Bayesian Calibration of Computer Models

Marc C. Kennedy and Anthony O'Hagan

University of Sheffield

Summary. We consider prediction and uncertainty analysis for systems which are approximated using complex mathematical models. Such models, implemented as computer codes, are often generic in the sense that by suitable choice of some of the model's input parameters the code can be used to predict behaviour of the system in a variety of specific applications. However, in any specific application the values of necessary parameters may be unknown. In this case, physical observations of the system in the specific context are used to learn about the unknown parameters. The process of fitting the model to the observed data by adjusting the parameters is known as calibration. Calibration is typically effected by ad hoc fitting, and after calibration the model is used, with the fitted input values, to predict future behaviour of the system. We present a Bayesian calibration technique which improves on this traditional approach in two respects. First, the predictions allow for all sources of uncertainty, including the remaining uncertainty over the fitted parameters. Second, they attempt to correct for any inadequacy of the model which is revealed by discrepancy between the observed data and the model predictions from even the best-fitting parameter values.
The method is illustrated using data from a nuclear radiation release at Tomsk, and from a more complex simulated nuclear accident exercise.

Keywords: CALIBRATION; COMPUTER EXPERIMENTS; DETERMINISTIC MODELS; GAUSSIAN PROCESS; INTERPOLATION; MODEL INADEQUACY; SENSITIVITY ANALYSIS; UNCERTAINTY ANALYSIS.

1. OVERVIEW

1.1. Computer models and calibration

Various sciences use mathematical models to describe processes that would otherwise be very difficult to analyse, and these models are typically implemented in computer codes. Often, the mathematical model is highly complex, and the resulting computer code is large and may be expensive in terms of the computer time required for a single run. Nevertheless, running the computer model will be much cheaper than making direct observations of the process. Sacks, Welch, Mitchell and Wynn (1989) give a number of examples. The codes we consider are deterministic: that is, running the code with the same inputs always produces the same output.

Computer models are generally designed to be applicable to a wide range of particular contexts. However, in order to use a model to make predictions in a specific context it may be necessary first to *calibrate* the model using some observed data. To illustrate this process we introduce a simple example. Two more examples are described in detail in Section 2.2.

In order to decide on a dose regime (e.g. size, frequency and release rates of tablets) for a new drug, a pharmacokinetic model is used. This models the movement of the drug through various 'compartments' of the patient's body and its eventual elimination (e.g. by chemical reactions or excretion). Such a model allows the consequences of any given dose regime to be explored. However, in order to use the the model for a particular drug it is necessary to specify rates with which it moves between different body compartments, such as the rate of transfer from stomach to blood, or of elimination from the liver. Some or all of these rates will be specific to the drug in question. and it is through requiring the user to specify them as inputs to the code that the model achieves applicability to a wide range of drugs. These rates will, of course, not be known for a given drug, and so experiments are performed to obtain observational data. It is not possible to obtain the rates themselves. Instead, the data are observations of certain *outputs* of the pharmacokinetic model, e.g. concentration in blood or urine at certain time points. Crudely put, calibration is the activity of adjusting the unknown rate parameters until the outputs of the model *fit* the observed data.

More generally, a computer model will have a number of *context-specific* inputs that define a particular situation in which the model is to be used. When, as is often the case, the values of one or more of the context-specific inputs are unknown, observations are used to learn about them. This is calibration.

In current practice, calibration invariably consists of searching for a set of values of the unknown inputs such that the observed data fit as closely as possible, in some sense, to the corresponding outputs of the model. These values are considered as estimates of the context-specific inputs, and the model is then used to predict behaviour of the process in this context by setting these inputs to their estimates.

Clearly, this 'plug-in' prediction treats the context-specific inputs as if they were known. The reality is that they are only estimated, and residual uncertainty about these inputs should be recognised in subsequent predictions from the model.

We present in this paper a Bayesian approach to the calibration of computer models. We represent the unknown inputs as a parameter vector $\boldsymbol{\theta}$. Using the observed data we derive the posterior distribution of $\boldsymbol{\theta}$, which in particular quantifies the 'residual uncertainty' about $\boldsymbol{\theta}$. This uncertainty is fully accounted for when using the computer model subsequently for prediction, by obtaining a predictive distribution. The principles of Bayesian predictive inference are set out in Aitchison and Dunsmore (1975), but the problem of computer code calibration has a number of complicating features.

Our approach treats the computer code as simply a 'black box'. We make no use of information about the mathematical model implemented by the code, except insofar as this may be represented through the prior information about the relationship between inputs and outputs. Clearly, methods that open up the 'black box' and exploit its structure might prove to be more powerful than our approach, but would be correspondingly more complex to apply. This is a potentially important topic for future research.

1.2. Outline of this paper

Section 2 provides a detailed analysis of issues associated with computer codes. In particular, there are several other sources of uncertainty in the use of computer codes besides uncertainty about context-specific inputs, and calibration for the purpose of prediction is just one of a number of topics in the statistical analysis of computer code outputs. Section 2 begins with a careful study of uncertainties in computer models, illustrating the possible sources with some detailed examples. It ends with a review of previous work on calibration and related problems.

Our Bayesian method is built on a general Bayesian approach to inference about unknown functions. A review of the relevant theory and published work is given in Section 3. Our basic model and analysis is presented in Section 4, and practical application issues are addressed in Section 5. Section 6 presents a case study based on real data. Section 7 offers some conclusions and directions for further work.

2. STATISTICAL ANALYSIS OF COMPUTER CODE OUTPUTS

2.1. Uncertainties in computer models

The widespread application of computer models is accompanied by a widespread concern about quantifying the uncertainties prevailing in their use. The following is one way of classifying the various sources of uncertainty.

- **Parameter uncertainty**. We have already discussed the matter of uncertainty about the values of some of the computer code inputs. We can think of those inputs as unknown parameters of the model. Generally, they specify features of a particular application context, but they may also be more global parameters, assumed to have a common value over a range of contexts or even in all contexts.

- **Model inadequacy**. No model is perfect. Even if there is no parameter uncertainty, so that we know the true values of all the inputs required to make a particular prediction of the process being modelled, the predicted value will not equal the true value of the process. The discrepancy is model inadequacy. Since the real process may itself exhibit random variability, we define model inadequacy to be the difference between the true *mean* value of the real-world process and the code output at the true values of the inputs. Note that this definition is not precise until we understand what are meant by true values of inputs and the true value of the process. Given that there is model inadequacy, we cannot think of the true input values as those which lead to perfect predictions being output, so how *can* we define true values for the uncertain input parameters? This issue is discussed in Section 4.3.

- **Residual variability**. The model is supposed to predict the value of some real process under conditions specified by the inputs. In practice, the process may not always take the same value if those conditions are repeated. We call this variation of the process even when the conditions are fully specified, residual variability. There are really two sources of uncertainty combined in one here. The process itself may be inherently unpredictable and stochastic, but it may also be that this variation would be eliminated (or at least reduced) if only we were able to recognise and to specify within the model some more conditions. The latter is effectively another form of model inadequacy, where the model lacks detail to discriminate between conditions which actually lead to different process values. However, we define the true process value to be a mean value averaged over these unrecognised conditions, as well as with respect to intrinsic random variation; model inadequacy has been defined relative to this true mean value of the process. The variability due to unrecognised conditions is deemed to be part of residual variability.

- **Parametric variability**. It is often desired to use the model to predict the process when some of the conditions specified in the inputs are uncontrolled and unspecified. In a sense, this is the opposite of the problem of the model inputs being insufficiently detailed, which contributes to residual variability. Here, the inputs require more detail than we desire (or are able) to use. By leaving some of the input parameters unspecified, and allowing them to vary according to an appropriate joint distribution, the predicted process value acquires an extra uncertainty that we will call parametric variability.

- **Observation error**. In tackling the calibration problem, we will be making use of actual observations of the process. We should allow for the possibility of observation error. This adds further uncertainty over and above residual variation, although in practice it may not be feasible to separate them (see Section 4.2).

- **Code uncertainty**. The output of the computer code given any particular configuration of inputs is in practice not known until we actually run it with those inputs. *In principle*, we could say that it is not really unknown because the output is a known function of the inputs. After all, there is a mathematical model which, at least implicitly, defines that function. Nevertheless, *in practice* the relationship is so complex that it needed to be implemented in a computer code (which may even take hours to run), and it is not realistic to say that the output is known for given inputs before we actually run the code and see that output. It may not be practical to run the code to actually observe the output for every input configuration of interest, in which case uncertainty about code output needs to be acknowledged. Examples are given in Section 2.2.

2.2. Examples

It is helpful to be able to relate the various concepts, definitions and notation in this paper to some concrete examples of real computer codes. The following are just two of many examples that could be given.

Gaussian plume model. In the field of radiological protection, a simple Gaussian plume model (Clarke, 1979) is used to predict the dispersion and subsequent deposition of radioactive material following an accidental release. Under these circumstances the detailed input information required to run more complex models is not available.

The code inputs can be divided into those defining the atmospheric conditions at the time of the accident (wind direction, wind speed, atmospheric stability) and those defining the nature of the release (source term, source location, release height, release duration, deposition velocity).

The dispersion of radionuclides is a highly complex process involving various chemical and environmental processes which are not directly observable. Many simplifying assumptions are made in the Gaussian plume model, typically resulting in a high degree of model inadequacy. For example, the speed and direction of the wind are assumed to remain constant during the travel time of the released particles.

Even for this simplified model many of the inputs have parameter uncertainty associated with them. The source term, which represents the amount of material released, and the deposition velocity, which is the rate at which material in the air at ground level is deposited on the ground, are examples. The appropriate value for the deposition velocity is very difficult to determine (see Jones, 1981). Default values are often used based on the size of the particles and the type of terrain over which the material passes. The height and duration of the release, and the wind speed and direction may also have associated parameter uncertainty.

The Gaussian plume model is cheap. We can make many thousands of runs within a very short space of time, so for practical purposes code uncertainty is not an issue and we can treat the function as known. This is not true of more complex atmospheric dispersion models.

During the course of an actual release, measurements of deposition are made by different organisations using a variety of devices. The process of making these measurements is not straightforward. Typically a sample of grass is taken from a site and analysed in a laboratory. Measurement error is introduced at each stage of the measurement process.

The above discussion relates to the use of the plume model in a specific accident scenario where parameter uncertainty represents beliefs about true values of the input parameters for an actual release. We may also want to consider a risk assessment context, where we want to predict possible contamination in the future. An accident could occur at any time, and therefore the source term, wind speed and wind direction are random, and the inputs are subject to parametric variability.

Hydrological model: Hydrological models are used to predict groundwater flow, for example to predict the movement of contaminants in the soil or the level of discharge from a stream after rainfall. We consider the model described in Romanowicz *et al.* (1994), which predicts stream discharge. Inputs to the model include a time series of measured rainfall data, the rate of evapotranspiration, average effective transmissivity of the soil T_0 when the profile is just saturated, and a constant m which is related to the level of subsurface drainage.

Parameter uncertainty about T_0 and m follows from the fact that the model is used to predict water flows through complex geological structures which are not directly observable. The measured rainfall data will include random measurement error, which is an example of parametric variability. Model inadequacy arises from the simplifications introduced by the modelling process, as with the Gaussian plume model.

Romanowicz *et al.* (1994) use measurements of actual stream discharge to learn about the values of m and T_0. However, as they point out it is not possible to measure the true value of the flow process, since we can only make point measurements of the heterogeneous pattern of flow. This relates to the idea of residual variability described in Section 2.1, and we would define the 'true flow' to be an averaged value.

2.3. *Statistical methods and previous work*

Some of the early work in the field of statistical analysis of computer code outputs was primarily concerned with *interpolation.* That is, given data comprising outputs at a sample of input configurations, the problem is to estimate the output at some other input configuration for which the code has not yet been run. This is relevant when the code is particularly large and expensive to run. An important review of this work is Sacks, Welch, Mitchell and Wynn (1989), and some more recent references are Currin *et al.* (1991), Morris *et al.* (1993), Bates *et al.* (1995) and Kennedy and O'Hagan (2000a).

