space $\mathcal{X}$. Therefore, we consider the effect of adding candidate design ξ on the values of $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, in particular, the reduction in uncertainty provided by evaluating computer model at the candidate design, ξ, to obtain $f(\xi)$ and expressed via

$$Cov_{\mathrm{F}_{[1]}}[f(\mathbf{x}), f(\xi)]\big[Var_{\mathrm{F}_{[1]}}[f(\xi)]\big]^{-1} Cov_{\mathrm{F}_{[1]}}[f(\xi), f(\mathbf{x})].$$

This term is considered to play an important role in the process of Bayesian updating since it contains a candidate design ξ. In Bayes Linear, this term corresponds to (partial) resolved variance of $f(\mathbf{x})$ by $f(\xi)$ given $\mathrm{F}_{[1]}$ (Goldstein and Wooff, 2007). The partial resolved variance is considered to be useful in evaluating the effect of adding $f(\xi)$ to $\mathrm{F}_{[1]}$ on $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$ and therefore is crucial in guiding the data collection process. By employing our nonstationary GP model, we express

$Cov_{\mathrm{F}_{[1]}}[f(\mathbf{x}), f(\xi)]$ as

$$\sum_{l=1}^{L}\Big[\lambda_l(\mathbf{x})k_l(\mathbf{x},\xi)\Lambda_l(\xi) - \lambda_l(\mathbf{x})k_l(\mathbf{x},\mathrm{X}_{[1]})\Lambda_l(\mathrm{X}_{[1]})\big[K_{[1]}\big]^{-1}\Lambda_l(\mathrm{X}_{[1]})k_l(\mathrm{X}_{[1]},\xi)\Lambda_l(\xi)\Big]$$
$$-\sum_{l=1}^{L}\lambda_l(\mathbf{x})k_l(\mathbf{x},\mathrm{X}_{[1]})\Lambda_l(\mathrm{X}_{[1]})\big[K_{[1]}\big]^{-1}\Big(\sum_{j\neq l}^{L}\Lambda_j(\mathrm{X}_{[1]})k_j(\mathrm{X}_{[1]},\xi)\Lambda_j(\xi)\Big). \tag{6.8}$$

In particular, the adjusted covariance between $f(\mathbf{x})$ and $f(\xi)$ given $\mathrm{F}_{[1]}$ in equation (6.8) builds a connection between $f(\mathbf{x})$ with $\mathbf{x} \in \mathcal{X}^1$ and $f(\xi)$. Conclusions about the adjusted covariance are very similar to the findings of $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ described in subsection 6.3.1. At an arbitrary input point $\mathbf{x} \in \mathcal{X}^1$ in our calculations of the adjusted covariance between $f(\mathbf{x})$ and $f(\xi)$ given $\mathrm{F}_{[1]}$, we attempt to include the information about the geometry of the design, $\mathrm{X}_{[1]}$, across the input space together with our nonstationary GP model definition to guide the decision wherein input space to place a candidate design ξ.

We are interested to consider an extreme case for $\mathbf{x} \in \mathcal{X}_l^1$ with $\lambda_l(\mathbf{x}) \gg \lambda_j(\mathbf{x}), j = \big\{1,\ldots,L\big\}/l$. In particular, we obtain the following approximation to the adjusted covariance between $f(\mathbf{x})$ and $f(\xi)$ given $\mathrm{F}_{[1]}$,

$$Cov_{\mathrm{F}_{[1]}}[f(\mathbf{x}), f(\xi)] \approx k_l(\mathbf{x},\xi)\Lambda_l(\xi) - k_l(\mathbf{x},\mathrm{X}_{[1]})\Lambda_l(\mathrm{X}_{[1]})\big[K_{[1]}\big]^{-1}\Lambda_l(\mathrm{X}_{[1]})k_l(\mathrm{X}_{[1]},\xi)\Lambda_l(\xi)$$
$$-k_l(\mathbf{x},\mathrm{X}_{[1]})\Lambda_l(\mathrm{X}_{[1]})\big[K_{[1]}\big]^{-1}\Big(\sum_{j\neq l}\Lambda_j(\mathrm{X}_{[1]})k_j(\mathrm{X}_{[1]},\xi)\Lambda_j(\xi)\Big). \tag{6.9}$$

The first line of equation (6.9) corresponds to the computation of adjusted covariance between $f(\mathbf{x})$ and $f(\xi)$ given $\mathrm{F}_{[1]}$ by employing a region-specific kernel together with a region-specific mixture component at ξ. Since we include weighted precision, $\Lambda_l(\mathrm{X}_{[1]})\big[K_{[1]}\big]^{-1}\Lambda_l(\mathrm{X}_{[1]})$, as part of this expression, for which we expect to observe entries of a matrix close to zero that relate to points in $\mathrm{X}_{[1]}$ with low probabilities of being allocated to $\mathcal{X}_l$. This, as a result, will lead to less effect or discard of information about the model behaviour from input points in ξ with low probabilities of being allocated to $\mathcal{X}_l$. The terms of the second line of equation (6.9) are used to update our prior covariance between $f(\mathbf{x})$ and $f(\xi)$ by employing the information from the input points in ξ placed in input regions l and $j, j = \big\{1,\ldots,L\big\}/l$. This

term highlights that we do not believe in boundary discontinuities between two input regions imposed by partition-based nonstationary GP models. Based on equations (6.8) and (6.9), we conclude that at an arbitrary input point $\mathbf{x} \in \mathcal{X}_l^1, l = 1, \ldots, L$ we mainly use in our predictions the information about the model behaviour from input points in candidate design, ξ, placed in and around the same input region $\mathcal{X}_l^1$.

Since we stated in subsection 6.3.1 that the predictive variance term values $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ varies across the input space, in particular we observe larger values of predictive variance at $\mathbf{x} \in \mathcal{X}_L^1$, than the same term being computed at $\mathbf{x} \in \mathcal{X}_1^1$. Based solely on one of the variance-based criteria, we prefer to choose candidate design, ξ, with a majority of design point being placed in the high-variability region, as we expect to obtain a greater reduction in the uncertainty expressed via $Cov_{\mathrm{F}_{[1]}}[f(\mathbf{x}), f(\xi)]\big[Var_{\mathrm{F}_{[1]}}[f(\xi)]\big]^{-1} Cov_{\mathrm{F}_{[1]}}[\mathrm{F}_{[1]}, f(\mathbf{x})]$, leading to the lower values of $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$ at an arbitrary input point in $\mathcal{X}^1$.

6.3.3 Linking BDC to nonstationary GP model

We return to consider each term of the proposed BDC. The primary purpose of this subsection is to link our findings about $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$ behaviour across $\mathcal{X}^1$ with a modified form of BDC employed with nonstationary GP emulator.

In subsection 5.3.2, we demonstrated that $\Psi_1(\xi)$ contains an integrand function with a membership rule, that depends on the implausibility function $\mathcal{I}(\mathbf{x})$ and we express $\Psi_1(\xi)$ in equation (6.3) as a sum of L distinct terms $\Psi_{1l}(\xi), l = 1, \ldots, L$, defined as

$$\Psi_{1l}(\xi) = \int\int_{\mathcal{X}_l^1} \mathbb{1}\left\{\frac{|z - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a\right\} \pi(f(\xi)|\mathrm{F}_{[1]}) d\mathbf{x} df(\xi). \tag{6.10}$$

The first term of the expected loss function expresses our aim to choose candidate design points, ξ, that would lead us to a tighter calibration. One way to achieve a tighter calibration is to obtain computer model runs, $f(\xi)$, that would reduce the values of predictive variance, $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, across $\mathcal{X}^1$. In particular, by reducing our uncertainty about model behaviour, we mainly expect to retain input

points with conditional mean close to observation z inside wave 2 NROY space.

Since we operate with a nonstationary response, which is explicitly modelled by our proposed nonstationary GP emulator, one option is to place the majority of candidate design points in the high-variability region $\mathcal{X}_L^1$. As a result, we expect to observe a greater reduction in the predictive variance term $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$ at the input points $\mathbf{x} \in \mathcal{X}_L^1$. However, equation (6.10) demonstrates that the contribution from Ψ_{1l} towards the final value of $\Psi_1(\xi)$ also depends on the size of $\mathcal{X}_l^1$, since we are computing an integrand function over the input region $\mathcal{X}_l^1, l = 1, \ldots, L$. For instance, if after performing Wave 1 of history matching the volume of $\mathcal{X}_L^1$ is small relative to $\mathcal{X}^1$, we will observe smaller contribution from $\Psi_{1L}(\xi)$ towards the final value of $\Psi_1(\xi)$.

Similar to $\Psi_1(\xi)$, we express the third term of BDC in equation (6.5) as a sum of L distinct terms $\Psi_{3l}(\xi), l = 1, \ldots, L$, defined as

$$\begin{aligned}\Psi_{3l}(\xi) = \int\int_{\mathcal{X}_l^1} \Bigg[\Phi\Bigg(\frac{z + a\sqrt{Var[e] + Var[\eta]} - E_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]}} \Bigg) \\ - \Phi\Bigg(\frac{z - a\sqrt{Var[e] + Var[\eta]} - E_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]}} \Bigg) \Bigg] \\ \times \pi(f(\xi)|\mathrm{F}_{[1]}) d\mathbf{x} df(\xi).\end{aligned} \tag{6.11}$$

In subsection 5.3.3, we demonstrated that lower value of $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$ obtained at $\mathbf{x} \in \mathcal{X}^1$ lead to larger absolute values of the two CDF functions conditioned on the expected value, $E_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$, being close to observation z within pre-specified tolerance to model error and observation error. By reducing our uncertainty about model behaviour, we mainly retain input points $\mathbf{x} \in \mathcal{X}^1$ inside the expected "true" NROY space with conditional mean close to observation z. Since we operate with a nonstationary model response, placing a large proportion of points in ξ in $\mathcal{X}_L^1$ will lead to the reduction in uncertainty about model behaviour in this particular region and result into greater change in $\Psi_{3L}(\xi)$ term value. However, similar to the analysis of the first term of BDC, the contribution from $\Psi_{3L}(\xi)$ towards the final value of the third term of BDC, $\Psi_3(\xi)$, is largely determined by the relative volume of $\mathcal{X}_L^1$.

Finally, we consider $\Psi_2(\xi)$ of the BDC, which in equation (6.4) is expressed as a sum of L separate terms $\Psi_{2l}(\xi), l = 1, \ldots, L$, defined as

$$\begin{aligned}\Psi_{2l}(\xi) = \int\int_{\mathcal{X}_l^1} \Bigg[& \Phi\Bigg(\frac{z + a\sqrt{Var[e] + Var[\eta]} - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}}\Bigg) \\ & - \Phi\Bigg(\frac{z - a\sqrt{Var[e] + Var[\eta]} - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}}\Bigg)\Bigg] \\ & \times \mathbb{1}\Bigg\{\frac{|z - E_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a\Bigg\} \pi(f(\xi)|\mathrm{F}_{[1]}) d\mathbf{x} df(\xi).\end{aligned} \tag{6.12}$$

The integrand in equation (6.12) contains the product of integrands from equation (6.10) and equation (6.11), i.e. the membership of an arbitrary input point $\mathbf{x} \in \mathcal{X}^1$ at the next iteration of history matching is now weighted by the difference of two CDFs. Similar arguments about the effect of reducing our uncertainty about the model behaviour on the value of integrand of $\Psi_{2l}(\xi)$ conditioned on the expetcted value being close to observation z are derived. As before we observe that the contribution, effect, from $\Psi_{2l}(\xi)$ towards the final term $\Psi_2(\xi)$ largely depends on the relative volume of $\mathcal{X}_l^1$.

The variance-based design criteria commonly employed as part of the sequential design with nonstationary GP models tend to encourage a selection of candidate design in the high-variability input region since obtaining computer model runs is expected to reduce our uncertainty about the model behaviour in this region. On the contrary, the BDC employed with our proposed nonstationary GP model has a number of unique and important features. Firstly, by employing our nonstationary GP model, we could explicitly include inside our BDC computation the variability in uncertainty about the model behaviour across $\mathcal{X}^1$ and how our uncertainty changes with the addition of candidate design. Secondly, the composition of reduced input space is considered inside BDC computation by expressing each individual term of BDC as a sum of L region-specific contributions towards these terms. For instance, by placing a significant amount of points ξ in $\mathcal{X}_L^1$ would change region-specific BDC terms, i.e. $\Psi_{1L}(\xi), \Psi_{2L}(\xi)$ and $\Psi_{3L}(\xi)$, due to the reduction in uncertainty about model behaviour in this region. However, the effect from these terms towards the

final terms of BDC, i.e. $\Psi_1(\xi), \Psi_2(\xi)$ and $\Psi_3(\xi)$ are limited if the relative volume of $\mathcal{X}_L^1$ is small. In this case, our BDC would indicate that the candidate design ξ with the majority of input points placed in the high-variability region is unfavourable choice to perform wave 2 of history matching.

In this Chapter, we consider the process of obtaining BOD for wave 2 of history matching and behaviour of individual components of BDC. The motivation behind this focus is that it is more convenient to draw a connection between the BDC individual terms and the proposed nonstationary GP model. In particular in our expressions of $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ and $Cov_{\mathrm{F}_{[1]}}[f(\mathbf{x}), f(\xi)]$ we explicitly used our kernel mixture formulation. However, we would like to point out that BDC employed with our proposed nonstationary GP model is not limited to $m = 2$ Wave of history matching, since for $m > 2$ the kernel mixture is contained inside $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$ and $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$.

6.4 Implementation details

In this section, we discuss a number of practical details that we implement to compute BDC with a nonstationary GP emulator. We start by describing the process of performing m waves of history matching.

1. Start with $m = 1$ and generate $\mathrm{X}_{[1]}$ by employing one of the available space-filling approaches and obtain computer model runs $\mathrm{F}_{[1]}$. Follow the procedure described in section 4.3 for fitting a nonstationary GP emulator by defining a prior distribution on $f(\mathbf{x})$, denoted $\pi\big(f(\mathbf{x})\big)$, a Gaussian process distribution with mean function

$$E\big[f(\mathbf{x})\big] = h(\mathbf{x})^T \boldsymbol{\beta}$$

and covariance function

$$\begin{aligned} Cov\big[f(\mathbf{x}), f(\mathbf{x}')\big] &= k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\sigma}^2, \Delta, \boldsymbol{\tau}^2) \\ &= \sum_{l=1}^{L} \hat{\lambda}_l(\mathbf{x}) \hat{\lambda}_l(\mathbf{x}') k_l(\mathbf{x}, \mathbf{x}'; \sigma_l^2, \boldsymbol{\delta}_l) + \mathbb{1}\big\{\mathbf{x} = \mathbf{x}'\big\} \sum_{l=1}^{L} z_l(\mathbf{x}) z_l(\mathbf{x}') \tau_l^2. \end{aligned}$$

Fix the number of input regions L and fix GP hyperparameters at MAP, $\Theta_{MAP} = \{\boldsymbol{\beta}_{MAP}, \boldsymbol{\sigma}^2_{MAP}, \Delta_{MAP}, \boldsymbol{\tau}^2\}$, throughout the whole Bayesian updating process in order to reduce computational efforts. The form of the mixture model is fixed, i.e. for $\mathbf{x} \in \mathcal{X}$ the values of a vector of mixture components $\boldsymbol{\lambda}(\mathbf{x}) = \big(\lambda_1(\mathbf{x}), \ldots, \lambda_L(\mathbf{x})\big)$ are computed for M posterior samples (after warm-up) and fixed at the mean value over the posterior samples denoted by $\hat{\boldsymbol{\lambda}}(\mathbf{x}) = \big(\hat{\lambda}_1(\mathbf{x}), \ldots, \hat{\lambda}_L(\mathbf{x})\big)$:

$$\hat{\boldsymbol{\lambda}}(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M} \boldsymbol{\lambda}(\mathbf{x}, A_m).$$

We use $E_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ and $Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]$ to obtain wave 1 NROY space, $\mathcal{X}^1$, which is defined as

$$\mathcal{X}^1 = \left\{\mathbf{x} \in \mathcal{X} : \frac{|z - E_{\mathrm{F}_{[1]}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\mathrm{F}_{[1]}}[f(\mathbf{x})]}} \leq a\right\}$$

2. At wave $m > 1$, we start with a prior distribution $\pi_{m-1}\Big(f(\mathbf{x})\Big)$ for $f(\mathbf{x})$ as

$$f(\mathbf{x})|\Theta_{MAP}, \hat{\boldsymbol{\lambda}}(\mathbf{x}) \sim GP\Big(E_{\langle \mathrm{F}\rangle_{[m-1]}}[f(\mathbf{x})], Var_{\langle \mathrm{F}\rangle_{[m-1]}}[f(\mathbf{x})]\Big)$$

and proceed to obtain an optimal Bayesian Design by minimizing $\Psi(\xi)$ using an optimization algorithm, where the size of ξ is pre-specified and fixed. We denote the derived Bayesian Optimal Design as $\mathrm{X}_{[m]} = \arg\min_{\xi} \Psi(\xi)$.

3. We generate computer model runs $\mathrm{F}_{[m]}$ at the design $\mathrm{X}_{[m]}$ and update the probability distribution for $f(\mathbf{x})$ to derive $\pi_m\big(f(\mathbf{x})\big)$, which is

$$f(\mathbf{x})|\{\mathrm{X}_{[m]}, \mathrm{F}_{[m]}\}, \Theta_{MAP}, \hat{\boldsymbol{\lambda}}(\mathbf{x}) \sim GP\Big(E_{\{\langle \mathrm{F}\rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})], Var_{\{\langle \mathrm{F}\rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]\Big).$$

4. We employ $E_{\{\langle \mathrm{F}\rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]$ and $Var_{\{\langle \mathrm{F}\rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]$ to obtain wave m NROY space, which is defined as

$$\mathcal{X}^m = \left\{\mathbf{x} \in \mathcal{X}^{m-1} : \frac{|z - E_{\{\langle \mathrm{F}\rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F}\rangle_{[m-1]}, \mathrm{F}_{[m]}\}}[f(\mathbf{x})]}} \leq a\right\}$$

Repeat Steps 2-4 until, for instance, experimental budget has been exhausted or no change in the NROY space size is observed.

We remark on the process of computing BDC and searching for an optimal design for wave m of history matching in practice. In section 6.3, we decomposed the individual terms of BDC into L terms computed over an input region with characteristic model behaviour. We redefined $\Psi(\xi)$ such as

$$\Psi(\xi) = \sum_{l=1}^{L} \Psi_{1l}(\xi) - 2\sum_{l=1}^{L} \Psi_{2l}(\xi) + \sum_{l=1}^{L} \Psi_{3l}(\xi) = \sum_{l=1}^{L} \Big(\Psi_{1l}(\xi) - 2\Psi_{2l}(\xi) + \Psi_{3l}(\xi) \Big),$$

where the integrand function of individual terms $\Psi_{1l}(\xi)$, $\Psi_{2l}(\xi)$ and $\Psi_{3l}(\xi)$ are computed over the input region $\mathcal{X}_l^{m-1} \subset \mathcal{X}^{m-1}$. We define $\mathcal{X}_l^{m-1}$ and $\mathcal{X}^{m-1}$ as

$$\begin{aligned} &\mathcal{X}^{m-1} = \cup_{l=1}^{L} \mathcal{X}_l^{m-1} \\ &\mathcal{X}_l^{m-1} \subset \mathcal{X}^{m-1} \\ &\text{s.t. } \mathbf{x} \in \mathcal{X}_l^{m-1} \Leftrightarrow \lambda_l(\mathbf{x}) > \lambda_j(\mathbf{x}), j = \big\{1, \ldots, L\big\}/l. \end{aligned}$$

However, in practice, we do not disaggregate the computation of individual terms of BDC with respect to $\mathcal{X}_l^{m-1}, l = 1, \ldots, L$ since this does not provide us with any computational gains. The main reason of employing this input space fragmentation in our description of proposed approach in section 6.3 is to demonstrate that our proposed BDC does not only take into account the variability in model response behaviour across the reduced input space via nonstationary GP model defined for $f(\mathbf{x})$, but it also includes the information about the decomposition of $\mathcal{X}^{m-1}$. In practice, we perform BDC computation in a similar way described in section 5.4.

Similar to Chapter 5, we face the following numerical problem in searching for optimal design: for fixed sample size n_m and Bayesian Design Criterion $\Psi(\xi)$, find $\xi = (\mathbf{x}_{1m}, \ldots, \mathbf{x}_{n_m m})$ that minimizes $\Psi(\xi)$, i.e. $\xi^* = \arg\min \Psi(\xi)$. We are faced with $n_m \times p$-dimensional optimization problem, which is more challenging since we are operating with a larger number of design points n_m and input parameters p.

In subsection 5.5.2, in our simulation study, we considered the NROY space obtained at wave 1, $\mathcal{X}^1$, as a continuous region. In this Chapter, we adopt the

sampling design on a discrete region since it is much easier to implement in input space with higher dimensions (Zhu and Stein, 2005). For our simulation studies in section 6.5 and section 6.6, to obtain design for wave m, we will assume that $\mathcal{X}^{m-1}$ is a finite set of input points of size N_{m-1}, and ξ to be any subset of $\mathcal{X}^{m-1}$ with fixed size n_m.

To obtain an optimal (or locally optimal) design for wave m, we adapt the simulated annealing algorithm (SAA), which has been widely used in design problems (Lark, 2002; Zimmerman, 2006; Woods, 2010; Williamson, 2015).

6.5 Application Study 1

We consider the SANDU/REF case in the SCM that was presented in Chapter 3. As part of history matching, we encounter a nonstationary response, `qv500`, water vapour at 500 metres, against five input parameters from one of the parameterization schemes developed by climate modellers from the HIGH-TUNE project. Table 6.1 provides all the necessary information such as the observation value on LES, z, together with the variance of observation error, $Var[e]$, and the variance of model discrepancy, $Var[\eta]$, in order to perform history matching (HM).

We perform two waves of HM. At wave 1 of HM, described in subsection 6.5.1, we compare the effect of employing stationary and nonstationary GP emulators in HM on the shape and size of the NROY space. However, this comparison is not the main focus of this Chapter. In subsection 6.5.2, we proceed to employ BDC in obtaining BOD to perform wave 2 of HM. We are interested in comparing the size and shape of wave 2 NROY space produced by an updated nonstationary GP emulator with BOD, "naive" design and an arbitrary design. An arbitrary design is generated by choosing points in wave 1 NROY space with implausibility values close to 3. Effectively, the majority of points from the arbitrary design are located on the borders of wave 1 NROY space. We could draw similarities between this design approach and uniform sampling in NROY space for a high-dimensional problem. In particular, it is extremely challenging to obtain input points with implausibility less than 1.5, 2 and 3 when operating with a model with a large number of input

parameters and a small size of NROY space relative to the original input space (Williamson and Vernon, 2013).

z	$Var[e]$	$Var[\eta]$	a
13.41	0.0078	0.005	3

Table 6.1: Information to perform history matching in application study 1.

6.5.1 Wave 1

We generate a maximin LHC of size 90 as our design for wave 1 HM, denoted as $X_{[1]}$, and we obtain a vector of climate model runs, $F_{[1]}$, computed at this design. We chose a space-filling design since we are interested to learn as much as possible about the model response behaviour across the whole input space $\mathcal{X}$.

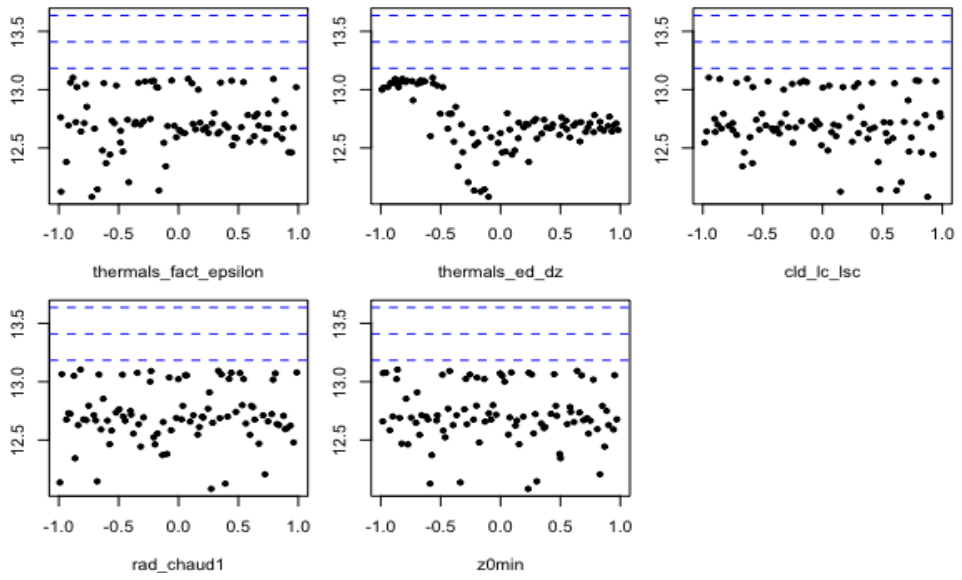

Figure 6.1: `qv500` response against five input parameters on the standardized scale. The blue dashed lines correspond to z plus and minus $2\big(Var[e] + Var[\eta]\big)^{1/2}$. The values of $z, Var[e]$ and $Var[\eta]$ are provided in Table 6.1.

From Figure 6.1, we observe the change in the response variability against the input parameter `thermals_ed_dz`. In particular, we identify three different regions of characteristic response behaviour. For `thermals_ed_dz` < -0.5, values of `qv500` are nearly constant; for $-0.5 <$ `thermals_ed_dz` < 0.5, there is an increase in the variability of response; and, for `thermals_ed_dz` > 0.5, we again observe a stabilisation in the model response. In regards to the other input parameters, we

find two levels of response behaviour, i.e. a nearly constant behaviour of `qv500` at 13.0 and more complex behaviour in the range between 12.1 and 12.8.

We use ExeterUQ software, presented in Chapter 3, to construct an "out of the box", stationary GP emulator. First, we obtain a form of regression function, $h(\mathbf{x})$, by employing a stepwise regression and following the procedure presented by Williamson et al. (2013). However, contrary to the approach adopted in Chapter 3 to specifying only parameters selected into $h(\mathbf{x})$ inside the GP covariance function, $k(\cdot, \cdot; \sigma^2, \boldsymbol{\delta}, \tau^2)$, we define all input parameters, since this form of covariance function is used for the consecutive iterations of HM. The main reason for this modification is that the input parameter that is "less active" at wave 1 could become "active" at the next iteration of HM (for details about "active" and "less active" parameters see subsection 2.3.5).

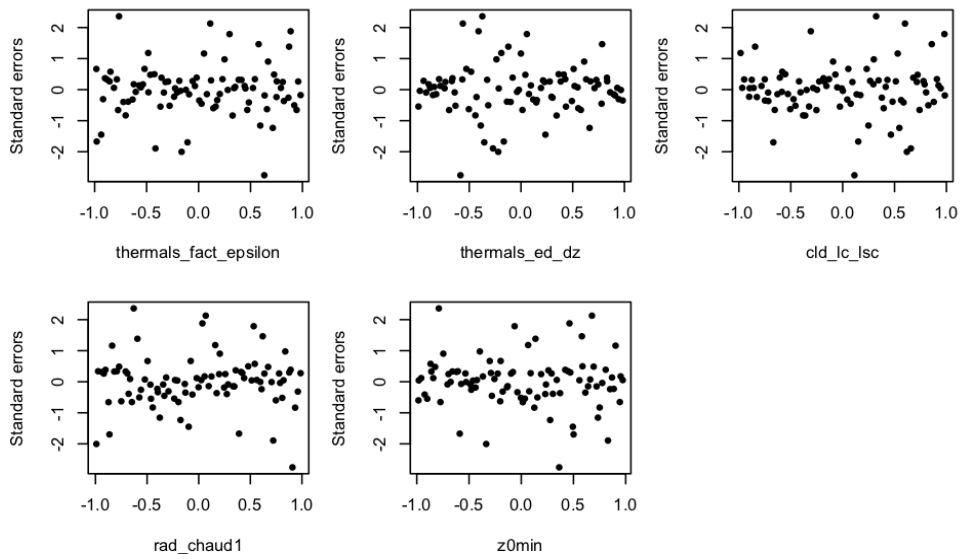

Figure 6.2: e_i (LOO standardized errors) against the input parameters.

