UNIVERSITY OF CALIFORNIA
SANTA CRUZ

BAYESIAN NONPARAMETRIC METHODS FOR EMULATION, SENSITIVITY ANALYSIS, AND CALIBRATION OF COMPUTER SIMULATORS

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

STATISTICS AND STOCHASTIC MODELING

by

Marian Farah

December 2011

For Professor
Marc Mangel

Thank you very much
for your support
and mentorship.

~ Marian Farah
October, 28, 2011

The Dissertation of Marian Farah
is approved:

Professor Athanasios Kottas, Chair

Professor Herbert Lee

Professor Marc Mangel

Professor Robin D. Morris

Dean Tyrus Miller
Vice Provost and Dean of Graduate Studies

UNIVERSITY OF CALIFORNIA
SANTA CRUZ

BAYESIAN NONPARAMETRIC METHODS FOR EMULATION, SENSITIVITY ANALYSIS, AND CALIBRATION OF COMPUTER SIMULATORS

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

STATISTICS AND STOCHASTIC MODELING

by

Marian Farah

December 2011

The Dissertation of Marian Farah
is approved:

Professor Athanasios Kottas, Chair

Professor Herbert Lee

Professor Marc Mangel

Professor Robin D. Morris

Dean Tyrus Miller
Vice Provost and Dean of Graduate Studies

Contents

List of Figures

List of Tables

Abstract

Bayesian Nonparametric Methods for

Emulation, Sensitivity Analysis, and Calibration of Computer Simulators

by

Marian Farah

Computer simulators are commonly used to model physical processes or the behavior of real-world systems in order to make predictions or to obtain estimates for a region of the process/system where observations are difficult to acquire. Understanding and quantifying uncertainty in the simulator's inputs and output is important for assessing its utility. This thesis focuses on uncertainty analysis of computationally expensive computer simulators from a Bayesian perspective. In particular, new Bayesian tools are developed to answer important questions in emulation, sensitivity analysis, and calibration. Specifically, a fully Bayesian approach to sensitivity analysis of deterministic simulators is presented, where the emulator is built through a Gaussian process prior. This approach is used to carry out sensitivity analysis of the Leaf-Canopy Model (LCM), a radiative transfer model that simulates the interaction of sunlight with vegetation. Additionally, a new Bayesian framework for emulation and calibration of stochastic simulators is developed, where the emulator is built from flexible nonparametric Dirichlet process mixtures. This framework is applied to the emulation and calibration of PROPSET, a computer code that simulates the bombardment of spaceborne microelectronics with high-energy protons. Finally, in order to connect calibration results for PROPSET with results from heavy ion experiments, a semiparametric Bayesian isotonic regression approach is developed in order to quantify uncertainty in the cross-section vs linear energy transfer curve, one of PROPSET's key inputs.

For the women of Syria

Acknowledgments

I am grateful to have had the support and guidance of many outstanding people throughout my education. I am incredibly fortunate to have Thanasis Kottas as my PhD advisor. This thesis would not have been possible without his mentorship. Thanasis taught me a great deal about writing, attention to detail, soccer, beer, and of course, Bayesian nonparametrics. Thanasis is a fierce statistician. I admire him greatly, and I look forward to continued collaborative work with him in the future.

Of course, I am grateful to all AMS faculty for providing an excellent learning environment and for being generous with their time and advice whenever I needed them. In particular, I would like to thank Herbie Lee for inspiring me to work harder, for helping open many professional doors for me, and for being a good friend. I would like to also thank Robin D. Morris for taking my phone calls every time I called to discuss the tuning of proposals for Metropolis-Hastings steps. Robin provided me with invaluable advice on MCMC matters as well as personal life, and I am very lucky to have him on my dissertation committee. I am also grateful to Marc Mangel for teaching me that behind every statistical modeling challenge is an exciting scientific problem waiting to be solved. Even though I am not his student, Marc enthusiastically gave me an office in his lab and invited me to his lab retreats and meetings. I am very grateful for his support and candid advice over the last four years.

Of course, this thesis would not have been possible without the interesting scientific problems and applications that I used to illustrate the statistical methodologies developed in this document. I would like to acknowledge all those who contributed data and software. In partic-

ular, I would like to thank Barry D. Ganapol and Roberto Furfaro (University of Arizona) for providing the LCM runs, Gary Swift (Xilinx) and Steve Guertin (JPL) for generously providing the heavy-ion and proton test data, and Pat O'Neil (NASA-JSC) and Chuck Foster for providing PROPSET.

From the moment I arrived in Santa Cruz, AMS faculty and staff have guided and supported me, making sure I was provided every opportunity to succeed. I am grateful to Bruno Sansó for keeping me on track, Raquel Prado for helping me navigate my first year in grad school and a difficult time in my personal life, Pascale Garaud for teaching me how to be a good teaching assistant, Tracie Tucker for making grad school more manageable, Susan Leach for making grad school more fun, Derek Pearson for providing top notch IT support, and my fellow AMS grad students for making grad school awesome!

Last but not least, I would like to thank my family and friends, for everything I have ever accomplished has only been possible through their unconditional love and support. Thank you dad for introducing me to geometry, and for making me believe that a girl can do math. Thank you mom for making me focus on my education, for protecting me from becoming another teenage bride in Syria, and for fighting to bring your six children to the United States. Thank you Farah, Ghosoun, Rajaa, Johnny, and Remy for being the best siblings in the world and for always inspiring me to work harder—I love you all very much. Finally, thank you Mark for giving me a sense of perspective and for teaching me to be patient and brave. I admire you, I love you dearly, and I look forward to a life of adventures together.

Chapter 1

Introduction

A computer simulator, computer model, computer simulation, or a computer experiment is a computer program (or code) that simulates an abstract (mathematical) model describing the behavior of a real-world system. First the mathematical model is developed, then it is coded into a computer simulation, which is run over a set of input configurations in order to obtain outputs that give an insight into the structure of the modeled system. Typically, the amount of actual observed data of the modeled system is very limited, and thus the output of the computer simulator is vital for understanding the behavior of the system. In this thesis, we refer to the mathematical model and its computer code implementation as a computer *simulator*.

There are two types of computer simulators: deterministic and stochastic. With deterministic simulators, the same set of inputs to the simulator always results in the same output. Examples of deterministic simulators include models for radiative transfer through the Earth's atmosphere, climate models, and fluid dynamics simulations modeled as systems of differential equations. With stochastic simulators, the same set of inputs results in different (or a distribu-

tion of) outputs. Stochastic simulators rely on pseudo random number generators, which are initialized with a seed. If the seed is known, then one can predict exactly the output of the simulator, so the output is not truly random but "pseudorandom". However, as long as the simulator uses a "good" pseudorandom number generator, the output is considered random enough for statistical purposes. Stochastic computer simulators are being increasingly used in technology and science to model random systems, e.g., in population dynamics, biological processes, queueing networks in a communication system, and nuclear interactions. Even with the incredible advances in the speed and power of computers over the last couple of decades, there are still simulators that push the most powerful computers to their limits.

There is great interest in quantifying uncertainty and identifying its sources in computer simulators in order to improve the quality of scientific inference (and decisions made) based on their outputs. Bayesian statistics provides a natural choice for quantifying uncertainty at every stage of model development, and over the last 25 years, Bayesian methods have been utilized to develop tools for analysis of computer simulators. In this thesis, we explore solutions, from the Bayesian perspective, for important questions in analysis of computationally expensive computer simulators in the areas of emulation, sensitivity analysis, and calibration.

Emulation involves constructing a computationally inexpensive statistical model that gives an accurate approximation to the simulator output. The surrogate model is called an *emulator*, and it is constructed using a few key runs of the simulator. In the case of deterministic simulators, the simulator may be viewed as a function $f(\cdot)$ that maps inputs x into an output $y = f(x)$. Then for any input configuration x, a Bayesian emulator provides an entire probability distribution for $f(x)$. The mean of that distribution can be viewed as a point estimate of $f(x)$,

and the distribution around that mean describes how close it is likely to be to $f(x)$. When the simulator is stochastic, it maps x to the random variable $y = f(x)$. A major contribution of this thesis revolves around emulation of stochastic simulators, an area which has received little attention in the statistical literature.

Sensitivity analysis (SA) is concerned with investigating how uncertainty in the simulator inputs impacts uncertainty in the simulator output. In particular, SA is used to identify which inputs are most influential in inducing the uncertainty in the output. Such information is important for understanding the behavior of the simulator and determining where better input information is needed and where the model might be improved. The most commonly used sensitivity measures are based on formulating uncertainty in the simulator inputs using a joint probability distribution and then analyzing the induced uncertainty in the output.

For computationally cheap simulators, sensitivity measures are obtained through a "brute force" approach, where the simulator is run over a dense grid of input configurations specified according to their joint probability distribution. Here, millions of runs might be required, which makes such calculations impractical for computationally expensive simulators. To obviate this computational burden, SA computations are carried out using cheap emulator runs (which are produced from a single set of training runs of the simulator). Using the emulator introduces uncertainty into the calculations. However, this uncertainty is quantifiable since the emulator is a fully specified statistical model. That is, since the emulator is a probability distribution for (deterministic or stochastic) f(x), any measures based on f(x) will have an induced probability distribution through the emulator.

Sensitivity analysis is concerned with uncertainty in the simulator output. However,

there can also be uncertainty about the simulator inputs. If this is the case, calibration is performed to approximate their values (or distributions). Here, the simulator output is combined with observational data of the real-world system in order to learn about the uncertain inputs. Additionally, calibration may be used in conjunction with validation to account for the discrepancy between the simulator and the system it is modeling. This is typically implemented by modeling the difference between the simulator and the physical process using a bias function. Again, when the simulator is computationally expensive, emulation plays an important role in the analysis.

1.1 The components of this thesis

This thesis has two major components with the methodological statistical work presented from the Bayesian perspective. The first component includes work on sensitivity analysis of deterministic simulators, with application to radiative transfer models (RTMs). The sensitivity of the simulator output is measured via the calculation of the main effects, which summarize the influence of each input on the output, and by the sensitivity indices, which are variance-based measures that apportion the simulator output variance among the inputs. Most of the existing literature focuses on emulators formulated through Gaussian process (GP) priors, with semi-Bayesian estimates of the main effects and ad-hoc estimates, without uncertainty quantification, of the sensitivity indices.

In Chapter 2, we develop a fully Bayesian GP model for the simulator output and derive expressions that allow for efficient calculation of Bayesian point estimates and standard

errors for the main effects of the simulator inputs. This approach also enables sampling the entire posterior distribution of the sensitivity indices in a coherent way by combining posterior samples from the GP emulator with a Monte Carlo approach that allows for the simultaneous propagation of the input distribution. This work has been applied to the Leaf-Canopy Model (LCM), a radiative transfer model that simulates the light reflected by a vegetated region of the Earth as observed by a satellite sensor.

The second component of this thesis includes methodological work on the emulation and calibration of stochastic computer simulators. The few examples in the literature of analysis for stochastic simulators build upon GP-based emulation models for deterministic simulators and use plug-in estimators for the GP prior. In this thesis component, we explore two approaches. The first is semiparametric and involves approximating the simulator output by a parametric distribution with smoothly varying parameter(s) modeled using GP priors. The second approach involves the development of a new framework that models the joint density of the simulator input(s) and output using a flexible nonparametric mixture model. Although other prior models can also be used, this approach focuses on Dirichlet process (DP) mixtures using a curve fitting method to build the emulator through the implied conditional distribution for the output given the inputs. For calibration, the two modeling approaches are extended to allow borrowing of strength between the two different sources of data (field observations and simulator data). In Chapter 3, we develop the methodological and technical details of both approaches and illustrate them using a couple of different synthetic examples that highlight the strength (and limitation) of both approaches.

In Chapter 4, we present the motivating scientific problem for the proposed methodol-

ogy on analysis of stochastic simulators, and carry out the required modeling and calculations. Here, the motivation is estimating an important parameter (which is a parametric curve) for the prediction of "soft errors" in spaceborne microelectronics due to radiation in Earth's orbit. Uncertainty about this curve is quantified using data from particle accelerator experiments and PROPSET, which is a stochastic simulator that models the bombardment of a microchip with high-energy protons in order to determine the effect of radiation on spaceborne microelectronics. PROPSET is computationally expensive, which motivates the emulator approach in order to carry out the analysis involved in model calibration. Additionally, modeling extensions to experimental data analysis is presented at the end of this chapter using a nonparameteric prior for the curve.

Overall conclusions and future work directions are summarized in Chapter 5.

1.2 What is Bayesian Statistics?

"*It is unanimously agreed that statistics depends somehow on probability. But, as to what probability is and how it is connected with statistics, there has seldom been such complete disagreement and breakdown of communication since the Tower of Babel. Doubtless, much of the disagreement is merely terminological and would disappear under sufficiently sharp analysis.*"

—Leonard Savage (1954)

The most common interpretation of probability is, perhaps, the "frequentist" one, where a probability is interpreted as the rate (or limiting frequency) at which an event occurs

in a large number of trials that are independently repeatable under identical conditions. This interpretation is associated with random physical systems where it is possible to repeat an experiment, like rolling dice. However, not all systems are random and not all experiments are repeatable. A Bayesian can assign a probability to an event even when it is unrepeatable or no random process is involved. From the Bayesian perspective, a probability is interpreted as a degree of belief that an event will occur, and observations have the power to modify (or update) that belief. The remainder of this chapter gives a very gentle and brief introduction to Bayesian analysis.

"Bayesian" refers to the Reverend Thomas Bayes $(1702-1761)$, who is known for formulating the "inverse-probability" theorem named after him. Given events A and B, and assuming their probability distributions are discrete, Bayes' theorem states that $p(B|A) = p(A|B)\,p(B)/p(A)$, with $p(A) = p(A|B)p(B) + p(A|B^c)p(B^c)$. The superscript c refers to the complement event, $p(B|A)$ is called the posterior probability, and $p(A)$ is its normalizing constant; $p(A|B)$ is called the likelihood, and $p(B)$ is called the prior. Bayes's theorem can be easily extended to account for more than two events and to model continuous probability distributions.

In general, Bayesian treatment of a statistical problem proceeds as follows: A hypotheses is expressed through probability distributions for observable data, y. These probability distributions depend on unknown parameter(s), θ, which may be a single value, a function, or a distribution. Current knowledge (or uncertainty) about θ is expressed through the prior distribution, $p(\theta)$. Information contained in new data, y, regarding θ is expressed in the likelihood, $p(y \mid \theta)$. Then, the updated probability distribution (or posterior) for θ, is obtained by combining the likelihood with the prior as $p(\theta \mid y) \propto p(y \mid \theta) \times p(\theta)$ (Cowles et al., 2009), where $\propto$

denotes “proportional to”. Thus, the Bayesian paradigm provides a natural way to structure the data and knowledge (or lack of it), in order to produce updated answers.

If the posterior distribution is in the same family of probability distributions as the prior, then the prior is called “conjugate”. Conjugate priors are convenient because the posteriors are in closed form. One may also work with conditionally conjugate priors, where part of the parameter vector is conditioned on the other part, and obtain conditionally conjugate posteriors. However, in realistically complex models, conjugate priors may not appropriately represent prior uncertainly, so nonconjugate priors are used leading to posteriors that are not available in closed form. If this is the case, posterior inference is carried out by sampling from the posterior distribution using simulation.

Markov chain Monte Carlo (MCMC) algorithms are the standard posterior simulation methods used in the Bayesian world. A Markov chain, is a random process where the next state of the process depends only on the present state, and not on any states in the past. Monte Carlo originally referred to a method of using probability distribution functions to approximate intractable integrals. Putting the two techniques together, MCMC constructs a Markov chain to sample from a probability distribution, whose stationary distribution is the target posterior (or marginal posterior) distributions. Implementations of MCMC include the Metropolis-Hastings (M-H) algorithm (Metropolis et al., 1953; Hastings, 1970), and the Gibbs sampler (Geman and Geman, 1984; Gelfand and Smith, 1990). The M-H algorithm proceeds as follows:

1. Initialize θ at some value.

2. Given the current state is θ^t, generate θ^* from a proposal distribution $q(\theta^* \mid \theta^t)$.

3. Compute the M-H acceptance probability, α as

$$\alpha = \frac{p(y \mid \theta^*)\, p(\theta^*)\, q(\theta^t \mid \theta^*)}{p(y \mid \theta^t)\, p(\theta^t)\, q(\theta^* \mid \theta^t)}$$

4. Set $\theta^{t+1} = \theta^*$ with probability α, or $\theta^{t+1} = \theta^t$ with probability $1 - \alpha$.

5. Repeat steps $2 - 4$ as necessary.

A clear advantage to this algorithm is seen in step 3, where the normalizing constant, $p(y)$, of the posterior is not required for the calculations, and only tractable expressions (densities specified up to a constant of integration) are used. Additionally, this algorithm is applicable when θ is multidimensional (or a vector), where updating may be carried out for each individual component, groups of components, or the entire vector.

Often, a large number of MCMC steps is needed in order to sufficiently sample the posterior distribution, and care must be taken to monitor convergence and verify independent sampling. For example, plotting the time series of the chain provides a way for visually assessing its convergence. Draws from the "burn-in" period, which is the part of the chain before stationarity is reached are discarded, and only post burnin samples are used for posterior inference. If needed, approximately independent samples can be obtained by "thining" the remaining samples by storing only every *n*th sample. More efficient sampling may be achieved by adjusting or "tuning" the proposal distribution aiming for an acceptance rate of about 25%. For convergence diagnostics and independent sampling efficiency measures, see for example Gelman and Rubin (1992), Raftery and Lewis (1992), Geweke (1992), and Gelman et al. (1996).

The Gibbs sampler is a special case of the M-H algorithm, where the proposal dis-

tribution $q(\theta^* \mid \theta^t) = q(\theta^* \mid y)$, for which the proposed value is always accepted (i.e., $\alpha = 1$). Gibbs sampling allows for the conditional posterior distributions of the target distribution to be sampled directly by generating samples for each parameter conditional on the current values of the others. Gibbs sampling is, therefore, attractive because it does not require tuning of a proposal distribution. However, it is only applicable if the conditional posterior distributions are easy to sample from. M-H and Gibbs sampling are heavily utilized to carry out posterior inference under the Bayesian methods developed in the next three chapters.

Chapter 2

Bayesian Inference for Sensitivity Analysis of Deterministic Computer Simulators, with an Application to Radiative Transfer Models

In this chapter we develop Bayesian inferential methods for sensitivity analysis of deterministic computer simulators. Sensitivity analysis is a valuable tool in model development, calibration, and validation, since it investigates how uncertainty in the model inputs impacts uncertainty in the model output. The sensitivity of the deterministic simulator output is typically measured via the calculation of the "main effects," which provide a summary of the influence of each input on the model output, and by the "sensitivity indices," which are variance-based measures that give the expected reduction in output uncertainty when the true value of the input is known (Saltelli et al., 2000; Oakley and O'Hagan, 2004).

Calculating the main effects and sensitivity indices requires the evaluation of multidi-

mensional integrals over the input space of the simulator. Thus, standard numerical integration methods (e.g., Monte Carlo integration or multidimensional quadrature) are infeasible when the simulator is computationally expensive. This problem has been tackled through building a statistical emulator, which is a computationally efficient statistical approximation of the simulator output. Then, fast emulator runs are used in place of slow simulator runs for the analysis.

The most popular statistical emulator is the Gaussian process (GP). It is a distribution over a function, where any finite collection of the function outputs has a multivariate normal distribution. For tutorials on GP models, see Neal (1998), Mackay (1998), and Rasmussen and Williams (2006). The GP emulator is a convenient model because the mathematical properties of the multivariate normal distribution carry over to the GP allowing for efficient analysis. Using GP emulators for deterministic simulators dates back to the work of Sacks et al. (1989); see, e.g., the book by Santner et al. (2003). Moreover, in more recent years there has been an upsurge in research activity on Bayesian methods for analysis of computer simulators; see, e.g., Kennedy and O'Hagan (2001), Craig et al. (2001), Oakley and O'Hagan (2002), Higdon et al. (2004), Goldstein and Rougier (2006), Bayarri et al. (2007b), Gramacy and Lee (2008), and Bayarri et al. (2009).

In Bayesian modeling, the GP is commonly used as a prior model for an unknown function, such as the output of a computer simulator. Full mathematical details of the Bayesian approach to GP emulation can be found in Kennedy and O'Hagan (2001). Once posterior inference is obtained using the GP model, the main effects and sensitivity indices can be determined using runs from the emulator's posterior predictive distribution, which are substantially faster than those obtained using the simulator. Oakley and O'Hagan (2004) develop a semi-Bayesian

estimation approach for the main effects based on fixed "range of dependence" parameters for the GP. They also propose approximate point estimates for the sensitivity indices based on ratios of posterior expectations to estimate the posterior expectation of ratios of variances; a similar approach, albeit under a likelihood estimation setting for the GP parameters, is used by Morris et al. (2008).

In this chapter, we develop an approach that unifies sensitivity analysis tools and extends semi-Bayesian approaches to a fully Bayesian methodology. The starting point of our approach involves approximating a computationally expensive simulator by a fully Bayesian GP model. Based on runs of the GP posterior predictive distribution, we develop an approach to full inference for global sensitivity analysis. First, we calculate Bayesian point estimates of the main effects and their associated uncertainties. This approach is appealing because it utilizes analytic expressions to estimate the main effects based on the GP model, which results in efficient computation. Next, we design a method to obtain full posterior distributions of different types of sensitivity indices over the input space of the model. This latter method expands the inference scope of the earlier work in Oakley and O'Hagan (2004) and Morris et al. (2008).

The motivating application for this work is provided by the Leaf-Canopy Model (LCM) computer simulatior (Ganapol et al., 1999). Developed at the University of Arizona, the LCM is a radiative transfer model that describes the interaction of sunlight with vegetation. The methodology developed here is applied to the LCM to estimate the main effects and sensitivity indices of each of its inputs at 8 different MODIS spectral bands that are sensitive to vegetation; MODIS (Moderate Resolution Imaging Spectroradiometer) is a key instrument

aboard the Terra and Aqua satellites (Justice et al., 1998).

The outline of this chapter is as follows. Section 2.1 develops the methodology for sensitivity analysis, with some of the technical details provided in Appendix A. Section 2.2 includes a description of the LCM model and reports inference results on sensitivity analysis for its inputs. Finally, Section 2.3 concludes with a summary and discussion of future work.

2.1 Methods

To prepare the ground for the proposed methodology, Section 2.1.1 reviews GP-based emulation for computer simulators. The approach to inference for sensitivity analysis is presented in Section 2.1.2. In particular, Section 2.1.2.1 develops a computationally efficient approach to point and interval estimation for the main effects, whereas in Section 2.1.2.2, we describe a method to sample the entire posterior distribution of the sensitivity indices.

2.1.1 Gaussian process emulation

An emulator is a computationally efficient statistical model that is used to approximate a computationally expensive simulator. Denote by $f(\mathbf{v})$ the simulator output as a function of input vector $\mathbf{v} = (v_1, \ldots, v_k)$. Given a set of training model runs $D = \{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$, where $\mathbf{x}_i = (x_{1i}, \ldots, x_{ki})$ is the i–th realized design input vector, and $y_i = f(\mathbf{x}_i)$ is the corresponding output, the emulator treats the computer simulator as a black box and uses D to estimate $f(\cdot)$. Following the work of Sacks et al. (1989) and Kennedy and O'Hagan (2001), GPs are widely used to model the computer code output function $f(\cdot)$. The advantage of a GP emulator

is that it is a fully specified statistical model that requires one carefully chosen set of model runs, and therefore, it is both tractable and efficient. While the GP approximation introduces uncertainty into the computation of the main effects and sensitivity indices, this uncertainty is quantifiable.

Under a GP prior for function $f(\cdot)$, for any finite set of input points $(\mathbf{v}_1, \ldots, \mathbf{v}_N)$, where $\mathbf{v}_i = (v_{1i}, \ldots, v_{ki})$ for $i = 1, \ldots, N$, the joint distribution of the outputs $(f(\mathbf{v}_1), \ldots, f(\mathbf{v}_N))$ is multivariate normal. Furthermore, GP models typically assume that the output is a smooth function of its inputs, that is, nearby locations in the input space produce outputs that are stochastically close in value. A GP is fully specified by its mean function, $\mathrm{E}(f(\mathbf{v}))$, and positive definite covariance function, $\mathrm{Cov}(f(\mathbf{v}_i), f(\mathbf{v}_j))$. We assume constant mean function, $\mathrm{E}(f(\mathbf{v})) = \mu$, and a stationary covariance function, $\mathrm{Cov}(f(\mathbf{v}_i), f(\mathbf{v}_j)) = \tau^2 \mathrm{Corr}(f(\mathbf{v}_i), f(\mathbf{v}_j))$, which is taken to be isotropic (depends only on $|\mathbf{v}_i - \mathbf{v}_j|$) with constant variance, τ^2, and a product power exponential correlation of the form,

$$R_{\boldsymbol{\phi}} = \mathrm{Corr}(f(\mathbf{v}_i), f(\mathbf{v}_j)) = \exp\left\{ -\sum_{\ell=1}^{k} \phi_\ell |v_{\ell i} - v_{\ell j}|^{a_\ell} \right\}, \tag{2.1}$$

where $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_k)$, with $\phi_\ell > 0$, is the vector of "range of dependence" parameters, which control the dependence strength in each of the component directions of $\mathbf{v}$. Here, $a_\ell \in [1, 2]$ are the "smoothness" parameters, which are typically fixed (or a highly informative prior is used for them) based on a combination of prior knowledge about $f(\cdot)$ and computational considerations (e.g., Higdon et al., 2004). For example, a value of $a_\ell = 2$ implies that $f(\cdot)$ is a smooth infinitely differentiable function, whereas smaller values of a_ℓ result in rougher (continuous) realizations.

The form of the GP correlation function in (2.1) corresponds to a choice commonly used in the statistical literature on GP emulation for computer simulators. In our context, it facilitates computing of some of the integrals needed for estimation of the main effects (see Section 2.1.2.1). However, the approach to sensitivity analysis proposed in Section 2.1.2 is sufficiently generic to allow application with more general choices for the GP covariance function; see Rasmussen and Williams (2006) for a review of other classes of covariance functions.

To obtain the set of training data, D, for the GP emulator, we use a Latin Hypercube design (McKay et al., 1979) to generate the design matrix of the model inputs and calculate the corresponding outputs using the simulator. We treat the functional form of the simulator output, $f(\cdot)$, as unknown and specify a prior for it in the form of the isotropic GP discussed above. We note that it is often useful (or necessary) to add a small (fixed) *jitter* term to the covariance function for numerical stability. This is a standard computational strategy in Bayesian nonparametric regression as well as analysis of simulators with GP priors (e.g., Neal, 1998; Higdon et al., 2004; Gramacy and Lee, 2011).

Given D, there are n induced variables from the GP representation for $f(\cdot)$, $y_i = f(\mathbf{x}_i)$, for $i = 1,\ldots,n$, with induced prior $(f(x_1),\ldots,f(x_n)) \sim N_n\left(\mu\mathbf{1}_n, \tau^2 R_{\boldsymbol{\phi}}\right)$. Here, $\mathbf{1}_n$ is the n–dimensional vector with all elements equal to 1, and $R_{\boldsymbol{\phi}}$ is the $n \times n$ observed correlation matrix with (i,j)–th element given by $\exp\left\{-\sum_{\ell=1}^{k} \phi_\ell |x_{\ell i} - x_{\ell j}|^{a_\ell}\right\}$. To complete the Bayesian model for the GP emulator, we fix a_ℓ, $\ell = 1,\ldots,k$, and place (independent) priors on the hyperparameters of the GP, μ, τ^2, and ϕ_ℓ, $\ell = 1,\ldots,k$. Thus, the joint posterior distribution of all parameters, $\boldsymbol{\psi} = (\mu, \tau^2, \boldsymbol{\phi})$, is given by $p(\boldsymbol{\psi} \mid D) \propto N_n\left(\mathbf{y}|\mu\mathbf{1}_n, \tau^2 R_{\boldsymbol{\phi}}\right) p(\mu)p(\tau^2)p(\phi_1)\ldots p(\phi_k)$, where $\mathbf{y} = (y_1,...,y_n)$. Samples from $p(\boldsymbol{\psi} \mid D)$ are obtained using Markov chain Monte Carlo

(MCMC) posterior simulation as discussed in Appendix A.

Analysis of simulator output performed using runs of the emulator have an additional level of uncertainty, since those runs are an approximation of the computer code output. We account for this uncertainty by performing any further analysis over the posterior predictive distribution of the GP. For any generic input, $\mathbf{v} = (v_1, \ldots, v_k)$, which is not part of the design, we can obtain the posterior predictive distribution for $\tilde{y} = f(\mathbf{v})$. Specifically,

$$p(\tilde{y} \mid D) = \int N\left(\tilde{y} \mid m(\mathbf{v}), s^2(\mathbf{v})\right) p(\boldsymbol{\psi}|D)\, d\boldsymbol{\psi}, \tag{2.2}$$

$$m(\mathbf{v}) \equiv \mathrm{E}\left(\tilde{Y}|\boldsymbol{\psi}, D\right) = \mu + \boldsymbol{r}^T(\mathbf{v}) R_{\boldsymbol{\phi}}^{-1}(\mathbf{y} - \mu \mathbf{1}_n), \tag{2.3}$$

$$s^2(\mathbf{v}) \equiv \mathrm{Var}\left(\tilde{Y}|\boldsymbol{\psi}, D\right) = \tau^2\left(1 - \boldsymbol{r}^T(\mathbf{v}) R_{\boldsymbol{\phi}}^{-1} \boldsymbol{r}(\mathbf{v})\right), \tag{2.4}$$

where $\boldsymbol{r}(\mathbf{v})$ is the $n \times 1$ vector with i–th element given by $\mathrm{Corr}\left(f(\mathbf{v}), f(\mathbf{x}_i)\right)$.

The joint predictive distribution for $(\tilde{y}, \tilde{y}') = (f(\mathbf{v}), f(\mathbf{v}'))$ corresponding to generic inputs $\mathbf{v} = (v_1, \ldots, v_k)$ and $\mathbf{v}' = \left(v_1', \ldots, v_k'\right)$ is given by $p(\tilde{y}, \tilde{y}' \mid D) =$
$\int p(\tilde{y}, \tilde{y}' \mid \boldsymbol{\psi}) p(\boldsymbol{\psi} \mid D)\, d\boldsymbol{\psi}$, where $p(\tilde{y}, \tilde{y}' \mid \boldsymbol{\psi})$ is bivariate normal with (2×1) mean vector $\boldsymbol{\omega}(\mathbf{v}, \mathbf{v}')$, and (2×2) covariance matrix $C(\mathbf{v}, \mathbf{v}')$, such that

$$\boldsymbol{\omega}(\mathbf{v}, \mathbf{v}') = \mu \mathbf{1}_2 + R^T\left(\mathbf{v}, \mathbf{v}'\right) R_{\boldsymbol{\phi}}^{-1}(\mathbf{y} - \mu \mathbf{1}_n), \tag{2.5}$$

$$C(\mathbf{v}, \mathbf{v}') = \tau^2\left(B\left(\mathbf{v}, \mathbf{v}'\right) - R^T\left(\mathbf{v}, \mathbf{v}'\right) R_{\boldsymbol{\phi}}^{-1} R\left(\mathbf{v}, \mathbf{v}'\right)\right), \tag{2.6}$$

where $B(\mathbf{v}, \mathbf{v}')$ is the (2×2) correlation matrix for $(f(\mathbf{v}), f(\mathbf{v}'))$, and $R(\mathbf{v}, \mathbf{v}')$ is the $(n \times 2)$ matrix, where the elements of the first column are given by $\mathrm{Corr}\left(f(\mathbf{v}), f(\mathbf{x}_i)\right)$, $i = 1, \ldots, n$, and

the elements of the second column by $\text{Corr}\left(f(\mathbf{v}'), f(\mathbf{x}_i)\right)$, $i = 1, \ldots, n$.

2.1.2 Fully Bayesian inference for global sensitivity analysis

The key idea behind variance-based approaches to sensitivity analysis is to decompose the output function $y = f(\mathbf{v})$ into summands of increasing dimensionality (Sobol, 1993). Specifically, given a k–dimensional input space,

$$y = f(\mathbf{v}) = f_0 + \sum_{\ell=1}^{k} f_\ell(v_\ell) + \sum_{1 \le \ell < m \le k} f_{\ell,m}(v_\ell, v_m) + \cdots + f_{1,2,\ldots,k}(v_1, \ldots, v_k).$$

Here, f_0 is the global mean given by $f_0 = \text{E}(Y) = \int_{\mathbf{v}} f(\mathbf{v})\, dH(\mathbf{v})$, where $H(\mathbf{v}) = \prod_{\ell=1}^{k} H_\ell(v_\ell)$ is the uncertainty distribution of the inputs comprising independent components $H_\ell(v_\ell)$. The next k terms are the main effects, where $f_\ell(v_\ell)$ is the main effect of input v_ℓ, providing a measure of the influence of input v_ℓ on the computed output. For $\ell = 1, \ldots, k$, $f_\ell(v_\ell) = \text{E}(Y|v_\ell) - \text{E}(Y) = \int_{\mathbf{v}_{-\ell}} f(\mathbf{v})\, dH(\mathbf{v}_{-\ell}|v_\ell) - \text{E}(Y)$, where $\mathbf{v}_{-\ell}$ denotes input vector $\mathbf{v}$ excluding element v_ℓ. Because of the independent components of the uncertainty distribution, the conditional distribution $H(\mathbf{v}_{-\ell}|v_\ell)$ simplifies to $H(\mathbf{v}_{-\ell})$. The remaining terms of the decomposition are the interactions, which quantify the combined influence on the output of two or more inputs taken together. For instance, the first-order interactions, $f_{\ell,m}(v_\ell, v_m) = \text{E}(Y|v_\ell, v_m) - f_\ell(v_\ell) - f_m(v_m) - \text{E}(Y)$.

Sobol (1993) shows that based on this output decomposition, and assuming independence between the input variables in the uncertainty distribution, the total variance, $\text{Var}(Y) = W$,

can also be decomposed as the sum of partial variances,

$$W = \sum_{\ell=1}^{k} W_\ell + \sum_{1\le\ell<m\le k} W_{\ell,m} + \cdots + W_{1,2,\ldots,k}, \tag{2.7}$$

where $W_\ell = \mathrm{Var}\left(f_\ell(v_\ell)\right) = \mathrm{Var}(\mathrm{E}(Y|v_\ell))$, $W_{\ell,m} = \mathrm{Var}\left(f_{\ell,m}(v_\ell, v_m)\right)$, and analogously for the higher order terms. Hence, the sensitivity indices are given by

$$S_\ell = \frac{W_\ell}{W}, \;\; S_{\ell,m} = \frac{W_{\ell,m}}{W}, \;\; \ldots \;\; , \;\; S_{1,2,\ldots,k} = \frac{W_{1,2,\ldots,k}}{W},$$

where S_ℓ is the *first-order sensitivity index* for input v_ℓ, which measures the fractional contribution of that input to the variance of $f(\mathbf{v})$; $S_{\ell,m}$, for $\ell \neq m$, is the *second-order sensitivity index*, which measures the contribution of interaction due to inputs v_ℓ and v_m on the variance of $f(\mathbf{v})$, and analogously for the higher order terms. The decomposition in (2.7) standardizes the sensitivity indices, that is,

$$\sum_{\ell=1}^{k} S_\ell + \sum_{1\le\ell<m\le k} S_{\ell,m} + \cdots + S_{1,2,\ldots,k} = 1.$$

Introduced by Homma and Saltelli (1996), the *total sensitivity index*, S_ℓ^T, is a further related measure, defined by the sum of all the sensitivity indices involving input v_ℓ. Specifically, $S_\ell^T = 1 - (W_{-\ell}/W)$, $\ell = 1, \ldots, k$, where $W_{-\ell} = \mathrm{Var}(\mathrm{E}(Y|\mathbf{v}_{-\ell}))$ is the total contribution to $\mathrm{Var}\left(f(\mathbf{v})\right)$ due to all inputs except v_ℓ. A large difference between S_ℓ and S_ℓ^T for the ℓ–th input indicates an important role of interaction terms involving that input on the variation in the output.

The definition of the main effects and sensitivity indices involves expectations with respect to the simulator output function $y = f(\mathbf{v})$. Therefore, if we approximate the output function by a GP model, we must account for this approximation by computing $\mathrm{E}^*\{\mathrm{E}(Y) \mid D\}$, $\mathrm{E}^*\{\mathrm{E}(Y|v_\ell) \mid D\}$, $\mathrm{E}^*\{S_\ell \mid D\}$, and $\mathrm{E}^*\{S_\ell^T \mid D\}$, where $\mathrm{E}^*\{\cdot \mid D\}$ indicates expectations with respect to the GP posterior predictive distribution, $p(\tilde{y} \mid D)$, developed in Section 2.1.1. As pointed out by Oakley and O'Hagan (2004), $\mathrm{E}^*\{S_\ell \mid D\}$, and $\mathrm{E}^*\{S_\ell^T \mid D\}$, which are posterior expectations of ratios of random variables, cannot be derived analytically. Instead, Oakley and O'Hagan (2004) obtain approximate point estimates for S_ℓ and S_ℓ^T by computing the ratio of expectations over $p(\tilde{y} \mid D)$, where $\mathrm{E}^*\{S_\ell \mid D\}$ is approximated by the ratio of $\mathrm{E}^*\{\mathrm{Var}(\mathrm{E}(Y|v_\ell)) \mid D\}$ and $\mathrm{E}^*\{\mathrm{Var}(Y) \mid D\}$, and analogously for $\mathrm{E}^*\{S_\ell^T \mid D\}$. The approximation of the sensitivity indices through ratios of expectations is also used in the likelihood approach of Morris et al. (2008).

2.1.2.1 Point estimates and uncertainty bands for the main effects

Here, we develop fully Bayesian point estimates for the main effects accompanied by a measure of posterior predictive uncertainty. The corresponding expressions result in relatively straightforward computing owing to the conditional normality structure of the GP emulator. The approach is similar to the one in Oakley and O'Hagan (2004), but extends their empirical Bayesian method based on fixed range parameters for the GP correlation function.

