Bayesian Calibration of Expensive Multivariate Computer Experiments

Richard D. Wilkinson

University of Sheffield

This chapter is concered with how to calibrate a computer model to observational data when the model produces multivariate output and is temporally expensive to run. The significance of considering models with long run times is that they can only be run at a limited number of different inputs, ruling out a brute-force Monte Carlo approach. Consequently, all inference must be done with a limited ensemble of model runs. A probabilistic approach is taken here, with the aim being to find a probability distribution which represents our uncertainty about the true model inputs given the observational data and the computer model. We assume statistical models for the measurement errors on the observed data and for the discrepancy between the model and reality. We also describe a statistical model for our uncertainty about the computer model's value at untried input values.

We take a Bayesian approach and describe our prior beliefs about the model and update these beliefs after observing an ensemble of model runs. Gaussian process priors are used as a flexible semi-parametric family to describe our beliefs, and the posterior distribution of the process can be considered as a meta-model of the computer model. We refer to the meta-model as an *emulator* of the computer *simulator* (Sacks *et al.* 1989). This approach allows beliefs to be described about the model output at input configurations not in the original design.

Computational Methods for Large-Scale Inverse Problems and Quantification of Uncertainity. Edited by People on Earth
© 2001 John Wiley & Sons, Ltd

The Bayesian approach to calibration described here was first given by Kennedy and O'Hagan (2001). Their approach was for univariate computer models, and we extend that here to deal with multivariate models. We use principal component analysis to project the multivariate model output onto a lower dimensional space, and then use Gaussian processes to emulate the map from the input space to the lower dimensional space. We can then reconstruct from the subspace to the original data space. This gives a cheap surrogate for the computer model that can be used for calibration.

The layout of this paper is as follows. In Section 1 and 2 we introduce problem and in Section 2 we describe the notation and the calibration framework. In Section 3 we introduce the idea of principal component emulation and in Section 4 we give details of how to use this approach to calibrate multivariate models. To illustrate the methodology we use the University of Victoria intermediate complexity climate model, which we will calibrate to observational data collected throughout the latter half of the twentieth century. The model is introduced at the end of Section 1 and is returned to at the end of each subsequent section.

0.1 Introduction

The process of fitting a model to data has different names depending on the discipline. It is variously known as an inverse problem, data assimilation, parameter estimation, calibration, or as we prefer to think of it, as a Bayesian inference problem. In this section we carefully describe the problem and set up the notation needed for the developments described in the following sections. We consider the problem in which we have a computer model of a physical system along with some observations of the system. The aim is to combine the science captured by the computer model with the physical observations, to learn about parameter values and initial conditions for the system. There are three sources of information that we want to incorporate:

1. The computer model, $m(\cdot)$, built using expert scientific knowledge.
2. Field observations of the physical system, $\mathcal{D}_{\text{field}}$.
3. Other background information and expert knowledge, such as prior distributions of parameters and information about measurement error and model discrepancy.

We consider each of these sources in turn.

The computer model

The computer model, $m(\cdot)$, is considered to be a map from the input space $\Theta \times \mathcal{T}$, to the output space $\mathcal{Y} \subset \mathbb{R}^n$. Here we distinguish between two different types of input parameters: $\theta \in \Theta$ are the calibration parameters that we wish to estimate, and $t \in \mathcal{T}$ are control parameters or index parameters that are assumed to be known.

The calibration parameters θ may be physical constants, context specific constants, or fudge factors needed to make the model perform well. These are parameters which would not need to be specified if we were doing a physical experiment. We take the best input approach and assume that there is a single 'best' value of θ, which we label $\hat{\theta}$, such that the model run at $\hat{\theta}$ most accurately represents the field data given the imposed error structure.

The control parameters $t \in \mathcal{T}$ may be context indicating inputs (for example, they might specify yearly industrial CO_2 emissions), known constants, or output index variables. For multivariate models there is an input-output dichotomy, in that we can treat the model as multivariate or alternatively we could add an index variable to the inputs and consider the model to be univariate. For example, part of the predictions of the UVic climate model introduced below are atmospheric CO_2 concentrations for the years 1800-1999. The approach taken in this chapter is to view this as a 200 dimensional multivariate computer model. An alternative approach would be to add an index variable i to the input, where i indicates which year's CO_2 concentration we wish to predict. This would allow us to view the model as univariate and apply the methods of Kennedy and O'Hagan (2001). However, in practice if the number of outputs is large then there are considerable challenges involved both computationally in emulating the function, and theoretically in specifying a suitable covariance function for the emulator, especially if the outputs are of different type.

Our approach in this chapter is to treat the models as multivariate. We suppress any dependence on context indicating inputs and known constants for notational clarity, although their inclusion does not change the analysis. At certain points we add in an index variable t for the outputs and write $m(\theta, t)$ when this helps with the exposition, however for the most part we just write $m(\theta)$. The usage will be clear from the context. A final point to note is that it is not always clear which constants we should treat as unknown. For while the true physical value of a parameter may be known, it may be that using a different value will lead to better predictions with the computer model. This judgement needs careful thought in conjunction with consideration of the beliefs of the modellers.

