Bayesian Sensitivity Analysis of Bifurcating Nonlinear Models

W.Becker[a], K.Worden and J. Rowson
Department of Mechanical Engineering,
University of Sheffield, Mappin Street, Sheffield S1 3JD, UK.
email: **william.becker@jrc.ec.europa.eu**

Abstract
Sensitivity analysis allows one to investigate how changes in input parameters to a system affect the output. When computational expense is a concern, metamodels such as Gaussian processes can offer considerable computational savings over Monte Carlo methods, albeit at the expense of introducing a data modelling problem. In particular, Gaussian processes assume a smooth, non-bifurcating response surface. This work highlights a recent extension to Gaussian processes which uses a decision tree to partition the input space into homogeneous regions, and then fits separate Gaussian processes to each region. In this way, bifurcations can be modelled at region boundaries and different regions can have different covariance properties. To test this method, both the treed and standard methods were applied to the bifurcating response of a Duffing oscillator and a bifurcating FE model of a heart valve. It was found that the treed Gaussian process provides a practical way of performing uncertainty and sensitivity analysis on large, potentially-bifurcating models, which cannot be dealt with by using a single GP, although an open problem remains how to manage bifurcation boundaries that are not parallel to coordinate axes.

Keywords: Sensitivity analysis, Gaussian process, CART, metamodel, bifurcation

1 Introduction

Uncertainty analysis (UA) is a field of increasing interest in computer modelling, both within and beyond the realm of engineering. Advances in the theory of numerical simulation (such as the continuing advances in finite element (FE) analysis and computational fluid dynamics), coupled with a steady increase in available processing power, have enabled the use of increasingly sophisticated simulations to model complicated real-world processes. However, the increased complexity of these models tends to require a greater amount of information to be specified at the input. Examples of input variables within the context of engineering models could include material properties, loads, dimensions and temperatures. Any of these inputs can be subject to some uncertainty; for example, the operating temperature of a structure could be within a wide range, or the material properties could vary naturally (a good example of this is in modelling biomaterials – see [1]). The question, central to UA, then arises: what is the uncertainty in the model output, given the input uncertainty?

A furtherance of UA is *Sensitivity Analysis* (SA), which evaluates the contribution of inputs (and sets of inputs) to the output uncertainty, and the effects of varying parameters within specified ranges. The motivations of SA are various (Saltelli provides an extensive list [2]), including the identification of particularly influential parameters to see whether their uncertainty can be reduced somehow (thereby

[a] Current address: European Commission - Joint Research Centre, IPSC - Econometrics and Applied Statistics Unit, TP361, Via Enrico Fermi 2749, I-21027 Ispra (VA), Italy

reducing output uncertainty), and gaining deeper understanding of a model's response to its inputs. In general, SA is seen as a useful tool to increase the robustness and overall quality of computer simulations, and often goes hand in hand with UA.

Introducing some notation, let the model to be investigated be denoted as $f(\boldsymbol{x})$, where $\boldsymbol{x}$ is a d-dimensional vector of *uncertain* inputs (there may also be a number of model inputs that are regarded as known, which will not be considered here). The model may indeed be a very complex system of nonlinear equations, but from a "black box" point of view it can simply be regarded as a function of inputs. The model could yield any number of outputs, but attention here will be restricted to a scalar output $y = f(\boldsymbol{x})$.

In order to perform an uncertainty analysis, it is necessary to somehow quantify the uncertainty on the inputs. Although there are many ways of doing this – see the recent book by Klir for a summary [3] – this article will consider the probabilistic framework. It is therefore assumed from here on that the inputs are random variables $\{X_i\}_{i=1}^{d}$, and that a joint probability density function (pdf) $p(\boldsymbol{x})$ can be defined over them. This lightly skips over the often-troublesome problem of eliciting the pdf, though an extensive discussion on this matter can be found in [4].

From the probabilistic perspective, the aim of UA is to estimate the pdf of the output random variable Y, given the $p(\boldsymbol{x})$ assigned to the inputs. In practice, this is often simplified to the estimation of the expected value $E(Y)$, and the variance var(Y). SA, on the other hand, requires a quantification of the sensitivity of y to changes in the inputs. Saltelli discusses a great number of ways to do this [5]. Two simple types of SA are simply ranking variables in order of importance (*screening*); and taking partial derivatives of y with respect to individual inputs (*local SA*). Screening is usually done as a precursor to a more sophisticated SA, to filter out unimportant variables. Local SA is also useful, but only considers small perturbations about the nominal values of input parameters. In the case of complex nonlinear models with wide uncertainties, this approach is unsatisfactory.

The most informative approach is known as *global SA*, in which the variation of y is examined over the whole of the input space. By far the most widely-used method of doing this, proposed by Sobol' [6], involves decomposing var(Y) into portions attributable to inputs and subsets of inputs. Specifically, a variance decomposition is used (under an assumption of independence between inputs) such that,

$$\mathrm{var}(Y) = \sum_i V_i + \sum_i \sum_{j>i} V_{ij} + \ldots + V_{12\ldots d} \tag{1}$$

$$V_i = \mathrm{var}(E(Y \mid X_i))$$

$$V_{ij} = \mathrm{var}\{E(Y \mid X_i, X_j)\} - \mathrm{var}\{E(Y \mid X_i)\} - \mathrm{var}\{E(Y \mid X_j)\}$$

$$\vdots$$

The quantity V_i represents the amount that var(Y) would be reduced, if X_i were to become known, therefore giving a global measure of importance of that variable. The higher order terms V_{ij} represent variance due to interactions between variables, *additional* to the variance caused by the inputs acting alone. There are 2^d-1 terms in this decomposition, with interactions up to the final d-way interaction between all variables (the last term in Equation (1)). Typically these quantities are standardised by dividing by var(Y), giving e.g. $S_i = V_i/\mathrm{var}(Y)$, where S_i is known as the main effect index (MEI), and the S_{ij}, S_{ijk}, etc. follow from similar definitions. Note that this implies that,

$$\sum_i S_i + \sum_i \sum_{j>i} S_{ij} + \ldots + S_{12\ldots k} = 1 \tag{2}$$

Since there may be a great number of terms in Equation (2), Homma and Saltelli proposed a further quantity S_{Ti}, known as the total sensitivity index (TSI) [7], which is defined as:

$$S_{Ti} = \frac{E\{\mathrm{var}(Y \mid X_{-i})\}}{\mathrm{var}(Y)} = 1 - \frac{\mathrm{var}\{E(Y \mid X_{-i})\}}{\mathrm{var}(Y)} \tag{3}$$

This measure is therefore the sum of the variance caused by X_i and all its interactions with other variables. Finally, another useful (but qualitative) indicator of sensitivity is to plot the *main effect* $E(Y \mid X_i)$ against X_i, which illustrates how the output varies with respect to variations in a single variable.

All the quantities addressed here can be expressed as integrals, e.g.

$$E(Y \mid X_i) = \int_{\mathrm{X}_{-i}} f(\boldsymbol{x})\, p(\boldsymbol{x}_{-i} \mid x_i) d\boldsymbol{x}_{-i} \tag{4}$$

where Ξ_{-i} denotes the sample space (support) of the variables X_{-i}, where $-i$ denotes all variables except i. If the function $f(\boldsymbol{x})$ were tractable, this could be done analytically, however in the majority of practical cases it is not (consider a large FE model, for instance). In such cases, the usual approach is to use the Monte Carlo (MC) method [8], using specific estimators built for the purpose [9, 10]. To achieve an acceptable level of accuracy, this requires sampling the model a large number N_{MC} of times across the input space (at least several hundred, often thousands) for every quantity to be calculated – i.e. $d \times N_{\mathrm{MC}}$ samples for the MEIs, plus the same number again for the TSIs. The key problem with this approach is that many large models require minutes, hours or longer to evaluate $f(\boldsymbol{x})$ at a single input point, therefore the cost of a thorough sensitivity analysis can be completely prohibitive.

In order to reduce computational expense, a class of methods exist which involve building an emulator (equivalently a *metamodel* or *surrogate model*) which imitates the behaviour of the original model, but is considerably (computationally) cheaper to run. The emulator is trained from a small number of n training data, consisting of model runs at selected values of input variables, the idea being that $n << N_{MC}$. It may then be used to provide uncertainty and sensitivity estimates either by applying MC techniques to the emulator, or even better, by analytically propagating uncertainty if the emulator is sufficiently tractable. Many types of emulators exist: well-known choices range from linear models (including linear combinations of basis functions, such as polynomials, Fourier expansions and other orthogonal expansions), radial basis functions, artificial neural networks, splines and support vector machines, all of which are discussed in [11]. Some newer approaches include "gradient boosting machines" [12] (a method based on nested "weak learners"), ACOSSO [13] (a method using multivariate splines and variable selection), "multivariate adaptive regression splines" (MARS) [14] (using an optimised combination of linear splines), and Gaussian processes (GPs) [15]. Comparisons of these methods can be found in [16-19].

It is clear that desirable properties of an emulator include efficiency (the ability to emulate a model for as few training data as possible), and ideally tractability (though this is often circumvented using MC). Perhaps most importantly is the ability to emulate a wide class of model responses, which could be called "flexibility" or "generality". As a simple example, consider fitting a linear regression to data that varies sinusoidally – it is clear that this will provide a very poor approximation of the true model. Changing basis functions or using some of the more sophisticated emulators described above allows the emulation of nonlinear models, but severe nonlinearities pose difficulties. Of particular interest to this paper is the case when a model has a discontinuity in its response, perhaps caused by a bifurcation. Emulators such as ACOSSO and MARS are based on splines, which assume continuous functions, therefore are not suited to such problems. Similarly, GPs assume a smooth response.

This work investigates a new emulator introduced by Gramacy and Lee [20] which uses Classification and Regression Trees (CARTs) to divide input space, and fits GPs in each region. If the divisions are located at bifurcation points, this gives the possibility to faithfully model severely nonlinear data. Such "treed GPs" were originally developed to model heteroskedastic data (data with a non-constant variance). In this work the treed GP will be tested in the context of uncertainty and sensitivity analysis of bifurcating or discontinuous models. While the approach here is not the invention of the authors of this paper, the intention of this paper is to present this cutting-edge emulation method in a walkthrough format that is approachable by statistically-minded engineers, in a sense "bridging the gap" between the computer

science and engineering communities. At the same time, its relevance to mechanical engineering is demonstrated on two case studies that illustrate the power of the approach.

This paper is organised as follows: in Section 2 the theory of GPs is introduced, after which treed GPs are discussed in Section 3. In Sections 4 and 5, treed and non-treed GPs are applied to bifurcating data generated by a Duffing oscillator, followed by a bifurcating biomechanical FE model. Conclusions follow.

2 Gaussian Process Regression

The concept of the Gaussian Process has been around for many years, but was first proposed for use in the context of emulation of computer codes by Sacks *et al*. in 1989 [21]. Since then, the power of GPs has been exploited to model many varied types of computer code; Kennedy *et al*. provide a summary of recent applications [22]. For a very comprehensive treatment of GPs in a wider context see [23].

A GP can be thought of as a *distribution over functions*, i.e. the random variable of the distribution is a function rather than a single number or fixed-length vector. Rather than returning a crisp output value *y* for any given input point $\boldsymbol{x}$ (as in a standard regression), the GP returns a specification for a Gaussian probability distribution. This means that for any given set of input points, the corresponding output values are distributed joint-normally. Importantly, this concept applies to combinations of known points and unknown points, which provides the mechanism for training the GP – probability distributions of unobserved points can be conditioned on known training data.

The GP must be trained using a set of training data, which consists of *n* input-output pairs – i.e. it is composed of a vector of output points $\mathbf{y} = (y_1, y_2, ..., y_n)^T$ and a $d \times n$ matrix of training input points $\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$. The training of the GP is a machine learning problem and as such the number and location of the training points are crucial factors in determining the quality of the emulator. This problem will not be discussed here at length, but detailed information can be found in [24].

The GP is trained according to the principles of Bayesian inference (see e.g. [25]), which means that prior assumptions are made about the model to give a *prior distribution* then a set of training data is used to update the prior assumptions, resulting in a *predictive distribution*. The training is completed by marginalising hyperparameters (where possible), yielding the *posterior distribution*. The prior distribution described here is that used by Oakley and O'Hagan [15]. The treed GP uses a slightly different hierarchical prior, which is described in Section 3.4.1.