The only form of uncertainty accounted for in this work is code uncertainty. Model inadequacy, residual variation and observation error are not relevant because there is no attempt to predict the real process as distinct from the computer model output, and observations of that process are not used. All input parameters are supposed known, and not subject to parameter uncertainty or parametric variation. The statistical approach used in this work is based on representing the computer code output as an unknown function of its inputs, and modelling that function as a stochastic process.

Another problem that has been tackled by statistical methods for some time is that of *uncertainty analysis.* The objective of uncertainty analysis is to study the distribution of the code output that is induced by probability distributions on inputs. The input parameter distributions may be formulations of parameter uncertainty, i.e. parameters whose values are unknown, or of parametric variability, i.e. parameters whose values are left unspecified. The simplest approach to uncertainty analysis is a Monte Carlo solution in which configurations of inputs are drawn at random from their distribution. The code is then run for each sample input configuration and the resulting set of outputs is a random sample from the output distribution to be studied. See Helton (1993) for a review.

The Monte Carlo method for uncertainty analysis is simple but becomes impractical when the code is costly to run, because of the large number of runs required. More efficiency is claimed for Latin Hypercube sampling (McKay *et al.*, 1979; Stein, 1987; Owen, 1992) compared with simple Monte Carlo, see for example Crick *et al.* (1988) and Helton *et al.* (1991). Aslett *et al.* (1998) combine a Monte Carlo approach to uncertainty analysis with statistical interpolation of the code, effectively using the interpolator as a cheap surrogate for the code.

Haylock and O'Hagan (1996) present a quite different Bayesian approach to uncertainty analysis based on a Gaussian process prior model. They derive the posterior mean and variance of the output distribution, and this is extended to posterior estimation of the distribution function and the density function of the output distribution by Oakley and O'Hagan (1998).

These methods of uncertainty analysis take account of parameter uncertainty and parametric variation in addition to code uncertainty. However, the objective is still focussed on the code output, in this case in the form of the output distribution, rather than on the process itself. There is therefore no treatment of model inadequacy, residual variation or observation error.

Another problem associated with computer codes is *sensitivity analysis*, whose goal is to characterise how the code output responds to changes in the inputs, with particular reference to identifying inputs to which the output is relatively sensitive or insensitive. A good source for the large literature on this subject is Saltelli *et al.* (2000). Whilst much of this makes no use of statistical methods there are some notable exceptions. See for example Helton (1993), Morris (1991), Iman and Conover (1980), Welch *et al.* (1992), Morris (1991), Homma and Saltelli (1996), and O'Hagan *et al.* (1999). In Draper *et al.* (1999), sensitivity analysis is applied across a range of 'scenarios', using the model averaging ideas of Draper (1995). As with interpolation, these statistical approaches to sensitivity analysis only take account of the source of uncertainty common to all statistical analysis of computer code outputs, namely code uncertainty.

The final topic in our survey of this field is *calibration.* As discussed in Section 1.1, the traditional way of estimating unknown parameters is by an ad hoc search for the best fitting values. Some account is thereby taken of observation error, residual variation and model inadequacy, but only implicitly through the measure of fit discrepancy. This measure is not generally developed by modelling these error terms in any explicit way, and is usually entirely heuristic. Also, since the estimated values are then treated as if they were known, the subsequent predictions take no account of the (remaining) parameter uncertainty.

In contrast, the GLUE (Generalised Likelihood Uncertainty Estimation) method of Romanowicz *et al.* (1994) does allow fully for parameter uncertainty. The approach is effectively Bayesian. An initial Monte Carlo sample is drawn from what amounts to the prior distribution of the unknown inputs, and is then weighted by a likelihood term. Predictions are made using all the sampled input configurations, and the result is a weighted sample from the posterior predictive distribution. For instance, the weighted sample mean is an estimate of the mean of this predictive distribution.

It would be possible within the GLUE method to allow for the code uncertainty arising from having only a sample of runs, also to allow for parametric variation by extending the Monte Carlo method to allow for draws from the unspecified inputs at the prediction stage, but these are not done in the published literature to date. As in the more traditional calibration approach, the likelihood is rather heuristic and only very loosely based on modelling the discrepancy between code outputs and the real process. Model inadequacy, residual variation and observation errors are not distinguished or modelled explicitly. No account is taken of them in prediction, the objective still being to estimate the code output rather than reality.

Another Bayesian approach to calibration is given by Craig *et al.* (1996, 1999). They employ modelling for the relationship between code inputs and output that is akin to the Gaussian process model mentioned in connection with much other work in this field. In Craig *et al.* (1996), the primary objective is to make the search for a best fit calibration more efficient and systematic. Their modelling reflects the iterative nature of the search, and their methods adopt the Bayes Linear philosophy of Goldstein (1986, 1988) and Wooff (1992), as opposed to a fully specified Bayesian analysis. The approach is extended in Craig *et al.* (1999) to treat the question of prediction following calibration.

Cox *et al.* (1992) describe a calibration method that is similar to the traditional search for best fitting parameters, but which replaces an expensive code with the much cheaper interpolator obtained using the Gaussian process model. Their method does not account for remaining parameter uncertainty at the prediction stage. See also Cox *et al.* (1996).

An attempt to combine prior expert opinion on both the calibration parameters and the model output is given by Raftery, Givens and Zeh (1995) using an approach they call Bayesian synthesis. This was criticised by Wolpert (1995) and Schweder and Hjort (1996), and in a follow-up paper, Poole and Raftery (1998) propose an alternative called Bayesian melding. Neither method explicitly recognises model inadequacy and the underlying computer code is supposed to be simple enough for code uncertainty to be ignored.

Our analysis in the present paper is the first attempt to model, and to take account of explicitly, *all* the sources of uncertainty arising in the calibration and subsequent use of computer models. We freely acknowledge that our method as currently implemented is not fully Bayesian, because we estimate hyperparameters by (approximate) posterior modes. To take full account of uncertainty in hyperparameters is another topic for further research, but we believe that our present methodology is the fullest treatment to date of computer code uncertainties, and argue that it may be adequate in practice.

3. BAYESIAN INFERENCE FOR FUNCTIONS

3.1. *Gaussian processes*

The use of Gaussian processes has been mentioned several times in Section 2.3. They are being increasingly used in current statistical research, and will be employed in this paper to model both the computer code output and model inadequacy. It is therefore appropriate to review the key theoretical and practical issues, together with some discussion of alternative models.

Let $f(\cdot)$ be a function mapping an input $\boldsymbol{x} \in \mathcal{X}$ into an output $y = f(\boldsymbol{x})$ in $\mathbb{R}$. The input space $\mathcal{X}$ can be arbitrary, but is typically a subset of $\mathbb{R}^q$ for some q, so that we can write $\boldsymbol{x}$ as a vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_q)$. We constrain y to be a scalar, $y \in \mathbb{R}$, for simplicity here and throughout this paper, but it is equally possible for y to be a vector in $\mathbb{R}^{q'}$. We regard $f(\cdot)$ as an unknown function, and in a Bayesian framework it therefore becomes a random function. The Gaussian process is a flexible and convenient class of distributions to represent prior knowledge about $f(\cdot)$. In a non-Bayesian framework, we might treat $f(\cdot)$ as if it were drawn randomly from some population of functions, and postulate a Gaussian process model for the distribution of functions in that population. This is indeed the implied interpretation of the non-Bayesian work in Sacks, Welch, Mitchell and Wynn (1989), for instance, although these authors are clearly also aware of the Bayesian interpretation. In the present authors' opinion, the Bayesian interpretation is far more natural, and will be used throughout this paper.

Formally, $f(\cdot)$ has a Gaussian process distribution if for every $n = 1, 2, 3, \ldots$ the joint distribution of $f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)$ is multivariate normal for all $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathcal{X}$. In particular, $f(\boldsymbol{x})$ is normally distributed for all $\boldsymbol{x} \in \mathcal{X}$.

The distribution is characterised by its mean function $m(\cdot)$, where $m(\boldsymbol{x}) = E\{f(\boldsymbol{x})\}$, and its covariance function $c(\cdot, \cdot)$, where $c(\boldsymbol{x}, \boldsymbol{x}') = \text{cov}\{f(\boldsymbol{x}), f(\boldsymbol{x}')\}$. We use the notation $f(\cdot) \sim N\big(m(\cdot), c(\cdot, \cdot)\big)$ to denote the assertion that $f(\cdot)$ has a Gaussian process distribution with mean function $m(\cdot)$ and covariance function $c(\cdot, \cdot)$.

The use of Gaussian processes to represent prior distributions (or frequentist models) for unknown functions dates back at least to Kimeldorf and Wahba (1970) and O'Hagan (1978). Both used the Gaussian process effectively to model a regression function in a nonparametric way. Although O'Hagan (1978) gave a very general treatment of a Gaussian process regression function, the full potential for Gaussian process models did not begin to be exploited until much later.

As we have seen in Section 2.3, their use to represent deterministic computer codes was described by Sacks, Welch, Mitchell and Wynn (1989), who reviewed work of this kind over a period of several years. Independently, Diaconis (1988) and O'Hagan (1991, 1992) described their application for arbitrary deterministic functions and for the traditional problems of numerical analysis—interpolation, integration, optimisation. It is worth noting that the numerical analysis literature also includes work on methods that are optimal for randomly generated functions, including Gaussian processes. See Novak (1988), for example.

The Gaussian process also underlies, implicitly or explicitly, the methods of geostatistics, also known as kriging. See Matheron (1963) for the classical non-Bayesian theory and Omre (1987), Handcock and Stein (1993) for Bayesian versions in which the Gaussian process appears explicitly as a prior distribution. In geostatistics, the $\mathcal{X}$ space is geographic, usually two-dimensional but occasionally three-dimensional. The function $f(\cdot)$ represents some characteristic that might be measured at any point in some geographic region. A major concern in geostatistics is estimation of the covariance function or, equivalently, the semi-variogram. See in particular Stein (1999). Other examples of the use of Gaussian processes to model unknown functions are reviewed in a recent paper by Neal (1999).

3.2. *Modelling issues*

The Gaussian process is used in practice for much the same reasons that normal distributions are used so ubiquitously in statistical theory and modelling. They are convenient, flexible and often quite realistic. It is, of course, important that normality, and specifically joint normality, is a reasonable representation of prior knowledge or beliefs about

$f(\cdot)$. Transformation may be useful in this context, just as it is in other applications of normal-theory models.

Given that a Gaussian process is a reasonable modelling choice, the mean and covariance functions should then be specified to reflect detailed prior knowledge about $f(\cdot)$. For instance, if stationarity was a feature of prior beliefs, so that the prior distribution of $f(\boldsymbol{x})$ is the same as that of $f(\boldsymbol{x}+\boldsymbol{d})$ for any $\boldsymbol{d} \in \mathcal{X}$ (and where the operation of addition on $\mathcal{X}$ is defined), $m(\cdot)$ will be a constant and $c(\boldsymbol{x},\boldsymbol{x}')$ a function only of $\boldsymbol{x}-\boldsymbol{x}'$.

In general, $m(\cdot)$ may be any function on $\mathcal{X}$ but $c(\cdot,\cdot)$ must have the property that for every $n = 1, 2, \ldots$ the variance-covariance matrix of $f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_n)$ (comprising elements $c(\boldsymbol{x}_i,\boldsymbol{x}_j)$) is non-negative definite for all $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n \in \mathcal{X}$. Some conditions for a covariance function to be non-negative in this sense are given in Cressie (1991).