The focus here is on calibrating models with long run times, where what we mean by long depends on the situation (number of inputs and outputs, amount of field data, amount of computer power available, etc.). A consequence of this cost will be that we will only have a limited ensemble of model runs available to us. In other words, there will be a set of N design points $D = \{\theta_i : i = 1, \dots, N\}$ for which we know the output of the model $\mathcal{D}_{\text{sim}} = \{y_i = m(\theta_i) : i = 1, \dots, N\}$. We refer to these model runs as the simulated data, $\mathcal{D}_{\text{sim}}$. Note that D should be chosen to be a space filling design such as a maxi-min Latin hypercube or a Sobal sequence. In Section 3 we make assumptions about the continuity and smoothness of $m(\cdot)$ that allows us to predict its value at inputs θ not in the original design.

Field observations

We assume that we have observations of the physical system, $\mathcal{D}_{\text{field}}$, that directly correspond to outputs from the computer model. We let $\zeta(t)$ represent reality at t, where t is an index variable such as time or location etc. The assumption made here is that the field data is a measurement of reality at t with independent Gaussian error. That is

$$\mathcal{D}_{\text{field}}(t) = \zeta(t) + \epsilon(t)$$

where $\epsilon(t) \sim N(\mu_t, \sigma_t^2)$. It will usually be the case that $\mu_t = 0$ for all t, and often the case that we have homoscedastic errors so that $\sigma_t^2 = \sigma^2$ for all t, however neither of these assumptions is necessary for the analysis. Often the mean and variance parameters will be known, and will be reported with the data.

Note that the assumption here is that we observe reality with independent random noise. A common and incorrect assumption in data assimilation approaches is to assume that we observe the model prediction plus independent random noise. If there is any discrepancy between the model and reality, this is assumption is wrong and could lead to serious errors.

Other background information

Calibration is primarily about combining the physics in the model with field observations of the system to produce estimates for $\hat{\theta}$. However, there will often be expert knowledge available that has not been built into the model. Part of this knowledge will be prior information about the likely best input values, gained through previous experiments and reading the literature etc. Ideally, information should be elicited from the experts before they observe either the ensemble of model runs or the field data, however in practise this will often not be the case. It is extremely rare to be completely uncertain about $\hat{\theta}$, indeed it is hard to imagine a situation where values as diverse as $1, 10^{100}, 10^{10000}$ are all held to be equally likely. We represent this prior knowledge as a probability distribtion $\pi(\theta)$ over the range of possible inputs Θ. Although we are using probability to represent our uncertainty about θ, that does not mean that we believe it to be a random value, just unknown. Jaynes (2003) gives an introduction and justification for why the Bayesian paradigm is the right way to treat uncertainty and Garthwaite *et al.* (2005) give an excellent introduction to elicitation of experts beliefs.

As well as prior information about θ, there may also be prior knowledge about other aspects of the experiment. For example, we have already commented that the structure and magnitude of the measurement error of the field observations is often known and can be elicited from the relevant experts. The modellers may also know something about how well the simulator $m(\cdot)$ models reality $\zeta(\cdot)$. As explained in more detail below, when making inferences about $\hat{\theta}$, it is important to account for any discrepancy between the model and reality. We denote this model discrepancy by $\delta(\cdot)$ and ask the model builders to provide information about how and where the model may be wrong. They may, for example, have more confidence in some of the

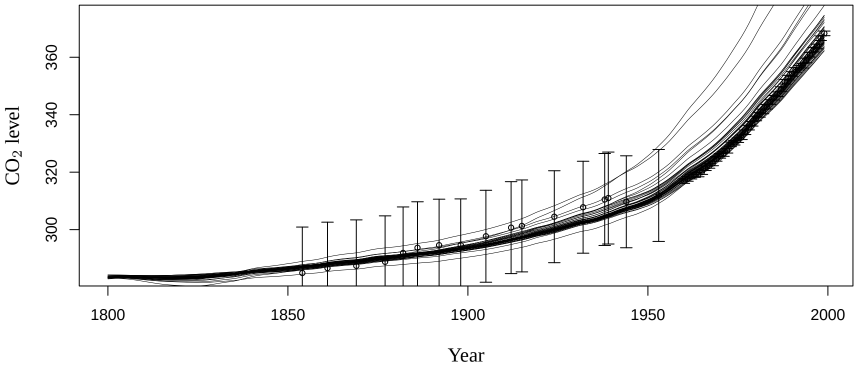

Figure 1 Ensemble of 47 model runs of the UVic climate model for a design on two inputs Q_{10} and K_c. The output (black lines) gives the atmospheric CO$_2$ predictions for 1800-1999, and the 57 field observations are shown as circles with error bars of two standard deviations.

model outputs than others, or they may have more faith in the predictions in some contexts than in others.

