Computer model calibration as a method of design

Carl Ehrett

April 19, 2018

1 Introduction

1.1 Computer experiments

Suppose that one wishes to improve one's understanding of, say, the movement of people in a crowd escaping from a building in a crisis situation. This is an example of an area in which field data are extremely difficult to acquire. Merely assembling a crowd of research subjects in one place is costly and difficult. Asking them to flee a building may result in behaviors which are unlike those in real crisis situations – but which may nonetheless present unacceptable physical risk to the subjects. Inducing them to flee through the generation of a (real or apparent) crisis is similarly infeasible. Observational data are likewise scarce here, since panic-inducing crises are by their nature difficult to predict and chaotic in ways that hinder the reliable collection of data.

In the face of these difficulties, computer models offer an alternative to the choice between attempting field data collection and giving up on the hope of progress. Using existing theory concerning human psychology and movement, it is possible to construct a computer model simulating the behavior of people evacuating from a large building. For example, the SIMULEX model described by Thompson & Marchant (1995) allows one to observe simulated evacuation behaviors in any specified building layout, using any desired physical distribution of individuals, whose individual relevant characteristics (walking speed, initial bodily orientation, etc) may be controlled by the researcher. Thus, computer models provide a means to collect data which might otherwise be largely inaccessible.

The study of computer models from a statistical perspective calls for specialized tools and techniques. Gaussian processes (GPs) are a popular tool for modeling the output of computer code. There are three reasons for this popularity: (1) The use of a GP does not require detailed foreknowledge of the approximate parametric form of the computer model. Researchers often lack such foreknowledge in the case of complex computer models. (2) GPs easily interpolate the observed data. This is an advantage when the observations come from deterministic computer code that is free of observation error. (3) The variance of a GP provides a natural form of uncertainty quantification. A Bayesian approach to the study of computer models is undertaken by Currin *et al.* (1991); the authors approach GPs as prior distributions on the unknown form of the computer model. A frequentist applications of GPs to computer models is provided by Sacks *et al.* (1989), who use GPs not only for estimating uncertainty but also as the basis for their approach to experimental design in the area of deterministic computer models. Santner *et al.* (2003) offer a comprehensive discussion of to the use of GPs for prediction, design, and sensitivity analysis with respect to computer experiments from both frequentist and Bayesian perspectives. It is against the background of these works that the past two decades of research into computer model calibration takes place.

1.2 Computer model calibration

Suppose that we wish to use the SIMULEX model to compare two different proposed building codes to be enforced in, say, St. Louis, Missouri. We may use average walking speed and average interpersonal distance as input parameters for this model, both to settle the initial physical distribution of people throughout the building and to influence their behavior during evacuation. It is well-established that average walking speed (Bornstein & Bornstein, 1976) and interpersonal distance (Sorokowska *et al.*, 2017) vary across locales. These values may be unknown for the case of St. Louis. Thus we may wish to find the true values for average walking speed and interpersonal distance in St. Louis; we may wish, in other words, to *calibrate* these parameters in the model.

Broadly, in model calibration, we may consider a model to be of the form $\eta(x, \theta)$, where (x, θ) comprise all inputs to the model. Control inputs — inputs under the control of the researcher (in the evacuation example, this would include the building layout) comprise x, whereas θ is the set of calibration inputs — parameters the values of which are not under researcher control, but rather are unknown values which must be estimated for successful simulation. Thus where f describes the true system, we consider the model to be

$$f(x, \theta) = f(x) = \eta(x, \theta) + \delta(x) \tag{1}$$

where δ describes the model discrepancy – i.e., the bias of the model as an estimate of the real system. Notice that we may write $f(x) = f(x, \theta)$ since θ does not vary in reality. To undertake model calibration, we must have access to at least some observations of the real system; it is to these real observations that we calibrate the computer model.

Much interest in the past two decades has centered on Bayesian methods for model calibration. The appeal of a Bayesian approach to model calibration lies in the fact that the calibration parameters are a source of uncertainty for the model. This uncertainty should be quantified so that its effect on the model can be made explicit. We can thus use Bayesian methods to arrive at a posterior distribution on the calibration parameters which balances our prior knowledge about the calibration parameters with what can be learned from the available data, and which also allows for accurate uncertainty quantification on the model outputs.

The work of Kennedy & O'Hagan (2001) has been influential in this area. Kennedy and O'Hagan offer a Bayesian approach to computer model calibration that allows for the uncertainty of the calibration parameters in the predictions of the resulting calibrated model. This area is furthered by Higdon *et al.* (2004), who develop an approach that undertakes model calibration with quantification of the related uncertainty, as well as explicitly incorporating uncertainty regarding the computer model output, the bias of the computer model, and uncertainty due to observation variance (of field data). That approach is further refined and exemplified by Williams *et al.* (2006). Loeppky *et al.* (2006) offer an MLE-based alternative to the Bayesian approach promulgated by Kennedy and O'Hagan, intending thereby to improve the identifiability of the calibration parameters in the face of model discrepancy. Bayarri *et al.* (2007b) extends the approach of Kennedy and O'Hagan, providing a framework that is intended to allow for simultaneous validation and calibration of a computer model (using the same training data). Though they work within the Bayesian framework set by Kennedy and O'Hagan, they mitigate the integrity of the Bayesian analysis through what they call "modularization". Modularization refers to separating sources of information, so that the model has distinct components or "modules", rather than allowing all information to combine into a single analysis under the umbrella of Bayes' theorem. Liu *et al.* (2009) focus directly on the use of modularization, exploring its advantages and disadvantages; amongst other potential motivations, they show that modularization can improve the identifiability of calibration parameters. Bayarri *et al.* (2007a) furthers the project of Bayarri *et al.* (2007b) by applying the latter's methodology to functional data. To do so, the methodology for dealing with scalar data is hierarchically applied to the coefficients of a wavelet representation of the functional data. Similarly, Paulo *et al.* (2012) focuses on applying the lessons of Bayarri *et al.* (2007b) to computer models with multivariate output.