A useful device is to model $m(\cdot)$ and $c(\cdot,\cdot)$ hierarchically. In the case of $m(\cdot)$ we can use the linear model structure

$$m(\cdot) = \boldsymbol{h}(\cdot)^T\boldsymbol{\beta}, \tag{1}$$

where $\boldsymbol{h}(\cdot) = (h_1(\cdot), h_2(\cdot), \ldots, h_p(\cdot))^T$ is a vector of p known functions over $\mathcal{X}$ and $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)^T$ is a vector of p unknown coefficients which are given a prior distribution at the next stage in the hierarchy. Thus $\boldsymbol{h}(\cdot)$ describes a class of shapes and (1) expresses a belief that $f(\cdot)$ may be approximated by a function in this class. For instance $\boldsymbol{h}(x) = (1, x, \ldots, x^{p-1})^T$ defines $m(\cdot)$ to be a polynomial of degree $p-1$ when x is a scalar.

As a prior distribution for $\boldsymbol{\beta}$, the multivariate normal distribution is a convenient choice. For instance, it has the property that $f(\cdot)$ remains a Gaussian process marginally after integrating out $\boldsymbol{\beta}$. Of course, the prior distribution should be specified to reflect genuine belief rather than convenience, but in practice prior information about hyperparameters such as $\boldsymbol{\beta}$ will typically be weak. The conventional representation of weak prior information through the improper uniform density $p(\boldsymbol{\beta}) \propto 1$ is therefore often used.

Notice that we can separate the mean and covariance structures by writing

$$f(\boldsymbol{x}) = m(\boldsymbol{x}) + e(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{x})^T\boldsymbol{\beta} + e(\boldsymbol{x}) \tag{2}$$

using (1), where $e(\boldsymbol{x})$ is a zero mean Gaussian process, with covariance function $c(\cdot,\cdot)$. The modelling of $c(\cdot,\cdot)$ is very important because it is through correlation between $f(\boldsymbol{x})$ and $f(\boldsymbol{x}')$ that we express a view that $f(\boldsymbol{x})$ and $f(\boldsymbol{x}')$ should be similar if $\boldsymbol{x}$ and $\boldsymbol{x}'$ are sufficiently close in $\mathcal{X}$, and thereby express a belief in smoothness of $f(\cdot)$. In the following sections we shall generally adopt a hierarchical model for $c(\cdot,\cdot)$ with first stage

$$c(\cdot,\cdot) = \sigma^2 r(\boldsymbol{x}-\boldsymbol{x}'), \tag{3}$$

where $r(\cdot)$ is a correlation function having the property $r(0) = 1$. This formulation expresses stationarity in prior information about $e(\cdot)$; we have a common (unknown) variance σ^2 and correlation that only depends on $\boldsymbol{x}-\boldsymbol{x}'$. The correlation function is further expressed in terms of other unknown hyperparameters; for instance if $\boldsymbol{x} = (x_1, \ldots, x_q)$ is a vector,

$$r(\boldsymbol{x}-\boldsymbol{x}') = \exp\{-\sum_{j=1}^{q} \omega_j (x_j - x_j')^2\}. \tag{4}$$

Then at the next stage we would express prior distributions for the variance σ^2 and the roughness parameters $\omega_1, \ldots, \omega_q$.

Of course (4) is just one possible formulation. A more general expression that has been widely used replaces $(x_j - x_j')^2$ by $|x_j - x_j'|^\alpha$, where α may have a specified value or be another hyperparameter. Even more generally, we can allow a different α_j in each dimension. Another generalisation of (4) would be to set $r(\boldsymbol{d}) = \exp(-\boldsymbol{d}^T \boldsymbol{\Omega} \boldsymbol{d})$, where $\boldsymbol{\Omega}$ is an unknown symmetric positive-definite matrix, which in (4) has the form $\boldsymbol{\Omega} = \mathrm{diag}(\omega_1, \ldots, \omega_q)$.

In geostatistics it is normal to invest considerable effort in estimating $\sigma^2 r(\boldsymbol{x} - \boldsymbol{x}')$, or equivalently $\sigma^2\{r(0) - r(\boldsymbol{x} - \boldsymbol{x}')\}$, which is there known as the *semi-variogram.* The geostatistics literature contains proposals for a wide range of semi-variogram forms. See for instance Cressie (1991), Handcock and Wallis (1994), Stein (1999), Chilès and Delfiner (1999).

3.3. Other Bayesian nonparametric models

The Gaussian process is a flexible and popular form for prior information about functions. It is of course possible to model an unknown function parametrically, asserting that it is definitely a member of some parametric family, so that prior information is expressed as a prior distribution for the parameters. An obvious example is representing a regression function parametrically in a linear model. In contrast, Gaussian processes are nonparametric because they do not constrain the function to be in a specific parametric family. The hierarchical form may, however, be viewed via (2) as *semi*-parametric, combining a parametric underlying mean $(\boldsymbol{h}(\cdot)^T \boldsymbol{\beta})$ with a Gaussian process residual term $(e(\cdot))$.

The literature of Bayesian nonparametric and semiparametric methods is growing rapidly. In addition to work using Gaussian processes, already referred to, various other ways have been proposed to express prior beliefs about functions. One alternative to Gaussian processes for modelling general functions is to represent the function as a linear combination of components of a set of basis functions. Smith and Kohn (1998) discuss this generally, contrasting splines and other bases. In the same volume of Dey *et al.* (1998), Vidakovic (1998) considers wavelet bases and Rios Insua and Müller (1998) deal with neural networks, which employ sigmoidal basis functions. We can note here two connections between Gaussian processes and the basis function approach. First, Neal (1996) has shown that Gaussian processes can be viewed as neural networks with an infinite number of hidden nodes. Second, the posterior mean from a Gaussian process is a linear combination of the set of basis functions $c(\boldsymbol{x}, \boldsymbol{x}_i)$ formed by the correlation functions centred at the observations.

Another approach to modelling general functions is given by Liu and Arjas (1998), who model a growth curve by a piecewise linear process. All of these approaches may be applied to other kinds of functions such as computer code outputs.

There is also a wide literature on more specialised models for unknown distributions, see other papers in Dey *et al.* (1998) and the review of Walker *et al.* (1999). However, these are not directly relevant to computer code outputs.

4. BAYESIAN CALIBRATION

4.1. Calibration and variable inputs

In the calibration problem we must distinguish between two groups of inputs to the computer model. One group comprise the unknown context-specific inputs that we wish to learn about; we refer to these as the calibration inputs. The calibration inputs are supposed to take fixed but unknown values $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{q_2})$ for all the observations that will be used for calibration, and for all the instances of the true process that we wish to use the calibrated model to predict. The other group comprises all the other model inputs whose values might change when we use the calibrated model. These are referred to as the variable inputs. The variable inputs are assumed to have known values for each of the observations that will be used for calibration. In any subsequent use of the model their values will either be known or subject to parametric variability. For example in Section 2.2, variable inputs include the (x, y) coordinates of the Gaussian plume model or rainfall levels in the hydrological model.

We denote the output of the computer model when the variable inputs are given values $\boldsymbol{x} = (x_1, \ldots, x_{q_1})$ and when the calibration inputs are given values $\boldsymbol{t} = (t_1, \ldots, t_{q_2})$ by $\eta(\boldsymbol{x}, \boldsymbol{t})$. Notice that we distinguish between the unknown value $\boldsymbol{\theta}$ of the unknown calibration inputs, that corresponds to the particular real process for which we wish to calibrate the model, from the (known) value $\boldsymbol{t}$ that one sets as inputs when running the model. We never observe output from the model without knowing all the inputs specifying that run. We refer to $\boldsymbol{\theta}$ as the (vector of) *calibration parameters.*

We denote the true value of the real process when the variable inputs take values $\boldsymbol{x}$ by $\zeta(\boldsymbol{x})$. Notice that only the variable inputs are needed here. For the computer code we can vary the calibration inputs but they are fixed for the real process.

The calibration data comprise the n observations $\boldsymbol{z} = (z_1, \ldots, z_n)^T$, where z_i is an observation of $\zeta(\boldsymbol{x}_i)$ for known variable inputs $\boldsymbol{x}_i$, but subject to error. In addition, we have the outputs $\boldsymbol{y} = (y_1, \ldots, y_N)^T$ from N runs of the computer code, where

$$y_j = \eta(\boldsymbol{x}_j^*, \boldsymbol{t}_j)$$

and both the variable inputs $\boldsymbol{x}_j^*$ and the calibration inputs $\boldsymbol{t}_j$ are known for each run. The full set of data available for the analysis is $\boldsymbol{d}^T = (\boldsymbol{y}^T, \boldsymbol{z}^T)$. Note that generally N will be much larger than n, since even if the computer code is expensive or time-consuming to run it will still be much cheaper than obtaining observations of the real process.

A further comment in connection with variable inputs is that we are treating the computer code output for any given set of inputs as a scalar. In practice, computer codes typically produce many outputs for a given run. It is not generally necessary, however, to regard the output as being multivariate. The reason is that we can define one or more variable inputs to index the outputs. For instance, a Gaussian plume model will typically be implemented in a computer code that for a given run, outputs concentrations at every point in a two dimensional grid. Rather than thinking of this as a highly multivariate output, we can define two variable inputs to index a given point on the grid. We can think of the variable inputs as being augmented by these two new indexing inputs, and a run as producing a single output at the specified point.

4.2. Model

We represent the relationship between the observations z_i, the true process $\zeta(\cdot)$ and the computer model output $\eta(\cdot, \cdot)$ in the equation

$$z_i = \zeta(\boldsymbol{x}_i) + e_i = \rho\eta(\boldsymbol{x}_i, \boldsymbol{\theta}) + \delta(\boldsymbol{x}_i) + e_i, \tag{5}$$

where e_i is the observation error for the ith observation, ρ is an unknown regression parameter and $\delta(\cdot)$ is a model inadequacy function that is *independent* of the code output $\eta(\cdot, \cdot)$.

Consider the observation error e_i first. Strictly, this also includes any residual variation as well as observation error (see Section 2.1). We do not imagine having replication of observations in circumstances where not only the variable inputs but also all the unrecognised conditions were the same, so it is not possible to separate the two sources of uncertainty.

We will suppose that the e_is are independently distributed as $N(0, \lambda)$. (The assumption of normality may require a transformation of the raw data, as in our use of log-deposition in Section 6.)

Now consider the implication of (5) that

$$\zeta(\boldsymbol{x}) = \rho\eta(\boldsymbol{x}, \boldsymbol{\theta}) + \delta(\boldsymbol{x}), \tag{6}$$

with $\eta(\cdot, \cdot)$ and $\delta(\cdot)$ independent. This is, of course, merely one way of modelling the relationship between the code output and reality. As a partial justification, it can be given a formal derivation from a kind of Markov property, as follows. Assume that we know $\boldsymbol{\theta}$ and can make as many runs of the code as we wish, to observe $\eta(\boldsymbol{x}, \boldsymbol{\theta})$ for various different $\boldsymbol{x}$. Suppose first that in order to predict $\zeta(\boldsymbol{x}')$ at some specific point $\boldsymbol{x}'$ we would regard it as sufficient to observe the output $\eta(\boldsymbol{x}', \boldsymbol{\theta})$ of a single run at the same value $\boldsymbol{x}'$ of the variable inputs. This is the Markov assumption, and can be shown to imply (6) (see O'Hagan, 1998), except that ρ may formally depend on $\boldsymbol{x}$. The further assumption that ρ is constant seems natural and follows if we have *stationary* processes $\eta(\cdot, \cdot)$, $\delta(\cdot)$ and $\zeta(\cdot)$. Despite this argument, we repeat that (6) is just one possible formulation; equally cogent arguments could probably be evinced in favour of other models. In the examples that we have tried, (6) does seem reasonable, but more experience is needed to explore this aspect of modelling.

The meaning of the *true* values $\boldsymbol{\theta}$ of the calibration parameters is addressed in Section 4.3 below.