Given the generic input $\mathbf{v} = (v_1, \ldots, v_k)$, the distribution of the predicted emulator output, $\tilde{y} = f(\mathbf{v})$, is given by (2.2). In order to determine the main effect of input v_ℓ, we need to calculate $\mathrm{E}^*\{\mathrm{E}(Y) \mid D\}$ and $\mathrm{E}^*\{\mathrm{E}(Y|v_\ell) \mid D\}$. Regarding the posterior point estimate for the

global mean, we obtain

$$\begin{aligned}
\mathrm{E}^*\{\mathrm{E}(Y)\mid D\} &= \int_{f(\mathbf{v})} \mathrm{E}(Y)\, p(f(\mathbf{v})\mid D)\, df(\mathbf{v}) \\
&= \int_{f(\mathbf{v})} \mathrm{E}(Y) \left\{ \int_{\boldsymbol{\psi}} p(f(\mathbf{v})|\boldsymbol{\psi})\, p(\boldsymbol{\psi}\mid D)\, d\boldsymbol{\psi} \right\} df(\mathbf{v}) \\
&= \int_{\boldsymbol{\psi}} \left\{ \int_{f(\mathbf{v})} \left\{ \int_{\mathbf{v}} f(\mathbf{v})\, dH(\mathbf{v}) \right\} p(f(\mathbf{v})|\boldsymbol{\psi})\, df(\mathbf{v}) \right\} p(\boldsymbol{\psi}\mid D)\, d\boldsymbol{\psi} \\
&= \int_{\boldsymbol{\psi}} \left\{ \int_{\mathbf{v}} \left\{ \int_{f(\mathbf{v})} f(\mathbf{v})\, p(f(\mathbf{v})|\boldsymbol{\psi})\, df(\mathbf{v}) \right\} dH(\mathbf{v}) \right\} p(\boldsymbol{\psi}\mid D)\, d\boldsymbol{\psi} \\
&= \int_{\boldsymbol{\psi}} \left\{ \int_{\mathbf{v}} m(\mathbf{v}) \prod_{\ell=1}^{k} dH_\ell(v_\ell) \right\} p(\boldsymbol{\psi}\mid D)\, d\boldsymbol{\psi}.
\end{aligned}$$

Here, we assume independent components in the uncertainty distribution for the inputs, which, for simpler notation, are taken to be uniform over a normalized range of values in $(0,1)$ for each input. Then, using (2.2)–(2.4), we obtain

$$\begin{aligned}
\mathrm{E}^*\{\mathrm{E}(Y)\mid D\} &= \int_{\boldsymbol{\psi}} \left\{ \int_{\mathbf{v}} \left\{ \mu + \boldsymbol{r}^T(\mathbf{v}) R_{\boldsymbol{\phi}}^{-1} (\mathbf{y} - \mu \mathbf{1}_n) \right\} \prod_{\ell=1}^{k} dH_\ell(v_\ell) \right\} p(\boldsymbol{\psi}\mid D)\, d\boldsymbol{\psi} \\
&= \int_{\boldsymbol{\psi}} \left\{ \mu + \mathbf{T}^T R_{\boldsymbol{\phi}}^{-1} (\mathbf{y} - \mu \mathbf{1}_n) \right\} p(\boldsymbol{\psi}\mid D)\, d\boldsymbol{\psi},
\end{aligned}$$

where $\mathbf{T}$ is the $(n\times 1)$ vector with i–th element $\prod_{\ell=1}^{k} \left\{ \int_0^1 \exp\left(-\phi_\ell |v_\ell - x_{\ell i}|^{a_\ell}\right) dv_\ell \right\}$. Note that the elements of $\mathbf{T}$ can be computed analytically if $a_\ell = 1$, for $\ell = 1,\ldots,k$, which is the specification for the exponential correlation function. Under this specification, the i–th element of $\mathbf{T}$ is written as $\prod_{\ell=1}^{k} \left\{ {\phi_\ell}^{-1} \left(2 - e^{-\phi_\ell x_{\ell i}} - e^{-\phi_\ell (1-x_{\ell i})}\right) \right\}$. Under other specifications for a_ℓ, $\mathbf{T}$ must be approximated using numerical integration methods (e.g., Simpson's rule).

Turning to the posterior point estimate for $\mathrm{E}(Y|u_j)$, for any specified value u_j of the

j–th input, let $\mathbf{v}_j = (v_1, \ldots, u_j, \ldots, v_k)$ and $f(\mathbf{v}_j) = f(v_1, \ldots, u_j, \ldots, v_k)$.

$$\mathrm{E}(Y|u_j) = \int_{\{v_\ell : \ell \neq j\}} f(v_1, \ldots, u_j, \ldots, v_k) \prod_{\{\ell : \ell \neq j\}} dH_\ell(v_\ell). \tag{2.8}$$

Then, we can derive

$$\begin{aligned}
\mathrm{E}^*\left\{\mathrm{E}(Y|u_j) \mid D\right\} &= \int_{f(\mathbf{v}_j)} \mathrm{E}(Y|u_j)\, p(f(\mathbf{v}_j) \mid D)\, df(\mathbf{v}_j) \\
&= \int_{f(\mathbf{v}_j)} \mathrm{E}(Y|u_j) \left\{ \int_{\boldsymbol{\psi}} p(f(\mathbf{v}_j)|\boldsymbol{\psi})\, p(\boldsymbol{\psi} \mid D)\, d\boldsymbol{\psi} \right\} df(\mathbf{v}_j)
\end{aligned}$$

Assuming independent components in the uncertainty distribution for the inputs, and later applying (2.2)–(2.4), we obtain,

$$\begin{aligned}
\mathrm{E}^*\left\{\mathrm{E}(Y|u_j) \mid D\right\} &= \int_{\boldsymbol{\psi}} \left\{ \int_{f(\mathbf{v}_j)} \left\{ \int_{\{v_\ell : \ell \neq j\}} f(\mathbf{v}_j) \prod_{\{\ell : \ell \neq j\}} dH_\ell(v_\ell) \right\} p(f(\mathbf{v}_j)|\boldsymbol{\psi})\, df(\mathbf{v}_j) \right\} \\
&\qquad \times p(\boldsymbol{\psi} \mid D)\, d\boldsymbol{\psi} \\
&= \int_{\boldsymbol{\psi}} \left\{ \int_{\{v_\ell : \ell \neq j\}} \left\{ \int_{f(\mathbf{v}_j)} f(\mathbf{v}_j)\, p(f(\mathbf{v}_j)|\boldsymbol{\psi})\, df(\mathbf{v}_j) \right\} \prod_{\{\ell : \ell \neq j\}} dH_\ell(v_\ell) \right\} \\
&\qquad \times p(\boldsymbol{\psi} \mid D)\, d\boldsymbol{\psi} \\
&= \int_{\boldsymbol{\psi}} \left\{ \int_{\{v_\ell : \ell \neq j\}} m(\mathbf{v}_j) \prod_{\{\ell : \ell \neq j\}} dH_\ell(v_\ell) \right\} p(\boldsymbol{\psi} \mid D)\, d\boldsymbol{\psi} \\
&= \int_{\boldsymbol{\psi}} \left\{ \mu + \mathbf{T}_j^T(u_j) R_{\boldsymbol{\phi}}^{-1} (\mathbf{y} - \mu \mathbf{1}_n) \right\} p(\boldsymbol{\psi} \mid D)\, d\boldsymbol{\psi},
\end{aligned}$$

where $\mathbf{T}_j(u_j)$ is the $(n \times 1)$ vector with i–th element given by

$$\exp\left(-\phi_j|u_j - x_{ji}|^{a_j}\right) \times \prod_{\{\ell:\ell\neq j\}} \left\{ \int_0^1 \exp\left(-\phi_\ell|v_\ell - x_{\ell i}|^{a_\ell}\right) dv_\ell \right\}.$$

For a measure of (posterior predictive) uncertainty associated with the estimate of the main effects, we use $\text{Var}^*\left\{\text{E}\left(Y|u_j\right) - \text{E}(Y) \mid D\right\}$, which is given by

$$\begin{aligned}
&\text{Var*}\left\{\text{E}(Y \mid u_j) \mid D\right\} + \text{Var*}\left\{\text{E}(Y) \mid D\right\} - 2\text{Cov*}\left\{\text{E}(Y \mid u_j), \text{E}(Y) \mid D\right\} = \\
&\text{E*}\left\{(\text{E}(Y \mid u_j))^2 \mid D\right\} - \left(\text{E*}\left\{\text{E}(Y \mid u_j) \mid D\right\}\right)^2 + \text{E*}\left\{(\text{E}(Y))^2 \mid D\right\} - \\
&\left(\text{E*}\left\{\text{E}(Y) \mid D\right\}\right)^2 - 2\left(\text{E*}\left\{\text{E}(Y \mid u_j)\text{E}(Y) \mid D\right\} - \text{E*}\left\{\text{E}(Y \mid u_j) \mid D\right\}\text{E*}\left\{\text{E}(Y) \mid D\right\}\right)
\end{aligned} \tag{2.9}$$

Because we already have the expressions for $\text{E}^*\left\{\text{E}\left(Y|u_j\right) \mid D\right\}$ and $\text{E}^*\left\{\text{E}\left(Y\right) \mid D\right\}$, what is needed is expressions for $\text{E}^*\left\{\left(\text{E}\left(Y|u_j\right)\right)^2 \mid D\right\}$, $\text{E}^*\left\{\left(\text{E}\left(Y\right)\right)^2 \mid D\right\}$, and $\text{E*}\left\{\text{E}(Y \mid u_j)\text{E}(Y) \mid D\right\}$.

First, we derive the expression for $\text{E}^*\left\{(\text{E}(Y|u_j))^2 \mid D\right\}$. Denote $\mathbf{v}_j$ as before and $\mathbf{v'}_j = \left(v'_1, \ldots, u_j, \ldots, v'_k\right)$. Then, $\left(\text{E}\left(Y|u_j\right)\right)^2$ is given by

$$\left(\int_{\{v_\ell:\ell\neq j\}} f\left(\mathbf{v}_j\right) \prod_{\{\ell:\ell\neq j\}} dH_\ell(v_\ell)\right)^2 = \iint_{\substack{\{v_\ell:\ell\neq j\} \\ \{v'_\ell:\ell\neq j\}}} f(\mathbf{v}_j) f(\mathbf{v'}_j) \prod_{\{\ell:\ell\neq j\}} dH_\ell(v_\ell) dH_\ell(v'_\ell).$$

Therefore, taking the expectation with respect to the bivariate posterior predictive distribution for $(f(\mathbf{v}_j), f(\mathbf{v'}_j))$, developed in Section 2.1.1, we obtain

$$
\begin{aligned}
\mathrm{E}^*\left\{\left(\mathrm{E}\left(Y|u_j\right)\right)^2 \mid D\right\} &= \int \left(\mathrm{E}\left(Y|u_j\right)\right)^2 p(f(\mathbf{v}_j), f(\mathbf{v}'_j) \mid D) df(\mathbf{v}_j) df(\mathbf{v}'_j) \\
&= \int \left(\mathrm{E}\left(Y|u_j\right)\right)^2 \left\{\int_{\boldsymbol{\psi}} p\left(f(\mathbf{v}_j), f(\mathbf{v}'_j) \mid \boldsymbol{\psi}\right) p(\boldsymbol{\psi} \mid D)\, d\boldsymbol{\psi}\right\} df(\mathbf{v}_j) df(\mathbf{v}'_j) \\
&= \int_{\boldsymbol{\psi}} \left\{ \iint_{\substack{\{v_\ell:\ell\neq j\} \\ \{v'_\ell:\ell\neq j\}}} \left\{ \int f\left(\mathbf{v}_j\right) f\left(\mathbf{v}'_j\right) p\left(f(\mathbf{v}_j), f(\mathbf{v}'_j) \mid \boldsymbol{\psi}\right) df(\mathbf{v}_j) df(\mathbf{v}'_j) \right\} \right. \\
&\quad \left. \times \prod_{\{\ell:\ell\neq j\}} dH_\ell(v_\ell) dH_\ell(v'_\ell) \right\} p\left(\boldsymbol{\psi} \mid D\right) d\boldsymbol{\psi} \\
&= \int_{\boldsymbol{\psi}} \left\{ \iint_{\substack{\{v_\ell:\ell\neq j\} \\ \{v'_\ell:\ell\neq j\}}} \mathrm{E}\left(f(\mathbf{v}_j) f(\mathbf{v}'_j) \mid \boldsymbol{\psi}\right) \prod_{\{\ell:\ell\neq j\}} dH_\ell(v_\ell) dH_\ell(v'_\ell) \right\} \\
&\quad \times p\left(\boldsymbol{\psi} \mid D\right) d\boldsymbol{\psi}.
\end{aligned}
$$

Next, using the standard covariance identity, we obtain

$$
\mathrm{E}\left(f(\mathbf{v}_j) f(\mathbf{v}'_j) \mid \boldsymbol{\psi}\right) = \mathrm{Cov}\left(f(\mathbf{v}_j), f(\mathbf{v}'_j) \mid \boldsymbol{\psi}\right) + \mathrm{E}\left(f(\mathbf{v}_j) \mid \boldsymbol{\psi}\right) \mathrm{E}\left(f(\mathbf{v}'_j) \mid \boldsymbol{\psi}\right), \tag{2.10}
$$

where the expectation and covariance terms are taken over the conditional bivariate normal distribution for $(f(\mathbf{v}_j), f(\mathbf{v}'_j)) \mid \boldsymbol{\psi}, D$ with mean vector and covariance matrix given by (2.5) and (2.6), respectively. Denote by $R_1 \equiv R_1\left(v_1, \ldots, u_j, \ldots, v_k\right)$ and $R_2 \equiv R_2\left(v'_1, \ldots, u_j, \ldots, v'_k\right)$ the first and second columns, respectively, of the $(n \times 2)$ matrix $R(\mathbf{v}, \mathbf{v}')$ in (2.6). Note that the input vectors, $(v_1, \ldots, u_j, \ldots, v_k)$ and $\left(v'_1, \ldots, u_j, \ldots, v'_k\right)$, have common element u_j. Therefore, R_1 is the $(n \times 1)$ vector with elements $\exp\left(-\phi_j |u_j - x_{ji}|^{a_j} - \sum_{\{\ell:\ell\neq j\}} \phi_\ell |v_\ell - x_{\ell i}|^{a_\ell}\right)$, for $i = 1, ..., n$, and analogously for R_2, replacing v_ℓ with v'_ℓ. Then using (2.3) and (2.4), we obtain

$$
\mathrm{E}(\tilde{Y} \mid \boldsymbol{\psi}) = \mu + R_1^T R_{\boldsymbol{\phi}}^{-1} (y - \mu 1_n) \quad \text{and} \quad \mathrm{E}(\tilde{Y}' \mid \boldsymbol{\psi}) = \mu + R_2^T R_{\boldsymbol{\phi}}^{-1} (y - \mu 1_n) \tag{2.11}
$$

$$\mathrm{Cov}(\tilde{Y},\tilde{Y}' \mid \boldsymbol{\psi}) = \tau^2 \left\{ \exp\left(-\sum\nolimits_{\{\ell:\ell\neq j\}} \phi_\ell |v_\ell - v'_\ell|^{a_\ell} \right) - R_1^T R_{\boldsymbol{\phi}}^{-1} R_2 \right\} \tag{2.12}$$

Substituting (2.11) and (2.12) in (2.10), we obtain for each $j = 1, ..., k$,

$$\begin{aligned} \mathrm{E}^* \left\{ (\mathrm{E}(Y|u_j))^2 \mid D \right\} = \int_{\boldsymbol{\psi}} \left\{ \tau^2 \left(b - T_j^T(u_j) R_{\boldsymbol{\phi}}^{-1} T_j(u_j) \right) + \left(\mu + T_j^T(u_j) R_{\boldsymbol{\phi}}^{-1} (y - \mu 1_n) \right)^2 \right\} \\ \times p(\boldsymbol{\psi} \mid D) d\boldsymbol{\psi}, \end{aligned} \tag{2.13}$$

where $b = \prod_{\{\ell:\ell\neq j\}} \left\{ \int_0^1 \int_0^1 \exp\left(-\phi_\ell |v_\ell - v'_\ell|^{a_\ell}\right) dv_\ell dv'_\ell \right\}$.

Next, we derive the expression for $\mathrm{E}^* \left\{ (\mathrm{E}(Y))^2 \mid D \right\}$ as follows.

$$\begin{aligned} \mathrm{E}^* \left\{ (\mathrm{E}(Y))^2 \mid D \right\} &= \int \int \int_{\boldsymbol{\psi}} (\mathrm{E}(Y))^2 p(f(\mathbf{v}), f(\mathbf{v}') \mid \boldsymbol{\psi}) p(\boldsymbol{\psi} \mid D) d\boldsymbol{\psi} df(\mathbf{v}) df(\mathbf{v}') \\ &= \int_{\boldsymbol{\psi}} \int \int \int_{\mathbf{v}'} \int_{\mathbf{v}} f(\mathbf{v}) f(\mathbf{v}') p(f(\mathbf{v}), f(\mathbf{v}') \mid \boldsymbol{\psi}) p(\boldsymbol{\psi} \mid D) \\ &\quad \times \prod_{\ell=1}^{k} dH_\ell(v_\ell) \prod_{\ell=1}^{k} dH_\ell(v'_\ell) df(\mathbf{v}) df(\mathbf{v}') d\boldsymbol{\psi} \\ &= \int_{\boldsymbol{\psi}} \int_{\mathbf{v}'} \int_{\mathbf{v}} \mathrm{E}\left(f(\mathbf{v}) f(\mathbf{v}') \mid \boldsymbol{\psi} \right) \prod_{\ell=1}^{k} dH_\ell(v_\ell) \prod_{\ell=1}^{k} dH_\ell(v'_\ell) p(\boldsymbol{\psi} \mid D) d\boldsymbol{\psi} \end{aligned}$$

As before, we obtain $\mathrm{E}(f(\mathbf{v}) f(\mathbf{v}') \mid \boldsymbol{\psi})$ using the covariance formula $\mathrm{E}(f(\mathbf{v}) f(\mathbf{v}') \mid \boldsymbol{\psi}) = \mathrm{Cov}(f(\mathbf{v}), f(\mathbf{v}') \mid \boldsymbol{\psi}) + \mathrm{E}(f(\mathbf{v}) \mid \boldsymbol{\psi}) \mathrm{E}(f(\mathbf{v}') \mid \boldsymbol{\psi})$, where:

- $\mathrm{E}(f(\mathbf{v}) \mid \boldsymbol{\psi}) = \mu + \boldsymbol{r}^T R_{\boldsymbol{\phi}}^{-1} (y - \mu 1_n)$, with $\boldsymbol{r}$ the $(n \times 1)$ vector with i–th element given by $\mathrm{Corr}(f(\mathbf{v}), f(\mathbf{x}_i) \mid \boldsymbol{\psi}) = \exp\left[-\sum_{\ell=1}^{k} \phi_\ell \mid v_\ell - x_{\ell i} \mid^{a_\ell} \right]$

- $\mathrm{E}(f(\mathbf{v}') \mid \boldsymbol{\psi}) = \mu + \boldsymbol{r'}^T R_{\boldsymbol{\phi}}^{-1} (y - \mu 1_n)$, with $\boldsymbol{r'}$ the $(n \times 1)$ vector with i–th element given by $\mathrm{Corr}(f(\mathbf{v}'), f(\mathbf{x}_i) \mid \boldsymbol{\psi}) = \exp\left[-\sum_{\ell=1}^{k} \phi_\ell \mid v'_\ell - x_{\ell i} \mid^{a_\ell} \right]$

- $\mathrm{Cov}\left(f(\mathbf{v}), f(\mathbf{v}') \mid \boldsymbol{\psi}\right) = \tau^2 \left\{ \exp\left[-\sum_{\ell=1}^{k} \phi_\ell \mid v'_\ell - v_\ell \mid^{a_\ell} \right] - \boldsymbol{r}^T R_{\boldsymbol{\phi}}^{-1} \boldsymbol{r}' \right\}$.

Therefore,

$$\mathrm{E}^*\left\{ (\mathrm{E}(Y))^2 \mid D \right\} = \int_{\boldsymbol{\psi}} \left\{ \tau^2 \left(g - \mathbf{T}^T R_{\boldsymbol{\phi}}^{-1} \mathbf{T} \right) + \left(\mu + \mathbf{T}^T R_{\boldsymbol{\phi}}^{-1} (\mathbf{y} - \mu \mathbf{1}_n) \right)^2 \right\} \times p(\boldsymbol{\psi} \mid D) d\boldsymbol{\psi}' \tag{2.14}$$

where $g = \prod_{\ell=1}^{k} \left\{ \int_0^1 \int_0^1 \exp\left(-\phi_\ell \mid v_\ell - v'_\ell \mid^{a_\ell}\right) dv_\ell dv'_\ell \right\}$.

Finally, we derive an expression for $\mathrm{E}^*\left\{ \mathrm{E}(Y \mid u_j)\mathrm{E}(Y) \mid D \right\}$, which we can write as

$$\begin{aligned}
\mathrm{E}^*\left\{ \mathrm{E}(Y \mid u_j)\mathrm{E}(Y) \mid D \right\} &= \int\int\int_{\boldsymbol{\psi}} \mathrm{E}(Y \mid u_j)\mathrm{E}(Y) p(f(\mathbf{v}_j), f\left(\mathbf{v}'\right) \mid \boldsymbol{\psi}) \\
&\quad \times p(\boldsymbol{\psi} \mid D) d\boldsymbol{\psi} d f(\mathbf{v}_j) d f\left(\mathbf{v}'\right) \\
&= \int_{\boldsymbol{\psi}} \int\int\int_{\mathbf{v}'} \int_{\{v_\ell : \ell \neq j\}} f(\mathbf{v}_j) f\left(\mathbf{v}'\right) p(f(\mathbf{v}_j), f\left(\mathbf{v}'\right) \mid \boldsymbol{\psi}) p(\boldsymbol{\psi} \mid D) \\
&\quad \times \prod_{\{\ell : \ell \neq j\}} dH_\ell(v_\ell) \prod_{\ell=1}^{k} dH_\ell\left(v'_\ell\right) d f(\mathbf{v}_j) d f\left(\mathbf{v}'\right) d\boldsymbol{\psi} \\
&= \int_{\boldsymbol{\psi}} \int_{\mathbf{v}'} \int_{\{v_\ell : \ell \neq j\}} \mathrm{E}\left(f(\mathbf{v}_j) f\left(\mathbf{v}'\right) \mid \boldsymbol{\psi} \right) \prod_{\{\ell : \ell \neq j\}} dH_\ell(v_\ell) \prod_{\ell=1}^{k} dH_\ell\left(v'_\ell\right) \\
&\quad \times p(\boldsymbol{\psi} \mid D) d\boldsymbol{\psi}.
\end{aligned}$$

We obtain $\mathrm{E}\left(f(\mathbf{v}_j) f(\mathbf{v}') \mid \boldsymbol{\psi}\right)$ using the covariance formula, which requires:

- $\mathrm{E}\left(f(\mathbf{v}_j) \mid \boldsymbol{\psi}\right) = \mu + \boldsymbol{r}_j^T R_{\boldsymbol{\phi}}^{-1}(y - \mu 1_n)$, with $\boldsymbol{r}_j$ the $(n \times 1)$ vector with i–th element given by $\mathrm{Corr}\left(f(\mathbf{v}_j), f(\mathbf{x}_i) \mid \boldsymbol{\psi}\right) = \exp\left[-\sum_{\{\ell : \ell \neq j\}} \phi_\ell \mid v_\ell - x_{\ell i} \mid^{a_\ell} \right] \times \exp\left[-\phi_j \mid u_j - x_{ji} \mid^{a_j} \right]$

- $\mathrm{E}\left(f(\mathbf{v}') \mid \boldsymbol{\psi}\right) = \mu + \boldsymbol{r}'^T R_{\boldsymbol{\phi}}^{-1}(y - \mu 1_n)$, with $\boldsymbol{r}'$ the $(n \times 1)$ vector with i–th element given

by $\mathrm{Corr}\left(f(\mathbf{v}'), f(\mathbf{x}_i) \mid \boldsymbol{\psi}\right) = \exp\left[-\sum_{\ell=1}^{k} \phi_\ell \mid v'_\ell - x_{\ell i} \mid^{a_\ell}\right]$

- $\mathrm{Cov}\left(f(\mathbf{v}_j), f(\mathbf{v}') \mid \boldsymbol{\psi}\right) = \tau^2\Big\{\exp\left[-\sum_{\{\ell:\ell\neq j\}} \phi_\ell \mid v'_\ell - v_\ell \mid^{a_\ell}\right] \exp\left[-\phi_j \mid v'_j - u_j \mid^{a_j}\right]$
  $-\boldsymbol{r}_j^T R_{\boldsymbol{\phi}}^{-1} \boldsymbol{r}'\Big\}$.

Therefore,

$$\mathrm{E}^*\left\{\mathrm{E}(Y \mid u_j)\mathrm{E}(Y) \mid D\right\} = \int_{\boldsymbol{\psi}} \Bigg\{ \tau^2 \left[b\,\phi_j^{-1}\left(2 - e^{-\phi_j u_j} - e^{-\phi_j(1-u_j)}\right) - \mathbf{T}_j^T(u_j) R_{\boldsymbol{\phi}}^{-1} \mathbf{T} \right] + \times \left(\mu + \mathbf{T}_j^T(u_j) R_{\boldsymbol{\phi}}^{-1}(\mathbf{y} - \mu \mathbf{1}_n)\right)\left(\mu + \mathbf{T}^T R_{\boldsymbol{\phi}}^{-1}(\mathbf{y} - \mu \mathbf{1}_n)\right) \Bigg\} \times p(\boldsymbol{\psi} \mid D) d\boldsymbol{\psi}, \tag{2.15}$$

Note that the scalars b and g, which appear in (2.13), (2.14), and (2.15), are available analytically under the exponential correlation function. In particular, letting $a_\ell = 1$, for $\ell = 1, \ldots, k$, we obtain $b = \prod_{\{\ell:\ell\neq j\}} \left\{2\phi_\ell^{-2}\left(e^{-\phi_\ell} + \phi_\ell - 1\right)\right\}$, and $g = \prod_{\ell=1}^{k} \left\{2\phi_\ell^{-2}\left(e^{-\phi_\ell} + \phi_\ell - 1\right)\right\}$.

2.1.2.2 Full inference for the sensitivity indices

The approach of Section 2.1.2.1 cannot be extended to estimate the sensitivity indices. Instead of relying on approximate point estimates for S_ℓ and S_ℓ^T, $\ell = 1, ..., k$, we propose to sample from the posterior distributions for the sensitivity indices by computing at every MCMC sample of the GP emulator all the expectations needed for the definition of the S_ℓ and S_ℓ^T, that is, $\mathrm{Var}(Y) = \mathrm{E}(Y^2) - (\mathrm{E}(Y))^2$, $\mathrm{E}\left((\mathrm{E}(Y|u_j))^2\right)$, and $\mathrm{E}\left((\mathrm{E}(Y|\boldsymbol{u}_{-j}))^2\right)$. Letting $\{\mathbf{v} = (v_1, \ldots, v_k), y = f(\mathbf{v})\}$ be a generic run of the simulator, the expectation and variance of y

are given by

$$\mathrm{E}(Y) = \int_{\mathbf{v}} f(\mathbf{v}) \prod_{\ell=1}^{k} dH_\ell(v_\ell) \quad \text{and} \quad \mathrm{Var}(Y) = \int_{\mathbf{v}} f^2(\mathbf{v}) \prod_{\ell=1}^{k} dH_\ell(v_\ell) \; - \; (\mathrm{E}(Y))^2 .$$

For a generic value u_j of the j–th input, squaring the expression for $\mathrm{E}(Y|u_j)$ in (2.8) and then taking its expectation, we obtain

$$\begin{aligned}
\mathrm{E}\left((\mathrm{E}(Y|u_j))^2\right) &= \int \left\{ \int_{\{v_\ell : \ell \neq j\}} f(v_1, \ldots, u_j, \ldots, v_k) \prod_{\{\ell : \ell \neq j\}} dH_\ell(v_\ell) \right\}^2 dH_j(u_j) \\
&= \int \left\{ \int_{\{v_\ell : \ell \neq j\}} \int_{\{v'_\ell : \ell \neq j\}} f(v_1, \ldots, u_j, \ldots, v_k) \, f\left(v'_1, \ldots, u_j, \ldots, v'_k\right) \right. \\
&\qquad \left. \prod_{\{\ell : \ell \neq j\}} dH_\ell(v_\ell) \prod_{\{\ell : \ell \neq j\}} dH_\ell(v'_\ell) \right\} dH_j(u_j) \\
&= \int_{\mathbf{v}} \int_{\{v'_\ell : \ell \neq j\}} f(\mathbf{v}) \, f\left(v'_1, \ldots, v_j, \ldots, v'_k\right) \prod_{\ell=1}^{k} dH_\ell(v_\ell) \prod_{\{\ell : \ell \neq j\}} dH_\ell(v'_\ell).
\end{aligned}$$

Regarding the expectations needed for the total sensitivity indices, let $\boldsymbol{u}_{-j} = (u_1, \ldots, u_{j-1}, u_{j+1}, \ldots, u_k)$. Then, $\mathrm{E}(Y|\boldsymbol{u}_{-j}) = \int f(v_j, \boldsymbol{u}_{-j}) \, dH_j(v_j)$, and, analogously to the derivation above,

$$\mathrm{E}\left((\mathrm{E}(Y|\boldsymbol{u}_{-j}))^2\right) = \int \int f(u_j, \boldsymbol{u}_{-j}) \, f\left(v'_j, \boldsymbol{u}_{-j}\right) dH_j(v'_j) \prod_{\ell=1}^{k} dH_\ell(u_\ell).$$

At each MCMC posterior sample of the GP emulator, the posterior distributions for the first-order sensitivity index, S_j, and the total sensitivity index, S_j^T, are sampled by evaluating

all the expectations that enter their definition,

$$S_j = \frac{\mathrm{Var}\left(\mathrm{E}(Y|u_j)\right)}{\mathrm{Var}(Y)} = \frac{\mathrm{E}\left((\mathrm{E}(Y|u_j))^2\right) - (\mathrm{E}(Y))^2}{\mathrm{Var}(Y)} \tag{2.16}$$

$$S_j^T = \frac{\mathrm{Var}(Y) - \mathrm{Var}\left(\mathrm{E}(Y|\boldsymbol{u}_{-j})\right)}{\mathrm{Var}(Y)} = 1 - \frac{\mathrm{E}\left((\mathrm{E}(Y|\boldsymbol{u}_{-j}))^2\right) - (\mathrm{E}(Y))^2}{\mathrm{Var}(Y)}. \tag{2.17}$$

The computation involves Monte Carlo integration based on samples from the uncertainty distribution, in the spirit of techniques from Saltelli (2002), but extending the approach to account for the GP approximation to the computer simulator output.

Specifically, we begin by generating input sample matrix $\mathbf{M}$ of size $B \times k$,

$$\mathbf{M} = \begin{pmatrix} v_{1,1} & v_{1,2} & \dots & v_{1,k} \\ \vdots & \vdots & \dots & \vdots \\ v_{B,1} & v_{B,2} & \dots & v_{B,k} \end{pmatrix}$$

where each row of $\mathbf{M}$ is drawn independently from the uncertainty distribution over the simulator inputs, $H(\mathbf{v}) = \prod_{\ell=1}^{k} H_\ell(v_\ell)$.

Next, we generate k input sample matrices, $\mathbf{N}_j$, for $j = 1, \dots, k$, of size $B \times k$ each,

$$\mathbf{N}_j = \begin{pmatrix} v'_{1,1} & v'_{1,2} & \dots & v_{1,j} & \dots & v'_{1,k} \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ v'_{B,1} & v'_{B,2} & \dots & v_{B,j} & \dots & v'_{B,k} \end{pmatrix}$$

where the j–th column of matrix $\mathbf{N}_j$ equals the j–th column of matrix $\mathbf{M}$, but the remaining elements of each row of $\mathbf{N}_j$ form independent random samples from the corresponding marginal

of the uncertainty distribution, $\prod_{\{\ell:\ell\neq j\}} H_\ell(v_\ell)$.

Finally, we generate k input sample matrices, $\mathbf{N}_{-j}$, for $j = 1,\ldots,k$, of size $B \times k$ each,

$$\mathbf{N}_{-j} = \begin{pmatrix} v_{1,1} & v_{1,2} & \ldots & v'_{1,j} & \ldots & v_{1,k} \\ \vdots & \vdots & \ldots & \vdots & \ldots & \vdots \\ v_{B,1} & v_{B,2} & \ldots & v'_{B,j} & \ldots & v_{B,k} \end{pmatrix}$$

where matrices $\mathbf{N}_{-j}$ and $\mathbf{M}$ have all columns in common except the j–th one; the $v'_{b,j}$, $b = 1,...,B$, are randomly sampled from $H_j(v_j)$.

Now, the Monte Carlo simulation, based on the posterior samples from the GP emulator, proceeds as follows:

- For each MCMC posterior sample for $\boldsymbol{\psi} = (\mu, \tau^2, \boldsymbol{\phi})$, obtain the following posterior predictive samples according to (2.2): for each row b of M, sample $\tilde{y}_b$, then compute $\tilde{y}_b^2$; for each row b of N_j, sample $\tilde{y}'_{b,j}$; and for each row b of N_{-j}, sample $\tilde{y}'_{b,-j}$.

- Obtain the posterior sample for $\mathrm{E}(Y)$ and $\mathrm{E}(Y^2)$ by computing $B^{-1}\sum_{b=1}^{B}\tilde{y}_b$ and $B^{-1}\sum_{b=1}^{B}\tilde{y}_b^2$, respectively.

- For $j = 1,\ldots,k$, obtain the posterior sample for $\mathrm{E}\left((\mathrm{E}(Y|u_j))^2\right)$ and $\mathrm{E}\left((\mathrm{E}(Y|u_{-j}))^2\right)$ through $B^{-1}\sum_{b=1}^{B}\tilde{y}_b\tilde{y}'_{b,j}$ and $B^{-1}\sum_{b=1}^{B}\tilde{y}_b\tilde{y}'_{b,-j}$, respectively.

- Compute the posterior realizations for the first-order sensitivity indices, S_j, and the total sensitivity indices, S_j^T, by evaluating expressions (2.16) and (2.17), respectively, using the posterior samples above for the required expectations.

Because the posterior samples for the variances in expressions (2.16) and (2.17) are evaluated through differences of expectations, negative values for the S_j and/or S_j^T posterior realizations can arise in the implementation of the method. This issue can be overcome by appropriate choice of the Monte Carlo sample size B. For instance, for the results reported in Section 2.2.2, we used $B = 250,000$; with the exception of the occurrence of negative values for sensitivity indices supported by values close to 0, the estimated posterior distributions in Figure 2.3 were similar under $B = 25,000$.

In previous work (Oakley and O'Hagan, 2004; Morris et al., 2008), ad-hoc estimates for the sensitivity indices were obtained by computing the ratio of point estimates for the variances in the definition of S_j. Here, we are able to estimate the entire distribution of each sensitivity index allowing for the uncertainty of the sensitivity indices to be determined. While repeated samples from the GP predictive distribution are required for computing (2.16) and (2.17), these are computationally inexpensive emulator runs, which are substantially faster than those obtained using the computer simulator. We note that a similar approach was briefly discussed in Taddy et al. (2009) based on a treed GP prior (Gramacy and Lee, 2008) for the simulator output.

Finally, note that the method discussed in this section can, in principle, be also applied to obtain the posterior distribution for the main effects. However, the approach requires for each input a (relatively large) number of sample matrices (of size $B \times k$ each) over a (relatively fine) grid in the input space; each of these matrices can be used to obtain the posterior sample for $\mathrm{E}(Y|v_j = w_q)$, where w_q is the q–th grid point for the j–th input. Hence, in practice, the approach becomes prohibitively computationally expensive even for moderate dimensions for the input space. The approach of Section 2.1.2.1 offers a practically feasible alternative to point

and interval estimation for the main effects. At the same time, we note that the method of this section is entirely generic with regard to the statistical model emulator utilized for the computer simulator, whereas the approach of Section 2.1.2.1 is specific to GP emulators.

2.2 Application

Radiative Transfer Models (RTMs) are widely used in geoscience and remote sensing for the estimation and prediction of the properties of Earth's coupled dynamical system. Typically implemented in complex computer programs, RTMs simulate the reflection of electromagnetic radiation off the surface of the Earth and its propagation through the atmosphere. While RTMs are deterministic computer simulators, there is uncertainty about the values of their inputs. To illustrate the methods developed in this chapter, we use the Leaf-Canopy Model (LCM) (Ganapol et al., 1999), a RTM that simulates light reflected by vegetation as measured by sensors mounted on orbiting satellites. In Section 2.2.1, we provide a brief description of the LCM simulator, and in Section 2.2.2, we study the sensitivity of the LCM output to uncertainty in its inputs.

2.2.1 Leaf-Canopy Model

Radiative transfer models (RTMs) are deterministic simulators that model the interaction of light with a medium. They provide a tool to aid in remote sensing as applied to biosphere and ecosystem dynamics, where Earth observing satellites provide observations that are used to produce wide array of data products, e.g. measurements of sea surface temperature, polar

ice coverage, and plant type. Such data products are used in a wide variety of further scientific studies, and also as inputs to important policy decisions, especially those concerning the impact of human activity on the biosphere. Many of the data products are produced by inverting an RTM, which is implemented as a complex computer code to simulate the upwelling radiation at the top of the atmosphere (and so observed by the satellite) as a function of the biosphere parameters (e.g., land cover type, available water, leaf chemistry).

The Leaf-Canopy Model (LCM) is an RTM, which models light reflected by a vegetated region of the Earth, as observed by a satellite sensor. The LCM was developed by the Vegetation Modeling Transport Group (University of Arizona), in collaboration with the Ecosystem Science and Technology Branch at NASA Ames in support of MODIS, a key instrument aboard the Terra and Aqua satellites. The goal of the LCM is to capture the essential biophysical processes associated with the interaction between light and vegetation. This goal is accomplished through combining two different radiative transfer algorithms: one at the leaf level (LEAFMOD), and the other at the canopy level (CANMOD) (Ganapol et al., 1999).

In the forward mode, LEAFMOD uses the leaf's thickness, scattering profile, and absorption profile to calculate hemispherical reflectances and transmittances and the directional distribution of the radiance exiting the leaf surfaces. The leaf absorption profile is constructed from biochemical concentrations, and specific absorptivities of chlorophyll and carotenoids, protein, lignin and cellulose, and water (Ganapol et al., 1998). In the inverse mode LEAFMOD uses leaf thickness and spectral measurements from the LOPEX leaf database to determine the scattering profile of the leaf. The LOPEX leaf species archive stores experimentally obtained spectral properties for many common species (Hosgood et al., 1995).