Above, I have described model calibration as a matter of estimating the true values of unknown parameters. Indeed, that is the sort of model calibration that will form the focus on the present work. However, in a broad sense of the term, model calibration takes a second form as well. For convenience, we may distinguish this second form by referring to it as *tuning*, rather than as calibration. In model tuning, a computer model includes inputs which must be calibrated, but which have no particular interpretation; i.e., tuning parameters do not represent the truth regarding some value of the system being modeled. The work of Brynjarsdóttir & O'Hagan (2014) emphasizes the distinction between calibration and tuning. They show that whereas a well-modeled discrepancy function ($\delta(\cdot)$ in Equation 1) may not be necessary for tuning, since the tuning process can accomplish much of the work of a discrepancy function, an accurate discrepancy function is crucial for calibration, for the same reason. Letting the calibration process take on the duties of the discrepancy function reduces the identifiability of the calibration parameters.

Another extension of the framework of Kennedy & O'Hagan (2001) and Bayarri *et al.* (2007b) is the *state-aware* calibration of Brown & Atamturktur (2018). When one knows or suspects that the appropriate value of a calibration parameter θ is dependent upon the other inputs x, state-aware calibration allows for the calibration procedure to estimate θ as $\theta(x)$. This may obviate the need for a discrepancy function, either because the true value of the calibration parameter does indeed vary with the other inputs, or (in a tuning problem) because varying the tuning parameter can accomplish the work of a discrepancy function in aligning the computer model response with reality.

1.2.1 Gaussian processes

In principle, model calibration[1] need not rely on a GP emulator, or any other sort of emulator; one could (e.g.) complete a full Bayesian analysis via an MCMC chain that involves running the relevant computer model at each iteration of the chain. However, computer models are frequently too computationally expensive to allow for such profligacy. Instead, a computationally tractable emulator can be constructed using a sample of observations from the computer model. As described in Section 1.1, GPs are popular prior distributions on computer model output, because (1) their use does not require detailed foreknowledge of the model function's parametric form, (2) GPs easily interpolate the (deterministic) computer model output, and (3) GPs facilitate uncertainty quantification through the variance of the posterior GP. In this section, I provide brief background on Gaussian processes and their use in regression broadly and in computer model calibration specifically.

Gaussian processes can be thought of as generalizations of multivariate normal random variables. Whereas a multivariate normal random variable is a random vector of finite length, a Gaussian process can be thought of as a random function. The value of the function at any finite selection of points is itself a multivariate normal random variable. Just as a multivariate random variable is characterized by its mean vector and covariance matrix, a Gaussian process is fully characterized by its mean function $\mu : D \to \mathbb{R}$ and covariance function $C : D \times D \to \mathbb{R}$, where D is the domain of the process. Thus for any points x, y in the domain of the Gaussian process, $\mu(x)$ gives the mean of the Gaussian process at x, and $C(x, y)$ gives the covariance between the values of the Gaussian process at points x and y. As a special case, then, $C(x, x)$ gives the marginal variance of the GP at x.

O'Hagan (1978) suggested the use of GPs as priors for modeling unknown functions. Given observations $\mathbf{f}$ from the function f being modeled, an updated mean and covariance function $\mu^*(x) = \mu(x)|\mathbf{f}$ and $C^*(x, y) = C(x, y)|\mathbf{f}$ describe a new GP that has been "trained" on the observations $\mathbf{f}$. O'Hagan did not focus in that work on applications to computer models. The use of GPs to produce a computationally efficient predictor $\hat{\eta}(x)$ of expensive computer code $\eta(x)$ given observations of code output at $\mathbf{x} = \{x_i\}_{i=1}^n$ is promulgated by Sacks *et al.* (1989) and explored at length by Santner *et al.* (2003). Since computer code is typically deterministic, these applications differ from O'Hagan's focus in that the updated GP is induced to interpolate the observations $\boldsymbol{\eta} = (\eta(x_1), \cdots, \eta(x_n))^T$. The updated mean allows for prediction of the computer model at points not observed, with uncertainty quantified by the updated covariance function.

Suppose, for example, that we begin with a prior GP with constant mean function $\mu(x) = 0\ \forall x$. Notice that we may use our covariance function C to define an $n \times n$ matrix $\mathbf{C}_{\mathbf{x},\mathbf{x}}$ such that the i, j entry of $\mathbf{C}_{\mathbf{x},\mathbf{x}}$ is equal to $C(x_i, x_j)$. We may wish to train our GP on these n observations, and then examine the resulting posterior GP at the points $\mathbf{x}' = \{x'_i\}_{i=1}^m$. Recall that the GP at the points $\mathbf{x}'$ is a multivariate normal random variable of length m, which is fully characterized by its mean vector $\mu^*_{\mathbf{x}'}$ and covariance matrix $\mathbf{C}^*_{\mathbf{x}',\mathbf{x}'}$. We can find these as:

$$
\begin{aligned}
\mu^*_{\mathbf{x}'} &= \mathbf{C}_{\mathbf{x}',\mathbf{x}} \cdot \mathbf{C}^{-1}_{\mathbf{x},\mathbf{x}} \cdot \boldsymbol{\eta} \\
\mathbf{C}^*_{\mathbf{x}'} &= \mathbf{C}_{\mathbf{x}',\mathbf{x}'} - \mathbf{C}_{\mathbf{x}',\mathbf{x}} \cdot \mathbf{C}^{-1}_{\mathbf{x},\mathbf{x}} \cdot \mathbf{C}_{\mathbf{x},\mathbf{x}'}
\end{aligned}
\tag{2}
$$

A visualization of this example can be found in Figure 1. Here, $n = 5$, and $x' = (-5, -4.75, , 4.75, 5)$.