It is not intended here to provide a full description of the GP, since this article is intended to focus on the use of treed GPs. For a more detailed description of Bayesian SA using GPs, see [15, 26].

2.1 Prior Specification

The GP is completely defined by a mean function *m* and covariance function *c*, defining a mean vector and covariance matrix of the output distribution for any given set of inputs, i.e.,

$$f(\boldsymbol{x}) \sim \mathrm{GP}\left(m(\boldsymbol{x}), \sigma^2 c(\boldsymbol{x}, \boldsymbol{x}')\right) \tag{5}$$

where $\boldsymbol{x}'$ is any neighbouring point to $\boldsymbol{x}$ and σ^2 is a scaling parameter to be learned from the data. The prior mean (i.e. the first "guess" about the form of the data) is defined here as a linear fit through the training data,

$$m(\boldsymbol{x}) = E\{f(\boldsymbol{x}) \mid \mathbf{w}\} = \boldsymbol{\varphi}(\boldsymbol{x})^T \mathbf{w}\,, \tag{6}$$

where $\boldsymbol{\varphi}(\boldsymbol{x})^T$ is a specified vector of *q* basis functions of $\boldsymbol{x}$, and $\mathbf{w}$ is the corresponding *q*-dimensional vector of coefficients. For simplicity, $\boldsymbol{\varphi}(\boldsymbol{x})^T$ is chosen here to be $(1, \boldsymbol{x}^T)$, which represents the belief that

the data is close to a global linear model with residuals modelled by a GP. $\boldsymbol{\varphi}(\boldsymbol{x})^T$ can however be extended to other functions if there is information to suggest that this is appropriate. The covariance between outputs is specified as a function of any two input points such that,

$$\sigma^2 c(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{cov}\{f(\boldsymbol{x}), f(\boldsymbol{x}') \mid \sigma^2\} \tag{7}$$

where $c(\boldsymbol{x}, \boldsymbol{x}')$ is a covariance function that decreases as $|\boldsymbol{x} - \boldsymbol{x}'|$ increases, and satisfies $c(\boldsymbol{x}, \boldsymbol{x}) = 1$ for all $\boldsymbol{x}$. The function used here is the following,

$$\mathrm{cov}\{f(\boldsymbol{x}), f(\boldsymbol{x}') \mid \sigma^2, B\} = \sigma^2 \exp\{-(\boldsymbol{x} - \boldsymbol{x}')^T B(\boldsymbol{x} - \boldsymbol{x}')\}, \tag{8}$$

where B is a $d \times d$ diagonal matrix of length-scales, representing the roughness of the output with respect to the individual input parameters (that B is diagonal requires an assumption of independence between inputs). Note that, implicit in (9), is the idea that covariance between function values is expressible in terms of distance between inputs in the input/feature space; this is a manifestation of the 'kernel trick' as discussed in [11]. The choice of covariance function is free but essentially fixes the expansion basis of the GP; with the squared-exponential covariance in (9), the GP becomes a radial-basis function interpolator. Equations (6) and (8) together imply a prior mean for any vector of outputs $\mathbf{y} = (f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), ..., f(\boldsymbol{x}_n))^T$ as an n-variate normal distribution given by,

$$\mathbf{y} \mid \mathbf{w}, \sigma^2, B \sim \mathrm{N}(\Phi\mathbf{w}, \sigma^2 A) \tag{9}$$

$$\Phi^T = (\phi(\boldsymbol{x}_1), ..., \phi(\boldsymbol{x}_n)),$$

$$A = \begin{pmatrix} 1 & c(\boldsymbol{x}_1, \boldsymbol{x}_2) & \cdots & c(\boldsymbol{x}_1, \boldsymbol{x}_n) \\ c(\boldsymbol{x}_2, \boldsymbol{x}_1) & 1 & & \vdots \\ \vdots & & \ddots & \\ c(\boldsymbol{x}_n, \boldsymbol{x}_1) & \cdots & & 1 \end{pmatrix}$$

Note that (9) is conditional on the quantities $\mathbf{w}, \sigma^2$ and B, which are known as hyperparameters, since they are controlling parameters that define the behaviour of the model (a much fuller explanation of this can be found in [23]). Following the Bayesian approach, prior distributions are assigned to these where possible. In the case of $\mathbf{w}$ and σ^2, an *improper* prior is assigned, meaning that it is only described as a proportional relationship, i.e.

$$p(\mathbf{w}, \sigma^2) \propto \frac{1}{\sigma^2} \tag{10}$$

which implies an infinite prior variance on $\mathbf{w}$, and a diminishing probability with increasing σ^2. This is also called a *weak prior* because it implies very little information about $\mathbf{w}$ and σ^2. More on improper priors can be found in [25].

The remaining hyperparameter B cannot be marginalised analytically; therefore it will be treated from here as known. It must still be estimated however; this is done by maximum *a posteriori* estimation – this is treated in more detail in [27].

2.2 Posterior Distribution

In order to make predictions about any unknown function value $f(\boldsymbol{x}^{\bullet})$ at a new and potentially previously unseen input point $\boldsymbol{x}^{\bullet}$, the prior distribution of $f(\boldsymbol{x}^{\bullet})$ is conditioned on the training data $\mathbf{y}$ (see e.g. [23, 26]), after which the remaining unknown hyperparameters can are marginalised, i.e.

$$p\left(f(\boldsymbol{x}^{\bullet})\,|\,\mathbf{y}\right)=\int p\left(f(\boldsymbol{x}^{\bullet}),\mathbf{w},\sigma^{2}\,|\,\mathbf{y}\right)d\mathbf{w}d\sigma^{2} \tag{11}$$

The full steps of this integration are lengthy, but can be found in [28]. The marginalisation finally results in a posterior student's t-distribution over $f(\boldsymbol{x}^{\bullet})$, and a student's t-process over the whole function $f(\boldsymbol{x})$, which is given by,

$$f(\boldsymbol{x})\,|\,\mathbf{y}\sim t_{n-q}\{m^{**}(\boldsymbol{x}),\hat{\sigma}^{2}c^{**}(\boldsymbol{x},\boldsymbol{x}')\} \tag{12}$$

where,

$$m^{*}(\boldsymbol{x})=\phi(\boldsymbol{x})^{T}\hat{\mathbf{w}}+\mathbf{t}(\boldsymbol{x})^{T}A^{-1}(\mathbf{y}-\Phi\mathbf{w}) \tag{13}$$

$$c^{*}(\boldsymbol{x},\boldsymbol{x}')=c(\boldsymbol{x},\boldsymbol{x}')+(\phi(\boldsymbol{x})^{T}-\mathbf{t}(\boldsymbol{x})^{T}A^{-1}\Phi)(\Phi^{T}A^{-1}\Phi)(\phi(\boldsymbol{x}')^{T}-\mathbf{t}(\boldsymbol{x}')^{T}A^{-1}\Phi)^{T} \tag{14}$$

$$\hat{\mathbf{w}}=(\Phi^{T}A^{-1}\Phi)^{-1}\Phi^{T}A^{-1}\mathbf{y}$$

$$\hat{\sigma}^{2}=\frac{\mathbf{y}^{T}\{A^{-1}-A^{-1}\Phi(\Phi^{T}A^{-1}\Phi)^{-1}\Phi^{T}A^{-1}\}\mathbf{y}}{(n-q-2)}$$

$$\mathbf{t}(\boldsymbol{x}^{\bullet})^{T}=(c(\boldsymbol{x}^{\bullet},\boldsymbol{x}_{1}),...,c(\boldsymbol{x}^{\bullet},\boldsymbol{x}_{n})),$$

The GP is now fully fitted to the training data. Predictions are made using the posterior mean in equation (13) and confidence limits with (14). A very useful feature of this emulator is that it is actually a tractable expression – the integrals to calculate sensitivity indices can be evaluated analytically. Details of this can be found in [15, 26].

3 Bayesian Treed GPs

A Classification and Regression Tree (CART) is a system used to recursively divide data into classes that are increasingly homogeneous within their divisions in some respect [29]. A "treed GP" divides the input space into regions and fits a GP to each region, so that discontinuities can be handled at region boundaries [20].

An example of a CART applied to 2-dimensional data is shown in Figure 1. The tree starts with the full set of training data. At each split in the tree (known as a "node"), a rule is used of the form $x_i \le s$ to divide the data using a single input variable. If a data point agrees with this rule, it goes to the left "branch". Otherwise, it goes to the right. This process is repeated as many times as necessary on any variables until the data is sorted according to some criterion, at which point the data reaches the ``terminal node" or "leaf" of the tree and is assigned either a category (in the case of a classification tree), or some numerical value (a basic regression tree). In fact, the concept may be extended to fitting more complex regression models at each leaf (i.e. the terminal node contains a parameter set describing a local emulator). This forms the basis for "treed GPs".

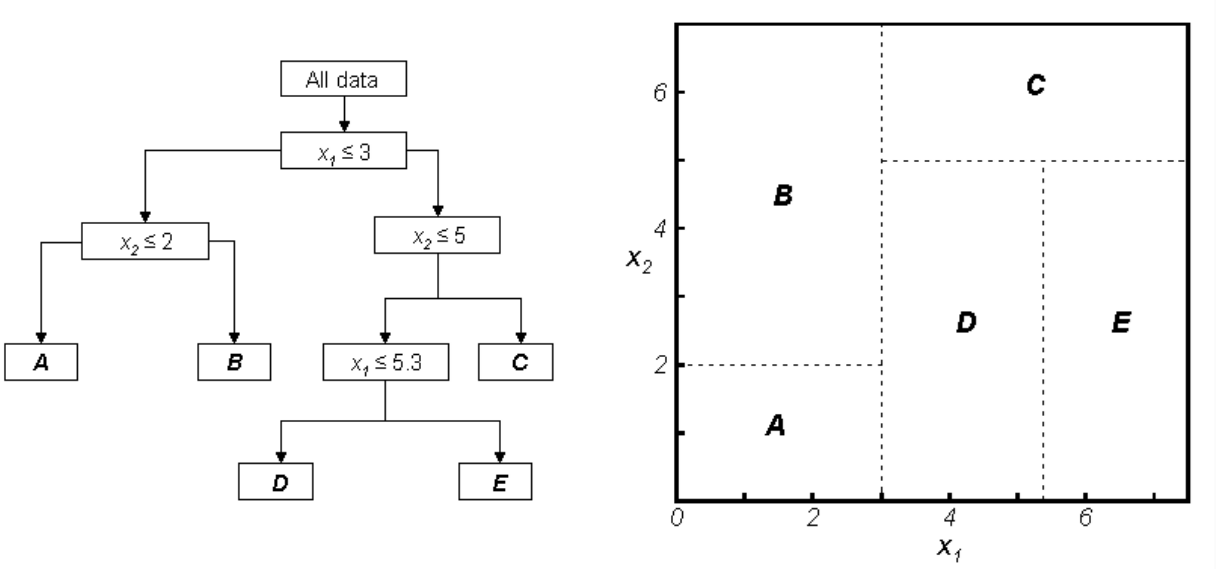

Figure 1: An example of a 2D CART (left) and corresponding divisions of input space (right)

The result of this splitting of data is that the input space is divided into *R* non-overlapping regions, as illustrated in Figure 1 (right). Note that since the splits are performed on one variable at a time, all the partitions are made using lines parallel to the axes. This will be seen to be a possible limitation later on, because when splits are required to be non-parallel it may require a complicated tree to satisfactorily partition the data.

Leaving aside for a moment the issue of what is actually *at* the leaf of the tree, the main problem to overcome is that of constructing a tree that partitions the data to some required level of accuracy, ideally by the simplest means possible. In other words, it is desirable to have a tree that is no more complicated than is necessary to partition the data, to avoid the problem of over-fitting (note that this is based on the principle of Occam's Razor [30]). The simplest approach to "growing" the tree is to use a *greedy algorithm* that optimises some criterion at each split. The problem with the greedy algorithm is that although the splits are locally optimal, there is no guarantee that this will result in a globally optimal solution. The sub-optimality of greedy algorithms is demonstrated by the classic "travelling salesman problem" (see e.g. [31]). Some improvement may be gained by "pruning" the tree, which involves simplifying the tree by removing nodes, based on some strategy such as a cost-complexity criterion or using cross validation. Further methods for tree construction are discussed in [29].