Before performing any inference with the stationary GP emulator described above, we perform diagnostic checks such as Leave-One -Out (LOO) Cross-Validation. We obtain the LOO standardized residuals, e_i, and produce scatter plots of LOO standardized residuals against input parameters in Figure 6.2. In particular, we observe a clear relationship between the obtained errors and the input parameter `thermals_ed_dz`. Similar to the scatter plots in Figure 6.1, we could identify three distinct regions of standardized errors' behaviour. For `thermals_ed_dz` < -0.5, the

standardized errors' values are centred around zero with very low variability; while, for $-0.5 <$ `thermals_ed_dz` < 0.5, the standardized errors' variability increases, i.e. ranging from -2.5 to 2.5. In the region of input space where `thermals_ed_dz` > 0.5, the variability of the standardized errors decreases with the range of values being between -1 and 1.

Contrary to the approach presented in Chapter 4, where we specified a linear function $g(\mathbf{x})$ as part of a categorical regression in equation (4.4) and obtained the number of input regions L by considering AIC_{mod} score, we decided to define $g(\mathbf{x}) = (1, x_2)$, where x_2 corresponds to the input parameter `thermals_ed_dz`, and $L = 3$, based on our expert opinion and the standardized errors scatter plots in Figure 6.2. Figure 6.3 presents the performance of our mixture model, and we conclude that the selected form of $g(\mathbf{x})$ and $L = 3$ are good choices for a mixture model.

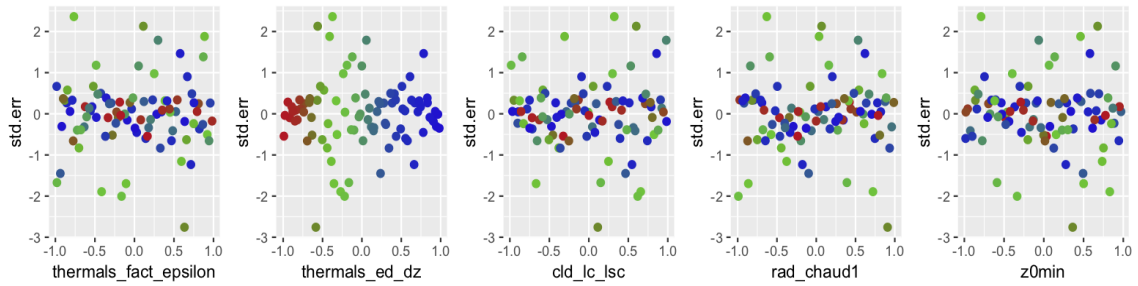

Figure 6.3: Mixture model performance: coloured e_i (LOO standardized errors) against input parameters, where the deep red colour corresponds to a higher probability of a point being allocated to region 1 (low variability region), the deep blue colour corresponds to the higher probability of a point being allocated to region 2, while the green colour corresponds to a higher probability of a point being allocated to region 3 (high variability region).

In this subsection, we perform a comparison between history matching results obtained for stationary and nonstationary GP emulators. To perform history matching we start by generating a random Latin Hypercube sample of 100,000 points in parameter space. Using a stationary GP emulator at wave 1 gives an NROY space of size of 16.22% of the original parameter space $\mathcal{X}$. By employing a nonstationary GP emulator; instead, we are able to rule out approximately 7% more of the original parameter space $\mathcal{X}$, leaving 9.19% of $\mathcal{X}$ as an NROY.

We produce the NROY density and minimum implausibility plots for 2D projections for the following parameters `thermals_fact_epsilon`, `thermals_ed_dz` and `cld_lc_lsc` in Figure 6.4. Each panel on the upper triangle shows the proportion of parameter settings behind each pixel that are NROY. Grey regions are completely ruled out. The lower triangle shows minimum implausibility plots. We plot the value of the smallest implausibility found in each pixel. For comparative purposes, the plots have the same orientation as those on the upper triangle. From NROY density plots and minimum implausibility plots in Figure 6.4, we observe that by employing a stationary GP emulator the region of the input space with small values of `thermals_ed_dz` is retained in the NROY space, due to underconfidence of the stationary GP emulator in this region. In contrast, this input region has been ruled out by our proposed nonstationary GP emulator. From NROY density plots obtained for the nonstationary GP emulator in Figure 6.4, we discover a close to linear relationship between `thermals_fact_epsilon` and `cld_lc_lsc`, and rule out the input space with high values of both of these parameters.

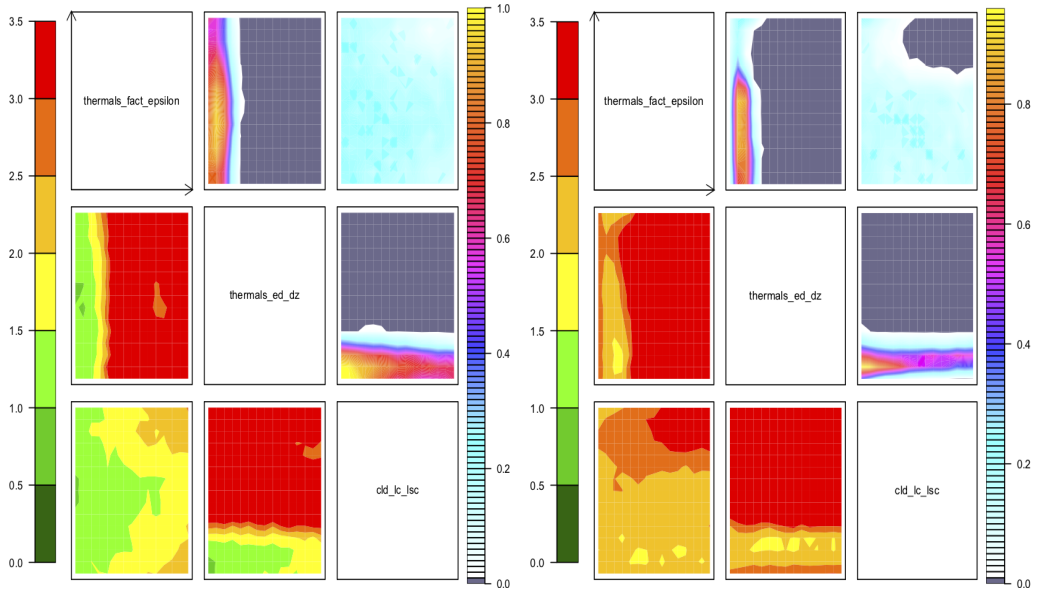

Figure 6.4: NROY density plots (*upper triangle*) and minimum implausibility plots (*lower triangle*) for `Xp` of NROY space produced by a stationary GP emulator (*on the left*) and a nonstationary GP emulator (*on the right*). Each panel plots either NROY density or minimum implausibility for a pair of parameters. NROY densities, for each pixel on any panel in the *upper triangle*, represent the proportion of points in `Xp` behind that pixel that are NROY and are indicated by the colour whose scale is indicated on the right. Minimum implausibilities, for each pixel on any panel on the *lower triangle* of the picture, represent the smallest implausibility found in `Xp`. These plots are oriented the same way as those on the *upper triangle*, for the ease of visual comparison.

We proceed to perform wave 2 HM by considering the nonstationary GP emulator only and the NROY space obtained with this emulator.

6.5.2 Wave 2

To perform wave 2 HM, we start by obtaining $\mathrm{X}_{[2]}$ in $\mathcal{X}^1$. We specify $n_2 = 82$ since only 8 design points from $\mathrm{X}_{[1]}$ are retained in $\mathcal{X}^1$, and we are interested in a high-density ensemble in $\mathcal{X}^1$ to improve the performance of the GP emulator in this reduced region. We study the effect of different design choices for wave 2 HM on the $\mathcal{X}^2$ obtained with an updated nonstationary GP emulator. In particular, we consider three design choices. "Naive" design or space-filling design (Naive Design) is generated by maximizing the minimum Euclidean distance between design points similar to the approach presented in subsection 5.5.2. The second design choice is an arbitrary design (Arbitrary Design), which is obtained by randomly

sampling n_2 points in $\mathcal{X}^1$ with implausibility values close to 3. Finally, we also obtained a Bayesian Optimal Design (BOD) by employing a simulated annealing algorithm. We specified $N = 5000$ as the number of Monte Carlo (MC) samples for $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}$ from the NROY space $\mathcal{X}^1$ and $f(\xi)^{(1)}, \ldots, f(\xi)^{(N)}$ from the distribution $\text{MVN}\big(E_{\text{F}_{[1]}}[f(\xi)], Var_{\text{F}_{[1]}}[f(\xi)]\big)$. We defined Arbitrary Design as a starting design for our optimization algorithm. We attempted to use Naive Design as a starting design for our optimization algorithm; however we failed to obtain any improvement in the BDC score given MC error.

Figure 6.5 shows the input space plots for a selection of input parameters with points classified as being in NROY after a single wave of HM in grey together with design candidates for wave 2 in blue and space-filling design for wave 1 in green. Each row in Figure 6.5 corresponds to the input space plots produced for a design candidate for wave 2 HM described above. From the input plots between `thermals_ed_dz` and `thermals_fact_epsilon` for BOD and Naive Design, we observe a number of points are placed in the corner of $\mathcal{X}^1$, i.e. in the region where the value of `thermals_ed_dz` is close to -0.5 and the value of `thermals_fact_epsilon` is close to 1. It is hard to conclude that Naive Design is space-filling across $\mathcal{X}^1$ since we observe a number of clusters of points in the input plot between `thermals_ed_dz` and `thermals_fact_epsilon` and the input plot between `cld_lc_lsc` and `thermals_ed_dz`. This could be caused by limitations in the optimization approach as well as a large number of candidate design points n_2. The allocation of points of BOD across the $\mathcal{X}^1$ vaguely resembles the allocation of points of Naive Design. In regards to Arbitrary Design, the majority of points are placed on the borders of the $\mathcal{X}^1$, in particular close to the lower bound of `thermals_ed_dz`, i.e. `thermals_ed_dz` close to -1.

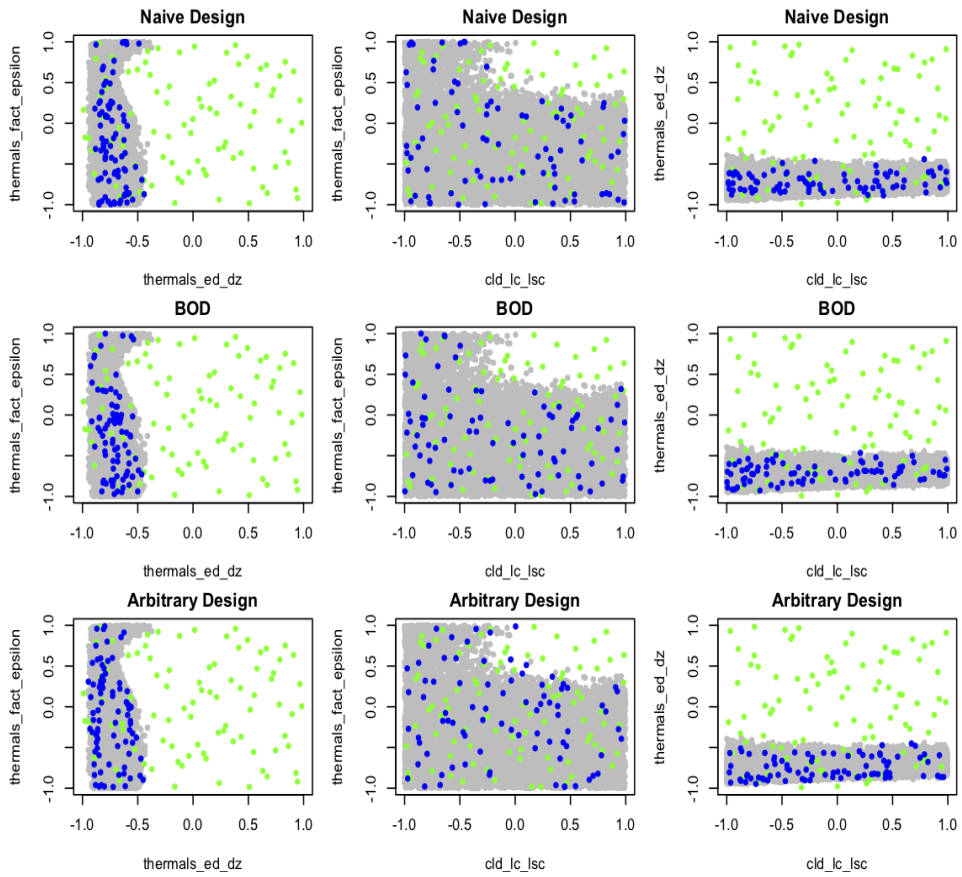

Figure 6.5: Comparison between "naive" design (Naive Design), Bayesian Optimal Design (BOD) and arbitrary design (Arbitrary Design). On each row a parameter plot shows points classified as being NROY after wave 1 of HM in grey together with design candidates for wave 2 (blue) and space-filling design for wave 1 (green).

We proceed to compute BDC at each of the design choices. From Table 6.2 and Figure 6.6, we observe that the lowest BDC score 0.162 corresponds to BOD. In general, BOD is better than Naive Design with BDC score 0.168; however, we still consider Naive Design as competitive given Monte Carlo (MC) error. The BDC at Arbitrary Design generated the largest value of the score, i.e. 0.186, which indicates that we consider Arbitrary Design as an unfavourable choice for wave 2 HM design guided by the proposed BDC.

Type	BDC	std. error	BDC-2×std.error	BDC+2×std.error
BOD	0.162	0.0024	0.157	0.167
Naive Design	0.168	0.0024	0.163	0.173
Arbitrary Design	0.186	0.0025	0.181	0.191

Table 6.2: Bayesian Design Criterion (BDC) computed at BOD, Naive Design and Arbitrary Design. The second column corresponds to the BDC score, the third column is the Monte Carlo (MC) standard error on the BDC score, the fourth and the fifth columns correspond to the BDC value plus and minus two MC standard errors.

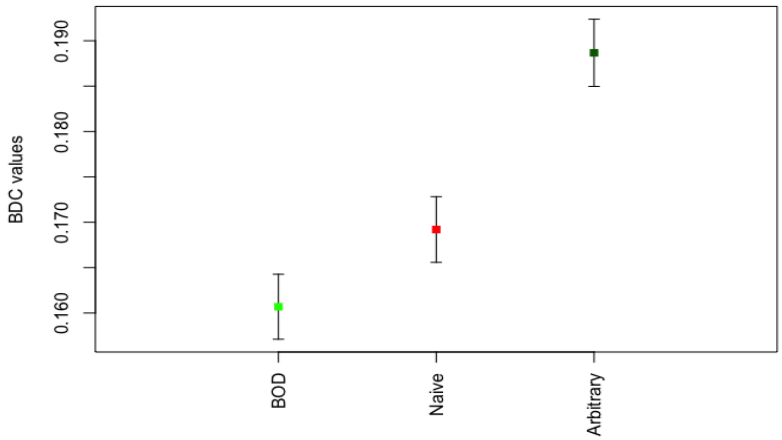

Figure 6.6: Plots of computed Bayesian Design Criterion (BDC) together with two Monte Carlo (MC) standard error bars for three design choices. We specified $N = 5000$ Monte Carlo samples in the Bayesian Design Criterion computation.

We proceed to compare the individual components of BDC for all three design choices. We start by considering the predictive variance, $Var_{\{\mathrm{F}_{[1]}, f(\xi)\}}[f(\mathbf{x})]$, computed over $\mathcal{X}^1$, where ξ corresponds to each of the design choice. Figure 6.7 demonstrates the predictive variance plots for 2D projections of parameters, `thermals_ed_dz` and `thermals_fact_epsilon`. We start by specifying the number of pixels, n_{res}, for plots. The map is shown by fixing the two parameters labelled for each pixel at the value of a pixel and exploring a 100-point Latin Hypercube in the other three dimensions of a climate model similar to the way Williamson et al. (2013) produced NROY density and implausibility plots. The value behind each

pixel is the mean value of the predictive variance obtained as

$$\frac{1}{N_{\text{res}}}\sum_{i=1}^{N_{\text{res}}} Var_{\text{F}_{[1]},f(\xi)^{(i)}}[f(\mathbf{x}^{(i)})] \quad x_1^{(i)} = x_1^l, l = 1,\ldots,n_{\text{res}}$$

$$x_2^{(i)} = x_2^m, m = 1,\ldots,n_{\text{res}},$$

where N_{res} is a number of input points inside the pixel, $N_{\text{res}} = 100$, and x_1 and x_2 correspond to the input parameters `thermals_ed_dz` and `thermals_fact_epsilon`. Red and orange coloured areas indicate higher uncertainty about model behaviour in this particular region, while white coloured areas correspond to input regions with lower predictive uncertainty. Grey regions are completely ruled out after a single wave of history matching.

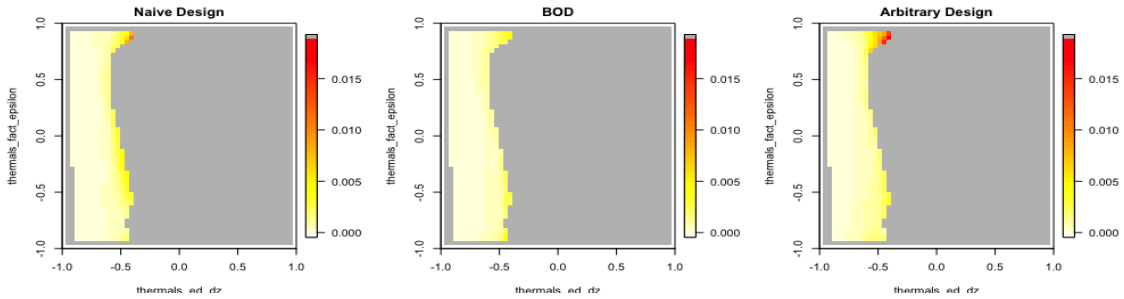

Figure 6.7: Comparison between "naive" design (Naive Design), Bayesian Optimal Design (BOD) and arbitrary design (Arbitrary Design) in the Application Study 1 for 2D projections over the NROY space. Each panel plots the predictive variance for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. The value behind each pixel on any panel represents the mean value of predictive variance found by fixing the two parameters at the plotted location and varying the other 3 dimensions of parameter space.

We observe similarities in the predictive variance representation produced for BOD and Naive Design in Figure 6.7. The largest value of predictive variance is obtained for Arbitrary Design in the corner of $\mathcal{X}^1$, where `thermals_ed_dz` is close to -0.5, and `thermals_fact_epsilon` is close to 1.0. The parameter plots for BOD and Naive Design in Figure 6.5 demonstrate that a number of points in these candidate designs are placed in this region, which, as a result, led to lower values of predictive variance.

Since we observe the difference in the predictive variance between our design options only locally, we decided to concentrate our analysis in the reduced input region,

i.e. in the range $-1 <$ `thermals_ed_dz` < 0 and $0 <$ `thermals_fact_epsilon` < 1.

Similar to a comparative study performed in section 5.5, we are interested in considering the contributions towards Term 1, $\Psi_1(\xi)$, in equation (5.13), over $\mathcal{X}^1$ for each candidate. The first column plots in Figure 6.8 demonstrate the maps produced by fixing the two parameters labelled for each pixel at the value of a pixel and exploring a 100-point Latin Hypercube in the other three dimensions. The value behind each pixel is the proportion of points that are expected to constitute wave 2 NROY space as part of BDC computation and obtained as

$$\frac{1}{N_{\text{res}}} \sum_{i=1}^{N_{\text{res}}} \Psi_1\big(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}\big), \quad x_1^{(i)} = x_1^l, l = 1, \ldots, n_{\text{res}}, \\ x_2^{(i)} = x_2^m, m = 1, \ldots, n_{\text{res}}, \tag{6.13}$$

where $\Psi_1\big(\xi; \mathbf{x}^{(i)}, f(\xi)^{(i)}\big)$ is given in equation (5.19). We consider the value behind each pixel as the contribution towards Term 1, $\Psi_1(\xi)$.

Since we employed a nonstationary GP in our BDC computation, we investigate the variability in the contributions towards $\Psi_1(\xi)$ over $\mathcal{X}^1$ determined by $\Psi_{1l}(\xi)$ with $l = 1, 2, 3$, given in equation (6.10). We consider $\Psi_{1l}(\xi)$ as the expected volume of retained input points from input space $\mathcal{X}_l$ with characteristic model response behaviour after performing two waves of history matching. In practice, the map of contributions to $\Psi_{1l}(\xi)$ for each design candidate is obtained in the same way as $\Psi_1(\xi)$, i.e. by employing the equation (6.13) with an extra condition that $\lambda_l(\mathbf{x}^{(i)}) > \lambda_j(\mathbf{x}^{(i)}), j = \{1, \ldots, L\}/l$. In the second and third column plots, the values behind each pixel correspond to the mean contributions to the values of $\Psi_{11}(\xi)$ and $\Psi_{13}(\xi)$, respectively. We do not produce plots of contributions towards $\Psi_{12}(\xi)$, since we failed to obtain any input points $\mathbf{x} \in \mathcal{X}^1$ with $\lambda_2(\mathbf{x}) > \lambda_l(\mathbf{x}), l = 1, 3$. This finding indicates that the input space $\mathcal{X}_2$ has been completely ruled out after a single wave of history matching.

In the plots in Figure 6.8, red and orange areas correspond to input regions with the highest proportion of input points that we expect to retain in wave 2 NROY space, and therefore the largest contributions towards $\Psi_1(\xi)$ or its region-specific

components, precisely $\Psi_{11}(\xi)$ and $\Psi_{13}(\xi)$. On the contrary, white areas correspond to input regions with the proportion of input points that we expect to retain in wave 2 NROY space close to zero. In the first column plots in Figure 6.8, grey regions correspond to regions that are completely ruled out after a single wave of history matching. In the second and third column plots, grey regions also correspond to input regions not considered inside the computation of region-specific components of Term 1.

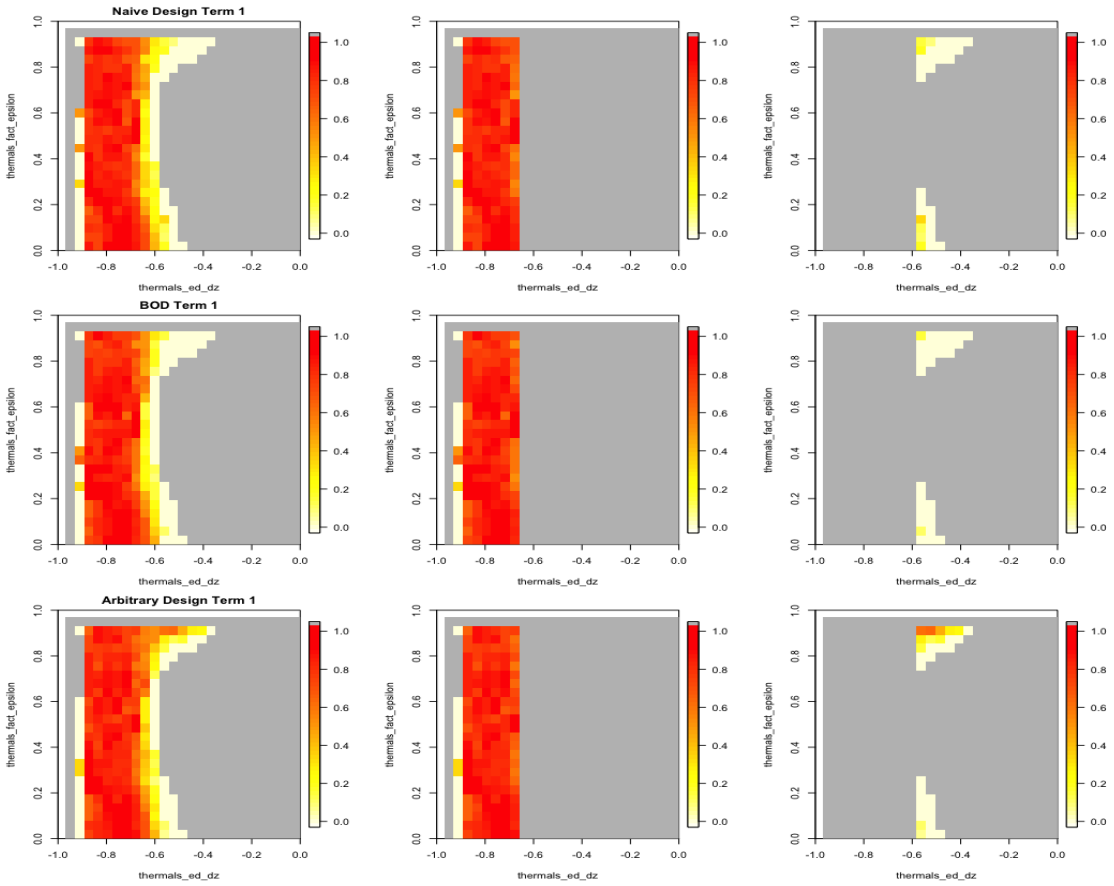

Figure 6.8: Comparison of contributions towards Term 1, $\Psi_1(\xi)$, between "naive" design (Naive Design), Bayesian Optimal Design (BOD) and arbitrary design (Arbitrary Design) in the Application Study 1 for 2D projections over the NROY space for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. *First column*: the value behind each pixel represents the proportion of points behind that pixel in the remaining 3 dimensions that are expected to remain in wave 2 NROY space. The ruled out input space is in grey. *Second column*: the value behind each pixel corresponds to the proportion of points in input region $\mathcal{X}_1^1$ that are expected to remain in wave 2 NROY space. The ruled out input space and $\mathcal{X}_3^1$ are in grey. *Third column*: the value behind each pixel corresponds to the proportion of points in input region $\mathcal{X}_3^1$ that are expected to remain in wave 2 NROY space. The ruled out input space and $\mathcal{X}_1^1$ are in grey.

From the second column plots in Figure 6.8, we observe no difference in represen-

tation of $\Psi_{11}(\xi)$ over $\mathcal{X}_1^1$ (low-variability input region) for all three design candidates. We conclude that by employing any design candidate, we expect to retain this part of the reduced input space inside wave 2 NROY since the expected values obtained at $\mathbf{x} \in \mathcal{X}_1^1$ are close to the observation z and our uncertainty about the model behaviour in this region is low, which is confirmed by plots of predictive variance in Figure 6.7. Also, from these plots, we conclude that $\mathcal{X}_1^1$ dominates the reduced input space, $\mathcal{X}^1$, and the volume of $\mathcal{X}_1^1$ is large relative to $\mathcal{X}^1$.

On the contrary, from the third column plots in Figure 6.8 we observe that the representation of $\Psi_{13}(\xi)$ over $\mathcal{X}_3^1$ (high-variability input region) obtained for Arbitrary Design is distinct. In particular, from the panel plot for Arbitrary Design, we observe orange shading in the corner of $\mathcal{X}_3^1$, with `thermals_ed_dz` close to -0.5 and `thermals_fact_epsilon` close to 1. We have a higher expectation that this part of input space will be retained inside wave 2 NROY space due to higher values of predictive variance. We also conclude that the volume of $\mathcal{X}_3^1$ is small relative to $\mathcal{X}^1$.

Similar to Term 1 plots in Figure 6.8, we produced comparative plots for Term 2 and Term 3, together with their region-specific components. However, it is hard to detect a major difference between these plots for design candidates (see Appendix C.3). Instead, we decided to compare individual terms of BDC for BOD and Arbitrary Design, since summary results provided by Table 6.2 and Figure 6.6 indicate that BOD is a better design choice than Arbitrary Design to perform wave 2 of HM for this particular example.

The top row plots in Figure 6.9 show the difference in contributions towards the individual terms of BDC over $\mathcal{X}^1$ between BOD and Arbitrary Design. The value behind each pixel corresponds to the difference in the mean contributions towards each term of BDC obtained for BOD and Arbitrary Design following the procedure in equation (6.13), and in Appendix C. The bottom row plots in Figure 6.9 demonstrate the difference in contributions to the terms of BDC between BOD and Arbitrary Design computed over the high-variability input region $\mathcal{X}_3^1$. From Figure 6.9, we conclude that the differences in contributions towards each term of BDC are mainly observed over the high-variability input region since Arbitrary Design does not contain

enough points in this region, and an updated GP emulator is uncertain about the model behaviour in this region.