Though the approach of O'Hagan (1978) was Bayesian, it did not provide for Bayesian means of selecting a covariance function C. Neal (1998) extends the work of O'Hagan in this direction. Savitsky *et al.* (2011) describes a method for variable selection in GP regression via spike-and-slab mixture priors on the parameters of the covariance function. Shang & Chan (2013) provide an umbrella generalization of existing methods of GP regression under a framework of what they call generalized GP models. All of these works treat GP regression generally, and do not focus specifically on the use of GPs to build emulators for computer model.

[1]In this work, "calibration" will usually be used broadly to include model tuning; where it is used in such a way as to exclude tuning, this will be made clear in context.

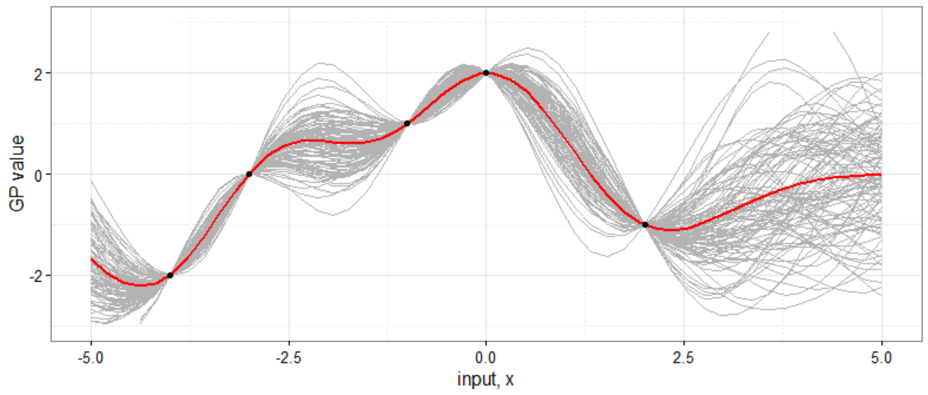

Figure 1: Example of a Gaussian process trained to interpolate five data points (black dots). The posterior mean function is shown in red; the gray lines are 50 draws from the posterior GP.

By contrast, much recent attention following the work of Santner *et al.* (2003) and Sacks *et al.* (1989) has focused specifically on the use of GPs for computer model emulation. Kennedy & O'Hagan (2001) themselves develop an influential framework for Bayesian analysis using GPs specifically for computer model calibration. Kennedy *et al.* (2006) showcase this use of GP emulators for uncertainty and sensitivity analyses. Bastos & O'Hagan (2009) describes both numerical and graphical diagnostic techniques for assessing when a GP emulator of a computer model is successful, as well as discussion of likely causes of poor diagnostic results. While most work in the area of GP emulation uses stationary covariance functions (in which $C(x, y) = C(|x-y|)$ depends only on the distance between x and y, and not on their location in the input domain) and quantitative inputs, efforts have been made to branch away from these core uses. Gramacy & Lee (2008) use treed partitioning to deal with a nonstationary computer model. Qian *et al.* (2008) explore methods for using GP emulators that include both quantitative and qualitative inputs.

Aside from Kennedy & O'Hagan (2001), recent applications of GP emulation specifically to problems of calibration have focused largely on the works of Williams *et al.* (2006) and Bayarri *et al.* (2007b). It will be helpful here to provide an illustrative summary of the approach taken by Williams *et al.* (2006), both to exemplify the use of GPs for computer model calibration and because the approach utilized in the present work closely follows theirs.

Consider that we have inputs $x \in \mathbb{R}^p$ and $t \in \mathbb{R}^q$ scaled to the unit hypercube, and observations

$$y(x_i) = f(x_i) + \epsilon(x_i), \quad i = 1, \cdots, n, \tag{3}$$

where $f(\cdot)$ is the true system and $\epsilon(x_i)$ is known measurement error. Then by Equation 1 we have

$$y(x_i) = \eta(x_i, \theta) + \delta(x_i) + \epsilon(x_i), \quad i = 1, \cdots, n \tag{4}$$

where $\eta(\cdot, \cdot)$ is the computer model, θ is the best[2] setting of the vector of calibration parameters, and $\delta(\cdot)$ is the discrepancy function describing the bias of $\eta(\cdot, \cdot)$ as an estimate of $f(\cdot)$.

Williams *et al.* define the GP prior for modeling $\eta(\cdot, \cdot)$ as follows. Let the mean function $\mu(x, t) = c$, c a constant. Set the covariance function to have the (stationary) form

$$C((x,t),(x',t')) = \frac{1}{\lambda_\eta} \prod_{k=1}^{p} \exp\left(-\beta_k^\eta |x_k - x_k'|^{\alpha_\eta}\right) \times \prod_{k=p+1}^{p+q} \exp\left(-\beta_k^\eta |t_k - t_k'|^{\alpha_\eta}\right) \tag{5}$$

where $\lambda_e ta$ is the marginal precision of the GP; each β_k describes the strength of the GP's dependence on one of the elements of the input vectors x, t; and α_η describes the smoothness of the GP.

[2]In the case of calibration, the "best" setting will be the true setting of that parameter; in a case of tuning rather than calibration, the "best" setting would instead be the optimal setting for minimizing model bias.

The authors place the following priors on the hyperparameter:

$$\pi(c) \sim N(0, v) \tag{6}$$

$$\pi(\lambda_\eta) \sim \text{Gamma}(5,5), \quad \lambda_\eta > 0 \tag{7}$$

$$\pi\left(\rho_k^\eta\right) \sim \text{Beta}(1, 0.1), \quad k = 1, \cdots, p+q \tag{8}$$

where $\rho_k^\eta = \exp(-\beta_k^\eta)/4)\ \forall k$. The parameters of the Gamma and Beta distributions are chosen to encourage λ_η to be close to one, and β_k to be low for all k (encouraging strong dependence; i.e., we antecedently expect each of our inputs to be influential). Furthermore, the authors let $v \to 0$, i.e., the GP is assumed to have constant mean $c = 0$.