The CANMOD algorithm combines the leaf spectral information coming from LEAFMOD with canopy structural parameters (Leaf Area Index (LAI) and leaf angle distribution (LAD)), soil reflectance, and sun angle, and computes, at any given wavelength, the radiative regime within and at the top of the canopy by solving a radiative transfer equation. LAI is the area of the leaves on a canopy divided by the area of the ground covered by the canopy, and is thus a dimensionless quantity. LAD describes the orientation of the leaves and it takes four discrete values: planophile (horizontal), erectophile (vertical), plagiophile (at 45 degrees), and extremophile (both horizontal and vertical).

Figure 2.1 shows a flowchart for the operation of the LCM simulation, which can be explained as follows. First, a leaf type is specified, and its absorption profile is constructed based on its biochemical components. Then, the leaf scattering profile is determined by executing LEAFMOD in the inverse mode. Given the constructed scattering and absorption profiles of the leaf and its thickness, LEAFMOD is run in the forward mode to compute the leaf optical properties (i.e., leaf reflectance and transmittance). Then, the output of LEAFMOD is fed to CANMOD together with LAI, LAD, soil reflectance, the sun angle, and the wavelength (between 400 and 2100 nm), to compute the canopy hemispherical reflectance.

Thus, from the two coupled algorithms, the LCM inputs include leaf chemistry variables (chlorophyll, water fraction, lignin/cellulose, and protein), leaf thickness, soil reflectance, canopy architecture (LAI and LAD), wavelength, and sun angle (Ganapol et al., 1999). Table 2.1 lists the LCM inputs and their ranges, and Table 2.2 includes the 8 bands (or groups of wavelengths) used by the LCM along with their corresponding MODIS band numbers. In our analysis, the LAD variable is set to planophile, and the sun angle is set to zenith.

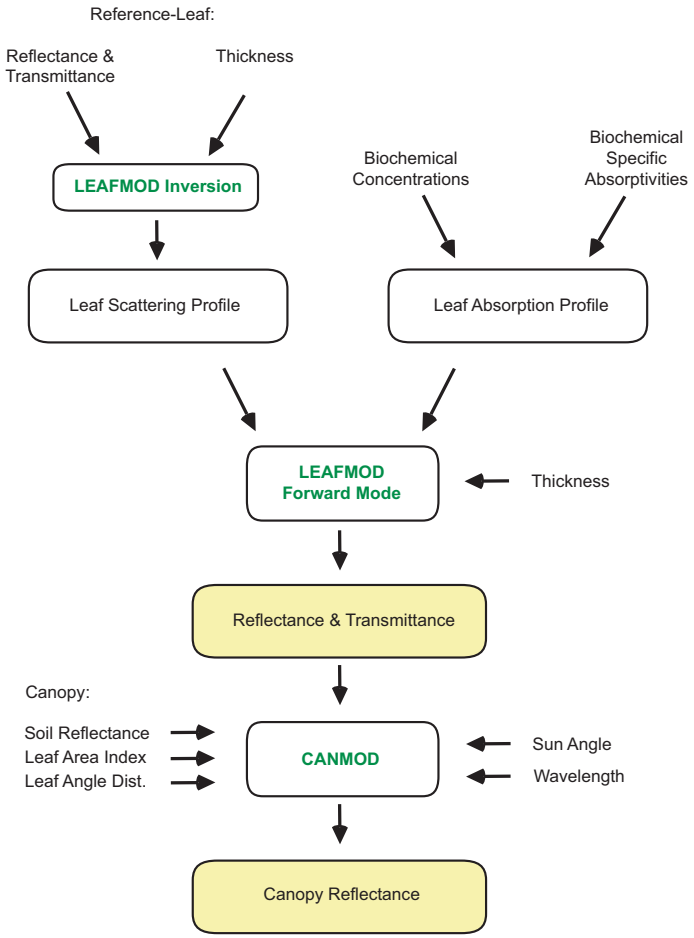


Figure 2.1: Flowchart for the LCM simulator.

2.2.2 Sensitivity Analysis Results for the LCM Simulator

We apply the Bayesian approach to the GP emulator using a training set of 250 LCM runs based on a Latin Hypercube design at each of the 8 MODIS bands (see Table 2.2). We use the exponential correlation function, setting in (2.1) $a_\ell = 1$, for $\ell = 1, \ldots, k = 7$ (a jitter term was not needed). We place a normal prior on μ, an inverse-gamma prior on τ^2, and a Unif$(0, b_{\phi_\ell})$ prior on each ϕ_ℓ, $\ell = 1, \ldots, 7$, assuming prior independence for all hyperparameters. Details on prior specification as well as MCMC posterior simulation for the GP model parameters are

Input	Min	Max
LAI	0	8
Chlorophyll ($\mu g/cm^2$)	0	100
Water fraction	0.1	0.8
Protein (mg/cm^2)	0.1	1
Lignin/Cellulose (mg/cm^2)	0.1	6
Thickness (cm)	0.01	0.1
Soil reflectance	0.3	1.3

Table 2.1: Ranges of inputs to the LCM. LAI, water fraction, and soil reflectance parameters are dimensionless.

band #	wavelength (nm)	MODIS band
1	469	ref3
2	555	ref4
3	1240	ref5
4	1640	ref6
5	2130	ref7
6	667	ref13
7	748	ref15
8	870	ref16

Table 2.2: Wavelength for each band used and the corresponding MODIS band number. Bands are in the MODIS band order, not in the wavelength order.

provided in Appendix A. We have also experimented with gamma priors of varying dispersion for each ϕ_ℓ, which resulted in nearly identical posteriors. For the uncertainty distribution, we assume independent uniform components over the ranges given in Table 2.1 for each input variable.

Figure 2.2 shows plots of the main effects for the 7 normalized input variables and their uncertainty intervals for each of the 8 MODIS bands, based on the approach of Section 2.1.2.1. The interval estimates are computed by adding and subtracting two standard deviations, calculated from the variance point estimate given in (2.9). Normalizing the range of values of the inputs to the unit interval allows all the main effects to be plotted together. The roughness seen in the plotted lines of means and uncertainty bands is due to the use of the exponential correlation function in our analysis, but this should not affect the overall trends of the main effects. In general, the larger the variation of the main effect plot, the greater the influence of that input on the LCM output. For the visible spectrum (bands 1, 2 and 6), the LCM is most sensitive to chlorophyll, where an increase in chlorophyll produces a decrease in the LCM

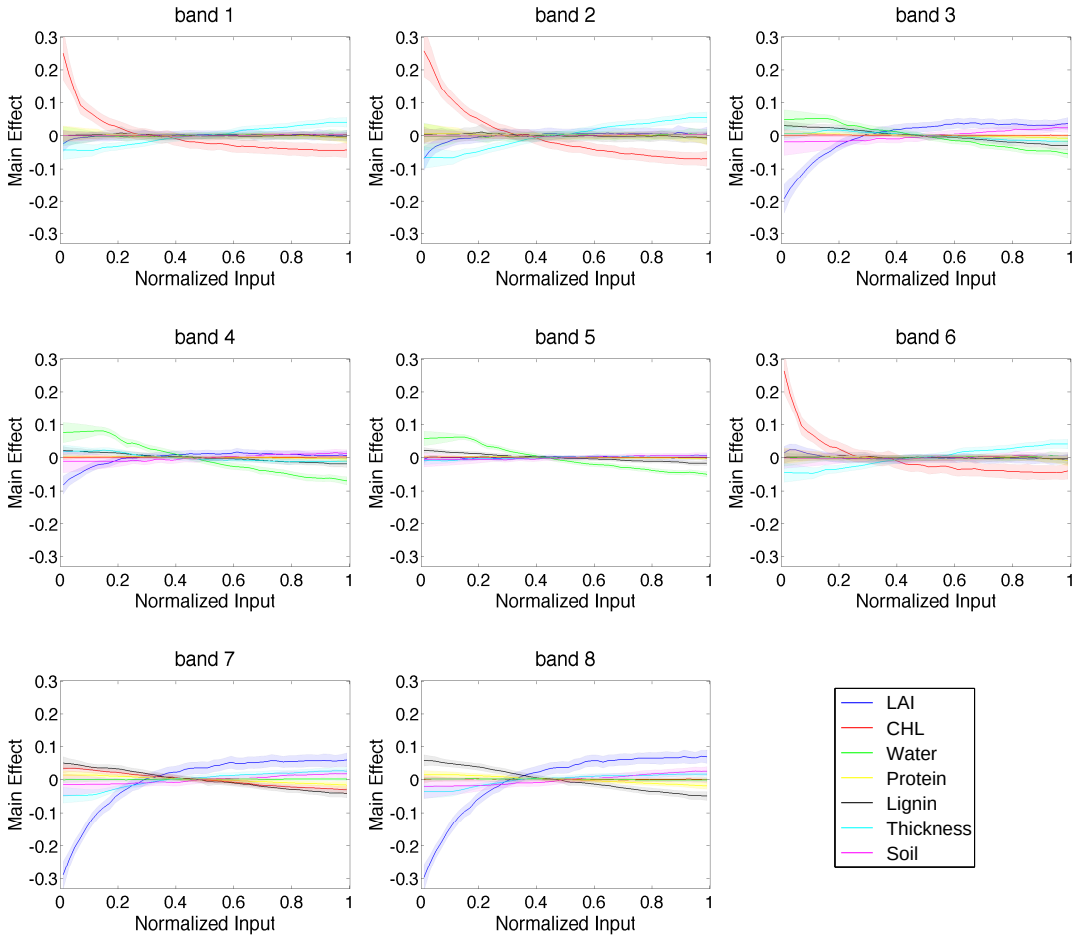


Figure 2.2: Posterior point estimates $\pm$ 2 standard deviations of the main effects for the LCM simulator at 8 MODIS bands.

output. For near infrared (bands 3, 7 and 8), the LCM is most sensitive to LAI, where an increase in LAI produces an increase in the LCM output. Finally, for short infrared bands (bands 4 and 5), the chlorophyll effect is diminished, while LAI and water are dominant for band 4, and water becomes more influential for band 5. In general, we observe that all dominant inputs have non-linear main effects.

Figure 2.3 shows boxplots of posterior samples for the first-order and total sensitivity

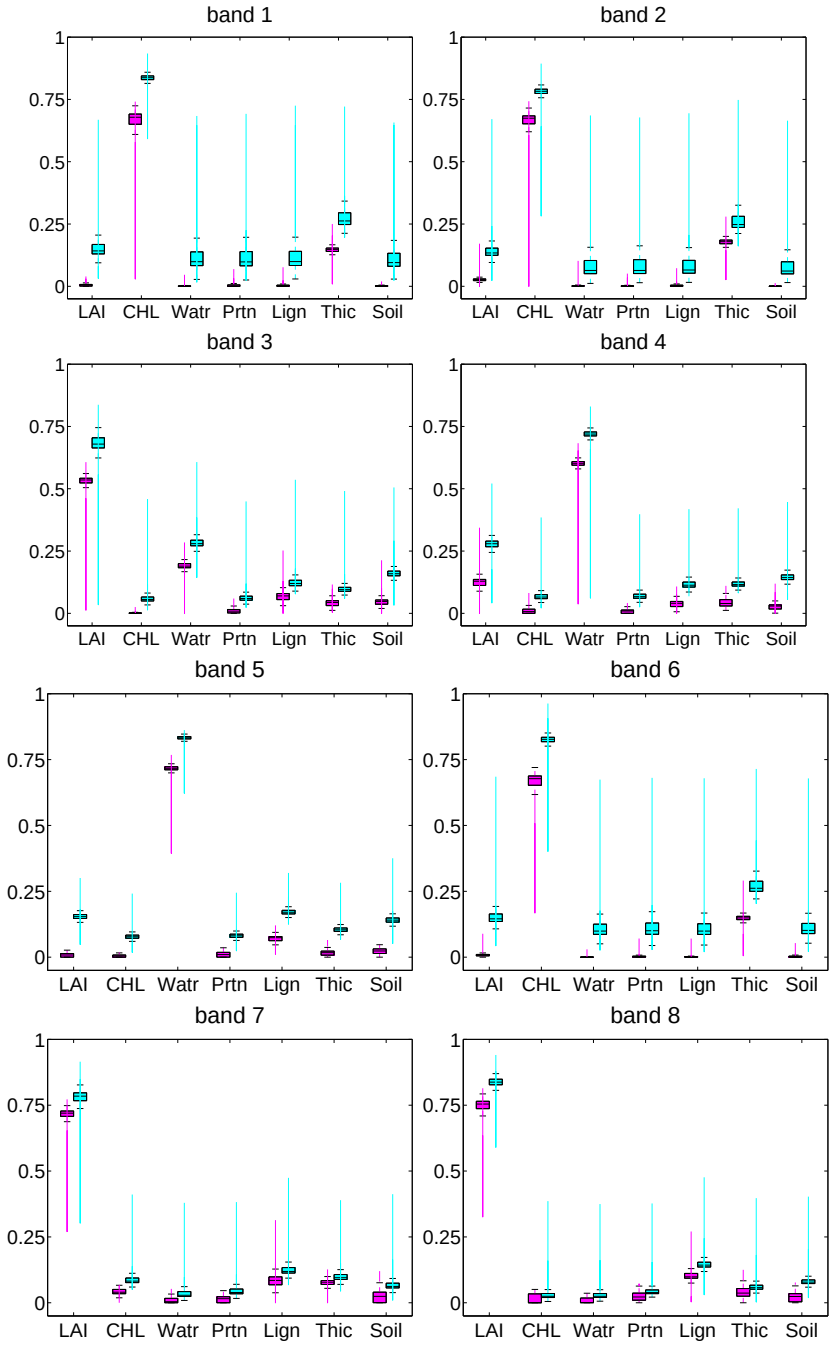


Figure 2.3: Posterior distributions of sensitivity indices for the LCM inputs at 8 MODIS bands; first-order sensitivity indices in magenta, and total sensitivity indices in cyan. The horizontal line inside a box indicates the median; the horizontal lines outside the box indicate the 95% region; and the whiskers stretching beyond them indicate the region of outliers.

indices, obtained using the method of Section 2.1.2.2. The results indicate that inputs with influential main effects are also major contributors to the variation in the LCM, i.e., they have large sensitivity indices (first-order and total). In particular, chlorophyll has the largest indices in the visible light spectrum (bands 1, 2 and 6), LAI has the largest indices in near infrared (bands 3, 7 and 8), and water has the largest indices in short infrared bands (bands 4 and 5).

Additionally, we observe that many inputs with negligible first-order sensitivity indices have non-negligible total sensitivity indices. A substantial difference between S_j and S_j^T of the j–th input indicates an important role of interaction terms involving that input on the variation in the output. For example, the median difference between first and total sensitivity indices for band 1 is greater than 0.10 for all inputs, which implies that interaction indices play a role in controlling uncertainty in the output. However, for band 7 the median difference between first and total sensitivity indices is between 0.02 and 0.06, which implies that perhaps the interaction indices have a negligible effect on output uncertainty. Finally, it is clear from the results in Figure 2.3 that comparing the entire distributions for the S_j, and contrasting them with the distributions for the S_j^T, provides a more informative approach to sensitivity analysis than comparison of point estimates for the sensitivity indices.

2.3 Discussion and Future Work

We have presented a framework for Bayesian global sensitivity analysis of deterministic simulators. The basis of the methodology is statistical model approximation (emulation) of the simulator output, which is built from Gaussian process (GP) priors. We have discussed

an approach to full inference for the main effects and sensitivity indices of the simulator inputs. The approach combines draws from the posterior predictive distribution of the GP emulator and Monte Carlo samples from the input uncertainty distribution to obtain samples from the posterior distributions of the sensitivity indices. Our approach fully accounts for uncertainty due to the emulation approximation as well as the Monte Carlo integration. We have also derived expressions which enable ready computing of Bayesian point estimates and standard errors for the main effects of the simulator inputs while fully accounting for uncertainty due to the GP emulator estimation . The methodology has been applied to the Leaf-Canopy Model (LCM), a radiative transfer model for the interaction of sunlight with vegetation, to identify the most influential inputs at different spectral bands.

A practically important extension of the work for the LCM involves calibration using remote-sensed and LCM simulator data. In particular, we are interested in estimating the distribution of LAI, which represents the aggregate amount of leaf material in a region, and so it is useful in describing the behavior of the vegetation surface on Earth. Hence, LAI a key parameter used in climate and ecological models that quantify the exchange of fluxes of energy, mass, and momentum between the land surface and the atmosphere (Bacour et al., 2006; Houborg et al., 2007).

Chapter 3

Emulation and Calibration of Stochastic Simulators

3.1 Introduction

Stochastic computer simulators are being increasingly used in technology and science to model random systems, e.g., in population dynamics, biological processes, queuing networks in a communication system, and nuclear interactions. For a simulator to be stochastic, the underlying model must contain an element of randomness that produces different results for the same input configuration. While analysis of deterministic computer simulators is a well studied field, research for analysis of stochastic computer simulators is still in its infancy, with emulation heavily based on Gaussian process (GP) modeling. In general, existing approaches make fairly strong assumptions regarding the stochastic mechanism for the computer model data and, when considered in the analysis, the field data.

While the need for quick statistical estimates of computationally expensive stochastic simulators is becoming more recognized by the scientific community, examples of emulation of stochastic simulators are still few and far between (Wilkinson, 2009), with the emerging literature relying heavily on GP modeling. For example, Iooss et al. (2009) model the stochastic simulator output using a parametric distribution that belongs to the exponential family. A link function is specified, and the mean and dispersion parameters of the simulator output distribution are modeled by two interlinked, generalized additive models, or GPs. Henderson et al. (2009, 2010) model the logit transformation of the simulator output with a normal distribution and place two independent GP priors on its mean and log standard deviation. While the authors propose a fully Bayesian model, the actual implementation of this approach involves several simplifications including fixing the values of the unknown mean and variance of the normal distribution to their sample estimate in order to train the emulator. Then, once the emulator is trained, the posterior distributions of the GPs are replaced with their conditional (posterior predictive) means, and the GP parameters are replaced with their posterior means.

With regard to calibration, a small number of papers, within the computational biological community, have discussed Bayesian approaches by placing priors on the model's calibration parameters, and then using computationally expensive simulator runs (or non-statistical approximations) and observed data to infer the distributions of the calibration parameters; see, for example, Golightly and Wilkinson (2006), Boys et al. (2008), and Golightly and Wilkinson (2008). In our review of the statistical literature, Henderson et al. (2009, 2010) use an emulation based approach to Bayesian calibration, where the analysis is split into two stages analogous to the approach in Bayarri et al. (2007a). However, the authors opt for a semi-Bayesian implemen-

tation, as discussed above, to make the MCMC more computationally feasible. Also, Gramacy and Lee (2010) introduce a framework for optimization and calibration of deterministic simulators and show that it can be extended to stochastic simulators based on a GP prior for the mean of the simulator output distribution.

In this chapter, we propose a new framework for emulation and calibration of stochastic simulators, where the emulator is built using a flexible nonparametric mixture model for the joint distribution of the simulator inputs and output(s). In particular, we work with Dirichlet process (DP) mixtures (Müller and Quintana, 2004) and use a curve fitting approach to build the emulator through the implied conditional distribution for the output given the inputs; see section 3.3.1. The nonparametric mixture model allows for greater flexibility than traditional approaches based on GP priors by relaxing assumptions of stationarity and normality; it models the entire distribution associated with the simulator, and not just the mean of the output process; and it can more readily be extended to incorporate multivariate, discrete/continuous output from the stochastic simulator.

For calibration, we develop an approach to link the simulator data with field observed data using a two-stage approach analogous to the one discussed in Bayarri et al. (2007a). In the first stage, the posterior distribution of the emulation model parameters are estimated solely based on simulator data, and then in the calibration stage, the posterior distribution of the calibration parameters is approximated based on the combined simulator and field data as well as the estimated posteriors of the emulation model parameters; see Section 3.3.2. The DP mixture approach to emulation and calibration is illustrated using two synthetic examples; see Sections 3.3.3 and 3.4.

Before we present the nonparametric mixture methodology, we examine a GP approach to emulation and calibration analogous to that considered by Henderson et al. (2009, 2010). This allows us to connect with the literature and compare our methods to the GP prior approach. Specifically, we study a simpler emulator by giving the mean of the normal simulator output distribution a GP prior, while giving the variance a parametric prior; see Section 3.2.1. For calibration, a similar model is used for the field data, and then a fully Bayesian inferential approach is developed as well as a two-stage approach similar to that used in Bayarri et al. (2007a); see Section 3.2.2. The GP prior approach is illustrated in the example given in Section 3.2.3.

3.2 The Gaussian process approach to emulation and calibration

The GP is commonly used as a prior model for the output of deterministic simulators, and, recently, as the prior model for a parameter of a parametric distribution in order to model the distribution of a stochastic simulator's output. Here, we emulate a stochastic simulator by modeling the distribution of its output using a normal distribution, with a mean that has a GP prior and a variance that has an inverse-gamma prior; see Section 3.2.1. Then, calibration is performed using a similar model for field data; see Section 3.2.2.

3.2.1 Emulation

In their work, Henderson et al. (2009, 2010) model the output of the stochastic simulator using a normal distribution and place two independent GP priors, with stationary co-

variance functions, on its mean and log standard deviation. Here, we focus on using the non-parametric prior for the mean only. Our derivations are presented for a simulator that has a calibration input, t, a control input, x, and univariate output y. The work presented here can be extended when the number of inputs is greater than two. However, in practice, analysis is significantly complicated under the GP approach with a large number of inputs, especially with respect to prior specification and posterior simulation for the covariance function parameters; moreover, the Gaussianity and customary stationarity assumptions become suspect with higher dimensional input spaces. The new approach to emulation, developed in Section 3.3.1, overcomes some of these restrictions.

Suppose the design vector of the calibration input is $\boldsymbol{t}^* = (t_1^*, \ldots, t_m^*)$, and the design vector of the control input is $\boldsymbol{x}^* = (x_1^*, \ldots, x_m^*)$. Let the simulator output be $\boldsymbol{y} = (y_1, \ldots, y_m)$, corresponding to design matrix $(\boldsymbol{t}^*, \boldsymbol{x}^*)$. Thus, the simulator data, $D^* = \{(y_j, t_j^*, x_j^*) : j = 1, \ldots, m\}$. We consider the following model for the simulator data:

$$y_j \mid \eta(\cdot), \tau^2 \overset{ind}{\sim} N(\eta(t_j^*, x_j^*), \tau^2)\ j = 1, \ldots, m.$$

with a GP prior assigned to the random function $\eta(\cdot)$, which defines the mean of the simulator output distribution. We work with a simple GP prior specification involving a constant mean function, μ, and a product power exponential correlation function, with range parameters $\boldsymbol{\phi} = (\phi_1, \phi_1)$, and constant variance λ^2. For the simulator data, this induces an m–variate normal distribution with mean vector $\mu \mathbf{1}_m$, where $\mathbf{1}_m$ is the $m \times 1$ vector with all elements equal to one, and covariance matrix $\lambda^2 R_{\boldsymbol{\phi}}$, with $R_{\boldsymbol{\phi}}$ the correlation matrix with (i, j)–th element $\exp(-\phi_1 \mid t_i^* - t_j^* \mid^{a_1} - \phi_2 \mid x_i^* - x_j^* \mid^{a_2})$, for $i = 1, \ldots, m$, $j = 1, \ldots, m$. We fix $a_1 = a_2 = a$ based on prior knowledge regarding $\eta(\cdot)$ and/or computational considerations. Then we place

the following standard priors on the parameters of the GP: $\mu \sim N(a_\mu, b_\mu)$, $\tau^2 \sim IG(a_{\tau^2}, b_{\tau^2})$, $\lambda^2 \sim IG(a_{\lambda^2}, b_{\lambda^2})$, and $p(\phi_1) = p(\phi_2) = \text{Unif}(0, b_{\boldsymbol{\phi}})$. Here, IG denotes the inverse-gamma distribution.

Denote the m induced values of the stochastic process η at the design input points by $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m)$, where $\delta_j = \eta(t_j^*, x_j^*)$, $j = 1, \ldots, m$. Then the joint posterior is given by

$$p(\boldsymbol{\delta}, \tau^2, \mu, \lambda^2, \boldsymbol{\phi} \mid D^*) \propto \left(\prod_{j=1}^{m} N(y_j; \delta_j, \tau^2) \right) \times N_m\left(\boldsymbol{\delta}; \mu \mathbf{1}_m, \lambda^2 R_{\boldsymbol{\phi}}\right) \tag{3.1}$$
$$\times IG(\tau^2; a_{\tau^2}, b_{\tau^2}) \times N(\mu; a_\mu, b_\mu) \times IG(\lambda^2; a_{\lambda^2}, b_{\lambda^2}) \times p(\boldsymbol{\phi}).$$

Samples from $p(\boldsymbol{\delta}, \tau^2, \mu, \lambda^2, \boldsymbol{\phi} \mid D^*)$ are obtained using MCMC posterior simulation as discussed in Appendix A.3. Then, interpolation over a grid of input values is carried out using the posterior predictive distribution. For a generic input, $\boldsymbol{\nu} = (t_0, x_0)$, which is not part of the design, let the corresponding output be y_0. The way in which the posterior predictive distribution, $p(y_0 \mid D^*)$, is obtained depends on whether or not one marginalizes over the GP prior. For instance, if we proceed without marginalization, then

$$p(y_0 \mid D^*, (t_0, x_0)) = \int N(y_0; \delta_0, \tau^2) p(\delta_0 \mid \boldsymbol{\delta}, \mu, \lambda^2, \boldsymbol{\phi}) p(\boldsymbol{\delta}, \tau^2, \mu, \lambda^2, \boldsymbol{\phi} \mid D^*)$$
$$\times d(\delta_0, \boldsymbol{\delta}, \tau^2, \mu, \lambda^2, \boldsymbol{\phi}), \tag{3.2}$$

where δ_0 is the predicted mean at the new input. Here, the full model, including the new observation, is given by

$$p(y_0,\delta_0,\boldsymbol{\delta},\tau^2,\mu,\lambda^2,\boldsymbol{\phi} \mid D^*) \propto N(y_0;\,\delta_0,\tau^2)\prod_{j=1}^{m} N(y_j;\delta_j\,,\tau^2\,)$$

$$\times N_{m+1}\left(\begin{pmatrix}\delta_0\\ \boldsymbol{\delta}\end{pmatrix};\,\mu\mathbf{1}_{m+1}\,,\,\lambda^2\begin{pmatrix}1 & R_{\boldsymbol{\phi}}(y_0,\boldsymbol{y})\\ R^T_{\boldsymbol{\phi}}(y_0,\boldsymbol{y}) & R_{\boldsymbol{\phi}}\end{pmatrix}\right)$$

$$\times\, p(\tau^2)\times p(\mu)\times p(\lambda^2)\times p(\boldsymbol{\phi}), \tag{3.3}$$

where $R_{\boldsymbol{\phi}}(y_0,\boldsymbol{y})$ is the $1\times m$ vector with j-th element $\exp\{-\phi_1 \mid t_0^*-t_j^* \mid^a -\phi_2 \mid x_0^*-x_j^* \mid^a\}$, for $j=1,\ldots,m$. Hence, $p(\delta_0 \mid \boldsymbol{\delta},\mu,\lambda^2,\boldsymbol{\phi})$ is a normal distribution with mean $\mu + R_{\boldsymbol{\phi}}(y_0,\boldsymbol{y})\, R_{\boldsymbol{\phi}}^{-1}$ $(\boldsymbol{\delta}-\mu\mathbf{1}_m)$, and variance $\lambda^2\left(1-R_{\boldsymbol{\phi}}(y_0,\boldsymbol{y})R_{\boldsymbol{\phi}}^{-1}R^T_{\boldsymbol{\phi}}(y_0,\boldsymbol{y})\right)$. The distribution of $\delta_0 \mid \boldsymbol{\delta};\mu,\lambda^2,\boldsymbol{\phi}$ is then used together with the posterior samples for the parameters of the emulation model to sample from the posterior predictive distribution, using (3.2). The approach can be extended to predict at any new set of new input points, where the new sampling is now from multivariate conditional normal distributions.

The distribution of δ_0 may be compared with the mean of the simulator output at $\boldsymbol{v}=(t_0,x_0)$ to check if the emulator is providing a reasonable estimate. Another way to check if the emulator is doing a reasonable job is by looking at the distribution of $p(y_0 \mid D^*)$ at a fixed new input $\boldsymbol{v}=(t_0,x_0)$ and comparing it to the distribution of the simulator output at that input. Starting with an appropriate grid over the input space, the MCMC sample of the emulator parameters is used to generate normal densities of y_0 over each grid point. Then, posterior mean and interval estimates for the output densities may be constructed and compared to the density of the simulator output at each grid point.

3.2.2 Calibration

Let the unknown calibration input be θ, the control input be x, and the field observation be z. We assume the field data responses arise from a normal distribution with only its mean depending on the calibration and control input variables, i.e., $z \sim N(\mu(\theta,x),\sigma^2)$. We can then follow standard practice for deterministic simulators to model, in general, $\mu(\theta,x) = \eta(\theta,x)+b(x)$, where $b(x)$ is the bias function. In order to carry out calibration, we need a hierarchical model that links the field observations with the simulator data. Suppose that we have n field observations with outputs $\boldsymbol{z} = (z_1,\ldots,z_n)$ corresponding to field control input vector $\boldsymbol{x} = (x_1,\ldots,x_n)$ (i.e., the field data $D = \{(z_i,x_i) : i = 1,\ldots,n\}$), and recall from section 3.2.1, the simulator data is $D^* = \{(y_j,t_j^*,x_j^*) : j = 1,\ldots,m\}$. Assuming that the simulator is a good representation of reality (ignoring bias), the hierarchical model for both sources of data (simulator and field) becomes,

$$z_i \mid \eta(\cdot),\theta,\sigma^2 \overset{ind}{\sim} N(\eta(\theta,x_i),\sigma^2),\ i = 1,\ldots,n,$$

$$y_j \mid \eta(\cdot),\tau^2 \overset{ind}{\sim} N(\eta(t_j^*,x_j^*),\tau^2),\ j = 1,\ldots,m,$$

with the GP prior for $\eta(\cdot)$ defined in Section 3.2.1. We place the same standard priors on the parameters of the GP, an inverse-gamma prior on σ^2, i.e., $\sigma^2 \sim IG(a_{\sigma^2},b_{\sigma^2})$, and an appropriate prior distribution for θ; in the examples studied later in the chapter, we experiment with uniform priors over a plausible range for the calibration parameter.

3.2.2.1 MCMC approach to full inference for calibration

Denote the parameters of the GP prior for η by $\boldsymbol{\psi}$. Then,

$$p(\theta,\boldsymbol{\delta},\sigma^2,\tau^2,\boldsymbol{\psi}\mid D,D^*)\propto\prod_{i=1}^{n}N(z_i;\eta(\theta,x_i),\sigma^2)\prod_{j=1}^{m}N(y_j;\eta(t_j^*,x_j^*),\tau^2)$$
$$\times N_{n+m}\left(\boldsymbol{\delta}_{n+m};\,\mu\mathbf{1}_{n+m},\,\lambda^2\begin{pmatrix}R_{\boldsymbol{\phi}}(\boldsymbol{z}) & R_{\boldsymbol{\phi}}(\boldsymbol{z},\boldsymbol{y})\\ R_{\boldsymbol{\phi}}^T(\boldsymbol{z},\boldsymbol{y}) & R_{\boldsymbol{\phi}}(\boldsymbol{y})\end{pmatrix}\right)$$
$$\times p(\theta)\times IG(\sigma^2;a_{\sigma^2},b_{\sigma^2})\times IG(\tau^2;a_{\tau^2},b_{\tau^2})\times p(\boldsymbol{\psi}),$$

where $R_{\boldsymbol{\phi}}(\boldsymbol{z})=\exp(-\phi_2\mid x_p-x_q\mid^a)$, for $p=1,\ldots,n,\ q=1,\ldots,n$; $R_{\boldsymbol{\phi}}(\boldsymbol{z},\boldsymbol{y})=\exp\{-\phi_1\mid\theta-t_q^*\mid^a$ $-\phi_2\mid x_p-x_q^*\mid^a\}$, for $p=1,\ldots,n,q=1,\ldots,m$; $R_{\boldsymbol{\phi}}(\boldsymbol{y})=\exp\{-\phi_1\mid t_p^*-t_q^*\mid^a-\phi_2\mid x_p^*-x_q^*\mid^a\}$, $p=1,\ldots,m,q=1,\ldots,m$. Note that the vector $\boldsymbol{\delta}$ in this joint posterior extends the one given in (3.1), where now for the first n elements, $\delta_i=\eta(\theta,x_i)$.

Marginalizing the function $\eta(\cdot)$ over its GP prior, we obtain

$$p(\theta,\sigma^2,\tau^2,\boldsymbol{\psi}\mid D,D^*)\propto$$
$$N_{n+m}\left(\begin{pmatrix}\boldsymbol{z}\\ \boldsymbol{y}\end{pmatrix};\mu\mathbf{1}_{n+m},\Sigma=\begin{pmatrix}\sigma^2I_n & \mathscr{O}_{n\times m}\\ \mathscr{O}_{m\times n} & \tau^2I_m\end{pmatrix}+\lambda^2\begin{pmatrix}R_{\boldsymbol{\phi}}(\boldsymbol{z}) & R_{\boldsymbol{\phi}}(\boldsymbol{z},\boldsymbol{y})\\ R_{\boldsymbol{\phi}}^T(\boldsymbol{z},\boldsymbol{y}) & R_{\boldsymbol{\phi}}(\boldsymbol{y})\end{pmatrix}\right)$$
$$\times p(\theta)\times IG(\sigma;a_{\sigma^2},b_{\sigma^2})\times IG(\tau^2;a_{\tau^2},b_{\tau^2})\times p(\boldsymbol{\psi}) \tag{3.4}$$

where $\mathscr{O}_{n\times m}$ and $\mathscr{O}_{m\times n}$ are the zero matrices of sizes $n\times m$ and $m\times n$ respectively.

Sampling from the posterior distribution of the calibration parameter proceeds by MCMC simulation from (3.4). Let $\boldsymbol{w}=(\boldsymbol{z},\boldsymbol{y})$. We find that only the posterior full condi-

tional for μ has a closed form. In particular, $\mu \mid \theta, \sigma^2, \tau^2, \lambda, \boldsymbol{\phi}, D, D^*$ is normal with mean $m = S\left(\mathbf{1}_{n+m}^T \Sigma^{-1} \boldsymbol{w} + a_\mu / b_\mu\right)$, and variance $S = \left(\mathbf{1}_{n+m}^T \Sigma^{-1} \mathbf{1}_{n+m} + 1/b_\mu\right)^{-1}$. Thus, we end up with one Gibbs step for μ, and M-H steps for all the other parameters in the calibration model. For σ^2, τ^2, λ^2, we propose on the log scale from appropriate normal distributions; for $\boldsymbol{\phi}$, we propose on the log scale from a bivariate-normal; and for θ, we propose from a normal distribution. As mentioned in the introduction of this thesis, care is needed to tune those proposals in order to efficiently explore the posterior distributions.

In this work, we found the proposal for $\boldsymbol{\phi}$ to be the most challenging to tune. Obviously, this challenge is amplified when the number of inputs is large, or even moderate, since the addition of each input requires adding a corresponding range of dependence parameter to the model, leading to a growing covariance matrix for their proposal distribution, where each element of this matrix requires tuning. Additionally, with more inputs, new runs of the simulator will be required increasing the dimension of the covariance matrix in the likelihood, which has to be inverted at each MCMC iteration.

3.2.3 Illustrative toy example

To illustrate the methods discussed in Sections 3.2.1 and 3.2.2, we construct a simple stochastic simulator with two inputs and one output such that $y \sim N(2x^*t^* + \cos(t^*), 0.07^2)$, and we consider inputs over the unit square. Figure 3.1 shows a plot of the mean function of this simulator. Obviously, this synthetic example does not correspond to a realistic stochastic simulator—it is based on a normal distribution throughout the input space, changing only in the mean with the inputs and having a constant variance. It is used mainly to check the performance

of the GP approach developed above. Using a latin hypercube design over the two-dimensional input space, where $x^* \in (0,1)$ and $t^* \in (0,1)$ (see Figure 3.2 (a)), we obtain $m = 100$ simulator outputs over this design. Figure 3.2 (b) shows these generated outputs plotted around the mean surface of the simulator. Here, the underlying distribution of the simulator is a normal distribution with a smoothly varying mean, so we expect that the GP emulator to capture its shape reasonably well.

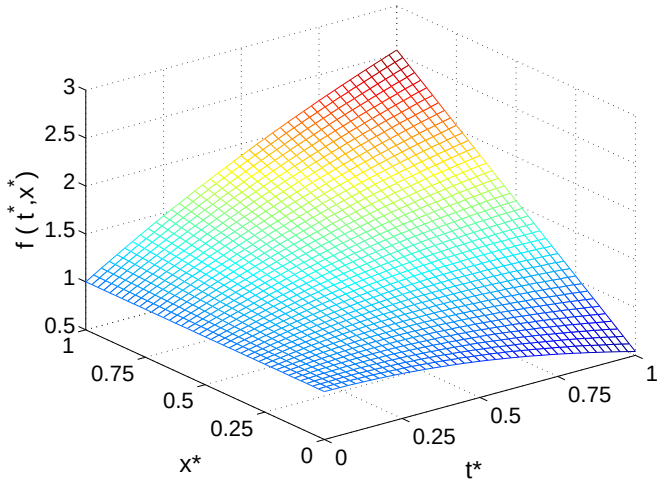
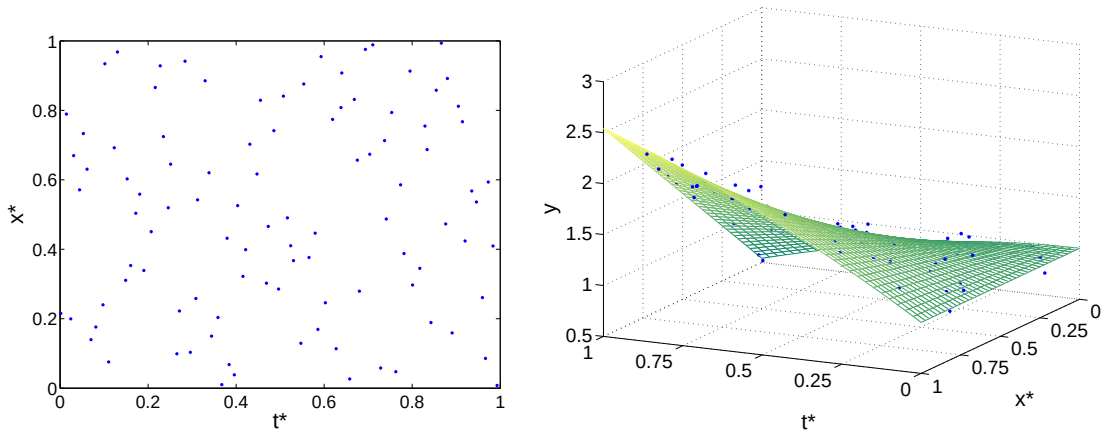


Figure 3.1: Surface corresponding to $f(t^*,x^*) = 2x^*t^* + \cos(t^*)$.