We represent prior information about both the unknown functions $\eta(\cdot, \cdot)$ and $\delta(\cdot)$ by Gaussian processes: $\eta(\cdot, \cdot) \sim N\big(m_1(\cdot, \cdot), c_1((\cdot, \cdot), (\cdot, \cdot))\big)$ and $\delta(\cdot) \sim N\big(m_2(\cdot), c_2(\cdot, \cdot)\big)$. In each case, we model the mean and variance functions hierarchically as in Section 3.2. Adopting the linear model form (1) with weak prior distributions, we have $m_1(\boldsymbol{x}, \boldsymbol{t}) = \boldsymbol{h}_1(\boldsymbol{x}, \boldsymbol{t})^T \boldsymbol{\beta}_1, m_2(\boldsymbol{x}) = \boldsymbol{h}_2(\boldsymbol{x})^T \boldsymbol{\beta}_2$, and

$$p(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2) \propto 1. \tag{7}$$

We write the combined vector as $\boldsymbol{\beta} = (\boldsymbol{\beta}_1^T, \boldsymbol{\beta}_2^T)^T$. For the covariance functions we will not specify any particular forms at present, but will suppose that they are expressed in terms of some further hyperparameters that we denote by $\boldsymbol{\psi}$. We also denote $(\rho, \lambda, \boldsymbol{\psi})$ collectively by $\boldsymbol{\phi}$. The complete set of parameters therefore comprises the calibration parameters $\boldsymbol{\theta}$, the location parameters $\boldsymbol{\beta}$ and the hyperparameters $\boldsymbol{\phi}$. It is reasonable to suppose that prior information about $\boldsymbol{\theta}$ is independent of the others, and with (7) we suppose that the prior distribution takes the form

$$p(\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\phi}) = p(\boldsymbol{\theta})p(\boldsymbol{\phi}). \tag{8}$$

4.3. True parameter values

We raised the question of the meaning of true parameter values in Section 2.1. We now discuss this important topic in the context of our model (6) for the true process $\zeta(\cdot)$, although the main points are relevant to any discussion of the calibration of computer models.

Our statistical model is formulated through the equation (5), which can be viewed as defining a non-linear regression model. (The analogy is far from perfect, but provides some useful insight.) The computer code itself defines the regression function through the term $\rho\eta(\boldsymbol{x}_i, \boldsymbol{\theta})$, with parameters ρ and $\boldsymbol{\theta}$. The other two terms can be viewed as together representing (non-independent) residuals.

In this framework, we can see that the notion of a true value for $\boldsymbol{\theta}$ has the same meaning and validity as the true values of regression parameters. The true $\boldsymbol{\theta}$ is a 'best-fitting' $\boldsymbol{\theta}$, in the sense of representing the data $z_1, \ldots, z_n$ faithfully according to the error structure specified for the residuals.

Now the developers of the computer model will generally have given concrete physical meanings to the calibration inputs, but the true values of these physical quantities do not necessarily equate to $\boldsymbol{\theta}$. This is inevitable in calibration when we do not believe that the model can ever be a perfect fit. It may be that the physically true value of a calibration parameter gives a worse fit, and less accurate future prediction, than another value. It is dangerous to interpret the estimates of $\boldsymbol{\theta}$ obtained by calibration as estimates of the true *physical* values of those parameters.

In regression modelling, assuming one of the parameters known corresponds to fitting a simpler, more limited class of regression functions. Fixing a parameter constrains the form of the regression function, whereas adding more unknown parameters increases the flexibility of the class of regression functions and allows a better fit to the data. By analogy, we see that if we claim to know the value of one of the calibration parameters in $\boldsymbol{\theta}$, even if this is genuinely the true physical value of that parameter, we restrict the form of the code output and may have a worse fit to the data. The discrepancy will of course be taken up by the model inadequacy function $\delta(\cdot)$, but may also lead to the calibrated estimates of other components of $\boldsymbol{\theta}$ being further from their true physical interpretations. We will generally have far fewer real observations with which to estimate $\delta(\cdot)$ than code outputs with which to estimate $\eta(\cdot, \cdot)$, so it makes sense to make the code fit as well as possible.

So it may be reasonable to treat an input as unknown, and therefore part of the calibration parameter $\boldsymbol{\theta}$, even if we believe its true physical value to be known. Particularly if this is an influential parameter (as might be revealed by a sensitivity analysis), allowing it to deviate from the true physical value may produce an empirically better computer model of reality. Its prior distribution would be centered at the true physical value, reflecting an expectation of the model's accuracy, but with a non-zero variance.

All models are wrong, and to suppose that inputs should always be set to their 'true' values when these are 'known' is to invest the model with too much credibility in practice. Treating a model more pragmatically, as having inputs that one can 'tweak' empirically, can increase its value and predictive power.

4.4. *Posterior distribution*

In the remaining subsections of this section, we present the posterior analysis of the calibration problem and subsequent prediction of the true phenomenon using the calibrated code. To save space, we give only an outline of the development here. For fuller mathematical details the reader is referred to Kennedy and O'Hagan (2000b).

The first step is to derive the posterior distribution of the parameters $\boldsymbol{\theta}, \boldsymbol{\beta}$ and $\boldsymbol{\phi}$. The full data vector $\boldsymbol{d}$ is normally distributed given $(\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\phi})$, and this will yield the likelihood function. To express its mean vector and variance matrix we require some more notation.

We denote the set of points at which the code outputs $\boldsymbol{y}$ are available by $D_1 = \{(\boldsymbol{x}_1^*, \boldsymbol{t}_1), \ldots, (\boldsymbol{x}_N^*, \boldsymbol{t}_N)\}$. Similarly, we denote the set of points for the observations $\boldsymbol{z}$ of the real process by $D_2 = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. Augmenting each of these points by the calibration parameters $\boldsymbol{\theta}$, we define $D_2(\boldsymbol{\theta}) = \{(\boldsymbol{x}_1, \boldsymbol{\theta}), \ldots, (\boldsymbol{x}_n, \boldsymbol{\theta})\}$. If we now let $\boldsymbol{H}_1(D_1)$ denote the matrix with rows $\boldsymbol{h}_1(\boldsymbol{x}_1^*, \boldsymbol{t}_1)^T, \ldots, \boldsymbol{h}_1(\boldsymbol{x}_N^*, \boldsymbol{t}_N)^T$, the expectation of $\boldsymbol{y}$ is $\boldsymbol{H}_1(D_1)\boldsymbol{\beta}_1$. In analogous notation, the expectation of $\boldsymbol{z}$ is $\rho\boldsymbol{H}_1(D_2(\boldsymbol{\theta}))\boldsymbol{\beta}_1 + \boldsymbol{H}_2(D_2)\boldsymbol{\beta}_2$. Hence

$$E(\boldsymbol{d} \mid \boldsymbol{\theta}, \boldsymbol{\beta}, \phi) = \boldsymbol{m}_d(\boldsymbol{\theta}) = \boldsymbol{H}(\boldsymbol{\theta})\boldsymbol{\beta},$$

where

$$\boldsymbol{H}(\boldsymbol{\theta}) = \begin{pmatrix} \boldsymbol{H}_1(D_1) & \boldsymbol{0} \\ \rho\boldsymbol{H}_1(D_2(\boldsymbol{\theta})) & \boldsymbol{H}_2(D_2) \end{pmatrix}.$$

To express the variance matrix of $\boldsymbol{d}$, define $\boldsymbol{V}_1(D_1)$ to be the matrix with (j, j') element $c_1((\boldsymbol{x}_j^*, \boldsymbol{t}_j), (\boldsymbol{x}_{j'}^*, \boldsymbol{t}_{j'}))$, so that this is the variance matrix of $\boldsymbol{y}$. Define $\boldsymbol{V}_1(D_2(\boldsymbol{\theta}))$ and $\boldsymbol{V}_2(D_2)$ similarly, and let $\boldsymbol{C}_1(D_1, D_2(\boldsymbol{\theta}))$ be the matrix with (j, i) element $c_1((\boldsymbol{x}_j^*, \boldsymbol{t}_j), (\boldsymbol{x}_i, \boldsymbol{\theta}))$. Then

$$\mathrm{var}(\boldsymbol{d} \mid \boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\phi}) = \boldsymbol{V}_d(\boldsymbol{\theta}) = \begin{pmatrix} \boldsymbol{V}_1(D_1) & \rho\boldsymbol{C}_1(D_1, D_2(\boldsymbol{\theta}))^T \\ \rho\boldsymbol{C}_1(D_1, D_2(\boldsymbol{\theta})) & \lambda\boldsymbol{I}_n + \rho^2\boldsymbol{V}_1(D_2(\boldsymbol{\theta})) + \boldsymbol{V}_2(D_2) \end{pmatrix}$$

where $\boldsymbol{I}_n$ is the $n \times n$ identity matrix.

With prior distribution (8) we now obtain the full joint posterior distribution

$$p(\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\phi} \mid \boldsymbol{d}) \propto p(\boldsymbol{\theta})p(\boldsymbol{\phi})f(\boldsymbol{d}; \boldsymbol{m}_d(\boldsymbol{\theta}), V_d(\boldsymbol{\theta})), \tag{9}$$

where $f(\cdot; \boldsymbol{m}_d(\boldsymbol{\theta}), V_d(\boldsymbol{\theta}))$ is the $N(\boldsymbol{m}_d(\boldsymbol{\theta}), V_d(\boldsymbol{\theta}))$ density function. Note that we have explicitly shown dependence on $\boldsymbol{\theta}$ but $\boldsymbol{m}_d(\boldsymbol{\theta})$ also depends on $\boldsymbol{\beta}$ and ρ, while $\boldsymbol{V}_d(\boldsymbol{\theta})$ depends on all of $\boldsymbol{\phi}$.

4.5. Estimating hyperparameters

Since the exponent of (9) is quadratic in $\boldsymbol{\beta}$, we can integrate $\boldsymbol{\beta}$ out analytically to give $p(\boldsymbol{\theta}, \boldsymbol{\phi} \mid \boldsymbol{d})$. This, however, is an even more complex function of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ than (9). A fully Bayesian analysis would now integrate out the hyperparameters $\boldsymbol{\phi}$ as well to leave the posterior distribution $p(\boldsymbol{\theta} \mid \boldsymbol{d})$ of the calibration parameters. However, $p(\boldsymbol{\theta}, \boldsymbol{\phi} \mid \boldsymbol{d})$ is a highly intractable function of $\boldsymbol{\phi}$. Even with the most parsimonious parametrisation of $c_1((\cdot,\cdot),(\cdot,\cdot))$ and $c_2(\cdot,\cdot)$, to integrate over $\boldsymbol{\phi}$ numerically would entail at least a six dimensional quadrature. Since much of the methodology that we develop herein may be rather computationally intensive even conditional on fixed values of $\boldsymbol{\phi}$, the full Bayesian analysis will not typically be practical. It is important to note also that $p(\boldsymbol{\theta}, \boldsymbol{\phi} \mid \boldsymbol{d})$ will generally be improper with respect to $\boldsymbol{\phi}$ if $p(\boldsymbol{\phi})$ is improper. To adopt a fully Bayesian analysis will therefore demand full and careful consideration of prior information regarding the hyperparameters.

We propose instead to derive plausible estimates of the components of $\boldsymbol{\phi}$ and then to act as if these were fixed. Thus, for inference about $\boldsymbol{\theta}$ we will use its conditional posterior given the estimated values of $\boldsymbol{\phi}$. We propose estimating the hyperparameters in two stages. In the first stage we use just the code output data $\boldsymbol{y}$ to estimate the hyperparameters $\boldsymbol{\psi}_1$ of $c_1((\cdot,\cdot),(\cdot,\cdot))$. There is some information about $\boldsymbol{\psi}_1$ in the observational data $\boldsymbol{z}$, but (a) $\boldsymbol{z}$ depends also on the other hyperparameters and (b) the number n of observations in $\boldsymbol{z}$ will typically be very much smaller than the number N of output values in $\boldsymbol{y}$. Therefore very little is lost by this simplification. In the second stage we use $\boldsymbol{z}$ to estimate ρ, λ and the hyperparameters $\boldsymbol{\psi}_2$ of $c_2(\cdot,\cdot)$, having now fixed $\boldsymbol{\psi}_1$.