The authors similarly model the discrepancy term as a GP, also with mean zero, and covariance function

$$C_\delta(x, x') = \frac{1}{\lambda_\delta} \prod_{k=1}^{p} \exp\left(-\beta_k^\delta |x_k - x_k'|^{\alpha_\delta}\right), \tag{9}$$

with priors

$$\pi(\lambda_\delta) \sim \text{Gamma}(a_\delta, b_\delta) \tag{10}$$

$$\pi(\rho_k^\delta) \sim \text{Beta}(1, 0.3). \tag{11}$$

where $\rho_k^\delta = \exp(-\beta_k^\delta/4)\ \forall k$.

Then where $\boldsymbol{\eta} = (\eta(x_1,t_1), \cdots, \eta(x_n,t_n))^T$ are the simulation observations, $\mathbf{y} = (y(x_{n+1}), \cdots, y(x_{n+m}))^T = (y(x_{n+1},\theta), \cdots, y(x_{n+m},\theta))^T$ are the field observations, $\mathcal{D} = (\mathbf{y}^T \boldsymbol{\eta}^T)^T$, $\boldsymbol{\beta}^\eta = (\beta_1^\eta, \cdots, \beta_{p+q}^\eta)^T$, and $\boldsymbol{\beta}^\delta = (\beta_1^\delta, \cdots, \beta_{p+q}^\delta)^T$, we have the distribution of $\mathcal{D}$ as

$$\mathcal{D}|\theta, c, \lambda_\eta, \beta^\eta, \lambda_\delta, \beta^\delta, \mathbf{C}_y \sim N(c \cdot \mathbf{1}_{n+m}, \mathbf{C}_\mathcal{D}) \tag{12}$$

where $\mathbf{C}_y$ an $m \times m$ matrix in which the i, j entry is the (known) observation variance $C_{obs}(x_i, x_j)$ for $n < i, j \le n+m$, and $\mathbf{C}_\mathcal{D}$ is a matrix with its i, j entry equal to

$$C((x_i, t_i), (x_j, t_j)) + I(i, j > n) \cdot (C_{obs}(x_i, x_j) + C_\delta(x_i, x_j)) \tag{13}$$

Thus, the joint posterior density given by the model is

$$\pi(\theta, c, \lambda_\eta, \rho^\eta, \lambda_\delta, \rho^\delta|\mathcal{D}) \propto \pi(\mathcal{D}|\theta, c, \lambda_\eta, \beta^\eta, \lambda_\delta, \beta^\delta, \mathbf{C}_y) \times \pi(c) \times \pi(\lambda_\eta) \times \pi(\rho^\eta) \times \pi(\lambda_\delta) \times \pi(\rho^\delta) \tag{14}$$

Note that where a discrepancy function is not included in the model and the mean c is treated as a constant, (14) simplifies greatly; where furthermore λ_η and ρ^η are estimated via maximum likelihood (as in Kennedy & O'Hagan (2001)), (14) simplifies down merely to $\pi(\mathcal{D}|\theta, c, \lambda_\eta, \beta^\eta, \lambda_\delta, \beta^\delta, \mathbf{C}_y)$. Markov chain Monte Carlo methods are useful for evaluating (14); the next section takes up this topic.

1.2.2 Markov chain Monte Carlo methods

The central idea of Markov chain Monte Carlo (MCMC) integration is to construct a Markov chain which has as its equilibrium distribution the target distribution one wishes to explore. The Markov chain is observed, and beyond an initial "burn-in" period during which the chain is allowed to approach its equilibrium distribution, samples are considered to be drawn approximately from the target distribution.

For example, consider a model with full distribution given by (14), but where discrepancy is not included and λ_η, ρ^η are found via maximum likelihood estimation. Then the full distribution is given by $\pi(\theta, c, \lambda_\eta, \rho^\eta, \lambda_\delta, \rho^\delta|\mathcal{D}) \propto \pi(\mathcal{D}|\theta, c, \lambda_\eta, \beta^\eta, \lambda_\delta, \beta^\delta, \mathbf{C}_y) \times \pi(c)$. A simple means of exploring this distribution via MCMC would begin with an initial guess $\theta^{(1)}, c^{(1)}$. At the i^{th} step for $i = 2, \cdots$, using a proposal distribution $\pi_{prop}(\cdot, \cdot|\theta^{(i-1)}, c^{(i-1)})$ from which we may easily sample directly, we draw a new "proposed" sample θ^*, c^*. We then with probability α accept this new proposed sample, setting $(\theta^{(i)}, c^{(i)}) = (\theta^*, c^*)$; otherwise we reject the proposed sample and let $(\theta^{(i)}, c^{(i)}) = (\theta^{(i-1)}, c^{(i-1)})$. In doing so, we have

$$\alpha = \frac{\pi(\theta^*, c^*|\mathcal{D})}{\pi(\theta^{(i-1)}, c^{(i-1)}|\mathcal{D})} = \frac{\pi(\mathcal{D}|\theta^*, c^*, \lambda_\eta, \beta^\eta, \lambda_\delta, \beta^\delta, \mathbf{C}_y) \times \pi(c^*)}{\pi(\mathcal{D}|\theta^{(i-1)}, c^{(i-1)}, \lambda_\eta, \beta^\eta, \lambda_\delta, \beta^\delta, \mathbf{C}_y) \times \pi(c^{(i-1)})} \tag{15}$$

Note that in order to proceed in this way, it is necessary that π_{prop} be a symmetric distribution. The version of MCMC described in this example is known as the Metropolis-Hastings algorithm; it was initially described by Metropolis *et al.* (1953) and generalized by Hastings (1970). A thorough exposition of the technique, its theoretical foundation, and its relation to other varieties of MCMC is provided by Chib & Greenberg (1995).