We fit this model using the exponential correlation function ($a = 1$), and we specify the priors for the model parameters based on vague knowledge of the data's center and range; see prior specification discussion in Appendix A. Specifically, we let $p(\mu) \sim N(\text{mean}(\boldsymbol{y}), 20 \times \text{Var}(\boldsymbol{y}))$, $\tau^2 \sim IG(2.02, 0.15)$, $\lambda^2 \sim IG(2.5, 3.5)$, and give ϕ_1 and ϕ_2 each a Unif$(0, 10)$ prior. Here, the MCMC is run for a total of $42,000$ iterations, with the first $2,000$ discarded as burn-in. Figure 3.3 shows posterior density plots of the parameters of the model. We observe reasonably good learning for all parameters.



(a) (b)

Figure 3.2: (a) Latin hypercube design over the two dimensional input space. (b) Simulator data plotted in blue around the mean surface of the simulator.

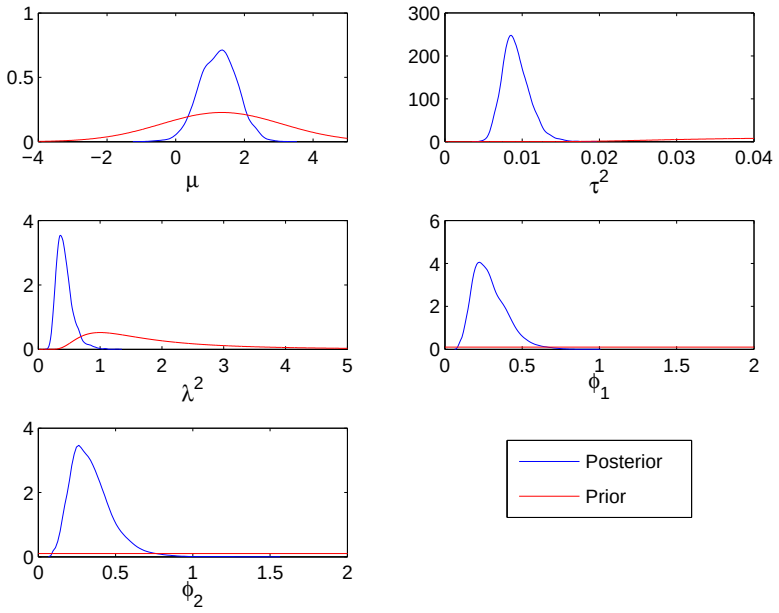


Figure 3.3: Density plots of the parameters of the emulation model using the exponential correlation function and independent Unif$(0, 10)$ priors for ϕ_1 and ϕ_2.

We also fit the emulation model using the correlation function with $a = 1.99$ (since using $a = 2$, which is the specification for the Gaussian correlation function, leads to numerical singularities in the covariance matrix). We use similar priors to the ones used with the exponential correlation function for μ, τ^2, and λ^2, but we widen the range of the uniform priors of ϕ_1 and ϕ_2 to $(0, 30)$. The MCMC is run for 42,000 iterations, with the first $2,000$ iterations discarded as burn-in. Figure 3.4 summarizes posterior inference of the model parameters, where we observe good posterior learning for all them.

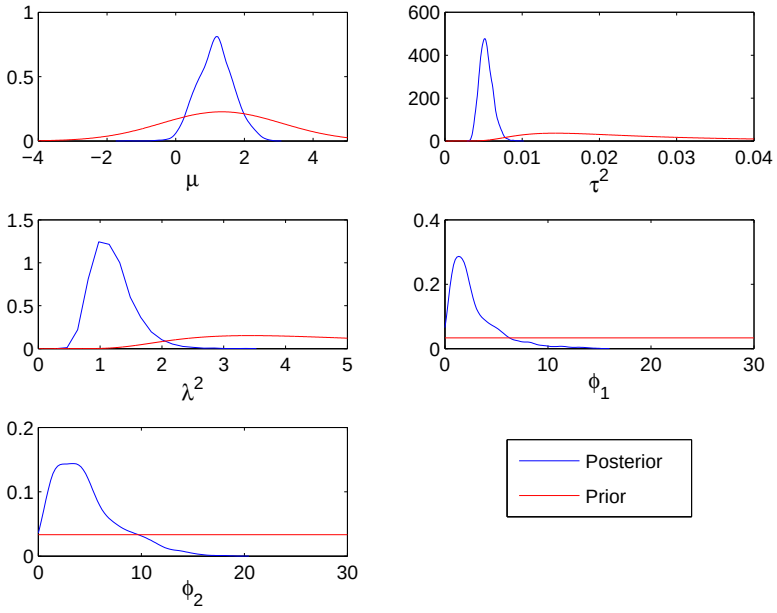
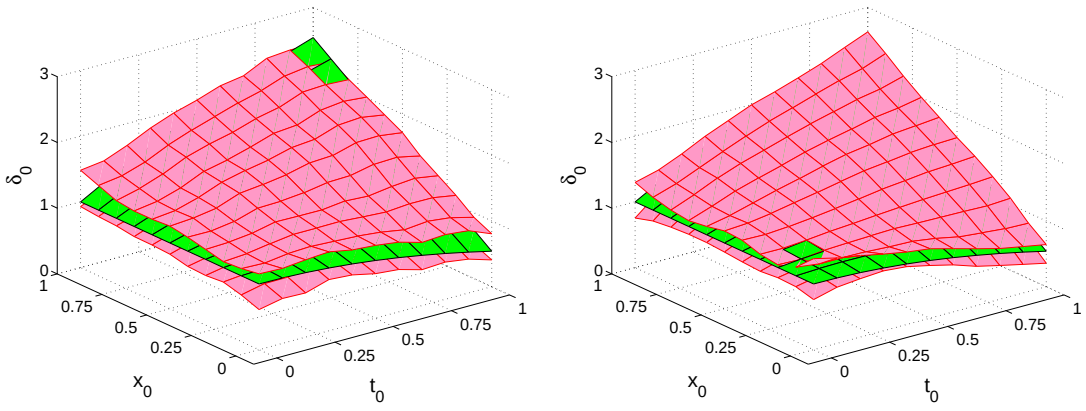


Figure 3.4: Density plots of the parameters of the emulation model using the correlation function with $a = 1.99$ and independent Unif$(0, 30)$ priors for ϕ_1 and ϕ_2.

Next, we obtain posterior predictive samples over a regular equally-spaced grid on the 2-dimensional input space. In Figure 3.5, we plot the mean surface of the simulator output distribution and the 95% posterior uncertainty surfaces of the predicted mean, δ_0, under the GP emulator; results are given for both correlation functions (corresponding to $a = 1$ and $a = 1.99$).

Under the exponential correlation function, the 95% surfaces "envelope" the mean surface of the simulator, except near input point $(x_0, t_0) = (1, 1)$, while under the Gaussian correlation function, the 95% surfaces miss the simulator mean near input point $(x_0, t_0) = (0.2, 0.1)$.



(a) (b)

Figure 3.5: 95% posterior uncertainty surfaces of the emulator's predicted mean, δ_0, (red) vs the mean surface of the simulator output (green) under (a) the exponential correlation function, and (b) the correlation function with $a = 1.99$.

Finally, we construct 90% interval estimates for the emulator output densities and compare them to the corresponding density of the simulator output at three input points. Figure 3.6 includes plots under the exponential correlation function, while Figure 3.7 includes those plots under the correlation function corresponding to $a = 1.99$. We find that inference for density estimation is more accurate using the second correlation function.

Next, we examine calibration, where we are interested in the scenario that only the range of the calibration parameter might be known a priori. We generate the synthetic field data, for a sample of size $n = 30$, using the same mean function we used to generate the simulator data, but with a larger variance than the one used to generate the synthetic simulator

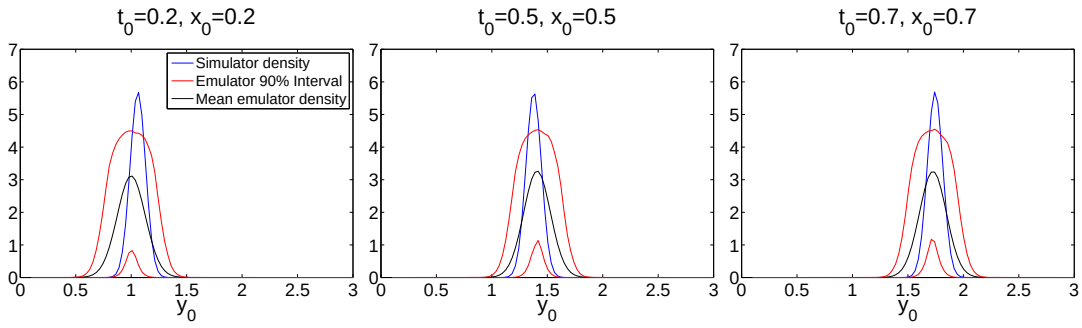


Figure 3.6: Comparison of the GP emulator's 95% density regions to the simulator's density, using the exponential correlation function ($a = 1$).

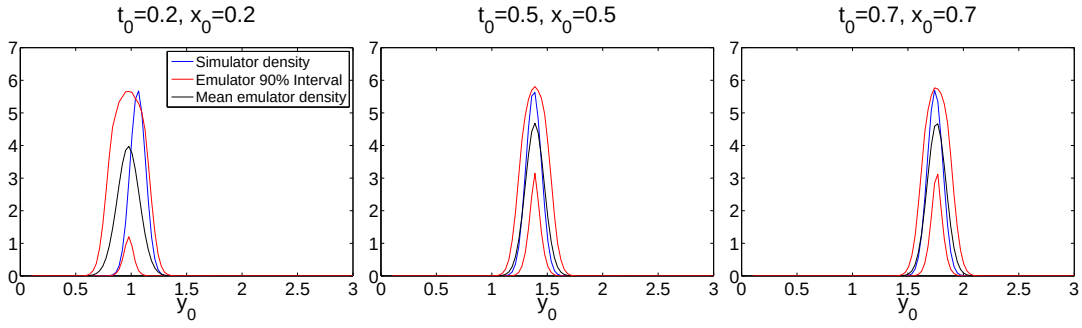
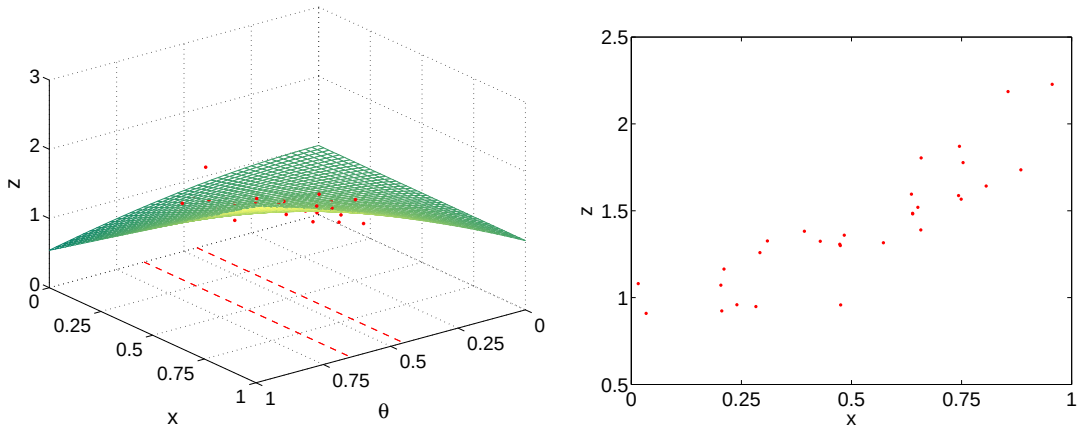


Figure 3.7: Comparison of the GP emulator's 95% density regions to the simulator's density, using the correlation function with $a = 1.99$.

data. Specifically, let z be the field observation corresponding to observed control input x and unobserved calibration input θ. Then, each observation z_i, $i = 1,\ldots,n$, is generated according to the following rule: $z_i \sim N(2x_i\theta + \cos(\theta), 0.14^2)$. In order to generate the field data, we give θ a normal distribution with mean 0.55 and standard deviation 0.05 and generate x from a uniform distribution on $(0,1)$. Figure 3.8 shows plots of the field data in 3-dimensions and 2-dimensions.



(a) (b)

Figure 3.8: (a) A 3-dimensional plot of the synthetic field data (in red) scattered about the mean surface of the process (green). The dotted red lines indicate the 95% probability region for the true distribution of θ. (b) Observed field data, where $n = 30$.

First, we obtain MCMC samples of the parameters for the calibration model using the exponential correlation function. We give μ a normal prior with mean equal to the mean of $\boldsymbol{w} = (\boldsymbol{y},\boldsymbol{z})$ and variance equal to $20 \times \text{Var}(\boldsymbol{w})$. We keep the same inverse-gamma priors for τ^2 and λ^2 as in the emulation step, assign σ^2 an inverse-gamma prior centered around the variance of $\boldsymbol{z}$, namely $IG(2.02, 0.23)$, and give θ a uniform prior on $(0,1)$. Figures 3.9 and 3.10 summarize the posterior inference for the model parameters. The posterior of θ captures the

true distribution reasonably well, but uncertainty is slightly over estimated. The posterior for σ^2 peaks near the true value of the observational error.

Next, we investigate posterior inference using the GP prior with correlation function corresponding to $a = 1.99$. We keep the same priors for all the parameters except ϕ_1 and ϕ_2, which we give uniform prior on $(0, 20)$. Figures 3.11 and 3.12 summarize the posterior inference. Here, there is little learning for λ^2 and μ. However, the posterior of σ^2 peaks near the true value of the observational error. Also, the posterior of θ resembles its true distribution reasonably accurately.

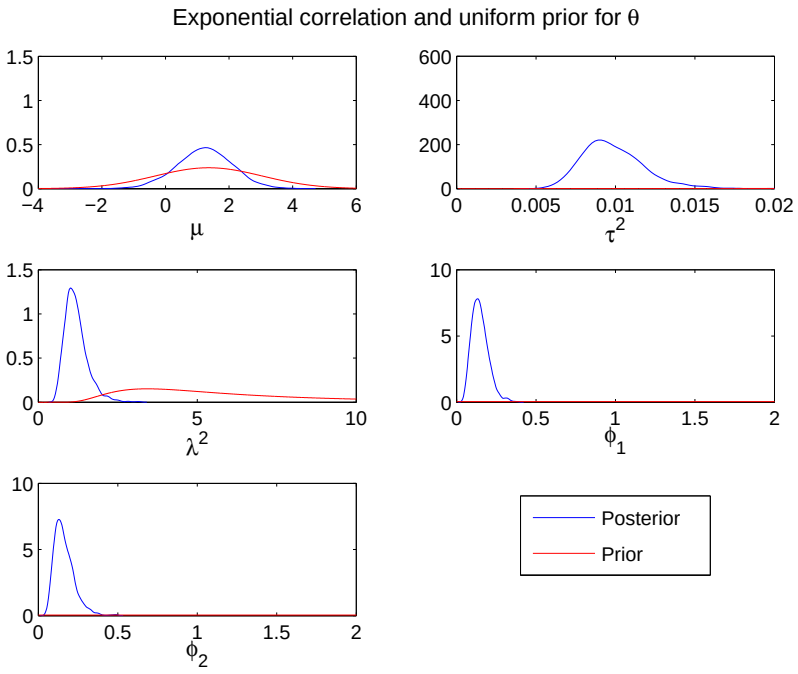
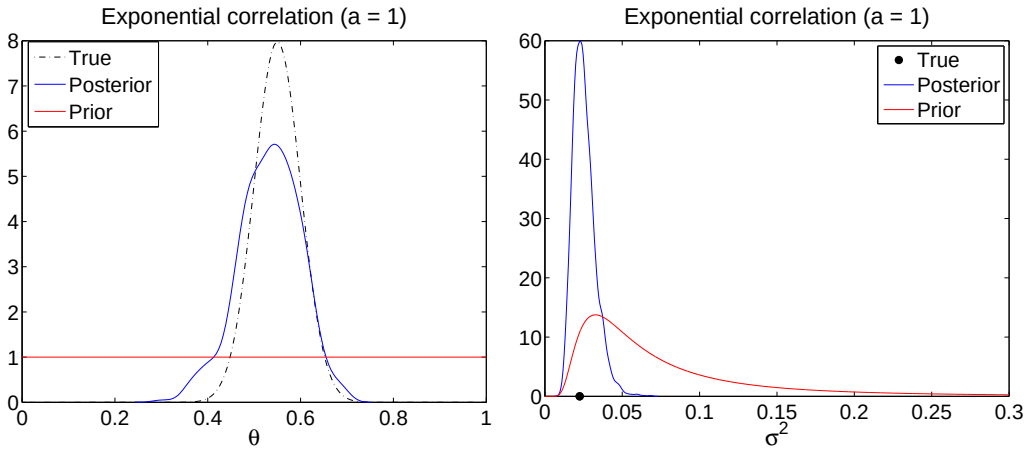


Figure 3.9: Posterior vs prior density plots of the parameters for the full calibration model, using the exponential correlation function and independent Unif$(0, 10)$ priors for ϕ_1 and ϕ_2.



(a) (b)

Figure 3.10: (a) The true density vs the prior and posterior densities for the calibration parameter, θ . (b) The true value of σ^2 as well as its prior and posterior density plots.

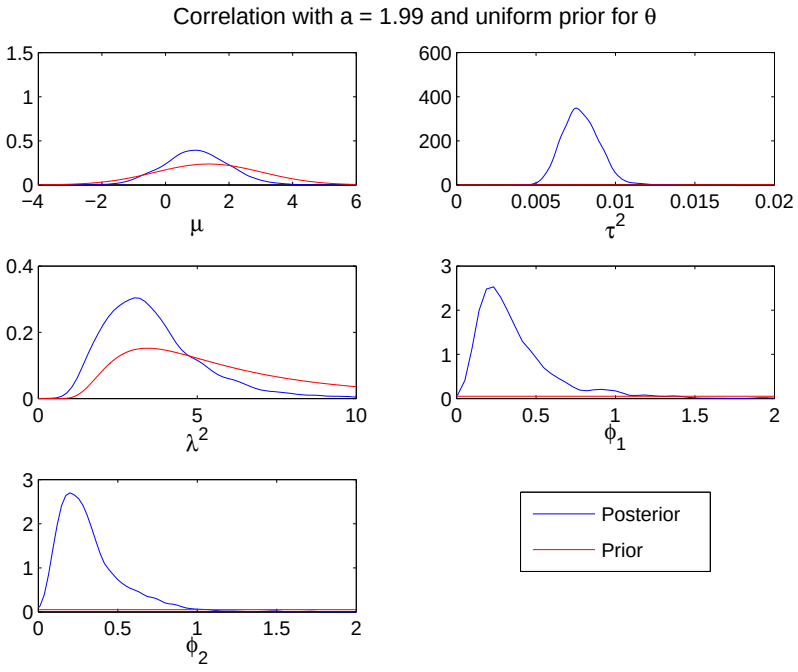
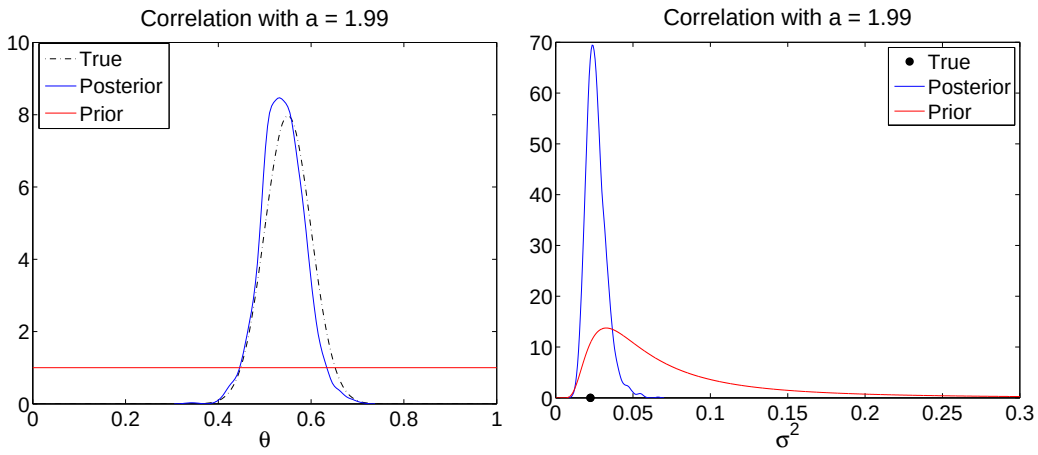


Figure 3.11: Posterior vs prior density plots of the calibration model parameters, using the correlation function with $a = 1.99$ and independent Unif$(0, 20)$ priors for ϕ_1 and ϕ_2.



(a) (b)

Figure 3.12: (a) The true density vs the prior and posterior densities for the calibration parameter, θ . (b) The true value of σ^2 as well as its prior and posterior density plots.

3.3 Emulation and calibration of stochastic simulators using Dirichlet process mixtures

Currently in the literature, there is limited work on the modeling of a stochastic computer simulator's output, with some recent work focusing still on using Gaussian process (GP) priors. In general, existing approaches make fairly strong assumptions regarding the stochastic mechanism for the computer model data and, when considered in the analysis, the field data. In Section 3.3.1, we introduce a new frame work for modeling the stochastic simulator output using a Bayesian nonparametric (BNP) mixture model. In Section 3.3.2, we develop a calibration model to link the simulator data with field observed data using a two-stage procedure. Then, the DP mixture methodology for emulation and calibration is illustrated using two synthetic examples; see Sections 3.3.3 and 3.4.

3.3.1 Nonparametric mixture modeling and inference for emulation

Mixture models are used for flexibly estimating a complex distribution of a population. Traditional finite mixture models assume the data come from a finite mixture of parametric densities, such as normals or multinomials, conditioned on a finite number of unknown parameters. Thus, they require grouping the data into a specific number of components (or latent clusters) as well as specifying the underlying parametric distributions. The BNP mixture model relaxes the requirement for specifying the number of underlying clusters and avoids dependence on parametric assumptions by allowing for the data to drive the number of clusters through assuming a random (nonparametric) mixing distribution. Due to advances in simulation-based model fitting, BNP mixture models are now employed in many areas of statistical inference, including density estimation, curve fitting, regression models for survival data, time series modeling, and regression with structured error distributions (e.g., West et al., 1994; Escobar and West, 1995; Müller et al., 1996; Kuo and Mallick, 1997; Müller et al., 1997; Hirano, 2002; Gelfand and Kottas, 2003; Dunson, 2005; Kottas and Krnjajić, 2009).

We will begin with emulation, thus considering data from the stochastic computer model. Denote the controllable input by x, the calibration input by t, and the corresponding output by y. Here, both the inputs and output may be multivariate. The stochastic simulator data, (y,t,x), are modeled using the mixture model

$$f(y,t,x;G) = \int k(y,t,x;\boldsymbol{\beta})dG(\beta), \tag{3.5}$$

where $k(y,t,x;\boldsymbol{\beta})$ is a parametric family of density functions (or kernels) indexed by the param-

eter vector $\boldsymbol{\beta}$, and G is random mixing distribution. Because G is random, $f(y,t,x;G)$ is also random. The generic mixture formulation in (3.5) can be simplified by considering a product form for the kernel, e.g., with separate kernel components for the outputs and the inputs or for y, t, and x. Moreover, more structured versions (e.g., for settings where viewing some of the inputs in x as random is less natural) can also be developed depending on the application.

The motivations and advantages of the BNP mixture modeling framework for emulation include:

- We can model a multivariate, discrete/continuous output from the stochastic simulator, more readily than under customary GP approaches.
- Unlike current emulation methods for stochastic simulators, we model nonparametrically the entire distribution associated with the simulator, and not just the mean of the output process (or the first two moments).
- Greater flexibility is allowed via G, the nonparametric mixing distribution, with regard to the resulting distributional shapes, which may have practical implications at the calibration stage by improving the inference for the calibration parameters.

In this work, we model the random distribution G with a Dirichlet process (DP) prior, which is one of the most popular Bayesian nonparametric priors. Formally introduced by Ferguson (1973), the DP is a stochastic process that generates (almost surely) discrete random distributions. Thus, it is a distribution over distributions on a particular space, $\mathscr{X}$. Two parameters are needed to specify a DP: a positive real number, α, and a parametric distribution, G_0. It is called a Dirichlet process because its finite dimensional marginal distribu-

tions are Dirichlet distributed. Specifically, let G_0 be a (parametric) distribution over $\mathscr{X}$ and $\alpha > 0$. Then, the DP generates random distributions G on $\mathscr{X}$ such that for any finite measurable partition, $B_1, \ldots, B_k$ of $\mathscr{X}$, $(G(B_1), \ldots, G(B_k)) \sim \text{Dirichlet}(\alpha G_0(B_1), \ldots, \alpha G_0(B_k))$. G_0, called the "base" distribution, is the mean of the DP; that is, for any measurable set $B \subset \mathscr{X}$, $\mathrm{E}(G(B)) = G_0(B)$, and α, called the concentration parameter, can be viewed as a precision parameter, since $\text{Var}(G(B)) = G_0(B)(1 - G_0(B))/(\alpha + 1)$. The larger α is, the smaller the variance, and the closer we expect a realization from the DP to be to the mean distribution G_0. We will use $DP(\alpha, B_0)$ to denote the DP prior with parameters α, and G_0.

The construction by Sethuraman (1994) provides a convenient representation for the DP, as it provides a constructive definition of the random discrete distributions generated by the DP. In particular, if G is a realization from $DP(\alpha, G_0)$, then it is (almost surely) of the form

$$G = \sum_{\ell=1}^{\infty} \omega_\ell \delta_{\vartheta_\ell}, \quad \vartheta_\ell \overset{iid}{\sim} G_0; \tag{3.6}$$

$$\omega_1 = V_1, \omega_\ell = V_\ell \prod_{j<\ell} (1 - V_j), \ell = 2, 3, \ldots, \text{ with } V_\ell \overset{iid}{\sim} \text{Beta}(1, \alpha)$$

That is, G has an (almost sure) representation as a countable mixture of point masses, where the locations, ϑ_ℓ, of the point masses arise independently from G_0, and their associated weights, ω_ℓ, are randomly generated by a "stick-breaking" procedure based on i.i.d. draws from a $\text{Beta}(1, \alpha)$ distribution.

An appealing feature of the DP prior model is that it can be centered around familiar parametric models, G_0. However, the discreteness constraint of the DP may be inappropriate in many applications. A simple extension to remove this constraint is to introduce a convolution

with a parametric kernel distribution $K(\cdot;\theta)$, defining a random mixture distribution through $F(\cdot;G) = \int K(\cdot;\theta)dG(\theta),\ G \sim DP(\alpha, G_0)$. The corresponding mixture density function is $f(\cdot;G) = \int k(\cdot;\theta)dG(\theta)$, where $k(\cdot;\theta)$ is the density of $K(\cdot;\theta)$. Such models, referred to as DP mixture models, are widely used in the Bayesian nonparametrics literature (e.g., Müller and Quintana, 2004).

We model the density of the joint distribution of the simulator inputs and output using the mixture model in (3.5) using a $DP(\alpha, G_0)$ prior for G. Here, k is a parametric kernel with mixing parameters $\boldsymbol{\beta}$, and $G_0 \equiv G_0(\boldsymbol{\beta})$ is an appropriate parametric distribution (with parameters $\boldsymbol{\psi}$) for the kernel mixing parameters. Then, the approximation of the stochastic simulator output density can be obtained through $f(y|(t,x);G) = f(y,t,x;G)/f(t,x;G)$, from which inference for any functional of the conditional density can arise, for instance, the mean of the emulator is constructed through the distribution of $E(Y \mid t,x;G)$. Letting $\boldsymbol{v} = (t,x)$, we can write

$$E(Y \mid \boldsymbol{v};G) = \int y\, f(y \mid \boldsymbol{v};G)\, dy = \int y \frac{f(y,\boldsymbol{v};G)}{f(\boldsymbol{v};G)}\, dy = \frac{1}{f(\boldsymbol{v};G)} \int \left\{ \int y\, f(y,\boldsymbol{v};\boldsymbol{\beta})\, dy \right\} dG(\boldsymbol{\beta}) \tag{3.7}$$

This expression (and other functionals of the conditional density) will depend on the form of the mixture kernel. In the remainder of this chapter, we work with a multivariate normal kernel, but other choices are possible depending on the application.

3.3.1.1 Posterior inference

As with section 3.2.1, let the design vector over the calibration input be $\boldsymbol{t}^* = (t_1, \ldots, t_m^*)$, the design vector over the control input be $\boldsymbol{x}^* = (x_1, \ldots, x_m^*)$, and the simulator output be $\boldsymbol{y} =$

$(y_1,\ldots,y_m)$, corresponding to design matrix $(\boldsymbol{t}^*,\boldsymbol{x}^*)$. Thus, the simulator data $D^* = \{(y_j,t_j^*,x_j^*) : j = 1,\ldots,m\}$. Let $\boldsymbol{u}_j = (y_j,t_j^*,x_j^*)$. For simpler notation, we let k be the trivariate normal kernel with $\boldsymbol{\beta}_j = (\tilde{\boldsymbol{\mu}}_j,\tilde{\Sigma}_j)$. In the presence of more inputs, the trivariate normal kernel can be easily extended to a higher dimensional multivariate normal to accommodate the higher dimensional input space.

Given the DP mixture model in (3.5), the hierarchical model formulation for the simulator data is given by

$$\begin{aligned} \boldsymbol{u}_j|\tilde{\boldsymbol{\mu}}_j,\tilde{\Sigma}_j &\overset{ind}{\sim} N_3(\boldsymbol{u}_j;\tilde{\boldsymbol{\mu}}_j,\tilde{\Sigma}_j),\ j = 1,\ldots,m \\ (\tilde{\boldsymbol{\mu}}_j,\tilde{\Sigma}_j)|G &\overset{iid}{\sim} G,\ j = 1,\ldots,m \\ G|\alpha,\boldsymbol{\psi} &\sim DP\left(\alpha,G_0(\boldsymbol{\psi})\right). \end{aligned} \tag{3.8}$$

Then, priors may be placed on α, and $\boldsymbol{\psi}$. Here, the MCMC approach to inference would follow standard approaches such as marginal posterior methods, where the model is marginalized over the random mixing distribution G (e.g., Escobar and West, 1995), or conditional methods such as the blocked Gibbs sampler (e.g., Ishwaran and James, 2001), where a truncation approximation is applied to G. In general, the blocked Gibbs sampler offers a more efficient approach to MCMC posterior simulation, and we thus follow this approach here.

Recall, under its constructive definition, G has an almost sure representation as $G = \sum_{\ell=1}^{\infty}\omega_\ell\delta_{\boldsymbol{\vartheta}_\ell}$. In order to obtain posterior inference for the emulator, we take advantage of the stick-breaking representation of the DP using a finite truncation approximation. Specifically, for a finite N, G in (3.6) is approximated by $G_N = \sum_{\ell=1}^{N} p_\ell\delta_{\boldsymbol{\vartheta}_\ell}$, where $\boldsymbol{\vartheta}_1,\ldots,\boldsymbol{\vartheta}_N \overset{iid}{\sim} G_0$, and the

weights $\boldsymbol{p} = (p_1, \ldots, p_N)$ arise through a stick-breaking process (with truncation):

$$p_1 = V_1;\ p_\ell = V_\ell \prod_{r=1}^{\ell-1}(1-V_r),\ \ell = 1,\ldots,N-1;\ p_N = \prod_{r=1}^{N-1}(1-V_r) = 1 - \sum_{\ell=1}^{N-1} p_\ell,$$

where $V_\ell \overset{iid}{\sim} \text{Beta}(1,\alpha)$, $\ell = 1,\ldots,N-1$. One way to specify N is through $E(\sum_{\ell=1}^{N} \omega_\ell)$, which can be shown to equal $1-(\alpha/(\alpha-1))^N$. Thus, N can be chosen such that $(\alpha/(\alpha+1))^N = \varepsilon$ for a small ε.

Next, we let $\boldsymbol{\vartheta}_\ell = (\boldsymbol{\mu}_\ell, \Sigma_\ell)$, for $\ell = 1,\ldots,N$, and introduce latent configuration parameters $\boldsymbol{L} = (L_1,\ldots,L_m)$, where each L_j takes values in $\{1,\ldots,N\}$ such that $L_j = \ell$ if and only if $(\tilde{\boldsymbol{\mu}}_j, \tilde{\Sigma}_j) = (\boldsymbol{\mu}_\ell, \Sigma_\ell)$, for $j = 1,\ldots,m; \ell = 1,\ldots,N$. Then, we complete the model with appropriate priors for all the hyperparameters and obtain the following hierarchical model:

$$\begin{aligned}
\boldsymbol{u}_j \mid (\boldsymbol{\mu},\boldsymbol{\Sigma}), L_j &\overset{ind}{\sim} \text{N}_3\left(\boldsymbol{u}_j; \boldsymbol{\mu}_{L_j}, \Sigma_{L_j}\right),\ j = 1,\ldots,m \\
L_j \mid \boldsymbol{p} &\overset{iid}{\sim} \sum_{\ell=1}^{N} p_\ell\, \delta_\ell(L_j),\ j = 1,\ldots,m \\
\boldsymbol{p} \mid \alpha \sim p(\boldsymbol{p};\alpha) &= \alpha^{N-1} p_N^{\alpha-1} (1-p_1)^{-1} (1-(p_1+p_2))^{-1} \times \cdots \times \left(1 - \sum_{\ell=1}^{N-2} p_\ell\right)^{-1} \\
(\boldsymbol{\mu}_\ell, \Sigma_\ell) \mid \boldsymbol{m}, V, S &\overset{iid}{\sim} G_0(\boldsymbol{\mu}_\ell, \Sigma_\ell; \boldsymbol{m}, V, S) = \text{N}_3(\boldsymbol{\mu}_\ell; \boldsymbol{m}, V)\ \text{IW}_3(\Sigma_\ell; \nu, S) \\
\alpha \sim \text{Gamma}\left(a_\alpha, b_\alpha\right), \boldsymbol{m} &\sim \text{N}_3\left(\boldsymbol{a_m}, B_{\boldsymbol{m}}\right), V \sim \text{IW}_3\left(a_V, B_V\right), S \sim \text{W}_3\left(a_S, B_s\right),
\end{aligned}$$

where $\text{W}_3\left(a_S, B_S\right)$ is a Wishart distribution with a_S degrees of freedom and symmetric positive definite 3×3 scale matrix B_S. Also, $\text{IW}_3\left(a_V, B_V\right)$ is an inverse-Wishart distribution with a_V degrees of freedom and symmetric positive definite 3×3 scale matrix B_V. The Wishart and

inverse-Wishart distributions are multivariate generalization of the gamma and inverse-gamma distributions, respectively, and provide conjugate priors for inverse covariance and covariance matrices. Note that the joint prior for $\boldsymbol{p} = p_1, \ldots, p_N$ given α, corresponds to a special case of the generalized Dirichlet distribution (Connor and Mosimann, 1969). Under this model, the joint posterior distribution, $p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{L}, \boldsymbol{p}, \alpha, \boldsymbol{m}, V, S \mid D^*)$, is proportional to

$$\prod_{j=1}^{m} \left\{ \mathrm{N}_3(\boldsymbol{u}_j; \boldsymbol{\mu}_{L_j}, \Sigma_{L_j}) p(L_j \mid \boldsymbol{p}) \right\} p(\boldsymbol{p}; \alpha) \left\{ \prod_{\ell=1}^{N} \mathrm{N}_3(\boldsymbol{\mu}_\ell; \boldsymbol{m}, V) \mathrm{IW}_3(\Sigma_\ell; \nu, S) \right\}$$
$$\times\, p(\alpha)\, p(\boldsymbol{m})\, p(V)\, p(S). \tag{3.9}$$

The details of sampling from the full conditional distributions of this posterior model are provided in Appendix B.