Example 0.1.1 (UVic Climate Model) *In order to demonstrate the methodology we introduce an example from climate science which we present along with the theory. We use the University of Victoria Earth System Climate Model (UVic ESCM) coupled with a dynamic vegetation and terrestrial carbon cycle and an inorganic ocean carbon cycle (Meissner et al. 2003). The model was built in order to study potential feedbacks in the terrestrial carbon cycle and to see how these affect future climate predictions. We present a simplified analysis here, with full details available in Ricciuto et al. We consider the model to have just two inputs, Q_{10} and K_c, and to output a time-series of atmospheric CO$_2$ values. Input Q_{10} controls the temperature dependence of respiration and can be considered a carbon source, whereas K_c is the Michaelis–Menton constant for CO$_2$ and controls the sensitivity of photosynthesis and can be considered a carbon sink. The aim is calibrate these two parameters to the Keeling and Whorf (2005) sequence of atmospheric carbon dioxide measurements. Each model run takes approximately two weeks of computer time and we have an ensemble of 47 model runs with which to perform the analysis. The model output and the field observations are shown in Figure 1.*

0.2 Statistical Calibration Framework

Calibration is the process of judging which input parameter values are consistent with the field data, the model and any prior beliefs. The Bayesian approach to calibration is to find the posterior distribution of the best input parameter given these three sources of information; namely, we aim to find

$$\pi(\hat{\theta}|\mathcal{D}_{\text{sim}}, \mathcal{D}_{\text{field}}, E),$$

where E represents the prior information, $\mathcal{D}_{\text{sim}}$ the ensemble of model runs, and $\mathcal{D}_{\text{field}}$ the field observations. This posterior distribution gives relative weights to all $\theta \in \Theta$, and represents our beliefs about the best input value in the light of the computer experiment and the field data.

The posterior distribution of $\hat{\theta}$ is proportional to its likelihood multiplied by its prior distribution:

$$\pi(\hat{\theta}|\mathcal{D}_{\text{sim}}, \mathcal{D}_{\text{field}}, E) \propto \pi(\mathcal{D}_{\text{sim}}, \mathcal{D}_{\text{field}}|\hat{\theta}, E)\pi(\hat{\theta}|E)$$

and so to compute the calibration posterior we require the likelihood of the data $\pi(\mathcal{D}_{\text{sim}}, \mathcal{D}_{\text{field}}|\hat{\theta}, E)$. Often, the hardest part of any calibration is specification of this likelihood, as once we have the prior and the likelihood, finding the posterior distribution is in theory just an integral calculation. In practise however, this will usually require careful application of a numerical integration technique such as a Markov Chain Monte Carlo (MCMC) algorithm.

To specify the likelihood we need to state how reality relates to the computer model, and what we mean by the calibrated input value $\hat{\theta}$. We assume that reality at t, $\zeta(t)$, is equal to the computer model output when run at its 'best value' $\hat{\theta}$ (here t is an index of the outputs) plus a model discrepancy term $\delta(t)$ which captures the failings of the model. We assume that $m(\hat{\theta}, t)$ is sufficient for the model in the calibration, in the sense that once we know $m(\hat{\theta})$ we can not learn anything further about reality from the computer model. Note that $\hat{\theta}$ is the best value here only in the sense of most accurately representing the data according to the specified error structure. The value found for $\hat{\theta}$ need not coincide with the true physical value of θ, and so the calibration parameters are model parameters, not physical parameters. This point should be strongly stressed to the experts when eliciting prior distributions for $\hat{\theta}$. We assume that the field data is a direct measurement of reality recorded with some independent measurement error $\epsilon(t)$. Mathematically, the calibration framework is

$$\begin{aligned}\zeta(t) &= m(\hat{\theta}, t) + \delta(t)\\ \mathcal{D}_{\text{field}}(t) &= \zeta(t) + \epsilon(t).\end{aligned}$$

Note that the input-output duality allows us to write the inference framework as

$$\mathcal{D}_{\text{field}} = m(\hat{\theta}) + \delta + \epsilon$$

by writing all quantities as vectors. Once we have made distributional assumptions about $\delta(t)$, $\epsilon(t)$ and possibly $m(\hat{\theta}, t)$, this framework allows us to calculate a likelihood function for the data.

The decision to include a model error term, $\delta(\cdot)$, in the calibration is motivated by several ideas. Firstly, if we do not model the discrepancy, we would be making the assumption that the field data is just the model output plus independent random errors. Without the model error term this independence assumption will be wrong. Secondly, as stated by Box (1976), all models are wrong. There are a variety of reasons why this should be. Perhaps not all physical processes have been included, or perhaps the model equations are solved using a numerical approximation, and so on. It is important to account for this added source of uncertainty in any predictions, as if not, we may have undue confidence in our predictions.

Goldstein and Rougier (2008) take this idea further and introduce the idea of a reified model. The reified model is the version of the model we would run if we had unlimited computing resources. So for example, in global climate models the earth's surface is split into a grid of cells and the computation assumes each cell is homogeneous (UVic uses a 100×100 grid across the earth's surface with eight ocean depths). If infinite computer resources were available we could let the grid size tend to zero, giving a continuum of points across the globe. While clearly an impossibility, thinking about the reified model helps us to break down the model error into more manageable chunks; we can consider the difference between the actual computer model and the reified model, and then the difference between the reified model and reality. This approach may provide a way to help the modellers think more carefully about the model discrepancy term $\delta(\cdot)$.