A related variant of MCMC – in fact, a special case of Metropolis-Hastings, as shown by Gelman (1992) – is Gibbs sampling, initially described by Geman & Geman (1984). In using Gibbs sampling, one finds the conditional distribution of each of the parameters one wishes to sample, using the most recent samples of all other parameters. For example, we can convert the above Metropolis-Hastings illustration to a simple case of Gibbs sampling as follows. Begin as before, with initial guesses $\theta^{(1)}$ and $c^{(1)}$. Thereafter, sample θ and c not together, but rather in alternating draws. That is, at the i^{th} step for $i = 2, \cdots$, draw $\theta^{(i)}$ conditional on $c = c^{(i-1)}$, then draw $c^{(i)}$ conditional on $\theta = \theta^{(i)}$. Thus, for example, when drawing $c^{(i)}$, its conditional distribution would be proportional to $\pi(\mathcal{D}|\theta^{(i)}, c^*, \lambda_\eta, \beta^\eta, \lambda_\delta, \beta^\delta, \mathbf{C}_y) \times \pi(c^*)$. If the conditional distribution is something from which we may easily draw directly, then we can draw $c^{(i)}$ that way; otherwise, we can use a so-called Metropolis-within-Gibbs scheme, and draw $c^{(i)}$ similarly to what was described above for the Metropolis-Hastings algorithm: draw a proposed c^* from a proposal distribution, find the appropriate ratio α_c, and accept or reject c^* accordingly. In the application considered in the present work, Metropolis-within-Gibbs will be used.

Recall that a symmetric proposal distribution is required in order for the Metropolis-Hastings algorithm to proceed as described in the above illustration of that technique. However, the algorithm can accommodate an asymmetric proposal distribution with only slight complication. A frequent reason for using an asymmetric proposal distribution is the existence of boundary constraints on the parameter being sampled. Thus, for example, in the application considered in the present work, one of the calibration parameters is the thickness (in mm) of the material forming a wind turbine blade. Expert opinion was used to set the support of this parameter to be the range [10mm, 25mm]. Good mixing in the MCMC chain is promoted by choosing a proposal density such that the distribution $\pi_{prop}(\cdot|\tau)$ has mean τ. With boundary constraints, this requires an asymmetric distribution. When $\pi_{prop}(\cdot|\tau)$ is asymmetric, this changes the above illustration only by way of requiring us to calculate the acceptance probability α as follows:

$$\alpha = \frac{\pi(\theta^*, c^*|\mathcal{D})}{\pi(\theta^{(i-1)}, c^{(i-1)}|\mathcal{D})} \times \frac{\pi_{prop}(\theta^{(i-1)}, c^{(i-1)}|\theta^*, c^*)}{\pi_{prop}(\theta^*, c^*|\theta^{(i-1)}, c^{(i-1)})}. \tag{16}$$

1.2.3 Computational difficulties

Note that the joint posterior density given in (14) above includes $\mathbf{C}_\mathcal{D}$, which, depending on how many field and simulation observations one has, may be of fearsomely high dimension. This can lead to computational difficulties in calculating α in the course of the MCMC routine. Difficulties arise in two ways. Firstly, the likelihood given in (14) evaluated at a given point can be so small as to be vulnerable to significant round-off error. Secondly, the poor conditioning of $\mathbf{C}_{x,x}$ in (2) can make it difficult to invert and find the determinant of this matrix, as must be done in the course of the MCMC to find the relevant likelihoods. The latter problem can be alleviated by adding a small nugget to $\mathbf{C}_{x,x}$; i.e., (abusing notation somewhat) we can set $\mathbf{C}_{x,x} = \mathbf{C}_{x,x} + \xi \cdot \mathbf{I}_{\dim(\mathbf{x})}$ for some very small value of ξ, e.g., $\xi = 10^{-4}$. For a more sophisticated way of selecting the nugget size, see **?**. Note that adding a nugget here is equivalent to adding a small amount of observation variance for the simulator observations. That is, in adding this nugget, we no longer require that the GP emulator precisely interpolate the simulation observations. However, for very small nuggets, this effect is so small as to be insignificant, though the computational benefits remain. Thus nothing is lost in continuing to think of the GP emulator as interpolating the simulation observations.

The other problem – round-off error due to small likelihoods – can be alleviated through substituting the use of log-likelihoods in the MCMC routine. Thus, rather than finding α in (16 directly, it is preferable to find

$$\log \alpha = \log \pi(\theta^*, c^*|\mathcal{D}) + \log \pi_{prop}(\theta^{(i-1)}, c^{(i-1)}|\theta^*, c^*) - \log \pi(\theta^{(i-1)}, c^{(i-1)}|\mathcal{D}) - \log \pi_{prop}(\theta^*, c^*|\theta^{(i-1)}, c^{(i-1)}) \tag{17}$$

Of course, in order to enjoy the computational benefits of this approach, one must perform calculations on the log scale from the start; it would not do to, e.g., find $\pi(\theta^*, c^*|\mathcal{D})$ on its original scale and then take its log.