Next, let $\boldsymbol{v} = (t, x)$ be a generic input, and consider the mean of the emulator given in (3.7), which, here, is given by

$$\begin{aligned} E(Y \mid \boldsymbol{v}; G) &= \frac{1}{f(\boldsymbol{v}; G)} \int y \left\{ \int N_3(y, \boldsymbol{v}; \boldsymbol{\mu}, \Sigma)\, dG(\boldsymbol{\mu}, \Sigma) \right\} dy \\ &= \frac{1}{f(\boldsymbol{v}; G)} \int \left\{ \int y\, N_3(y, \boldsymbol{v}; \boldsymbol{\mu}, \Sigma)\, dy \right\} dG(\boldsymbol{\mu}, \Sigma) \\ &= \frac{1}{f(\boldsymbol{v}; G)} \int N_2(\boldsymbol{v}; \boldsymbol{\mu}^{\boldsymbol{v}}, \Sigma^{\boldsymbol{vv}}) \left\{ \int y\, N(y \mid \boldsymbol{v}; \boldsymbol{\mu}, \Sigma)\, dy \right\} dG(\boldsymbol{\mu}, \Sigma) \\ &= \frac{1}{f(\boldsymbol{v}; G)} \int N_2(\boldsymbol{v}; \boldsymbol{\mu}^{\boldsymbol{v}}, \Sigma^{\boldsymbol{vv}}) \left\{ \boldsymbol{\mu}^{y|\boldsymbol{v}} \right\} dG(\boldsymbol{\mu}, \Sigma) \end{aligned} \tag{3.10}$$

where $\boldsymbol{\mu}^{y|\boldsymbol{\nu}} = \boldsymbol{\mu}^{y} + \Sigma^{y\boldsymbol{\nu}}\left(\Sigma^{\boldsymbol{\nu}\boldsymbol{\nu}}\right)^{-1}\left(\boldsymbol{\nu} - \boldsymbol{\mu}^{\boldsymbol{\nu}}\right)$, and

$$\boldsymbol{\mu} = \begin{pmatrix} \mu^{y} \\ \boldsymbol{\mu}^{\boldsymbol{\nu}} \end{pmatrix} = \begin{pmatrix} \mu^{y} \\ \mu^{t} \\ \mu^{x} \end{pmatrix}, \quad \Sigma = \begin{pmatrix} \Sigma^{yy} & \Sigma^{y\boldsymbol{\nu}} \\ \Sigma^{\boldsymbol{\nu}y} & \Sigma^{\boldsymbol{\nu}\boldsymbol{\nu}} \end{pmatrix} = \begin{pmatrix} \Sigma^{yy} & \Sigma^{yt} & \Sigma^{yx} \\ \Sigma^{ty} & \Sigma^{tt} & \Sigma^{tx} \\ \Sigma^{xy} & \Sigma^{xt} & \Sigma^{xx} \end{pmatrix}.$$

Using the truncation approximation, G_N, to the mixing distribution G, we obtain

$$\begin{aligned} E\left(Y \mid \boldsymbol{\nu}; G_N\right) &= \frac{\sum_{\ell=1}^{N} p_\ell N_2\left(\boldsymbol{\nu}; \boldsymbol{\mu}_\ell^{\boldsymbol{\nu}}, \Sigma_\ell^{\boldsymbol{\nu}\boldsymbol{\nu}}\right)\left\{\mu_\ell^{y} + \Sigma_\ell^{y\boldsymbol{\nu}}\left(\Sigma_\ell^{\boldsymbol{\nu}\boldsymbol{\nu}}\right)^{-1}\left(\boldsymbol{\nu} - \boldsymbol{\mu}_\ell^{\nu}\right)\right\}}{\sum_{\ell=1}^{N} p_\ell N_2\left(\boldsymbol{\nu}; \boldsymbol{\mu}_\ell^{\boldsymbol{\nu}}, \Sigma_\ell^{\boldsymbol{\nu}\boldsymbol{\nu}}\right)} \\ &= \sum_{\ell=1}^{N} p'_\ell(\boldsymbol{\nu})\left\{\mu_\ell^{y} + \Sigma_\ell^{y\boldsymbol{\nu}}\left(\Sigma_\ell^{\boldsymbol{\nu}\boldsymbol{\nu}}\right)^{-1}\left(\boldsymbol{\nu} - \boldsymbol{\mu}_\ell^{\nu}\right)\right\}, \end{aligned}$$

where $p'_\ell(\boldsymbol{\nu}) = p_\ell N_2\left(\boldsymbol{\nu}; \boldsymbol{\mu}_\ell^{\boldsymbol{\nu}}, \Sigma_\ell^{\boldsymbol{\nu}\boldsymbol{\nu}}\right) / \left\{\sum_{\ell=1}^{N} p_\ell N_2\left(\boldsymbol{\nu}; \boldsymbol{\mu}_\ell^{\boldsymbol{\nu}}, \Sigma_\ell^{\boldsymbol{\nu}\boldsymbol{\nu}}\right)\right\}$. Thus, the mean of the simulator output distribution is modeled using a weighted mixture of "linear regressors" with weights that depend on the inputs, resulting in flexible inference for the mean of the simulator through density estimation for the join output-input distribution. Similarly, we obtain expressions for $E\left(Y \mid t; G_N\right)$, and $E\left(Y \mid x; G_N\right)$, with

$$E\left(Y \mid t; G_N\right) = \frac{\sum_{\ell=1}^{N} p_\ell N\left(t; \mu_\ell^{t}, \Sigma_\ell^{tt}\right)\left\{\mu_\ell^{y} + \Sigma_\ell^{yt}\left(\Sigma_\ell^{tt}\right)^{-1}\left(t - \mu_\ell^{t}\right)\right\}}{\sum_{\ell=1}^{N} p_\ell N\left(t; \mu_\ell^{t}, \Sigma_\ell^{tt}\right)}, \tag{3.11}$$

$$E\left(Y \mid x; G_N\right) = \frac{\sum_{\ell=1}^{N} p_\ell N\left(x; \mu_\ell^{x}, \Sigma_\ell^{xx}\right)\left\{\mu_\ell^{y} + \Sigma_\ell^{yx}\left(\Sigma_\ell^{xx}\right)^{-1}\left(x - \mu_\ell^{x}\right)\right\}}{\sum_{\ell=1}^{N} p_\ell N\left(x; \mu_\ell^{x}, \Sigma_\ell^{xx}\right)}. \tag{3.12}$$

Generalizing these expressions to a larger dimensional input space is easily done since they are based on the properties of the conditional normal distribution.

Thus, using the posterior samples of $\boldsymbol{\mu}$,$\boldsymbol{\Sigma}$, and $\boldsymbol{p}$, we can obtain the posterior distribu-

tion of these expectations. Inference for the conditional simulator output density $f(y \mid t, x; G_N)$ at any fixed value $\boldsymbol{\nu}_0 = (t_0, x_0)$ is available through

$$\begin{aligned} f(y \mid \boldsymbol{\nu}_0; G_N) &= \frac{f(y, \boldsymbol{\nu}_0; G_N)}{f(\boldsymbol{\nu}_0; G_N)} = \frac{\sum_{\ell=1}^{N} p_\ell N_3(y, \boldsymbol{\nu}_0; \boldsymbol{\mu}_\ell, \Sigma_\ell)}{\sum_{\ell=1}^{N} p_\ell N_2(\boldsymbol{\nu}_0; \boldsymbol{\mu}_\ell^{\boldsymbol{\nu}_0}, \Sigma_\ell^{\boldsymbol{\nu}_0\boldsymbol{\nu}_0})} \\ &= \frac{\sum_{\ell=1}^{N} p_\ell N_2(\boldsymbol{\nu}_0; \boldsymbol{\mu}_\ell^{\boldsymbol{\nu}_0}, \Sigma_\ell^{\boldsymbol{\nu}_0\boldsymbol{\nu}_0}) N(y \mid \boldsymbol{\nu}_0; M_\ell^y, S_\ell^y)}{\sum_{\ell=1}^{N} p_\ell N_2(\boldsymbol{\nu}_0; \boldsymbol{\mu}_\ell^{\boldsymbol{\nu}_0}, \Sigma_\ell^{\boldsymbol{\nu}_0\boldsymbol{\nu}_0})} \end{aligned}$$

where $\boldsymbol{\mu}_\ell^{\boldsymbol{\nu}_0}$, and $\Sigma_\ell^{\boldsymbol{\nu}_0\boldsymbol{\nu}_0}$ correspond to the mean and variance of the marginal normal distribution for $\boldsymbol{\nu}_0$ induced by $N_3(y, \boldsymbol{\nu}_0; \boldsymbol{\mu}_\ell, \Sigma_\ell)$, $M_\ell^y = \boldsymbol{\mu}_\ell^y + \Sigma_\ell^{y\boldsymbol{\nu}_0} \left(\Sigma_\ell^{\boldsymbol{\nu}_0\boldsymbol{\nu}_0}\right)^{-1} (\boldsymbol{\nu}_0 - \boldsymbol{\mu}_\ell^{\boldsymbol{\nu}_0})$, and $S_\ell^y = \Sigma_\ell^{yy} - \Sigma_\ell^{y\boldsymbol{\nu}_0} \left(\Sigma_\ell^{\boldsymbol{\nu}_0\boldsymbol{\nu}_0}\right)^{-1} \Sigma_\ell^{\boldsymbol{\nu}_0 y}$. Inference for $f(y \mid t_0; G_N)$ and $f(y \mid x_0; G_N)$ proceeds in a similar fashion.

3.3.1.2 Prior Specification

To specify the parameters of the priors of the DP mixture model, we assume vague knowledge of the center and range of the simulator data, which in many applications may be reasonable to expect based on the physical interpretation of the control input and output variables. Let (c_y, c_t, c_x) be rough prior guesses at the center of the output and inputs, and let r_y, r_t, and r_x be the corresponding ranges, and denote by R the diagonal matrix with diagonal elements $(r_y/4)^2$, $(r_t/4)^2$, and $(r_x/4)^2$, which are taken to be prior guesses for the variability of the data.

Now, consider the single component mixture $N_3(\cdot; \boldsymbol{\mu}, \Sigma)$, which is the limiting case of the DP mixture model with $\alpha \to 0^+$. Using the laws of total expectation and total variance, the marginal prior mean and covariance of the data under this model are $\boldsymbol{a_m}$ and $(\nu - d - 1)^{-1} a_S B_S + (a_V - d - 1)^{-1} B_V + B_{\boldsymbol{m}}$, respectively, where d is the dimension of the vector of output and inputs (which is 3 if we only consider one output, one control input, and one calibration input). Then,

for default priors, we set $\boldsymbol{a_m} = (c_y, c_t, c_x)$, multiply R by an inflation factor, say 3, and split it evenly between the three components of the marginal variance, i.e., set $(\nu - d - 1)^{-1} a_S B_S = R$, $(a_V - d - 1)^{-1} B_V = R$, and $B_{\boldsymbol{m}} = R$. Thus, we need to specify values for the scaling parameters ν, a_S, and a_V, in order to solve for B_s and B_V. Note that smaller values of $(\nu - d - 1)^{-1} a_S$ result in a more dispersed prior for S, and that $d+2$ is the smallest integer value of a_V that results in finite expectation but yields the largest possible dispersion for the prior of V. Taking these two facts into consideration, we set $\nu = a_V = a_S = 2 \times (d+2)$.

The prior choice for the DP precision parameter, α, is determined by the role it plays in the DP mixture model. Recall that for the $DP(\alpha, G_0)$ prior, α controls how close a realization G is to G_0. In the DP mixture model, α controls the distribution of the number of distinct components, m^*, of the vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_m)$ in (3.8), and therefore the number of distinct components of the mixture (Antoniak, 1974; Escobar and West, 1995; Liu, 1996). In particular, for moderately large m, $\mathrm{E}(m^* \mid \alpha) \approx \alpha \log((\alpha + m)/\alpha)$, which can be further averaged over the prior for α to obtain a prior estimate for $\mathrm{E}(m^*)$. In practice, larger values of α yield higher prior probabilities for larger m^*.

3.3.2 An approach for inference for calibration

We will consider an approximate Bayesian inference approach to calibration, in the spirit of the modular approach suggested by Bayarri et al. (2007a). In stage 1, using only the simulator data $D^* = \{(y_j, t_j^*, x_j^*), j = 1 \ldots, m\}$, we obtain posterior samples for the parameters of the DP mixture emulator. The details for this stage are found in the emulation step (Section 3.3.1). In stage 2, the field data is modeled as $z_i \overset{iid}{\sim} N(z_i; h(x_i, \theta, G_N), \sigma^2)$, where $h(x_i, \theta, G_N) =$

$E(Y \mid x_i, \theta; G_N)$ is the conditional expectation for y under the DP mixture model emulator, given by (3.10) under the multivariate normal kernel. Then, priors are specified for θ and σ^2, and their posterior distribution is approximated with an MCMC procedure using posterior samples of the emulation parameters obtained in stage 1.

Using a normal distribution to model z_i makes sense for continuous output (perhaps after transformation) and is consistent with the normal kernel used for the emulation model; but it could be replaced with a different distribution, e.g., for discrete outputs. More importantly, this is a pragmatic approach for problems with limited field data (as in the application considered in Chapter 4), hence, the simplifying assumptions: parametric distribution connecting only its mean with the emulator, which is based on the more flexible BNP model. For applications with larger field samples, a fully nonparametric model is discussed in Section 3.5.

Given our modeling assumptions, if we consider the joint model for both the simulator data and the field observed data, the posterior $p(\theta, \sigma^2, \boldsymbol{\psi} \mid D, D^*)$is proportional to

$$\prod_{i=1}^{n} N(z_i; h(x_i, \theta, G_N), \sigma^2) \times p(\theta) \times p(\sigma^2) \times \prod_{j=1}^{m} \sum_{\ell=1}^{N} p_\ell k(y_j; t_j^*, x_j^*; \boldsymbol{\mu}_\ell, \Sigma_\ell) \times p(\boldsymbol{\psi})$$

where $\boldsymbol{\psi}$ is the vector of the parameters in the emulation model, that is, $\boldsymbol{\psi}$ includes $(\boldsymbol{p}_\ell, \boldsymbol{\mu}_\ell, \Sigma_\ell)$, α, and the parameters of G_0. Then, the marginal posterior distribution for θ and σ^2 is given by $p(\theta, \sigma^2 \mid D, D^*) = \int p(\theta, \sigma^2 \mid \boldsymbol{\psi}, D) p(\boldsymbol{\psi} \mid D, D^*)\, d\boldsymbol{\psi}$. Under the modular approach, we approximate $p(\boldsymbol{\psi} \mid D, D^*)$ with $p(\boldsymbol{\psi} \mid D^*)$. Thus, MCMC samples from the (approximate) posterior of θ and σ^2 are obtained through $\int_{\boldsymbol{\psi}} \left\{ \prod_{i=1}^{n} N(z_i; h(x_i, \theta, G_N), \sigma^2) \right\} p(\sigma^2) p(\theta) p(\boldsymbol{\psi} \mid D^*)\, d\boldsymbol{\psi}$. One of the advantages of splitting the analysis into two stages is being able to use the block Gibbs

sampler to obtain posterior inference for the DP mixture parameters.

Working with an inverse-gamma prior for σ^2 with parameters a_{σ^2}, b_{σ^2}, and an appropriate prior for θ based on expert knowledge, the approximate posterior $p(\theta, \sigma^2 \mid D, D^*)$ is obtained through $\int_{\boldsymbol{\psi}} \left\{\prod_{i=1}^{n} N(z_i; h(x_i, \theta, G_N), \sigma^2)\right\} IG(\sigma^2; a_{\sigma^2}, b_{\sigma^2})\, p(\theta) p(\boldsymbol{\psi} \mid D^*)\, d\boldsymbol{\psi}$. Thus, the approximate posterior full conditional distribution for σ^2 is proportional to

$$\exp\left(-\frac{b_{\sigma^2}}{\sigma^2}\right) \exp\left(-\frac{1}{2\sigma^2}\Sigma_{i=1}^{n}\left(z_i - h(x_i, \theta, G_N)\right)^2\right) \times \left(\frac{1}{\sigma^2}\right)^{\frac{n}{2}+a_{\sigma^2}+1},$$

which yields the inverse-gamma distribution, $IG\left(\frac{n}{2} + a_{\sigma^2}, \{\Sigma_{i=1}^{n}\left(z_i - h(x_i, \theta, G_N)\right)^2\}/2 + b_{\sigma^2}\right)$, whereas for θ, it is proportional to $\exp\left(-\Sigma_{i=1}^{n}\left(z_i - h(x_i, \theta, G_N)\right)^2/(2\sigma^2)\right) \times p(\theta)$. Thus, we need a Gibbs step for σ^2, and M-H step for θ. The MCMC sampling would proceed according to the following algorithm:

1. Start at the current MCMC states $\sigma^{2^{(b)}}$ and $\theta^{(b)}$, having obtained sample $\boldsymbol{\psi}^{(b+1)}$ from the DP mixture emulation stage.

2. Draw $\sigma^{2^{(b+1)}}$ conditional on $\theta^{(b)}$ and $\boldsymbol{\psi}^{(b+1)}$ from
   $IG\left(n/2 + a_{\sigma^2}, \{\Sigma_{i=1}^{n}\left(z_i - h(x_i, \theta^{(b)}, \boldsymbol{\psi}^{(b+1)})\right)^2\}/2 + b_{\sigma^2}\right)$.

3. Draw $\theta^{(*)}$ from the proposal $N(\theta^{(*)}; \theta^{(b)}, b^2_{\theta^{(*)}})$, then conditional on $\sigma^{2^{(b+1)}}$ and $\boldsymbol{\psi}^{(b+1)}$, the M-H ratio on the log scale, $\log(\alpha_{\text{M-H}})$, is given by

$$-\frac{1}{2\sigma^{2^{(b+1)}}}\left(\Sigma_{i=1}^{n}\left(z_i - h(x_i, \theta^{(*)}, \boldsymbol{\psi}^{(b+1)})\right)^2 - \Sigma_{i=1}^{n}\left(z_i - h(x_i, \theta^{(b)}, \boldsymbol{\psi}^{(b+1)})\right)^2\right)$$
$$+\log\left(p\left(\theta^{(*)}\right)\right) - \log\left(p\left(\theta^{(b)}\right)\right),$$

and $\theta^{(*)}$ is accepted with probability $\min(\alpha_{\text{M-H}}, 1)$.

4. Repeat steps $2-3$ as necessary.

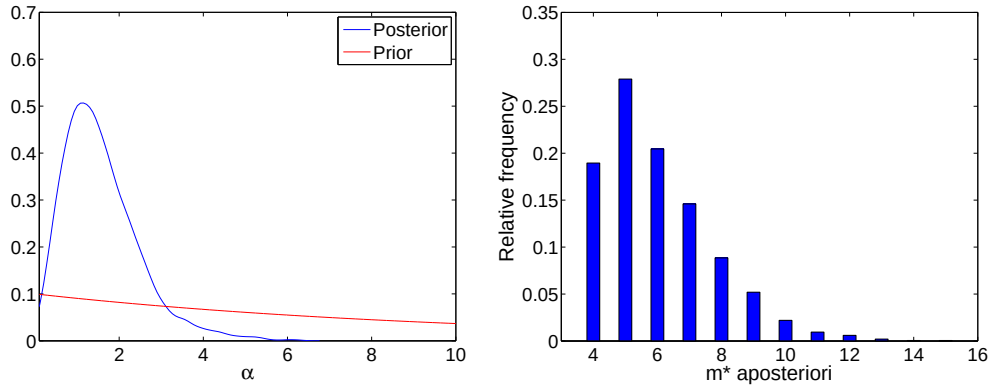
3.3.3 Illustrative toy example

To illustrate the methods discussed in Section 3.3, we work with the synthetic data example presented in Section 3.2.3, where $x^* \in (0,1)$, $t^* \in (0,1)$, and the synthetic simulator output, y, is generated from a normal distribution with mean equal to $2x^*t^* + \cos(t^*)$ and standard deviation 0.07. We use the same training data, where $m = 100$; see Figure 3.2.

For the emulation step, we set the maximum number of components to $N = 40$. Then, we specify the priors according to the guidelines discussed in Section 3.3.1.2. Specifically, we let $\nu = 6$, and specify the priors for $\boldsymbol{m}$, V, and S through the empirical mean and variance of the simulator data. Finally, we give α a gamma prior with a shape parameter $a_\alpha = 1$ and rate parameter $b_\alpha = 0.1$ (i.e., α is exponential with mean 10), so the mean apriori is 10, implying a prior belief of large number of underlying clusters (with average of 23 and effective range of 1 to about 36).

Figure 3.13 summarizes the posterior distribution for α and m^*, where the posterior density for α is concentrated about 1.57, and the average cluster size a posteriori is $m^* = 6$. Thus, there is significant learning in this case. Figure 3.14 shows a plot of the 95% posterior uncertainty surfaces, under the DP mixture emulator, for the mean of the simulator output distribution vs the mean surface of the simulator output. The emulator's estimated mean region captures the shape of the simulator mean reasonably well but misses some points in three corners of the gird. As far as output density estimation is concerned, Figure 3.15 shows that the emu-

lator captures the distribution of the output at the selected input configurations reasonably well beating the fit under the GP using the exponential correlation function and performing slightly better than the GP with the Gaussian correlation function for $(t_0, x_0) = (0.2, 0.2)$. Thus, the DP mixture emulator performs well even though this synthetic simulator example corresponds to a setting quite different from the DP mixture model.



(a) (b)

Figure 3.13: (a) Posterior inference for α. (b) The relative frequency aposteriori of m^*.

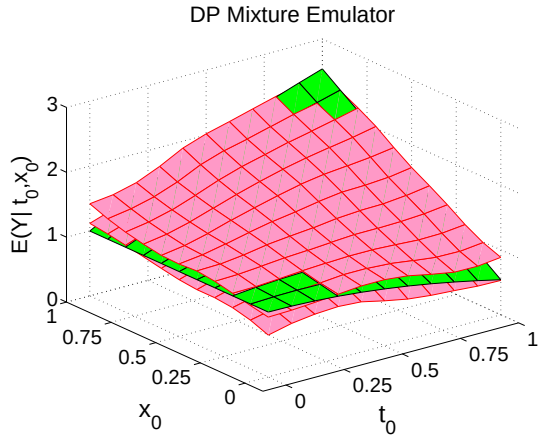


Figure 3.14: 95% posterior uncertainty surfaces given by the DP mixture emulator (red) for the mean surface of the simulator outputs (green).

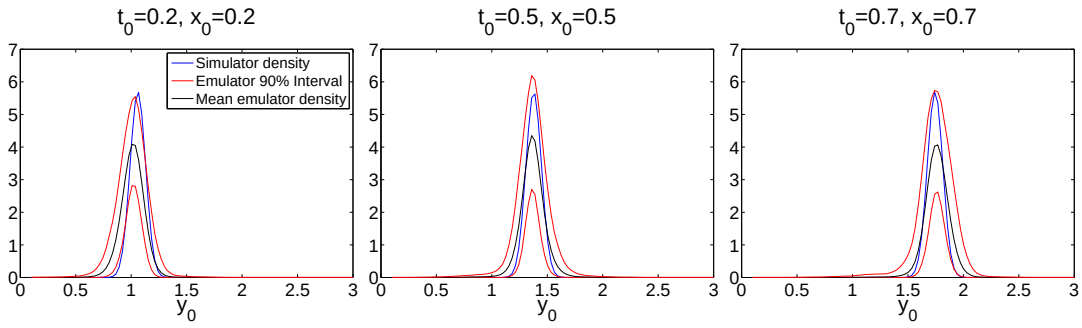
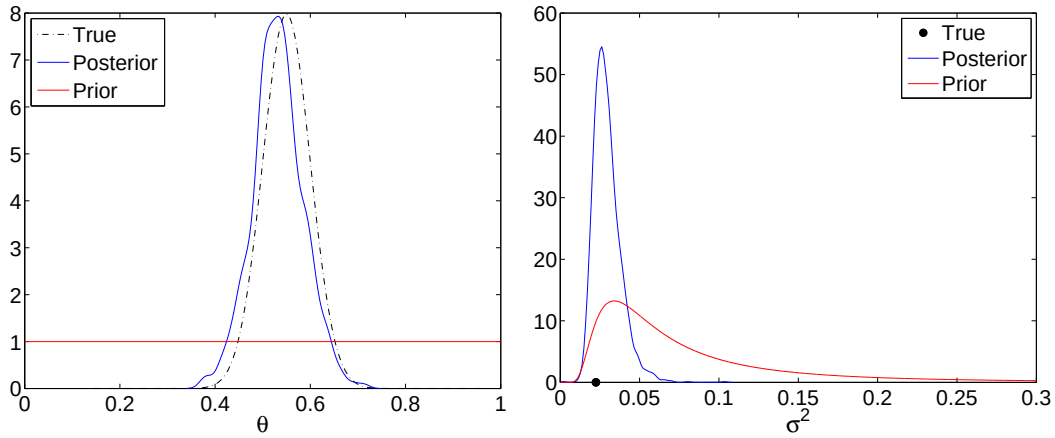


Figure 3.15: Posterior mean and 95% uncertainty intervals under the DP mixture emulator for the simulator output density at three input configurations.

For calibration, we use the same 30 observations that were used for calibration in Section 3.2.3; see Figure 3.8. The results presented here are based on the emulation step with 100 simulator training runs. Figure 3.16 shows that the two-stage procedure estimates the distribution of θ well. Also, the posterior of σ^2 covers the true value of observational error.



(a) (b)

Figure 3.16: Results using the two-stage calibration approach based on the DP mixture emulator using $m = 100$ simulator training runs and $n = 30$ field observations. (a) The true density plot vs the prior and posterior density plots for the calibration parameter θ. (b) The true value of σ^2 as well as its prior and posterior density plots.

3.4 A more complex example

While the synthetic example used in Sections 3.2.3 and 3.3.3 illustrates the two different emulation methods (based on the GP prior and the DP mixture) developed in this chapter, it does not correspond to a realistic scenario. For a stochastic simulator (and the corresponding real physical process), it would be more realistic for the calibration variable to interact with the output variable in a stochastic fashion, which is not the case in the earlier example. Thus, for this example we work with a stochastic simulator that takes into account the joint stochasticity of the process output and control input. As a test case, we assume that the output distribution of the actual underlying physical process arises (or can be well approximated) through a mixture structure for the joint distribution of the output and calibration variables.

Specifically, we assume that the controllable input x has a uniform distribution over a particular range. Then, conditional on x, the joint density of the output y and the calibration input θ is a mixture of three bivariate normal distributions (see Figure 3.18 (a)), i.e.,

$$f(y,\theta \mid x) = \sum_{k=1}^{3} q_k(x) N_2\left((y,\theta); \boldsymbol{m}_k = \begin{pmatrix} m_{yk} \\ m_{\theta k} \end{pmatrix}, \Sigma_k = \begin{pmatrix} \Sigma_k^{yy} & \Sigma_k^{y\theta} \\ \Sigma_k^{\theta y} & \Sigma_k^{\theta\theta} \end{pmatrix}\right)$$

with means $\boldsymbol{m}_1 = (4.13, 0.3), \boldsymbol{m}_2 = (4.15, 0.55)$, and $\boldsymbol{m}_3 = (4.3, 0.85)$; and covariances

$$\Sigma_1 = \begin{pmatrix} 0.0150 & -0.0050 \\ -0.0050 & 0.0025 \end{pmatrix}, \Sigma_2 = \begin{pmatrix} 0.001 & 0 \\ 0 & 0.001 \end{pmatrix}, \Sigma_2 = \begin{pmatrix} 0.010 & 0.005 \\ 0.005 & 0.010 \end{pmatrix}.$$

The weights $q_k(x), k = 1,2,3$, are generated according to the following rule: Let $x \in$

$(0,1)$, then $q_1(x) = 0.05x + 0.1$, $q_2(x) = 0.4x^2 + 0.05$, and $q_3(x) = 1 - q_1(x) - q_2(x)$. The conditional output density $f(y \mid \theta, x) = f(y, \theta \mid x)/f(\theta \mid x)$, where

$$f(\theta \mid x) = \int_y f(y, \theta \mid x) dy = \sum_{k=1}^{3} q_k(x) \int_y N_2\left((y, \theta); \boldsymbol{m}_k, \Sigma_k\right) dy \tag{3.13}$$

$$= \sum_{k=1}^{3} q_k(x) N\left(\theta; m_{\theta k}, \Sigma_k^{\theta\theta}\right). \tag{3.14}$$

Thus, the structure we assume for the joint density $f(y, \theta \mid x)$ specifies a distribution for θ through (3.14), which was not possible in the earlier example. Figure 3.17 shows the density plot for the implied distribution of θ. Later, when we specify a range of design values over the calibration parameter for the simulator run, we will assume vague knowledge of the "effective" range based on this "true" distribution.

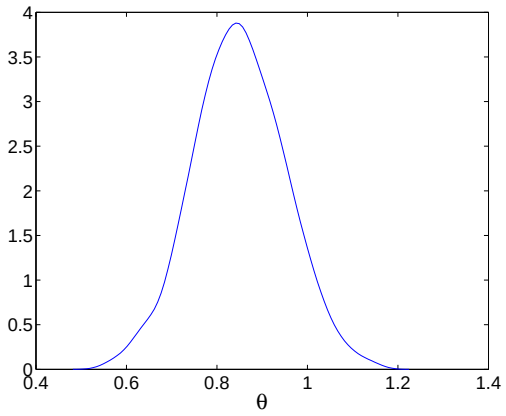


Figure 3.17: The density of the calibration parameter θ implied by (3.14).

Furthermore, $f(y \mid \theta, x) = \sum_{k=1}^{3} q_k(x) N_2\left((y, \theta); \boldsymbol{m}_k, \Sigma_k\right) / \left(\sum_{k=1}^{3} q_k(x) N\left(\theta; m_{\theta k}, \Sigma_k^{\theta\theta}\right)\right)$, and $N_2\left((y, \theta); \boldsymbol{m}_k, \Sigma_k\right) = N(y \mid \theta; a_k, S_k^2) N\left(\theta; m_{\theta k}, \Sigma_k^{\theta\theta}\right)$, where $a_k = m_{yk} + \Sigma_k^{y\theta} \left(\Sigma_k^{\theta\theta}\right)^{-1} (\theta -$

$m_{\theta k}$), and $S_k^2 = \Sigma_k^{yy} - \Sigma_k^{y\theta}\left(\Sigma_k^{\theta\theta}\right)^{-1}\Sigma_k^{\theta y}$. Thus, the final expression for $f(y\mid,\theta,x)$ becomes

$$f(y \mid \theta,x) = \sum_{k=1}^{3} \frac{q_k(x)N\left(\theta; m_{\theta k}, \Sigma_k^{\theta\theta}\right)}{\sum_{k=1}^{3} q_k(x)N\left(\theta; m_{\theta k}, \Sigma_k^{\theta\theta}\right)} N(y \mid \theta; a_k, S_k^2), \tag{3.15}$$

i.e., a mixture of three normal densities in y, with means that depend on θ, and with mixture weights that depend on both θ and x. Note that based on (3.15), the output mean surface can be written as

$$\mathrm{E}(y \mid \theta,x) = \sum_{k=1}^{3} \frac{q_k(x)N\left(\theta; m_{\theta k}, \Sigma_k^{\theta\theta}\right)}{\sum_{k=1}^{3} q_k(x)N\left(\theta; m_{\theta k}, \Sigma_k^{\theta\theta}\right)}\, a_k. \tag{3.16}$$

Assuming the simulator provides an accurate representation of reality, its output will be generated according to (3.15). Figure 3.18 shows a plot of the simulator's mean surface. Figure 3.19 shows a simulator run over appropriate ranges for the calibration and control variables. Because the underlying bivariate distributions have three different covariance matrices, we observe heterogeneous variance in the simulator output.

We generate two different sets of runs from the simulator using $m = 250$ and $m = 400$ according to (3.15) using a Latin hypercube design over the inputs for each set, with $x^* \in (0,1)$ and $t^* \in (0.3, 1.3)$, and we carry out analysis under the DP mixture emulator framework. In our implementation of this approach, we set the maximum number of clusters, N, to 40 for the DP mixture, and we specify the priors according to the guidelines discussed in section 3.3.1.2. Specifically, we let $\nu = 6$, and specify the priors for $\boldsymbol{m}$ and V, and S through the empirical mean and variance of the simulator data. Finally, we give α an exponential prior with mean 10. We

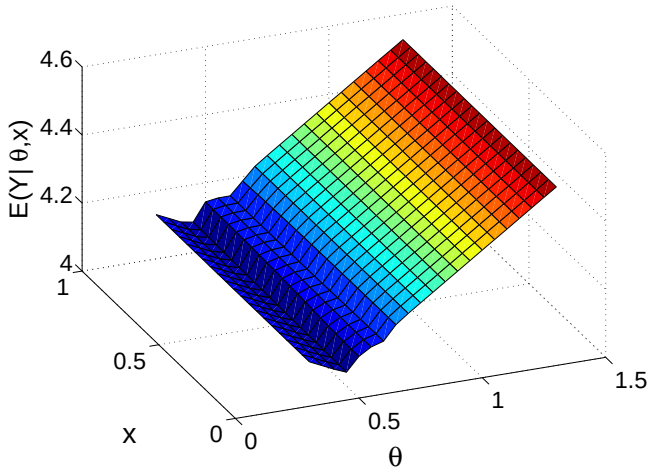
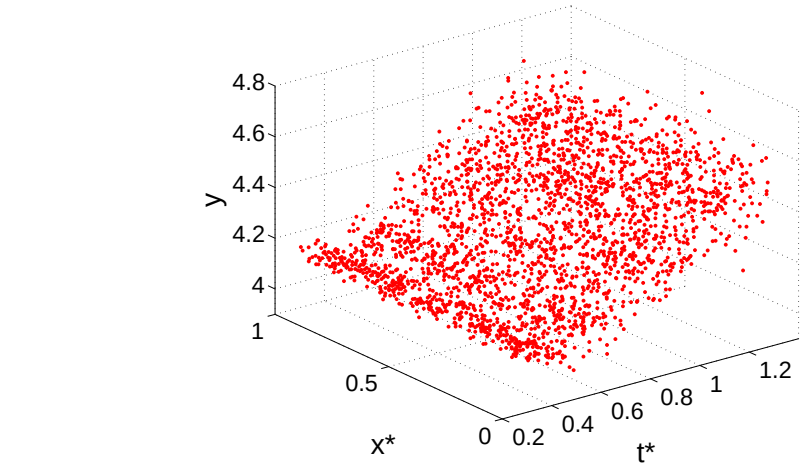


Figure 3.18: The stochastic simulator's mean surface, $\mathrm{E}(y \mid \theta, x)$, given in (3.16).

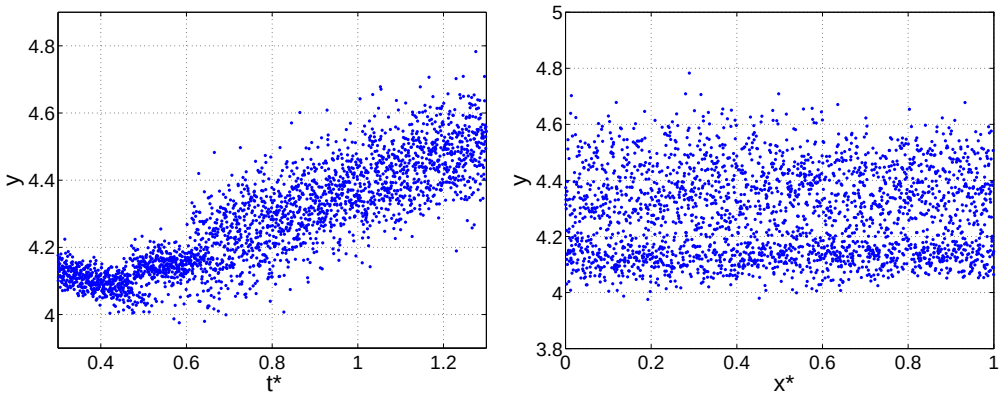
find that this prior choice, which was also used in Section 3.3.3, works well as a default prior for emulation.

Figure 3.20 shows 95% posterior uncertainty surfaces for the mean of the simulator output distribution over a grid of inputs, (t_0, x_0) using $m = 250$ and $m = 400$. The shape of the simulator mean is captured reasonably by the emulator mean in both cases. However, the region near $t_0 = 0.5$ lies outside the 95% uncertainty intervals for the emulator based on $m = 250$, but under $m = 400$, that region is within the 95% uncertainty intervals. In figure 3.21, we compare posterior inference for the output density at three input configurations to the true simulator output density. We find that the emulator provides a reasonable fit for the true simulator output density.

For the calibration step, $n = 15$ (synthetic) field observations are generated from a normal distribution with mean equal to the mean of the simulator, given in (3.16), and variance $\sigma^2 = 0.1^2$. First x is generated from a uniform distribution on $(0,1)$, and then θ is generated

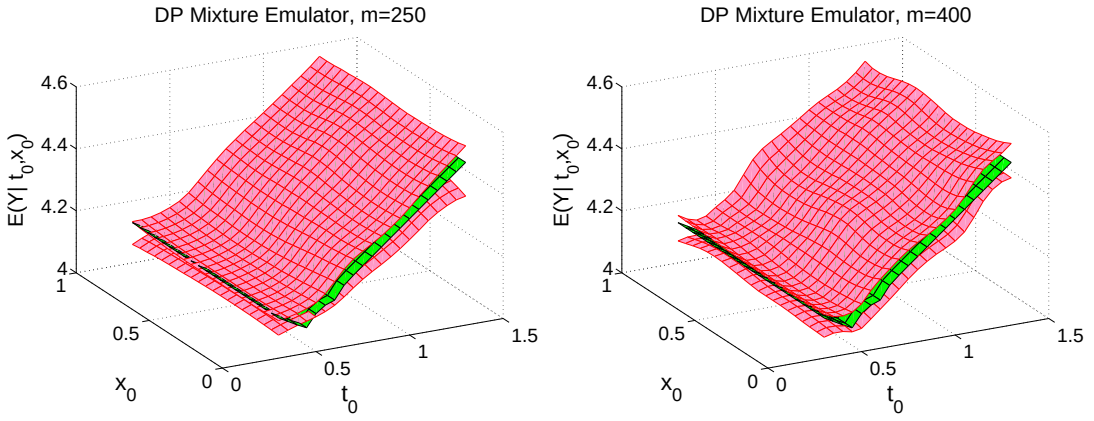


(a)



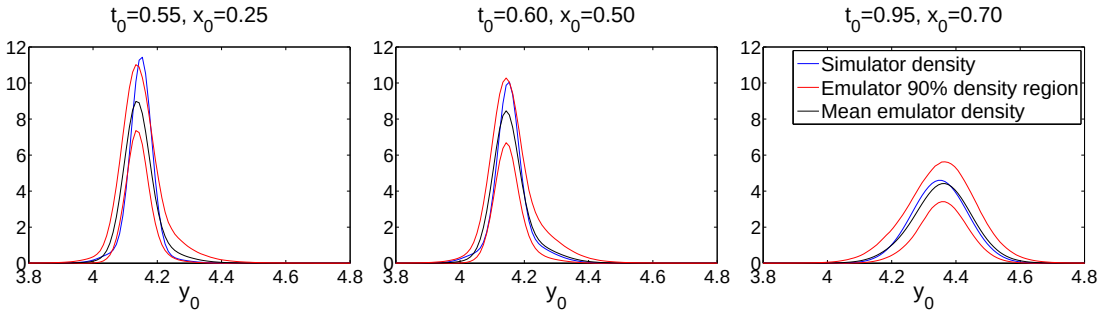
(b) (c)

Figure 3.19: (a) Outputs from a large simulator run generated according to (3.15) using a uniform 100×100 grid over $t^* \in (0.3, 1.3)$ and $x^* \in (0, 1)$. (b)-(c) The corresponding 2-dimensional plots showing the values of t^* and x plotted against the simulator outputs y.



(a) $m = 250$ (b) $m = 400$

Figure 3.20: Posterior inference under the DP mixture emulator using simulator runs of sizes $m = 250$, and $m = 400$. We plot the 95% posterior uncertainty surfaces given by the DP mixture emulator (red) for the mean of the simulator output distribution (green).

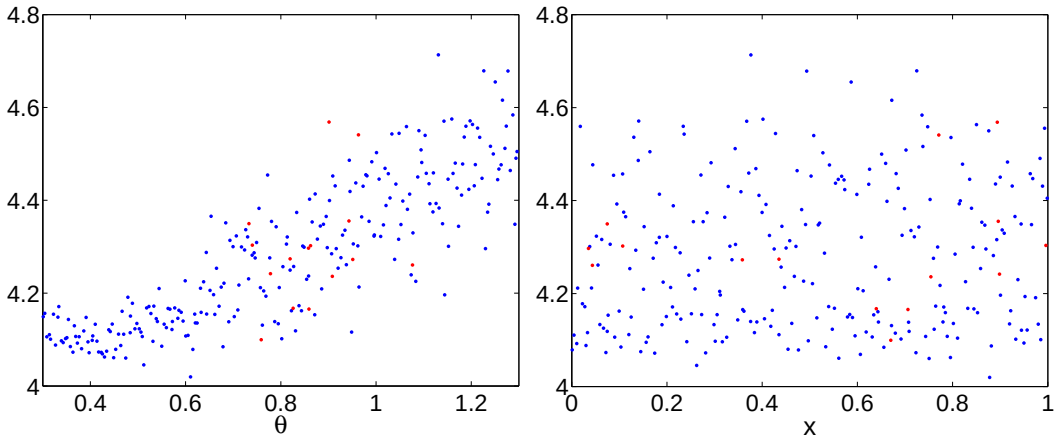


(a) (b) (c)

Figure 3.21: Posterior inference for densities under the DP mixture approach.

from (3.14). And finally the field observed output is generated from the normal specified above.