If the computer model is quick to run, then we can essentially assume that its value is known for all possible input configurations, as in any inference procedure we can simply evaluate the model whenever its value is needed. In this case, calculation of the calibration posterior

$$\pi(\theta|\mathcal{D}_{\text{field}}, m, E) \propto \pi(\mathcal{D}_{\text{field}}, |\theta, m, E)\pi(\theta|E),$$

where m represents the computer model, is relatively easy as the calibration framework gives that

$$\mathcal{D}_{\text{field}} - m(\hat{\theta}) = \delta + \epsilon.$$

Given distributions for the model discrepancy δ and measurement error ϵ we can calculate the likelihood of the field data, and thus can find the posterior distribution. If the model is not quick running, then the model's value is unknown at all input values other than those in design D. This uncertainty about the model output at untried input configurations is commonly called *code uncertainty*. If we want to account for this source of uncertainty in the calibration then we need a statistical model in order to describe our beliefs about the output value for all possible input values. This is the topic of the next section.

0.3 Principal Component Emulation

0.3.1 Emulation

If the computer model, $m(\cdot)$, is temporally expensive to evaluate, then its value is unknown at all input values except those in a small ensemble of model runs. We assume that the code has been run N times for all inputs in a space-filling design $D = \{\theta_i \in \Theta : i = 1, \ldots, N\}$ to produce output $\mathcal{D}_{\text{sim}} = \{m(\theta_i) \in \mathbb{R}^n : i = 1, \ldots, N\}$, and that further model runs are not available. For any $\theta \notin D$ we are uncertain about the value of the model for this input. However, if we believe that the model is a smooth continuous function of the inputs, then we can learn about $m(\theta)$ by looking at ensemble members run with inputs close to θ. We could, for example, choose to predict $m(\theta)$ by linearly interpolating from the closest ensemble members. The function used to interpolate and extrapolate $\mathcal{D}_{\text{sim}}$ to other input values is commonly called an emulator, and there is extensive literature on emulation (sometimes called meta-modelling) for computer experiments (see Santner *et al.* (2003) for references).

We use a Bayesian framework to build an emulator which accurately captures our beliefs about the model. We can elicit prior distributions about the shape of the function, for example, do we expect linear, quadratic or sinusoidal output, and about the smoothness and variation of the output, for example, over what kind of length scales do we expect the function to vary. A convenient and flexible semiparametric family that is widely used to build emulators are Gaussian processes. They can be used to give predictions of the model's value at any input, with the predictions in the form of Gaussian probability distributions over the output space. They can also incorporate a wide range of prior beliefs about both the prior mean structure and the covariance between points. For univariate computer models we write

$$\eta(\cdot)|\beta, \lambda, \sigma^2 \sim GP(g(\cdot), \sigma^2 c(\cdot, \cdot))$$

where $g(\theta) = \beta^T h(\theta)$ is a prior mean function which is usually taken to be a linear combination of a set of regressor functions, $h(\cdot)$, where β represents the vector of coefficients. The prior variance is assumed here to be stationary across the input range and is written as the product of a prior at-a-point variance $\sigma^2 = \mathbb{V}\text{ar}(\eta(\theta))$, and a correlation function $\mathbb{C}\text{orr}(m(\theta_1), m(\theta_2)) = c(\theta_1, \theta_2)$. Common choices for the correlation function include the Matérn function and the exponential correlation functions, such as the commonly used squared exponential family

$$c(\theta_1, \theta_2) = \exp\left[-(\theta_1 - \theta_2)^T \Lambda (\theta_1 - \theta_2)\right].$$

Here $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$ is a diagonal matrix containing the roughness parameters. The λ_i represent how quickly we believe the output varies as a function of the input, and can be thought of as a measure of the smoothness of the function.

Once we observe the ensemble of model runs $\mathcal{D}_{\text{sim}}$, we update the prior beliefs to find the posterior distribution. If we choose a conjugate prior distribution for β such as an uninformative improper distribution $\pi(\beta) \propto 1$, or a Gaussian distribution, then we can integrate out β to find the posterior

$$m(\cdot)|\mathcal{D}_{\text{sim}}, \lambda, \sigma^2 \sim GP(g^*(\cdot), \sigma^2 c^*(\cdot, \cdot))$$

for modified functions $g^*(\cdot)$ and $c^*(\cdot, \cdot)$. Details of the calculation and forms for g^* and c^* can be found in Rasmussen and Williams (2006) and many other texts. It is not possible to find a conjugate prior distribution for the roughness parameters, so we take an empirical Bayes approach and give each λ_i a prior distribution and then find its maximum a posteriori value and fix λ_i at this value, approximating $\pi(m(\cdot)|\mathcal{D}_{\text{sim}}, \sigma^2)$ by $\pi(m(\cdot)|\mathcal{D}_{\text{sim}}, \hat{\lambda}, \sigma^2)$. If we give σ^2 an inverse chi-squared distribution it is possible to integrate it out analytically, however, this leads to a t-process distribution for $m(\cdot)$ which is inconvenient later, and so we leave σ^2 and use MCMC to integrate it out numerically later in the analysis.

The Gaussian process emulator approach described above is for univariate models. For multivariate outputs we could build separate independent emulators for each output, although this has the disadvantage of ignoring the correlations between the outputs and will generally perform poorly if the size of the ensemble is small (as we are throwing away valuable information). Conti and O'Hagan (2007) provide an extension of the above approach which allows us to model a small number of multivariate outputs capturing the correlations between them, and Rougier (2008) describes an outer product emulator which takes advantage of some mathematical tricks to make computational savings and so can be used on a larger number of dimensions if we are prepared to make some fairly general assumptions about the form of the regressors and the correlations. Both of these approaches require careful thought about what correlations can be expected between output dimensions. This can be difficult to think about, especially with modellers who may not have much experience with either probability or statistics. They are also both limited by the size of problem that can be tackled, although Rougier (2008) made great advances on this front. For models with hundreds or thousands of outputs a direct emulation approach may not be feasible, and so here we use a data reduction method to reduce the size of the problem to something more manageable.