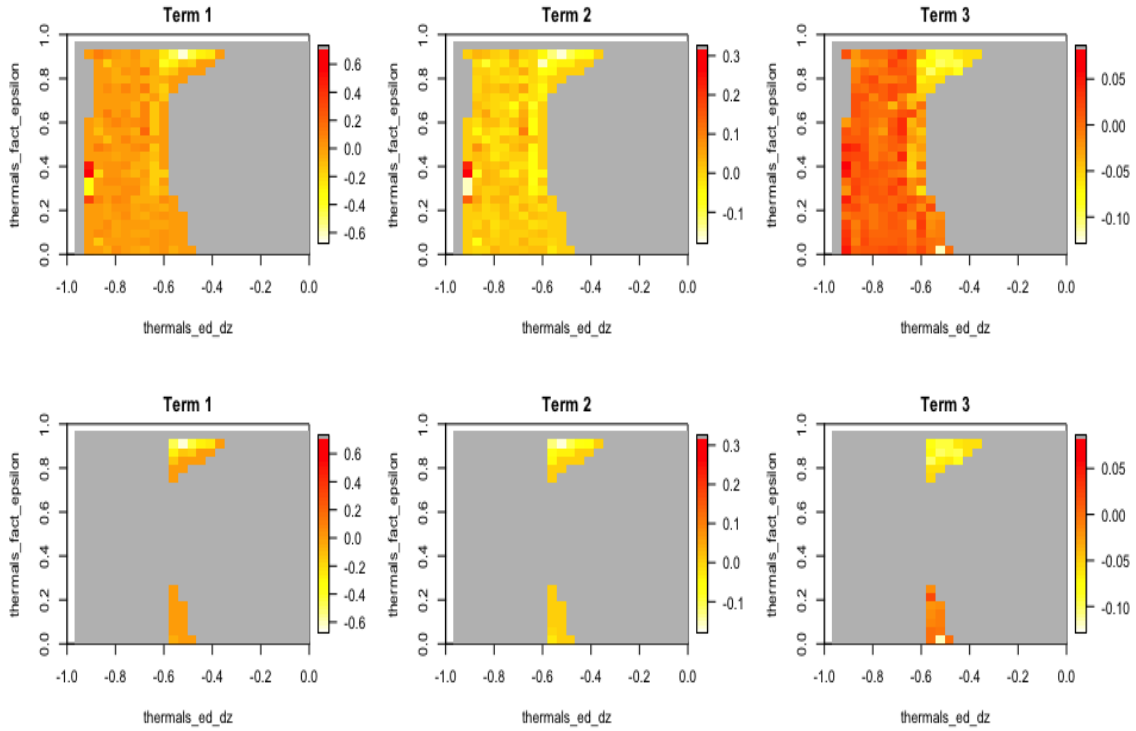

Figure 6.9: Difference in contributions towards terms of BDC between Bayesian Optimal Design (BOD) and arbitrary design (Arbitrary Design) in the Application Study 1 for 2D projections over the NROY space for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. *Top row*: the value behind each pixel represents the difference in mean contribution towards terms of BDC. The ruled out input space is in grey. *Bottom row*: the value behind each pixel represents the mean contribution towards terms of BDC computed over $\mathcal{X}_3^1$. The ruled out input space and $\mathcal{X}_1^1$ are in grey.

From the top row panel plots in Figure 6.9, we observe similarities in difference of contributions values towards Term 1 and Term 2. From the bottom row panel plots for Term 1 and Term 2, we observe a yellow ridge in the corner of $\mathcal{X}_3^1$ with `thermals_ed_dz` close to -0.5 and `thermals_fact_epsilon` close to 1. In the context of Term 2, the interpretation is that the nonstationary GP model with Arbitrary Design provides us with higher expectation that there is an overlap between NROY space at wave 2 and "true" NROY space than the nonstationary GP model with BOD. However, the scale of difference observed for Term 2 is smaller than for Term 1, which is due to the weighting determined by a difference of two CDF functions. The plot for Term 3 in Figure 6.9 demonstrates that in input region with $-0.6 <$

thermals_ed_dz < -0.4 and $0.3 <$ thermals_fact_epsilon < 1.0 we obtain lower values of contribution towards Term 3 with BOD than with Arbitrary Design. This finding indicates that we have a lower expectation that this region of input space is part of "true" NROY space with BOD than with Arbitrary Design. Therefore we obtain lower values of BDC for BOD, which indicates that it is a favourable design choice for the second iteration of history matching.

We are interested in comparing wave 2 HM results for all three design options. We start by producing the model runs, $\mathrm{F}_{[2]}$, at each design, $\mathrm{X}_{[2]}$. Similarly to the approach described in subsection 5.5.2, we update the probability distribution for $f(\mathbf{x})$ to derive $\pi_2\big(f(\mathbf{x})\big)$ and obtain the NROY space after wave 2 HM.

Table 6.3 provides the summary results of history matching for each design candidate. We obtain an NROY space for Naive Design and BOD of similar sizes. From Figure 6.10, we observe that shape of wave 2 NROY space produced with BOD resembles the one produced with Naive Design. The region of $\mathcal{X}_3^1$ previously discussed during the detailed analysis of the BDC components has been ruled out for both designs since a number of design points have been placed in this region. In contrast, this region has been retained inside wave 2 NROY space obtained with Arbitrary Design.

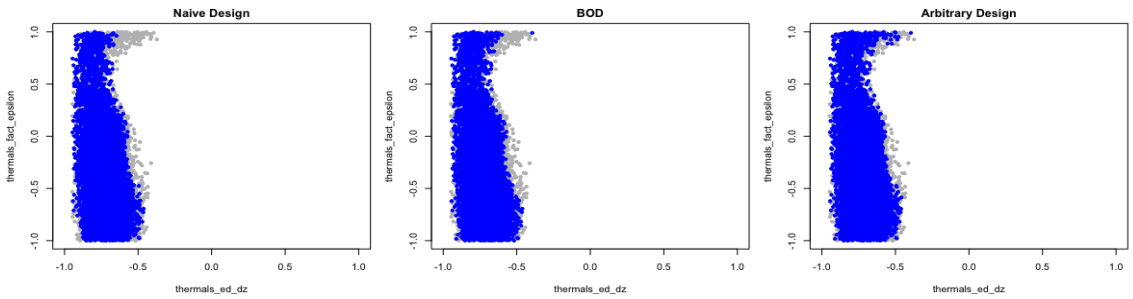

Figure 6.10: Comparison between "naive" design (Naive Design), Bayesian Optimal Design (BOD) and arbitrary design (Arbitrary Design). Each parameter plot shows points classified as being NROY after wave 1 of HM (grey) together with points classified as being NROY after wave 2 of HM (blue).

Type	NROY size
Naive Design	7.08%
BOD	7.09%
Arbitrary Design	7.49%

Table 6.3: Summary of history matching results after wave 2 for candidate designs used to update a nonstationary GP emulator.

Based on Figure 6.10 and Table 6.3, we observe similarities between wave 2 HM results obtained with BOD and Naive Design, and we conclude that for this particular application study BOD provided us with small gains compared to Naive Design.

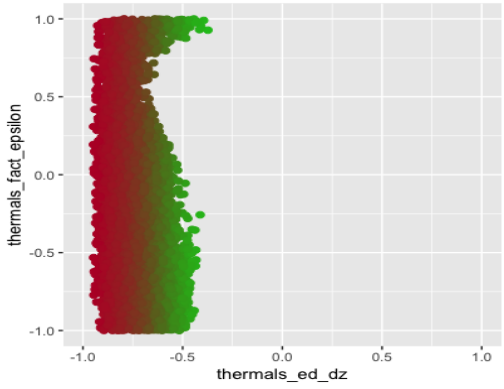

Figure 6.11: Input space plot showing $\mathcal{X}^1$, wave 1 NROY space, in Application Study 1. Points with a higher probability of being allocated to region 1 (red) and points with a higher probability of being allocated to region 3 (green).

In Figure 6.11, we produce a parameter plot of $\mathcal{X}^1$ against `thermals_ed_dz` and `thermals_fact_epsilon`. The points in green correspond to the input points with a higher probability of being in $\mathcal{X}_3^1$, while the points in red correspond to the input points with a higher probability of being in $\mathcal{X}_1^1$. We deduce that the number of retained points in $\mathcal{X}^1$ that are part of a high-variability region is considerably smaller than the number of input points allocated to a low-variability region. Therefore we expect to observe that contributions from $\Psi_{i3}(\xi)$ are smaller than $\Psi_{i1}(\xi)$ with $i = 1, 2, 3$ towards the individual terms of BDC. As a result sampling design points in the high-variability region, $\mathcal{X}_3^1$, which would be encouraged by variance-based design criteria employed with nonstationary GP models, would affect $\Psi_{i3}(\xi), i = 1, 2, 3$ terms, but because of the relatively small volume of $\mathcal{X}_3^1$, we would observe very subtle changes in the values of BDC terms. In this case, a space-filling design over $\mathcal{X}^1$ is considered to be a good choice as a design for wave 2 of history matching,

which is confirmed by BDC score.

6.6 Application Study 2

We perform a second application study with two modifications to history matching settings from Application Study 1. The first modification is that we specified an observation, z, closer to the model response. Figure 6.12 demonstrates that the model values produced by SCM are now within the error intervals around the modified value of the observation z. We note that these values are synthetic and have no connection to the LES values generated for the SANDU/REF case by climate modellers. The motivation for specifying this particular value of an observation is that we are interested in obtaining a wave 1 NROY space that contains both input points from the region with low variability in model response, $\mathcal{X}_1^1$, as well as input points from the region with high variability in model response, $\mathcal{X}_3^1$. However, contrary to Application Study 1 in section 6.5, we are interested in retaining $\mathcal{X}_3^1$ with a larger volume as part of $\mathcal{X}^1$. This particular application study is intended to investigate how the BDC values change when the size of volume $\mathcal{X}_1^1$ is close to the size of volume $\mathcal{X}_3^1$, which means that we are mainly interested in the effect of candidate design ξ on the values of $E_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$ and $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$ inside the integrand functions of individual BDC terms.

If we are not computationally or financially constrained and we could obtain a large number of computer model runs, the space-filling design could be considered as an acceptable choice for sampling design points for next iteration of refocusing which we demonstrated in section 6.5. However, in this simulation study, we decided to lower the number of design points for wave 2 of history matching by specifying $n_2 = 40$ to emulate a situation when the computer model runs are expensive to obtain. We are interested in observing if these two modifications could provide us with any different results than the ones from Application Study 1 in section 6.5. Table 6.4 provides the summary information for history matching for Application Study 2.

z	$Var[e]$	$Var[\eta]$	a
13.17	0.0078	0	3

Table 6.4: Information to perform history matching in application study 2.

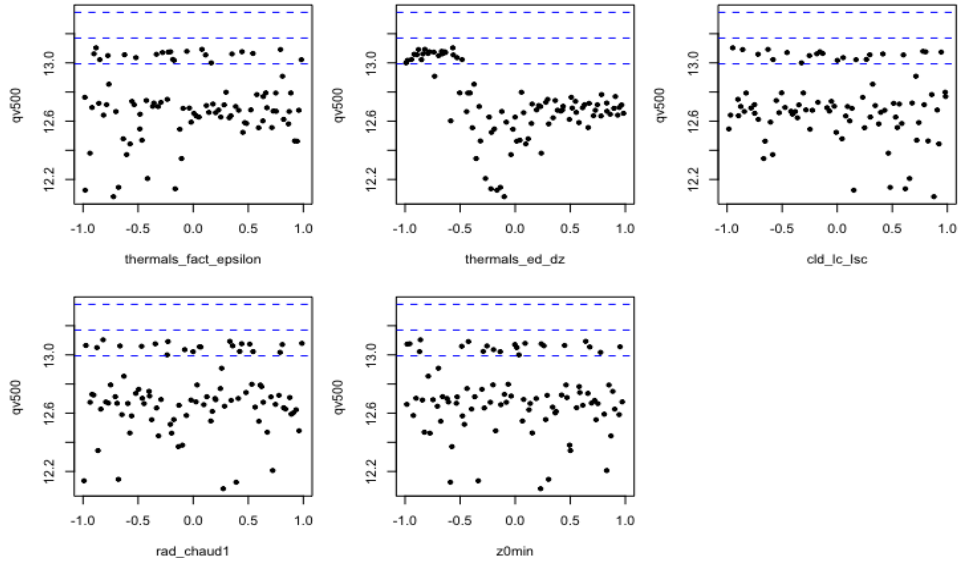

Figure 6.12: `qv500` response against five input parameters on the standardized scale. The blue dashed lines correspond to z plus and minus $2\big(Var[e] + Var[\eta]\big)^{1/2}$. The values of z, $Var[e]$ and $Var[\eta]$ are provided in Table 6.4.

6.6.1 Wave 1

We use precisely the same stationary and nonstationary GP emulators constructed during Application Study 1 in subsection 6.5.1. To perform history matching, we start by generating a random Latin Hypercube sample of 100,000 points in the parameter space. Using a stationary GP emulator at wave 1 gives an NROY space that is 34.24% of the original parameter space $\mathcal{X}$. By employing a nonstationary GP emulator; instead, we are able to rule out nearly 10% more of the original space, leaving 24.98% of $\mathcal{X}$ as NROY.

We produce the NROY density and minimum implausibility plots for 2D projections of the parameters, `thermals_fact_epsilon`, `thermals_ed_dz` and `cld_lc_lsc`. The left history matching plot of Figure 6.13 demonstrates that employing a stationary GP emulator we obtain an NROY space split into two disjoint regions of the parameter space, which is mainly caused by the emulator being underconfident in

the input region with `thermals_ed_dz` close to 1. To continue with history matching, this model response by running more waves, we would need to build separate emulators for two disjoint regions (Salter and Williamson, 2016).

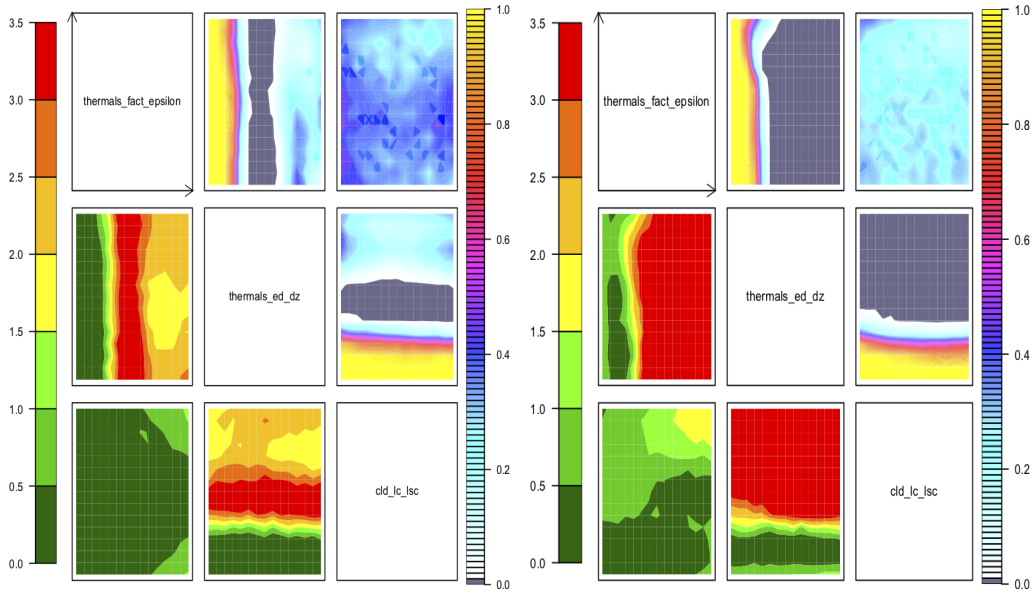

Figure 6.13: NROY density plots (*upper triangle*) and minimum implausibility plots (*lower triangle*) for `Xp` of NROY space produced by stationary GP emulator (*on the left*) and nonstationary GP emulator (*on the right*). Each panel plots either NROY density or minimum implausibility for a pair of parameters. NROY densities, for each pixel on any panel in the *upper triangle* of the picture, represent the proportion of points in `Xp` behind that pixel that are NROY and are indicated by the colour whose scale is indicated on the right. Minimum implausibilities, for each pixel on any panel on the *lower triangle* of the picture, represent the smallest implausibility found in `Xp`. These plots are oriented the same way as those on the *upper triangle*, for the ease of visual comparison.

The right history matching plot of Figure 6.13 demonstrates a single NROY region obtained with a nonstationary GP emulator. Before performing wave 2 of history matching we decided to consider wave 1 NROY space decomposition. We produced a parameter plot of $\mathcal{X}^1$ against `thermals_ed_dz` and `thermals_fact_epsilon`. The points in green are classified as points with a higher probability of being allocated to $\mathcal{X}_3^1$, whereas points in red are classified as points with a higher probability of being allocated to $\mathcal{X}_1^1$. We observe from Figure 6.14 that we managed to retain in wave 1 NROY space a larger number of input points that are considered as part of an input space with high variability in model response, $\mathcal{X}_3^1$, by modifying the value of observation, z.

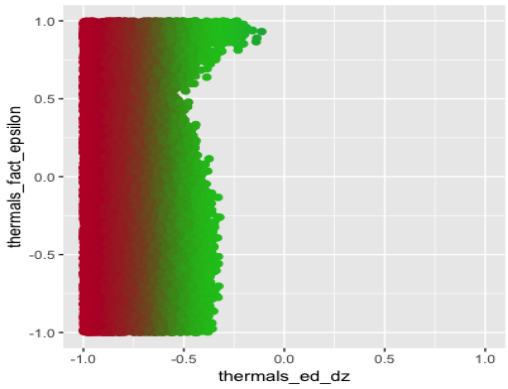

Figure 6.14: Input space plot showing $\mathcal{X}^1$, wave 1 NROY space, in Application Study 2. Points with a higher probability of being allocated to region 1 (red) and points with a higher probability of being allocated to region 3 (green).

We proceed to perform wave 2 of history matching by employing a nonstationary GP emulator and the NROY space obtained with this emulator.

6.6.2 Wave 2

To perform wave 2 HM, we need to design an ensemble in wave 1 NROY space, $\mathcal{X}^1$, of size $n_2 = 40$. Similar to Application Study 1, we are interested in studying the effect of different design choices for wave 2 of history matching on the $\mathcal{X}^2$ obtained with an updated nonstationary GP emulator with each design choice. "Naive" design or space-filling design (Naive Design) and an arbitrary design (Arbitrary Design) with randomly sampled points in $\mathcal{X}^1$ with implausibility close to 3 are generated in the same way as described in subsection 6.5.2. We attempted to obtain the BOD by employing the simulated annealing algorithm. We specified $N = 7500$ as the number of Monte Carlo samples for $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}$ from the NROY space $\mathcal{X}^1$ and $f(\xi)^{(1)}, \ldots, f(\xi)^{(N)}$ from the distribution $\text{MVN}\big(E_{\text{F}_{[1]}}[f(\xi)], Var_{\text{F}_{[1]}}[f(\xi)]\big)$ since we are operating with a larger NROY space than in section 6.5. We also used Naive Design as a starting design for our optimization algorithm. However, we failed to generate the design candidate with BDC score lower than the BDC score of Arbitrary Design due to the inefficiency of employed optimization algorithm. We refer to the obtained design after optimization as Design 1.

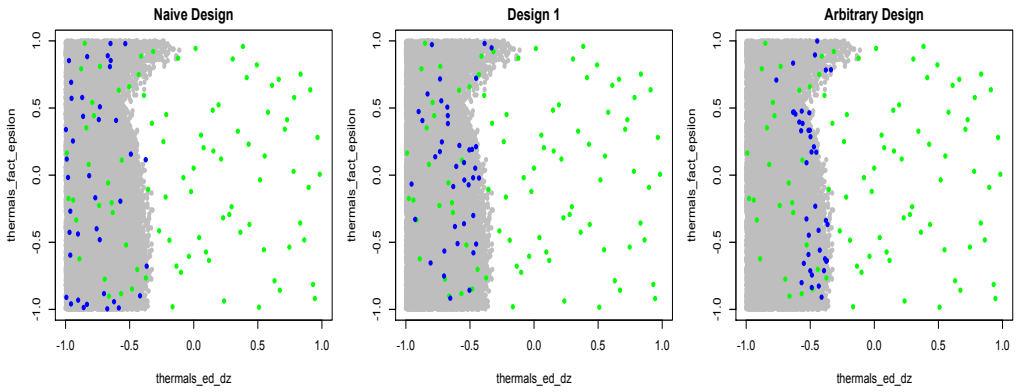

Figure 6.15: Comparison between "naive" design (Naive Design), candidate design (Design 1) and arbitrary design (Arbitrary Design). Each panel plot shows points classified as being NROY after wave 1 HM in grey together with design candidates for wave 2 (blue) and space-filling design for wave 1 (green).

Figure 6.15 demonstrates the allocation of three design choices over wave 1 NROY space and in relation to the wave 1 design. For each design choice, we produced an image of input points retained as part of NROY space after wave 1 of history matching against input parameters `thermals_ed_dz` and `thermals_fact_epsilon` in grey. Green and blue points correspond to the design used to perform wave 1 HM, $\mathrm{X}_{[1]}$, and candidate design to perform wave 2, respectively. We observe from the left panel plot in Figure 6.15 that a number of points from Naive Design are placed along the borders of the NROY space with `thermals_ed_dz` close to -1. We refer to the image of wave 1 NROY space decomposition in Figure 6.14 to deduce that the majority of input points in Naive Design are placed in the input space region with low variability in model response. The right panel plot in Figure 6.15 demonstrates the allocation of points from Arbitrary Design. We observe that the majority of these points are placed on the border of $\mathcal{X}^1$ with `thermals_ed_dz` close to -0.5. The central panel demonstrates the allocation of points from Design 1, and we observe that in general the design points are spread across $\mathcal{X}^1$, with a number of points being placed in the region with `thermals_ed_dz` close to -0.5. As before we refer back to the image in Figure 6.14 to conclude that points in BOD are placed mainly in the input region with high-variability in model response, $\mathcal{X}_3^1$, as well as in the input region where the model response behaviour changes. Meanwhile, all of the points in Arbitrary Design are in the input region with high variability in model

response.

Similar to Application Study 1, we proceed to compute the BDC at each of the design choices. From Table 6.5 and Figure 6.16, we conclude that the lowest BDC score (0.0484) is obtained for Arbitrary Design. The second best BDC score (0.0530) corresponds to Design 1, which we consider to be competitive, given MC error. The largest BDC score is obtained for Naive Design (0.0716), which indicates that this type of design is unfavourable to perform wave 2 HM for this application study. Interestingly, in this application study, Arbitrary Design is considered to be a good choice among three candidate designs to perform wave 2 HM by our proposed BDC. This result contradicts the general notion that sampling points with implausibility close to 3 is not a favourable design choice.

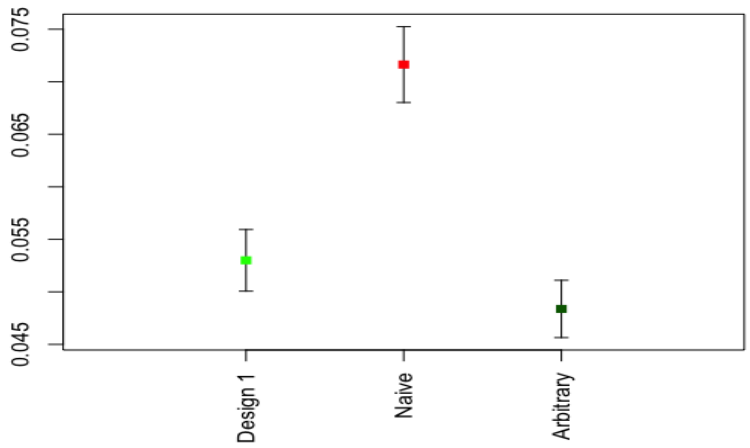

Figure 6.16: Plots of computed Bayesian Design Criterion (BDC) together with two Monte Carlo (MC) standard error bars for three design options for wave 2 of HM. We specified $N = 7500$ Monte Carlo samples in the Bayesian Design Criterion computation.

Type	BDC	std. error	BDC-2×std.error	BDC+2×std.error
Design 1	0.0530	0.0015	0.0501	0.0559
Naive Design	0.0716	0.0018	0.0680	0.0752
Arbitrary Design	0.0484	0.0014	0.0456	0.0511

Table 6.5: Bayesian Design Criterion (BDC) computed at Design 1, Naive Design and Arbitrary Design. The second column corresponds to BDC score, the third column is the Monte Carlo (MC) standard error on the BDC score, the fourth and fifth columns correspond to the BDC value plus and minus two MC standard errors.

We proceed to consider different components of BDC for all three design can-

didates. We start by generating plots of predictive variance for 2D projections of parameters `thermals_ed_dz` and `thermals_fact_epsilon` for each design choice in Figure 6.17 (see subsection 6.5.2 for more details on how to produce these plots). Red and orange coloured areas indicate of the regions of input space with high predictive uncertainty, while white coloured areas correspond to regions of input space with low predictive uncertainty. Grey regions are completely ruled out at wave 1 HM.

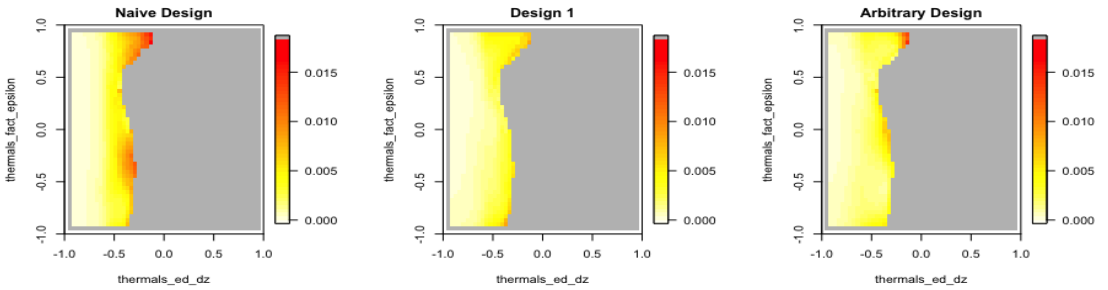

Figure 6.17: Comparison between "naive" design (Naive Design), candidate design (Design 1) and arbitrary design (Arbitrary Design) in the Application Study 2 for 2D projections over the NROY space. Each panel plots the predictive variance for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. The value behind each pixel on any panel represents the mean value of predictive variance found by fixing two parameters at the plotted location and varying the other 3 dimensions of parameter space.

From the left panel plot of Figure 6.17, we observe that the largest values of predictive variance are obtained for Naive Design on the border of the $\mathcal{X}^1$ near `thermals_ed_dz` $= -0.5$, which corresponds to the input region with high variability in model response. On the contrary, the predictive variance values obtained for Design 1 and Arbitrary Design are relatively low in this region due to a number of points in these two candidate designs being placed in this input region. Therefore we conclude that both Design 1 and Arbitrary Design provides us with the reduction in uncertainty about the model behaviour in this particular region.