Figure 3.22 shows 2-dimensional plots of the simulator and field data. Here, we assume a uniform prior over the range of θ, as specified with the simulator. For σ^2, we base our prior information on some knowledge of the range of the field observed data, and we use an inverse-gamma centered on the observed data variance. The posterior distribution of θ estimates the true distribution of θ very well; see Figure 3.23. Also, the posterior of σ^2 reasonably estimates the value of the observational error. Thus, the two-stage calibration procedure works very well in this case.

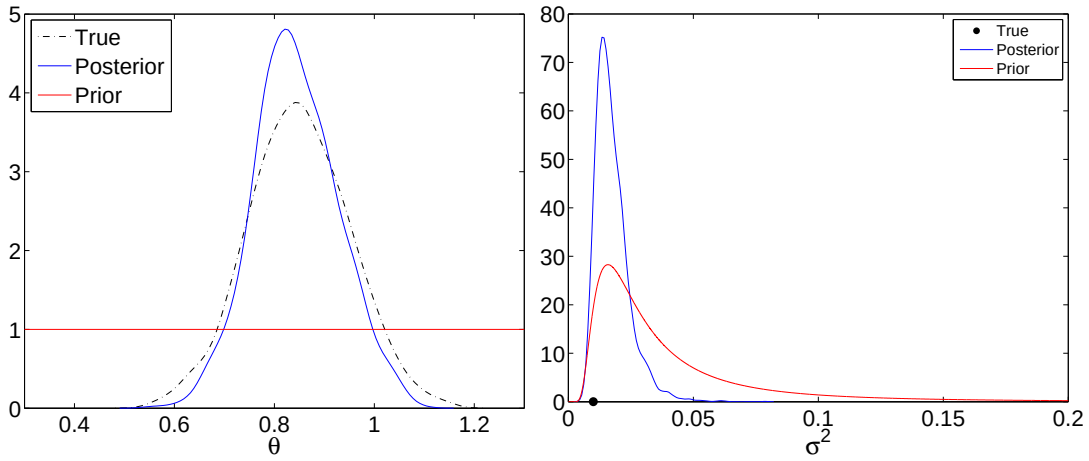


(a) (b)

Figure 3.22: Two dimensional plots of the synthetic simulator training data, $m = 250$, (blue); and field observations, $n = 15$, (red).

3.5 Conclusions and future work

We have introduced a new framework for emulation of stochastic computer simulators using DP mixtures. Unlike current emulation methods for stochastic simulators, this new mod-



(a) (b)

Figure 3.23: Inference using the two-stage calibration procedure with DP mixture emulation, where $m = 250$, and $n = 15$.

eling approach allows for modeling the entire distribution associated with the simulator, and not just the mean of the output process (or the first two moments). Additionally, this approach relaxes assumptions, such as normality and stationarity, allowing for greater flexibility. Furthermore, the DP mixture emulation is a more natural choice for a multivariate, discrete/continuous output from the stochastic simulator.

With regard to calibration, we found that the two-stage calibration procedure used in conjunction with the DP mixture emulator works pretty well in capturing the true distribution of the calibration parameter. In future calibration analysis, a bias function will be introduced to account for discrepancy between the simulator and the process it is modeling. One way to model the bias function is letting it be a component of the mean of the normal distribution used for the observed experimental data.

A natural extension of the DP mixture approach is to have a fully nonparametric

model for the field observed data. Here, we will need moderate to large sample sizes for the field data. Hence, different applications (e.g., environmental applications) will be needed to illustrate the methodology instead of applications where field data is scarce. For simplicity of notation, assume that the process under consideration has a univariate response y and a univariate calibration input variable θ. A control input may be included as well, but, for now, we proceed assuming a single input. The simulator modeling this process will produce runs (y_j^*, t_j^*), $j = 1, \ldots, m$, where t_j^* is the design value of the (calibration) input, and y_j^* is the simulator output associated with it. Then, using (3.5) and a normal kernel, the statistical model given by the emulator for the joint density of the simulator data becomes

$$f^s(y^*, t^*; G^s) = \int N_2(y^*, t^*; \boldsymbol{\mu}, \Sigma) dG^s(\boldsymbol{\mu}, \Sigma), \tag{3.17}$$

where the superscript s indicates that this mixture model is for the simulator data.

Observed field data of the underlying process are z_i, $i = 1, \ldots, n$. We treat those observations as noisy measurements of the underlying process, and let $z_i = y_i + \varepsilon_i$, where y_i is the process response, and $\varepsilon_i \sim N(0, \sigma^2)$. Then, we model the joint density of the process response and unknown input as

$$f^p(y, \theta; G^p) = \int N_2(y, \theta; \boldsymbol{\mu}, \Sigma) dG^p(\boldsymbol{\mu}, \Sigma), \tag{3.18}$$

where the superscript p indicates that this model is for the process data. Thus, given the properties of the bivariate normal distribution, the marginal density for the process response is given

by $f^p(y;G^p) = \int N(y,;\boldsymbol{\mu}^y,\Sigma^{yy})dG^p(\boldsymbol{\mu},\Sigma)$. Assuming the process response is independent of the error term, we obtain the following mixture model for the field observations,

$$f^F(z;G^p,\sigma^2) = \int N(z;\mu^y,\Sigma^{yy}+\sigma^2)dG^p(\boldsymbol{\mu},\Sigma), \tag{3.19}$$

where the superscript F indicates that this is the implied model for the field observed data.

Thus, we have constructed three flexible models for the simulator, process, and field data. A natural way to link these models is by requiring a dependent prior model for the random mixing distributions G^s and G^p. Assuming DP priors for both, their dependence may be built through the weights of their stick-breaking representation assuming a common centering distribution $G_0(\boldsymbol{\mu},\Sigma)$. If we are only interested in emulation, then an approximation of the stochastic simulator output density can be obtained through $f^s(y^* \mid t^*;G^s)$. Moreover, inference for calibration will arise through the marginal density $f^p(\theta;G^p)$ given the field observed data as well as the simulator data. Here, prior information for θ may be incorporated in $f^p(\theta;G^p)$, say, through its prior expectation. Furthermore, validation can be carried out by comparing the densities $f^s(\cdot \mid t^*;G^s)$ and $f^p(\cdot \mid \theta;G^p)$. The modeling will get more complex with more inputs, but this approach holds promise for problems with moderate to large sample sizes for the field data, and will thus be explored in future research.

Chapter 4

Study of radiation effects in spaceborne microelectronics by combining data from lab experiments and a stochastic simulator

4.1 Motivation

The work in this chapter is concerned with the vulnerability of spaceborne microelectronics to single event upsets (SEUs) or "soft errors". SEUs are non-destructive errors in microelectronic devices that result from free charge created by ionization near a sensitive node due to a high-energy charged particle, in the solar wind or the cosmic ray environment, striking the device. SEUs normally appear as transient pulses in logic or support circuitry, or as bitflips in memory cells or registers (Petersen et al., 1992). In the space environment, there are two main causes of SEUs: heavy ions and high energy protons. Heavy ions cause "direct" ionization SEUs, where if the ion particle traveling through the device deposits sufficient charge, a

SEU may occur. Protons typically cause “indirect” ionization SEUs, where a proton causes a nuclear reaction near a sensitive node producing nuclear fragments or recoil particles (i.e., heavy ions); those particles then create an ionization effect potentially causing a SEU.

Characterizing the vulnerability of a spaceborne electronic device to SEUs is critical for the success of space missions. To measure the susceptibility of a semiconductor device to single event upsets, ground testing is conducted by exposing it to high-energy heavy ions or protons produced in a particle accelerator. More recently, stochastic computer simulators, which typically use Monte Carlo techniques, are providing tools to determine the radiation effects on microelectroics in space. Typically, the fundamental quantity of interest is the upset rate of the device for the space radiation environment specified by a particular orbit.

It is known that the upset rate depends on the linear energy transfer (LET) of the incident particles, the cross-section of interaction, and the fluence (Petersen et al., 1992). LET is a measure of the energy transferred to a device per unit length when an ionizing particle passes through it, and the common unit for LET is $MeV \cdot cm^2/mg$; fluence, given in $particles/cm^2$, is the number of particles that intersect a unit area; cross-section is a measure of the device upset response to ionizing radiation, and it is calculated as the number of errors divided by fluence. The interaction cross-section is assumed to be monotonically increasing with LET (Xapsos et al., 1993; Petersen, 1996). For proton induced SEUs, the cross-section depends on the incident proton energy in MeV.

Heavy ion ground testing has a very involved protocol (Schwank et al., 2008), and in 2008 it cost $1000 per hour (Petersen, 2008). Of interest to the device testing community is studying whether a device’s cross-section vs. LET curve obtained from proton testing gives a

good estimate of the cross-section vs. LET curve obtained from heavy ion testing. That is, if we have proton test data, can we learn the cross-section vs. LET curve parameters for heavy ion upsets, avoiding the additional expense of heavy ion testing? This question motivates part of the methodology of this chapter.

To answer this question, we must consider the data coming from both the heavy ion and proton tests. The observed data from a heavy ion test consist of the upset count, the fluence, and LET. However, from the proton test, we observe the proton energy in addition to the upset count and the fluence. Thus, proton testing cannot be used on its own to estimate the parameters of the cross-section vs. LET curve. However, using a simulator that models proton testing and takes in the parameters of the cross-section vs. LET curve as inputs, we can estimate those parameters by calibrating the simulator using both proton test data and simulator data.

Here, we consider PROPSET, a stochastic simulator that calculates the frequency of proton induced SEUs in a spaceborne microelectronics device (Foster et al., 2006). Because the parameters of the cross-section vs. LET curve obtained through proton testing are an input to PROPSET, we can estimate that input through the calibration (or inversion) of PROPSET. However, PROPSET is computationally expensive, which motivates our work of building an emulator for it in order to carry out the analysis involved in model calibration. To do that, we rely on methods developed in Chapter 3 of this thesis.

The outline of this chapter is as follows. In Section 4.2, we model the cross-section vs. LET curve for the heavy ion testing using a Bayesian formulation for a standard parametric model utilized in the relevant literature. In Section 4.3, we discuss the PROPSET stochastic simulator and calibrate it for the cross-section vs. LET curve parameters. Section 4.3 includes

a discussion of the results of the approaches from the two earlier sections. In Section 4.4, we extend methods in Section 4.1 by developing a semiparametric model for the cross-section vs. LET curve and compare the new model with customary parametric approaches.

4.2 Modeling the cross-section vs. LET curve for heavy ion testing

Our focus is on modeling and predictive inference for the cross-section vs. LET curve, which is taken to be an increasing function of LET, ℓ, with a plateau related to the physical cross-section of the device. Upsets are caused when the charge deposited in the device's sensitive volume exceeds a critical amount. Charge deposition is proportional to energy deposition, so the cross-section vs. LET curve is monotonically increasing. In particular, the cross-section vs. LET curve can be generically written as $\sigma_0 G(\ell)$, where σ_0 is the limiting cross-section, and $G(\cdot)$ is a cumulative distribution function (cdf) on $\mathbb{R}^+$.

Standard practice is to assume $G(\cdot)$ is the cdf of a Weibull distribution, which may have a location parameter corresponding to the threshold LET (e.g., Petersen et al., 1992; Pickel, 1996; Swift et al., 2008). Other models (e.g., lognormal) can be used (Petersen, 1996), but here, we focus on the Weibull case with two parameters (shape and scale), so we assume that small values of LET may result in an upset (although the approach can be extended include the threshold LET as an unknown parameter of the model). The data, D, for a heavy ion test of a device are obtained in the form $D = \{(c_i, f_i, \ell_i) : i = 1, \ldots, n\}$, where c_i is the upset count, f_i is the fluence, and ℓ_i is the LET value. The observed cross-section is calculated as c_i / f_i, which is set equal to $\sigma_0 G(\ell_i) = \sigma_0 \left(1 - \exp\left(-(\ell_i / w)^s\right)\right)$. Then, σ_0 and the Weibull CDF parameters, (w, s), are, typi-

cally, estimated from D using (weighted) least squares, thus avoiding probabilistic modeling of the response distribution for the upset counts.

We take the approach of modeling the upset counts assuming a Poisson distribution $c_i \overset{ind}{\sim} \text{Poisson}(\mu_i)$, where $\mu_i = f_i \sigma_0 G(\ell_i)$, and $G(\cdot)$ is the Weibull CDF. Thus, in this model, the mean of the Poisson response distribution is specified by the cross-section vs. LET curve, $\sigma_0 G(\cdot)$, adjusted by the fluences, f_i. We complete the model with appropriate priors for σ_0, w, and s. Thus, the joint posterior is given by

$$p(\sigma_0, w, s \mid D) \propto \prod_{i=1}^{N} \left\{ \text{Poisson}\left(c_i; f_i \sigma_0 \left(1 - \exp\left(-(\ell_i/w)^s\right)\right)\right) \right\} \times p(\sigma_0) p(w) p(s) \tag{4.1}$$

We assume a dispersed gamma prior for σ_0, and two independent exponential priors for w, and s, which result in a gamma posterior full conditional for σ_0 (sampled with a Gibbs step) and a nonstandard posterior full conditional for (w,s) (sampled jointly with a M-H step). Although we do not do it here, informative priors may be used for all three parameters; in particular an informative prior for σ_0 chosen to be uniform up to a maximum value $\sigma_{\max}$. An absolute upper-bound for σ_0 is the physical size of the device. This can be reduced, in consultation with a device expert. The parameters of more informative exponential priors for w and s may be determined by assuming a relatively informative range of LET values (say, twice the observed range), which we use as an estimate of the difference between the 0.975 and 0.025 percentiles of $G(\dot{)}$, with the midrange value used as a rough estimate of the median of $G(\dot{)}$. This results in two equations with two unknowns, which are the elements of $\boldsymbol{\psi}$. The solutions are taken to be the means of the exponential priors for w and s.

LET ($MeV \cdot cm^2/mg$)	Fluence ($\times 10^3$)	Counts
1.7	3000, 10000, 30000	14, 66, 263
3.5	30000, 30000, 30000, 30000, 27600, 36000	1721, 1722, 1291, 1342, 1007, 1350
9.5	450, 300, 300, 300, 300, 300, 2000, 2000, 2000, 1750, 2000, 2000	161, 109, 114, 114, 118, 131, 696, 766, 632, 524, 690, 688
13.9	2000, 2000, 2000	800, 691, 748

Table 4.1: Heavy ion test data for the MC7447AT PowerPC. The counts are of the $0 \rightarrow 1$ upsets in the cache bits. Data courtesy JPL.

4.2.1 Results

The data used to illustrate the approach developed in Section 4.2 are obtained from a heavy ion test performed on a Freescale MC7447AT PowerPC device. The test data for this device are given in Table 4.1, where the counts correspond to the observed $0 \rightarrow 1$ upsets. Here, the counts are collected for the whole device, but the analysis is performed on a per-bit scale. The number of storage bits for this device is $253,184$. Thus, the observed cross-section per bit values are computed as $c/(253,184\, f)$. Note that we have repeated measurements, with fluences and counts (f_{ij}, c_{ij}), $i = 1,...,n_j$, corresponding to distinct LET value ℓ_j, $j = 1,...,P$, where $P = 4$ (with $n = \sum_{j=1}^{P} n_j = 24$). To accommodate for the repeated measurements, the posterior distribution given in (4.1) becomes

$$p(\sigma_0, w, s \mid D) \propto \prod_{j=1}^{P} \prod_{i=1}^{n_j} \left\{ \text{Poisson}\left(c_{ij}; f_{ij}\sigma_0\left(1 - \exp\left(-(\ell_i/w)^s\right)\right)\right) \times p(\sigma_0)p(w)p(s)$$

The priors for σ_0, w, and s are specified as discussed in Section 4.2. Then, MCMC is used to obtain posterior samples for σ_0 through a Gibbs step, and posterior samples from w,

s, through M-H steps. After that, posterior samples for the cross-section vs. LET curve are constructed as $\sigma_0\,(1-\exp\,(-(\ell_i/w)^s))$ over a sufficiently fine grid of LET values. Posterior inference for the cross-section vs. LET curve parameters is summarized in Figure 4.1. We see that the data has clearly defined the distributions of the parameters, and their uncertainties are quite small. Finally, based on samples from these distributions, posterior estimates of the cross-section vs. LET are constructed. Figure 4.2 summarizes posterior inference for the cross-section vs. LET curve. We find that the uncertainty about the curve increases with LET, and on average, the plateau is reached at about LET$=14$. Additionally, we note that posterior inference for the curve is not sensitive to prior choice for the curve parameters.

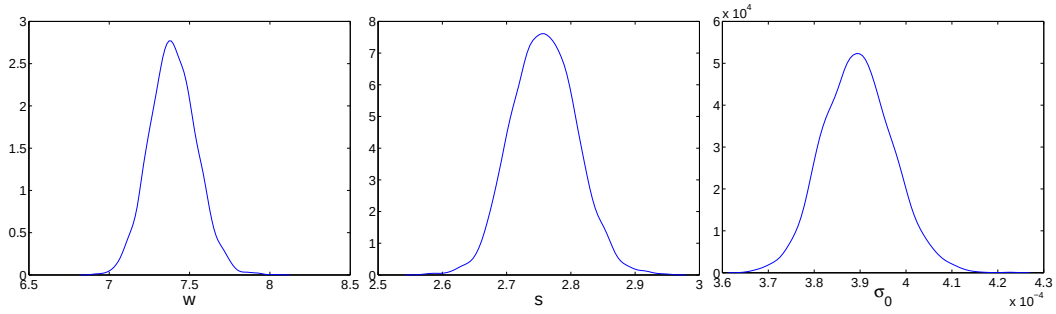


Figure 4.1: Posterior densities of the cross-section vs. LET curve parameters based on heavy ion testing.

4.3 Calibration of PROPSET for the parameters of the cross-section vs. LET curve

The Proton Upset Monte Carlo simulation (PROPSET) is a stochastic simulator that calculates the frequency of proton-induced SEUs in a spaceborne microelectronics device (Fos-

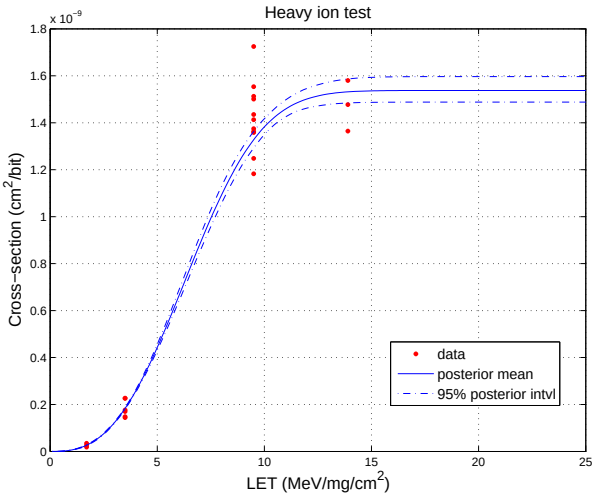


Figure 4.2: Point estimates and 95% probability intervals for the cross-section vs. LET curve based on data obtained from heavy ion testing performed on MC7447AT PowerPC. The observed cross-sections are in red.

ter et al., 2006). The code simulates the bombardment of a device with high-energy protons. The collision of each proton with the silicon of the chip is modeled, and the paths of the nuclear fragments (light fragments and heavy ions) produced by this collision are tracked. Then, if the energy (charge) deposited by the fragments into the "sensitive volume" of the chip exceeds a critical threshold, a SEU occurs. It takes about 20 minutes to execute a PROPSET run at a single proton energy on a laptop running at 700 MHz (Foster et al., 2006). Thus, PROPSET is computationally expensive which motivates our work on developing an emulator for it.

The stochasticity of PROPSET comes from its nuclear physics component, which is simulated using the FLUKA code (Ferrari et al., 2005; Ballarini et al., 2007). Specifically, FLUKA simulates the collision of protons with the individual nuclei of the silicon and tracks the trajectories and energies of the resulting light fragments and recoil nucleii (heavy ion) through the silicon slab. If a fragment or recoil particle passes through the sensitive volume, the energy

deposited by that particle is calculated, and added to the energy deposition from all other particles from the same nuclear event that passed through the sensitive volume. If the total energy deposition exceeds a threshold, a SEU is counted, and the tracking of that particle is terminated. The energy deposition threshold is determined by the thickness and area of the sensitive volume as well as the LET associated with it given the Weibull CDF parametrization (Foster et al., 2006).

Inputs to PROPSET include: the incident proton energy, e, in MeV, the thickness, th, of the sensitive volume of the device in μm, and the parameters of the generated heavy ion cross-section vs. linear energy transfer (LET) curve, namely σ_0, w, and s. σ_0 is the limiting cross-section given in cm^2, and w and s are dimensionless quantities. A location (or shift) parameter, representing threshold LET, for the Weibull CDF can also be specified, but in our work, we set that parameter to zero. In addition to specifying these inputs, one must specify the chip width, cw, in order to run PROPSET. Once the inputs are specified, the code is run with a large number (order of magnitude of 10^{10})of incident protons, then the number of resulting upsets is counted.

Since PROPSET is computationally expensive, calibration will be carried out using an emulator approximation. Calibration requires two sources of data: PROPSET simulator data, and field observations from an actual proton test. Proton test data are obtained in the form $D_p = \{(c_i, f_i, e_i) : i = 1, \ldots, n\}$, where c_i is the upset count, f_i is the fluence, and e_i is the proton energy value. The observed cross-section is calculated as c_i / f_i per bit. Thus, only one input is observed, which means we need to calibrate PROPSET for the other five. While we are able to obtain a moderate size run from PROPSET using a 100–core cluster, we are faced with a some

formidable challenges for calibration. First, homogeneity and stationarity of variances are not reasonable assumptions for the output of PROPSET. Second, there is little prior information about the five calibration inputs, so we may not be able to justify anything other than uniform priors. Third, proton testing experiments typically result in only a few observations.

Should we decide to perform the calibration of PROPSET using the GP prior approach, an appropriate covariance function must be specified. In this case, a stationary covariance might not be appropriate. Additionally, some computational issues must be considered. For example, the posteriors of the covariance function parameters (including difficult-to-tune range of dependence parameters) will have to be sampled using M-H steps, and inference under the smooth Gaussian covariance function may not be feasible. These issues lead us to take the more pragmatic approach of working with the modular DP mixture model for emulation, where the blocked Gibbs sampler is employed in the emulation stage, then the second stage calibration procedure based on it.

4.3.1 Emulation of PROPSET

Here, we work with the same device used in Section 4.2. In order to run PROPSET, we needed to specify a value for chip width; however, we only had information about a plausible range (based on expert opinion) for it, so we included chip width as an input in the analysis. We obtained a training set of 256 PROPSET runs based on a Latin Hypercube design over its six inputs $(w, s, \sigma_0, cw, th, e)$, where cw, th, and e denote, chip width, thickness, and proton energy, respectively. The ranges of these inputs are given in Table 4.2. For emulation, we follow the DP mixture approach discussed in Section 3.3. Because the cross-sections are either zero or close

Input	Minimum	Maximum
w	3	18
s	1	6
σ_0 (cm^2)	0.0001	0.001
cw (μm)	0.10	0.75
th (μm)	0.2	5
e (MeV)	6.5	200

Table 4.2: Ranges of inputs to PROPSET for MC7447AT PowerPC.

to zero, we work with a transformation $y = \log\left(\text{cross-section} + (9 \times 10^{-11})\right) + 19$.

For $m = 256$, let the design vectors over the calibration inputs be $\boldsymbol{w}^* = (w_1^*, \ldots, w_m^*)$, $\boldsymbol{s}^* = (s_1^*, \ldots, s_m^*)$, $\boldsymbol{\sigma_0}^* = (\sigma_{01}^*, \ldots, \sigma_{0m}^*)$, $\boldsymbol{cw}^* = (cw_1^*, \ldots, cw_m^*)$, and $\boldsymbol{th}^* = (th_1^*, \ldots, th_m^*)$; let the design vector over the control input be $\boldsymbol{e}^* = (e_1, \ldots, e_m^*)$; and let the transformed simulator output be $(\boldsymbol{y} = y_1, \ldots, y_m)$, corresponding to design matrix $(\boldsymbol{w}^*, \boldsymbol{s}^*, \boldsymbol{\sigma_0}^*, \boldsymbol{cw}^*, \boldsymbol{th}^*, \boldsymbol{e}^*)$. Thus, the PROPSET simulator data $D^* = \{(y_j, w_j^*, s_j, \sigma_{0j}^*, cw_j, th_j, e_j) : j = 1, \ldots, m\}$. We model the density of the PROPSET data using the DP mixture model

$$f(y_j, t_j^*, x_j^*; G) = \int N_7(y_j, w_j^*, s_j, \sigma_{0j}^*, cw_j, th_j, e_j; \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}}) dG(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}})$$

$$G|\alpha, G_0 \sim DP(\alpha, G_0)$$

Letting $\boldsymbol{u}_j^* = (y_j, w_j^*, s_j, \sigma_{0j}^*, cw_j, th_j, e_j)$, and working with the truncation approximation to the stick-breaking representation of G, we end up with the posterior model in (3.9). Then samples from the full conditional distributions of this posterior model are obtained according to the blocked Gibbs sampler, for which the details are given in Appendix B. The priors are

specified assuming vague knowledge of the center and range of the PROPSET data as discussed in Section 3.3.1.2. Specifically, we use the empirical means and variances of the data to specify the priors for the centering distribution, we let $\nu = 12$, and we give α a gamma prior with a shape parameter $a_\alpha = 1$ and rate parameter $b_\alpha = 0.1$, so the mean apriori is 10, implying a prior belief of moderate number of underlying clusters (with average of 23 and effective range of 1 to about 36). Finally, we set the maximum number of clusters to $N = 40$.

First, we examine posterior inference for the simulator's conditional means. Figure 4.3 includes plots of the point estimate and 95% posterior intervals of the simulator's conditional means given each input (marginalized over the others). The emulator provides a reasonable fit in every case. Note, the expressions to generate these plots are readily available (see (3.11) and (3.12)), but under the GP prior approach expressions for these posterior conditional means are difficult to obtain. We further investigate inference for the conditional mean of the simulator output by considering pairs of the calibration inputs. Figure 4.4 includes plots of emulator's posterior point estimates (mean surfaces) for $\mathrm{E}(y \mid w, s)$, $\mathrm{E}(y \mid w, \sigma_0)$, and $\mathrm{E}(y \mid s, \sigma_0)$. These plots do not reveal any new trends when we account for two inputs together.

Next, we examine posterior inference for the simulator densities. In particular, we plot point and interval estimates at nine different locations for energy, while fixing the other inputs at their mean values; see Figure 4.5. We observe bimodality with lower energy values, heavy left-tails with mid-range energy values, and apparent symmetry with high energy values. The DP mixture is flexible enough to allow for capturing different distributional shapes. These results are in agreement with the conditional expectation plot for energy in Figure 4.3.

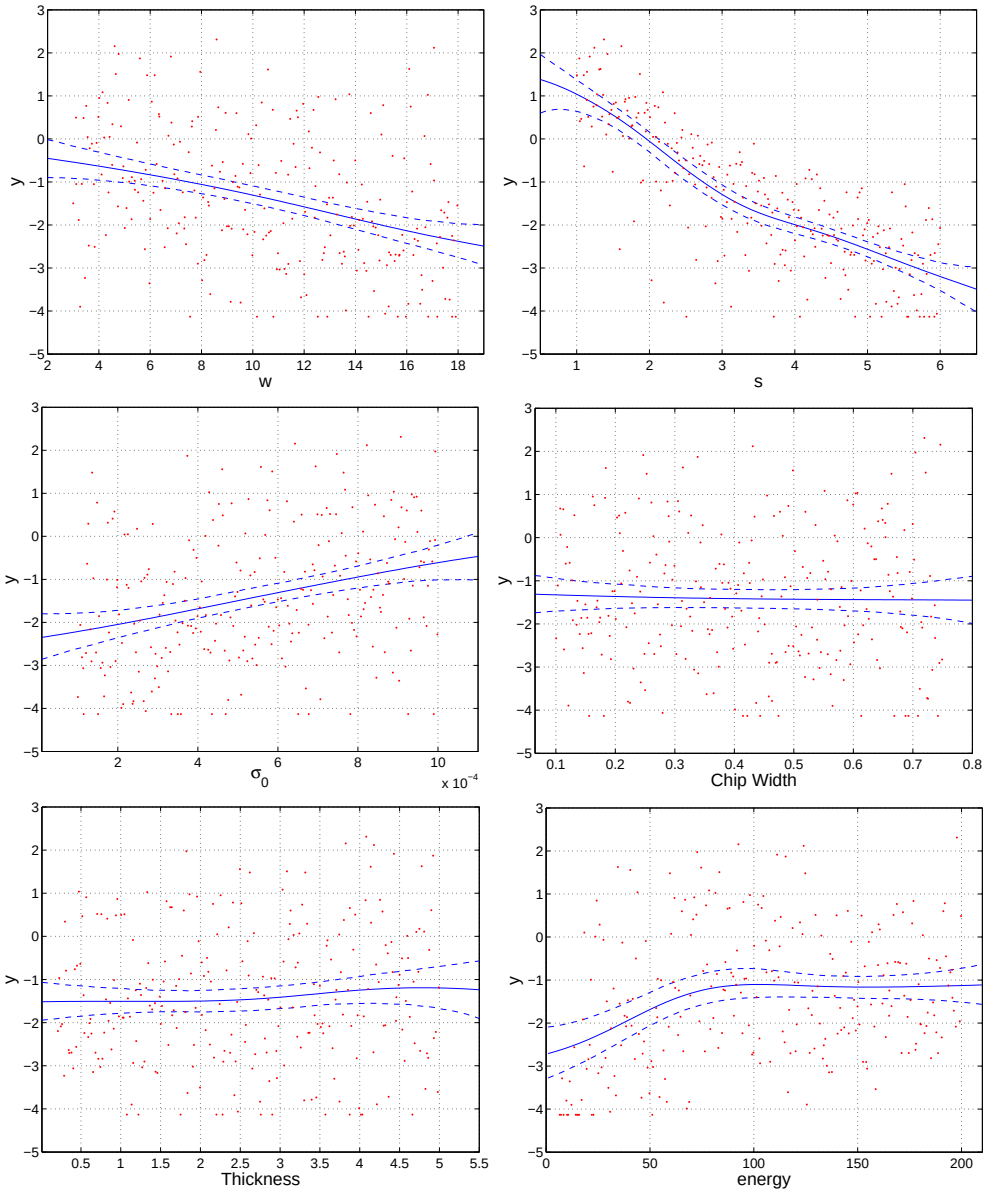


Figure 4.3: Data from the 256 PROPSET runs in red, where each input is plotted against the transformed cross-section. The blue solid line and dotted lines corresponds to the emulator's point estimate and 95% posterior intervals of $\mathrm{E}(y|input)$, respectively, for each of the six inputs.

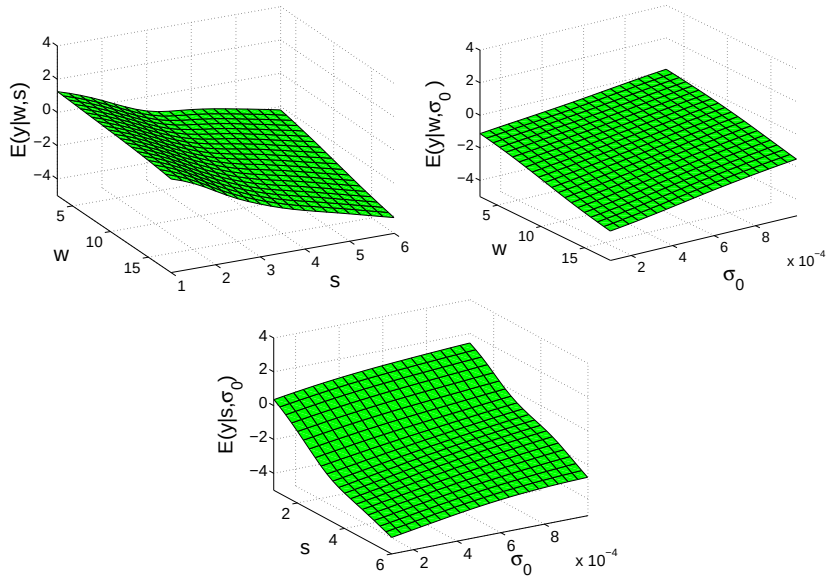


Figure 4.4: The emulator's posterior point estimates (mean surfaces) for $\mathrm{E}(y \mid w,s)$, $\mathrm{E}(y \mid w,\sigma_0)$, and $\mathrm{E}(y \mid s,\sigma_0)$.

4.3.2 Calibration of PROPSET

To calibrate PROPSET, we employ the two-stage approach discussed in Section 3.3.2. The observed data, D, for the calibration stage, given in Table 4.3, come from a proton test conducted on the same device used in Section 4.2.1. Here, proton energy is the only observed input, which leaves five inputs for calibration.

We begin by using only PROPSET data, D^*, to obtain posterior samples for the parameters of the DP mixture emulator. Then, letting $\boldsymbol{\theta} = (w, s, \sigma_0, cw, th)$ and taking the transformation $z = \log\left(\text{cross-section}_{\text{observed}} + (9 \times 10^{-11})\right) + 19$, we model the transformed observations as $z_i \overset{iid}{\sim} N(z_i; h(e_i, \boldsymbol{\theta}, G_N), \sigma^2)$, $i = 1, \ldots, 8$, where $h(e_i, \boldsymbol{\theta}, G_N) = E(Y \mid e_i, \boldsymbol{\theta}; G_N)$, which is the conditional expectation for y under the DP mixture model emulator, given by (3.10) under the multivariate normal kernel. Then, as discussed in Section 3.3.2, the approximate posterior

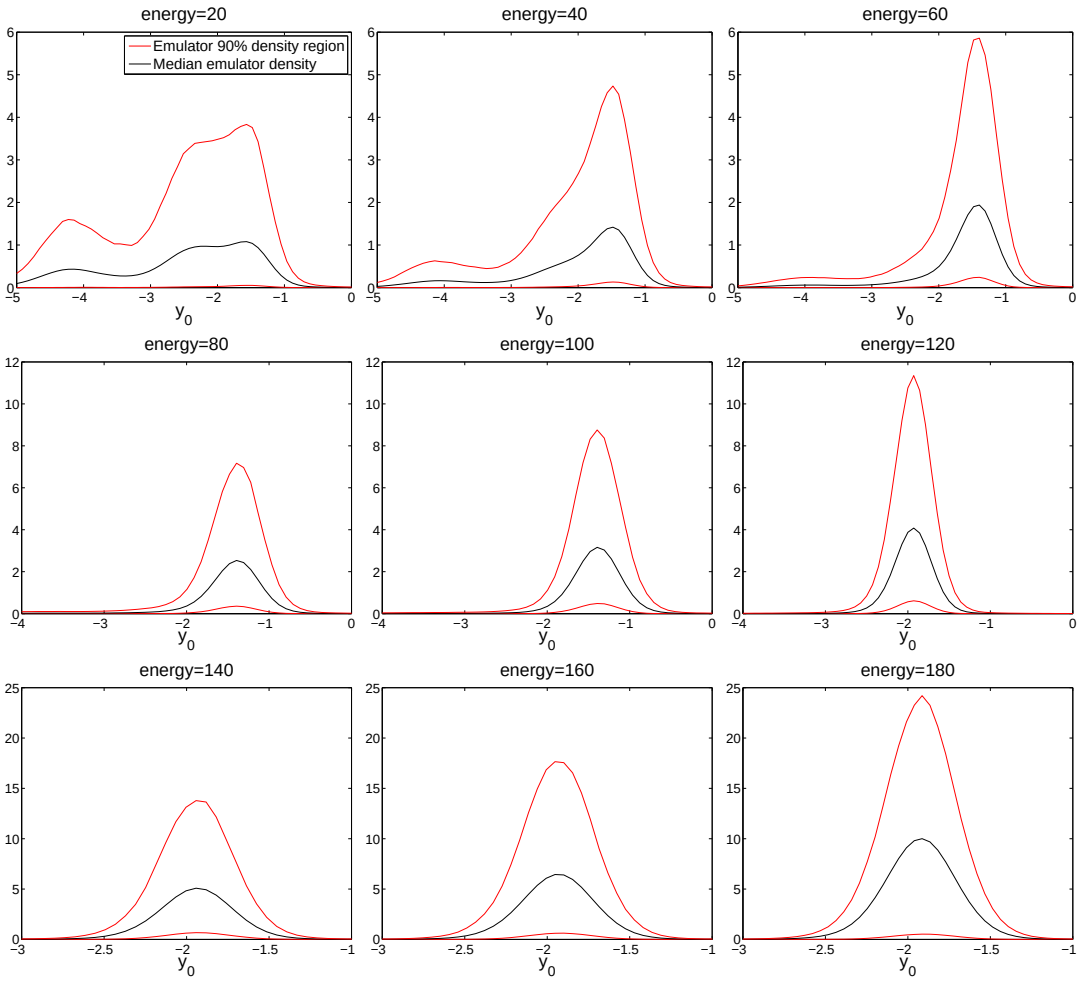


Figure 4.5: Posterior point and 90% interval estimates of the output densities at nine different locations for energy, with the other inputs fixed at their mean values.

Energy (MeV)	Fluence ($\times 10^3$)	Counts
52.3	10,300,000	22
89	49,900,000	80
199	61,500,000	111
51	50,000,000	110
27.5	49,900,000	104
15	47,600,000	92
10	50,000,000	65
6.6	95,000,000	74

Table 4.3: Proton test data for the MC7447AT PowerPC. The counts are of the $0 \rightarrow 1$ upsets in the cache bits. Data courtesy JPL.