0.3.2 Principal Component Emulation

We take an approach here similar to Higdon *et al.* (2008), and use a dimension reduction technique to project the output from the computer model onto a subspace with a smaller number of dimensions and then build emulators of the map from the input space to the reduced output space. The only requirement of the dimension reduction is that there is a method for reconstruction to the original output space. We use principal component analysis here (also known as the method of empirical orthogonal functions), as the projection is then guaranteed to be the optimal linear projection, in terms of minimizing the average reconstruction error. A schematic plot of this idea is shown in Figure 2. The computer model $m(\cdot)$ is a function from input space Θ to output space $\mathcal{Y}$. Principal component analysis provides a map from

$$\begin{tikzcd}
\Theta \arrow[rr, "m(\cdot)"] \arrow[dr, "\eta^{pc}(\cdot)"'] & & \mathcal{Y} \arrow[dl, shift right, "\text{PCA}"'] \\
& \mathcal{Y}^{pc} \arrow[ur, shift right, "\text{PCA}^{-1}"'] &
\end{tikzcd}$$

Figure 2 Schematic plot of the idea behind principal component emulation. Θ is the input space, $\mathcal{Y}$ the output space, and $m(\cdot)$ the computer model. We let $\eta^{pc}(\cdot)$ denote the Gaussian process emulator from Θ to principal subspace $\mathcal{Y}^{pc}$.

full output space $\mathcal{Y}$ to reduced space $\mathcal{Y}^{pc}$. We build Gaussian process emulators to map from Θ to $\mathcal{Y}^{pc}$ and then use the inverse of the original projection (also a linear projection) to move from $\mathcal{Y}^{pc}$ to $\mathcal{Y}$. This gives a computationally cheap map from the input space Θ to the output space $\mathcal{Y}$ which does not use the model $m(\cdot)$. This cheap surrogate, or emulator, approximately interpolates all the points in the ensemble (it is approximate due to the error in the principal component reconstruction) and gives probability distributions for the model output for any value of the input.

Principal component analysis is a linear projection of the data onto a lower dimensional subspace (the principal subspace) such that the variance of the projected data is maximised. It is commonly done via an eigenvalue decomposition of the correlation matrix, but for reasons of computational efficiency, we will use a singular value decomposition of the data here. Let Y denote an $N \times n$ matrix with row i the i^{th} run of the computer model, $Y_{i\cdot} = m(\theta_i)$ (recall that the model output is n dimensional and that there are N runs in the ensemble $\mathcal{D}_{\text{sim}}$). The dimension reduction algorithm can then be described as follows:

1. Centre the matrix. Let $\boldsymbol{\mu}$ denote the row vector of column means, let Y' be the matrix Y with $\boldsymbol{\mu}$ subtracted from each row ($Y' = Y - \boldsymbol{\mu}\mathbf{1}$) so that the mean of each column of Y' is zero. We might also choose to scale the matrix, so that the variance of each column is one.

2. Calculate the singular value decomposition
$$Y' = U\Gamma V^*.$$
V is an $n \times n$ unitary matrix containing the principal components (the eigenvectors) and V^* denotes its complex conjugate transpose. Γ is an $N \times n$ diagonal matrix containing the principal values (the eigenvalues) and is ordered so that the magnitude of the diagonal entries decreases across the matrix. U is an $N \times N$ matrix containing the left singular values.

3. Decide on the dimension of the principal subspace, n^* say ($n^* < n$). An orthonormal basis for the principal subspace is then given by the first n^* columns of V (the leading n^* eigenvectors) which we denote as V_1 (an $n \times n^*$ matrix). Let V_2 denote the matrix containing the remaining columns of V.

4. Project Y' onto the principal subspace. The coordinates in the subspace (the factor scores) are found by projecting onto V_1:

$$Y^{pc} = Y'V_1.$$

   The i^{th} row of Y^{pc} then denotes the coordinate of the i^{th} ensemble member in the space $\mathcal{Y}^{pc}$.

Some comments:

- Note that the principal component analysis is done across the columns of the matrix rather than across the rows as is usual. The result is that the eigenvalues are of the same dimension as the original output with the leading eigenvalue often taking the general form of the output.

- There is no established method for deciding on the dimension n^* of the principal subspace. The percentage of variance explained (sum of the corresponding eigenvalues in Γ) is often used as a heuristic, with the stated aim being to explain 95% or 99% of the variance. We must also decide which components to include in V_1. It may be found that components which only explain a small amount of the variance (small eigenvalues) are important predictively, as was found in principal component regression (Jolliffe 2002). One method of component selection is through the use of diagnostic plots as explained below.