2 Calibration for design

Suppose that a researcher has a fairly reliable computer model of a given system. Suppose furthermore that

2.1 Desired observations

2.1.1 Level of desired data

2.1.2 Set target via desired data, or prior distribution?

2.2 Model shortcoming

2.2.1 Observation error

2.2.2 Model discrepancy

2.3 Field observations and model discrepancy

2.3.1 Pre-calibration

2.3.2 Joint calibration

2.4 Hyperparameter estimation

3 Application

3.1 Project background

3.2 Emulation of finite element simulator

Blah

3.2.1 Wind turbine blade simulator

Blah

3.2.2 Mathematical basis for the emulator

Blah

3.2.3 Experimental design

Blah

3.2.4 Covariance parameters

Blah

Finding covariance parameters via MCMC Blah

Grid optimization Blah

Gradient method Blah

4 MCMC using the emulator

Blah

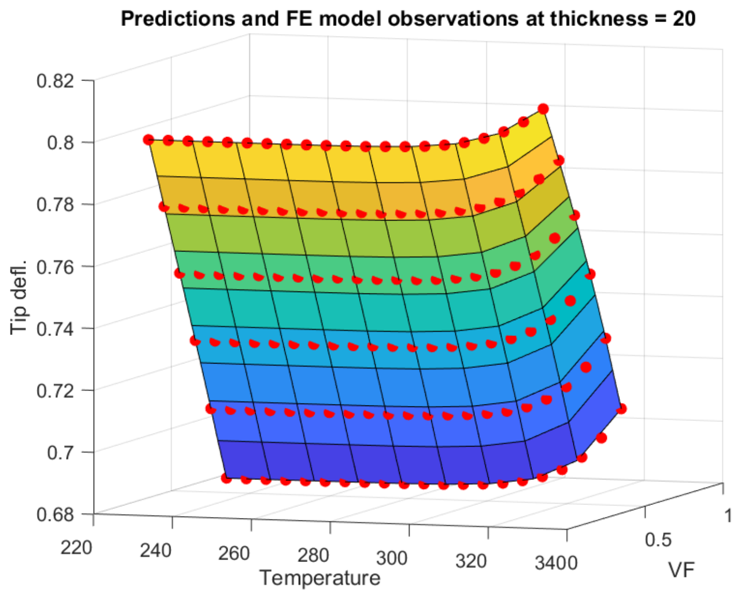

Figure 2: A slice of the GP emulator (restricted to the output for tip deflection) at thickness =20mm. Red dots are observations from the simulator.

4.1 MCMC methods

Blah

4.2 The model

Blah

4.2.1 Desired observation variance

	Heteroskedastic, constant	Homoskedastic, constant	Heteroskedastic, prior	Homoskedastic, prior
Deflection	0.749	0.729	0.659	0.709
Rotation	0.0904	0.0865	0.0773	0.0843
Cost	276.16	236.11	350.80	233.95

Table 1: Comparison of model outputs, where the desired data outputs are assumed to be either homoskedastic or heteroskedastic, with either a specified constant variance or a $1/\sigma^2$ prior.

4.2.2 Full model and likelihood

Blah

4.2.3 Convergence difficulties

Blah

4.2.4 Implementation of the Metropolis-Hastings algorithm

Blah

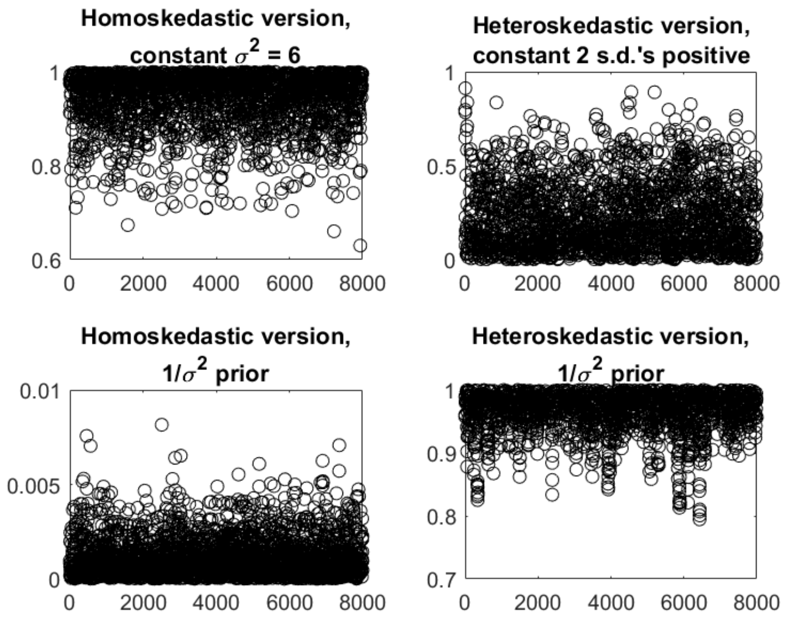

Figure 3: MCMC results at various observation variance settings.

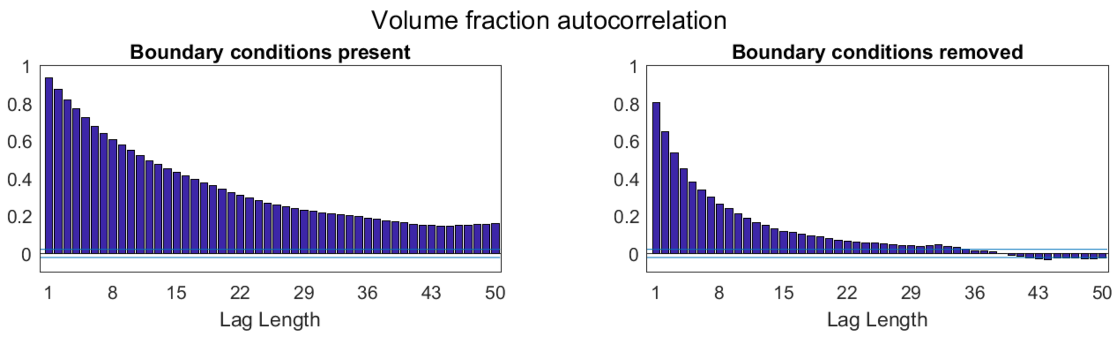

Figure 4: Auto-correlation for draws both with and without the elimination of boundary conditions.