An alternative way of growing CARTs using a Bayesian approach has been suggested by Chipman *et al.* [32], where each leaf contains a Gaussian distribution (i.e. a regression tree), and a prior distribution over models is conditioned on training data to give a posterior (trained) distribution. This was later extended to use linear regression models at each leaf [33]. Most recently, Gramacy and Lee extended the Bayesian CART much further by using it to fit GPs over each region of input space [34]. However, in all cases the principles of the tree construction itself remain essentially the same, being based on Chipman's original paper. For the purposes of clarity therefore, the following sections will discuss the Bayesian approach to CART construction in the context of a simple regression tree.

3.1 Overview

The Bayesian approach to CARTs considers the problem from a probabilistic perspective. Following the usual Bayesian procedure, a prior distribution *over models* is created, which will ultimately be conditioned on training data to produce a posterior distribution. The posterior distribution can then be used to either find the best model, or to use some kind of model-averaging procedure to provide probabilistic estimates.

To implement such a strategy requires a mechanism for specifying distributions over models. In the case of CART, a model is specified by both a tree structure (which will be called T), and a set of parameters $\boldsymbol{\theta}$ that specifies the regression parameters at each leaf, for all leaves. While it is easy to see how distributions can be assigned to numerical parameters, describing a distribution over trees is somewhat less straightforward. In fact, it turns out that the tree distribution does not need to be expressed in a closed form, since Markov Chain Monte Carlo (MCMC) is used to explore the posterior distribution. Therefore, it is only necessary to be able to sample implicitly from a prior tree distribution, which can be done by a series of probabilistic tree-generating rules. This is explained in more detail in Section 3.2.1.

In the case of a simple regression tree, the parameters $\boldsymbol{\theta}$ represent means and variances at each leaf. In particular, the following explanation will use the simple "mean-shift" model described by Chipman as an example [32]. In this model, it is assumed that the data is distributed with a separate mean μ_v for each leaf $\{r_v\}_{v=1}^{R}$ but is governed by a global variance measure σ^2 that is the same for all leaves. This leads to a direct specification of the likelihood function $p(\mathbf{y} \mid X, \boldsymbol{\theta}, T)$. Details will not be given here but can be found in [28, 32].

In order to obtain the posterior distribution, it is first necessary to define a prior distribution over models $p(\boldsymbol{\theta},T)$. This can conveniently be divided such that,

$$p(\boldsymbol{\theta},T) = p(\boldsymbol{\theta} \mid T)\, p(T) \tag{15}$$

which allows the tree prior to specified independently of the regression parameters. This is treated in more detail in Section 3.2.

The prior may now be combined with the model likelihood, which is dependent on the type of model used at each leaf. If the parameter prior has a carefully chosen form, it is possible to analytically marginalise the model parameters, i.e.

$$p(Y \mid X,T) = \int p(\mathbf{y} \mid X, \boldsymbol{\theta}, T) p(\boldsymbol{\theta} \mid T) d\boldsymbol{\theta} \tag{16}$$

Now using Bayes' theorem the posterior distribution over trees can be found up to a proportional constant,

$$p(T \mid X,Y) \propto p(Y \mid X,T)\, p(T) \tag{17}$$

By searching the posterior distribution in Equation (17), the most probable trees may be selected given the supplied training data. For example, if a single tree is required, it might be reasonable to pick the maximum *a posteriori* (MAP) tree - i.e. the tree with the highest posterior probability. In fact, the best option is to use a model averaging strategy; this is discussed further in Sections 3.3 and 3.4.2.

3.2 Prior Specification

The model prior is split into a tree prior and a conditional parameter prior, as noted in Equation (15). These are dealt with separately.

3.2.1 Tree Prior

The tree prior $p(T)$ cannot be expressed in a closed form. However, it is possible to create a stochastic process that produces a series of trees based on specified rules that successively split terminal nodes. Implicitly, this allows sampling from an indirectly-specified prior tree distribution. Since the posterior tree distribution from Equation (17) is explored by MCMC (see later), this definition of $p(T)$ is not a problem in practice. Importantly, the tree prior is independent of the model parameters at the leaves – it only specifies the structure of the trees and the associated splitting rules at this point.

The tree-generating process proceeds as follows. At the very beginning, one starts with an "empty tree", which consists of a single root node. But for the sake of illustration, let it be assumed that a non-empty tree has been created, and it is desired to move to a new tree from this present state. In order to make the first split, it is necessary to choose first, the node $\eta \in T$ to be split (in the case of the empty tree there is of course no choice); second, the variable x_i on which to perform the split; and last, the value s_η which dictates the value of x_i at which to perform the split. With the new splitting rule, two "child nodes" are created. This new tree can be considered as a sample from $p(T)$, and the same process can be applied repeatedly to generate a series of samples.

To deal with the selection of the node to be split, Chipman suggests a splitting rule p_{SPLIT} such that,

$$p_{\text{SPLIT}}(\eta, T) = \alpha(1+\delta_\eta)^{-\beta} \tag{18}$$

where δ_η is the depth of node η (the number of splits above η), and α and β are parameters that can be set to control the shape of the prior distribution. Since this rule is a decreasing function of δ_η, it encourages the growth of "bushy" trees (trees whose terminal nodes do not vary too much in depth) which discourages over-fitting. The shape parameters can be adjusted empirically by generating samples from $p(T)$ and plotting histograms of criteria of interest, for example, by examining the number of terminal nodes.

Now, given that a split has been made, the variable x_i on which to make the split, and the split value s_η must be chosen. This is done by assigning a further rule $p_{\text{RULE}}(\rho|\eta,T)$, which dictates the probability of the splitting rule ρ by a uniform distribution on the available x_i, and a further uniform distribution on the available split values, conditional on each x_i. "Available", in this context, refers to split variables and split values that do not lead to empty leaves. Note that s_η could take any value, but there are only a finite number of possible data splits – for example, the midpoints between all the data values along a given x_i. Therefore p_{RULE} is a discrete distribution. By extension, this implies that $p(T)$ is also a discrete distribution, albeit over a very large number of possible trees for most data sets.

3.2.2 Parameter Prior

In choosing the parameter prior $p(\boldsymbol{\theta},T)$, it is convenient to choose a form that allows the analytical marginalisation described in Equation (16). In this case, the conjugate prior of the Gaussian likelihood function is used, which is a normal distribution over each μ_v and an inverse-gamma distribution over σ^2. This now yields an analytical expression for the marginal likelihood of the trees after marginalising the parameters (by performing the integral in Equation (16)); see [20] for the full expression. The parameters of the parameter prior can be chosen with guidance from the training data – see [32]. Therefore, the two terms of the posterior distribution in (17) are now available.

3.3 Searching the Posterior Distribution

Now that the prior distribution and likelihood have been specified, the posterior distribution of trees $p(T|X,Y)$ (having integrated out the model parameters) is expressed by Equation (17), but only up to a constant of proportionality. Additionally, an exhaustive search of all models is usually impossible due to the sheer number of possible trees. However, the Metropolis-Hastings (MH) algorithm [35, 36] provides answers to both of these problems.

The MH algorithm is a form of Markov Chain Monte Carlo (MCMC) estimation, which allows sampling from a distribution even when the normalising constant is unknown. The algorithm uses a form of rejection sampling based on an "acceptance ratio" to sample the posterior, favouring samples of higher probability. The very nice property of the MH algorithm is that since the acceptance probability is a ratio of posterior probability values, the normalising constant of the posterior actually cancels out. Therefore it

is only necessary to know the posterior distribution up to a constant of proportionality, as in Equation (17). Additionally, since the algorithm will *always* accept jumps to points that have a higher probability in the sample distribution, the MH search naturally gravitates towards areas (trees, in this case) of higher probability. A description of the full workings of the algorithm is outside the scope of this paper, but a good explanation can be found in [30].

In order to calculate the acceptance ratio, it is necessary to have some jumping distribution q that specifies the probability of jumping to a new tree, given the present tree state. Chipman suggests a distribution that may be defined implicitly by randomly selecting one of four rules (although other rules could easily be conceived), which are as follows,

1. GROW: Randomly pick a terminal node and use the p_{RULE} rule defined in the prior to split it into two child nodes.
2. PRUNE: Randomly pick the parent of two terminal nodes and collapse it into a terminal node by removing the splitting rule.
3. CHANGE: Randomly pick an internal (not terminal) node and randomly re-specify the splitting rule using p_{RULE} from the prior.
4. SWAP: Randomly pick a parent and child node that are not terminal nodes. Swap the splitting rules, unless the other child node has the same rule, in which case swap the splitting rule of the parent node with that of both child nodes.

The rules defined above give a reversible Markov chain, because the GROW and PRUNE rules are reversible counterparts. Furthermore, the CHANGE and SWAP rules are reversible when compared to themselves, therefore every possible move has a counterpart in the opposite direction.

3.4 Extension to Treed GPs

The previous explanation has focused on the use of a CART coupled with a simple mean-shift model to assign means and variances at each leaf of the tree. Although from a high-level perspective the method remains essentially the same when a GP is fitted at each leaf (since a GP is defined by a set of parameters $\boldsymbol{\theta}$ in the same way as the mean-shift model), in practice the details of its implementation are somewhat more complex. Since the principles of Bayesian CART have been shown in depth, it is not considered necessary to provide a detailed treatment of the construction of treed GPs, however, an outline will be given, and the main differences explained. For a full explanation, the reader is referred to [34].

The idea of treed GPs was introduced by Gramacy and Lee [20], who are also the authors of the R package "tgp" [37], which incorporates all the features described here, including the ability to perform UA and SA on both treed and standard GPs. This package was used to perform all the treed GP analyses in this article.

3.4.1 Prior Specification

At each of the R leaves $\{r_v\}_{v=1}^{R}$ of the tree a GP is defined. Gramacy *et al.* propose a GP for each region r_v of the same form used in Section 2.1, except further parameter uncertainty is accounted for by using a hierarchical specification of the hyperparameters. Rather than specifying an uninformative prior over the weights $\mathbf{w}$ and the variance scale σ^2 (refer back to Equation (10)), normal and inverse-gamma distributions are used respectively, the hyperparameters of which are also assigned distributions. This hierarchical model and its priors are specified as,

$$\mathbf{y}_v \mid \mathbf{w}_v, \sigma_v^2, A_v \sim \mathrm{N}_{n_v}(\Phi_v \mathbf{w}_v, \sigma_v^2 A_v) \tag{19}$$

$$
\begin{aligned}
&\mathbf{w}_\nu \mid \sigma_\nu^2, \tau_\nu^2, W, \mathbf{w}_0 \sim \mathrm{N}_m(\mathbf{w}_0, \sigma_\nu^2 \tau_\nu^2 W) \\
&\mathbf{w}_0 \sim \mathrm{N}_m(\mu, \mathbf{C}) \\
&\tau_\nu^2 \sim \mathrm{IG}(\alpha_\tau / 2, q_\tau / 2) \\
&\sigma_\nu^2 \sim \mathrm{IG}(\alpha_\sigma / 2, q_\sigma / 2) \\
&W^{-1} \sim \mathrm{W}((\rho_\nu V)^{-1}, \rho)
\end{aligned}
\tag{20}
$$

where W is an $m \times m$ matrix (m is the number of covariates in the training data), IΓ and Ω are the inverse-gamma and Wishart distributions respectively and the remaining undefined hyperparameters are treated as known. The covariance function is the same as that used in Section 2.1. Rather than estimating the roughness parameters B in the covariance function however, prior distributions are assigned to the diagonal elements of B, giving a fuller account of uncertainty. Details of these priors will not be presented here, but can be found in [20].

The hyperparameter priors from Equation (20) together constitute the conditional hyperparameter prior $p(\boldsymbol{\theta}|T)$. Additionally, the parameter likelihood is easily expressed from Equation (19). The tree prior $p(T)$ is the same used in the mean-shift CART from Section 3.2.