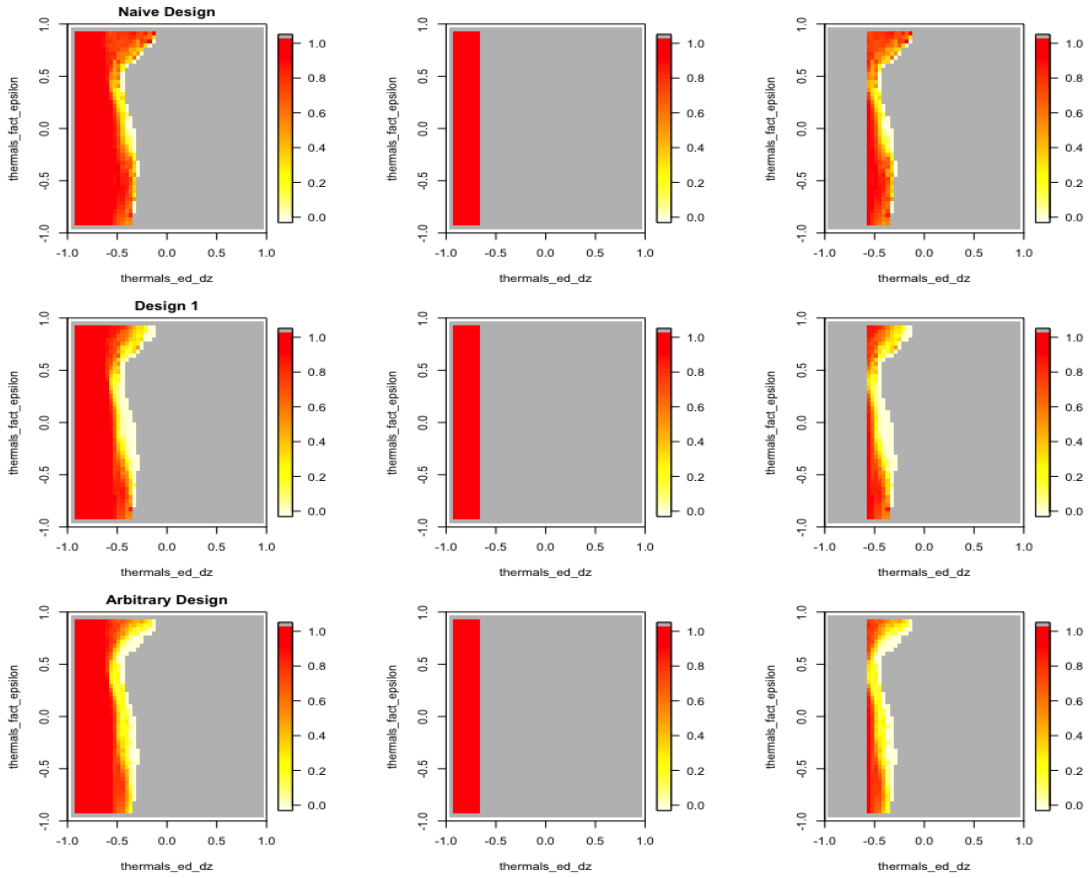

Figure 6.18: Comparison of contributions towards Term 1, $\Psi_1(\xi)$, between "naive" design (Naive Design), candidate design (Design 1) and arbitrary design (Arbitrary Design) in the Application Study 2 for 2D projections over the NROY space for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. *First column*: the value behind each pixel represents the proportion of points behind that pixel in the remaining 3 dimensions that are expected to remain in wave 2 NROY space. The ruled out input space is in grey. *Second column*: the value behind each pixel corresponds to the proportion of points in input region $\mathcal{X}_1^1$ that are expected to remain in wave 2 NROY space. The ruled out input space and $\mathcal{X}_3^1$ are in grey. *Third column*: the value behind each pixel corresponds to the proportion of points in input region $\mathcal{X}_3^1$ that are expected to remain in wave 2 NROY space. The ruled out input space and $\mathcal{X}_1^1$ are in grey.

Similar to the analysis in subsection 6.5.2, we proceed to consider the composition of contributions towards Term 1, $\Psi_1(\xi)$, over the reduced input space. The first column plot for each design candidate in Figure 6.18 shows the maps produced by fixing the two parameters labelled for each pixel at the value of a pixel and exploring 100 point Latin Hypercube in the other three dimensions. As before the value behind each pixel is the proportion of points that are expected to constitute wave 2 NROY space as part of BDC computation. Another interpretation of the value behind each pixel is the contribution towards the final value of Term 1, $\Psi_1(\xi)$. The second column plot for each design candidate in Figure 6.18 demonstrates the contributions towards $\Psi_1(\xi)$ over an input region with low variability in response, $\mathcal{X}_1^1$, determined by $\Psi_{11}(\xi)$. Meanwhile, the third column plot for each candidate design demonstrates the contributions towards $\Psi_1(\xi)$ over an input region with high variability in response, $\mathcal{X}_3^1$, determined by $\Psi_{13}(\xi)$.

In the panel plots in Figure 6.18, red and orange coloured areas correspond to input regions with the largest proportion of input points that we expect to retain as part of wave 2 NROY space, and therefore the largest contributions towards $\Psi_1(\xi)$ or its region-specific components, i.e. $\Psi_{11}(\xi)$ and $\Psi_{13}(\xi)$. White coloured areas correspond to input regions with the proportion of input points that we expect to retain as part of wave 2 NROY space close to zero. In the first column panel plots of Figure 6.18, grey regions correspond to regions that are completely ruled out after a single wave of history matching. In the second and third column panel plots of Figure 6.18, grey regions correspond to completely ruled out regions as well as the input regions in $\mathcal{X}^1$ not considered inside the computation of region-specific components of Term 1.

The second column plots in Figure 6.18 demonstrate no difference in contributions towards $\Psi_1(\xi)$ over $\mathcal{X}_1^1$ for each design candidate. We conclude that we expect to retain this part of reduced input space after the second iteration of history matching. More interesting observations appear in the third column plots in Figure 6.18. The representation of $\Psi_{13}(\xi)$ over $\mathcal{X}_3^1$ obtained for Naive Design is different from the ones obtained for Candidate 1 and Arbitrary Design. In particular, the red coloured

region at the border of $\mathcal{X}_3^1$ with `thermals_ed_dz` close to -0.5 obtained for Naive Design indicates that we have a higher expectation of retaining this input region after the second wave of history matching. We can easily draw a connection between $\Psi_{13}(\xi)$ and predictive variance values for Naive Design since we observe large values of predictive variance obtained in this region for Naive Design in Figure 6.17. Therefore, we conclude that this part of Wave 1 NROY space is expected to be retained due to persisting uncertainty about the model behaviour that is part of Term 1 calculation. In this case, our updated nonstationary GP model produces large values of predictive variance, $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$, at $\mathbf{x} \in \mathcal{X}_3^1$, which leads to the increasing contribution from these points towards the term $\Psi_{13}(\xi)$ in equation (6.10). Since in this simulation study we are operating with a larger volume of $\mathcal{X}_3^1$, we, therefore, observe a greater contribution from $\Psi_{13}(\xi)$ towards the final term $\Psi_1(\xi)$, which leads to the greater effect on the BDC score for Naive Design. On the contrary, we have a low expectation of retaining this part of the input space in wave 2 NROY space with Design 1 and Arbitrary Design. By sampling design points in this region, we obtain lower values of predictive variance, which leads to the lower contribution from $\mathbf{x} \in \mathcal{X}_3^1$ towards $\Psi_{13}(\xi)$ values and therefore towards the final value of $\Psi_1(\xi)$.

We provided the comparative plots for Term 2 and Term 3, together with their region-specific components in Appendix C.4. In this section, we compare the individual terms of BDC obtained for Arbitrary Design and Naive Design, since the lowest score of BDC is obtained for Arbitrary Design, and we treat it as our BOD. The top row plots in Figure 6.19 demonstrate the difference in contributions towards individual terms of BDC over $\mathcal{X}^1$ between Arbitrary Design and Naive Design. The value behind each pixel corresponds to the difference in mean contributions towards each term of BDC obtained for Arbitrary Design and Naive Design. The bottom row plots in Figure 6.19 demonstrate the difference in contributions to the terms of BDC between Arbitrary Design and Naive Design computed over the high-variability input region, $\mathcal{X}_3^1$, since we mainly observe the variability in the representation of individual terms of BDC computed over $\mathcal{X}_3^1$.

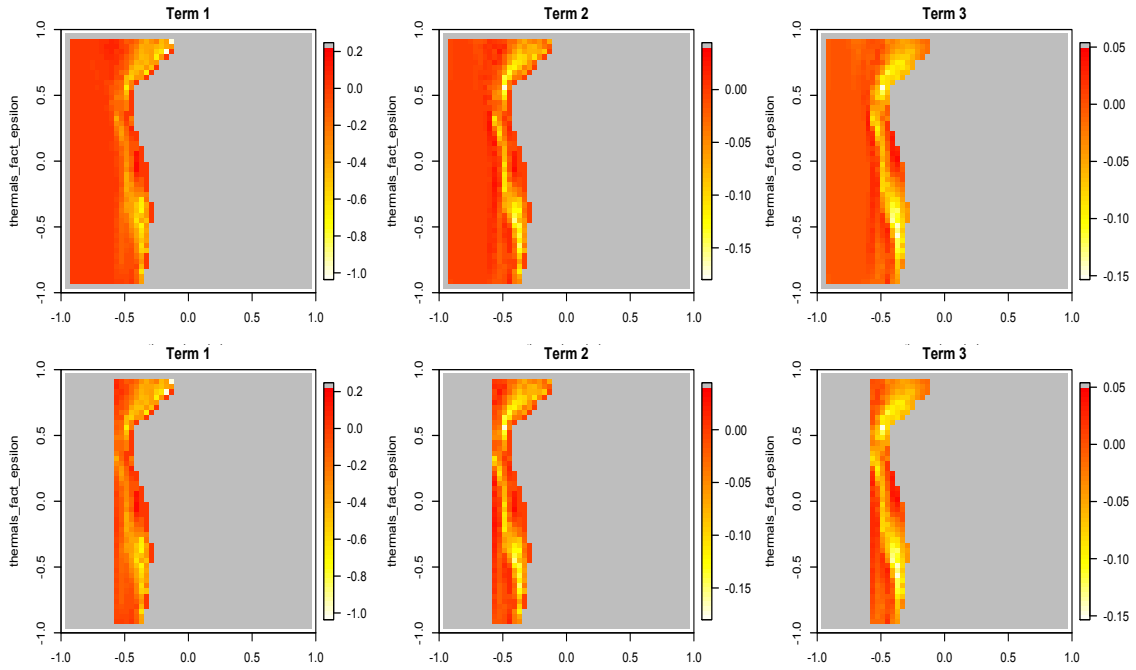

Figure 6.19: Difference in contributions towards terms of BDC between Bayesian Optimal Design (Arbitrary Design) and "naive", space-filling design (Naive Design) in the Application Study 2 for 2D projections over the NROY space for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. *Top row*: the value behind each pixel represents the difference in mean contribution towards terms of BDC. The ruled out input space is in grey. *Bottom row*: the value behind each pixel represents the mean contribution towards terms of BDC computed over $\mathcal{X}_3^1$. The ruled out input space and $\mathcal{X}_1^1$ are in grey.

From Figure 6.19, we observe similarities in the image of the difference between contributions for all three terms and their region-specific component over $\mathcal{X}_3^1$. In particular, updated nonstationary GP emulator with Naive Design produces larger values of contributions towards terms of BDC on the border of $\mathcal{X}_3^1$ with `thermals_ed_dz` close to -0.4. This is mainly caused by the prevailing uncertainty of updated nonstationary GP emulator with Naive Design about model behaviour in this region. As in subsection 6.5.2, the scale of difference observed for Term 3 and Term 2 is smaller than for Term 1.

We conclude that sampling design points in the high-variability region leads to the reduction in predictive variance, $Var_{\{\mathrm{F}_{[1]},f(\xi)\}}[f(\mathbf{x})]$, and affects the values of individual components of BDC. Since we are operating with a high-variability region of input space of a larger volume relative to $\mathcal{X}^1$ than in subsection 6.5.2, we observe an increasing effect from terms $\Psi_{13}(\xi)$, $\Psi_{23}(\xi)$ and $\Psi_{33}(\xi)$ towards the final values of

$\Psi_1(\xi), \Psi_2(\xi)$ and $\Psi_3(\xi)$ respectively.

Similar to Application Study 1 in subsection 6.5.2, we are interested in comparing the wave 2 HM results for all three design options. Table 6.6 provides the summary results of history matching for each design candidate. We obtain the smallest size of wave 2 NROY space with Arbitrary Design (22.6%), followed by Design 1 (23%). From Figure 6.20, we observe that the shape and size of $\mathcal{X}^2$ obtained with Design 1 is similar to the one obtained with Arbitrary Design. We managed to rule out the input space close to the border of $\mathcal{X}^1$ with the `thermals_ed_dz` near -0.5, which we pointed out during our description of plots in Figure 6.18 and Figure 6.19. In contrast, this input region is retained as part of $\mathcal{X}^2$ with Naive Design, since the updated GP emulator is still very uncertain about the model behaviour in this region confirmed by predictive variance representation in Figure 6.17.

Type	NROY size
Naive Design	23.85%
Design 1	23%
Arbitrary Design	22.6%

Table 6.6: Summary of history matching results after wave 2 for candidate designs used to update a nonstationary GP emulator.

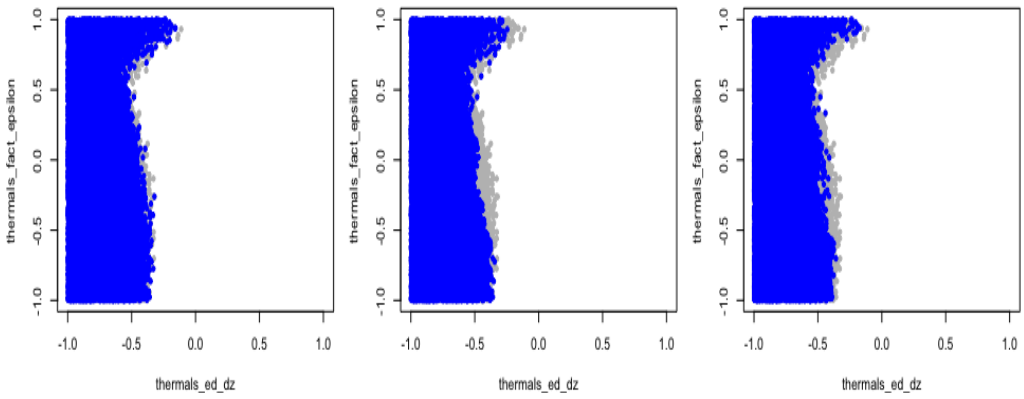

Figure 6.20: Comparison between "naive" design (Naive Design) (left panel plot), candidate design (Design 1) (central panel plot) and arbitrary design (Arbitrary Design) (right panel plot). Each parameter plot shows points classified as being NROY after wave 1 of HM (grey) together with points classified as being NROY after wave 2 of HM (blue).

6.7 Conclusion

We presented an application of our Bayesian Design Criterion (BDC) in obtaining a design for iterative refocussing with a nonstationary model response. For the first time, to our knowledge, a nonstationary GP model has been adapted and combined with a classical model-based design algorithm. To employ our proposed nonstationary GP model, we had to fix the number of input regions L. In Chapter 5, we managed to obtain three independent and interpretable terms of BDC. Employed with our proposed nonstationary GP model, we decomposed these terms further into contributions from L input regions towards these terms. This discovered feature is useful since it allowed us to incorporate the information about varying model response behaviour across the reduced input space as well as the composition of reduced input space, $\mathcal{X}^1 = \cup_{l=1}^{L} \mathcal{X}_l^1$.

We performed two simulation studies with a climate model output that provided us with different results. In particular, based on Application Study 1, we conclude that if after performing iterative refocusing we obtain an NROY space with a single dominant response behaviour, a "naive", space-filling design for the next iteration of history matching is considered to be a competitive design. This statement is supported by a low BDC score among the other candidate designs. However, in situations when an NROY space contains two or more input regions with distinct model behaviour, "naive", space-filling design may not be a good candidate for the next iteration of history matching. This particular type of design concentrates on the geometry of NROY space and does not contain any information about the variability of response. On the contrary, for both cases, our proposed BDC attempts to take into account this feature. We recognise the computational costs encountered in the process of obtaining BOD, especially when we are dealing with a large number of n_m and p, therefore we encourage our users at least to use BDC to rank their candidate designs. Our proposed criterion could provide our users with the mathematical justification for choosing a particular design for their problem under consideration, such as in Application Study 2 in Section 6.6, since we failed to obtain BOD.

A number of possible extensions could be introduced to the presented approach.

In section 6.4, we mentioned that we had fixed the values of nonstationary GP hyperparameters together with the form of the mixture model for computational reasons. However, it would be useful to consider in the future to perform an update for nonstationary GP hyperparameters that were explicitly considered by other nonstationary approaches, e.g. Montagna and Tokdar (2016).

We realise that we managed to obtain only a marginal change in the size and shape of the NROY space by employing our Bayesian Optimal Design strategy. In the future, we are planning to perform a more detailed simulation study with toy models to investigate the effect of the model behaviour together with the number of design points on the value of the BDC. We also plan to perform more waves of HM. However, we recognise that in order to perform this type of simulation study, we do require a more efficient optimization algorithm.

Chapter 7

Conclusion

In this thesis, we aimed to provide an interactive tool in Uncertainty Quantification (UQ) for modellers. We observed the failure of stationary GPs to capture nonstationary model response behaviour, which led to the development of a new type of nonstationary GP emulator. Operating with computationally complex computer models, we recognise the importance of the choosing of design points for calibration-history matching, for both stationary and nonstationary model responses.

In Chapter 3, we presented an application of `ExeterUQ`, in-house developed R and Stan-based software for modelling and quantifying uncertainties in complex computer models. We recognise that modellers, in general, require fast and robust statistical tools to assist with computer related tasks. Our tools, `ExeterUQ`, allow users to construct a range of GP emulators for a single or multiple metrics, validate the performance of obtained emulators, and perform history matching individually and independent from "experts" in statistics and uncertainty quantification. To ensure the robust performance of our developed tools, we specified prior distributions for GP emulators' hyperparameters employing Stan. The novelty of ExeterUQ is that it is an interactive tool for modellers to learn about the structural error term, η. As part of history matching, modellers are interested in investigating whether the limitations of a model's performance is caused by a poor parameterization scheme, structural error term, or a poor choice of free parameters. In particular, by employing our tools, modellers could specify the variance of model error, $Var[\eta]$, and perform history matching to observe patterns between model parameters in the

NROY space. For instance, if all space is ruled out, the model discrepancy specification has to be modified, whereas if there is a non-empty NROY space a better choice of free parameters could improve the model's performance. The importance of `ExeterUQ` is that these statistical tools coded inside `ExeterUQ` are used to assist the process-based tuning of cloud parameterization schemes. This modification is expected to improve the representation of the boundary-layer clouds in the global climate model, which affects climate model performance.

There are several possible extensions to our developed tools. We could add a number of functionalities to `ExeterUQ`: for instance, modellers are interested in performing uncertainty analysis and sensitivity analysis. Also, there is a possibility to adapt `ExeterUQ` into an `R` package and present it to a broader community of modellers.

During our close collaboration with climate modellers, we have observed a nonstationary (nonstandard) behaviour of model response across the input space. Our standard "out of the box" GP emulator, provided inside `ExeterUQ`, failed the diagnostic test, in particular Leave One Out cross-validation, i.e. we observed the heteroscedasticity in the standardised errors' behaviour against model inputs. In Chapter 4, we proposed a kernel mixture approach to the GP residual, offering a flexible and interpretable model for nonstationary response. Our method borrows the strength of input space partitioning and kernel choice to provide a single GP model that adapts to different behaviours that can lead to a continuous nonstationary GP. In section 4.3, we started by specifying a kernel mixture for our nonstationary GP emulator. We developed a mixture model for standardized residuals from a stationary GP fit to obtain mixture components inside a kernel mixture. The number of input regions, L, is found by comparing the fit of the mixture model using the modified AIC criterion, AIC_{mod}, and considering $L = 1, \ldots, 4$. The key novelty of our proposed nonstationary GP model with kernel mixture is that we employ the standard emulator diagnostics (cross-validation) to fit a nonstationary GP emulator, making it a practical tool for modellers, analysts and statisticians who are required to construct emulators quickly and efficiently.

There are several possible extensions to our developed methodology. Firstly, we mainly operate with a small number of design points, and therefore specifying the maximum number of input regions, L, to be 4 is important to ensure the convergence of posterior samples of mixture model parameters. A possible extension is to adapt the maximum number of input regions, L, to the number of design points. Another possible extension is to remove the 2 stage approach altogether by operating with joint prior distribution $\pi\big(\boldsymbol{\beta}, \boldsymbol{\sigma}_L^2, \boldsymbol{\tau}_L^2, \boldsymbol{\delta}_L, \boldsymbol{\lambda}(\mathbf{x})_L, L\big)$ to attempt full Bayesian inference. In Chapter 4, we employed a squared exponential correlation function in our prior kernel specification for the individual regions $l = 1, \ldots, L$, since it is the only form of correlation function provided by Stan. A possible extension is that we could specify different correlation functions for individual regions of input space. This modification could allow us to adapt our proposed GP emulator to model response with both inhomogeneities and discontinuities. For instance, a squared exponential correlation function could still be used as part of the kernel mixture to model inhomogeneities in response. In contrast, a neural network kernel could be used to model a jump in response (Mohammadi et al., 2019). In this case, there is the potential that a nonstationary GP model with kernel mixture could become a competitive alternative to deep GPs (Damianou and Lawrence, 2013; Roininen et al., 2018) in modelling response that exhibits both discontinuities and inhomogeneities, since deep GPs require a larger number of training points for fitting.

As part of multi-wave history matching, it is crucial to choose where in the current NROY space to sample design points to perform the next iteration of history matching, in particular if computer model runs are expensive to obtain. In Chapter 5, we propose to employ a Bayesian experimental design, where the Bayesian optimal design is found by minimizing the expected loss function. Our loss function compares the volume of NROY space obtained at the next iteration of history matching using an updated emulator for $f(\mathbf{x})$ with ensemble $\big\{\xi, f(\xi)\big\}$ to the volume of "true" NROY space obtained using a "perfect" emulator. This form of loss function is unique since it reflects our aim to cut out the implausible parameter space with fewer iteration of history matching. In section 5.3, we derived three independent and

interpretable terms of the proposed Bayesian Design Criterion (BDC). Our proposed BDC was compared to a design with good coverage over the current NROY space on a simple 2D demonstration. We recognise that Bayesian experimental design involves an expensive $p \times n_m$ dimensional optimisation problem. In practice, instead of searching for an optimal design, our proposed criterion could be used to consider and rank a range of candidate designs. Therefore, a mathematical justification could be provided for choosing a particular design for history matching based on BDC.

In Chapter 5, we fixed the values of GP hyperparameters at maximum a posteriori values (MAP) in the process of computing BDC to reduce the computational cost. This simplification effectively led our proposed approach to become pseudo-Bayesian. A possible extension is to integrate $\Psi(\xi)$ over the posterior distribution of GP hyperparameters. Another interesting question is the specification of the number of design points, n_m, to perform Wave m of history matching, as well as the division of runs across waves. In particular, in the future, we could consider how sensitive the BDC is to the number of candidate design points, as well as the inclusion of n_m inside $\Psi(\xi)$. We have only demonstrated BDC for an $f(\mathbf{x})$ with a single output (metric); therefore, a possible extension could be to adapt BDC to a vector of independent metrics, since model developers might be interested in considering a range of metrics at the same time.

In Chapter 6, we provided an extension to BDC by employing our nonstationary GP model with kernel mixture for history matching with nonstationary model response. The novelty of our proposed approach is that, for the first time, to our knowledge, a nonstationary GP model has been adapted with a classical model-based design criterion. To ensure the optimality of the obtained design, we had to fix the number of input regions L. We managed to derive that our proposed nonstationary GP model introduces an input space partition, i.e. $\mathcal{X}^{m-1} = \cup_{l=1}^{L} \mathcal{X}_l^{m-1}$, and therefore we could rewrite BDC as

$$\Psi(\xi) = \sum_{l=1}^{L} \Psi_l(\xi),$$

where the $\Psi_l(\xi)$ corresponds to the integrand function of $\Psi(\xi)$ computed over $\mathcal{X}_l^{m-1}$.

We conclude that the modified BDC does not only take into account the effect of candidate design, ξ, on $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$ and $Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$, but also the composition of NROY space $\mathcal{X}^{m-1}$. In general, variance-based design criteria, commonly used together with nonstationary GP models, encourages the sampling of design points in the input space with high variability in model response, $\mathcal{X}_L^{m-1}$. In contrast, our design criterion considers the size of the volume of this region relative to the current NROY space, $\mathcal{X}^{m-1}$. Two application studies with climate model output were performed to demonstrate the use of BDC in obtaining a design for Wave 2 of history matching. In section 6.5, the first application study indicated that in the case where the NROY space mostly consisted of a single input region with distinct model behaviour, a space-filling candidate design was a competitive choice to perform Wave 2 history matching. On the contrary, in section 6.6, we demonstrated that if an NROY space contained two or more input regions with distinct model behaviour, a space-filling candidate design was not a good choice to perform Wave 2 history matching indicated by our modified BDC.

Similar to Chapter 5, we fixed the nonstationary GP hyperparameters at maximum a posteriori values (MAP), and an obvious extension to BDC employed with our proposed nonstationary GP model is to include uncertainty in GP hyperparameters, i.e. $\Theta = \{\boldsymbol{\beta}, \Delta, \boldsymbol{\sigma}^2, \boldsymbol{\tau}^2\}$. However, applying a mixture model for standardized errors from a stationary fit as part of nonstationary GP model leads to strong assumptions about the model behaviour within the NROY space. In particular, model response behaviour across the input space in Wave m of history matching is the same as in Wave 1 of history matching. A possible solution to this problem is to remove a two-stage approach and specify a joint distribution $\pi\big(\boldsymbol{\beta}, \Delta, \boldsymbol{\sigma}^2, \boldsymbol{\tau}^2, \boldsymbol{\lambda}(\mathbf{x})\big)$. To ensure the optimality of the obtained design, we have to fix L throughout the whole process of multi-wave history matching, and an open research question is how BDC will behave if more than L regions of input space constitute NROY space.

Appendix A

Mathematical proofs

A.1 Simplification of $\Psi_1(\xi)$

We consider the argument of indicator function from equation (5.13) in detail,

$$
\begin{aligned}
-a &\leq \frac{z - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \\
z - a\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]} &\leq E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})] \\
&\leq z + a\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}.
\end{aligned}
$$

We proceed with expanding the expression for $E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]$ to derive an inequality for $f(\xi)$,

$$b_1 \leq cf(\xi) \leq b_2$$

where

$$
\begin{aligned}
b_1 = z - a\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]} - E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})] \\
+ Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x}), f(\xi)](Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)])^{-1} E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)]
\end{aligned}
$$

$$
\begin{aligned}
b_2 = z + a\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]} - E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})] \\
+ Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x}), f(\xi)](Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)])^{-1} E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)]
\end{aligned}
$$

and c is a $1 \times n_m$ vector, where n_m is a number of points in the candidate design ξ, and

$$c = Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x}), f(\xi)](Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)])^{-1}.$$

We treat $f(\xi)$ as a random variable and the distribution of $f(\xi)$ is multivariate normal:

$$f(\xi) \sim MVN\Big(E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)], Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)]\Big).$$

Based on the above, we derive a normal distribution for $cf(\xi)$ (Mardia et al., 1979, Theorem 3.1.1)

$$cf(\xi) \sim N\Big(cE_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)], cVar_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi)]c^T\Big).$$

We implement the above modifications to $\Psi_1(\xi)$, i.e.

$$\begin{aligned}
\Psi_1(\xi) &= \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\big\{b_1 \leq cf(\xi) \leq b_2\big\} \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) df(\xi) d\mathbf{x} \\
&= \int_{\mathcal{X}^{m-1}} E\big[\mathbb{1}\{b_1 \leq cf(\xi) \leq b_2\}\big] d\mathbf{x} \\
&= \int_{\mathcal{X}^{m-1}} P\big(b_1 \leq cf(\xi) \leq b_2\big) d\mathbf{x}.
\end{aligned}$$

Since we are operating with normal density function, we could use standard normal density function in our expansion to obtain

$$\Psi_1(\xi) = \int_{\mathcal{X}^{m-1}} \Big[\Phi(s_2) - \Phi(s_1)\Big] d\mathbf{x}$$

where

$$s_1 = \frac{z - a\sqrt{Var[e] + Var[\eta] + Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]} - E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]}{\sqrt{Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x}), f(\xi)]\Big(Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]\Big)^{-1} Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi), f(\mathbf{x})]}}$$

$$s_2 = \frac{z + a\sqrt{Var[e] + Var[\eta] + Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]} - E_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]}{\sqrt{Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x}), f(\xi)]\Big(Var_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\mathbf{x})]\Big)^{-1} Cov_{\langle \mathrm{F} \rangle_{[m-1]}}[f(\xi), f(\mathbf{x})]}},$$

and $\Phi(\cdot)$ is the standard normal cumulative distribution function (cdf).