Now we originally set out to model, and account for explicitly, all the sources of uncertainty identified in Section 2.1. The compromise proposed here means that we do not account *fully* for all these sources.

- By fixing λ at an estimated value, we do not account *fully* for observation error and residual uncertainty.
- By fixing ρ and the hyperparameters $\boldsymbol{\psi}_2$ of $c_2(\cdot,\cdot)$ at estimated values, we do not account *fully* for model inadequacy.
- By fixing the hyperparameters $\boldsymbol{\psi}_1$ of $c_1((\cdot,\cdot),(\cdot,\cdot))$ at estimated values, we do not account *fully* for code uncertainty.

Nevertheless, we should stress that in each case it is only the 'second order' effect of uncertainty about hyperparameters that is neglected, and we believe that our analysis captures the major part of all these sources of uncertainty. We therefore claim that our analysis *does* recognise *all* sources of uncertainty, and that it is more important to ensure that all sources are covered to this extent than to account for any missing hyperparameter uncertainty, at the cost of very much increased computation.

In geostatistics, it is generally recognised that kriging estimates are reasonably robust to the form of the covariance function, and even to roughness parameters in that function, but that the prediction variance will typically be very sensitive to roughness parameter values. Furthermore, such parameters are notoriously difficult to estimate. We are conscious that these considerations may give cause for concern about our treatment of hyperparameters and the choice of covariance structure. In our defence, we show in the example in Section 6 that our predictive variances calibrate well with held-back data, and this has been our experience with other examples, too. In Section 6.3 we present an investigation which shows for those data that the effect of acknowledging uncertainty in roughness parameters is small.

4.6. Calibration, prediction and uncertainty analysis

Having estimated the hyperparameters $\boldsymbol{\phi}$ we now condition on the estimates $\hat{\boldsymbol{\phi}}$, so that we regard the posterior distribution of the calibration parameters to be $p(\boldsymbol{\theta} \mid \boldsymbol{\phi} = \hat{\boldsymbol{\phi}}, \boldsymbol{d}) \propto p(\boldsymbol{\theta}, \hat{\boldsymbol{\phi}} \mid \boldsymbol{d})$. We can use this to make inference about $\boldsymbol{\theta}$, although its intractability means that numerical methods must be used. We discuss appropriate techniques in Section 5.

In practice, we will not generally be interested in inference about $\boldsymbol{\theta}$ as such. The purpose of calibration is to use the calibrated model for predicting the real process. We can think of calibration as a preliminary to addressing the other statistical problems of interpolation, sensitivity analysis and uncertainty analysis, described in Section 2.3. Thus, the problem of predicting the true process $z(\boldsymbol{x})$ at some specified variable inputs $\boldsymbol{x}$ can be seen as interpolating the function $z(\cdot)$.

The posterior distribution of $z(\cdot)$ conditional on the estimated hyperparameters $\boldsymbol{\phi}$ and the calibration parameters $\boldsymbol{\theta}$ is a Gaussian process. Its mean function is given by

$$E(z(\boldsymbol{x}) \mid \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{d}) = \boldsymbol{h}(\boldsymbol{x},\boldsymbol{\theta})^T \hat{\boldsymbol{\beta}}(\boldsymbol{\theta}) + \boldsymbol{t}(\boldsymbol{x},\boldsymbol{\theta})^T \boldsymbol{V}_d(\boldsymbol{\theta})^{-1}(\boldsymbol{d} - \boldsymbol{H}(\boldsymbol{\theta})\hat{\boldsymbol{\beta}}(\boldsymbol{\theta})), \tag{10}$$

where

$$\boldsymbol{h}(\boldsymbol{x},\boldsymbol{\theta}) = \begin{pmatrix} \rho \boldsymbol{h}_1(\boldsymbol{x},\boldsymbol{\theta}) \\ \boldsymbol{h}_2(\boldsymbol{x}) \end{pmatrix}$$

and

$$\boldsymbol{t}(\boldsymbol{x},\boldsymbol{\theta}) = \begin{pmatrix} \rho \boldsymbol{V}_1((\boldsymbol{x},\boldsymbol{\theta}), D_1) \\ \rho^2 \boldsymbol{V}_1((\boldsymbol{x},\boldsymbol{\theta}), D_2(\boldsymbol{\theta})) + \boldsymbol{V}_2(\boldsymbol{x}, D_2) \end{pmatrix}.$$

Its covariance function is given by

$$\begin{aligned}\text{cov}(z(\boldsymbol{x}), z(\boldsymbol{x}') \mid \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{d}) &= \rho^2 c_1((\boldsymbol{x}, \boldsymbol{\theta}), (\boldsymbol{x}', \boldsymbol{\theta})) + c_2(\boldsymbol{x}, \boldsymbol{x}') \\ &-\boldsymbol{t}(\boldsymbol{x}, \boldsymbol{\theta})^T \boldsymbol{V}_d(\boldsymbol{\theta})^{-1} \boldsymbol{t}(\boldsymbol{x}', \boldsymbol{\theta}) \\ &+(\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{\theta}) - \boldsymbol{H}(\boldsymbol{\theta})^T \boldsymbol{V}_d(\boldsymbol{\theta})^{-1} \boldsymbol{t}(\boldsymbol{x}, \boldsymbol{\theta}))^T \boldsymbol{W}(\boldsymbol{\theta})(\boldsymbol{h}(\boldsymbol{x}', \boldsymbol{\theta}) - \boldsymbol{H}(\boldsymbol{\theta})^T \boldsymbol{V}_d(\boldsymbol{\theta})^{-1} \boldsymbol{t}(\boldsymbol{x}', \boldsymbol{\theta})),\end{aligned}$$

where $\boldsymbol{W}(\boldsymbol{\theta}) = (\boldsymbol{H}(\boldsymbol{\theta})^T \boldsymbol{V}_d(\boldsymbol{\theta})^{-1} \boldsymbol{H}(\boldsymbol{\theta}))^{-1}$. By combining this distribution with $p(\boldsymbol{\theta} \mid \boldsymbol{\phi}, \boldsymbol{d})$, we can make inferences about $z(\boldsymbol{x})$, again using numerical computation methods. For instance to estimate $z(\boldsymbol{x})$ we might use its posterior mean $E(z(\boldsymbol{x}) \mid \boldsymbol{\phi}, \boldsymbol{d})$ (for the estimated values of $\boldsymbol{\phi}$), obtained by integrating $E(z(\boldsymbol{x}) \mid \boldsymbol{\theta}, \hat{\boldsymbol{\phi}}, \boldsymbol{d}))$ with respect to $p(\boldsymbol{\theta} \mid \hat{\boldsymbol{\phi}}, \boldsymbol{d})$.

Now suppose that we wish to predict the real process in the context where one or more of the variable inputs is subject to parametric variability, as discussed in Section 2.1. Section 2.2 gives examples of computer codes for which inputs may be unspecified in this way. The problem of uncertainty analysis is to study the (extra) uncertainty in model outputs induced by this parametric variability. Although uncertainty analysis for computer codes is typically formulated in this way, i.e. with concern for uncertainty in the code outputs, in the present context the larger challenge is to study uncertainty in the real process $z(\cdot)$.

We therefore consider the random variable $z(\boldsymbol{X})$, where the variable inputs $\boldsymbol{X}$ are now random, having a distribution $G_{\boldsymbol{X}}(\boldsymbol{x})$. (In practice, only a subset of the variable inputs will be subject to parametric variability, so $G_{\boldsymbol{X}}(\cdot)$ will be degenerate in the other dimensions.) The task of uncertainty analysis is now to make inference about the *distribution* of $z(\boldsymbol{X})$. In particular, we wish to make inference about properties of this distribution such as the mean $K = E_{\boldsymbol{X}}\{z(\boldsymbol{X})\} = \int_{\mathcal{X}} z(\boldsymbol{x})\, dG_{\boldsymbol{X}}(\boldsymbol{x})$, the variance $L = \text{var}_{\boldsymbol{X}}\{z(\boldsymbol{X})\} = K_2 - K^2$, where $K_2 = \int_{\mathcal{X}} z(\boldsymbol{x})^2\, dG_{\boldsymbol{X}}(\boldsymbol{x})$, or the value at some point g of the distribution function $F(g) = P_X\{z(\boldsymbol{X}) \le g\} = \int_{z(\boldsymbol{x})\le g} dG_{\boldsymbol{X}}(\boldsymbol{x})$. Inference about these or other summaries of the distribution of $z(\boldsymbol{X})$ may be derived from the posterior distribution of $z(\cdot)$. Details are given in Kennedy and O'Hagan (2000b).

It is equally straightforward to work in terms of an uncertainty analysis of the code output $\eta(\boldsymbol{x}, \boldsymbol{\theta})$, with respect to either or both of parametric variability in $\boldsymbol{x}$ and parametric uncertainty (after calibration) in $\boldsymbol{\theta}$.

We do not explicitly deal with sensitivity analysis in this paper: appropriate techniques are outlined in O'Hagan *et al.* (1999).

5. IMPLEMENTATION DETAILS

5.1. Design issues

We now consider a number of practical issues arising in the implementation of the theory in Section 4, beginning with the question of the choice of the sets of points at which the code is run and at which observational data are observed. The set D_2 of values $\boldsymbol{x}_i$ of the variable inputs for the calibration data will often not be a matter of choice. In our example in Section 6, for example, the available data are given to us. On the other hand, we will generally be able to choose the code design D_1 of points $(\boldsymbol{x}_j^*, \boldsymbol{t}_j)$. There is a considerable literature on the design of computer experiments— see for example Sacks, Schiller and Welch (1989), Morris *et al.* (1993), Morris and Mitchell (1995), Bates *et al.* (1996). All of this relates to the simpler problem of designing sets of code input values for the purpose of interpolating the code itself, or for uncertainty analysis of the code (Haylock, 1997). The problem of design for calibration is more complex and a topic for future research. We have so far chosen designs on a more heuristic basis.

First, existing work suggests that it is important for the code design to give good coverage of the region of $(\boldsymbol{x}, \boldsymbol{t})$ space of greatest interest. The variable input coordinates $\boldsymbol{x}_j^*$ should cover both the range of points $\boldsymbol{x}_i$ in the calibration data and the range of values over which we may wish to predict the process in future. The calibration input coordinates $\boldsymbol{t}_j$ should cover the range that is plausible for the true value $\boldsymbol{\theta}$ of the calibration parameters. The latter suggests a sequential design approach, beginning with values spanning the prior distribution of $\boldsymbol{\theta}$ then adding more points over the range covered by its posterior distribution. For examples of this kind of sequential design approach see Bernardo *et al.* (1992), Craig *et al.* (1996) and Aslett *et al.* (1998).

A second intuitive consideration is that there should be control values $\boldsymbol{x}_j^*$ in D_1 close to the values $\boldsymbol{x}_i$ in D_2 in order to learn about the relationship between the code and reality.

For our example in Section 6, we have no choice over the calibration design D_2. We set the code design D_1 to be the cartesian product of D_1 with a latin hypercube design for the calibration inputs. For the latter, we have used a maximin latin hypercube as described in Morris and Mitchell (1995). These designs give good coverage of the space and are evenly distributed in each one-dimensional projection. The use of a Cartesian product has some computational advantages which are briefly described in Kennedy and O'Hagan (2000b).