3.4.2 Posterior Search

Since the form of $p(\boldsymbol{\theta}|T)$ is considerably more complex than that used by Chipman *et al.*, numerical integration is used to marginalise parameters. In fact, the treed GP approach goes one step further, by using Reversible-Jump Markov Chain Monte Carlo (RJ-MCMC) sampling [38] (a form of MCMC that can jump between parameter spaces of different dimensions) to fully integrate out tree dependence as well, i.e.,

$$
p(Y \mid X) = \int p(Y \mid X, \boldsymbol{\theta}, T)\; p(\boldsymbol{\theta} \mid T)\; p(T)\; d\boldsymbol{\theta} dT \tag{21}
$$

This is now a formal model-averaging approach, since all uncertainty has been accounted for, not only in the parameter estimation, but also in the tree estimation (compare to Equation (16), where tree dependence is still present). The nice thing about averaging over trees is that although for any given tree the posterior mean will be discontinuous over leaf boundaries, when model averaging is used it will actually smooth out at the boundary, as a result of averaging over many possible models. Importantly for this investigation, however, it still retains the ability to model bifurcating data at leaf boundaries.

In order to sample from the joint posterior of $(\boldsymbol{\theta},T)$ (remembering that parameters have not been analytically marginalised here), MCMC is used to explore the model space, consisting of jumps between parameter sets and trees. Moves between trees and parameter sets are treated separately, so the sampling performs a move to a new set of parameters, then a new tree, then a parameter move again, and so on. In other words, samples are being drawn alternately from $p(\boldsymbol{\theta}|T)$, then $p(T|\boldsymbol{\theta})$. In order to do this, it is necessary to use a mixture of forms of MCMC.

For moves in parameter-space, Gibbs sampling [39] is used where possible. This allows a faster sampling of parameters in many cases. The necessary full conditional distributions can be derived for all parameters except those used in the covariance matrix (roughness parameters), therefore Gibbs steps may be used [34]. In the case of the roughness parameters, the MH acceptance ratio is used. The steps are performed over all parameters $\boldsymbol{\theta}_\nu$ for a single tree leaf, then repeated for all leaves, generating a set of parameters conditional on a tree $(\boldsymbol{\theta}|T)$.

Once a move to a new full set of tree parameters has been performed, it is necessary to move to a new tree, using a stochastic tree-jumping process similar to that used by Chipman *et al*. However, since the size of the parameter space (i.e. the length of the $\boldsymbol{\theta}$ vector) will change as a result of adding or subtracting leaves, in order to formally integrate out tree dependence it is necessary to use RJ-MCMC, which is a generalisation of the MH algorithm that can cope with jumps between parameter spaces of different dimensions, by using latent variables and a modified acceptance ratio. The details of RJ-MCMC will not

be discussed here, however the interested reader can refer to a good tutorial [40], which is perhaps a little more accessible than Green's original paper. The acceptance ratios for the moves used in the treed GP search can be found in [20].

3.4.3 Limiting Linear Models

One further additional feature used in the *tgp* package is the concept of Limiting Linear Models (LLMs). The response of a model in part (or all) of the input space may be linear, in which case fitting a GP is unnecessary, as well as risking over-fitting and numerical difficulties. In the case of a region where the data is sufficiently linear, the GP jumps to the LLM, which is a special case of Equation (19), such that,

$$\mathbf{y}_\nu \mid \mathbf{w}_\nu, \sigma_\nu^2 \sim \mathrm{N}_{n_\nu}(\Phi_\nu \mathbf{w}_\nu, \sigma_\nu^2 I) \tag{22}$$

where I is the $n_\nu \times n_\nu$ identity matrix. This is actually the Bayesian linear model considered by Chipman *et al.* as an extension of basic Bayesian CART [33]. The switch is implemented by a Boolean operator that chooses the GP or the LLM based on the estimated roughness matrix B, and a nugget (noise) parameter. The Boolean identifies situations where the roughness parameters are small (smooth response) and the estimated noise is high, corresponding to a near-linear GP. When incorporated into the prior distribution, this allows a switch to the LLM. The full details of this will not be discussed here, but can be found in [41]. It will however be seen to be a significantly useful feature in the following case studies.

3.4.4 Application to UA/SA

The fitted treed GP model is not a tractable expression that can provide analytical estimates of sensitivity indices, as in the case of the standard GP. However, predictions can be made relatively cheaply, which allows the use of Monte Carlo techniques. The approach of Gramacy *et al*. [42] is that at each tree (and corresponding parameter set) that is visited during the posterior search, output predictions are made over a large selected set of input locations based on a Latin hypercube design. For each set of predictions, the necessary integrals may be evaluated conditional on the predicted output by Monte Carlo integration, using the standard estimators [10, 43]. Although this approach is less elegant than the analytical methods used in Section 2, it does have the advantage that UA and SA estimates now account for uncertainty over possible models, and all hyperparameters are estimated in a fully Bayesian fashion. Additionally, rather than calculating sensitivities as ratios of expected values, the numerically-integrated sensitivity indices here are true estimates of the S_i. Full details of the numerical integration are found in [42]. In short, the treed GP can estimate all of the common variance-based sensitivity measures, as well as means and variances of the output.

3.5 Summary of Treed GPs

The treed GP can be constructed with essentially the same Bayesian approach to mean-shift CART models, albeit with the introduction of some new features, such as the hierarchical GP model and the necessary RJ-MCMC to integrate out tree dependence. The treed GP method relies heavily on numerical integration, but in terms of computational expense it has been found by the authors to be no more demanding than the single GPs described previously. This is for three reasons: first, when a split is identified, two smaller GPs are trained rather than one large one, which presents an immediate computational saving. Second, when the LLM is invoked, potentially large savings can be made since it precludes the necessity to invert the covariance matrix. Finally, although hyperparameters are estimated by MCMC, the standard GP also has to use an optimisation algorithm for roughness parameters, which is comparably time-consuming.

It should again be emphasised that the GP-based methods are only suitable for small training data sets (i.e. very computationally-expensive models), since the cost of training the GP is roughly $O(n^3)$. A test of the treed GP against the standard GP is now performed using first, a Duffing oscillator; and second, a bifurcating FE model.

4 Example: Duffing Oscillator

The Duffing oscillator is a very well-known example of a nonlinear system in structural dynamics. It is a sinusoidally-forced single degree of freedom system with a cubic stiffness term:

$$m\ddot{y} + c\dot{y} + ky + k_3 y^3 = P\cos(\omega t) \tag{23}$$

where m is a mass, c is a damping coefficient, k and k_3 are stiffness constants, P is the forcing amplitude and ω is the forcing frequency. $y, \dot{y}$ and $\ddot{y}$ denote the displacement, velocity and acceleration respectively.

The frequency response function (FRF), in the case of k_3=0, is single-valued at any frequency. However, when $k_3 \neq 0$, at certain frequencies the response of (23) can have three possible amplitudes, since they are the solutions of a cubic equation. Of these three, one amplitude is unstable and is never achieved in practice, however, as the system approaches a steady state it snaps to one of the two remaining amplitudes, either the high amplitude or the low one. Whether the system ends up in the high or low amplitude is determined by the initial conditions of the system: the initial displacement y_0 and the initial velocity $\dot{y}_0$. For a more detailed description of the response of Duffing oscillators the reader is referred to [44].

The amplitude response therefore provides a taxing system with which to test the GP and treed GP methods. Essentially the steady state amplitude can be seen as a function of the two initial conditions. But since the amplitude can only take on either the high value or low value, there will be bifurcations in the response. Examples of this are illustrated in the following section. In order to investigate response surfaces of varying complexity, several areas of parameter space were sampled, which will be referred to as cases 1-3. These are explained in the following pages. The values of parameters used here are m=1, c=20, k=1000, $k_3=5\times10^9$, P=10 and ω=170. These were chosen to show interesting regions of the parameter space, but were otherwise arbitrary.

4.1 Cases 1 and 2

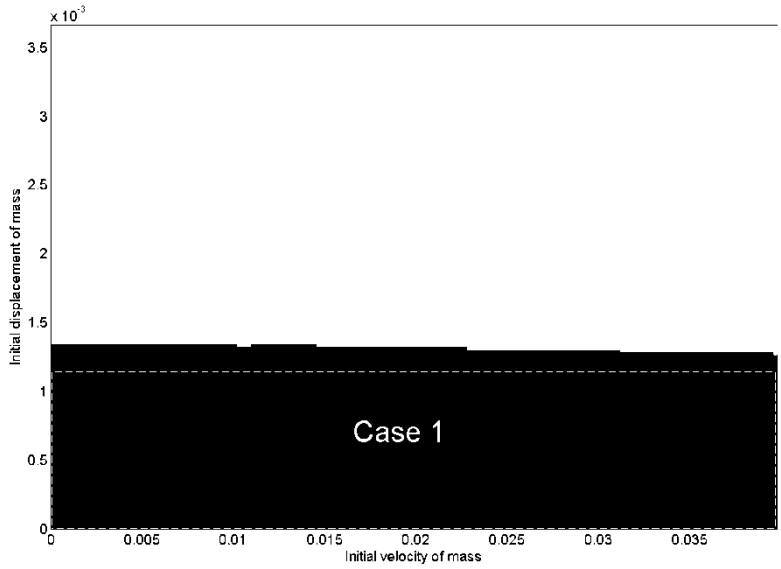

Figure 2: Case 2, single bifurcation (covering all parameter space shown here); case 1 region illustrated. White=high, black=low.

Figure 2 shows a high-resolution amplitude plot of two of the parameter regions investigated, cases 1 and 2. Case 1 was chosen to be the simplest possible response surface – an area where the amplitude is completely insensitive to parameter variations. As such, the sensitivity data and model fits are not illustrated here since it is a trivial problem. Case 2 provides a more interesting test. Expanding the range of initial displacement reveals a region of high amplitude response and a bifurcation between the low amplitude (blue) and high-amplitude (red) regions.

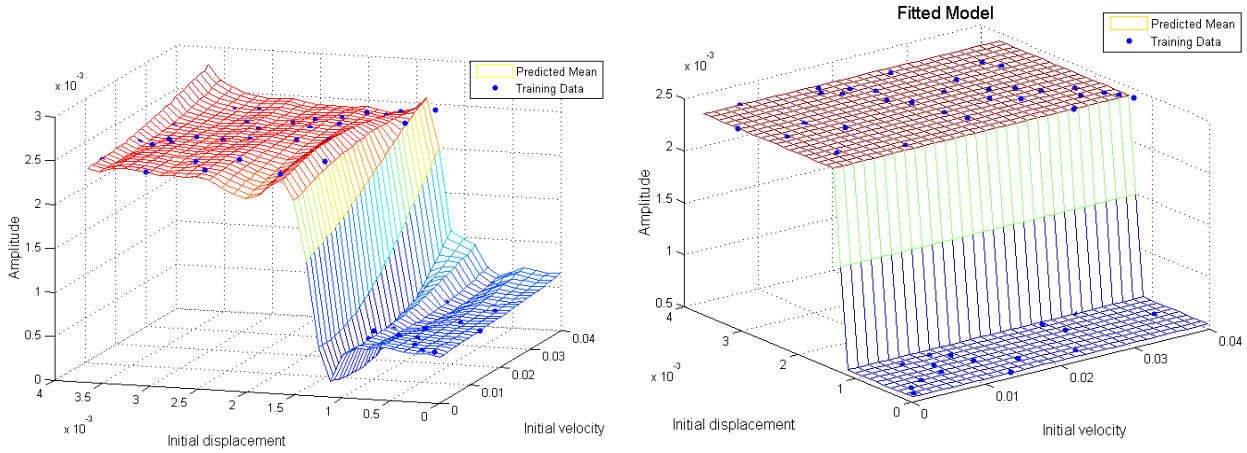

Figure 3: Case 2 fitted models; GP (left); Treed GP (right)

A set of 49 training points was drawn with a maximin Latin hypercube sampling strategy (see e.g. [24]). This was then used to train both the GP and treed GP models. Figure 3 shows surface plots of the fitted models and training samples. It is immediately clear that the treed GP models the data in a much closer way to the real response. The GP, on the other hand, is guilty of introducing a type of Gibb's phenomenon in order to accommodate the steep bifurcation, because the GP's roughness parameters are necessarily constant over the whole model. The bifurcation forces the roughness to be high in the direction of initial displacement, but this value is not suitable in the perfectly smooth regions on either side, causing undulations in areas of sparse training data. Conversely, the treed GP can snap to the linear model in the high and low regions, and easily model the bifurcation with a two-leaf tree. That the bifurcation does not appear perfectly vertical in Figure 3 (b) is due only to the resolution of predictive points used to plot the surface. The MAP tree for the treed GP is not illustrated here since it has only a single split at y_0=0.00122.