A.2 Simplification of $\Psi_3(\xi)$

$$
\begin{aligned}
&\Psi_3(\xi) = \\
&= \int \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\} \pi(f(\mathbf{x})|f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\Big\{ -a \leq \frac{z - f(\mathbf{x})}{\sqrt{Var[e] + Var[\eta]}} \leq a \Big\} \pi(f(\mathbf{x})|f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \\
&\times \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\{z - a\sqrt{Var[e] + Var[\eta]} \leq f(\mathbf{x}) \leq z + a\sqrt{Var[e] + Var[\eta]}\} \\
&\times \pi(f(\mathbf{x})|f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}^{m-1}} \left[\int_{z - a\sqrt{Var[e]+Var[\eta]}}^{z + a\sqrt{Var[e]+Var[\eta]}} \pi(f(\mathbf{x})|f(\xi), \langle \mathrm{F} \rangle_{[m-1]}) df(\mathbf{x}) \right] \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) d\mathbf{x} df(\xi)
\end{aligned}
$$

Since $f(\mathbf{x}) \sim N\Big(E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})], Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]\Big)$ we use the standard normal density function in our expansion to obtain

$$
\begin{aligned}
\Psi_3(\xi) = \int \int_{\mathcal{X}^{m-1}} \Bigg[&\Phi\Bigg(\frac{z + a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \Bigg) \\
- \Phi\Bigg(&\frac{z - a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F} \rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \Bigg) \Bigg] \pi(f(\xi)|\langle \mathrm{F} \rangle_{[m-1]}) d\mathbf{x} df(\xi).
\end{aligned}
$$

A.3 Simplification of $\Psi_2(\xi)$

$$
\begin{aligned}
\Psi_2(\xi) &= \int \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\{\mathbf{x} \in \mathcal{X}^m\}\mathbb{1}\{\mathbf{x} \in \mathcal{X}_T\}\pi\big(f(\mathbf{x})|f(\xi), \langle \mathrm{F}\rangle_{[m-1]}\big) \\
&\times \pi\big(f(\xi)|\langle \mathrm{F}\rangle_{[m-1]}\big) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}^{m-1}} \int \mathbb{1}\Big\{ \frac{|z - E_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \Big\} \\
&\times \mathbb{1}\Big\{ \frac{|z - f(\mathbf{x})|}{\sqrt{Var[e] + Var[\eta]}} \leq a \Big\} \pi\big(f(\mathbf{x})|f(\xi)\langle \mathrm{F}\rangle_{[m-1]}\big)\pi\big(f(\xi)|\langle \mathrm{F}\rangle_{[m-1]}\big) df(\mathbf{x}) d\mathbf{x} df(\xi) \\
&= \int \int_{\mathcal{X}^{m-1}} \mathbb{1}\Big\{ \frac{|z - E_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \Big\} \\
&\times \left[\int \mathbb{1}\Big\{ \frac{|z - f(\mathbf{x})|}{\sqrt{Var[e] + Var[\eta]}} \leq a \Big\} \pi\big(f(\mathbf{x})|f(\xi), \langle \mathrm{F}\rangle_{[m-1]}\big) df(\mathbf{x}) \right] \\
&\times \pi\big(f(\xi)|\langle \mathrm{F}\rangle_{[m-1]}\big) d\mathbf{x} df(\xi).
\end{aligned}
$$

We use the simplification introduced in Appendix A.2 to derive the final form, i.e.

$$
\begin{aligned}
\Psi_2(\xi) &= \int \int_{\mathcal{X}^{m-1}} \mathbb{1}\Big\{ \frac{|z - E_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]|}{\sqrt{Var[e] + Var[\eta] + Var_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \leq a \Big\} \\
&\times \left[\Phi\Big(\frac{z + a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \Big) \right. \\
&\left. - \Phi\Big(\frac{z - a\sqrt{Var[e] + Var[\eta]} - E_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}{\sqrt{Var_{\{\langle \mathrm{F}\rangle_{[m-1]}, f(\xi)\}}[f(\mathbf{x})]}} \Big) \right] \\
&\times \pi(f(\xi)|\langle \mathrm{F}\rangle_{[m-1]}) d\mathbf{x} df(\xi).
\end{aligned}
$$

Appendix B

ExeterUQ software

In this appendix, we provide some of the R code together with the Stan code that are crucial for the demonstration of key functionality of ExeterUQ software presented in Chapter 3.

B.1 Organisation of ExeterUQ

We start by describing the organisation of ExeterUQ. ExeterUQ is available via version control and shared between teams at Laboratoire de Météorologie Dynamique (LMD) and Centre National de Recherches Météorologiques (CNRM) and can be applied to any climate models of Meteo France. Together with our collaborators, we developed bash scripts that allow users to obstain SCM simulations, construct GP emulators for a range of metrics of interest and perform calibration to simulations from LES for 12 selected 1D cases that cover the main boundary layer clouds regimes in real time. The focus of this chapter is the functionalities of ExeterUQ that are necessary for the automatic and robust tuning. All primary functionalities in ExeterUQ are categorized according to the directories with short description provided below.

BuildEmulator is the "heart" of ExeterUQ. This directory contains all R and .stan files that are necessary to construct and compute inferences from stationary and multivariate GP emulators. The arrows from Figure B.1 indicate that this directory "communicates" with other directories in the repository. For instance

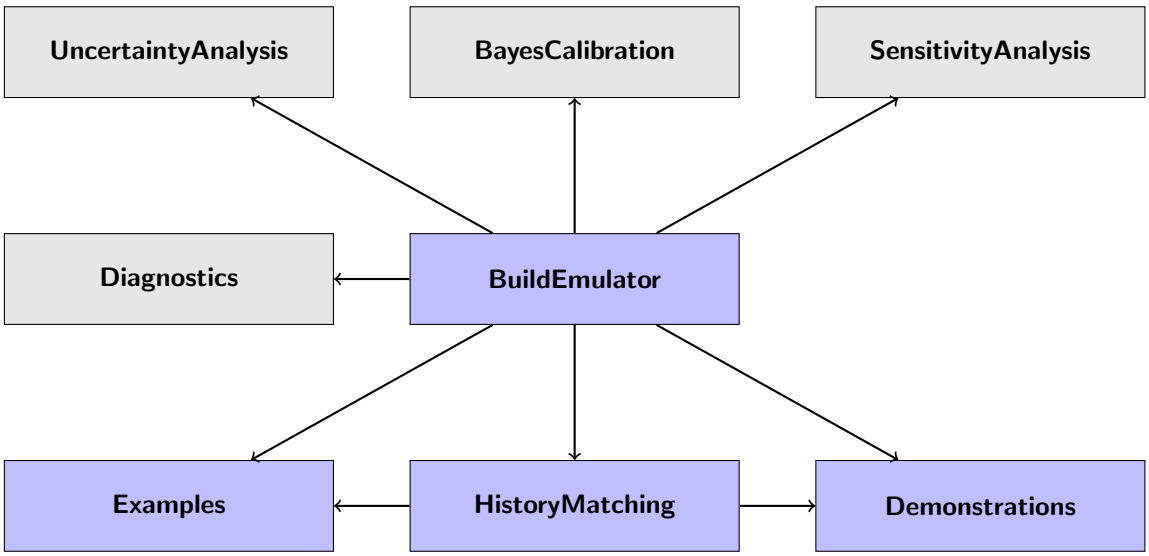

Figure B.1: Code organisation of `ExeterUQ` with nodes in *blue* corresponding to the currently available part of repository and nodes in *grey* corresponding to the parts of repository under development.

in order to perform history matching using R files from **HistoryMatching** directory, we are required to construct emulator using **BuildEmulator**.

HistoryMatching is a directory that contains R files used to perform history matching introduced in subsection 2.6.1.

Examples is a directory that contains simple examples on emulation, history matching and calibration.

Demonstrations is a directory that contains more complex examples, RData files with climate models simulations.

Diagnostics, **UncertaintyAnalysis**, **BayesCalibration** and **SensitivityAnalysis** are under development.

`ExeterUQ` has very different structure than a typical R package. R package contains the following core elements (Wickham, 2015):

DESCRIPTION: package metadata

R/: home of R code (.R files)

man/: documentation (this is where helpfiles are stored)

NAMESPACE: specifies what objects are exposed.

We have adopted a different structure for `ExeterUQ`, because our collaborators from HIGH-TUNE project, primary users of `ExeterUQ`, found our structure where each directory corresponds to the specific functionality easier to understand and relate to.

B.2 Stan programs in `ExeterUQ`

To use Stan, we are required to write a Stan program that directly computes the log-posterior density. The Stan program is written using the Stan modelling language on a separate file with a .stan extension. The result of operating with Stan program is a set of posterior simulations of the parameters in the model (Gelman et al., 2015).

We have produced two Stan programs for `ExeterUQ`, where each has its own role and functionality that we will discuss in detail. The first Stan program is written in `FitGP.stan` and is used to fit a GP model and extract a set of posterior simulations of the parameters, $\Theta = \left\{\boldsymbol{\beta}, \sigma, \boldsymbol{\delta}, \tau^2\right\}$. In particular, we specify in the `data` block of Stan program a series of n design points, $\mathrm{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, and the corresponding simulations, $\mathrm{F} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n))$ as well as the parameters for prior distribution for GP emulator hyperparameters discussed in detail in subsection **??**.

```
1  data {
2    int<lower=1> N1;      //number of ensemble members.
3    int<lower=1> pact;   //number of active inputs.
4    int<lower=1> pinact; //number of inactive inputs.
5    int<lower=1> p;//number of inputs.
6    int<lower=1> Np;//number of regression functions.
7    int SwitchDelta;//switch variable for cor length prior.
8    int SwitchNugget;//switch variable for nugget prior.
9    int SwitchSigma;//switch variable for sigma prior.
10
11 // parameters for prior specification.
```

```
  real SigSq; //parameter for sigma prior.
  real SigSqV; //parameter for sigma prior.
  real AlphaAct; //parameter for delta prior(active).
  real BetaAct; //parameter for delta prior(active).
  real AlphaInact;//parameter for delta prior(less active).
  real BetaInact; //parameter for delta prior(less active).
  real AlphaNugget;//parameter for nugget prior.
  real BetaNugget; //parameter for nugget prior.
  real AlphaRegress; // parameter for regression coef prior.
  real BetaRegress; //parameter for regression coef prior.
  real<lower=0> nuggetfix; //nugget parameter fixed.


  row_vector[p] X1[N1];   //design matrix.
  vector[N1] y1;          //vector of simulator output.
  matrix[N1, Np] H1;   //regression matrix.
}
```

We define in the `transformed data` block the variables that do not change when running the Stan program.

```
transformed data{
  real length_scale;
  length_scale = pow(sqrt(2.0), -1 );
}
```

In particular, we define an extra `length_scale` variable for the computation of covariance function, since the covariance function provided by Stan developers (cov_exp_quad) is slightly different to the commonly used squared exponential correlation function, i.e.

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\Big\{ -\frac{1}{2\rho^2} \sum_{i=1}^{p} \big(x_i - x_i'\big)^2 \Big\},$$

where ρ corresponds to the length_scale variable in the `transformed data` block,

which is equal to $\sqrt{1/2}$.

We proceed to defining parameters that are going to be sampled by Stan's samplers (HMC and NUTS) in `parametes` block, i.e. $\Theta = \left\{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}, \tau^2\right\}$

```
parameters{
 real<lower=0> nugget;
 real<lower=0> sigma;
 row_vector<lower=0>[p] delta_par;
 vector[Np] beta;
}
```

In `transformed parameters` block, we define new variables and expressions in terms of the variables declared in `parameters` block. In particular, we define Mu which corresponds to $H\boldsymbol{\beta}$ with regression matrix $H = \left[h(\mathbf{x}_1), \ldots, h(\mathbf{x}_n)\right]^T$ and Sigma which corresponds to the covariance matrix K, $n \times n$ covariance matrix with entries $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j; \sigma^2, \boldsymbol{\delta}, \tau^2)$.

```
transformed parameters{
 row_vector[p] XScaled[N1];
 vector[N1] Mu;
 matrix[N1, N1] Sigma;
 matrix[N1, N1] L;
 for(i in 1:N1) XScaled[i] = X1[i] ./ delta_par;
 //covariance matrix for design set
 Mu = H1*beta;
 Sigma = cov_exp_quad(XScaled, sigma, length_scale);
 for(k in 1:N1) Sigma[k, k] = Sigma[k, k] + nugget;
 L = cholesky_decompose(Sigma);
}
```

In the `model` block, we proceed to defining the probability of $\boldsymbol{F}$ conditioned on $\boldsymbol{X}$ and parameters Θ as Gaussian

$$\mathrm{F}|\Theta \sim MVN(H\boldsymbol{\beta}, K),$$

and priors for parameters introduced in subsection ??.

```
model{
  // Switch variable for prior specification for delta
  // '1' same prior specification for delta_par,
  //'2' separate prior specification for
  // 'active' and 'inactive' parameters.
  if(SwitchDelta == 1) delta_par ~ gamma(AlphaAct,BetaAct);
  else {
    for(i in 1:pact)
      delta_par[i] ~ gamma(AlphaAct,BetaAct);
    for(i in 1:pinact)
      delta_par[pact+i] ~ gamma(AlphaInact,BetaInact);
  }
  // Switch variable for prior specification for nugget
  // '1' concentrated prior around nuggetfix value
  // '2' inverse gamma specification for nugget.
  if(SwitchNugget == 1) nugget ~ normal(nuggetfix,0.00001);
  else nugget ~ inv_gamma(AlphaNugget, BetaNugget);
  // Switch variable for prior specification for sigma
  // '1' original prior specification
  // '2' lognormal prior specification
  if(SwitchSigma == 1) sigma ~ normal(SigSq, SigSqV);
  else sigma ~ lognormal(SigSq, SigSqV);
  if(Np > 1) beta[2:Np] ~ normal(AlphaRegress,BetaRegress);
  y1 ~ multi_normal_cholesky(Mu, L);
}
```

Since `ExeterUQ` software is extensively used within the research group in Uncertainty Quantification at the University of Exeter, we were interested to provide our researchers with a flexibility in their prior specification by introducing a switch parameter. In particular, when SwitchDelta, SwitchNugget and SwitchSigma are set at

value 1, we are operating with the default, "out of the box" prior specifications for GP hyperparameters used as part of HIGH-TUNE. However, if the corresponding switch parameters are set to some other arbitrary values, our researchers could use and experiment with their own prior specifications.

We proceed to describe the content of the second Stan program `PredictGen.stan`, which is used to produce predictions of GP emulator at unseen input $\mathbf{x}$, i.e. posterior mean and posterior variance values.

In this case, we use `data` block not only to specify a design matrix and the vector with the corresponding simulator output evaluations as well a validation matrix, a matrix of input parameters at which we are interested to produce predictions, and a set of posterior simulations of parameters, $(\boldsymbol{\beta}_i, \sigma_i, \boldsymbol{\delta}_i, \tau_i^2), i = 1, \ldots, M$.

```
data {
  int<lower=1> N1; //number of members in design.
  int<lower=1> N2; //number of members in validation.
  int<lower=1> p; //number of inputs.
  int<lower=1> M; //number of posterior draws.
  int<lower=1> Np; //number of regression functions.

  row_vector[p] X1[N1]; //design matrix.
  row_vector[p] X2[N2]; //validation matrix.
  vector[N1] y1; //vector of simulator output at design.
  matrix[N1, Np] H1; //regression matrix for design.
  matrix[N2, Np] H2; //regression matrix for validation.

  real<lower=0> sigma[M];
  real<lower=0> nugget[M];
  row_vector[Np] beta[M];
  row_vector[p] delta[M];
}
transformed data {
```

```
  real length_scale;
  length_scale = pow(sqrt(2.0), -1);
}
```

Contrary to `FitGP.stan`, we do not specify any variables in `parameters` block and we define an empty `model` block, as we do not intend to use any of the Stan's samplers (HMC and NUTS) to obtain the posterior predictive distribution for $f(\mathbf{x})$. Instead, we reuse the posterior samples for GP hyperparameters $\Theta = \{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\delta}, \tau^2\}$, i.e. the samples from the set of posterior simulations of the parameters, $(\boldsymbol{\beta}_i, \sigma_i, \boldsymbol{\delta}_i, \tau_i^2), i = 1, \ldots, M$, are used to compute at a new input $\mathbf{x}$, $m_i^*(\mathbf{x}) = E[f(\mathbf{x})|\{\mathrm{X}, \mathrm{F}\}, \boldsymbol{\beta}_i, \sigma_i, \boldsymbol{\delta}_i, \tau_i^2]$ and $C_i^*(\mathbf{x}, \mathbf{x}) = Cov[f(\mathbf{x}), f(\mathbf{x})|\{\mathrm{X}, \mathrm{F}\}, \boldsymbol{\beta}_i, \sigma_i, \boldsymbol{\delta}_i, \tau_i^2]$ defined in equations (2.6) and (2.7). We simulate $f^{(i)}(\mathbf{x}) \sim N(m_i^*(\mathbf{x}), C_i^*(\mathbf{x}, \mathbf{x})), i = 1, \ldots, M$, in order to obtain the distribution of points and compute the summary of this distribution, i.e.

$$\texttt{Expectation} = \frac{1}{M}\sum_{i=1}^{M} f^{(i)}, \quad \texttt{StandardDev} = \sqrt{\frac{1}{M-1}\sum_{i=1}^{M}(f^{(i)}(\mathbf{x}) - \texttt{Expectation})^2}.$$

```
model {
}
generated quantities {
  matrix[M, N2] predict_y;
  vector[N2] tmeans;
  vector[N2] tsds;
  for(m in 1:M) {
    row_vector[p] XS1[N1];
    row_vector[p] XS2[N2];
    matrix[N1, N1] A1;
    matrix[N2, N1] A2;
    matrix[N1, N1] inver;
    vector[N1] e;

```

```
    // scaled design matrix by cor length parameters.
    for(i in 1:N1) XS1[i] = X1[i] ./ delta[m];
    // scaled validation matrix by cor length parameters.
    for(i in 1:N2) XS2[i] = X2[i] ./ delta[m];
    // covariance matrix for design set.
    A1 = cov_exp_quad(XS1, sigma[m], length_scale);
    for(i in 1:N1) A1[i, i] = A1[i, i] + nugget[m];
    // inverse of covariance matrix for design set.
    inver = inverse(A1);
    // covariance matrix for validation set.
    A2 = cov_exp_quad(XS2, XS1, sigma[m], length_scale);
    // error vector.
    e = y1 - H1*to_vector(beta[m]);
    for(n in 1:N2) {
      real mu;
      real sigma_squared;
      // calculate mean
      mu = H2[n, ]*to_vector(beta[m]) + A2[n, ]*inver*e;
      // calculate variance
      sigma_squared = pow(sigma[m], 2.0) +
                      nugget[m] - A2[n, ]*inver*A2[n, ];
      predict_y[m, n] = normal_rng(mu, sqrt(sigma_squared));
    }
  }
  for(k in 1:N2) {
    tmeans[k] = mean(predict_y[, k]);
    tsds[k] = sd(predict_y[, k]);
  }
}
```

The main reason for obtaining these two summaries of posterior predictive distri-

bution of $f(\mathbf{x})$ is that `Expectation` and `StandardDev` are necessary for computing Implausibility function $\mathcal{I}(\mathbf{x})$ defined in equation (2.31) and as the result to perform history matching.

In general, the workflow of operating with Stan programs consist of compiling Stan program to C++ and then running it in R along with specified $\boldsymbol{X}$ and $\boldsymbol{F}$. Our collaborators tend to construct a number of GP emulators during one R session and compiling Stan program every single time they attempt to fit GP emulator seemed to be inefficient. We decided to save pre-compiled Stan programs, since pre-compiled Stan programs could be run immediately when called, avoiding compilation time.

The chunk of Rscript `BuildEmulator.R` demonstrates that when we open a new R session, we start by compiling and saving the pre-written static .stan files. The function `stanc` translates the Stan program to C++ code. We use `stan_model` function for C++ codes defined in `ccdode_fit` and `ccode_predict` plus other auxiliary code to be compiled into a dynamic shared object (DSO) and loaded. The loaded DSO, pre-compiled Stan programs, used to fit Gaussian Process (GP) and produce predictions are stored in `model_fit` and `model_predict` respectively and could be used to draw samples allowing inference to be performed for the model and data.

```
1  twd <- getwd()
2  tfile_loc = paste(twd,
3    "/BuildEmulator/FitGP.stan", sep = "")
4  tprednewfile_loc = paste(twd,
5    "/BuildEmulator/PredictGen.stan", sep = "")
6
7  ccode_fit <- stanc(file = tfile_loc)
8  model_fit <-stan_model(stanc_ret = ccode_fit)
9
10 ccode_predict <- stanc(file = tprednewfile_loc)
11 model_predict <- stan_model(stanc_ret = ccode_predict)
```

We note that our users have to re-compile Stan programs when they start a new R session or after implementing changes to `.stan` files such as changing the

form of priors for hyperparameters of GP model. However, we do not expect our primary users, collaborators from HIGH-TUNE, to introduce any changes to our Stan programs.

B.3 Constructing a GP emulator

In this section of Appendix, we aim to provide a step by step guide to constructing a GP emulator for a single metric referring to the example considered in Section 3.5. We start by defining a data frame `tData`, that contains input parameters `thermals_fact_epsilon`, `thermals_ed_dz`, `cld_lc_lsc` and `cld_tau_lsc`, and output/outputs of complex computer model, that we are attempting to model. The metric of interest, `theta500`, a potential temperature at 500 metres, is the last column of `tData` data frame. We also include an extra input variable `Noise`, which is a vector of realisations from normal distribution with mean, $\mu = 0$, and standard deviation, $\sigma = 0.5$, which importance in the process of deriving the form of regression function, $h(\cdot)$. Here is the first six rows of a generated data frame.

```
> head(tData)
  thermals_fact_epsilon thermals_ed_dz cld_lc_lsc cld_tau_lsc        Noise theta500
1            0.76934741     -0.4937967 -0.2427105  0.07549189 -0.005897532 306.0406
2           -0.03033414     -0.8150176 -0.2755129  0.68226792  0.024237992 306.2965
3            0.51600179     -0.3291489 -0.6693219  0.93619972 -0.042996625 306.1178
4            0.64953037      0.6158412  0.5485222 -0.10797432  0.038916278 305.9214
5           -0.47335511     -0.3900361  0.4158181  0.77471939 -0.067888845 306.3947
6            0.45442011     -0.9757470  0.2901544 -0.31185709  0.006014265 306.1657
> dim(tData)
[1] 90  6
```

The first two extensions contained in `tData`, 60 member LHC, are used as a design for our GP emulator, while the last extensions, defined as `tData.valid`, is used to validate the performance of GP emulator.

```
1 tData.valid <- tData[61:90, ]
2 tData <- tData[1:60, ]
```

B.3.1 Constructing the mean function

The process of constructing a GP emulator starts with obtaining a form of the mean function $h(\mathbf{x})$.

```
cands <- c("thermals_fact_epsilon", "thermals_ed_dz",
           "cld_lc_lsc", "cld_tau_lsc", "Noise")
myem.lm <- EMULATE.lm(Response="theta500",
 tData=tData, tcands=cands,tcanfacs=NULL,TryFouriers=TRUE,
 maxOrder=2,maxdf = 20)
```

The function `EMULATE.lm` allows our users to derive a more complex form of mean function by using a stepwise regression procedure (Draper and Smith, 1998). The same procedure for obtaining a mean function has been used by Williamson et al. (2013); Salter and Williamson (2016). Interestingly, input variable `Noise` is considered as one of the potential predictors and if it is selected by stepwise algorithm, we add no further terms. Effectively, we consider `Noise` as a stopping criteria for fitting a mean function of emulator. The role of `Noise` is discussed in detail in Salter and Williamson (2016).

There is a number of arguments of function `EMULATE.lm` that specify and control the form of input parameters that could be added to $h(\mathbf{x})$. For instance, `TryFouriers` is a logical argument with default `FALSE`. With `TryFouriers=TRUE` the function considers the Fourier transformation of input parameters added as potential terms to $h(\mathbf{x})$. The argument `maxOrder` corresponds to the maximum order of Fourier transformation for input parameters. Effectively we allow function to consider $j \leq$`maxOrder` for $\sin(j \times \pi \times x_i)$ and $\cos(j \times \pi \times x_i)$ with x_i as one of the input parameters, as potential terms for linear function of emulator. Lastly, `maxdf` corresponds to the maximum number of degrees of freedom that we are allowed to use for our regression model. In particular, we could add at greatest `maxdf` terms, predictors, to regression function with `maxdf`+1 parameters to estimate.

The output of `EMULATE.lm` is a list with the elements that are crucial in constructing full GP model.

```
> names(myem.lm)
```

```
 [1] "Names"              "linModel"
"mainEffects"         "Interactions"
 [5] "Factors"            "FactorInteractions"
 "ThreeWayInters"      "Fouriers"
 [9] "pre.Lists"          "DataString"
"ResponseString"
```

In particular, `Names` is a vector of strings corresponding to the candidates whose main effects are included in the final linear model and are considered inside the residual term of GP emulator. Another important part of output generated by `EMULATE.lm` is `linModel`. `linModel` is an object of class 'lm'. For instance, the function `summary` could be applied to print out the summary of the results.

B.3.2 Constructing a full GP emulator

After obtaining the form of regression function, $h(\mathbf{x})$, we are ready to construct a full GP emulator. We pass an obtained linear model to function's argument `meanResponse`.

```
myem.gp <- EMULATE.gpstan(meanResponse=myem.lm,
tData=tData, FastVersion=TRUE, additionalVariables=NULL)
```

Function `EMULATE.gpstan` contains the loaded DSO, pre-compiled Stan program stored in `model_fit` object that we discussed in subsection B.2. Inside function `EMULATE.gpstan` we use function `sampling` that calls sampler 'NUTS' (No-U-Turn samples) (Hoffman and Gelman, 2011; Betancourt 2017) to draw posterior samples from the model defined by `model_fit`. We pass Stan number of cores to estimate on (2) and number of iterations (2000 by default) and number of warmup iterations per chain (1000). We use multiple chains to make sure that the posterior distribution that we converge on is stable, and is not affected by starting values. A logical scalar argument, `FastVersion`, defaulting to `TRUE`, indicates whether to include the GP hyperparameters' values fixed at maximum a posteriori values (MAP) and some of the components of GP emulator for fast inference. If `FALSE`, these components are

not saved and used for inference.

We could pass to `additionalVariables` a vector of character strings specifying the input parameters that we would like to be considered in the model for residual term (covariance function). The default is `NULL` indicating that only main effects in the linear component are considered inside the residual term.

For pedagogical purposes we have considered how to construct linear component and GP component of surrogate model step by step using functions `EMULATE.lm` and `EMULATE.gpstan` respectively. In fact, our collaborators from `HIGH-TUNE` could use function `BuildStanEmulator` to generate full GP emulator in one go which is based on `EMULATE.lm` and `EMULATE.gpstan` functions.

B.3.3 Output generated by `EMULATE.gpstan`

`EMULATE.gp` returns an object `StanEmulator`. An object `StanEmulator` is a list containing the following elements and we consider some of these elements in detail.

```
> names(myem.gp)
 [1] "Names"              "linModel"
 [3] "mainEffects"        "Interactions"
 [5] "Factors"            "FactorInteractions"
 [7] "ThreeWayInters"     "Fouriers"
 [9] "pre.Lists"          "DataString"
[11] "ResponseString"     "Design"
[13] "tF"                 "H"
[15] "ParameterSamples"   "FastParts"
[17] "StanModel"          "prior.params"
[19] "init.list"
```

`ParameterSamples` is a list with elements `sigma`, `delta_par`, `beta` and `nugget`, where each element of the list contains the posterior samples of size 2000 (combining samples generated by each chain after warmup) for the corresponding parameter.

`FastParts` is the list with the following elements `tMAP`, `A`, `QA`, `Ldiff`. `tMAP` corresponds to the index of posterior samples at which we obtain MAP (maximum a posteriori) value. `A` is a matrix of covariances evaluated at design points, $K(\boldsymbol{X}, \boldsymbol{X})$. `QA` is a Cholesky decomposition of matrix `A`. `Ldiff` corresponds to the vector of computed $(QA)^{-1}(F - H\boldsymbol{\beta})$, a part of posterior mean that could be computed offline and used for fast inference.

`StanModel` is a `stanfit` object that is saved for diagnostics purposes, such as to assess the convergence of chains using `traceplot` function demonstrated in Figure B.2.

```
1 traceplot(myem.gp$StanModel)
```

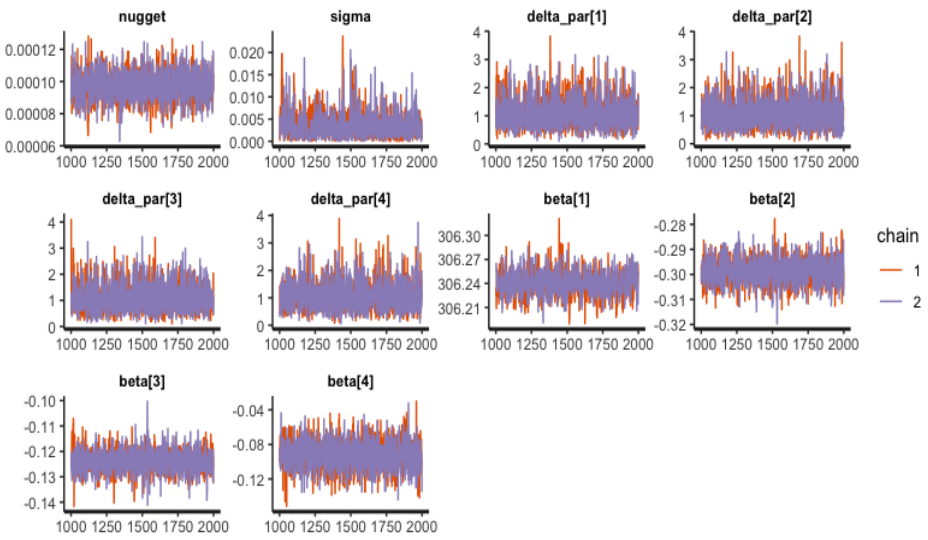

Figure B.2: Trace plots obtained from `myem.gp$StanModel`.