This leaves us with the coordinates of the ensemble in the principal subspace $\mathcal{Y}^{pc}$, with each row corresponding to the same row in the original design D. Gaussian processes can now be used to emulate this map. Usually, we will have $n^* > 1$, and so we still need to use a multivariate emulator such as that proposed by Rougier (2008). However, emulating the reduced map with n^* independent Gaussian processes often performs as well as using a fully multivariate emulator, especially if the size of the ensemble N is large compared with n^*. Another trick which helps with the emulation is to scale the matrix of scores so that each column has variance one. This helps with tuning the MCMC sampler for the σ^2 parameters in the Gaussian process covariance function, as it makes the n^* dimensions comparable with each other.

To reconstruct from the subspace $\mathcal{Y}^{pc}$ to the fullspace $\mathcal{Y}$ is also a linear transformation. We can post-multiply the scores by V_1^T to give a determinist reconstruction $Y'' = Y^{pc}V_1^T$. However, this does not account for the fact that by projecting into a n^*-dimensional subspace, we have discarded information in the dimension reduction. To account for this lost information we add random multiples of the eigenvectors which describe the discarded dimensions, namely V_2. We model these random multiples as zero-mean Gaussian distributions with variances corresponding to the relevant eigenvalues. This gives a stochastic rather than a deterministic reconstruction, which accounts for the error in the dimension reduction. In summary, we reconstruct as

$$Y'' = Y^{pc}V_1^T + \Phi V_2^T$$

where Φ is an $N \times (n - n^*)$ matrix with i^{th} column containing N draws from a $N(0, \Gamma_{n^*+i,n^*+i})$ distribution. We then must add the column means of Y to each row of Y'' to complete the emulator.

A useful diagnostic tool when building emulators are leave-one-out cross validation plots. These are obtained by holding back one of the N training runs in the ensemble, training the emulator with the remaining $N - 1$ runs, and then predicting the held back values. Plotting the predicted values, with 95% credibility intervals, against the true values for each output dimension gives valuable feedback on how the emulator is performing and ultimately allows us to validate the emulator. These plots can be used to choose the dimension of the principal subspace and which components to include. They are also useful for choosing which regressor functions to use in the specification of the mean structure. Once we have validated the emulator, we can then proceed to use it to calibrate the model.

Example 0.3.1 (UVic continued) *We use principal component emulation to build a cheap surrogate for the UVic climate model introduced earlier. Recall that the output of the model is a time-series of 200 atmospheric CO$_2$ predictions. Figure 3 shows the leave-one-out cross-validation plots for a selection of four of the 200 output points. The emulation was done by projecting the time-series onto a 10 dimensional principal subspace and then emulating each map with independent Gaussian processes before reconstructing the data back up to the original space of 200 values. A quadratic prior mean structure was used, $h(\theta_1, \theta_2) = (1, \theta_1, \theta_2, \theta_1^2, \theta_2^2, \theta_1\theta_2)^T$, as the cross-validation plots showed that this gave superior performance over a linear or constant mean structure, with only negligible further gains possible by including higher order terms. The plots show that the emulator is accurately able to predict the held back runs and that the uncertainty in our predictions (shown by the 95% credibility intervals) provide a reasonable measure of our uncertainty (with 91% coverage on average).*

0.4 Multivariate Calibration

Recall that our aim is to find the distribution of $\hat{\theta}$ given the observations and the model runs, namely

$$\begin{aligned}\pi(\hat{\theta}|\mathcal{D}_{\text{field}}, \mathcal{D}_{\text{sim}}) &\propto \pi(\mathcal{D}_{\text{field}}|\mathcal{D}_{\text{sim}}, \hat{\theta})\pi(\hat{\theta}|\mathcal{D}_{\text{sim}}) \\ &\propto \pi(\mathcal{D}_{\text{field}}|\mathcal{D}_{\text{sim}}, \hat{\theta})\pi(\hat{\theta})\end{aligned}$$

where we have noted that $\pi(\mathcal{D}_{\text{sim}}|\hat{\theta}) = \pi(\mathcal{D}_{\text{sim}})$ and so can be ignored in the posterior distribution of $\hat{\theta}$, leaving $\pi(\mathcal{D}_{\text{field}}|\mathcal{D}_{\text{sim}}, \hat{\theta})$ to be specified in order to find the

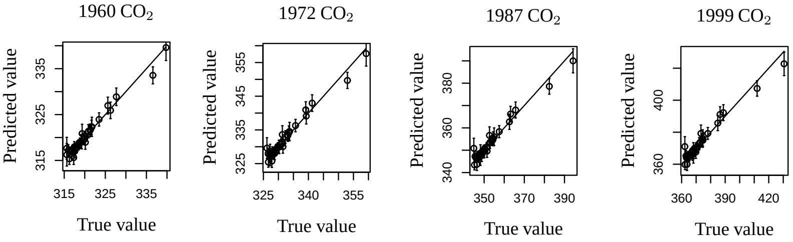

Figure 3 Leave-one-out cross validation plots for a selection of four of the 200 outputs. The error bars show 95% credibility intervals on the predictions. The two outliers seen in each plots are for model runs with inputs on the edge of the design. These points are predictions where we extrapolate rather than interpolate from the other model runs.

posterior. The calibration framework specified earlier

$$\mathcal{D}_{\text{field}}(t) = m(t, \hat{\theta}) + \delta(t) + \epsilon(t) \tag{1}$$

contains three different terms we need to model. Parameter $\hat{\theta}$ is chosen to make $m(\hat{\theta}, t)$ and $\delta(t)$ independent for all t (Kennedy and O'Hagan 2001), and the measurement error $\epsilon(t)$ is also independent of both terms. This allows us to specify the distribution of each part of Equation (1) in turn, and then calculate the distribution of the sum of the three components. If all three parts have a Gaussian distribution, then the sum will also be Gaussian.