Figure 4 shows a main effect plot of the input parameters for case 2. Both models capture the variation of the output with respect to each input quite well. The treed GP, however, captures the linearity of the high and low regions, and the bifurcation with more accuracy. The main effect of initial velocity is virtually flat in both models (since the bifurcation is not exactly parallel to the velocity axis it would not be expected to be *perfectly* flat) but the GP appears to show a little too much sensitivity, which is reflected in the slight gradient in the velocity direction in Figure 3.

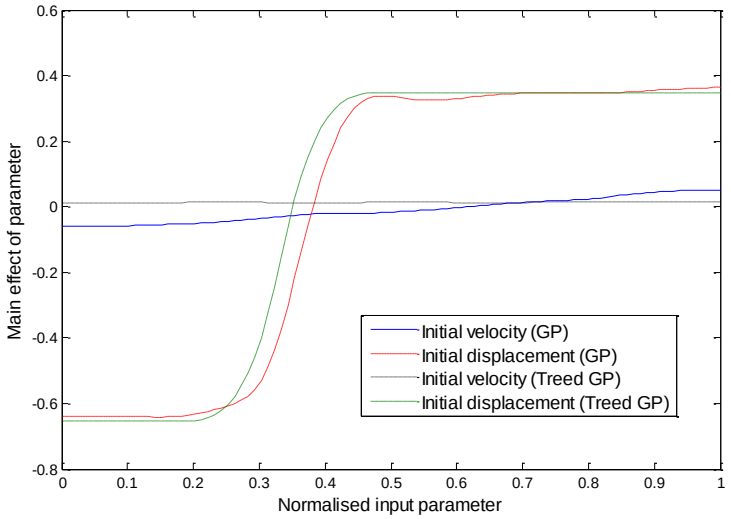

Figure 4: Main effect plots (case 2) GP and treed GP.

4.2 Case 3

Figure 5 illustrates the parameter space of case 3, which is an enlargement of the parameter space in case 2. This reveals several bifurcations, including one that is not perpendicular to either axis and also curved. This poses immediate problems for even the treed GP, because divisions of input space must be made perpendicular to the axes. An initial LHS sample of 49 training points was used to train the model, but initial analyses suggested that more training data was needed to model the complex response. The sample was therefore increased to 100 training points, the results of which are shown in Figure 6.

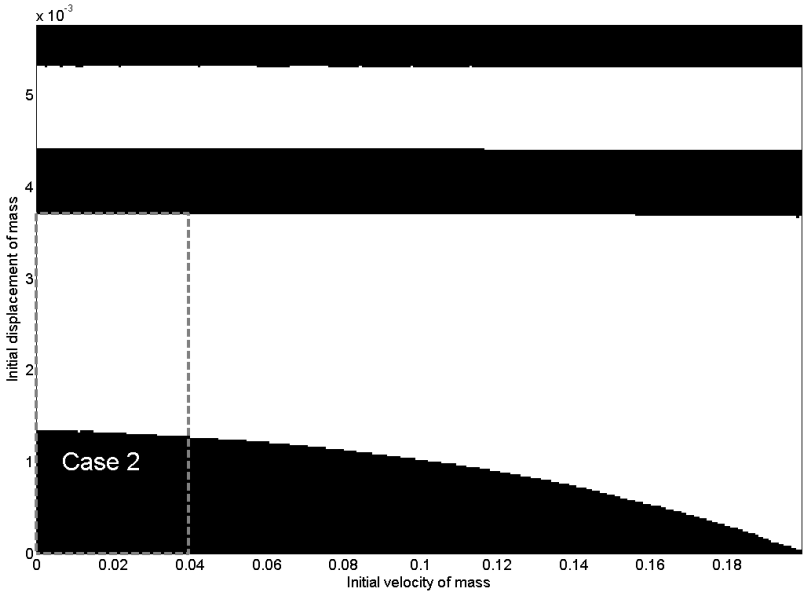

Figure 5: Case 3: multiple bifurcations (covering all parameter space shown here); case 2 region illustrated. White=high, black=low.

As in case 2, the GP cannot faithfully reproduce the response surface, introducing fluctuations in the surface as a result of the several bifurcations. It is clear that a standard GP is not suitable for modelling

this kind of surface (which is a well established fact). The treed GP, on the other hand, does a reasonable job. The bifurcations in the region of y_0=0.005 are captured quite well. However, the treed GP struggles with the curved bifurcation and tries to fit a linear surface in that region. This appears to be partly due to a lack of training data in that area; Figure 7(a) shows that in the lower right region there are few training points and subsequently the predictive uncertainty is high. However, this serves to illustrate one of the advantages of GPs: despite a poor fit, the high uncertainty in prediction is recognised and can be used to suggest locations for further training data. The MAP tree is shown in Figure 7(b) with the six leaves (regions in Figure 7(a)) as circled numbers and the variable splits shown at each branch. It is evident that a considerably more complex tree is required here than for case 2, and that ideally the tree should be even more complex than shown to more closely model the curved bifurcation.

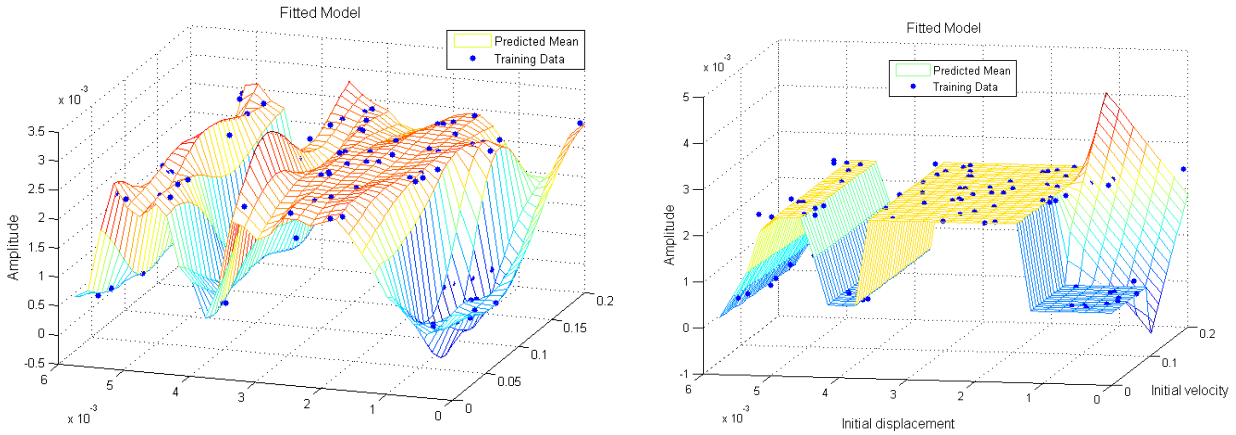

Figure 6: Case 3 fitted models; GP (left); Treed GP (right)

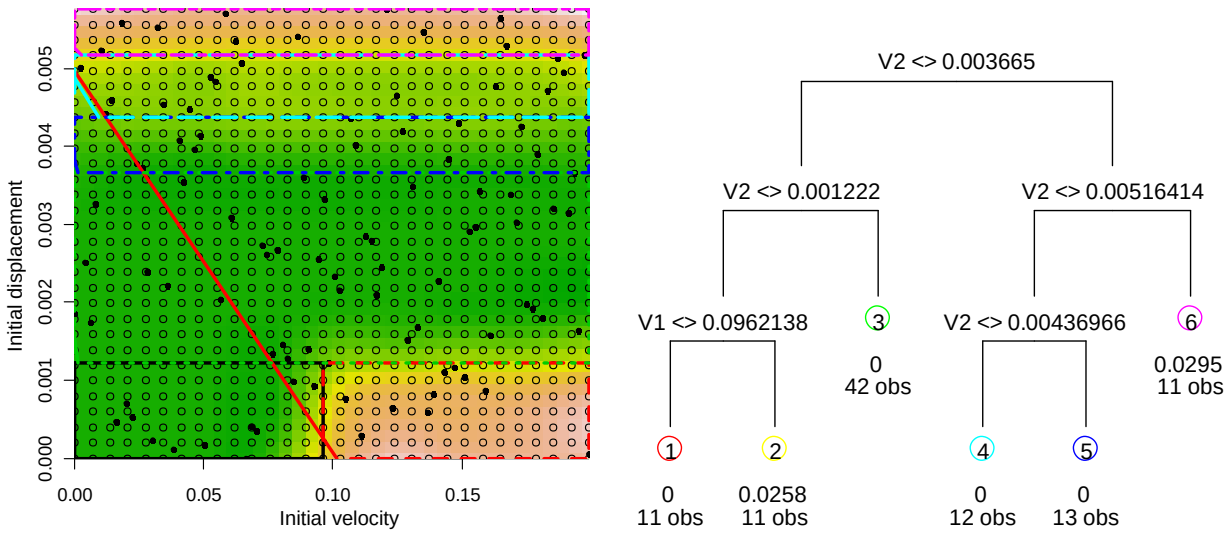

Figure 7: Case 3 predictive uncertainty and tree divisions (left, training data marked as dots, predictive points marked as circles); corresponding tree (right, V1=initial velocity, V2=initial displacement; circled numbers are leaf regions; "obs" denotes number of observations in that leaf; the other number is the MAP estimate of σ^2.

Model uncertainty is also reflected in the estimates of main effects and sensitivity indices. Figure 8 shows the main effect plots of both models. The main effect lines agree to a large extent, though the treed GP captures the large high-amplitude region more accurately. Figure 8(b) shows the 95% confidence intervals of the main effect plots. The uncertainty is markedly higher in the initial velocity line. This could be due to the region of high uncertainty discussed previously. Another effect of this is that although the y_0 main

effect has very small error in most regions, for the lowest values of y_0 the uncertainty increases substantially.

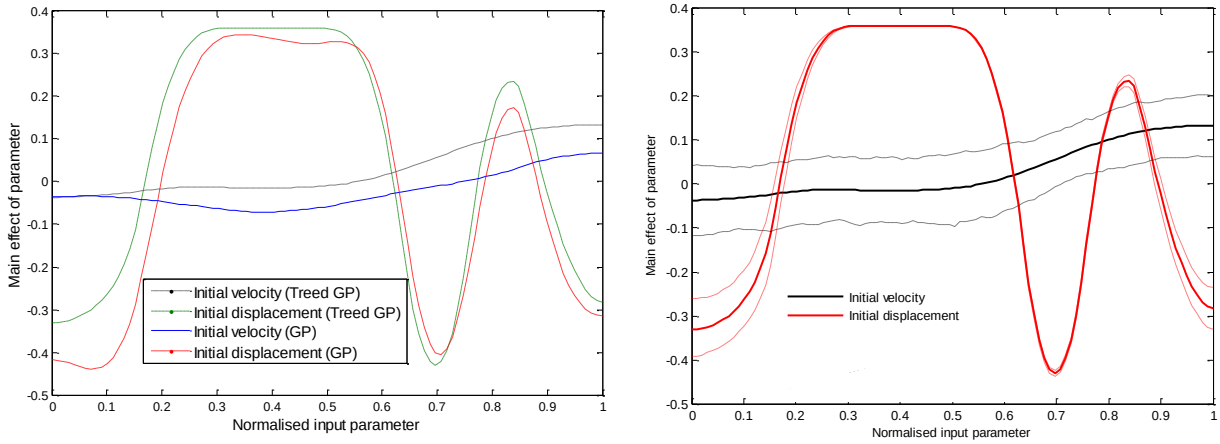

Figure 8: Case 3, posterior means of main effects: both models (left); treed GP with 95% confidence interval (right)

Posterior estimates of sensitivity indices are shown in Figure 9 for both the standard and treed GPs. The estimates are quite similar for the MEIs, though perhaps the most notable difference is that the TSI of initial velocity is somewhat higher in the treed GP model, suggesting more interaction variance between the two parameters. This is again possibly a result of the poorly-fitted region in the treed GP. Nevertheless the sensitivity indices are quite similar overall.