B.3.4 GP emulator for multivariate computer model output

We define a `tData`, a data frame which contains normalized input parameters values `thermals_fact_epsilon`, `thermals_ed_dz`, `cld_lc_lsc` and `cld_tau_lsc`, and four metrics of interest that we attempt to emulate, i.e. `theta500`, `qv500`, `zhneb` and `nebmax`.

We construct GP emulator for each metric of interest using the following code:

```
myem.lm =InitialBasisEmulators(tData=tData,
HowManyEmulators=4, additionalVariables = NULL)
```

which generates a list, where each element of the list contains an emulator object generated using a function `BuildStanEmulator` for each metric. These objects could be further used for diagnostics and calibration.

B.4 Diagnostics for GP emulators

In this section of Appendix, we describe the functions provided inside `ExeterUQ` software to perform diagnostics (validate) a GP emulator.

B.4.1 Leave One Out (LOO) diagnostics

The code provided in this part of Appendix corresponds to the diagnostics performed in subsection 3.6. In particular, we could generate Leave One Out (LOO) diagnostics plot using the following code

```
tLOOs = LOO.plot(StanEmulator = myem.gp,
                 ParamNames=names(tData)[1:nparam],
                 ResponseName = "theta500")
```

where `StanEmulator` corresponds to the object previously generated by two functions `EMULATE.lm` and `EMULATE.gp`, `ParamNames` is a string vector of input parameter namings and `ResponseName` corresponds to the response variable name.

We generate a data frame of size n, employing the procedure described in subsection 3.6.1, with columns `posterior mean` corresponds to $E\big[f(\mathbf{x}_j)|\{\boldsymbol{X}_{-j}, \boldsymbol{F}_{-j}\}\big]$ values and `lower quantile` and `upper quantile` correspond to

$$E\big[f(\mathbf{x}_j)|\{\boldsymbol{X}_{-j}, \boldsymbol{F}_{-j}\}\big] \pm 2 \times \sqrt{Var\big[f(\mathbf{x}_j)|\{\boldsymbol{X}_{-j}, \boldsymbol{F}_{-j}\}\big]} \quad j = 1, \ldots, n.$$

The first six rows of the obtained data frame is provided below.

```
> head(tLOOs)
```

```
  posterior mean lower quantile upper quantile
1        306.0410        306.0180        306.0640
2        306.3035        306.2793        306.3277
3        306.1059        306.0809        306.1308
4        305.9206        305.8969        305.9444
5        306.4025        306.3794        306.4255
6        306.1614        306.1364        306.1864
```

Additionally we could obtain the standardised errors from LOO diagnostics for example in subsection 3.6.1 by using the following R function:

```
std.error = CalStError(tLOOs, tData$theta500)
```

B.4.2 Validation plots for GP emulators

In the Appendix B.3, we defined a data frame `tData.valid` that was not used for constructing GP emulator and instead was used for validation, i.e. to check the predictive power of produced GP emulator on unobserved points.

```
Valid = ValidationStan(NewData = tData.valid,
                       Emulator = myem.gp, main = "")
```

`tData.valid` corresponds to the data frame of unobserved parameter values together with the model response. `tData.valid` has the same format as the `tData`. We pass `StanObject` to `Emulator` argument of a function. `main` is a string that could be used to produce an overall title of the diagnostic plot. This code has been used to produce diagnostic plots in subsection 3.6.2.

In order to generate predictions, `Expectation` and `StandardDev`, for each input point in `tData.valid` we use the pre-compiled Stan program object, `model_predict`, described in subsection B.2. We generate a data frame with columns `posterior mean` corresponding to `Expectation` values and `lower quantile` and `upper quantile` corresponding to `Expectation`$-2\times$`StandardDev` and `Expectation`$+2\times$`StandardDev` respectively. The first six rows of the obtained data frame is provided below.

```
> head(Valid)
  posterior mean lower quantile upper quantile
1        306.2373        306.2144        306.2602
2        306.2919        306.2689        306.3149
3        306.2001        306.1776        306.2225
4        306.0642        306.0404        306.0880
5        306.4464        306.4210        306.4718
6        305.8652        305.8413        305.8892
```

B.4.3 Validation Summary Statistics

The predictive power of produced GP emulator could be assessed using Root Mean Squared Error (RMSE) by calling `RMSE.Fun` function, where we pass the data frame generate by function `ValidationStan` to argument `pred` and a vector of true model values to argument `y.true`.

```
> RMSE.Fun(pred = Valid, y.true = tData.valid$theta500)
[1] 0.01243866
```

We also compute Interval Score (IS) by calling `S.Int.score` function, where argument `alpha` corresponds to the α parameter specified by user to compute equation (3.1).

```
> S.Int.score(pred = Valid, y.true = tData.valid$theta500,
alpha = 0.05)
[1] 0.05945927
```

B.5 Code to perform calibration in `ExeterUQ` software

In this section of Appendix, we provide the R code used to perform history matching described in Section 3.7.

B.5.1 History Matching

We provide a code used to perform Wave 1 HM by considering a single metric `theta500` in subsection 3.7.

We start by generating 10,000 points random LHC, stored in data frame `Xp`, to represent the parameter space, $\mathcal{X}$, at which we are going to compute the implausibility function defined previously.

```
Xp <- as.data.frame(2*randomLHS(10000, nparam)-1)
names(Xp) <- names(tData)[1:nparam]
```

In order to evaluate implausibility function, $\mathcal{I}(\mathbf{x})$, at each member of `Xp`, we use the `UniImplausibilityStan` function and we pass the values of z, $Var[\eta]$ and $Var[e]$ to `Obs`, `Discrepancy` and `ObsErr` respectively. For this demonstration we specify `Obs` $= 305.82$, `ObsErr` $= 0.0218$ and `Discrepancy` $= 0$. The argument `FastVersion` determines how we compute an expectation, $E[f(\mathbf{x})]$, and variance, $Var[f(\mathbf{x})]$, from emulator. If `FastVersion=FALSE`, we use pre-compiled Stan program `model_predict` described in subsection B.2 to obtain these values by using a whole set of posterior samples for GP hyperparameters. On the contrary, if `FastVersion=TRUE`, we fix GP hyperparameters at MAP values to obtain expectation and variance.

```
Timps <- UniImplausibilityStan(NewData=Xp,
Emulator=myem.gp, Discrepancy=Disc, Obs=tObs,
ObsErr=tObsErr, FastVersion = TRUE)
```

The output of `UniImplausibilityStan` function is a vector of length 10,000 of computed implausibility function values at the data frame `Xp`.

```
> head(Timps)
[1] 2.752236 3.230644 2.960496 4.973720 3.488240 2.110816
```

We are interested in visualizing the results of history matching by obtaining the implausibility plots. Firstly, we need to generate implausibility lists using function `CreateImpList`. For function `CreateImpList` we need to specify `ImpData`, a data

frame that contains the input points, `Xp`, and the corresponding values of implausibility function evaluated at `Xp`. `Cutoff` is a value that corresponds to the threshold, a, so that any value of $\mathcal{I}(\mathbf{x}) > a$ is deemed implausible. For this demonstration we specified `Cutoff` $= 3$ following the 3 sigma rule (Pukelsheim, 1994).

```
ImpData <- cbind(Xp,Timps)
Cutoff <- 3
VarNames <- names(Xp)
ImpList <- CreateImpList(whichVars = 1:nparam,
VarNames = VarNames,  ImpData = ImpData,
Resolution = c(15,15), whichMax=1, Cutoff = Cutoff)
```

In our demonstration the function `ImpList` generates a list of size 3:

`ImpList[[1]]` is a list of size 3 which is used to produce NROY density and minimum implausibility plots for the input parameter `thermals_fact_epsilon` against other three input parameters, i.e. `thermals_ed_dz`, `cld_lc_lsc` and `cld_tau_lsc`. Each member of the list `ImpList[[1]]` contains a matrix of dimensions 2×225, where the first row corresponds to the proportion of NROY space of `Xp` at each pixel and the second row corresponds to the minimum implausibility of `Xp` at each pixel.

`ImpList[[2]]` is a list of size 2 which is used to produce NROY density and minimum implausibility plots for the input parameter `thermals_ed_dz` against `cld_lc_lsc` and `cld_tau_lsc`.

`ImpList[[3]]` is a list of size 1 which is used to produce NROY density and minimum implausibility plots for the input parameter `cld_lc_lsc` against `cld_tau_lsc`.

Finally, we are ready to produce NROY density and minimum implausibility plots demonstrated in Figure 3.8, using function `imp.layoutm11`.

```
imp.layoutm11(ImpList,VarNames,newPDF=FALSE,
the.title=paste("InputSpace_wave",WAVEN,".pdf",sep=""),
```

```
newPNG=FALSE,newJPEG=FALSE,newEPS=FALSE,Points=NULL)
```

Setting one of the logical arguments `newPDF`, `newPNG`, `newJPEG`, `newEPS` equal to `TRUE` automatically generates NROY density and minimum implausibility plots in the corresponding image processing format.

B.5.2 Computation of multi-dimensional implausibility

In the same way as in subsection 3.7.1, we start by generating 10,000 points random LHC, stored in data frame `Xp`, to represent the parameter space, $\mathcal{X}$, at which we are going to compute the implausibility function, $\mathcal{I}_i(\mathbf{x}), i = 1, \ldots, 4$ for each of the metrics.

```
Xp <- as.data.frame(2*randomLHS(10000, nparam)-1)
names(Xp) <- names(tData)[1:nparam]
```

In order to compute implausibility function, $\mathcal{I}_i(\mathbf{x}), i = 1, \ldots, 4$, for each metric at each member of `Xp`, we use the `ManyImplausibilitiesStan` function. The function uses `StanEmulator` objects stored in the `myem.lm` to obtain expectation, $E[f_i(\mathbf{x})]$, and variance, $Var[f_i(\mathbf{x})]$, for each metric of interest. Then we proceed to computing implausibility function for each metric individually. We pass vectors of values of observations, variance of observation errors and variances of model discrepancy to arguments `Obs`, `tObsErr` and `Discrepancy` respectively.

```
> Disc
[1] 0 0 0 0
> tObs
[1] 3.058203e+02 1.618378e+01 1.521915e+03 9.296613e-02
> tObsErr
[1] 2.185544e-02 4.917087e-02 1.729015e+03 1.880364e-04
```

```
TMimpls<- ManyImplausibilitiesStan(NewData=Xp,
Emulator=myem.lm, Discrepancy=Disc, Obs=tObs,
ObsErr=tObsErr, FastVersion = TRUE)
```

```
Impdata=cbind(Xp,TMimpls)
VarNames=names(Xp)
```

The output of `ManyImplausibilitiesStan` function is a matrix of dimension $10,000 \times 4$ where each column corresponds to $\mathcal{I}_i(\mathbf{x}), i = 1, \ldots, 4$.

In order to visualize the NROY density plots and minimum implausibility plots for 2-D projections of NROY space we are required to use `CreateImpList` function to generate `ImpList` object, which is necessary for the image construction. We need to pass the number of metrics that we are considering to `nEms` argument of the function. The argument `whichMax` determines the form of implausibility measure that we are considering, i.e. `whichMax=1` corresponds to $\mathcal{I}_M$.

```
ImpList = CreateImpList(whichVars = 1:nparam,
VarNames=VarNames, ImpData=Impdata, nEms=nmetrique,
whichMax=1, Cutoff = Cutoff)
```

```
NROY1=which(rowSums(TMimpls<=Cutoff)>=nmetrique)
Xp2 = Xp[NROY1, ]
```

In order to produce the second highest implausibility we use the same function `CreateImpList`, but in this case we specify `whichMax=2`.

```
ImpList = CreateImpList(whichVars = 1:nparam,
VarNames=VarNames, ImpData=Impdata, nEms=nmetrique,
whichMax=2, Cutoff = Cutoff)
```

```
NROY1=which(rowSums(TMimpls<=Cutoff)>=nmetrique-1)
Xp2 <- Xp[NROY1, ]
```

B.5.3 Performing iterative refocussing in `ExeterUQ` software

In general, we follow the following steps to perform refocussing with `ExeterUQ` software:

1. An initial sample of size n is taken in parameter space using a K-extended Latin Hypercube design (Williamson, 2015) to obtain $\mathrm{X}_{[1]}$. SCM model runs are produced at the design values to obtain metrics of interest, giving $\mathrm{F}_{[1]}$. We define a list `EMULATOR.LIST` to store `StanEmulator` objects generated at each step of multi-wave calibration. We also define a vector, `Cutoff_vec`, to store a value of a, `Cutoff`, and a vector, `tau_vec`, to store a value of `tau`, where both values could be adjusted at each step of multi-wave calibration. The value of `tau` is important in specifying the form of implausibility function, i.e. we derive $\texttt{valmax} = \texttt{tau} + 1$, which we pass to argument `whichMax` of calibration functions.

2. Construct Gaussian Process emulators using `InitialBasisEmulators` and perform Leave-One-Out diagnostics to check the performance of obtained emulators. At wave k we save a collection of `StanEmulator` objects in a list `EMULATOR.LIST` with entry k. We also save the pre-specified a (threshold value) as the k^{th} entry of vector `Cutoff_vec` and an integer j, that determines the j^{th} maximum implausibility function, as the k^{th} entry of vector `tau_vec`. The modified objects, i.e. `EMULATOR.LIST`, `Cutoff_vec` and `tau_vec` are saved in RData file, `EMULATOR_LIST_MULT_METRIC.RData`.

3. Start with loading `EMULATOR_LIST_MULT_METRIC.RData` file. Generate a random LHC sample of 10,000 points in parameter space at which we are going to compute implausibilities. We are required to use `StanEmulator` objects from previous waves, as we note that $\mathbf{x}$ is nonimplausible at wave k only if it is nonimplausible for all the waves that precede it. We sample n from the current NROY space , $\mathcal{X}^{k-1}$, using random sampling instead of the entire parameter space to create a new design and obtain a new ensemble, $\left\{\mathrm{X}_{[k]}, \mathrm{F}_{[k]}\right\}$.

Appendix C

Bayesian Design Computation (BDC) plots

In this part of the appendix, we produce plots of representation of Term 2, $\Psi_2(\xi)$, and Term 3, $\Psi_3(\xi)$, obtained as part of Bayesian Design Criterion (BDC) computation.

C.1 BOD for Wave 1 of history matching

We are interested in picturing the contribution to Term2, $\Psi_2(\xi)$, over the original input space, $\mathcal{X}$. To produce these plots, we start by specifying a number of equally spaced intervals for x_1 and x_2, i.e. n_{res}, which determines the resolution, the number of pixels of obtained images. The value behind each pixel represents the proportion of points that are expected to be part of wave 1 NROY space and "true" NROY space and computed as

$$\frac{1}{N1}\sum_{i=1}^{N1}\frac{1}{N_{res}}\sum_{k=1}^{N_{res}}\left[\Phi\left(\frac{z+a\sqrt{Var[e]+Var[\eta]}-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}{\sqrt{Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\right)\right.$$
$$\left.-\Phi\left(\frac{z-a\sqrt{Var[e]+Var[\eta]}-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}{\sqrt{Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\right)\right]$$
$$\times\mathbb{1}\left\{\frac{|z-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]|}{\sqrt{Var[e]+Var[\eta]+Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\leq a\right\},$$
$$x_1^l\leq x_1^{(k)}\leq x_1^{l+1}, l=1,\ldots,n_{res}-1,$$
$$x_2^r\leq x_2^{(k)}\leq x_2^{r+1}, r=1,\ldots,n_{res}-1.$$

Red regions correspond to the input space where we expect to obtain the largest contribution towards the final value of Term 2. In contrast, white and yellow regions correspond to the input space with very limited contribution towards the final value of Term 2.

Similar steps are performed to depict the contributions to Term 3, $\Psi_3(\xi)$, over the original input space, $\mathcal{X}$. In this case, the value behind each pixel is the mean contribution to the value of Term 3 computed via

$$\frac{1}{N1}\sum_{i=1}^{N1}\frac{1}{N_{res}}\sum_{k=1}^{N_{res}}\left[\Phi\left(\frac{z+a\sqrt{Var[e]+Var[\eta]}-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}{\sqrt{Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\right)\right.$$
$$\left.-\Phi\left(\frac{z-a\sqrt{Var[e]+Var[\eta]}-E[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}{\sqrt{Var[f(\mathbf{x}^{(k)})|f(\xi)^{(i)},\Theta_i^{MAP}]}}\right)\right],$$
$$x_1^l \leq x_1^{(k)} \leq x_1^{l+1}, l = 1,\ldots,n_{res}-1,$$
$$x_2^r \leq x_2^{(k)} \leq x_2^{r+1}, r = 1,\ldots,n_{res}-1.$$

Red regions correspond to the input regions where we expect to obtain the largest contribution towards the final value of Term 3, effectively we have a high expectation that this region is part of "true" NROY space. On the contrary, white and yellow regions correspond to the input space with a limited contribution towards the final value of Term 3, effectively we have lower expectation that these regions are part of "true" NROY space.

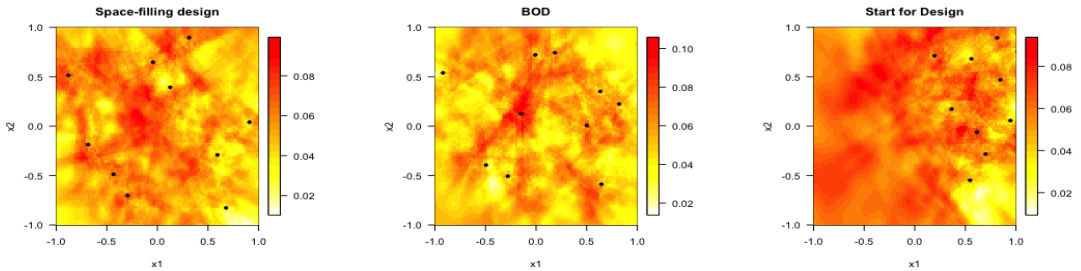

Figure C.1: The integrand of Term 2, $\Psi_2(\xi)$, computed over the input space, $\mathcal{X}$. Each pixel of plots represents the mean value of the integrand of Term 2, $\Psi_2(\xi)$, computed at input settings behind each pixel. Different design options are depicted as black points.

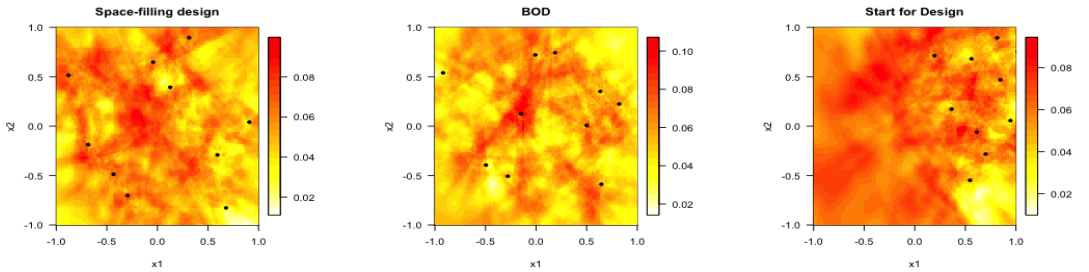

Figure C.2: The integrand of Term 3, $\Psi_3(\xi)$, computed over the input space, $\mathcal{X}$. Each pixel of plots represents the mean value of the integrand of Term 3, $\Psi_3(\xi)$, computed at input settings behind each pixel. Different design options are depicted as black points.

C.2 Bayesian Optimal Design and the Nonstationary GP model

We are interested in depicting the contribution to Term 2, $\Psi_2(\xi)$, over $\mathcal{X}^1$. We start by specifying the number of pixels n_{res} and generate parameter plots by fixing two parameters labelled for each pixel at the value of a pixel and exploring a 100 point Latin Hypercube in the other dimensions. The value behind each pixel is the mean contribution to the value of Term 2 computed via

$$\frac{1}{N_{\text{res}}}\sum_{i=1}^{N_{\text{res}}}\Psi_2\big(\xi;\mathbf{x}^{(i)},f(\xi)^{(i)}\big) \quad x_1^{(i)}=x_1^l, l=1,\ldots,n_{\text{res}}$$
$$x_2^{(i)}=x_2^m, m=1,\ldots,n_{\text{res}},$$

where $\Psi_2\big(\xi;\mathbf{x}^{(i)},f(\xi)^{(i)}\big)$ is provided in equation (5.20). Red region corresponds to the input space where we expect to obtain the largest contribution towards the final value of Term 2, while white and yellow region corresponds to the input space with a very limited contribution towards the final value of Term 2. We consider red region as the region of the input space where we expect to observe an overlap between wave 2 NROY space with and "true" NROY space as part of BDC computation.

To obtain region-specific contributions towards Term 2, i.e. $\Psi_{2l}(\xi)$ with $l=1,\ldots,L$, the procedure described above is performed with an extra condition that $\lambda_l(\mathbf{x}^{(i)})>\lambda_j(\mathbf{x}^{(i)}), j=\{1,\cdots,L\}/l$.

Similar steps are performed to depict the contribution to Term 3, $\Psi_3(\xi)$, over $\mathcal{X}^1$. In this case, the value behind each pixel is the mean contribution to the value of Term 3 computed via

$$\frac{1}{N_{\text{res}}}\sum_{i=1}^{N_{\text{res}}}\Psi_3\big(\xi;\mathbf{x}^{(i)},f(\xi)^{(i)}\big) \quad x_1^{(i)}=x_1^l, l=1,\ldots,n_{\text{res}}$$
$$x_2^{(i)}=x_2^m, m=1,\ldots,n_{\text{res}},$$

where $\Psi_3(\xi,\mathbf{x}^{(i)},f(\xi)^{(i)})$ is provided in equation (5.21). Red region corresponds to the input regions where we expect to obtain the largest contribution to the final value of Term 3, effectively we have a high expectation that this region is part of "true" NROY space. On the contrary, white and yellow regions correspond to the input space with a limited contribution towards the final value of Term 3, effectively we have lower expectation that these regions are part of "true" NROY space.

To obtain region-specific contributions towards Term 3, i.e. $\Psi_{3l}(\xi)$ with $l=1,\ldots,L$, the procedure described above is performed with an extra condition that $\lambda_l(\mathbf{x}^{(i)})>\lambda_j(\mathbf{x}^{(i)}), j=\{1,\ldots,L\}/l$.

C.3 Application Study 1

We provide representation of Term 2 and Term 3 over the reduced input space, $\mathcal{X}^1$, for Application Study 1, considered in Section 6.5.

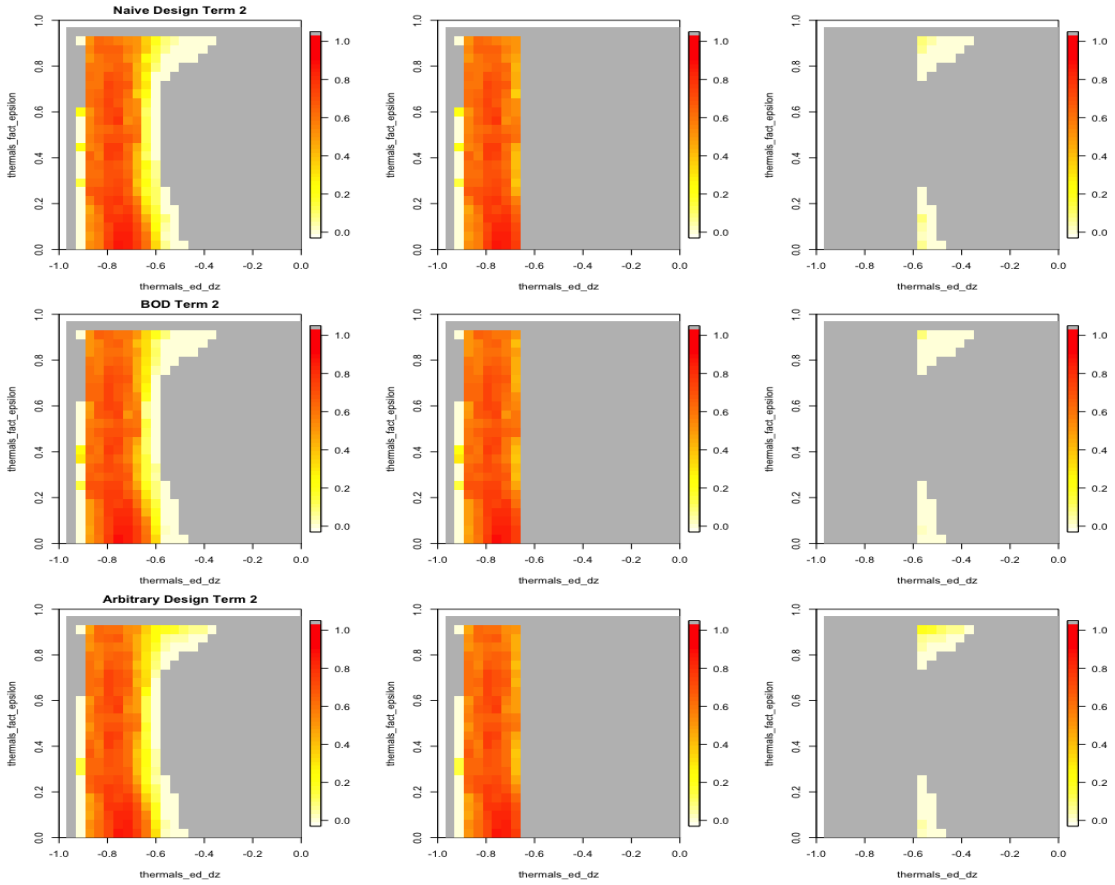

Figure C.3: Comparison of contributions towards Term 2, $\Psi_2(\xi)$, between "naive" design (Naive Design), Bayesian Optimal Design (BOD) and arbitrary design (Arbitrary Design) in the Application Study 1 for 2D projections over the NROY space for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. *First column*: the value behind each pixel represents the mean contribution towards the final value of Term 2. The ruled out input space is in grey. *Second column*: the value behind each pixel corresponds to the mean contribution towards Term 2 computed over $\mathcal{X}_1^1$, $\Psi_{21}(\xi)$. The ruled out input space and $\mathcal{X}_3^1$ are in grey. *Third column*: the value behind each pixel corresponds to the mean contribution towards Term 2 computed over $\mathcal{X}_3^1$, $\Psi_{23}(\xi)$. The ruled out input space and $\mathcal{X}_1^1$ are in grey.