Distributional choices for $\epsilon(t)$ and $\delta(t)$ will be specific to each individual problem, but usually measurement errors are assumed to be zero-mean Gaussian random variables. Our approach allows for heteroscedastic errors, with both known and unknown variances. Usually however, measurement errors will be reported with the data.

Kennedy and O'Hagan also recommend the use of Gaussian process priors for the discrepancy function to capture the error between the best model prediction and reality. While this is convenient mathematically, sensible forms for the discrepancy will need to be decided with the modellers in each case separately. Here we assume that $\delta(t)$ is modelled as a Gaussian process for ease of exposition.

Finally, we must find the distribution of $m(\hat{\theta}, t)$ using the principal component emulator. Before considering the map from Θ to $\mathcal{Y}$, we must first consider the distribution of the emulator $\eta^{pc}(\cdot)$ from Θ to $\mathcal{Y}^{pc}$. Using independent Gaussian processes to model the map from the input space to each dimension of the principal subspace (i.e., $\eta^{pc} = (\eta_1^{pc}, \ldots, \eta_{n^*}^{pc})$), we have that the prior distribution for $\eta_i^{pc}(\cdot)$ is

$$\eta_i^{pc}(\cdot)|\beta_i, \sigma_i^2, \lambda_i \sim GP(g_i(\cdot), \sigma_i^2 c_i(\cdot, \cdot)).$$

If we give β_i a uniform improper prior $\pi(\beta_i) \propto 1$, we can then condition on $\mathcal{D}_{\text{sim}}$ and integrate out β_i to find

$$\eta_i^{pc}(\cdot)|\mathcal{D}_{\text{sim}}, \sigma_i^2, \lambda_i \sim GP(g_i^*(\cdot), \sigma_i^2 c_i^*(\cdot, \cdot))$$

where

$$\begin{aligned} g_i^*(\theta) &= \hat{\beta}^T h(\theta) + t(\theta)^T A^{-1}(Y_{\cdot i}^{pc} - H\hat{\beta}) \\ c_i^*(\theta, \theta') &= c(\theta, \theta') - t(\theta)^T A^{-1} t(\theta') + (h(\theta)^T - t(\theta)^T A^{-1} H)(H^T A^{-1} H)^{-1} \\ &\qquad \times (h(\theta')^T - t(\theta')^T A^{-1} H)^T \end{aligned}$$

and

$$\begin{aligned} \hat{\beta}_i &= (H^T A^{-1} H)^{-1} H^T A^{-1} Y_{\cdot i}^{pc} \\ t(\theta) &= (c(\theta, \theta_1), \ldots, c(\theta, \theta_N)) \\ \{A_i\}_{jk} &= \{c_i(\theta_j, \theta_k)\}_{j,k=1,\ldots,N} \\ H^T &= (h(\theta_1), \ldots, h(\theta_N)). \end{aligned}$$

assuming the regressors, $h(\cdot)$, are the same for each dimension. Here, $Y_{\cdot i}^{pc}$ denotes the i^{th} column of matrix Y^{pc}, and $\theta_1, \ldots, \theta_N$ are the N design points for the ensemble of model runs. The reconstruction to the full space, $\eta^e(\cdot) = \eta^{pc}(\cdot)V_1^T + \Phi V_2^T$, then has posterior distribution

$$\eta^e(\theta)|\mathcal{D}_{\text{sim}}, \sigma^2, \lambda \sim N(g^*(\theta)V_1^T, \sigma^2 c^*(\theta, \theta)V_1 V_1^T + V_2 \Gamma' V_2^T)$$

where $g^* = (g_1^*, \ldots, g_{n^*}^*)$ and $\Gamma' = \text{diag}(\Gamma_{n^*+1, n^*+1}, \ldots, \Gamma_{n,n})$. As commented previously, we take an empirical Bayes approach and fix the roughness parameters at their maximum likelihood estimates. We do not integrate σ^2 out analytically for reasons of tractability, but leave them in the calculation and use MCMC to integrate them out numerically later.

If all three parts of Equation (1) are Gaussian then we can write down the likelihood of the field data conditional on the parameters:

$$\pi(\mathcal{D}_{\text{field}}|\mathcal{D}_{\text{sim}}, \sigma^2, \theta, \gamma_\delta)$$

where γ_δ are parameters required for the discrepancy term $\delta(t)$ (possibly also includes measurement error parameters). We elicit prior distributions for θ and γ_δ from the modellers and decide upon priors for σ^2 ourselves (emulators parameters are the responsibility of the person performing the emulation). We then use a Markov Chain Monte Carlo algorithm to find the posterior distributions. It is possible to write down a Metropolis-within-Gibbs algorithm to speed up the MCMC calculations, although we do not give the details here.