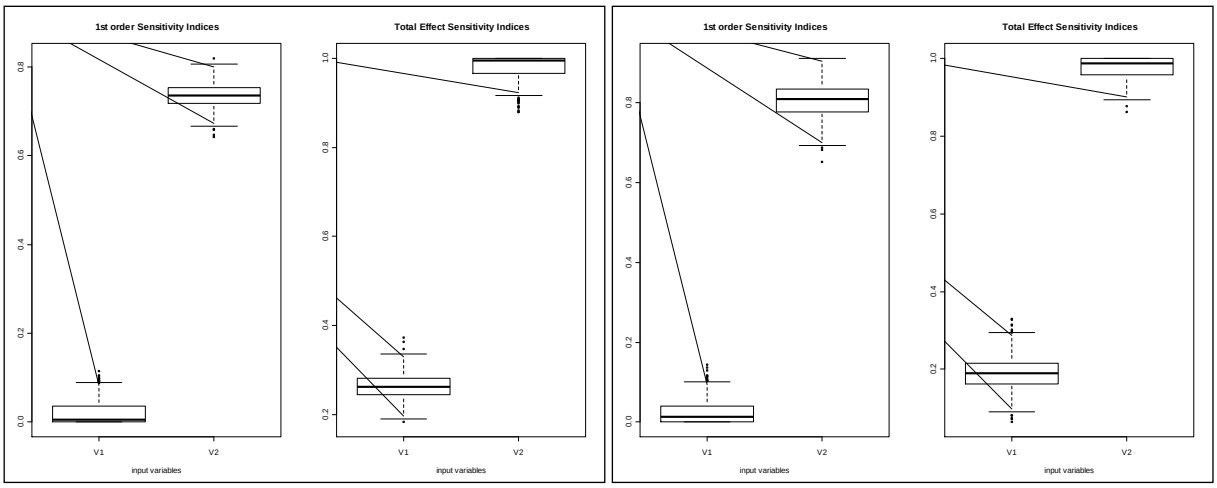

Figure 9: Posterior estimates of sensitivity indices: GP (left, V1=initial velocity, V2=initial displacement); treed GP (right)

5 Example: A Discontinuous FE Model

A second case study here demonstrates the use of the treed GP on a practical example of a discontinuous model, which is that of a rigid-stent prosthetic heart valve. This example follows from previous investigations by the authors into uncertainties in a finite element model of the human aortic valve [1, 45]. "Conventional" bioprosthetic valves (still widely in use) typically consist of three valve leaflets, usually harvested from pigs, mounted on a rigid stent. This is in contrast to the natural valve, which is connected to the flexible aorta. The rigid stent creates a system that is discontinuous, since, during the opening movement, the leaflet is forced to reverse its curvature and snaps though to its "open" position. Figure 10

illustrates this opening process on one of the leaflets. The model uses a constitutive soft tissue material model (see [1] for details) and is opened by a simple pressure load (measured here by a "scale factor").

Since the leaflet is an example of a buckling or snap-through problem, discontinuities were expected to occur as a result of varying the loading on the structure, and potentially the material properties of the leaflet. To keep the analysis relatively simple, two parameters were picked to vary - the pressure exerted on the leaflet, and the post-transition modulus of the soft tissue material model (a key material parameter).

100 points were used to train the emulators, using the usual maximin LHS strategy. This is a large amount of data for two dimensions, so the effect of reducing the number of training points is also investigated in Section 5.3 as a matter of interest. Full details of the model (including the input deck used for the software, LS-Dyna) can be found in [28].

5.1 Model Outputs

Figure 10 shows the movement of the leaflet over 12ms for a mid-range pressure load. This represents the movement of the leaflet for any area of parameter space where it reverses its curvature (i.e. the valve opens). It is clear that the movement of the leaflet is nonlinear over time, since at the point of reversal-of-curvature (ROC) the leaflet accelerates towards its reversed state. This is illustrated in more detail in Figure 11(a), which shows that the initial application of pressure is not sufficient to displace the leaflet by anything but a very small amount (i.e. in the first 5ms of the graph). However, when the leaflet reaches a critical pressure the displacement very quickly increases to the point where the leaflet has reversed its curvature. There are then oscillations due to the lack of material damping, although these do not affect the results to a significant degree. The critical pressure occurs at different times depending on the pressure scaling factor used for that run of the model. For the lowest pressure values the valve does not open at all.

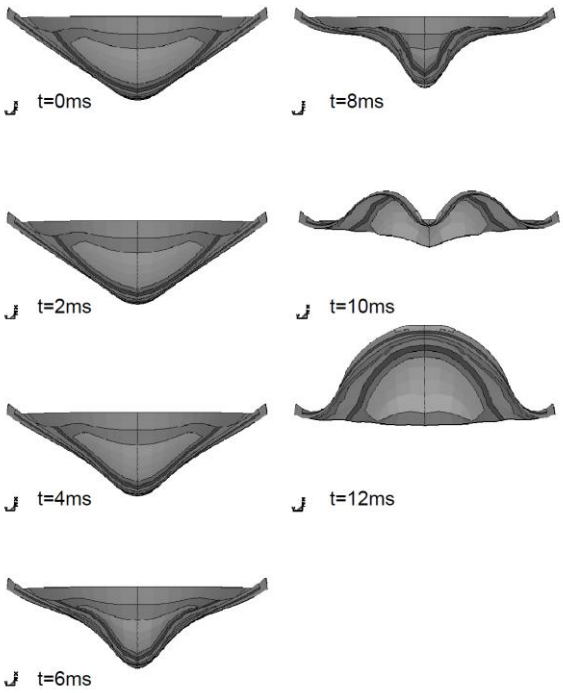

Figure 10: Top view of the opening process of the rigid-stent leaflet model over 12ms

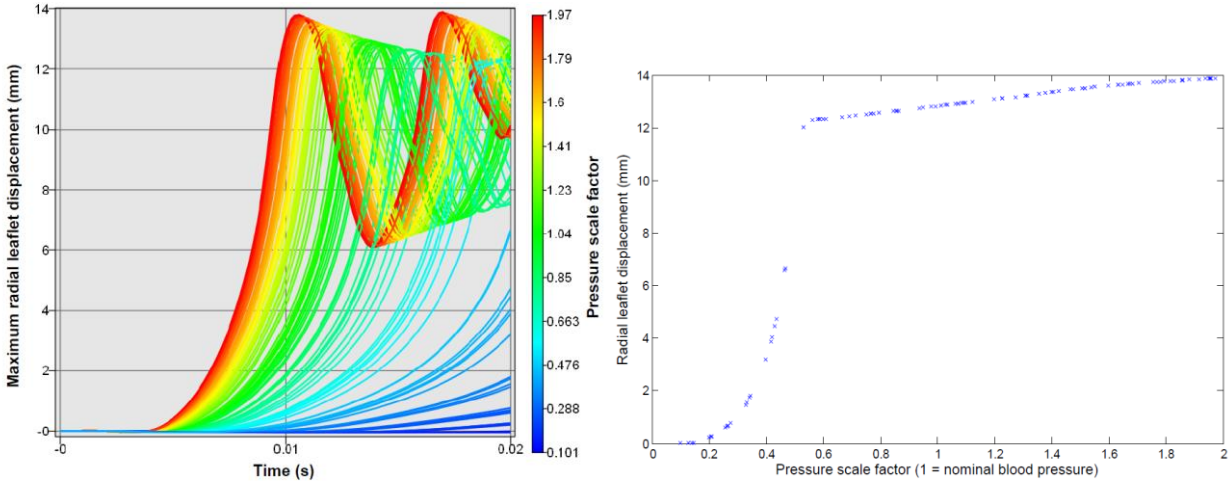

Figure 11: Radial leaflet displacement: (a) history plot for all runs (colours represent pressure values); (b) scatter plot of maximum leaflet displacement against pressure.

Examining a raw data plot of the maximum leaflet displacement against pressure (Figure 11 (b)), one thing that is immediately evident is that the material parameter (post-transition modulus) does not seem to be affecting the output in any way, since there is no scatter to the data in this plot. Instead, there is a very clear nonlinear relationship with pressure. Displacement increases in a kind of exponential fashion with pressure, representing the diminishing resistance of the leaflet as it reaches the ROC point. At a pressure factor of around 0.5 the pressure is sufficient to completely reverse the leaflet and any further displacement is due to stretching of the leaflet, which is roughly linear with increased pressure. This response does not constitute a discontinuity (although probably a discontinuity in the first derivative), but will be an interesting test of the GP given the abrupt changes in smoothness.

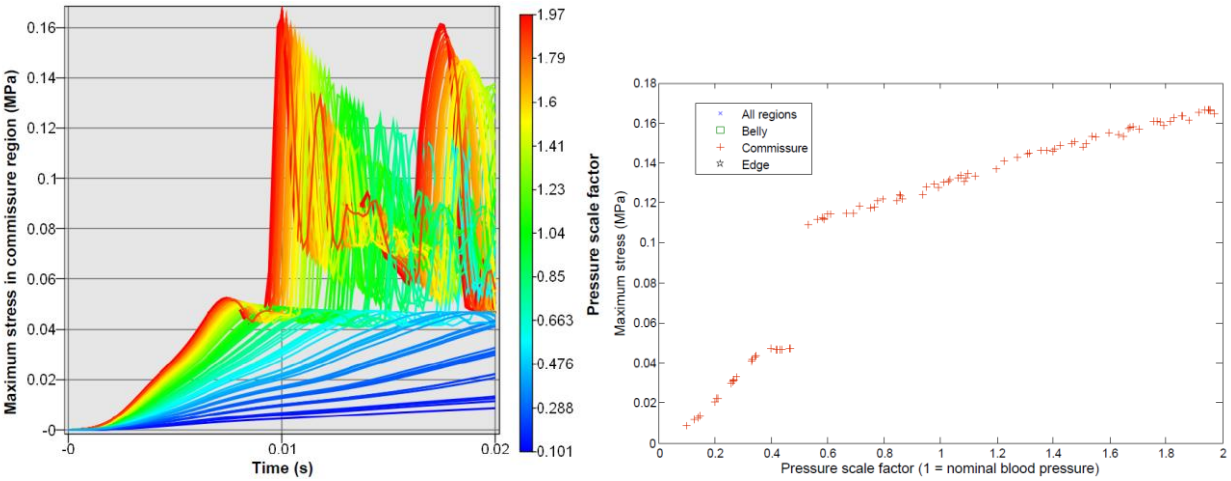

Figure 12: Stress in commissure region: (a) history plot for all runs (colours represent pressure values); (b) scatter plot of maximum stress against pressure.

Turning to the variation of stress yields some very interesting response plots. History plots of commissure stress are shown in Figure 12(a). There is a very sharp peak in stress that occurs at the point where the leaflet reverses its curvature (the earliest of these is at about 0.01s). This peak does not occur if the leaflet does not reverse its curvature, and since ROC occurs for a very small change in pressure, the response of *maximum* stress actually results in a discontinuity. Figure 12(b) shows this trend: at a pressure factor of around 0.5 there is a clear discontinuity at the point where pressure is just enough to open the valve.

The stress responses in other areas also produce discontinuities, however the response of commissure stress is the clearest. This and the displacement response will now be used to train emulators for both the standard and treed GPs.

5.2 Fitting Emulators

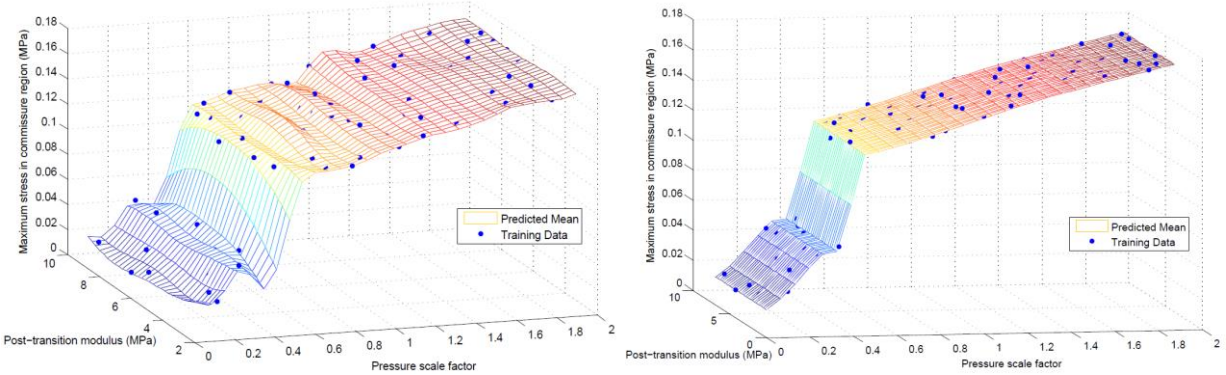

Figure 13: Posterior mean of maximum stress in commissure region with training data overlayed (dots): (a) GP; (b) Treed GP (split at scale factor = 0.467).