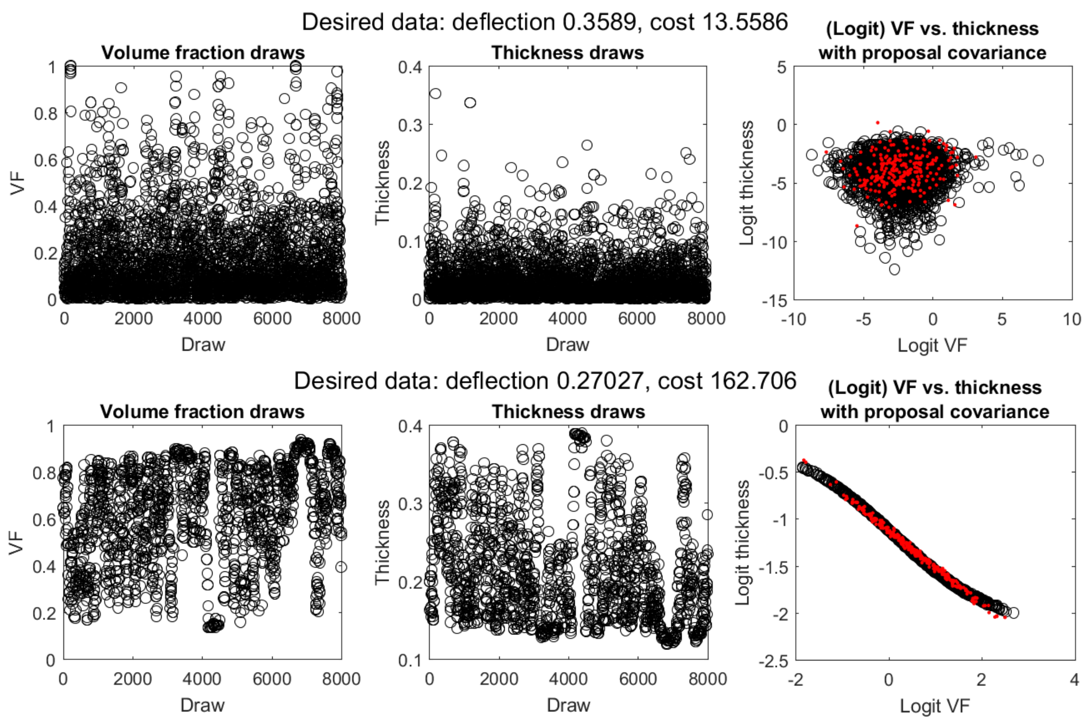

Figure 5: MCMC results for low deflection and cost (top row) and low deflection with easily achievable cost (bottom row).

4.3 Which data to desire?

Blah

4.3.1 Motivations behind the choice of desired data

Blah

4.3.2 Differing results

Desired data d	σ^2_{defl}	σ^2_{rot}	σ^2_{cost}	$\mu_{v\|d}$	$\mu_{h\|d}$	$\sigma^2_{v\|d}$	$\sigma^2_{h\|d}$
$(0,0,0)$	375.45	277.69	2.62	0.215	$4.01 \cdot 10^{-2}$	$4.41 \cdot 10^{-2}$	$1.92 \cdot 10^{-3}$
$(0.65, 0.077, 96)$	16.74	15.25	$4.62 \cdot 10^{-7}$	$1.09 \cdot 10^{-3}$	$3.36 \cdot 10^{-4}$	$1.02 \cdot 10^{-5}$	$9.97 \cdot 10^{-6}$

Table 2: Comparison of results for two different (low) values of d. Values listed are, respectively, the posterior means for the observation variance of each model output, posterior means for volume fraction (v) and thickness (h), and posterior variance of volume fraction and thickness.

4.4 Exponentially distributed desired data

Blah

4.4.1 Motivation

Blah

4.4.2 Implementation and results

Blah

4.5 Identifiability issues

Blah

5 Future work

Blah

5.1 Alternative means of handling cost

Blah

5.1.1 Removing cost from the model

Blah

5.1.2 Alternative priors for controlling cost

Blah

5.2 Building a desired data response surface

Blah

5.3 Implementing Hamiltonian Monte Carlo

Blah

5.3.1 Hamiltonian Monte Carlo

Blah

5.3.2 Benefits

Blah

5.4 Model discrepancy

Blah

6 Conclusion

Blah

References

Bastos, Leonardo S., & O'Hagan, Anthony. 2009. Diagnostics for Gaussian Process Emulators. *Technometrics*, **51**(4), 425–438.

Bayarri, M. J., Berger, J. O., Cafeo, J., Garcia-Donato, G., Liu, F., Palomo, J., Parthasarathy, R. J., Paulo, R., Sacks, J., & Walsh, D. 2007a. Computer Model Validation with Functional Output. *The Annals of Statistics*, **35**, 1874–1906.

Bayarri, Maria J, Berger, James O, Paulo, Rui, Sacks, Jerry, Cafeo, John A, Cavendish, James, Lin, Chin-Hsu, & Tu, Jian. 2007b. A Framework for Validation of Computer Models. *Technometrics*, **49**(2), 138–154.

Bornstein, Marc H., & Bornstein, Helen G. 1976. The pace of life. *Nature*, **259**(5544), 557–559.

Brown, D. Andrew, & Atamturktur, Sez. 2018. Nonparametric Functional Calibration of Computer Models. *Statistica Sinica*, **28**(feb), 721–742.

Brynjarsdóttir, Jenni, & O'Hagan, Anthony. 2014. Learning about physical parameters: The importance of model discrepancy. *Inverse Problems*, **30**(11).

Chib, Siddhartha, & Greenberg, Edward. 1995. Understanding the Metropolis-Hastings Algorithm. *The American Statistician*, **49**(4), 327.

Currin, Carla, Mitchell, Toby, Morris, Max, & Ylvisaker, Don. 1991. Bayesian Prediction of Deterministic Functions, with Applications to the Design and Analysis of Computer Experiments. *Journal of the American Statistical Association*, **86**(416), 953–963.

Gelman, Andrew. 1992. Iterative and Non-Iterative Simulation Algorithms. *Computinlg Science and Statistics (Interface Proceedings)*, **24**, 433–438.