$p(\boldsymbol{\theta}, \sigma^2 \mid D, D^*)$ is obtained through $\int_{\boldsymbol{\psi}} \left\{ \prod_{i=1}^{8} N(z_i; h(e_i, \boldsymbol{\theta}, G_N), \sigma^2) \right\} p(\sigma^2) p(\boldsymbol{\theta}) p(\boldsymbol{\psi} \mid D^*) \, d\boldsymbol{\psi}$, where $\boldsymbol{\psi}$ is the vector of the parameters in the emulation model.

We assume very little prior knowledge for all five calibration inputs and place independent uniform priors over their ranges, which were used in the emulation stage; see Table 4.2. Additionally, we give σ^2 the usual inverse-gamma prior with parameters a_{σ^2}, b_{σ^2} which are specified based on vague knowledge of the range of $\boldsymbol{z} = (z_1, \ldots, z_8)$. As shown in Section 3.3.2, we end up with an inverse-gamma posterior full conditional for σ^2, and a nonstandard posterior full conditional distribution for the vector $\boldsymbol{\theta}$. Thus, we sample these distributions using a Gibbs sampler for σ^2, and M-H sampler for $\boldsymbol{\theta}$, with $h(\cdot)$ being generated at every MCMC realization of the emulation stage parameters, $\boldsymbol{\psi}$.

Figure 4.6 summarizes posterior inference for the calibration parameters. We find that the calibration data resulted in some learning for *s*, but there is little learning for the rest of the parameters. This is not unexpected given that we only have eight field observations to work with, coupled with the fact that uninformative uniform priors are used. Finally, to

complete posterior inference for the calibration stage, posterior uncertainty is also quantified for the observational error σ^2; See Figure 4.7. Here, the prior was given a mean of $25 \times \text{Var}(z)$, and a large variance.

Based on the calibration results, we construct the cross-section curve, based on the scaled Weibull CDF (see Section 4.2), over a grid of LET values using samples from the posteriors of the calibration parameters. We find that the 95% uncertainty region is quite wide, which, perhaps, is not surprising since we only have eight proton test observations and uninformative uniform priors on the curve parameters. On average, the curve plateau's near LET$=23$ MeV/mg/cm^2, where the plateau value is approximately 2.4×10^{-9} cm^2/bit; see Figure 4.8.

Next, posterior inferences from PROPSET calibration and heavy ion test data (discussed in Section 4.2.1) are compared. We find that uncertainty is greater with the PROPSET calibration approach, but this can be partly explained by the fact that only eight proton test observations were used for the calibration, while 24 were used in the heavy ion test analysis. On average, the heavy-ion curve plateaus at about 1.5×10^{-9} cm^2/bit near LET $= 12$ MeV/mg/cm^2, while on average the proton curve plateaus at 2.4×10^{-9} cm^2/bit near LET$=23$ MeV/mg/cm^2. Even with these differences in the averages, results indicate promise, since with very limited proton test data, we obtain a point estimate reasonably close to the heavy-ion curve and uncertainty bands that include that curve.

In our investigation of the curve sensitivity to different prior specifications, we observed that σ_0 plays a significant role in controlling uncertainty. Recall, σ_0 is the limiting cross-section, so an absolute upper-bound for it is the physical size of the device, which can help guide the specification of the upper bound of the uniform prior for σ_0. In principle an

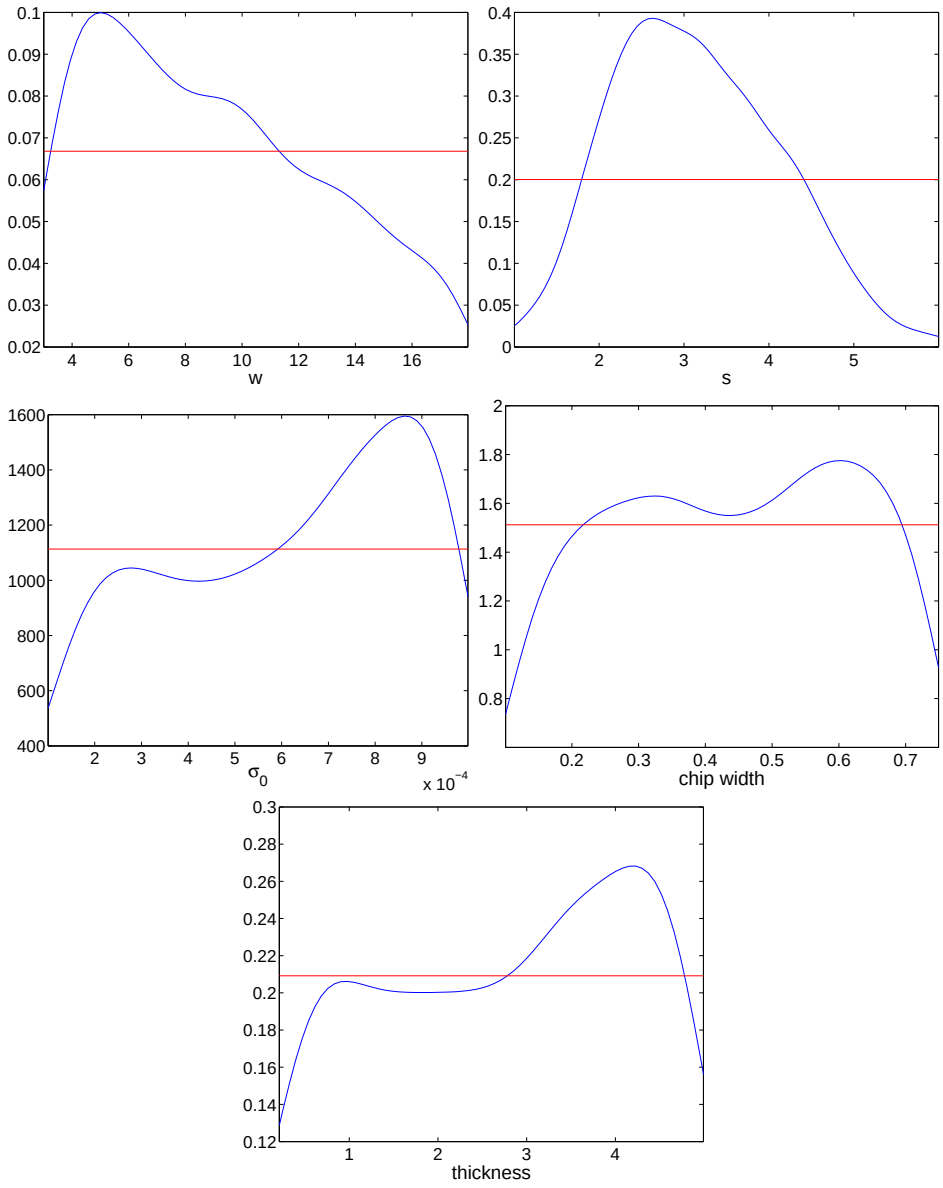


Figure 4.6: Posterior densities (blue) of the calibration parameters using uniform priors (red) over over their ranges, which were used in the emulation stage.

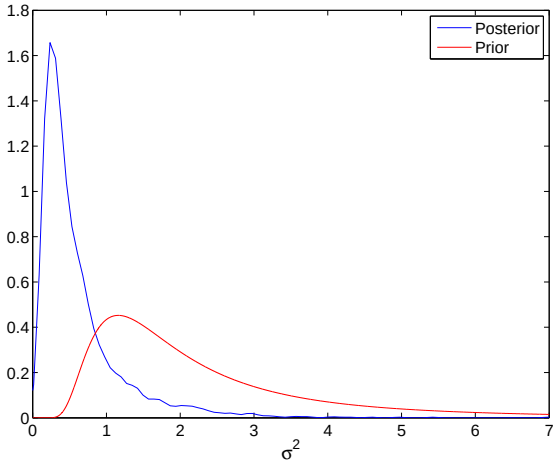


Figure 4.7: Posterior density of the observational error, σ^2 associated with the proton test data.

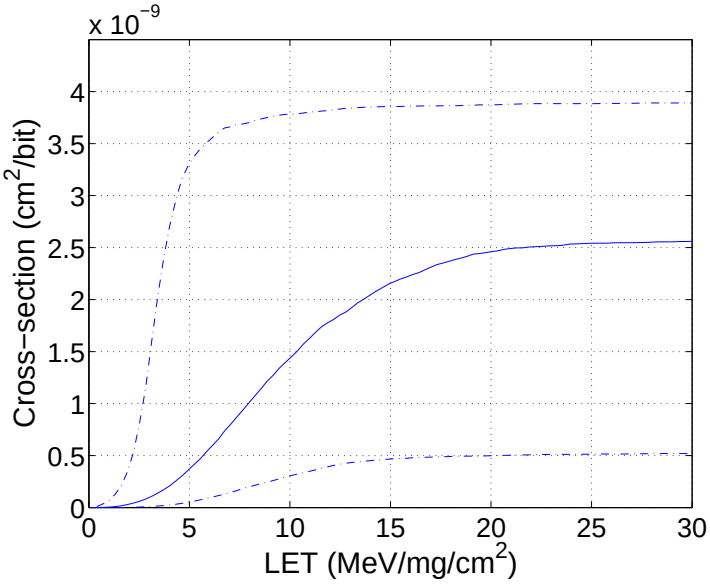


Figure 4.8: Posterior point estimates (solid blue) and 95% intervals (dotted blue) for the cross-section vs LET curve based on calibration of PROPSET, using uniform priors.

upper bound can be determined in consultation with a device expert. If we change the original Unif$(0.0001, 0.001)$ prior for σ_0 to Unif$(0.0001, 0.0007)$, keeping all the uniform priors for the other parameters unchanged, uncertainty about the proton cross-section vs. LET curve decreases and the expectation becomes closer to that for the heavy-ion curve expectation; see Figure 4.10 (a). If we specify an even more informative uniform prior based on a range close to the heavy-ion posterior results for σ_0, in particular, using a Unif$(0.00035, 0.00043)$ prior, uncertainty about the curve decreases considerably, and the plateau values for the heavy ion and proton tests agree; see Figure 4.10 (b).

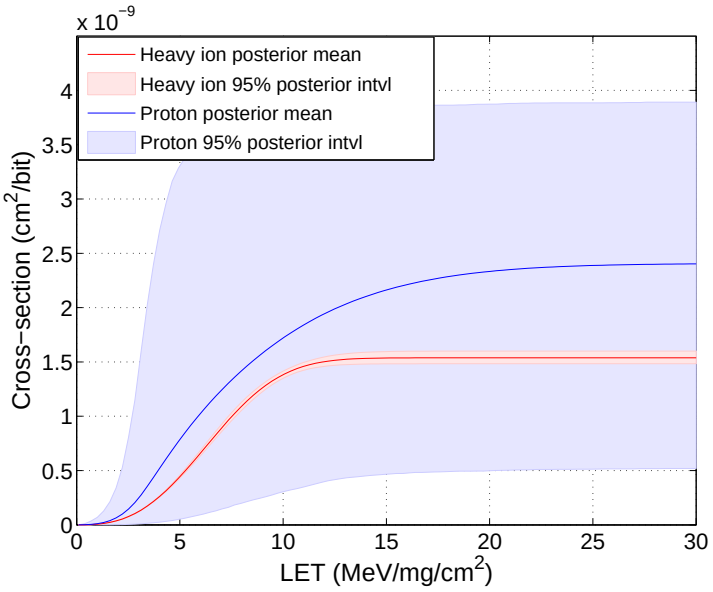
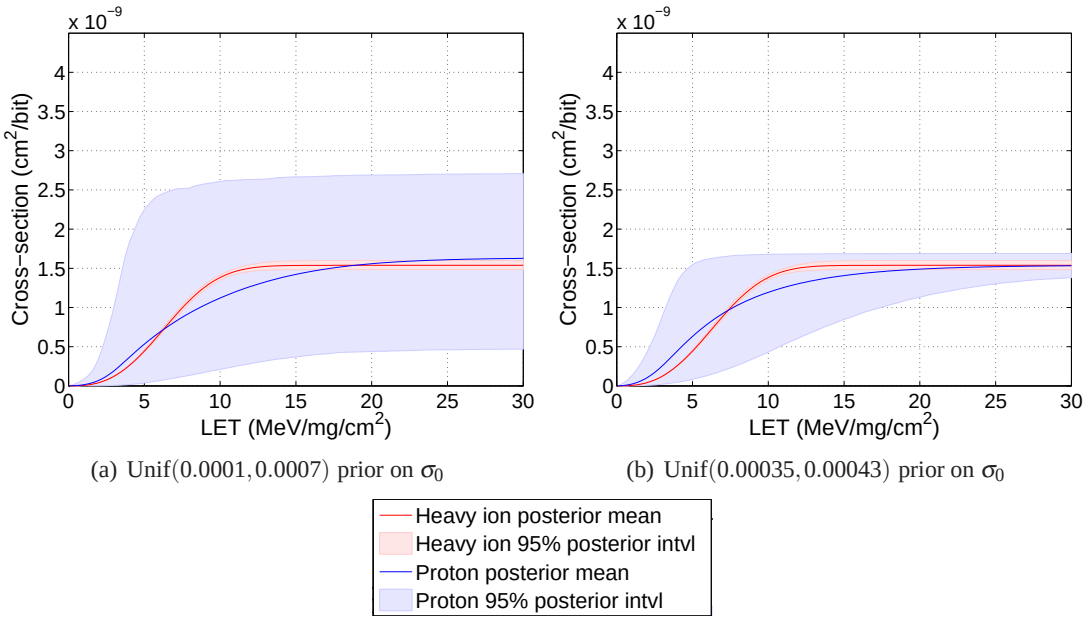


Figure 4.9: Posterior point estimates and 95% intervals for the cross-section vs. LET curve for proton test data (obtained from calibration of PROPSET) and heavy ion test data.



(a) Unif$(0.0001, 0.0007)$ prior on σ_0

(b) Unif$(0.00035, 0.00043)$ prior on σ_0

Figure 4.10: Posterior point estimates and 95% intervals for the cross-section vs. LET curve for proton test data (obtained from calibration of PROPSET, using two new priors for σ_0) and heavy ion test data.

4.4 Modeling extension for the cross-section vs. LET curve of heavy ion tests

Recall, in earlier analyses the cross-section vs. LET curve is calculated as $\sigma_0 G(\cdot)$, where $G(\cdot)$ is assumed to be a Weibull CDF (and sometimes a lognormal CDF is used); see Section 4.2. In this section, we investigate how the shape of $G(\cdot)$ impacts the on-orbit upset rate. The prediction of on-orbit upset rates from test data typically proceeds in two stages. The first stage of the analysis is to fit the cross-section vs. LET curve to the experimental data. In the second stage of the analysis, the estimated cross-section vs. LET curve is combined with the rectangular parallepiped approximation to the geometry of the sensitive volume and a model of the space radiation environment to produce predictions of on-orbit upset rates (Tylka

et al., 1997). A number of codes have been developed to compute on-orbit upset rates; in this work, we use CREME96 (Cosmic Ray Effects on Micro-Electronics), a widely-used code for modeling radiation environments in near-Earth orbits to evaluate radiation effects in spacecraft (Tylka et al., 1997; Petersen, 2008).

We address uncertainty in the cross-section vs. LET curve by treating the entire function $G(\cdot)$ as a key unknown parameter of the model. Specifically, we work with a Dirichlet process prior for $G(\cdot)$, which, along with a parametric prior for σ_0, defines the prior probability model for the mean of the response distribution, which is taken to be Poisson. Hence, from a methodological point of view, we develop a Bayesian semiparametric isotonic regression model for count responses. This approach allows the data to drive the shape of the cross-section vs. LET relationship, resulting in more robust predictive inference for the on-orbit upset rate than conventional models based on the Weibull forms for the cross-section curve. We illustrate the practical utility of our method using the heavy ion test data, D, given in Table 4.1, where $D = \{(c_{ij}, f_{ij}, \ell_i) : i = 1, ..., n_j,\ j = 1, \ldots, P\}$. Here, $P = 4$, but we will present the analysis for a generic integer P.

We model $c_{ij} \mid f_{ij}, \sigma_0, G(\cdot) \overset{ind.}{\sim} \text{Poisson}(f_{ij}\sigma_0 G(\ell_j))$, and $G \sim DP(\alpha, G_0)$, where α is the DP's precision parameter, and G_0 is the (parametric) centering distribution of the DP. In our context, the commonly used Weibull parametric form is a natural choice for G_0, i.e., we let $G_0 \equiv G_0(\ell; w, s) = 1 - \exp(-(\ell/w)^s)$. Alternatively, the lognormal CDF can be used, giving $G_0(\ell; m, w) = \Phi((\ln(\ell) - m)/w)$, where $\Phi(\cdot)$ is the standard normal CDF. Combining the DP prior with the observed data will give information about how close the shape of $G(\cdot)$ is to G_0.

Let $\boldsymbol{\psi}$ be the parameters of the centering distribution (so $\boldsymbol{\psi} = (w, s)$ in the Weibull case

and $\boldsymbol{\psi} = (m, w)$ in the lognormal case), and denote by $\beta_j = G(\ell_j)$, $j = 1, ..., P$, the parameters defining the CDF at the observed LET values, $\ell_1 < \ell_2 < ... < \ell_P$. Then, the DP prior for $G(\cdot)$ implies an ordered Dirichlet prior distribution for $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_P)$. Specifically,

$$p(\boldsymbol{\beta} \mid \alpha, \boldsymbol{\psi}) = \frac{\Gamma(\alpha)}{\prod_{j=1}^{P+1} \Gamma(d_j)} \beta_1^{d_1 - 1} (\beta_2 - \beta_1)^{d_2 - 1} \ldots (\beta_P - \beta_{P-1})^{d_P - 1} (1 - \beta_P)^{d_{P+1} - 1}, \qquad (4.2)$$

where $d_1 \equiv d_1(\alpha, \boldsymbol{\psi}) = \alpha G_0(\ell_1; \boldsymbol{\psi})$, $d_j \equiv d_j(\alpha, \boldsymbol{\psi}) = \alpha(G_0(\ell_j; \boldsymbol{\psi}) - G_0(\ell_{j-1}; \boldsymbol{\psi}))$, for $j = 2, \ldots, P$, and $d_{P+1} \equiv d_{P+1}(\alpha, \boldsymbol{\psi}) = \alpha(1 - G_0(\ell_P; \boldsymbol{\psi}))$.

To complete the model, we must specify priors for α, σ_0, and $\boldsymbol{\psi}$. For α, we use a gamma prior centered around 25 with moderate variance. For σ_0, we use a diffuse exponential prior, and for $\boldsymbol{\psi} = (w, s)$ (or $\boldsymbol{\psi} = (m, w)$), we use two independent exponential priors, the parameters of which are determined by assuming a fairly noninformative range of LET values (twice the observed range). Combining these priors with the Poisson likelihood and the prior in (4.2), the full posterior model $p(\boldsymbol{\beta}, \sigma_0, \alpha, \boldsymbol{\psi}; D)$ is proportional to

$$\prod_{j=1}^{p} \prod_{i=1}^{n_j} \left\{ (f_{ij} \sigma_0 \beta_j)^{c_{ij}} \exp(-f_{ij} \sigma_0 \beta_j) \right\} \times p(\boldsymbol{\beta}; \alpha, \boldsymbol{\psi})\, p(\alpha) p(\boldsymbol{\psi}) p(\sigma_0)$$

We use a hybrid MCMC posterior simulation algorithm to sample from this distribution. Specifically, the posterior full conditional for σ_0 is given by a gamma distribution, which allows for sampling using a Gibbs step. However, the DP prior hyperparameters enter the expression of the posterior distribution in a complex fashion that does not allow direct sampling. Thus, we sample from the posteriors of α and $\boldsymbol{\psi}$ using two M-H steps. Finally, to sample from

the posterior of $\boldsymbol{\beta}$, we use a carefully designed slice sampler (Damien et al., 1999; Neal, 2003). Briefly, the idea of slice sampling is to first sample from a latent variable $U \sim \text{Unif}\,(0, f(\beta_j))$, where $f(\beta_j)$ is proportional to $p(\beta_j \mid \sigma_0, \alpha, \boldsymbol{\psi}, D)$. This defines a "slice" of β_j where $f(\beta_j) > U$. The next value of β_j is sampled from this slice.

The MCMC algorithm provides samples of the posterior distribution for the cross-section vs. LET curve at the observed LET values. Full inference for the curve requires the posterior distribution of $G(\ell_k)$ over a sufficiently fine grid of LET values ℓ_k. In particular, extrapolating the random cdf $G(\cdot)$ beyond the largest and smallest observed LET values is necessary for prediction of the upset rate distribution. Such inference can also reveal important differences in the predicted upset rates between the customary Weibull or lognormal model and the proposed semiparametric alternative.

4.4.1 Predictive inference for the cross-section vs. LET curve

We begin by considering prediction at new LET values, $\tilde{\ell} = (\tilde{\ell}_1, \ldots, \tilde{\ell}_M)$, such that $\ell_j < \tilde{\ell}_1 < \cdots < \tilde{\ell}_M < \ell_{j+1}$, for any $j = 1, \ldots, P-1$. The full model, including the corresponding new $\tilde{\beta}_m = G(\tilde{\ell}_m)$, $m = 1, \ldots, M$, can be written as

$$\begin{aligned} p(\tilde{\boldsymbol{\beta}}, \boldsymbol{\beta}, \sigma_0, \alpha, \boldsymbol{\psi}; D) &\propto p(\tilde{\boldsymbol{\beta}}, \boldsymbol{\beta} \mid \alpha, \boldsymbol{\psi}) p(\sigma_0) p(\alpha) p(\boldsymbol{\psi}) \times \prod_{j=1}^{P} \prod_{j=1}^{n_j} \text{Poisson}(c_{ij}; f_{ij} \sigma_0 \beta_j) \\ &\propto p(\tilde{\boldsymbol{\beta}} \mid \boldsymbol{\beta}, \alpha, \boldsymbol{\psi}) p(\boldsymbol{\beta}, \sigma_0, \alpha, \boldsymbol{\psi}; D), \end{aligned}$$

where $\tilde{\boldsymbol{\beta}} = (\tilde{\beta}_1, \ldots, \tilde{\beta}_M)$, and $p(\tilde{\boldsymbol{\beta}}, \boldsymbol{\beta} \mid \alpha, \boldsymbol{\psi})$ is the joint prior for $(\tilde{\boldsymbol{\beta}}, \boldsymbol{\beta})$ induced by the

DP prior for $G(\cdot)$, with density that extends the form for $p(\boldsymbol{\beta} \mid \alpha, \boldsymbol{\psi})$ in (4.2). Specifically,

$$p(\tilde{\boldsymbol{\beta}}, \boldsymbol{\beta} \mid \alpha, \boldsymbol{\psi}) = \frac{\Gamma(\alpha)}{\Gamma(d'_{j+1}) \prod_{\substack{n=1 \\ n \neq j+1}}^{P+1} \Gamma(d_n) \prod_{m=1}^{M} \Gamma(\tilde{d}_m)} \ \beta_1^{d_1-1} (\beta_2 - \beta_1)^{d_2-1} \dots (\beta_j - \beta_{j-1})^{d_j-1}$$
$$\times (\tilde{\beta}_1 - \beta_j)^{\tilde{d}_1-1} (\tilde{\beta}_2 - \tilde{\beta}_1)^{\tilde{d}_2-1} \dots (\tilde{\beta}_M - \tilde{\beta}_{M-1})^{\tilde{d}_M-1} (\beta_{j+1} - \tilde{\beta}_M)^{d'_{j+1}-1}$$
$$\times (\beta_{j+2} - \beta_{j+1})^{d_{j+2}-1} \dots (\beta_P - \beta_{P-1})^{d_P-1} (1 - \beta_P)^{d_{P+1}-1}, \tag{4.3}$$

where $d'_{j+1} = \alpha(G_0(\ell_{j+1}; \boldsymbol{\psi}) - G_0(\tilde{\ell}_M; \boldsymbol{\psi}))$, $\tilde{d}_1 = \alpha(G_0(\tilde{\ell}_1; \boldsymbol{\psi}) - G_0(\ell_j; \boldsymbol{\psi}))$, and for $m = 2, \dots, M$, $\tilde{d}_m = \alpha(G_0(\tilde{\ell}_m; \boldsymbol{\psi}) - G_0(\tilde{\ell}_{m-1}; \boldsymbol{\psi}))$. Moreover, the d_j, $j = 1, \dots, P+1$, are as defined in Section 4.2. Hence, $p(\tilde{\boldsymbol{\beta}} \mid D) = \int p(\tilde{\boldsymbol{\beta}} \mid \boldsymbol{\beta}, \alpha, \boldsymbol{\psi}) p(\boldsymbol{\beta}, \sigma_0, \alpha, \boldsymbol{\psi}; D) \, \mathrm{d}\boldsymbol{\beta} \, \mathrm{d}\sigma_0 \, \mathrm{d}\alpha \, \mathrm{d}\boldsymbol{\psi}$, and therefore, using Monte Carlo integration based on the posterior draws from $p(\boldsymbol{\beta}, \sigma_0, \alpha, \boldsymbol{\psi}; D)$, we can sample $p(\tilde{\boldsymbol{\beta}} \mid D)$ by additional sampling from $p(\tilde{\boldsymbol{\beta}} \mid \boldsymbol{\beta}, \alpha, \boldsymbol{\psi})$.

Using the expressions for $p(\boldsymbol{\beta} \mid \alpha, \boldsymbol{\psi})$ and $p(\tilde{\boldsymbol{\beta}}, \boldsymbol{\beta} \mid \alpha, \boldsymbol{\psi})$ in (4.2) and (4.3), respectively, and the fact that $d_{j+1} = d'_{j+1} + \sum_{m=1}^{M} \tilde{d}_m$, the density for $p(\tilde{\boldsymbol{\beta}} \mid \boldsymbol{\beta}, \alpha, \boldsymbol{\psi})$ can be obtained as

$$\frac{\Gamma(d_{j+1})(\beta_{j+1} - \beta_j)^{-M}}{\Gamma(d'_{j+1}) \prod_{m=1}^{M} \Gamma(\tilde{d}_m)} \left(\frac{\tilde{\beta}_1 - \beta_j}{\beta_{j+1} - \beta_j} \right)^{\tilde{d}_1 - 1} \dots \left(\frac{\tilde{\beta}_M - \tilde{\beta}_{M-1}}{\beta_{j+1} - \beta_j} \right)^{\tilde{d}_M - 1} \left(\frac{\beta_{j+1} - \tilde{\beta}_M}{\beta_{j+1} - \beta_j} \right)^{d'_{j+1} - 1}.$$

Hence, $p(\tilde{\boldsymbol{\beta}} \mid \boldsymbol{\beta}, \alpha, \boldsymbol{\psi})$ can be readily sampled, since the density above corresponds to random vector $(\beta_j + (\beta_{j+1} - \beta_j)\omega_1, \dots, \beta_j + (\beta_{j+1} - \beta_j)\omega_M)$, where $(\omega_1, \dots, \omega_M)$ follows an ordered Dirichlet distribution with parameters $(\tilde{d}_1, \dots, \tilde{d}_M, d'_{j+1})$.

We next turn to extrapolation for the cross-section vs. LET curve beyond the largest observed LET value. Consider prediction at new LET values $\boldsymbol{\ell'} = (\ell'_1, \dots, \ell'_M)$, such that $\ell_P <$

$\ell'_1 < \cdots < \ell'_M$. The model augmented with $\boldsymbol{\ell'}$ and the corresponding $\boldsymbol{\beta}' = (\beta'_1, \ldots, \beta'_M)$, where $\beta'_m = G(\ell'_m)$, $m = 1, \ldots, M$, becomes $p(\sigma_0, \boldsymbol{\beta}', \boldsymbol{\beta}, \alpha, \boldsymbol{\psi} \mid D) = p(\boldsymbol{\beta}' \mid \boldsymbol{\beta}, \alpha, \boldsymbol{\psi}) p(\sigma_0, \boldsymbol{\beta}, \alpha, \boldsymbol{\psi} \mid D)$. In this case, $p(\boldsymbol{\beta}' \mid \boldsymbol{\beta}, \alpha, \boldsymbol{\psi})$ can be recognized as the probability density for the random vector given by $(\beta_P + (1-\beta_P)\omega_1, \ldots, \beta_P + (1-\beta_P)\omega_M)$, where $(\omega_1, \ldots, \omega_M)$ has an ordered Dirichlet distribution with parameters $(d'_1, \ldots, d'_M, d'_{M+1})$. Here, $d'_1 = \alpha\,(G_0(\ell'_1; \psi) - G_0(\ell_P; \boldsymbol{\psi}))$, $d'_m = \alpha\big(G_0(\ell'_m; \psi) - G_0(\ell'_{m-1}; \psi)\big)$, $m = 2, \ldots, M$, and $d'_{M+1} = \alpha\,(1 - G_0(\ell'_M; \psi))$, such that $d_{P+1} = \sum_{m=1}^{M+1} d'_m$.

Finally, extrapolation in the region below the smallest observed LET value is obtained in a similar fashion. Consider prediction at new LET values $\boldsymbol{\ell''} = (\ell''_1, \ldots, \ell''_M)$, such that $\ell''_1 < \cdots < \ell''_M < \ell_1$. Let $\boldsymbol{\beta}'' = (\beta''_1, \ldots, \beta''_M)$, where $\beta''_m = G(\ell''_m)$, for $m = 1, \ldots, M$, and $d''_1 = \alpha G_0(\ell''_1; \psi)$, $d''_m = \alpha\left(G_0(\ell''_m; \psi) - G_0(\ell''_{m-1}; \psi)\right)$, for $m = 2, \ldots, M$, and $d''_{M+1} = \alpha\,(G_0(\ell_1; \psi) - G_0(\ell''_M; \psi))$, such that $d_1 = \sum_{m=1}^{M+1} d''_m$. Then, the required conditional density, $p(\boldsymbol{\beta}'' \mid \boldsymbol{\beta}, \alpha, \boldsymbol{\psi})$, corresponds to the distribution of random vector $(\beta_1\omega_1, \ldots, \beta_1\omega_M)$, where $(\omega_1, \ldots, \omega_M)$ has an ordered Dirichlet distribution with parameters $(d''_1, \ldots, d''_M, d''_{M+1})$.

4.4.2 Results

For both the semiparametric models (with Weibull and lognormal centering distributions) and the parametric models (Weibull and lognormal parametric CDFs), we obtain MCMC samples of all the model parameters, then based on those we obtain posterior samples for $G(\cdot)$ over a fine grid of LET values and multiply by the samples from σ_0 to obtain posterior samples for the cross-section vs. LET curve. The results are summarized in Figure 4.11. Under the DP prior, the model is relatively invariant to the choice of the centering distribution, the main

difference being in the uncertainty bands between the second and third observed LET values and in the extrapolation region. We also note that for both semiparametric models, the 95% intervals become wider between the observations. The cross-section results were found to be not very sensitive to the priors on σ_0 and $\boldsymbol{\psi} = (w, s)$.

Examining the parametric Weibull and lognormal fits (Figure 4.11, right panel), it can be seen clearly that the choice of parametric function can have a large impact on the estimated cross-section vs. LET curve, especially in the extrapolation region. The lognormal CDF's 95% posterior curves completely miss the cross-section values at the largest observed LET value. Also, there is more agreement between the two semiparametric models and the Weibull CDF model, although both the parametric models under estimate curve uncertainty for small values of LET up to the third observed LET value.

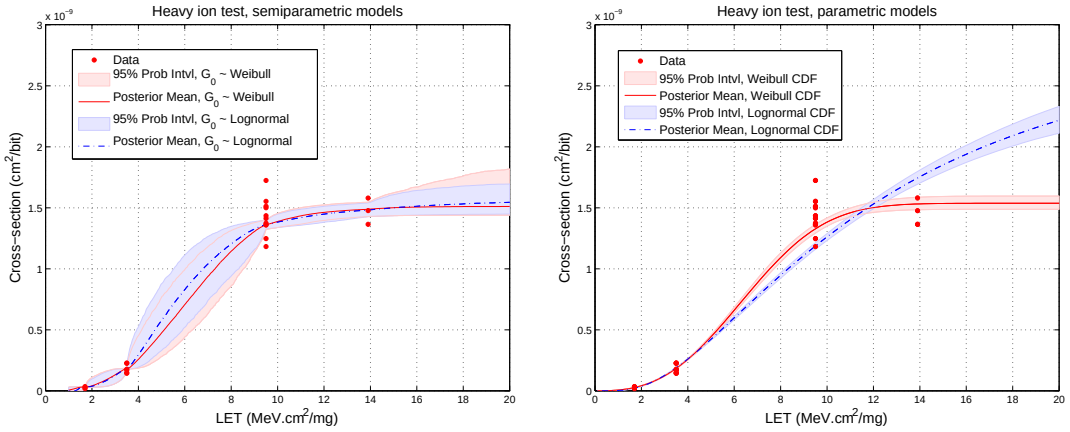


Figure 4.11: Point estimates and 95% probability intervals for the cross-section vs. LET curve under the semiparametric DP model (left panel) and under the parametric models.

Inference for the rest of the model parameters under both approaches (and models) is similar so to that obtained in Section 4.2.1, so we omit plotting those posteriors here. However,

we show the plot for the posterior of α (Figure 4.12) since it did not appear in the parametric model discussion earlier. We see that there is a small amount of learning for α, with greater values slightly favored (the posterior means are approximately 42 and 37 under the Weibull and lognormal centering distribution, respectively, compared with 25 for the prior mean) indicating less divergence from the centering distributions. We also tried uniform priors on $(0, 100)$ for α, but this did not change the resulting posterior inference for the cross-section vs. LET curve. In general, posterior inference for the curve was unaffected by the choice of centering distribution and insensitive to changes in the prior assumption for the hyperparameters of the model.

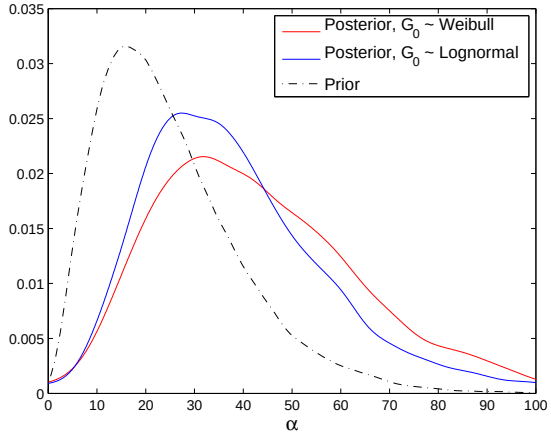


Figure 4.12: Posterior vs. prior densities for α under the semiparametric DP model using a Weibull centering distribution (blue), and a lognormal centering distribution (red).

Now that posterior samples of the cross-section vs. LET curve are available, they are used as input to CREME96 along with the geometry of the sensitive volume for the device and orbit specifications to predict the SEU rate. Because we have a distribution for the cross-section vs. LET curve at every LET value over a fine grid of LET values, the resulting SEU rate will be a distribution. For our analysis, we consider a nominal low-Earth orbit (450km altitude, 51.6

deg inclination, solar minimum, 100 mils aluminum shielding). In this orbit protons will also contribute to the upset rate; here, we consider only the contribution due to heavy-ion radiation and obtain results under each of the models entertained in the earlier analyses. Then using the same orbit information, we also calculate the upset rates for the proton test based on the resutls reported in Figure 4.8. Figure 4.13 plots the posterior densities for all five on-orbit upset rates. We find general agreement in the estimated on-orbit upset rates between all the models with the proton test upset rate distribution having the heaviest tail.

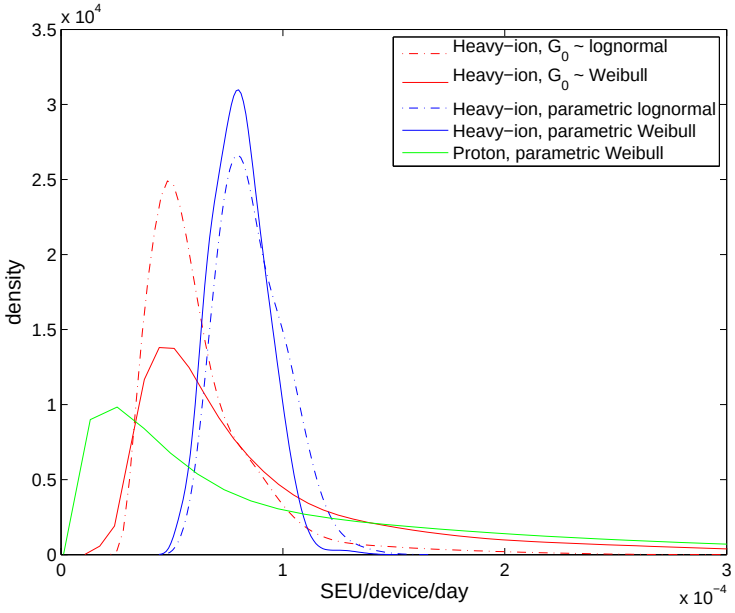


Figure 4.13: Posterior densities of the predicted on-orbit upset rates using the two semiparametric models and the two parametric models for the heavy-ion data as well as the parametric model for the proton test data used in the calibration of PROPSET.

4.5 Discussion

We have presented a Bayesian approach to the modeling of uncertainty in the parameters of the cross-section vs. LET curve based on heavy-ion and proton testing. The cross-section vs. LET curve is key to calculating the SEU rate in spaceborne microelectronics, and therefore its accurate estimation is important for the success of space missions. To quantify uncertainty about the curve parameters in the heavy ion test, initially a standard parametric model is used for the analysis. In the proton test case, PROPSET, a stochastic simulator, is calibrated in order to estimate the distributions of the curve parameters. Since PROPSET is computationally expensive, we emulate it using a new approach that we have developed, where the emulator is built from flexible nonparametric Dirichlet process mixtures. This new emulation approach allows for the modeling of non standard distributional shapes (multimodality and skewness) of the PROPSET output densities. Then, the results from the standard parametric model are compared with the results from the calibration of PROPSET. Finally, we investigate uncertainty in the cross-section vs. the LET curve is by treating the entire curve as unknown model parameter using a semiparametric isotonic regression model with a DP prior for the monotonically increasing regression function.

For the calibration of PROPSET, only eight experimental observations were available, leading to considerable uncertainty in the estimated cross-section vs. LET curve. Typically the amount of observed data from proton tests is very limited, which motivates the need for informative priors. Potentially, informative priors for σ_0 and chip width may be specified from expert knowledge about the dimensions of the semiconductor device, which we believe will help

reduce uncertainty in the results. Additionally, in future calibration analysis, a bias function can be introduced to account for discrepancy between PROPSET and the physical process that generates on-orbit upsets. The bias function will be a component of the mean of the normal distribution used for the observed experimental data.