The displacement and stress responses are now used to train the GP and treed GP emulators. The intention here is to examine the ability of both emulators to model discontinuous data from an FE model. Beginning with the clearest discontinuity, that of the commissure stress, Figure 13 shows the fitted emulator means from both the treed and standard GPs with the training data overlaid. The difference in quality of fit is immediately evident: similar to the Duffing oscillator response in Section 4, the standard GP is constrained by the necessity of assigning the same length-scales (roughness parameters) to all of the input space. As a result, it introduces oscillations close to the discontinuity that should not exist. Note that this is also with 100 data points for a two-dimensional input space, which might be considered a dense training set for a GP, if discontinuities were not known to exist.

Turning now to the treed GP (Figure 13(b)), the fit is considerably better. The MAP tree here uses a single split at a pressure scale factor of 0.467, which means the discontinuity is followed with much greater accuracy, since it occurs at the leaf boundary. The linear region above this value is assigned a limiting linear model (LLM), which is appropriate given the very linear training data. Below the discontinuity the data is assigned a GP, which can easily follow the nonlinearities in this region. Figure 14 shows the same plot but viewing only variation with pressure. Here the 5\% and 95\% quantiles are shown, which correspond to roughly two standard deviations either side of the mean. The confidence intervals are very narrow, particularly in the GP area. On the right side of the discontinuity the confidence intervals are somewhat wider, and of a constant width due to the LLM that has been fitted, which assigns a constant posterior variance to all points. Overall it must be said that the treed GP provides a very marked improvement over the standard GP, all the more given that the discontinuity is parallel to an axis, which is very easy for the tree to deal with.

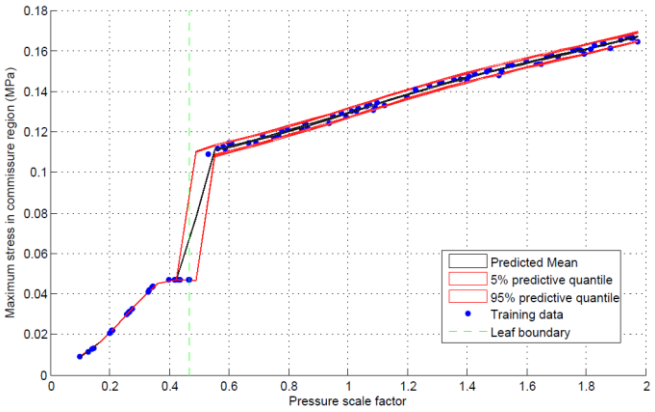

Figure 14: 2D view of treed GP fit to commissure stress data. Confidence intervals and leaf boundary of MAP tree shown.

Figure 14 raises a further possible advantage of the treed GP, which is in the context of extrapolation. For predictive points outside of the range of the training data, both the treed and standard GPs will tend to veer back to their prior means, which are linear regression fits through the training data. In the case of the standard GP this regression fit is taken over the *whole* set of training data. Taking the data in Figure 13 as an example, it is clear that if a flat surface is fitted to all the data, it not going to be particularly close to *any* of the training points. Since the GP will necessarily revert to this prior mean outside of the range of training data, even for predictive points that are a small distance away from the training data it will provide predictive estimates that are not intuitively realistic. On the other hand, the treed GP can take different prior means for different leaves of the tree. For example, in the linear region to the right of the discontinuity, predictions at higher pressures will follow the linear trend of that leaf, and not revert to a global prior mean (as in the case of the GP), therefore following the *local* trend rather than the global trend. Although extrapolation of data should always be treated with caution, the treed GP should produce predictive estimates that are more in line with intuition than the standard GP, particularly when a LLM is used.

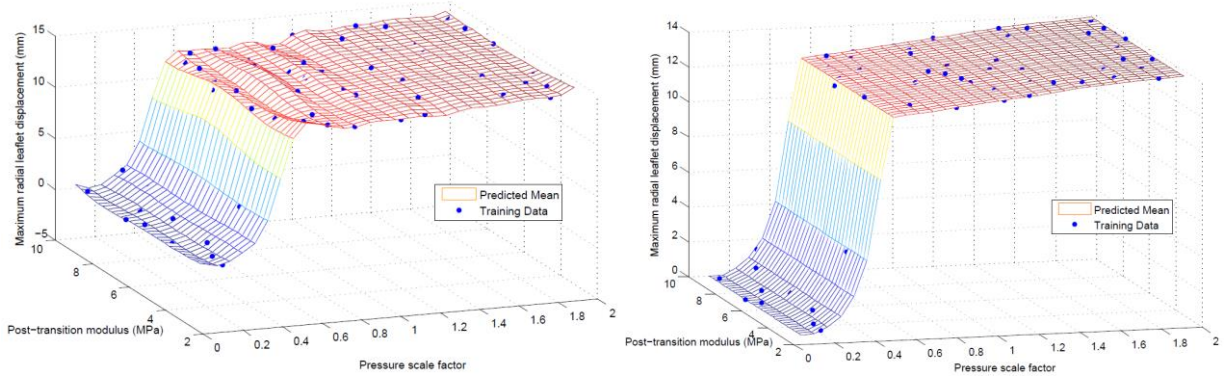

Figure 15: Posterior mean of maximum leaflet displacement with training data overlayed (dots): (a) GP; (b) treed GP.

Examining now the emulators' fits of the leaflet displacement data, the story is similar. Figure 15(a) shows the posterior mean of the GP. Here the fit is quite good but again there are undulations in the emulator fit that are not justified by the data, for the same reasons outlined before. Since there is no real discontinuity here, the GP still provides a reasonable fit however. The treed GP in Figure 15(b) is better still: in particular the abrupt change in gradient is captured much more faithfully. In fact, the data here is almost the ideal set for demonstrating the advantages of the treed GP: the split allows a GP to be assigned to the curved region, then the linear region is dealt with by the LLM, giving significant predictive improvements and computational savings over the standard GP.

5.3 Starving the Emulator

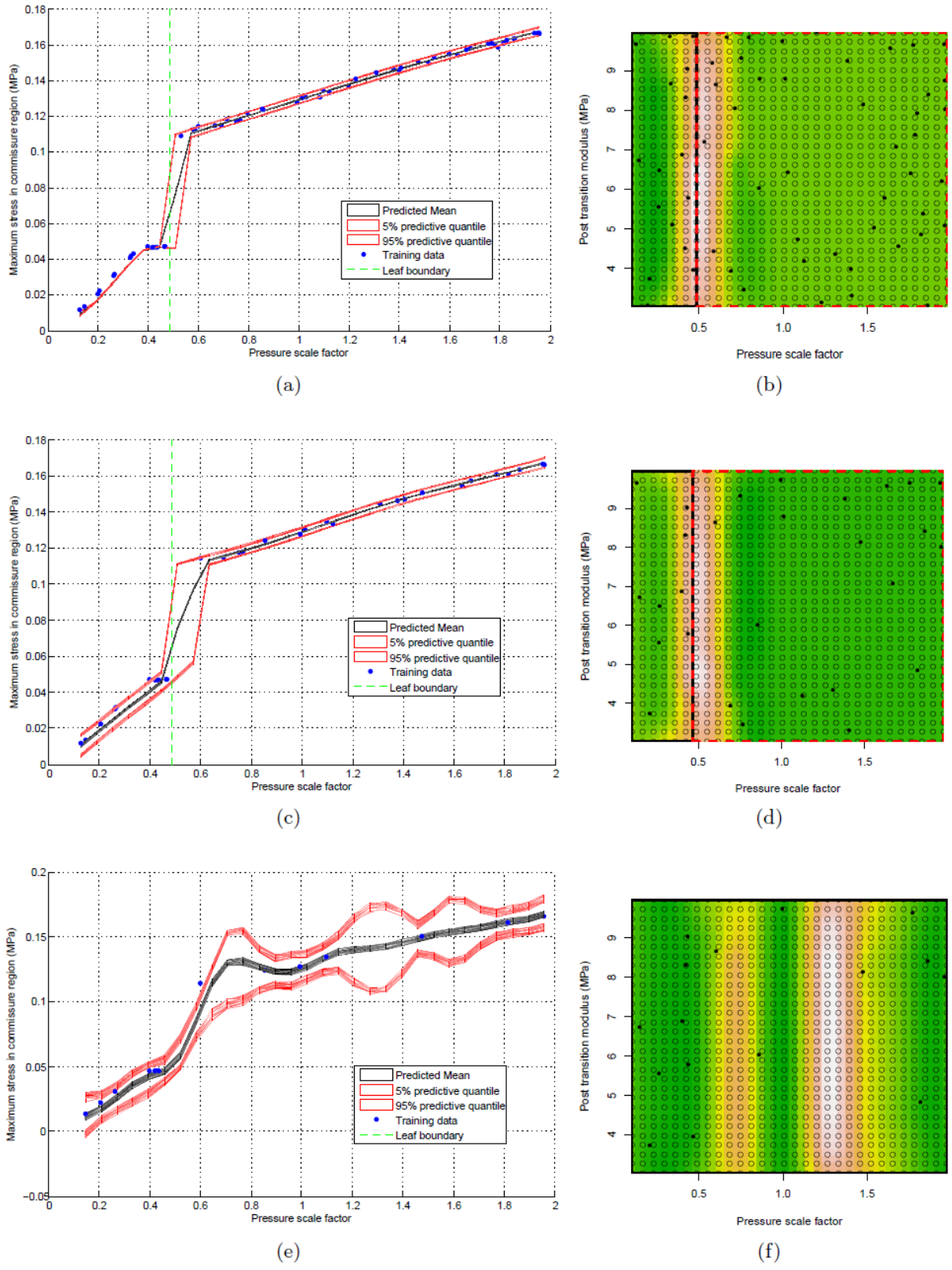

Figure 16: Treed GP posterior mean of commissure stress with reduced training data: (a) 60 points; (b) 60 points error plot; (c) 31 points; (d) 31 points error plot; (e) 17 points; (f) 17 points error plot.

Given that 100 training data points is quite a substantial number for a GP and treed GP, it is interesting to briefly investigate the effect of "starving" the emulators of data. Figure 16 shows the effect of successively reducing the training data set by removing random data points for the treed GP trained on the stress data. At 60 points the fit of the emulator is not significantly changed, however at 31 data points the fit is considerably worsened. The details of the response below the discontinuity are lost and the confidence intervals are increased, reflecting the increased uncertainty about predictive points. Despite this, the location of the discontinuity is still correctly identified.

When the training data is again reduced, to 17 points, the fit becomes poor. The discontinuity is now not identified and the data is modelled by a single GP, since there is not enough data to identify the higher-pressure region as being linear. The error plot in Figure 16(f) shows that the highest magnitude of error is no longer at the discontinuity, but in the linear region. The implication of this is that the treed GP is able to emulate a discontinuous system, but only when there is sufficient training data to invoke a division in input space. This is unsurprising, but it is particularly crucial to have enough data *in the vicinity* of the discontinuity, since a discontinuity may look like a smooth increase when training data is scarce, which can be modelled more simply by a single GP. It is possible that adjusting parameters in the tree prior may have some improvement on the fit if a discontinuity is suspected, but more data is always preferable.

6 Conclusions

This work has shown that the treed GP is capable of modelling bifurcating and discontinuous response surfaces that are beyond the capability of standard GPs and other standard nonlinear regression approaches, and lends itself readily to uncertainty and sensitivity analysis. The two case studies used here have highlighted that there is a need for such methods in engineering, since bifurcating responses can occur through, for example, the cubic stiffness term in the Duffing oscillator, or buckling in the case of the heart valve.