5.2. Modelling choices

In application of the theory, we need to specify $\boldsymbol{h}_1(\boldsymbol{x}, \boldsymbol{t})$, $\boldsymbol{h}_2(\boldsymbol{x})$, $c_1((\boldsymbol{x}, \boldsymbol{t}), (\boldsymbol{x}', \boldsymbol{t}'))$ and $c_2(\boldsymbol{x}, \boldsymbol{x}')$. $\boldsymbol{h}_1(\boldsymbol{x}, \boldsymbol{t})$ should be chosen to reflect beliefs about the general shape of the function $\eta(\boldsymbol{x}, \boldsymbol{t})$, and $\boldsymbol{h}_2(\boldsymbol{x})$ should be chosen to reflect beliefs about the shape of $\delta(\boldsymbol{x})$. In the latter case, particularly, we may not have any specific expectations to model through $\boldsymbol{h}_2(\cdot)$. Generally, it does not help to put components into these functions that are not motivated by actual prior knowledge. This is in contrast to parametric regression modelling, where adding extra regressor variables will in general produce improved fit. The Gaussian process is nonparametric and will adapt to whatever shape of function is suggested by the data, and will often do so better if spurious regressors are not included. In applications, therefore, unless there is prior information to suggest more complex modelling, we take $\boldsymbol{h}_1(\boldsymbol{x}, \boldsymbol{t}) = (1)$ and $\boldsymbol{h}_2(\boldsymbol{x}) = (1)$ as defaults. This means that $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ are scalars and represent unknown constant means for $\eta(\cdot, \cdot)$ and $\delta(\cdot)$.

For the covariance functions we again model parsimoniously. We generally adopt the form (3), (4), so that

$$\begin{aligned} c_1((\boldsymbol{x}, \boldsymbol{t}), (\boldsymbol{x}', \boldsymbol{t}')) &= \sigma_1^2 \exp\{-(\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{\Omega}_x (\boldsymbol{x} - \boldsymbol{x}')\} \exp\{-(\boldsymbol{t} - \boldsymbol{t}')^T \boldsymbol{\Omega}_t (\boldsymbol{t} - \boldsymbol{t}')\}, \quad (11) \\ c_2(\boldsymbol{x}, \boldsymbol{x}') &= \sigma_2^2 \exp\{-(\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{\Omega}_x^* (\boldsymbol{x} - \boldsymbol{x}')\}, \quad (12) \end{aligned}$$

with diagonal forms for $\boldsymbol{\Omega}_t, \boldsymbol{\Omega}_x$ and $\boldsymbol{\Omega}_x^*$.

It is important to recognise that these are not trivial modelling choices. The Gaussian forms for the covariance function imply a belief in differentiability of both $\eta(\cdot, \cdot)$ and $\delta(\cdot)$, and indeed imply a belief that these functions are analytic. This may be appropriate for the computer code $\eta(\cdot, \cdot)$ but these assumptions for both $\eta(\cdot, \cdot)$ and $\delta(\cdot)$ imply the same beliefs about the real world process $\zeta(\cdot)$, which will often be inappropriate. We return to this matter in Section 6.3.

Assuming diagonal forms for the roughness matrices implies that any elliptical anisotropy in the covariances is oriented along the individual parameter axes. Transformation may be relevant to make this assumption more realistic, and is implemented in our examples as described in Section 6. Another assumption in (11) is that there is separability between the calibration and variable inputs in the covariance structure of the code $\eta(\cdot,\cdot)$. Separability is frequently assumed in various statistical applications, particularly for modelling space-time processes, for instance by Haslett and Raftery (1989) and Oehlert (1993). A result in O'Hagan (1998) provides a characterisation of separability that may give a justification in some contexts, but it is primarily assumed in practice for reasons of convenience and parsimony.

Finally, we may remark that even the underlying assumption of stationarity may be questioned, particularly in respect of the real process. A reasonably tractable nonstationary alternative might be the localised regression model of O'Hagan (1978). Sampson and Guttorp (1992) give a very general nonparametric technique, but it would be much more difficult to fit this into our framework.

Our reasons for using these assumptions in our examples are briefly as follows. First, (11) and (12) facilitate computation by allowing analytical results that would otherwise need to be evaluated numerically (see Kennedy and O'Hagan, 2000b), greatly increasing computation times. We employ (4) rather than the more general $r(\boldsymbol{d}) = \exp(-\boldsymbol{d}^T\boldsymbol{\Omega}\boldsymbol{d})$ for reasons of parsimony. In general, hyperparameters are not well identified in these models. Quite different values for $\boldsymbol{\phi}$ may fit the data equally well, and produce comparable predictions. The form (4) is sufficiently flexible to allow for some anisotropy in the correlation structure, and indeed we adopt the isotropic form $r(\boldsymbol{d}) = \exp(-\omega\boldsymbol{d}^T\boldsymbol{d})$ wherever it seems acceptable.

We freely admit, therefore, that we employ these assumptions essentially for convenience and simplicity. However, we believe that to a large extent other assumptions could give very similar results. Some tentative support for this view is given by some findings reported in Section 6.3.

We also need to specify prior distributions $p(\boldsymbol{\theta})$ and $p(\boldsymbol{\phi})$ for the calibration parameters and hyperparameters respectively. We adopt a normal prior distribution for $\boldsymbol{\theta}$, after transformation where appropriate, again for computational convenience. Prior information about hyperparameters will often be weak, with the possible exception of λ. It is useful, however, to try to formulate some prior knowledge about roughness parameters like ω. A uniform prior distribution should generally not be used for such parameters because our modal estimation method will often give unrealistically large estimates. Where prior information is weak, the form $p(\omega) \propto \omega^{-1}$ is preferable.

5.3. *Computation*

The main computational issues concern the need for numerical integration with respect to the posterior distribution of $\boldsymbol{\theta}$, and the fact that we need to invert the matrix $\boldsymbol{V}_d(\boldsymbol{\theta})$ for each $\boldsymbol{\theta}$ value in that numerical integration.

If the code $\eta(\cdot,\cdot)$ is complex and computer-intensive, we will expect the number N of code evaluations available to be relatively small (and we expect n to be smaller still). Then the inversion of the $(N+n)\times(N+n)$ matrix $\boldsymbol{V}_d(\boldsymbol{\theta})$ may not be a serious problem. However, for a simpler code we may expect to be able to make larger numbers of runs to obtain more information about $\eta(\cdot,\cdot)$. Then N is potentially very large. In this case considerable computational savings are achieved by the code design D_1 having a Cartesian product form. See Kennedy and O'Hagan (2000b) for details.

Another device that might be considered for computation with large correlation matrices

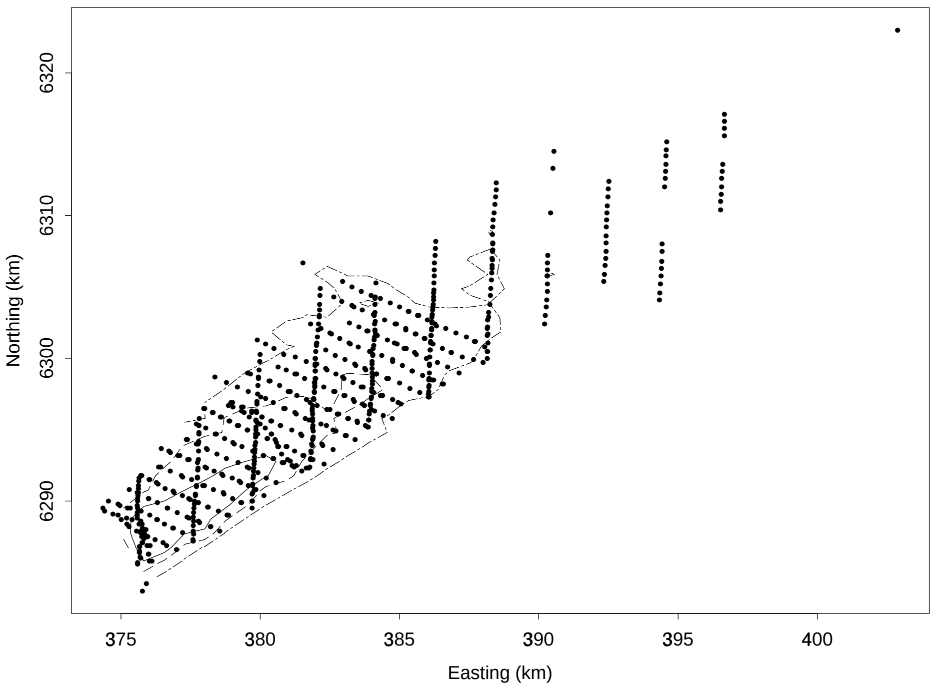

Fig. 1. Tomsk aerial survey of 695 Ru106 deposition measurements, with contours at heights of 11 (solid line), 10 (— — —) and 9 (— - —).

is the local computation approach; Vecchia (1988). However, it is not clear how that idea could be usefully applied in the more complex framework of calibration.

Turning now to the question of numerical integration with respect to $\boldsymbol{\theta}$, in our examples we use the iterative Gauss-Hermite quadrature method of Naylor and Smith (1982). This approach is realistic because the dimensionality of $\boldsymbol{\theta}$ is relatively low, so that quadrature is feasible, and because the code is relatively simple, so that we can afford to use Cartesian product rules and iteration. With more expensive codes or in somewhat higher dimensional $\boldsymbol{\theta}$ space it becomes important to use more efficient quadrature designs (for references see Evans and Schwartz, 1995). For high-dimensional $\boldsymbol{\theta}$, it may become necessary to use simulation methods of integration: we have not explored this yet.

6. EXAMPLE: TOMSK DATA

6.1. *Data and model*

We now present an analysis of data from an accident at the Tomsk-7 chemical plant in 1993. A detailed account of the accident is given in Shershakov *et al.* (1995). Measurements were made for three radionuclides. However, for this example we only consider the deposition of ruthenium 106 (Ru106). A total of 695 measurements of Ru106 deposition were made, at locations shown in Figure 1. The contour lines represent an interpolation of the log-deposition at these points. These data were obtained from an aerial survey which started close to the source and continued to approximately 40km downwind, in such a way that consecutive measurements are very close.

For the prior approximation $\eta(\cdot,\cdot)$ we use the logarithm of the Gaussian plume model described in Section 2.2, with a fixed level of background radiation added. The log transformation is used to better approximate the assumptions of normality made in our model. The critical unknown inputs for this model are the source term and deposition velocity. We therefore treat the logarithms of these as the calibration parameters $\boldsymbol{\theta}$. $z(\boldsymbol{x})$ represents the true log-deposition for variable inputs $\boldsymbol{x}$. The variable inputs in this case comprise two orthogonal linear functions of the northing and easting coordinates such that $\boldsymbol{x} = (0, 0)$ represents the source point and a point $\boldsymbol{x} = (x_1, x_2)$ is distance x_1 downwind from the source and distance x_2 from the plume centre line. The z_is are logarithms of observed depositions.

The pattern of deposition seen in Figure 1 has a well defined plume shape, and we would expect the Gaussian plume code to provide a reasonably good approximation.

A normal distribution is used to approximate prior beliefs about $\boldsymbol{\theta}$. The prior means were obtained from the National Radiological Protection Board (NRPB). The variances were set to 5 to represent vague prior knowledge, and prior covariances were assumed to be 0. These are realistic values for the variances, since the NRPB think the values could be a couple of orders of magnitude from the prior estimates.

In this example we have treated the plume code as a known function, for given values of the parameters, since it is practical to run the code for any input configuration of interest. Although it is perfectly possible to follow through the more complex analysis with code uncertainty, this simplification allowed us to perform some analyses to examine sensitivity to model assumptions (See Section 6.3). Much of the theory of Section 4 simplifies as a result. For example, the only correlation function to specify is $c_x(\cdot,\cdot)$. We first assume the simple product form (4) with two roughness parameters (ω_1, ω_2), corresponding to the directions parallel and perpendicular to the the plume axis, as the prevailing wind and the crosswind are believed to affect the deposition differently.

From the original 695 measurements, a subset of size 10 was chosen to represent a small sample of observed data similar to that which might be collected from ground measurements shortly after an accident. The points were chosen reasonably close to the source, but to avoid clustering were constrained so that each point is at least 5 measurement points from every other selected point. Clustered points lead to redundant information, which would make estimation of the model hyperparameters very difficult for such small datasets, and the constraint ensures the data are relatively dispersed. Additional points were chosen similarly and added to this subset, giving subsets of size 10, 15, 20 and 25 points. The point furthest from the source was deliberately included in the 25 point dataset.