Chapter 5

Conclusion

Quantifying uncertainty and identifying its sources in computer simulators is vital for accurate scientific inference. In the work presented here, we develop new Bayesian tools to model the uncertainty associated with computationally expensive computer simulators. The main contributions of this thesis are in the areas of sensitivity analysis, emulation and calibration. The motivations behind this work are complex scientific problems in remote sensing and aerospace engineering, where the simulator plays an important role in inference and prediction.

The methods developed in this thesis are presented in Chapters 2, 3, and 4. Chapter 2 includes the development of a fully Bayesian approach to quantify uncertainty in a deterministic simulator's output based on a GP prior for the output function. Here, easy to calculate point estimates and standard errors are derived for the main effects of the simulator inputs. Also, an algorithm for sampling the entire posterior distribution of the sensitivity indices is constructed by combining posterior samples from the GP emulator with a Monte Carlo approach for simulating from the input distribution. This work is applied to the Leaf-Canopy Model (LCM),

a deterministic simulator that models the light reflected by a vegetated region of the Earth as observed by a satellite sensor.

Chapter 3 includes the development of a new framework for the emulation and calibration of stochastic computer simulators, where two approaches are explored: The first relies on modeling both the simulator output and field observations using a parametric distribution with smoothly varying mean modeled using a GP prior; that is, for both emulation and calibration, the same model for the data is used. In the second approach, we develop a new method that involves modeling the joint density of the simulator's input and output using a flexible (nonparametric) DP mixture model, with the emulator formulated through the implied conditional distribution for the output given the inputs. For calibration, this modeling approach is extended to allow borrowing of strength between the two different sources of data (field observations and simulator data). Here, the field data are modeled using a parametric distribution with mean that has the same form as the mean of the emulation model. Both approaches are illustrated using synthetic data examples that highlight their strength and limitation.

In Chapter 4, we discuss the motivating scientific problem for the methodological work developed in Chapter 3 as well as new work in the area of Bayesian semiparametric isotonic regression for count responses. The motivation behind the work presented in this chapter is quantifying uncertainty in the rate of "soft errors" or single event upsets (SEUs) in spaceborne microelectronics. SEUs occur naturally in Earth's orbit due to heavy ions or protons interacting with a sensitive node in the device. Of scientific interest is investigating whether a device's cross-section vs. LET curve obtained from a proton collision gives a good estimate of the curve obtained from a heavy ion collision.

In order to obtain the curve parameters for the heavy ion case, we use a standard parametric model for the curve. For the proton case, we calibrate PROPSET, a stochastic simulator that models the bombardment of a microchip with protons and records the number of upsets. Both PROPSET data and proton test data are used for the calibration. Then the results from the parametric model for the heavy ions and the calibration model for PROPSET are compared. Finally, uncertainty about the curve for the heavy ion case is further examined by taking the entire curve to be an unknown parameter of the model. Since the cross-section vs. LET curve is monotonically increasing, we develop a Bayesian semiparametric isotonic regression model for it by using a Poisson distribution for the SEU counts and placing a DP prior on the CDF defining the monotonically increasing curve.

There are a number of future research ideas that extend the methods developed in this thesis. First, the new emulation framework for stochastic simulators (where the inputs and outputs are modeled jointly using DP mixtures) may be used to develop a new approach for sensitivity analysis of stochastic simulators by obtaining posterior distributions of the conditional densities needed to build sensitivity indices of any desired order. Second, a fully nonparametric mixture model may be developed for the field data used to calibrate a stochastic simulator in an analogous fashion to the emulation model developed in Section 3.3.1. Posterior inference for the calibration parameters will be based on both the simulator data and field data, which will be linked through the DP priors of their respective models. Here, we will need moderate to large sample sizes for the field data. Hence, different applications (e.g., environmental applications) will be needed to illustrate the methodology instead of applications where field data is scarce. Third, bias will need to be introduced into the calibration model in order to account for dis-

crepancy between the simulator and reality. Using the calibration model in Section 3.3.2, the bias function could be modeled as a component of the mean of the normal distribution used for the observed field data. As, a fully nonparametric model for the observed experimental data is developed, a bias function could also take the form of a nonparametric mixture or a GP.

Appendix A

GP posterior simulation technical details

A.1 Posterior inference for the GP emulator of a deterministic simulator

Following the GP model formulation in Section 2.1.1, the posterior distribution is given by $p(\mu, \tau^2, \phi \mid D) \propto N_n\left(y|\mu \mathbf{1}_n, \tau^2 R_\phi\right) p(\mu)p(\tau^2)p(\phi_1)...p(\phi_k)$. We place independent priors on the GP parameters, specifically, we use a $N(a_\mu, b_\mu)$ prior for μ, an $\Gamma^{-1}(a_\tau, b_\tau)$ prior for τ^2, and Unif$(0, b_{\phi_\ell})$ priors for ϕ_ℓ, $\ell = 1, \ldots, k$. Here, $\Gamma^{-1}(a,b)$ denotes the inverse-gamma distribution with mean $b/(a-1)$, provided $a > 1$.

Posterior simulation from $p(\mu, \tau^2, \phi \mid D)$ proceeds via Gibbs sampling. The full conditional posterior distribution for μ is normal with variance $S = \left(\tau^{-2} 1_n^T R_\phi^{-1} 1_n + b_\mu^{-1}\right)^{-1}$, and mean $M = S\left(\tau^{-2} 1_n^T R_\phi^{-1} y + a_\mu b_\mu^{-1}\right)$. For τ^2, the posterior full conditional is $\Gamma^{-1}(A,B)$, with $A = a_\tau + 0.5n$ and $B = b_\tau + 0.5\,(y - \mu 1_n)^T R_\phi^{-1}\,(y - \mu 1_n)$. The posterior full conditional for

each ϕ_ℓ, $\ell = 1,\ldots,k$, is proportional to

$$p(\phi_\ell \mid \mu, \tau^2, \phi_{-\ell}, D) \propto |R_\phi|^{-\frac{1}{2}} \exp\left(-\frac{1}{2\tau^2}(y-\mu 1_n)^T R_\phi^{-1}(y-\mu 1_n)\right) \times 1_{(0,b_{\phi_\ell})}(\phi_\ell)$$

which cannot be sampled directly. We use Metropolis-Hastings steps for each ϕ_ℓ based on a right-truncated exponential proposal distribution with density $d_\ell \exp(-d_\ell \phi_\ell)/\{1-\exp(-d_\ell b_{\phi_\ell})\}$. To choose the rate parameter d_ℓ (which is the only tuning parameter), we obtain an estimate $\tilde{\phi}_\ell$ of ϕ_ℓ (e.g., the MLE), set $\tilde{\phi}_\ell$ equal to the median of the proposal distribution, and solve for d_ℓ.

A.2 Prior specification of the GP parameters for the emulation of a deterministic simulator

We set $a_\tau = 2$, a value that yields infinite variance for the corresponding inverse-gamma prior (i.e., a relatively noninformative specification). To specify the hyperparameters a_μ, b_μ and b_τ, note that for each i, $\mathrm{E}(Y_i) = \mathrm{E}(\mathrm{E}(Y_i|\mu)) = \mathrm{E}(\mu) = a_\mu$, and $\mathrm{Var}(Y_i) = \mathrm{E}\left(\mathrm{Var}\left(Y_i|\tau^2\right)\right) + \mathrm{Var}(\mathrm{E}(Y_i|\mu)) = b_\tau + b_\mu$. Now, assume we have a prior guess for the center, c_y, and range, r_y, of the simulator output values. (For instance, such information is readily available for our application to the LCM simulator.) Then, we set $a_\mu = c_y$, and $b_\tau = b_\mu \approx (r_y/4)^2$, using $2(r_y/4)^2 \approx b_\tau + b_\mu$, with the extra inflation factor 2, and splitting the variance estimate equally between b_τ and b_μ.

Specifying prior information for ϕ_ℓ, $\ell = 1,\ldots,k$ is more difficult. One way to specify

b_{ϕ_ℓ} is based on the interpretation of ϕ_ℓ under the product power exponential correlation function in (2.1): for any fixed α_ℓ, it controls how fast the correlation decays with distance in the direction of the ℓ–th input x_ℓ. In particular, for $\alpha_\ell = 1$, $3/\phi_\ell$ is the "range of dependence", i.e., the value of the distance $d = |x_\ell - x'_\ell|$ that yields correlation approximately equal to 0.05. Hence, we could use, say, $0.1d_{max}$, where $d_{max} = \max|x_\ell - x'_\ell|$, as a rough guess at $3/\phi_\ell$ and specify b_{ϕ_ℓ} from $0.1d_{max} = 3/b_{\phi_\ell}$. For the application to the LCM simulator, we used the available range for each input variable to specify d_{max} and thus b_{ϕ_ℓ}. The resulting uniform priors for the ϕ_ℓ led to a significant amount of prior to posterior learning.

A.3 Posterior inference for emulation of stochastic simulators using GP priors

Here, we provide MCMC posterior simulation details for sampling from (3.1), which may be implemented in two ways. The first is by considering all the parameters of the model (without marginalizing over the GP prior). Working with the joint posterior in (3.1), we obtain the posterior full conditional distribution of the model parameters. The posterior full conditional for $\boldsymbol{\delta}$, $p(\boldsymbol{\delta} \mid \tau^2, \mu, \lambda^2, \boldsymbol{\phi}, D^*)$, is proportional to $\exp\left(-\frac{1}{2\tau^2}(\boldsymbol{y} - \boldsymbol{\delta}_m)^T(\boldsymbol{y} - \boldsymbol{\delta}_m)\right) \times \exp\left(-\frac{1}{2\lambda^2}(\boldsymbol{\delta} - \mu\mathbf{1}_m)^T R_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\delta} - \mu\mathbf{1}_m)\right)$. Therefore, $\boldsymbol{\delta} \mid \tau^2, \mu, \lambda^2, \boldsymbol{\phi}, D^*$ has a multivariate normal distribution with mean and covariance,

$\mathrm{E}(\boldsymbol{\delta} \mid \tau^2, \mu, \lambda^2, \boldsymbol{\phi}, D^*) = \mathrm{Cov}(\boldsymbol{\delta} \mid \tau^2, \mu, \lambda^2, \boldsymbol{\phi}, D^*) \times \left(\boldsymbol{y}/\tau^2 + \mu R_{\boldsymbol{\phi}}^{-1}\mathbf{1}_m/\lambda^2\right)$, and

$\mathrm{Cov}(\boldsymbol{\delta} \mid \tau^2, \mu, \lambda^2, \boldsymbol{\phi}, D^*) = \left(I_{(m\times m)}/\tau^2 + R_{\boldsymbol{\phi}}^{-1}/\lambda^2\right)^{-1}$.

The posterior full conditional for μ, $p(\mu \mid \boldsymbol{\delta}, \tau^2, \lambda^2, \boldsymbol{\phi}, D^*)$, is proportional to

$\exp\left(-\frac{1}{2\lambda^2}(\boldsymbol{\delta}-\mu\mathbf{1}_m)^T R_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\delta}-\mu\mathbf{1}_m)\right) \times \exp\left(-\frac{1}{2b_\mu}(\mu-a_\mu)^2\right)$. Thus, $\mu \mid \boldsymbol{\delta},\tau^2,\lambda^2,\boldsymbol{\phi},D^*$ is normal with $\mathrm{E}(\mu \mid \boldsymbol{\delta},\tau^2,\lambda^2,\boldsymbol{\phi},D^*) = \mathrm{Var}(\mu \mid \boldsymbol{\delta},\tau^2,\lambda^2,\boldsymbol{\phi},D^*) \times \left(\mathbf{1}_m^T R_{\boldsymbol{\phi}}^{-1}\boldsymbol{\delta}/\lambda^2 + a_\mu/b_\mu\right)$, and $\mathrm{Var}(\mu \mid \boldsymbol{\delta},\tau^2,\lambda^2,\boldsymbol{\phi},D^*) = \left(\mathbf{1}_m^T R_{\boldsymbol{\phi}}^{-1}\mathbf{1}_m/\lambda^2 + 1/b_\mu\right)^{-1}$.

The posterior full conditional for τ^2, $p(\tau^2 \mid \boldsymbol{\delta},\mu,\lambda^2,\boldsymbol{\phi},D^*)$ is proportional to $\left(\tau^2\right)^{-\frac{m}{2}} \exp\left(-\frac{1}{2\tau^2}(\boldsymbol{y}-\boldsymbol{\delta}_m)^T(\boldsymbol{y}-\boldsymbol{\delta}_m)\right) \times \left(\tau^2\right)^{-(a_{\tau^2})-1} \exp\left(-b_{\tau^2}/\tau^2\right)$. Therefore, $\tau^2 \mid \boldsymbol{\delta},\mu,\lambda^2,\boldsymbol{\phi},D^* \sim IG(a_{\tau^2}+m/2\,,\, b_{\tau^2}+(\boldsymbol{y}-\boldsymbol{\delta}_m)^T(\boldsymbol{y}-\boldsymbol{\delta}_m)/2)$. Similarly, we find that $\lambda^2 \mid \boldsymbol{\delta},\mu,\tau^2,\boldsymbol{\phi},D^* \sim IG(a_{\lambda^2}+m/2\,,\, b_{\lambda^2}+(\boldsymbol{\delta}-\mu\mathbf{1}_m)^T R_{\boldsymbol{\phi}}^{-1}(\boldsymbol{y}-\boldsymbol{\delta}_m)/2)$. Finally, the distribution $p(\boldsymbol{\phi} \mid \mu,\tau^2,\lambda^2,D^*) \propto \mid R_{\boldsymbol{\phi}} \mid^{-\frac{1}{2}} \exp\left(-\frac{1}{2\lambda^2}(\boldsymbol{\delta}-\mu\mathbf{1}_m)^T R_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\delta}-\mu\mathbf{1}_m)\right) \times p(\boldsymbol{\phi})$ is not in closed form. Thus, under this approach we need $m+3$ Gibbs steps (for $\boldsymbol{\delta},\mu,\tau^2,$ and, λ^2) and a M-H step for $\boldsymbol{\phi}$.

The second way to proceed with posterior inference is by marginalizing over the GP prior in (3.1), so the joint posterior model becomes

$$p(\tau^2,\mu,\lambda^2,\boldsymbol{\phi} \mid D^*) \propto N_m\left(\boldsymbol{y};\, \mu\mathbf{1}_m\,,\, \Sigma = \tau^2 I_m + \lambda^2 R_{\boldsymbol{\phi}}\right) \times IG(\tau^2;\, a_{\tau^2}\,,b_{\tau^2}) \times N(\mu;\, a_\mu\,,b_\mu) \times IG(\lambda^2;\, a_{\lambda^2}\,,b_{\lambda^2}) \times p(\boldsymbol{\phi})$$

This approach results in a normal posterior full conditional for μ, i.e., $\mu \mid \tau^2,\lambda^2,\boldsymbol{\phi},D^* \sim N(m,S)$, where $m = S\left(\mathbf{1}_m^T\Sigma^{-1}\boldsymbol{y} + a_\mu/b_\mu\right)$, and $S = \left(\mathbf{1}_m^T\Sigma^{-1}\mathbf{1}_m + 1/b_\mu\right)^{-1}$. However, the posterior full conditional distributions for τ^2, λ^2, and $\boldsymbol{\phi}$ are not in closed form. Thus, marginalizing over the GP prior, reduced the parameter space to 5 parameters, with one Gibbs step for μ and M-H steps for the τ^2, λ^2, and $\boldsymbol{\phi}$.

Appendix B

DP mixture conditional posterior sampling

Here, we provide the details of block Gibbs sampling for the posterior model in (3.9). We work with the posterior full conditional distribution of each of the model parameters, and use the notation $p(\text{parameter} \mid \dots)$ to denote conditioning that parameter on the rest of the parameters of the model.

• Updating $\boldsymbol{\mu}_\ell$ and Σ_ℓ, $\ell = 1,\dots,N$: Let m^* be the number of distinct elements (clusters) $\{L_i^* : i = 1,\dots,m^*\}$,

$$p(\boldsymbol{\mu}_\ell \mid \dots, D) \propto \mathrm{N}_3(\boldsymbol{\mu}_\ell; \boldsymbol{m}, V) \prod_{i=1}^{m^*} \prod_{\{j: L_j = L_i^*\}} \mathrm{N}_3(\boldsymbol{u}_j; \boldsymbol{\mu}_{L_i^*}, \Sigma_{L_i^*})$$

$$p(\Sigma_\ell \mid \dots, D) \propto \mathrm{IW}_3(\Sigma_\ell; ; \nu, S) \prod_{i=1}^{m^*} \prod_{\{j: L_j = L_i^*\}} \mathrm{N}_3(\boldsymbol{u}_i; \boldsymbol{\mu}_{L_i^*}, \Sigma_{L_i^*})$$

$\rightarrow$ if $\ell \notin \{L_i^* : i = 1,\dots,m^*\}$,

$\boldsymbol{\mu}_\ell$ is drawn from $G_0(.; \boldsymbol{m}, V) = \mathrm{N}_3(\boldsymbol{\mu}_\ell; \boldsymbol{m}, V)$, and

Σ_ℓ is drawn from $G_0(.; \nu, S) = \mathrm{IW}_3(\Sigma; \nu, S)$

$$\rightarrow \text{ if } \ell = L_i^* : i = 1,\ldots,m^*,$$

$$p(\boldsymbol{\mu}_{L_i^*} \mid \ldots, D) \propto \mathrm{N}_3(\boldsymbol{\mu}_{L_i^*}; \boldsymbol{m}, V) \prod_{\{j:L_j=L_i^*\}} \mathrm{N}_3(\boldsymbol{u}_j; \boldsymbol{\mu}_{L_i^*}, \Sigma_{L_i^*})$$

$$\sim \mathrm{N}_3\left(\left(V^{-1} + M_j^* \Sigma_{L_i^*}^{-1}\right)^{-1}\left(V^{-1}\boldsymbol{m} + \Sigma_{L_i^*}^{-1} \sum_{\{i:L_j=L_i^*\}} \boldsymbol{u}_j\right), \left(V^{-1} + M_j^* \Sigma_{L_i^*}^{-1}\right)^{-1}\right)$$

$$p(\Sigma_{L_i^*} \mid \ldots, D) \propto \mathrm{IW}_3(\Sigma_{L_i^*}; \nu, S) \prod_{\{j:L_j=L_i^*\}} \mathrm{N}_3(\boldsymbol{u}_j; \boldsymbol{\mu}_{L_i^*}, \Sigma_{L_i^*})$$

$$\sim \mathrm{IW}_3\left(\nu + M_j^*, S + \sum_{\{j:L_j=L_i^*\}} \left(\boldsymbol{u}_i - \boldsymbol{\mu}_{L_i^*}\right)\left(\boldsymbol{u}_i - \boldsymbol{\mu}_{L_i^*}\right)^T\right),$$

where M_i^* is the number of times each L_i^* appears in L, i.e., $M_i^* = |\{j : L_j = L_i^*\}|$.

• Updating $\boldsymbol{p} = (p_1, \ldots, p_N)$:

$$p(\boldsymbol{p} \mid \ldots, D) \propto \prod_{j=1}^{m} \left\{p(L_j \mid \boldsymbol{p})\right\} p(\boldsymbol{p}; \alpha) = p(\boldsymbol{p}; \alpha) \prod_{j=1}^{m} \sum_{\ell=1}^{N} p_\ell \delta(L_j) = p(\boldsymbol{p}; \alpha) \prod_{\ell=1}^{N} p_\ell^{M_\ell},$$

where M_ℓ is the number of times each ℓ appears in L, i.e., $M_\ell = |\{j : L_j = \ell\}|$.

$$p(\boldsymbol{p} \mid \ldots, D) \propto \alpha^{N-1} p_N^{\alpha-1} (1-p_1)^{-1} (1-(p_1+p_2))^{-1} \times \cdots \times \left(1 - \sum_{\ell=1}^{N-2} p_\ell\right)^{-1}$$

$$\times p_1^{M_1} \times \cdots \times p_N^{M_N}$$

$$\propto p_1^{(M_1+1)-1} \times \cdots \times p_{N-1}^{(M_{N-1}+1)-1} p_N^{(\alpha+M_N)-1}$$

$$\times (1-p_1)^{(\alpha+\sum_{\ell=2}^{N} M_\ell)-\{(M_2+1)+(\alpha+\sum_{\ell=3}^{N} M_\ell)\}}$$

$$\times \cdots \times \left(1 - \sum_{\ell=1}^{N-2} p_\ell\right)^{(\alpha+M_{N-1}+M_N)-\{(M_{N-1}+1)+(\alpha+M_N)\}}$$

Thus, $p(\boldsymbol{p} \mid \ldots, D)$ is a generalized Dirichlet distribution, $\mathrm{GD}(a', b')$, where:

$$a' = (M_1 + 1, M_2 + 1, \ldots, M_{N-1} + 1),$$
$$b' = \left(\alpha + \sum_{\ell=2}^{N} M_\ell, \alpha + \sum_{\ell=3}^{N} M_\ell, \ldots, \alpha + M_N \right).$$

Therefore, $p(\boldsymbol{p} \mid \ldots, D)$ may be sampled through independent lated Beta variables,

$$V_\ell^* \overset{ind}{\sim} \mathrm{Beta}\left(1 + M_\ell, \alpha + \sum_{r=\ell+1}^{N} M_r\right), \ell = 1, \ldots, N-1.$$
$$p_1 = V_1^*; p_\ell = V_\ell^* \prod_{r=1}^{\ell-1} (1 - V_r^*), \ell = 2, \ldots, N-1; p_N = 1 = \sum_{\ell=1}^{N-1} p_\ell.$$

• Updating L_j, $j = 1, \ldots, m$:

$$p(L_j \mid \ldots, D) \propto \mathrm{N}_3(\boldsymbol{u}_j; \boldsymbol{\mu}_{L_j}, \Sigma_{L_j})\, p(L_j \mid \boldsymbol{p}) = \mathrm{N}_3(\boldsymbol{u}_j; \boldsymbol{\mu}_{L_j}, \Sigma_{L_j}) \sum_{\ell=1}^{N} p_\ell\, \delta_\ell(L_j)$$
$$= \sum_{\ell=1}^{N} \tilde{p}_{\ell j}\, \delta_\ell(L_j)$$

That is, L_j is drawn from a discrete distribution on $\{1, \ldots, N\}$ with probabilities

$$\tilde{p}_{\ell j} = \frac{p_\ell\, \mathrm{N}_3(\boldsymbol{u}_j; \boldsymbol{\mu}_\ell, \Sigma_\ell)}{\sum_{\ell=1}^{N} p_\ell\, \mathrm{N}_3(\boldsymbol{u}_j; \boldsymbol{\mu}_\ell, \Sigma_\ell)},\ \ell = 1, \ldots, N.$$

• Updating α:

$$\begin{aligned} p(\alpha \mid, \boldsymbol{p}, D) \;&\propto\; p(\boldsymbol{p}; \alpha)\, p(\alpha) \;\propto\; \alpha^{N-1} p_N^{\alpha} \times \mathrm{Gam}(a_\alpha, b_\alpha) \\ &\propto \alpha^{N-1} \exp\{\alpha \log(p_N)\}\, \alpha^{a_\alpha - 1} \exp\{-b_\alpha\, \alpha\} \\ &\sim \mathrm{Gam}\left(N + a_\alpha - 1, b_\alpha - \log(p_N)\right) \end{aligned}$$

For numerical stability, compute $\log p_N = \log \prod_{r=1}^{N-1}(1 - V_r^*) = \sum_{r=1}^{N-1} \log(1 - V_r^*)$.

• Updating $\boldsymbol{m}$:

$$\begin{aligned} p(\boldsymbol{m} \mid \ldots, D) &\propto \left\{ \prod_{\ell=1}^{N} \mathrm{N}_3(\boldsymbol{\mu}_\ell; \boldsymbol{m}, V) \right\} p(\boldsymbol{m}) \\ &= \left\{ \prod_{i=1}^{m^*} \mathrm{N}_3(\boldsymbol{\mu}_{L_i^*}; \boldsymbol{m}, V) \right\} \mathrm{N}_3\left(\boldsymbol{a}_{\boldsymbol{m}}, B_{\boldsymbol{m}}\right) \\ &\sim \mathrm{N}_3\left(\left(m^* V^{-1} + B_{\boldsymbol{m}}^{-1}\right)^{-1} \left(V^{-1} \sum_{i=1}^{m^*} \boldsymbol{\mu}_{L_i^*} + B_m^{-1} \boldsymbol{a}_m \right), \left(m^* V^{-1} + B_{\boldsymbol{m}}^{-1}\right)^{-1} \right) \end{aligned}$$

• Updating V:

$$\begin{aligned} p(V \mid \ldots, D) &\propto \left\{ \prod_{i=1}^{m^*} \mathrm{N}_3(\boldsymbol{\mu}_{L_i^*}; \boldsymbol{m}, V) \right\} \mathrm{IW}_3\left(a_V, B_V\right) \\ &\sim \mathrm{IW}_3\left(a_V + m^*, B_V + \sum_{i=1}^{m^*} \left(\boldsymbol{\mu}_{L_i^*} - \boldsymbol{m}\right) \left(\boldsymbol{\mu}_{L_i^*} - \boldsymbol{m}\right)^T \right) \end{aligned}$$

• Updating S:

$$\begin{aligned} p(S \mid \ldots, D) &\propto \left\{ \prod_{i=1}^{m^*} \mathrm{IW}_3(\Sigma_{L_i^*}; \nu, S) \right\} \mathrm{W}_3\left(a_S, B_S\right) \\ &\sim \mathrm{W}_3\left(a_S + m^* \nu, \left(B_S^{-1} + \sum_{i=1}^{m^*} \Sigma_{L_i^*}^{-1} \right)^{-1} \right) \end{aligned}$$

Bibliography

Antoniak, C. E. (1974), "Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems," *The Annals of Statistics*, 2, 1152–1174.

Bacour, C., Baret, F., Beal, D., Weiss, M., and Pavageau, K. (2006), "Neural network estimation of LAI, fAPAR, fCover and LAIxCab, from top of canopy MERIS reflectance data: Principles and validation," *Remote Sensing of Environment*, 105, 313–325.

Ballarini, F., Battistoni, G., Brugger, M., Campanella, M., Carboni, M., Cerutti, F., Empl, A., Fass, A., Ferrari, A., Gadioli, E., Garzelli, M., Lantz, M., Mairani, A., Mostacci, A., Muraro, S., Ottolenghi, A., Patera, V., Pelliccioni, M., Pinsky, L., Ranft, J., Roesler, S., Sala, P., Scannicchio, D., Smirnov, G., Sommerer, F., Trovati, S., Villari, R., Vlachoudis, V., Wilson, T., and Zapp, N. (2007), "The physics of the FLUKA code: Recent developments," *Advances in Space Research*, 40, 1339–1349.

Bayarri, M., Berger, J., Paulo, R., Sacks, J., Cafeo, J., Cavendish, J., Lin, C., and Tu, J. (2007a), "A framework for validation of computer models," *Technometrics*, 49, 138–154.

Bayarri, M. J., Berger, J. O., Cafeo, J., Garcia-Donato, G., Liu, F., Palomo, J., Parthasarathy, R., Paulo, R., Sacks, J., and Walsh, D. (2007b), "Computer model validation with functional output," *Annals of Statistics*, 35, 1874–1906.

Bayarri, M. J., Berger, J. O., Kennedy, M., Kottas, A., Paulo, R., Sacks, J., Cafeo, J. A., Lin, C. H., and Tu, J. (2009), "Predicting vehicle crashworthiness: validation of computer models for functional and hierarchical data," *Journal of the American Statistical Association*, 929–943.

Boys, R., Wilkinson, D., and Kirkwood, T. (2008), "Bayesian inference for a discretely observed stochastic kinetic model," *Statistics and Computing*, 18, 125–135.

Connor, R. J. and Mosimann, J. E. (1969), "Concepts of independence for proportions with a generalization of the Dirichlet distribution," *Journal of the American Statistical Association*, 64, 194–206.

Cowles, K., Kass, R., and O'Hagan, T. (2009), "What is Bayesian Analysis?" *http://bayesian.org/Bayes-Explained*, retrieved 4 Aug, 2011.

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, 717–729.

Damien, P., Wakefield, J., and Walker, S. (1999), "Gibbs sampling for Bayesian non-conjugate and hierarchical models by using auxiliary variables," *Journal of the Royal Statistical Society. Series B*, 61, 331–344.

Dunson, D. B. (2005), "Bayesian semiparametric isotonic regression for count data," *Journal of the American Statistical Association*, 100, 618–627.

Escobar, M. D. and West, M. (1995), "Bayesian density estimation and inference using mixtures," *Journal of the American Statistical Association*, 90, 577–588.

Ferguson, T. S. (1973), "A Bayesian analysis of some nonparametric problems," *The Annals of Statistics*, 1, 209–230.

Ferrari, A., Sala, P., Fassò, A., and Ranft, J. (2005), "FLUKA: A multi-particle transport code," Tech. rep., CERN 2005-10, INFN/TC-05/11, SLAC-R-773.

Foster, C. C., O'Neill, P. M., and Kouba, C. (2006), "Monte Carlo simulation of proton upsets in Xilinx Virtex-II FPGA using a position dependent Q(crit) with PROPSET," *IEEE Transactions on Nuclear Science*, 53, 3494–3501.

Ganapol, B., Johnson, L., Hlavka, C., Peterson, D., and Bond, B. (1999), "LCM2: A coupled leaf/canopy radiative transfer model." *Remote Sensing of the Environment*, 70, 153–166.

Ganapol, B. D., Johnson, L. F., Hammer, P. D., Hlavka, C. A., and Peterson, D. L. (1998), "LEAFMOD: A New Within-Leaf Radiative Transfer Model," *Remote Sensing of Environment*, 63, 182–193.

Gelfand, A. E. and Kottas, A. (2003), "Bayesian semiparametric regression for median residual life," *Scandinavian Journal of Statistics*, 30, 651–665.

Gelfand, A. E. and Smith, A. F. M. (1990), "Sampling-based approaches to calculating marginal densities," *Journal of the American Statistical Association*, 85, 398–409.

Gelman, A., Roberts, G., and Gilks, W. (1996), "Efficient Metropolis jumping rules," in *Bayesian Statistics 5*, eds. Bernardo, J. M., Berger, J. O., Dawid, A. P., and Smith, A. F. M., Oxford University Press, pp. 599–607.

Gelman, A. and Rubin, D. B. (1992), "Inference from Iterative Simulation Using Multiple Sequences," *Statistical Science*, 7, 457–472.

Geman, S. and Geman, D. (1984), "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6, 721–741.

Geweke, J. (1992), "Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments," in *Bayesian Statistics*, eds. Bernardo, J. M., Berger, J. O., Dawid, A. P., and Smith, A. F. M., Oxford University Press, pp. 169–193.

Goldstein, M. and Rougier, J. C. (2006), "Bayes linear calibrated prediction for complex systems," *Journal of the American Statistical Association*, 101, 1132–1143.

Golightly, A. and Wilkinson, D. (2006), "Bayesian sequential inference for stochastic kinetic biochemical network models," *Journal of Computational Biology*, 13, 838–851.

— (2008), "Bayesian inference for nonlinear multivariate diffusion models observed with error," *Computational Statistics and Data Analysis*, 52, 1674–1693.

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, 1119–1130.

— (2010), "Optimization under unknown constraints," in *Bayesian Statistics 9*, eds. Bernardo, J. M., Bayarri, M. J., Berger, J. O., Dawid, A. P., Heckerman, D., Smith, A. F. M., and West, M., Oxford University Press, pp. 229–256.

— (2011), "Cases for the nugget in modeling computer experiments," *Statistics and Computing*, to appear.

Hastings, W. K. (1970), "Monte Carlo sampling methods using Markov Chains and their applications," *Biometrika*, 57, 97–109.

Henderson, D. A., Boys, R. J., Krishnan, K. J., Lawless, C., and Wilkinson, D. J. (2009), "Bayesian emulation and calibration of a stochastic computer model of mitochondrial DNA deletions in substantia nigra neurons," *Journal of the American Statistical Association*, 104, 76–87.

Henderson, D. A., Boys, R. J., and Wilkinson, D. J. (2010), "Bayesian Calibration of a Stochastic Kinetic Computer Model Using Multiple Data Sources," *Biometrics*, 66, 249–256.

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

Hirano, K. (2002), "Semiparametric Bayesian inference in autoregressive panel data models," *Econometrica*, 70, 781–799.

Homma, T. and Saltelli, A. (1996), "Importance measures in global sensitivity analysis of nonlinear models," *Reliability Engineering and System Safety*, 52, 1–17.

Hosgood, B., Jacquemoud, S., Andreoli, G., Verdebout, J., Pedrini, G., and Schmuck, G. (1995), "Leaf optical properties experiment (LOPEX93)," Tech. Rep. EUR16095EN, European Commission, Joint Research Centre, Institute for Remote Sensing Applications.

Houborg, R., Soegaard, H., and Boegh, E. (2007), "Combining vegetation index and model inversion methods for the extraction of key vegetation biophysical parameters using Terra and Aqua MODIS reflectance data," *Remote Sensing of Environment*, 106, 39–58.

Iooss, B., Ribatet, M., and Marrel, A. (2009), "Global sensitivity analysis of stochastic computer models with joint metamodels." *http://arxiv.org/abs/0802.0443*, retrieved 4 Aug, 2011.

Ishwaran, H. and James, L. F. (2001), "Gibbs sampling methods for stick-breaking priors," *Journal of the American Statistical Association*, 96, 161–173.

Justice, C., Vermote, E., Townshend, J., Defries, R., Roy, D., Hall, D., Salomonson, V., Privette, J., Riggs, G., Strahler, A., Lucht, W., Myneni, R., Knyazikhin, Y., Running, S., Nemani, R., Wan, Z., Huete, A., van Leeuwen, W., Wolfe, R., Giglio, L., Muller, J., Lewis, P., and Barnsley, M. (1998), "The Moderate Resolution Imaging Spectroradiometer (MODIS): land remote sensing for global change research," *IEEE Transactions on Geoscience and Remote Sensing*, 36, 1228–1249.

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

Kottas, A. and Krnjajić, M. (2009), "Bayesian semiparametric modelling in quantile regression," *Scandinavian Journal of Statistics*, 36, 297–319.

Kuo, L. and Mallick, B. (1997), "Bayesian semiparametric inference for the accelerated failure time model," *Canadian Journal of Statistics*, 25, 457–472.

Liu, J. S. (1996), "Nonparametric hierarchical Bayes via sequential imputations," *The Annals of Statistics*, 24, 911–930.

Mackay, D. (1998), "Introduction to Gaussian processes," *NATO ASI Series. Neural Networks and Machine Learning*, 168, 133–166.

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

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953), "Equations of state calculations by fast computing machines," *Journal of Chemical Physics*, 21, 1087–1091.

Morris, R. D., Kottas, A., Taddy, M., Furfaro, R., and Ganapol, B. D. (2008), "A Statistical framework for the sensitivity analysis of radiative transfer models," *IEEE transactions on Geoscience and Remote Sensing*, 46, 4062–4074.

Müller, P., Erkanli, A., and West, W. (1996), "Bayesian curve fitting using multivariate normal mixtures," *Biometrika*, 83, 67–79.

Müller, P. and Quintana, F. A. (2004), "Nonparametric Bayesian data analysis," *Statistical Science*, 19, 95–110.

Müller, P., West, M., and Maceachern, S. (1997), "Bayesian models for non-linear auto-regressions," *Journal of Time Series Analysis*, 18, 593–614.

Neal, R. M. (1998), "Regression and classification using Gaussian process priors," in *Bayesian Statistics 6*, eds. Bernando, J., Berger, J., Dawid, A., and Smith, A., Oxford University Press, pp. 475–501.

— (2003), "Slice sampling (with discussion)," *Annals of Statistics*, 31, 705–767.

Oakley, J. and O'Hagan, A. (2002), "Bayesian inference for the uncertainty distribution of computer model outputs," *Biometrika*, 89, 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*, 66, 751–769.

Petersen, E. (2008), "Soft error results analysis and error rate prediction," *IEEE Nuclear Space Radiation Effects Conference Short Course*.

Petersen, E., Pickel, J., Adams, J.H., J., and Smith, E. (1992), "Rate prediction for single event effects-a critique," *IEEE Transactions on Nuclear Science*, 39, 1577–1599.

Petersen, E. L. (1996), "Cross section measurements and upset rate calculations," *IEEE Transactions on Nuclear Science*, 43, 2805–2813.

Raftery, A. E. and Lewis, S. M. (1992), "Comment: One long run with diagnostics: Implementation strategies for Markov Chain Monte Carlo," *Statistical Science*, 7, 493–497.

Rasmussen, C. E. and Williams, C. K. I. (2006), *Gaussian processes for machine learning*, MIT Press.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989), "Design and analysis of computer experiments (C/R: p423-435)," *Statistical Science*, 4, 409–423.

Saltelli, A. (2002), "Making best use of model evaluations to compute sensitivity indices," *Computer Physics Communications*, 45, 280–297.

Saltelli, A., Chan, k., and Scott, E. M., e. (2000), *Sensitivity analysis*, John Wiley and Sons.

Santner, T., Williams, B., and Notz, W. (2003), *The design and analysis of computer experiments*, Springer-Verlag.

Schwank, J. R., Shaneyfelt, M. R., and E., D. P. (2008), "Radiation hardness assurance testing of microelectronic devices and integrated circuits: Test guideline for proton and heavy ion single-event effects," Tech. rep., Sandia National Laboratories.

Sethuraman, J. (1994), "A constructive definition of Dirichlet priors," *Statistica Sinica*, 4, 639–650.

Sobol, I. M. (1993), "Sensitivity estimates for non linear mathematical models," *Mathematical Modelling and Computational Experiments*, 1, 407–414.

Taddy, M., Lee, H. K. H., Gray, G. A., and Griffin, J. (2009), "Bayesian guided pattern search for robust local optimization," *Technometrics*, 51, 389–401.

Tylka, A., Adams, J., Boberg, P., Brownstein, B., Dietrich, W., Flueckiger, E., Petersen, E., Shea, M., Smart, D., and Smith, E. (1997), "CREME96: A revision of the cosmic ray effects on micro-electronics Code," *IEEE Transactions on Nuclear Science*, 44, 2150–2160.

West, M., Müller, P., and Escobar, M. (1994), *Hierarchical priors and mixture models with applications in regression and density estimation*, John Wiley.

Wilkinson, D. J. (2009), "Stochastic modelling for quantitative description of heterogeneous biological systems," *Nature Reviews Genetics*, 10, 122–133.

Xapsos, M., Weatherford, T., and Shapiro, P. (1993), "The shape of heavy ion upset cross section curves," *IEEE Transactions on Nuclear Science*, 40, 1812–1819.