The treed GP does have a shortcoming, in that it is not able to accurately model bifurcations that are not parallel to coordinate axes (Kim *et al*. [46] have indeed addressed this issue to some extent with the use of Voronoi tessellations). However, estimations of posterior predictive variance, combined with diagnostics such as cross-validation, allow bad fits to be easily identified, and further training data to be directed to regions of high uncertainty. It is also worth emphasising that when a bifurcation is not present, the treed GP will revert to a single GP, or indeed a Bayesian linear regression, which avoids over-fitting.

~~The ability of standard and treed GPs to model a discontinuous system has been examined for simple and more complex examples, with a case study of a Duffing oscillator and a discontinuous FE model. It is no surprise that a standard GP has some trouble modelling a bifurcation, because this forces the roughness hyperparameters to be high, which are then used over the full model, and are unsuitable in the perfectly flat high- and low-amplitude regions. If the training data were sufficiently dense, however, the GP would become increasingly accurate, though this removes one of the main attractions of GP-based SA. Nevertheless, the Duffing oscillator was chosen to be a tough test, and in a great number of cases the GP is an excellent way of modelling a smooth response surface. Furthermore, if high values of roughness parameters are noticed when training a GP, this may be used as a indicator that the GP is not correctly emulating the data. Cross-validation may also be used to further investigate the emulator fit.~~

~~The treed GP is rather better, but still has some shortcomings. In the case of a discontinuity that is parallel to a parameter axis, it can very accurately capture the discontinuity in a way that is beyond the ability of the standard GP. Furthermore, the computational cost of doing so is actually reduced compared to modelling the same thing with a single GP. The LLM technique also extends the flexibility of the model by recognising a linear surface, which allows the almost constant-amplitude regions to be modelled very effectively, providing there is sufficient training data to invoke the LLM. However, the treed GP has trouble in modelling discontinuities that are functions of more than one parameter (not parallel to any axis, or curved). This is clearly demonstrated in case 3 of the Duffing oscillator, where the treed GP fails to capture the curved discontinuity, resulting in areas of high error. In order to closely emulate such~~

~~discontinuities, a very complex tree would be required, since partitioning can only be performed parallel to axes.~~

~~Main effects and sensitivity indices can be readily gained from both emulators, which tend to agree to a large extent in the cases investigated. However, as a result of the over fitting of the data by the standard GP from the presence of discontinuities, the sensitivity of the output to some parameters can be over-estimated slightly. In both cases however, there are very significant computational savings over Monte Carlo estimates. Both emulators provide a powerful framework from which to apply SA to complex models.~~

~~On reducing the set of training data, it was found, as expected, that the quality of fit worsened. In particular, for small training data sets the discontinuity may not be identified at all. There is a need not only to have sufficient data overall, but particularly to have sufficient data in the vicinity of a discontinuity, if it exists. This could perhaps be achieved by an adaptive sampling procedure.~~

~~It is also noted that a further advantage of the tree-structured GP is that it may perform better in extrapolation. Since the tree-structured GP will revert to its local prior distribution outside of the range of training data, it is likely to produce better predictive estimates in extrapolation than a single GP, which will revert to the global prior (which may often be markedly different to local trends).~~

7 Acknowledgements

The authors would like to thank Ove Arup for supplying the FE software to make this possible and Dr Alaster Yoxall of Sheffield Hallam University for many useful discussions on the nature of FE analysis. Additional thanks goes to Tony O'Hagan and Marc Kennedy for the use of GEM-SA; and Robert Gramacy and Matthew Taddy for making publicly available the *tgp* package. The authors would also like to thank Dr Jeremy Oakley of the School of Mathematics and Statistics at the University of Sheffield for advice on various matters throughout the whole programme of research.

8 Bibliography

1. Becker, W., J. Rowson, J. Oakley, A. Yoxall, G. Manson, and K. Worden, *Bayesian sensitivity analysis of a model of the aortic valve.* Journal of Biomechanics, 2011. **44**(1): p. 1499–1506.
2. Saltelli, A., K. Chan, and E.M. Scott, *Sensitivity Analysis*. Wiley series in probability and statistics. 2000, New York: Wiley.
3. Klir, G.J., *Uncertainty and Information: Foundations of Generalized Information Theory*. 2006: Wiley-IEEE Press.
4. O'Hagan, A., *Uncertain judgements : eliciting experts' probabilities*. 2006, Chichester: Wiley. xiii, 321 p.
5. Saltelli, A., K. Chan, and E.M. Scott, *Sensitivity Analysis*. 2000, New York: Wiley.
6. Sobol', I.M., *Sensitivity estimates for nonlinear mathematical models.* Mathematical Modeling and Computational Experiment, 1993. **1**(4): p. 407-414.
7. Homma, T. and A. Saltelli, *Importance measures in global sensitivity analysis of nonlinear models.* Reliability Engineering and System Safety, 1996. **52**(1): p. 1-17.
8. Shreider, Y.A., *Method of Statistical Testing : Monte Carlo Method*. 1964, Amsterdam ; London: Elsevier.
9. Jansen, M.J.W., *Analysis of variance designs for model output.* Computer Physics Communication, 1999. **117**(1-2): p. 35-43.
10. Saltelli, A., P. Annoni, I. Azzini, F. Campolongo, M. Ratto, and S. Tarantola, *Variance based sensitivity analysis of model output. Design and estimator for the total sensitivity index* Computer Physics Communications, 2010. **181**(2): p. 259-270.
11. Bishop, C.M., *Pattern Recognition and Machine Learning*. 2006: Springer, New York.

12. Friedman, J.H., *Greedy function approximation: a gradient boosting machine.* The Annals of Statistics, 2001. **29**(5): p. 1189-1232.
13. Storlie, C.B., H.D. Bondell, B.J. Reich, and H.H. Zhang, *Surface estimation, variable selection, and the nonparametric oracle property*. Statistica Sinica, 2011. **21**(2): p. 679-705.
14. Friedman, J.H., *Multivariate adaptive regression splines (with discussion).* Annals of Statistics, 1991. **19**(1): p. 1-141.
15. Oakley, J.E. and A. O'Hagan, *Probabilistic sensitivity analysis of complex models: a Bayesian approach.* Journal Of The Royal Statistical Society Series B, 2004. **66**(3): p. 751-769.
16. Hussain, M.F., R.R. Barton, and S.B. Joshi, *Metamodeling: Radial basis functions, versus polynomials.* European Journal of Operational Research, 2002. **138**(1): p. 142-154.
17. Jin, R., W. Chen, and T.W. Simpson, *Comparative studies of metamodelling techniques under multiple modelling criteria*. Structural and Multidisciplinary Optimization, 2001. **23**(1): p. 1-13.
18. Storlie, C.B. and J.C. Helton, *Multiple predictor smoothing methods for sensitivity analysis: Example results.* Reliability Engineering System Safety, 2008. **93**(1): p. 55-77.
19. Storlie, C.B., L.P. Swiler, J.C. Helton, and C.J. Sallaberry, *Implementation and evaluation of nonparametric regression procedures for sensitivity analysis of computationally demanding models.* Reliability Engineering System Safety, 2009. **94**(11): p. 1735-1763.
20. Gramacy, R.B. and H.K.H. Lee, *Bayesian treed Gaussian process models with an application to computer modeling.* Journal of the American Statistical Association, 2008. **103**(483): p. 1119-1130.
21. Sacks, J., W.J. Welch, T.J. Mitchell, and H.P. Wynn, *Design and analysis of computer experiments.* Statistical Science, 1989. **4**(4): p. 409-435.
22. Kennedy, M.C., C.W. Anderson, S. Conti, and A. O'Hagan, *Case studies in Gaussian process modelling of computer codes.* Reliability Engineering and System Safety, 2006. **91**(10-11): p. 1301-1309.
23. Rasmussen, C.E. and C.K.I. Williams, *Gaussian Processes for Machine Learning*. Adaptive computation and machine learning. 2006, Cambridge, Mass. ; London: MIT. xviii, 248 p.
24. Santner, T.J., B.J. Williams, and W. Notz, *The Design and Analysis of Computer Experiments*. 2003, New York ; London: Springer. xii, 283 p.
25. Gelman, A., *Bayesian Data Analysis*. 2nd ed. ed. 2004, London: Chapman & Hall/CRC. xxv, 668 p.
26. Becker, W., J.E. Oakley, C. Surace, P. Gili, J. Rowson, and K. Worden, *Bayesian sensitivity analysis of a nonlinear finite element model*. Mechanical Systems and Signal Processing, 2012. http://dx.doi.org/10.1016/j.ymssp.2012.03.009
27. Haylock, R., *Bayesian Inference about Outputs of Computationally Expensive Algorithms with Uncertainty on the Inputs*. 1997, University of Nottingham.
28. Becker, W., *Uncertainty Propagation Through Large Nonlinear Models*, PhD Thesis, *Department of Mechanical Engineering*. 2011, The University of Sheffield: Sheffield. p. 209.
29. Breiman, L., *Classification and Regression Trees*. 1984, Belmont, California: Wadsworth.
30. MacKay, D.J.C., *Information Theory, Inference, and Learning Algorithms*. 2003: Cambridge University Press.
31. Cormen, T.H., *Introduction to Algorithms*. 2001: The MIT Press.
32. Chipman, H.A., E.I. George, and R.E. McCulloch, *Bayesian CART model search.* Journal of the American Statistical Association, 1998. **93**(443): p. 935-948.
33. Chipman, H.A., E.I. George, and R.E. McCulloch, *Bayesian treed models.* Machine Learning, 2002. **48**(1): p. 299-320.
34. Gramacy, R.B., *Bayesian Treed Gaussian Process Models*. 2005, University of California.
35. Hastings, W.K., *Monte Carlo sampling methods using Markov chains and their applications.* Biometrika, 1970. **57**(1): p. 97.
36. Metropolis, N., A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, and others, *Equation of state calculations by fast computing machines.* The Journal of Chemical Physics, 1953. **21**(6): p. 1087.

37. Gramacy, R.B., *tgp: An R package for Bayesian nonstationary, semiparametric nonlinear regression and design by treed Gaussian process models.* Journal of Statistical Software, 2007. **19**(9).
38. Green, P.J., *Reversible jump Markov chain Monte Carlo computation and Bayesian model determination*. Biometrika, 1995. **82**(4): p. 711-732.
39. Gelfand, A.E. and A.F.M. Smith, *Sampling-based approaches to calculating marginal densities.* Journal of the American statistical association, 1990. **85**(410): p. 398-409.
40. Andrieu, C., N. De Freitas, A. Doucet, and M.I. Jordan, *An introduction to MCMC for machine learning*. Machine Learning, 2003. **50**(1): p. 5-43.
41. Gramacy, R.B. and H.K.H. Lee, *Gaussian processes and limiting linear models.* Computational Statistics and Data Analysis, 2008. **53**(1): p. 123-136.
42. Gramacy, R.B. and M.A. Taddy, *Categorical inputs, sensitivity analysis, optimization and importance tempering with tgp version 2, an R package for treed Gaussian process models.* Journal of Statistical Software, 2010. **33**(6).
43. Jansen, M.J.W., W.A.H. Rossing, and R.A. Daamen, *Monte Carlo estimation of uncertainty contributions from several independent multivariate sources*, in *Predictability and Nonlinear Modelling in Natural Sciences and Economics*, G.v.S. J. Grasman, Editor. 1994, Kluwer Academic Publishing: Dordrecht. p. 335-343.
44. Worden, K., G. Manson, T.M. Lord, and M.I. Friswell, *Some observations on uncertainty propagation through a simple nonlinear system.* Journal of Sound and Vibration, 2005. **288**(3): p. 601-621.
45. Becker, W., Rowson, J., Oakley, J., Yoxall, A., Manson, G., Worden, K. *Bayesian Sensitivity Analysis of a Large Nonlinear Model*. in *ISMA 2008*. 2008. Leuven.
46. Kim, H.M., B.K. Mallick, and C.C. Holmes, *Analyzing nonstationary spatial data using piecewise Gaussian processes.* Journal of the American Statistical Association, 2005. **100**(470): p. 653-668.