6.2. Results

Conditional on each of the datasets, posterior means and variances of $z(\boldsymbol{x})$ were calculated for all the 670 'unobserved' points, and accuracy was assessed on the basis of the true values at these points. The following strategies were compared:

- Strategy 1. Using a Gaussian process interpolation of the physical observations alone, taking into account measurement error, but making no use of the Gaussian plume model.

- Strategy 2. Using Bayesian calibration and model inadequacy correction, as described in Section 4.6.

- Strategy 3. Using the Gaussian plume model with 'plug-in' input parameters. The physical data is not interpolated in any way. Instead, we select the input parameters by minimising the sum of squared differences between the model and the data.

The data used for each strategy were the same, comprising from 10 to 25 of the original data points. Table 1 gives the Root Mean Squared Errors (RMSE) of prediction for each of these strategies. For comparison, RMSE=0.84 is obtained using the code with input parameters fixed at their prior mean. Strategy 1 achieves little improvement over this value for the data samples used. Strategy 3 is the kind of 'best-fitting' calibration technique often used in practice. Given enough data points with which to calibrate the model, this method improves on the use of the code with prior mean for $\boldsymbol{\theta}$, but Strategy 2 is even better, because it also takes account of model inadequacy. Comparison of Strategies 1 and 3 shows that when the number of observations is small, the use of the code (suitably calibrated) is more accurate than simply interpolating the data.

$RMSE$	n=10	n=15	n=20	n=25
Strategy 1	0.75	0.76	0.86	0.79
Strategy 2	0.42	0.41	0.37	0.36
Strategy 3	0.82	0.79	0.76	0.66

Table 1. Root Mean Squared Errors based on n observations

To assess the significance of the improvement here, we note that the observations are log-depositions. So an error of 0.82 (Strategy 3, $n = 10$) corresponds to errors in predicting deposition by a factor of $\exp(0.82) = 2.3$, and reducing this to 0.42 (Strategy 2, $n = 10$) cuts the error in predicting deposition to a factor of 1.5. This is a genuinely useful improvement in the context of radiological protection.

The Quantile-Quantile plots in Figure 2 correspond to the standardised residuals based on Strategies 1 and 2 in the case $n = 25$. The plot for Strategy 2 clearly shows a better fit of the data to the predictive distribution than is achieved by Strategy 1. However, both plots show heavy-tailed characteristics, and this feature is explored further in the following section.

For larger datasets we would expect Strategies 1 and 2 to produce similar results. However, good prediction from interpolation of the data alone relies much more on an even distribution of design points over the prediction region. In most applications, optimal designs for physical observations are not practical. Our method makes greater use of the code in regions where there are few physical data points and the uncertainty about model inadequacy in these regions is reflected in the posterior variance. Further evidence of predictive improvements using Bayesian calibration and model inadequacy correction may be found in a second example described in Kennedy and O'Hagan (2000b).

6.3. Sensitivity to modelling assumptions

So far, we have made various modelling choices, particularly in relation to the correlation function, which will not be appropriate for all applications. We now briefly examine how

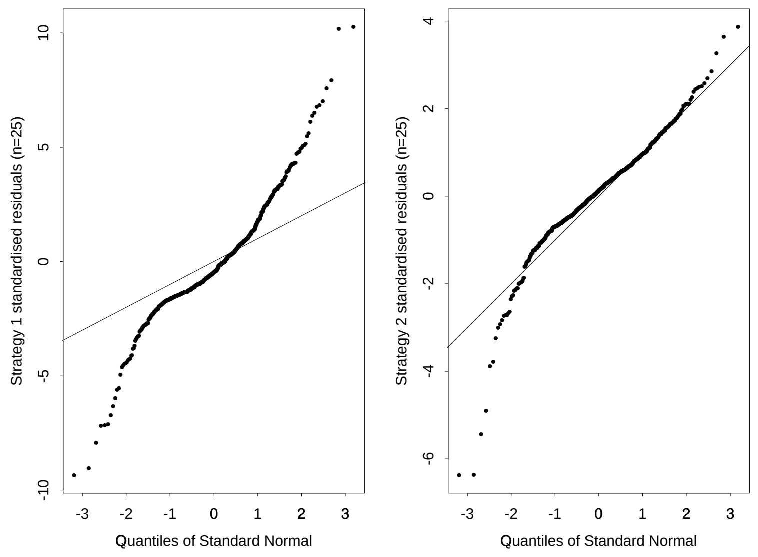

Fig. 2. Quantile-Quantile plots for Strategies 1 and 2 with $n = 25$

some alternative plausible modelling assumptions affect inferences in the case of the 25-point Tomsk data. Further details are given in Kennedy and O'Hagan (2000b).

The model described above (Strategy 2) will be referred to as M1. Three alternative models are outlined below. In M2 we relax the assumption that the hyperparameters are fixed, in M3 we use an alternative functional form for the correlation function, and in M4 we consider the isotropic form of the gaussian correlation function.

M2: Integration with respect to the roughness parameters. It was suggested in Section 4.5 that fixing hyperparameters at the posterior modal values, rather than treating them as uncertain, is an acceptable simplification of the model. It is often the case in models of this kind that inferences (especially posterior variances) are sensitive to the choice of the roughness parameters in the correlation function. In an attempt to take more account of the uncertainty about these parameters, we used a crude numerical method to integrate over ω_1, ω_2 in calculating the posterior predictive means and variances. These represent the roughness parameters in our non-isotropic product of 1-dimensional correlation functions.

M3: Isotropic Matérn correlation. The exponential form of the correlation function is appropriate if the inadequacy function is analytic, and therefore may not be the best for modelling physical systems. We carried out an analysis identical to the one described above but using the isotropic Matérn correlation function suggested by Handcock and Wallis (1994).

M4: Isotropic gaussian correlation function. The final model variation considered uses the isotropic gaussian correlation function $c(\boldsymbol{x}, \boldsymbol{x}') = \exp(-b|\boldsymbol{x} - \boldsymbol{x}'|^2)$. This is equivalent to assuming that $\omega_1 = \omega_2$ in M1. The estimated values of these parameters under M1 differ by a factor of 20. Under M4 we might therefore expect to see quite different inferences than we obtained with M1 if there is sensitivity to these roughness parameters.

These analyses suggest that (a) any improvement due to integrating with respect to the hyperparameters, as opposed to maximising, is likely to be small, and (b) the effect of using alternative covariance structures is also small. However, we do not propose that any of these models fits perfectly. Plots of prediction errors suggest that the true deposition surface exhibits local features that we are failing to predict. In some sense, all of the models have predictive distributions with tails that are too thin. This is an area for ongoing investigation, but is likely to be application-specific.

7. CONCLUSIONS AND FURTHER WORK

We have presented a Bayesian approach to calibrating a computer code using observations from the real process, and subsequent prediction and uncertainty analysis of the process which corrects for model inadequacy. The posterior summaries take account of all remaining sources of uncertainty.

We have treated the code as a 'black box', and the methods described in this paper are applicable to computer codes of arbitrary complexity.

As we have already mentioned in Section 5.1, important questions remain about the choice of design points. The physical observation sites will often be limited, as in the example presented here, but there will be situations in which these may be controlled. The designs already used to learn about code uncertainty might be used for D_2 in order to learn about model inadequacy. The problem of choosing designs D_1 and D_2 to give good calibration of the model is more difficult. The Cartesian product designs for D_1 described in Section 5 work well in the example of Section 6, in which the design D_2 is assumed to be fixed. These designs also facilitate the computations as outlined in Section 5.3.

We have carried out integration with respect to $\boldsymbol{\theta}$ by simple quadrature, which is feasible for low-dimensional $\boldsymbol{\theta}$ but would become impractical with larger numbers of calibration parameters. The obvious approach then is MCMC. However, the calibration distribution $p(\boldsymbol{\theta} \mid \hat{\boldsymbol{\phi}}, \boldsymbol{d})$ is a complicated function of $\boldsymbol{\theta}$, and it would appear to be difficult to simulate from this distribution using MCMC methods. Nevertheless, in our examples we have found it to be reasonably well approximated by a normal distribution (convergence of our iterative quadrature relies on a normal approximation). If this is true for high dimensional $\boldsymbol{\theta}$ it should be possible to use an approximate MCMC integration method.

In our examples, we have not considered the more important case in which the code output is multivariate. Effective calibration should ideally use all available code outputs and corresponding physical measurements. For example, in the nuclear accident application the NRPB would make a large number of air concentration measurements some time before the first ground deposition measurements are available. The air measurements can provide information about the unknown source terms, and therefore should be used in the analysis. The use of multiple code outputs and measurements is a topic for future research. It was suggested in Section 4.1 that multivariate outputs might be handled simply by creating additional input parameters. In the nuclear application we could in principle treat "measurement type" and "radionuclide" as inputs. However, for these types of input it would not be reasonable to make the kind of assumptions we make about the correlation structure.

We have discussed quite extensively the question of alternative covariance structures. Experiments reported in Section 6.3 indicated some degree of robustness, but also a need to consider models that allow for more localised structure. We need to explore our methods with more and varied applications.

Within our Gaussian process framework, it would be relatively straightforward to accommodate observations of derivatives of the code, as in O'Hagan (1992). Derivatives are sometimes available, but we have not examined this so far. Other more speculative topics for future research include opening up the 'black box', discussed in Section 1.1, and applications with high dimensional $\boldsymbol{\theta}$. Our examples hitherto have not gone beyond a few dimensions, yet calibration problems may have many calibration parameters to 'fit'. The simulation-based approach of Romanowicz *et al.* (1994) can tackle high dimensional $\boldsymbol{\theta}$ but does not allow for model inadequacy. We suspect that a preliminary dimension reduction exercise, as in Craig *et al.* (1999), offers the most promising approach in such cases.

ACKNOWLEDGEMENTS

This research was supported by research grant GR/K54557 from the Engineering and Physical Sciences Research Council, UK, with additional financial contributions from the National Radiological Protection Board and the Environment Agency. We thank Neil Higgins, Tom Charnock and their colleagues at the National Radiological Protection Board for providing us with the data, and for advice on the use of the Gaussian plume model. We are also genuinely grateful for many very helpful and pertinent comments from referees, that have led us to improve this paper in numerous ways.

REFERENCES

Aitchison, J. and Dunsmore, I. R. (1975) *Statistical Prediction Analysis.* Cambridge: University Press.

Aslett, R., Buck, R. J., Duvall, S. G., Sacks, J. & Welch, W. J. (1998) Circuit optimization via sequential computer experiments: Design of an output buffer. *Appl. Statist.*, **47**, 31–48.

Bates, R. A., Buck, R. J., Riccomagno, E. and Wynn, H. P. (1995) Experimental design and observation for large systems. *J. R. Statist. Soc.* B, **58**, 77–94.

Chiles, J. and Delfiner, P. (1999), *Geostatistics: Modeling Spatial Uncertainty*, New York: Wiley.

Clarke, R. H. (1979) The first report of a Working Group on Atmospheric Dispersion: A model for short and medium range dispersion of radionuclides released to the atmosphere, Harwell, NRPB-R91. London: HMSO.

Cox, D. D., Park, J. S., Sacks, J. and Singer, C. E. (1992) Tuning complex computer codes to data. *Proceedings of the 23rd Symposium of the Interface of Computing Science and Statistics, April 21–24, 1991, Seattle, WA*, Interface Foundation, Fairfax Station, VA, 266–271.

Cox, D. D., Park, J. S., Singer, C. E. (1996) A statistical method for tuning a computer code to a data base. *Tech. Rep 96-3*, Dept. of Statistics, Rice University.