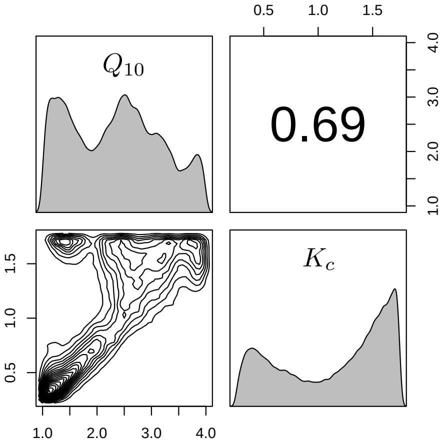

Figure 4 Marginal posterior distributions for two of the calibration parameters, Q_{10} and K_c, in the UVic climate model. The two plots on the leading diagonal show the individual marginal plots. The bottom left plot shows the pairwise marginal distribution, and the top right box shows the posterior correlation between Q_{10} and K_c.

Example 0.4.1 (UVic continued) *Figure 4 shows the marginal posterior distributions from calibrating the UVic model to the Keeling and Whorf (2005) observations. We use an autoregressive process of order one for the discrepancy term with* $\delta(t) = \rho\delta(t-1) + U$ *where* $U \sim N(0, \sigma_\delta^2)$*. We give* ρ *a* $\Gamma(5,1)$ *prior truncated at one, and* σ_δ^2 *a* $\Gamma(4, 0.6)$ *prior distribution. The Markov chains were run for 1,000,000 iterations. The first 200,000 samples were discarded as burn-in and the remaining samples were thinned to every tenth value leaving 80,000 samples. Uniform prior distributions were used for* Q_{10} *and* K_c *(*$Q_{10} \sim U[1,4]$ *and* $K_c \sim U[0.25, 1.75]$*), and* $\Gamma(1.5, 6)$ *priors were used for each of the emulator variances* σ^2*. Tests were done to check the sensitivity of the results to choice of prior distribution, and the analysis was robust to changes in priors for* σ^2 *and* γ_δ*, but not to changes in the priors for* Q_{10} *and* K_c*.*

This will not usually be the end of the calibration process. The results will be returned to the modellers, who may decide to use them to improve the model, before another calibration is performed.

0.5 Conclusions

In this chapter we have shown how to extended the calibration approach of Kennedy and O'Hagan (2001) to enable the calibration of computer models with a large number of multivariate outputs. Projecting the data onto a lower dimensional space enables existing emulator technology to be used to emulate the computer simulator. Principal component analysis provides a simple and natural projection onto a lower dimensional space and the data can easily be reconstructed to the full space with the inverse linear projection. The calibration takes account of measurement error, code uncertainty and model discrepancy, giving posterior distributions which incorporate expert knowledge as well as the model runs and field data.

It should be stressed that the resulting posteriors do not necessarily give estimates of the true value of physical parameters, but rather give values which lead the model to best explain the data. In order to estimate the true physical value of parameters, the model discrepancy function must be very carefully specified. This is still a new area of research and much remains to be done in the area of modelling discrepancy functions.

Bibliography

Box GEP 1976 Science and statistics. *Journal of the American Statistical Association*, **71**, 791–799.

Conti S and O'Hagan A 2007 Bayesian emulation of complex multi-output and dynamic computer models. In submission. Available as Research Report No. 569/07, Department of Probability and Statistics, University of Sheffield.

Garthwaite PH, Kadane JB and O'Hagan A 2005 Statistical methods for eliciting probability distributions. *Journal of the American Statistical Association* **100**, 680–701.

Goldstein M and Rougier JC 2008 Reified Bayesian Modelling and Inference for Physical Systems. To appear *Journal of Statistical Planning and Inference*.

Higdon D, Gattiker J, Williams B and Rightley M 2008 Computer Model Calibration Using High-Dimensional Output *Journal of the American Statistical Association* **103**, 570-583.

Jaynes ET 2003 *Probability Theory: The Logic of Science*, edited by Bretthorst GL, Cambridge University Press.

Jolliffe IT 2002 *Principal Component Analysis* 2nd edn. Springer.

Keeling CD and Whorf TP 2005 Atmospheric CO_2 records from sites in the SIO air sampling network. In *Trends: A Compendium of Data on Global Change*, Carbon Dioxide Information Analysis Center, Oak Ridge National Laboratory, U.S. Department of Energy, Tenn., U.S.A.

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

Meissner KJ, Weaver AJ, Matthews HD and Cox PM 2003 The role of land surface dynamics in glacial inception: a study with the UVic Earth System Model *Climate Dynamics* **21**, 515–537.

Rasmussen CE and Williams CKI 2006 *Gaussian Processes for Machine Learning*, MIT Press.

Ricciuto DM, Tonkonojenkov R, Urban N, Wilkinson RD, Matthews D, Davis KJ and Keller K Assimilation of oceanic, atmospheric, and ice-core observations into an Earth system model of intermediate complexity. *In submission*.

Rougier JC 2008 Efficient Emulators for Multivariate Deterministic Functions. To appear in *Journal of Computational and Graphical Statistics*.

Sacks J, Welch WJ, Mitchell TJ and Wynn HP 1989 Design and analysis of computer experiments. *Statistical Science* **4**, 409–423.

Santner TJ, Williams BJ and Notz W 2003 *The Design and Analysis of Computer Experiments*, Springer.