Geman, Stuart, & Geman, Donald. 1984. IEEE Transactions on Pattern Analysis and Machine Intelligence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6**(6), 721–741.

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

Hastings, W.K. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**(1), 97–109.

Higdon, Dave, Kennedy, Marc, Cavendish, James C., Cafeo, John A., & Ryne, Robert D. 2004. Combining Field Data and Computer Simulations for Calibration and Prediction. *SIAM Journal on Scientific Computing*, **26**(2), 448–466.

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

Kennedy, Marc C., Anderson, Clive W., Conti, Stefano, & O'Hagan, Anthony. 2006. Case studies in Gaussian process modelling of computer codes. *Reliability Engineering & System Safety*, **91**(10-11), 1301–1309.

Liu, F., Bayarri, M. J., & Berger, J. O. 2009. Modularization in Bayesian analysis, with emphasis on analysis of computer models. *Bayesian Analysis*, **4**(1), 119–150.

Loeppky, Jason L., Bingham, Derek, & Welch, William J. 2006. Computer Model Calibration or Tuning in Practice. *Technometrics.*

Metropolis, Nicholas, Rosenbluth, Arianna W., Rosenbluth, Marshall N., Teller, Augusta H., & Teller, Edward. 1953. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, **21**(6), 1087–1092.

Neal, Radford M. 1998. Regression and Classification Using Gaussian Process Priors. *Pages 475–501 of:* Bernardo, J M, Berger, J O, Dawid, A P, & Smith, A F M (eds), *Bayesian Statistics 6*, vol. 6. New York: Oxford University Press.

O'Hagan, A. 1978. Curve Fitting and Optimal Design for Prediction. *Journal of the Royal Statistical Society. Series B*, **40**(1), 1–42.

Paulo, Rui, García-Donato, Gonzalo, & Palomo, Jesús. 2012. Calibration of computer models with multivariate output. *Computational Statistics and Data Analysis*, **56**, 3959–3974.

Qian, Peter Z. G, Wu, Huaiqing, & Wu, C. F. Jeff. 2008. Gaussian Process Models for Computer Experiments With Qualitative and Quantitative Factors. *Technometrics*, **50**(3), 383–396.

Sacks, Jerome, Welch, William J., Mitchell, Toby J., & Wynn, Henry P. 1989. Design and Analysis of Computer Experiments. *Statistical Science*, **4**(4), 409–423.

Santner, Thomas J., Williams, Brian J., & Notz, William I. 2003. *The Design and Analysis of Computer Experiments*. New York: Springer.

Savitsky, Terrance, Vannucci, Marina, & Sha, Naijun. 2011. Variable Selection for Nonparametric Gaussian Process Priors: Models and Computational Strategies. *Statistical Science*, **26**(1), 130–149.

Shang, Lifeng, & Chan, Antoni B. 2013. On Approximate Inference for Generalized Gaussian Process Models. *Technical Report -City University of Hong Kong.*

Sorokowska, Agnieszka, Sorokowski, Piotr, Hilpert, Peter, Cantarero, Katarzyna, Frackowiak, Tomasz, Ahmadi, Khodabakhsh, Alghraibeh, Ahmad M., Aryeetey, Richmond, Bertoni, Anna, Bettache, Karim, Blumen, Sheyla, Błażejewska, Marta, Bortolini, Tiago, Butovskaya, Marina, Castro, Felipe Nalon, Cetinkaya, Hakan, Cunha, Diana, David, Daniel, David, Oana A., Dileym, Fahd A., Domínguez Espinosa, Alejandra del Carmen, Donato, Silvia, Dronova, Daria, Dural, Seda, Fialová, Jitka, Fisher, Maryanne, Gulbetekin, Evrim, Hamamcolu Akkaya, Aslhan, Hromatko, Ivana, Iafrate, Raffaella, Iesyp, Mariana, James, Bawo, Jaranovic, Jelena, Jiang, Feng, Kimamo, Charles Obadiah, Kjelvik, Grete, Koç, Frat, Laar, Amos, de Araújo Lopes, Fívia, Macbeth, Guillermo, Marcano, Nicole M., Martinez, Rocio, Mesko, Norbert, Molodovskaya, Natalya, Moradi, Khadijeh, Motahari, Zahrasadat, Mühlhauser, Alexandra, Natividade, Jean Carlos, Ntayi, Joseph, Oberzaucher, Elisabeth, Ojedokun, Oluyinka, Omar-Fauzee, Mohd Sofian Bin, Onyishi, Ike E., Paluszak, Anna, Portugal, Alda, Razumiejczyk, Eugenia, Realo, Anu, Relvas, Ana Paula, Rivas, Maria, Rizwan, Muhammad, Salkičević, Svjetlana, Sarmány-Schuller, Ivan, Schmehl, Susanne, Senyk, Oksana, Sinding, Charlotte, Stamkou, Eftychia, Stoyanova, Stanislava, Šukolová, Denisa, Sutresna, Nina, Tadinac, Meri, Teras, Andero, Tinoco Ponciano, Edna Lúcia, Tripathi, Ritu, Tripathi, Nachiketa, Tripathi, Mamta, Uhryn, Olja, Yamamoto, Maria Emília, Yoo, Gyesook, & Pierce, John D. 2017. Preferred Interpersonal Distances: A Global Comparison. *Journal of Cross-Cultural Psychology*, **48**(4), 577–592.

Thompson, Peter A, & Marchant, Eric W. 1995. A Computer Model for the Evacuation of Large Building Populations. *Fire Safety Journal*, **24**, 131–148.

Williams, Brian, Higdon, Dave, Gattiker, Jim, Moore, Leslie, McKay, Michael, & Keller-McNulty, Sallie. 2006. Combining experimental data and computer simulations, with an application to flyer plate experiments. *Bayesian Analysis*, **1**(4), 765–792.