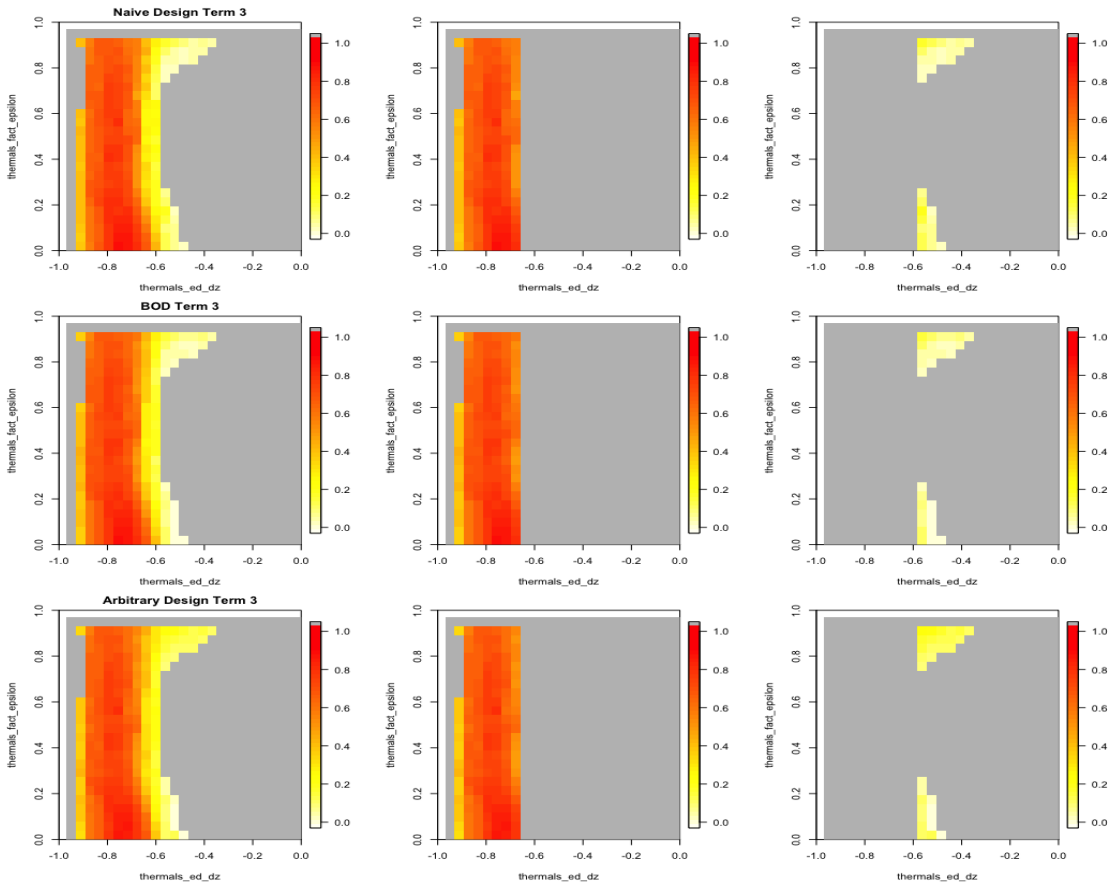

Figure C.4: Comparison of contributions towards Term 3, $\Psi_3(\xi)$, between "naive" design (Naive Design), Bayesian Optimal Design (BOD) and arbitrary design (Arbitrary Design) in the Application Study 1 for 2D projections over the NROY space for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. *First column*: the value behind each pixel represents the mean contribution towards the final value of Term 3. The ruled out input space is in grey. *Second column*: the value behind each pixel corresponds to the mean contribution towards Term 3 computed over $\mathcal{X}_1^1$, $\Psi_{31}(\xi)$. The ruled out input space and $\mathcal{X}_3^1$ are in grey. *Third column*: the value behind each pixel corresponds to the mean contribution towards Term 3 computed over $\mathcal{X}_3^1$, $\Psi_{33}(\xi)$. The ruled out input space and $\mathcal{X}_1^1$ are in grey.

C.4 Application Study 2

We provide representation of Term 2 and Term 3 over the reduced input space, $\mathcal{X}^1$, for Application Study 2, considered in Section 6.6.

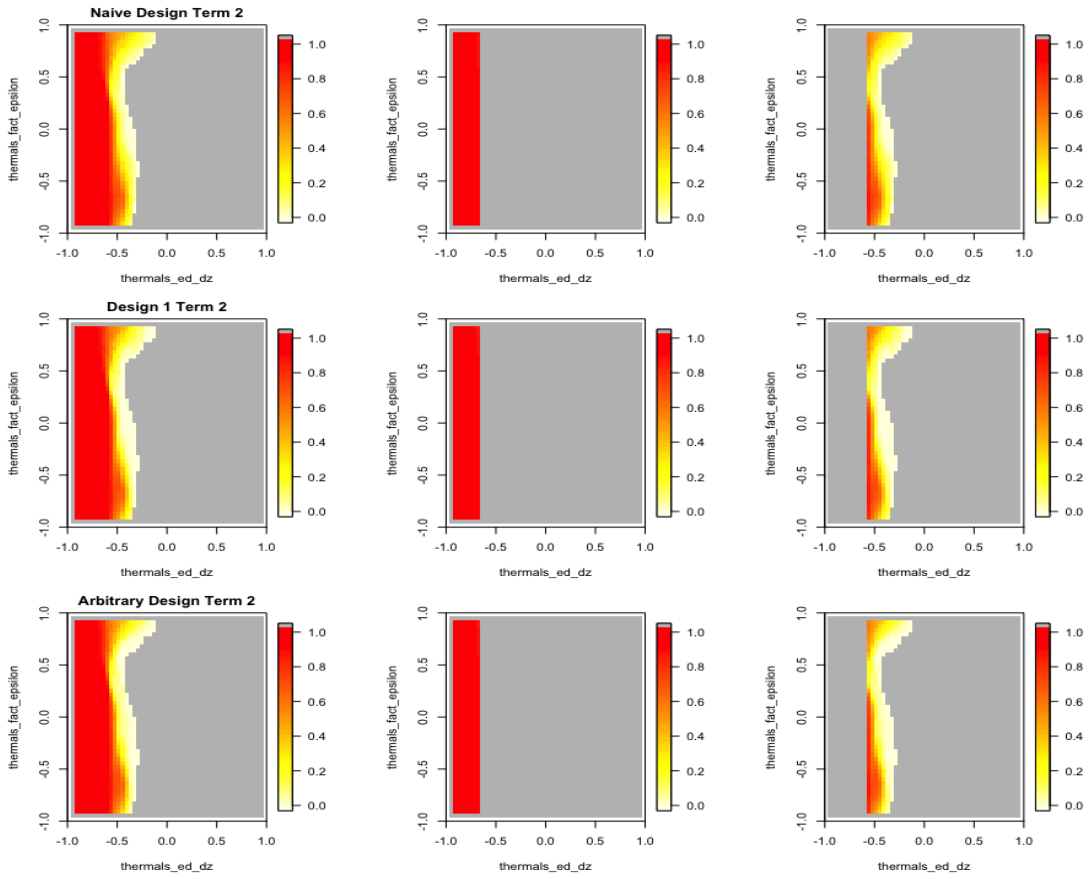

Figure C.5: Comparison of contributions towards Term 2, $\Psi_2(\xi)$, between "naive" design (Naive Design), candidate design (Design 1) and arbitrary design (Arbitrary Design) in the Application Study 2 for 2D projections over the NROY space for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. *First column*: the value behind each pixel represents the mean contribution towards the final value of Term 2. The ruled out input space is in grey. *Second column*: the value behind each pixel corresponds to the mean contribution towards Term 2 computed over $\mathcal{X}_1^1$, $\Psi_{21}(\xi)$. The ruled out input space and $\mathcal{X}_3^1$ are in grey. *Third column*: the value behind each pixel corresponds to the mean contribution towards Term 2 computed over $\mathcal{X}_3^1$, $\Psi_{23}(\xi)$. The ruled out input space and $\mathcal{X}_1^1$ are in grey.

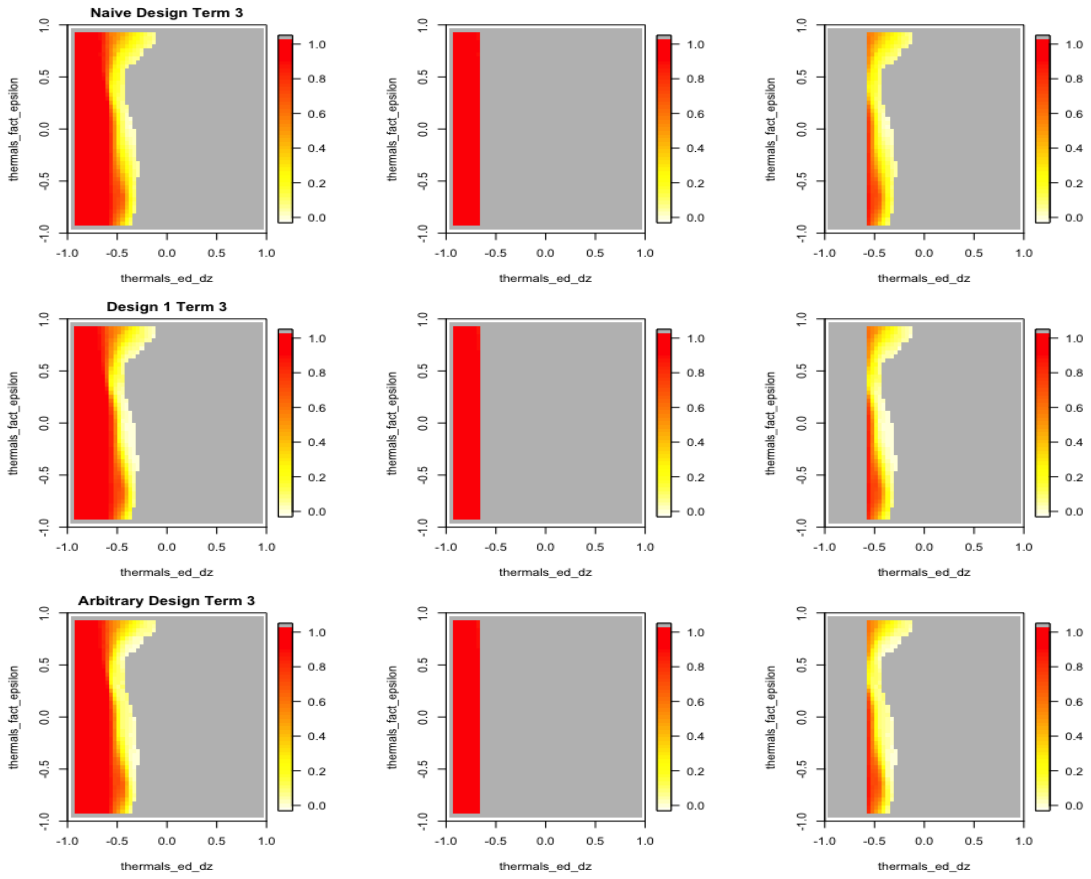

Figure C.6: Comparison of contributions towards Term 3, $\Psi_3(\xi)$, between "naive" design (Naive Design), candidate design (Design 1) and arbitrary design (Arbitrary Design) in the Application Study 2 for 2D projections over the NROY space for a pair of parameters `thermals_ed_dz` and `thermals_fact_epsilon`. *First column*: the value behind each pixel represents the mean contribution towards the final value of Term 3. The ruled out input space is in grey. *Second column*: the value behind each pixel corresponds to the mean contribution towards Term 3 computed over $\mathcal{X}_1^1$, $\Psi_{31}(\xi)$. The ruled out input space and $\mathcal{X}_3^1$ are in grey. *Third column*: the value behind each pixel corresponds to the mean contribution towards Term 3 computed over $\mathcal{X}_3^1$, $\Psi_{33}(\xi)$. The ruled out input space and $\mathcal{X}_1^1$ are in grey.

Appendix D

Emulator diagnostics

In Section 2.4, we emphasized the importance of checking the validity of a GP emulator in representing a model response. We perform a diagnostic check with Leave-one-out (LOO) cross-validation on ensemble $\{\mathrm{X}, \mathrm{F}\}$.

Leave-one-out cross-validation diagnostic plots are provided here for runs in the training set, i.e. ensemble $\{\mathrm{X}, \mathrm{F}\}$. Each plot shows the posterior mean and two standard deviation prediction intervals produced by emulators in black against input parameters of the model. The green and red points are the true model values, coloured "green" if they lie within two standard deviation prediction intervals and "red" if they lie outside.

D.1 The 2D "wavy" function

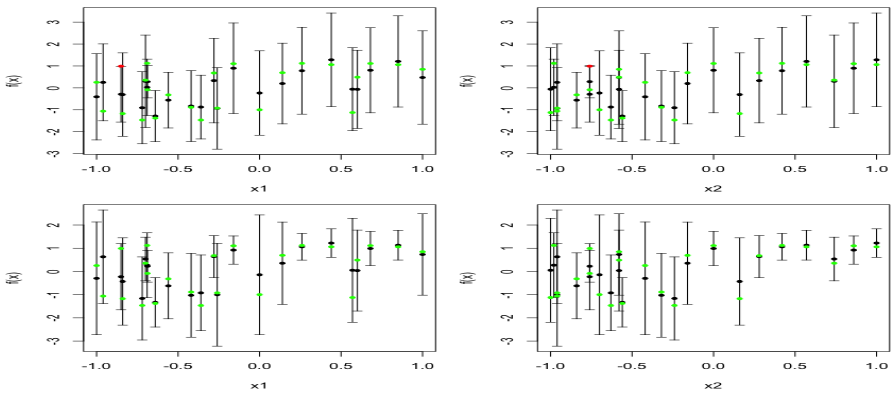

Figure D.1: Leave-one-out (LOO) cross-validation plots of stationary GP emulator (*top row*) and nonstationary GP emulator (*bottom row*) for 2D "wavy" function.

D.2 5D function

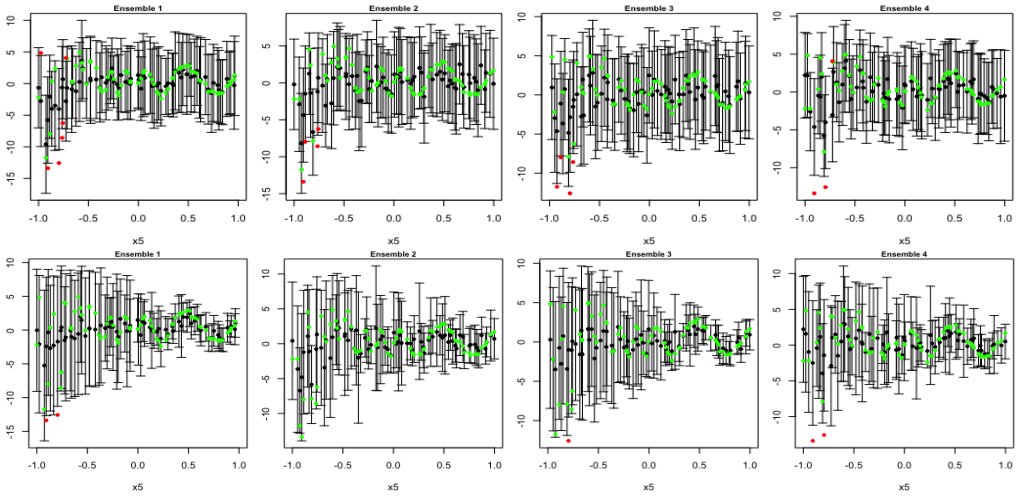

Figure D.2: Leave-one-out (LOO) cross-validation plots for four sub-designs against x_5 input obtained for stationary GP emulator (*top row*) and nonstationary GP emulator (*bottom row*).

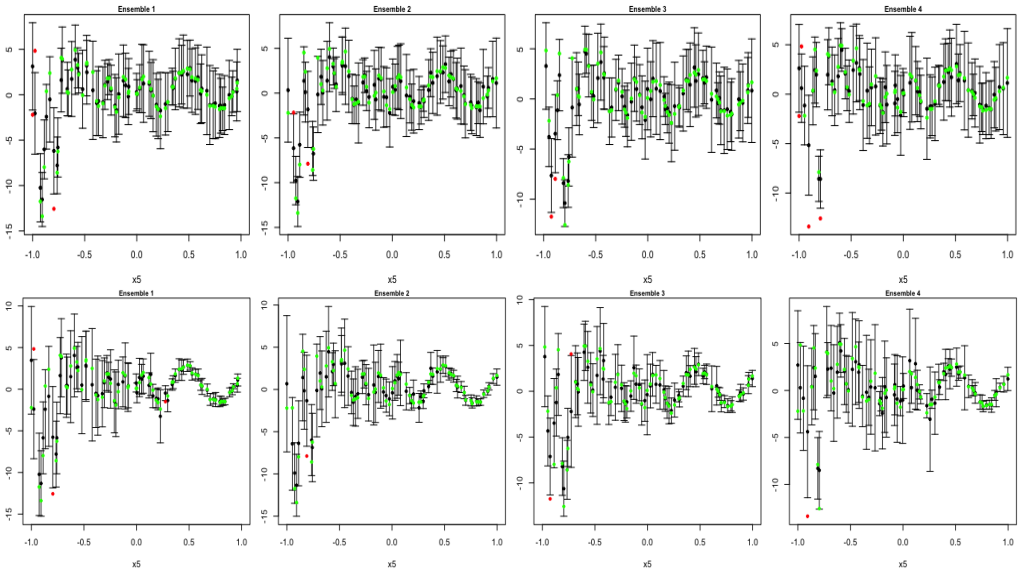

Figure D.3: Leave-one-out (LOO) cross-validation plots for four sub-designs against x_5 input obtained for stationary GP emulator (*top row*) and nonstationary GP emulator (*bottom row*) and considering x_5 as an "active" input.

D.3 Nugget predictor example

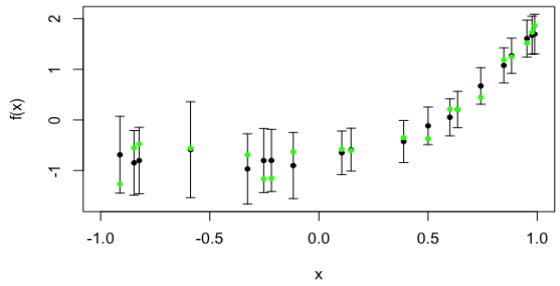

Figure D.4: Leave-one-out (LOO) cross-validation plots of nonstationary GP for a nugget predictor example.

D.4 Experiments with ARPEGE-Climat model

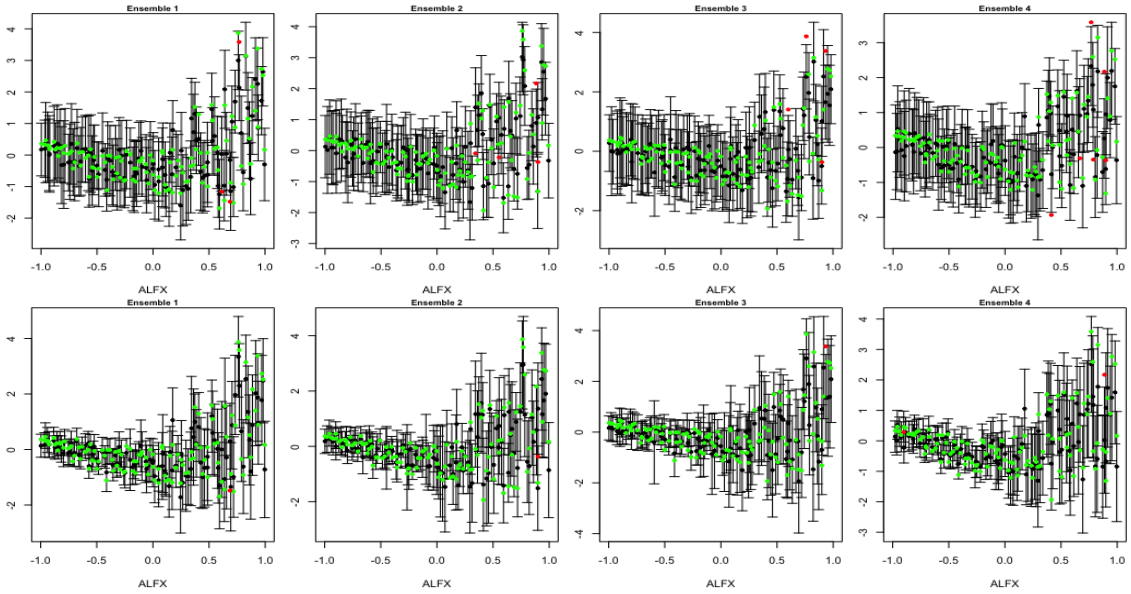

Figure D.5: Leave-one-out (LOO) cross-validation plots for four sub-designs against standardized ALFX input obtained for stationary GP emulator (*top row*) and non-stationary GP emulator (*bottom row*).

D.5 A semi-sphere centred at 0 with radius 1

We provide the diagnostic plots for a simple 2D toy function, a semi-sphere centred at 0 with radius 1, considered in section 5.5.

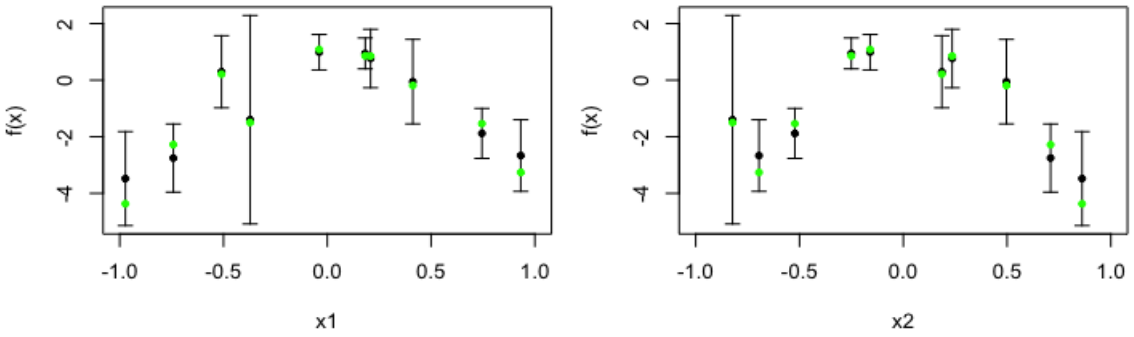

Figure D.6: Leave-one-out (LOO) cross-validation plots of stationary GP against inputs x_1 and x_2.

D.6 Application Studies

We provide the diagnostic plots for SCM `qv500` response against input parameters considered in Chapter 6.

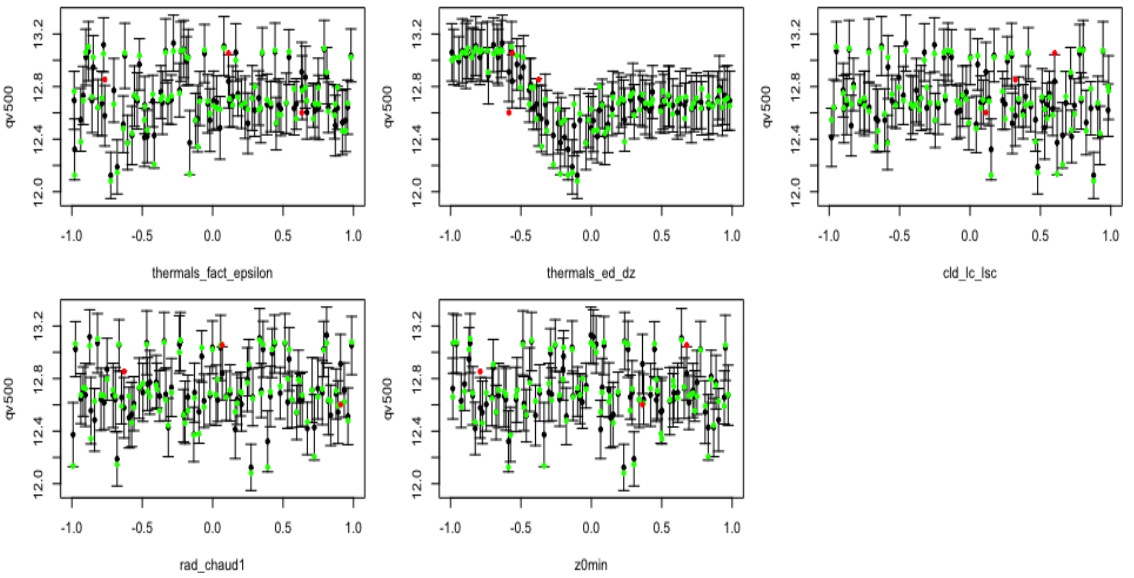

Figure D.7: Leave-one-out (LOO) cross-validation plots against model inputs obtained for stationary GP emulator.

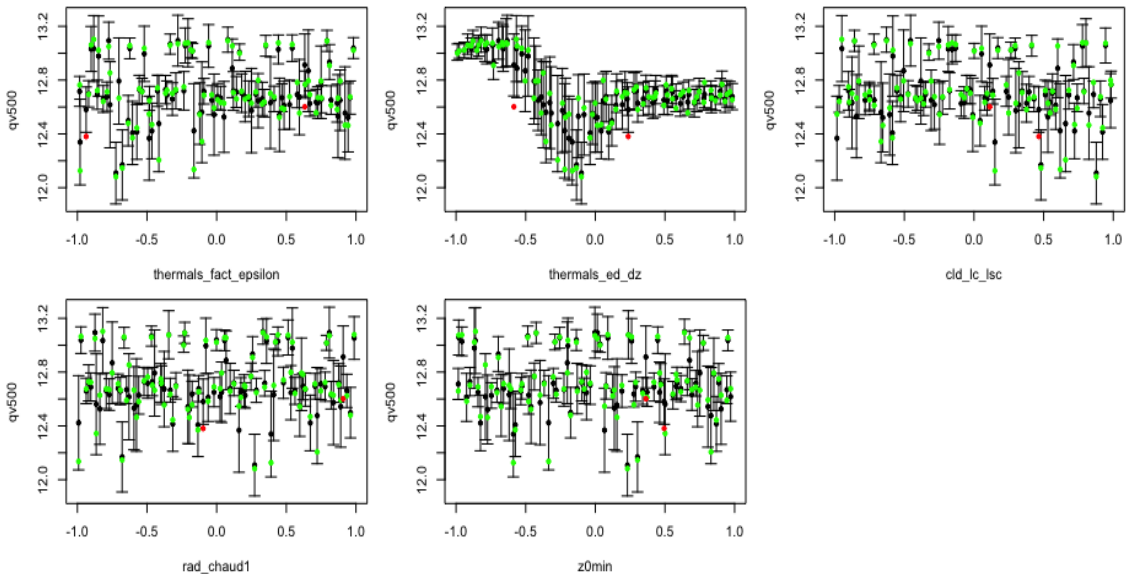

Figure D.8: Leave-one-out (LOO) cross-validation plots against model inputs obtained for nonstationary GP emulator.

Bibliography

Abdel-Lathif, A. Y., Roehrig, R., Beau, I., and Douville, H. (2018). Single-Column Modeling of Convection During the CINDY2011/DYNAMO Field Campaign With the CNRM Climate Model Version 6. *Journal of Advances in Modeling Earth Systems*, 10(3):578–602.

Adams, B. M., Bohnhoff, W., Dalbey, K., Eddy, J., Eldred, M., Gay, D., Haskell, K., Hough, P. D., and Swiler, L. P. (2009). Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: version 5.0 user's manual. *Sandia National Laboratories, Tech. Rep. SAND2010-2183*.

Adler, R. J. (1981). *The geometry of random fields*, volume 62. Siam.

Almond, R. G. (2014). A comparison of two MCMC algorithms for hierarchical mixture models. *CEUR Workshop Proceedings*, 1218:1–19.

Andrianakis, I. and Challenor, P. G. (2012). The effect of the nugget on Gaussian process emulators of computer models. *Computational Statistics and Data Analysis*, 56(12):4215–4228.

Andrianakis, I., McCreesh, N., Vernon, I., McKinley, T. J., Oakley, J. E., Nsubuga, R. N., Goldstein, M., and White, R. G. (2017). Efficient history matching of a high dimensional individual-based HIV transmission model. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):694–719.

Andrianakis, I., Vernon, I. R., McCreesh, N., McKinley, T. J., Oakley, J. E., Nsubuga, R. N., Goldstein, M., and White, R. G. (2015). 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.

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

ATI. mogp_emulator. `https://github.com/alan-turing-institute/mogp_emulator`. Accessed: 15-04-2020.

Ba, S. and Joseph, V. R. (2012). Composite Gaussian process models for emulating expensive functions. *Annals of Applied Statistics*, 6(4):1838–1860.

Ba, S. and Joseph, V. R. (2018). CGP:Composite Gaussian process models, R package.

Bachoc, F. (2013). Cross Validation and Maximum Likelihood estimations of hyper-parameters of Gaussian processes with model misspecification. *Computational Statistics & Data Analysis*, 66:55–69.

Banerjee, S., Gelfand, A. E., Knight, J. R., and Sirmans, C. F. (2004). Spatial Modeling of House Prices Using Normalized Distance-Weighted Sums of Stationary Processes. *Journal of Business and Economic Statistics*, 22(2):206–213.

Bastos, L. S. and O'Hagan, A. (2009). Diagnostics for Gaussian Process Emulators. *Technometrics*, 51(4):425–438.

Baudin, M., Dutfoy, A., Iooss, B., and Popelin, A.-L. (2017). Openturns: An industrial software for uncertainty quantification in simulation. *Handbook of uncertainty quantification*, pages 2001–2038.

Bayarri, M., Berger, J., Cafeo, J., Garcia-Donato, G., Liu, F., Palomo, J., Parthasarathy, R., Paulo, R., Sacks, J., Walsh, D., et al. (2007). Computer model validation with functional output. *The Annals of Statistics*, 35(5):1874–1906.