Craig, P. S., Goldstein, M., Rougier, J. C. and Seheult, A. H. (1999) Bayesian forecasting using large computer models. *Unpublished manuscript.*
Available at http://www.maths.dur.ac.uk/stats/physpred/prediction.ps

Craig, P. S., Goldstein, M., Seheult, A. H. and Smith, J. A. (1996) Bayes linear strategies for matching hydrocarbon reservoir history. In *Bayesian Statistics 5*, J. M. Bernardo, J. O. Berger, A. P. David and A. F. M. Smith, (eds.). Oxford: University Press. 69–95.

Crick, M. J., Hofer, E., Jones, J. A. and Haywood, S. M. (1988) Uncertainty analysis of the foodchain and atmospheric dispersion modules of MARC. Technical Report NRPB-R184, National Radiological Protection Board.

Cressie, N. A. C. (1991) *Statistics for Spatial Data.* New York: Wiley.

Currin, C., Mitchell, T., Morris, M. and Ylvisaker, D. (1991) Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments, *J. Amer. Statist. Assoc.*, **86**, 953–963.

Dey, D., Müller, P. and Sinha, D. (editors) (1998) *Practical Nonparametric and Semiparametric Bayesian Statistics.* New York: Springer-Verlag.

Diaconis, P. (1988) Bayesian numerical analysis. *Statistical Decision Theory and Related Topics IV*, **1**. (S. S. Gupta and J. Berger, eds.), New York: Springer, 163–175.

Draper, D. (1995) Assessment and propagation of model uncertainty (with discussion). *J. Roy. Statist. Soc.* B, **57**, 45–97.

Draper, D., Pereira, A., Prado, P., Saltelli, A., Cheal, R., Eguilior, S., Mendes, B. and Tarantola, S. (1999) Scenario and parametric uncertainty in GESAMAC: A methodological study in nuclear waste displosal risk assessment. *Comp. Phys. Comm.*, **117**, 142–155.

Evans, M. and Schwartz, T. (1995) Methods for approximating integrals in statistics with special emphasis on Bayesian integration problems (with discussion). *Statist. Sci.*, **10**, 254–272.

Goldstein, M. (1986) Separating beliefs. In *Bayesian Inference and Decision Techniques, Essays in Honour of Bruno de Finetti* (eds P. K. Goel and A. Zellner). Amsterdam: North-Holland.
—(1998) Adjusting belief structures. *J. R. Statist. Soc.* B, **50**, 133–154.

Handcock, M. S. and Stein, M. L. (1993) A Bayesian analysis of kriging. *Technometrics*, **35**, 403–410.

Handcock, M. S. and Wallis, J. R. (1994) An approach to statistical spatial-temporal modelling of meteorological fields (with discussion) *J. Amer. Statist. Assoc.*, **89**, 368–390.

Haslett, J. and Raftery, A. E. (1989) Space-time modelling with long-memory dependence: assessing Ireland's wind power resource (with discussion). *Applied Statistics*, 38, 1–50.

Haylock, R. (1997) Bayesian inference about outputs of computationally expensive algorithms with uncertainty on the inputs. Unpublished PhD Thesis, University of Nottingham.

Haylock, R. and O'Hagan, A. (1996) On inference for outputs of computationally expensive algorithms with uncertainty on the inputs. In *Bayesian Statistics 5*, J. M. Bernardo, J. O. Berger, A. P. David and A. F. M. Smith, (eds.). Oxford: University Press. 629–637.

Helton, J. C. (1993) Uncertainty and sensitivity analysis techniques for use in performance assessment for radioactive waste disposal, *Reliability Engineering and System Safety*, **42**, 327–367.

Helton, J. C., Garner, J. W., McCurley, R. D. and Rudeen, D. K. (1991) Sensitivity analysis techniques and results for performance assessment at the waste isolation pilot plant.

Technical Report SAND90-7103, Sandia National Laboratories, Albuquerque New Mexico.

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

Iman, R. L. and Conover, W. J. (1980) Small-sample sensitivity analysis techniques for computer models, with an application to risk assessment (with discussion). *Comm. Statist.–Theory and methods*, **A9**, 1749–1874.

Jones, J. A. (1981) The second report of a working group on atmospheric dispersion: A procedure to include deposition in the model for short and medium range atmospheric dispersion of radionuclides. Harwell, NRPB-R122. London: HMSO.

Kimeldorf, G. S. & Wahba, G. (1970) A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *Ann. Math. Statist.*, **41**, 495–502.

Kennedy, M. C. and O'Hagan, A. (2000a) Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, **87**, 1–13.

Kennedy, M. C. and O'Hagan, A. (2000b) Supplementary details on Bayesian calibration of computer codes. *Unpublished manuscript.*
Available at http://www.shef.ac.uk/~st1ao/ps/calsup.ps

Liu, L. and Arjas, E. (1998). A Bayesian model for fatigue crack growth. In Dey, D., Müller, P. and Sinha, D. (editors), *Practical Nonparametric and Semiparametric Bayesian Statistics*, 339–353. New York: Springer-Verlag.

Matheron, G. (1963) Principles of geostatistics. *Economic Geol.*, **58**, 1246–1266.

McKay, M. D., Conover, W. J. and Beckman, R. 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.

Morris, M. D. (1991) Factorial sampling plans for preliminary computational experiments. *Technometrics*, **33**, 161–174.

Morris, M. D. and Mitchell, T. J. (1995) Exploratory Designs for Computational Experiments, *J. Statist. Planng Inf.*, **43**, 381–402.

Morris, M. D., Mitchell, T. J. and Ylvisaker, D. (1993) Bayesian design and analysis of computer experiments: use of derivatives in surface prediction, *Technometrics*, **35**, 243–255.

Naylor, J.C. and Smith, A.F.M. (1982) Applications of a method for the efficient computation of posterior distributions, *Appl. Statist.*, **31**, 214–225.

Neal (1996) *Bayesian Learning for Neural Networks.* New York: Springer-Verlag.
—(1999) Regression and classification using Gaussian process priors. (1999) In: *Bayesian Statistics 6*, (J. M. Bernardo, J. O. Berger, A. P. Dawid and A.F.M. Smith, eds.), 475–501 (with Discussion). Oxford: University Press.

Novak, E. (1988) *Deterministic and Stochastic Error Bounds in Numerical Analysis,* Lecture Notes in Mathematics No. **1349**. Berlin: Springer-Verlag.

Oakley, J. E. and O'Hagan, A. (1998) Bayesian inference for the uncertainty distribution. Tech. Rep. University of Nottingham, Statistics Section.

Oehlert, G. W. (1993) Regional trends in sulfate wet deposition. *J. Amer. Statist. Soc.*, **88**, 390–399.

O'Hagan, A. (1978) Curve fitting and optimal design for prediction. *J. R. Statist. Soc.* B **40**, 1–42 (with discussion).

—(1991) Bayes-Hermite quadrature. *J. Statist. Planng Inf.*, **29**, 245-260.

—(1992) Some Bayesian numerical analysis. In: *Bayesian Statistics 4*, J. M. Bernardo, J. O. Berger, A. P. Dawid and A. F. M. Smith, (eds.). Oxford: University Press, 345–363, (with discussion).

—(1998) A Markov property for covariance structures. Tech. Rep. 98-13, University of Nottingham, Statistics Section. Also available at http://www.shef.ac.uk/ st1ao/ps/kron.ps

O'Hagan, A., Kennedy, M. C. and Oakley, J. E. (1999) Uncertainty analysis and other inference tools for complex computer codes. In: *Bayesian Statistics 6*, (J. M. Bernardo, J. O. Berger, A. P. Dawid and A.F.M. Smith, eds.), 503–524 (with Discussion). Oxford: University Press.

Omre, H. (1987) Bayesian kriging—merging observations and qualified guesses in kriging. *Math. Geol.*, **19**, 25–39.

Owen, A. B. (1992) A central limit for Latin hypercube sampling. J. R. Statist. Soc. B, **54**, 541–551.

Poole, D. and Raftery, A. E. (1998) Inference for deterministic simulation models: the Bayesian melding approach. Technical Report 346, Department of Statistics, University of Washington.

Raftery, A. E., Givens, G. H. and Zeh, J. E. (1995) Inference from a deterministic population dynamics model for bowhead whales (with discussion). *J. Amer. Statist. Assoc.*, **90**, 402–430.

Rios Insua, D. and Muller, P. (1998) Feedforward neural networks for nonparametric regression. In Dey, D., Müller, P. and Sinha, D. (editors), *Practical Nonparametric and Semiparametric Bayesian Statistics*, 181–193. New York: Springer-Verlag.

Romanowicz, R., Beven, K. and Tawn, J. A. (1994) Evaluation of Predictive uncertainty in nonlinear hydrological models using a Bayesian approach. In: *Statistics for the Environment 2: Water Related Issues* (V. Barnett and K. F. Turkman, eds.), 297–319. New York: Wiley.

Sacks, J. Schiller, S. B. and Welch, W. J. (1989) Designs for computer experiments. *Technometrics*, **31**, 41–47.

Sacks, J., Welch, W.J., Mitchell, T.J. and Wynn, H.P. (1989) Design and Analysis of Computer Experiments, *Statist. Sci.*, **4**, 409–435.

Saltelli, A., Chan, K. and Scott, M. (2000) (eds.) *Mathematical and Statistical Methods for Sensitivity Analysis.* New York: Wiley. In press.

Saltelli, A. and Sobol', I. M. (1995) About the use of rank transformation in sensitivity analysis of model output. *Reliability Engineering and System Safety*, **50**, 225–239.

Sampson, P. D. and Guttorp, P. (1992) Nonparametric estimation of nonstationary covariance structure. *J. Amer. Statist. Assoc.*, **87**, 108–119.

Schweder, T. and Hjort, N. L. (1996) Bayesian synthesis or likelihood synthesis — what does Borel's paradox say? *Reports of the International Whaling Commision*, **46**, 475–480.

Shershakov, V. M., Vakulovski, S. M., Borodin, R. V., Vozzhennikov, O. I., Gaziev, Y. L., Kosykh, V. S., Makhonto, V. S., Chumiciev, V. B., Korsakov, A. T., Martynenko, V. P. and Godko, A. (1995) Analysis and prognosis of radiation exposure following the accident at the Siberian chemical combine Tomsk-7, *Radiation Protection Dosimetry*, **59**, 93–126.

Smith, M. and Kohn, R. (1998) Nonparametric estimation of irregular functions with independent or autocorrelated errors. In Dey, D., Müller, P. and Sinha, D. (editors), *Practical Nonparametric and Semiparametric Bayesian Statistics*, 157–179. New York: Springer-Verlag.

Stein, M. L. (1987) Large sample properties of simulation using latin hypercube sampling. *Technometrics*, **29**, 143–151.
—(1999) *Interpolation of Spatial Data: Some Theory for Kriging.* New York: Springer-Verlag.

Vecchia, A. V. (1988). Estimation and identification for continuous spatial processes. *J. Roy. Statist. Soc. B*, **50**, 297–312.

Vidakovic, B. (1998) Wavelet-based nonparametric Bayes methods. In Dey, D., Müller, P. and Sinha, D. (editors), *Practical Nonparametric and Semiparametric Bayesian Statistics*, 133–155. New York: Springer-Verlag.

Walker, S. G., Damien, P., Laud, P. W. and Smith, A. F. M. (1999) Bayesian nonparametric inference for random distributions and related functions (with discussion). *J. Roy. Statist. Soc.* **B**, **61**, 485–527.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J. and Morris, M. D. (1992) Screening, predicting, and computer experiments, *Technometrics*, **34**, 15–25.

Wolpert, R. L. (1995) Comment on the paper by Raftery, Givens and Zeh. *J. Amer. Statist. Assoc.*, **90**, 426–427.

Wooff, D. A. (1992) [B/D] works. In *Bayesian statistics 4* (eds J. M. Bernardo, J.O.Berger, A. P. Dawid and A. F. M. Smith), 851–859. Oxford: University Press.