Beck, J. and Guillas, S. (2016). Sequential design with mutual information for computer experiments (mice): emulation of a tsunami model. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):739–766.

Bellprat, O., Kotlarski, S., Lüthi, D., and Schär, C. (2012). Objective calibration of regional climate models. *Journal of Geophysical Research: Atmospheres*, 117(D23).

Berger, J. O., De Oliveira, V., and Sansó, B. (2001). Objective Bayesian analysis of spatially correlated data. *Journal of the American Statistical Association*, 96(456):1361–1374.

Binois, M., Huang, J., Gramacy, R. B., and Ludkovski, M. (2018). Replication or exploration? Sequential design for stochastic simulation experiments. *Technometrics*, pages 1–43.

Bony, S. and Dufresne, J. (2005). Marine boundary layer clouds at the heart of tropical cloud feedback uncertainties in climate models. *Geophysical Research Letters*, 32(20):L20806.

Borsuk, M. E., Stow, C. A., and Reckhow, K. H. (2004). A Bayesian network of eutrophication models for synthesis, prediction, and uncertainty analysis. *Ecological Modelling*, 173(2-3):219–239.

Bower, R., Vernon, I., Goldstein, M., Benson, A., Lacey, C. G., Baugh, C., Cole, S., and Frenk, C. (2010). The parameter space of galaxy formation. *Monthly Notices of the Royal Astronomical Society*, 407(4):2017–2045.

Brochu, E., Cora, V. M., and De Freitas, N. (2010). A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*.

Brown, A., Cederwall, R., Chlond, A., Duynkerke, P., Golaz, J.-C., Khairoutdinov, M., Lewellen, D., Lock, A., MacVean, M., Moeng, C.-H., et al. (2002). Large-eddy simulation of the diurnal cycle of shallow cumulus convection over land. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 128(582):1075–1093.

Brynjarsdóttir, J. and O'Hagan, A. (2014). Learning about physical parameters: The importance of model discrepancy. *Inverse Problems*, 30(11):114007.

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., and Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of statistical software*, 76(1).

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

Challenor, P. (2011). Designing a computer experiment that involves switches. *Journal of Statistical Theory and Practice*, 5(1):47–57.

Chaloner, K. and Verdinelli, I. (1995). Bayesian experimental design: A review. *Statistical Science*, pages 273–304.

Chang, K.-L. and Guillas, S. (2018). Computer model calibration with large non-stationary spatial outputs: application to the calibration of a climate model. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*.

Chen, V. C., Tsui, K.-L., Barton, R. R., and Meckesheimer, M. (2006). A review on design, modeling and applications of computer experiments. *IIE transactions*, 38(4):273–291.

Chipman, H. A., George, E. I., and McCulloch, R. E. (1998). Bayesian CART Model Search. *Journal of the American Statistical Association*, 93(443):935–948.

Cohn, D. A. (1996). Neural network exploration using optimal experiment design. *Neural networks*, 9(6):1071–1083.

Couvreux, F., Guichard, F., Redelsperger, J.-L., Kiemle, C., Masson, V., Lafore, J.-P., and Flamant, C. (2005). Water-vapour variability within a convective boundary-layer assessed by large-eddy simulations and IHOP_2002 observations. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 131(611):2665–2693.

Craig, P., Goldstein, M., Seheult, A., and Smith, J. (1996). Bayes Linear strategies for matching hydrocarbon reservoir history. *Bayesian statistics*, 5:69–95.

Craig, P. S., Goldstein, M., Rougier, J. C., and Seheult, A. H. (2001). Bayesian forecasting for complex systems using computer simulators. *Journal of the American Statistical Association*, 96(454):717–729.

Craig, P. S., Goldstein, M., Seheult, A. H., and Smith, J. A. (1997). Pressure matching for hydrocarbon reservoirs: a case study in the use of Bayes linear strategies for large computer experiments. In *Case studies in Bayesian statistics*, pages 37–93. Springer.

Cressie, N. A. (1993). Statistics for spatial data: Wiley series in probability and mathematical statistics. *Find this article online*.

Cumming, J. A. and Goldstein, M. (2009). Small sample bayesian designs for complex high-dimensional models based on information gained using fast approximations. *Technometrics*, 51(4):377–388.

Cumming, J. A. and Goldstein, M. (2010). Bayes linear uncertainty analysis for oil reservoirs based on multiscale computer experiments. *O'Hagan, West, AM (eds.) The Oxford Handbook of Applied Bayesian Analysis*, pages 241–270.

Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D. (1988). A Bayesian approach to the design and analysis of computer experiments. Technical report, Oak Ridge National Lab., TN (USA).

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

Damianou, A. and Lawrence, N. (2013). Deep gaussian processes. In *Artificial Intelligence and Statistics*, pages 207–215.

Dancik, G. M. and Dorman, K. S. (2008). mlegp: statistical analysis for computer models of biological systems using R. *Bioinformatics*, 24(17):1966–1967.

Datta, A., Banerjee, S., Finley, A. O., and Gelfand, A. E. (2016). Hierarchical nearest-neighbor gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514):800–812.

De Oliveira, V. (2007). Objective Bayesian analysis of spatial data with measurement error. *Canadian Journal of Statistics*, 35(2):283–301.

Diallo, F. B., Hourdin, F., Rio, C., Traore, A.-K., Mellul, L., Guichard, F., and Kergoat, L. (2017). The Surface Energy Budget Computed at the Grid-Scale of a Climate Model Challenged by Station Data in West Africa. *Journal of Advances in Modeling Earth Systems*, 9(7):2710–2738.

Draper, N. R. and Smith, H. (1998). Selecting the "best" regression equation. *Applied regression analysis*, pages 327–368.

Dunlop, M. M., Girolami, M. A., Stuart, A. M., and Teckentrup, A. L. (2018). How deep are deep gaussian processes? *The Journal of Machine Learning Research*, 19(1):2100–2145.

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

Edwards, P. N. (2010). *A vast machine: Computer models, climate data, and the politics of global warming*. Mit Press.

Fang, K.-T., Li, R., and Sudjianto, A. (2005). *Design and modeling for computer experiments*. Chapman and Hall/CRC.

Ferrat, L. A., Goodfellow, M., and Terry, J. R. (2018). Classifying dynamic transitions in high dimensional neural mass models: A random forest approach. *PLoS computational biology*, 14(3):e1006009.

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

Fuentes, M. and Smith, R. (2001). A new class of nonstationary spatial models. *Technical report, North Carolina State University, Department of Statistics*.

Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., and Gelman, A. (2019). Visualization in Bayesian workflow. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 182(2):389–402.

Gelman, A., Lee, D., and Guo, J. (2015). Stan: A probabilistic programming language for Bayesian inference and optimization. *Journal of Educational and Behavioral Statistics*, 40(5):530–543.

Gelman, A., Simpson, D., and Betancourt, M. (2017). The prior can often only be understood in the context of the likelihood. *Entropy*, 19(10):555.

Gettelman, A. and Rood, R. B. (2016). *Demystifying Climate Models*, volume 2.

Gilks, W. R. and Berzuini, C. (2001). Following a moving target? Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(1):127–146.

Gladstone, R. M., Lee, V., Rougier, J., Payne, A. J., Hellmer, H., Le Brocq, A., Shepherd, A., Edwards, T. L., Gregory, J., and Cornford, S. L. (2012). Calibrated prediction of Pine Island Glacier retreat during the 21st and 22nd centuries with a coupled flowline model. *Earth and Planetary Science Letters*, 333:191–199.

Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

Golchi, S., Bingham, D., Chipman, H., and Campbell, D. (2015). Monotone emulation of computer experiments. *SIAM/ASA Journal on Uncertainty Quantification*, 3(1):370–392.

Goldstein, M. and Rougier, J. (2009). Reified Bayesian modelling and inference for physical systems. *Journal of Statistical Planning and Inference*, 139(3):1221–1239.

Goldstein, M. and Wooff, D. (2007). *Bayes linear statistics: Theory and methods*, volume 716. John Wiley & Sons.

Gong, Z., DiazDelaO, F., and Beer, M. (2016). Bayesian model calibration using subset simulation. *Risk, Reliability and Safety: Innovating Theory and Practice:*

Proceedings of ESREL 2016 (Glasgow, Scotland, 25-29 September 2016), page 293.

Gosling, J. P., Oakley, J. E., O'Hagan, A., et al. (2007). Nonparametric elicitation for heavy-tailed prior distributions. *Bayesian Analysis*, 2(4):693–718.

GPyTorch (2019). GPyTorch. `https://gpytorch.ai/`.

Gramacy, R. B. and Apley, D. W. (2015). Local Gaussian Process Approximation for Large Computer Experiments. *Journal of Computational and Graphical Statistics*, 24(2):561–578.

Gramacy, R. B. et al. (2007). tgp: an R package for Bayesian nonstationary, semiparametric nonlinear regression and design by treed Gaussian process models. *Journal of Statistical Software*, 19(9):6.

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

Gramacy, R. B. and Lee, H. K. (2012). Cases for the nugget in modeling computer experiments. *Statistics and Computing*, 22(3):713–722.

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

Green, P. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732.

Gu, M., Palomo, J., and Berger, J. O. (2018a). RobustGaSP: Robust Gaussian stochastic process emulation in R. *arXiv preprint arXiv:1801.01874*.

Gu, M., Wang, X., Berger, J. O., et al. (2018b). Robust Gaussian stochastic process emulation. *The Annals of Statistics*, 46(6A):3038–3066.

Hankin, R. K. (2005). Introducing BACCO, an R package for Bayesian analysis of computer code output. *Journal of Statistical Software*, 14(16):1–21.

Hartig, F., Dyke, J., Hickler, T., Higgins, S. I., O'Hara, R. B., Scheiter, S., and Huth, A. Connecting dynamic vegetation models to data-an inverse perspective. *Journal of Biogeography*, 39(12):2240–2252.

Hawkins, E. and Sutton, R. (2009). The potential to narrow uncertainty in regional climate predictions. *Bulletin of the American Meteorological Society*, 90(8):1095–1108.

Haylock, R. and O'Hagan, A. (1996). On inference for outputs of computationally expensive algorithms with uncertainty on the inputs. *Bayesian Statistics*, 5:629–637.

Heus, T. and Jonker, H. J. (2008). Subsiding shells around shallow cumulus clouds. *Journal of the Atmospheric Sciences*, 65(3):1003–1018.

Higdon, D., Gattiker, J., Williams, B., and Rightley, M. (2008). Computer Model Calibration Using High-Dimensional Output. *Journal of the American Statistical Association*, 103(482):570–583.

Higdon, D., Swall, J., and Kern, J. (1998). Bayesian statistics 6.

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

Homma, T. and Saltelli, A. (1996). Importance measures in global sensitivity analysis of nonlinear models. *Reliability Engineering & System Safety*, 52(1):1–17.

Hourdin, F., Găinusă-Bogdan, A., Braconnot, P., Dufresne, J.-L., Traore, A.-K., and Rio, C. (2015). Air moisture control on ocean surface temperature, hidden key to the warm bias enigma. *Geophysical Research Letters*, 42(24):10–885.

Hourdin, F., Mauritsen, T., Gettelman, A., Golaz, J.-C., Balaji, V., Duan, Q., Folini, D., Ji, D., Klocke, D., Qian, Y., Rauser, F., Rio, C., Tomassini, L., Watanabe, M., and Williamson, D. (2017). The Art and Science of Climate Model Tuning. *Bulletin of the American Meteorological Society*, 98(3):589–602.

Hourdin, F., Musat, I., Bony, S., Braconnot, P., Codron, F., Dufresne, J.-L., Fairhead, L., Filiberti, M.-A., Friedlingstein, P., Grandpeix, J.-Y., et al. (2006). The lmdz4 general circulation model: climate performance and sensitivity to parametrized physics with emphasis on tropical convection. *Climate Dynamics*, 27(7-8):787–813.

Iooss, B., Janon, A., Pujol, G., with contributions from Khalid Boumhaout, Veiga, S. D., Delage, T., Fruth, J., Gilquin, L., Guillaume, J., Le Gratiet, L., Lemaitre, P., Nelson, B. L., Monari, F., Oomen, R., Rakovec, O., Ramos, B., Roustant, O., Song, E., Staum, J., Sueur, R., Touati, T., and Weber, F. (2018). *sensitivity: Global Sensitivity Analysis of Model Outputs*. R package version 1.15.2.

Jackson, C. S., Sen, M. K., Huerta, G., Deng, Y., and Bowman, K. P. (2008). Error reduction and convergence in climate prediction. *Journal of Climate*, 21(24):6698–6709.

Jin, R., Chen, W., and Simpson, T. (2000). Comparative studies of metamodeling techniques under multiple modeling criteria. In *ISSMO Symposium on Multidisciplinary Analysis and Optimization*.

Kaufman, C. G. and Sain, S. R. (2010). Bayesian functional {ANOVA} modeling using Gaussian process prior distributions. *Bayesian Analysis*, 5(1):123–149.

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

Kim, H.-M., Mallick, B. K., and Holmes, C. C. (2005). Analyzing Nonstationary Spatial Data Using Piecewise Gaussian Processes. *Journal of the American Statistical Association*, 100(470):653–668.

Lam, C. Q. (2008). *Sequential adaptive designs in computer experiments for response surface model fit*. PhD thesis, The Ohio State University.

Lark, R. (2002). Optimized spatial sampling of soil for estimation of the variogram by maximum likelihood. *Geoderma*, 105(1-2):49–80.

Lee, L., Carslaw, K., Pringle, K., Mann, G., and Spracklen, D. (2011). Emulation of a complex global aerosol model to quantify sensitivity to uncertain parameters. *Atmospheric Chemistry and Physics*, 11(23):12253–12273.

Lindgren, F., Rue, H., and Lindström, J. (2011). An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(4):423–498.

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

Lopes, H. F. and Tsay, R. S. (2011). Particle filters and Bayesian inference in financial econometrics. *Journal of Forecasting*, 30(1):168–209.

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

MacDonald, B., Ranjan, P., and Chipman, H. (2015). Gpfit: An r package for fitting a gaussian process model to deterministic simulator outputs. *Journal of Statistical Software*, 64(1):1–23.

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

Mardia, K., Kent, J., and Bibby, J. (1979). Multivariate analysis. *Probability and mathematical statistics. Academic Press Inc.*

Marelli, S. and Sudret, B. (2014). UQLab: A framework for uncertainty quantification in Matlab. In *Vulnerability, uncertainty, and risk: quantification, mitigation, and management*, pages 2554–2563.

Marmin, S., Ginsbourger, D., Baccou, J., and Liandrat, J. (2018). Warped Gaussian Processes and Derivative-Based Sequential Designs for Functions with Heteroge-

neous Variations. *SIAM/ASA Journal on Uncertainty Quantification*, 6(3):991–1018.

Marsh, R., Edwards, N. R., and Shepherd, J. G. (2002). Development of a fast climate model (cgoldstein) for earth system science. *National Oceanography Centre, Southampton, European Way, Southampton SO14 3ZH*.

McKay, M. D., Beckman, R. J., and Conover, W. J. (1979). Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245.

McNeall, D., Challenor, P. G., Gattiker, J., and Stone, E. (2013). The potential of an observational data set for calibration of a computationally expensive computer model.

Mebane, W. and Sekhon, J. (2011). Genetic optimization using derivatives: the rgenoud package for R. *Journal of Statistical Software*, 42(11):1–26.

Mohammadi, H., Challenor, P., and Goodfellow, M. (2018). Emulating dynamic non-linear simulators using Gaussian processes. *arXiv preprint arXiv:1802.07575*.

Mohammadi, H., Challenor, P., Goodfellow, M., and Williamson, D. (2019). Emulating computer models with step-discontinuous outputs using Gaussian processes. *arXiv preprint arXiv:1903.02071*.

Montagna, S. and Tokdar, S. T. (2016). Computer Emulation with Nonstationary Gaussian Processes. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):26–47.

Morris, D. E., Oakley, J. E., and Crowe, J. A. (2014). A web-based tool for eliciting probability distributions from experts. *Environmental Modelling & Software*, 52:1–4.

Morris, M. D. and Mitchell, T. J. (1995). Exploratory Designs for Computer Experiments. *J. Statist. Plann. Inference*, 43:381–402.

Murphy, J. M., Sexton, D., Jenkins, G., Booth, B., Brown, C., Clark, R., Collins, M., Harris, G., Kendon, E., Betts, R., et al. (2009). UK climate projections science report: climate change projections.

Neal, R. M. (1997). Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. *arXiv preprint physics/9701026*.

Oakley, J. (2002). Eliciting Gaussian process priors for complex computer codes. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 51(1):81–97.

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

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

Oakley, J. E. and O'Hagan, A. (2007). Uncertainty in prior elicitations: a nonparametric approach. *Biometrika*, 94(2):427–441.

O'Hagan, A. (1998). Eliciting expert beliefs in substantial practical applications. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 47(1):21–35.

Oughton, R. H. and Craig, P. S. (2016). Hierarchical Emulation: A Method for Modeling and Comparing Nested Simulators. *Journal on Uncertainty Quantification*, 4:495–519.

Owen, N., Challenor, P., Menon, P., and Bennani, S. (2017). Comparison of surrogate-based uncertainty quantification methods for computationally expensive simulators. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):403–435.

Paciorek, C. J. (2003). *Nonstationary Gaussian processes for regression and spatial modelling*. PhD thesis, Citeseer.

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

Patelli, E. (2017). COSSAN: a multidisciplinary software suite for uncertainty quantification and risk management. *Handbook of uncertainty quantification*, pages 1909–1977.

Paulo, R. et al. (2005). Default priors for Gaussian processes. *The Annals of Statistics*, 33(2):556–582.

Plummer, M. et al. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In *Proceedings of the 3rd international workshop on distributed statistical computing*, volume 124. Vienna, Austria.

Pope, C. A., Gosling, J. P., Barber, S., Johnson, J., Yamaguchi, T., Feingold, G., and Blackwell, P. (2018). Modelling spatial heterogeneity and discontinuities using Voronoi tessellations.

Pukelsheim, F. (1994). The three sigma rule. *The American Statistician*, 48(2):88–91.

Randall, D. A., Xu, K.-M., Somerville, R. J., and Iacobellis, S. (1996). Single-column models and cloud ensemble models as links between observations and climate models. *Journal of Climate*, 9(8):1683–1697.

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

Ranjan, P., Lu, W., Bingham, D., Reese, S., Williams, B. J., Chou, C.-C., Doss, F., Grosskopf, M., and Holloway, J. P. (2011). Follow-up experimental designs for computer models and physical processes. *Journal of Statistical Theory and Practice*, 5(1):119–136.

Rasmussen, C. E. and Williams, C. K. I. (2004). *Gaussian processes for machine learning.*, volume 14.

Razavi, S., Tolson, B. A., and Burn, D. H. (2012). Review of surrogate modeling in water resources. *Water Resources Research*, 48(7).

Renard, B., Kavetski, D., Kuczera, G., Thyer, M., and Franks, S. W. (2010). Understanding predictive uncertainty in hydrologic modeling: The challenge of identifying input and structural errors. *Water Resources Research*, 46(5).

Ripley, B. D. (1991). *Statistical inference for spatial processes*. Cambridge university press.

Ripley, B. D. (2005). *Spatial statistics*, volume 575. John Wiley & Sons.

Rogers, S., Aftosmis, M., and Pandya, S. (2003). In *Automated CFD Parameter Studies on Distributed Parallel Computers. In 16th AIAA Computational Fluid Dynamics Conference. AIAA Paper*, volume 4229.

Roininen, L., Girolami, M., and Lasanen, S. (2018). Hyperpriors for Matérn fields with applications in Bayesian inversion. *arXiv preprint arXiv:1612.02989*.

Romanowicz, R., Beven, K., and Tawn, J. (1994). Evaluation of predictive uncertainty in nonlinear hydrological models using a Bayesian approach. *Statistics for the Environment*, 2:297–317.

Rougier, J. (2007). Probabilistic inference for future climate using an ensemble of climate model evaluations. *Climatic Change*, 81(3-4):247–264.

Rougier, J., Guillas, S., Maute, A., and Richmond, A. D. (2009). Expert Knowledge and Multivariate Emulation: The Thermosphere–Ionosphere Electrodynamics General Circulation Model (TIE-GCM). *Technometrics*, 51(4):414–424.

Roustant, O., Ginsbourger, D., and Deville, Y. (2012a). DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization.

Roustant, O., Ginsbourger, D., and Deville, Y. (2012b). Dicekriging, diceoptim: Two r packages for the analysis of computer experiments by kriging-based metamodelling and optimization. *Journal of Statistical Software*, 51(1):54p.

Rue, H. and Held, L. (2005). *Gaussian Markov random fields: theory and applications*. Chapman and Hall/CRC.

Rue, H., Martino, S., and Chopin, N. (2009). Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)*, 71(2):319–392.

Ryan, E. G., Drovandi, C. C., McGree, J. M., and Pettitt, A. N. (2016). A review of modern computational algorithms for bayesian optimal design. *International Statistical Review*, 84(1):128–154.

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

Salimbeni, H. and Deisenroth, M. (2017). Doubly stochastic variational inference for deep gaussian processes. In *Advances in Neural Information Processing Systems*, pages 4588–4599.

Saltelli, A., Chan, K., Scott, E. M., et al. (2000). *Sensitivity analysis*, volume 1. Wiley New York.

Salter, J. M. (2017). Uncertainty quantification for spatial field data using expensive computer models: refocussed Bayesian calibration with optimal projection.

Salter, J. M. and Williamson, D. (2016). A comparison of statistical emulation methodologies for multi-wave calibration of environmental models. *Environmetrics*, 27(8):507–523.

Salter, J. M., Williamson, D. B., Scinocca, J., and Kharin, V. (2018). Uncertainty quantification for spatio-temporal computer models with calibration-optimal bases.

Samper, F. J. and Neuman, S. P. (1989). Estimation of spatial covariance structures by adjoint state maximum likelihood cross validation: 1. theory. *Water Resources Research*, 25(3):351–362.

Sampson, P. D. and Guttorp, P. (1992). Nonparametric Estimation of Nonstationary Spatial Covariance Structure. *Journal of the American Statistical Association*, 87(417):108.

Santner, T. J., Williams, B. J., and Notz, W. I. (2003). *The Design and Analysis of Computer Experiments*. Springer Series in Statistics. Springer New York, New York, NY.

Schmidt, A. M. and O'Hagan, A. (2003). Bayesian Inference for Nonstationary Spatial Covariance Structure via Spatial Deformations. *Journal of the Royal Statistical Society, Series B*, 65(4):745–758.

Schwarz, G. (1978). Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2):461–464.

Sexton, D. M., Murphy, J. M., Collins, M., and Webb, M. J. (2012). Multivariate probabilistic projections using imperfect climate models part I: outline of methodology. *Climate dynamics*, 38(11-12):2513–2542.

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

Siebesma, A. P., Bretherton, C. S., Brown, A., Chlond, A., Cuxart, J., Duynkerke, P. G., Jiang, H., Khairoutdinov, M., Lewellen, D., Moeng, C.-H., et al. (2003). A large eddy simulation intercomparison study of shallow cumulus convection. *Journal of the Atmospheric Sciences*, 60(10):1201–1219.

Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. *Advances in Neural Information Processing Systems*, 18:1257.

Spiegelhalter, D. J., Thomas, A., Best, N. G., Gilks, W., and Lunn, D. (1996). BUGS: Bayesian inference using Gibbs sampling. *Version 0.5,(version ii) http://www. mrc-bsu. cam. ac. uk/bugs*, 19.

Stan Development Team (2017). Stan Modeling Language User's Guide and Reference Manual.

Stein, M. (1990a). Uniform asymptotic optimality of linear predictions of a random field using an incorrect second-order structure. *Ann. Statist.*, 18(2):850–872.

Stein, M. L. (1988). Asymptotically efficient prediction of a random field with a misspecified covariance function. *The Annals of Statistics*, pages 55–63.

Stein, M. L. (1990b). Bounds on the efficiency of linear predictions using an incorrect covariance function. *Ann. Statist.*, 18(3):1116–1138.

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

Tavassoli, Z., Carter, J. N., and King, P. R. (2005). An analysis of history matching errors. *Computational Geosciences*, 9(2-3):99–123.

Tavassoli, Z., Carter, J. N., King, P. R., et al. (2004). Errors in history matching. *SPE Journal*, 9(03):352–361.

Van Oijen, M., Rougier, J., and Smith, R. (2005). Bayesian calibration of process-based forest models: bridging the gap between models and data. *Tree Physiology*, 25(7):915–927.

van Vuuren, D. P., Edmonds, J., Kainuma, M., Riahi, K., Thomson, A., Hibbard, K., Hurtt, G. C., Kram, T., Krey, V., Lamarque, J.-F., Masui, T., Meinshausen, M., Nakicenovic, N., Smith, S. J., and Rose, S. K. (2011). The representative concentration pathways: an overview. *Climatic Change*, 109(1):5.

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

Vernon, I. R. and Goldstein, M. (2009). Bayes linear analysis of imprecision in computer models, with application to understanding galaxy formation. SIPTA.

Vial, J., Dufresne, J.-L., and Bony, S. (2013). On the interpretation of inter-model spread in CMIP5 climate sensitivity estimates. *Climate Dynamics*, 41(11-12):3339–3362.

Voldoire, A., Sanchez-Gomez, E., Salas y Mélia, D., Decharme, B., Cassou, C., Sénési, S., Valcke, S., Beau, I., Alias, A., Chevallier, M., Déqué, M., Deshayes, J., Douville, H., Fernandez, E., Madec, G., Maisonnave, E., Moine, M.-P., Planton,

S., Saint-Martin, D., Szopa, S., Tyteca, S., Alkama, R., Belamari, S., Braun, A., Coquart, L., and Chauvin, F. (2013). The CNRM-CM5.1 global climate model: description and basic evaluation. *Climate Dynamics*, 40(9-10):2091–2121.

Wagener, T. and Gupta, H. V. (2005). Model identification for hydrological forecasting under uncertainty. *Stochastic Environmental Research and Risk Assessment*, 19(6):378–387.

Wang, H. and Feingold, G. (2009). Modeling mesoscale cellular structures and drizzle in marine stratocumulus. Part I: Impact of drizzle on the formation and evolution of open cells. *Journal of the Atmospheric Sciences*, 66(11):3237–3256.

Warnes, J. and Ripley, B. (1987). Problems with likelihood estimation of covariance functions of spatial gaussian processes. *Biometrika*, 74(3):640–642.

Wickham, H. (2015). *R packages: organize, test, document, and share your code*. O'Reilly Media, Inc.

Williams, K. and Webb, M. (2009). A quantitative performance assessment of cloud regimes in climate models. *Climate dynamics*, 33(1):141–157.

Williamson, D. (2015). Exploratory ensemble designs for environmental models using k-extended Latin Hypercubes. *Environmetrics*, 26(4):268–283.

Williamson, D. and Blaker, A. T. (2014). Evolving Bayesian Emulators for Structured Chaotic Time Series, with Application to Large Climate Models. *SIAM/ASA Journal on Uncertainty Quantification*, 2:1–28.

Williamson, D., Blaker, A. T., Hampton, C., and Salter, J. (2015). Identifying and removing structural biases in climate models with history matching. *Climate Dynamics*, 45(5-6):1299–1324.

Williamson, D., Goldstein, M., Allison, L., Blaker, A., Challenor, P., Jackson, L., and Yamazaki, K. (2013). History matching for exploring and reducing climate model parameter space using observations and a large perturbed physics ensemble. *Climate Dynamics*, 41(7-8):1703–1729.

Williamson, D. and Vernon, I. (2013). Efficient uniform designs for multi-wave computer experiments. *arXiv preprint arXiv:1309.3520*.

Williamson, D. B., Blaker, A. T., and Sinha, B. (2017). Tuning without over-tuning: Parametric uncertainty quantification for the NEMO ocean model. *Geoscientific Model Development*, 10(4):1789–1816.

Wong, R. K., Storlie, C. B., and Lee, T. C. (2017). A frequentist approach to computer model calibration. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(2):635–648.

Woods, D. (2010). Robust designs for binary data: applications of simulated annealing. *Journal of Statistical Computation and Simulation*, 80(1):29–41.

Zhu, Z. and Stein, M. L. (2005). Spatial sampling design for parameter estimation of the covariance function. *Journal of Statistical Planning and Inference*, 134(2):583–603.

Zimmerman, D. L. (2006). Optimal network design for spatial prediction, covariance parameter estimation, and empirical prediction. *Environmetrics: The official journal of the International Environmetrics Society*, 17(6):635–652.