Diagnostics for Gaussian Process Emulators

Leonardo S. Bastos and Anthony O'Hagan
Department of Probability and Statistics
University of Sheffield
Sheffield, S3 7RH, UK
{l.bastos,a.ohagan}@shef.ac.uk

April 10, 2009

Abstract

Mathematical models, usually implemented in computer programs known as simulators, are widely used in all areas of science and technology to represent complex real-world phenomena. Simulators are often sufficiently complex that they take appreciable amounts of computer time or other resources to run. In this context, a methodology has been developed based on building a statistical representation of the simulator, known as an emulator. The principal approach to building emulators uses Gaussian processes. This work presents some diagnostics to validate and assess the adequacy of a Gaussian process emulator as surrogate for the simulator. These diagnostics are based on comparisons between simulator outputs and Gaussian process emulator outputs for some test data, known as validation data, defined by a sample of simulator runs not used to build the emulator. Our diagnostics take care to account for correlation between the validation data. In order to illustrate a validation procedure, these diagnostics are applied to two different data sets.

1 Introduction

Simulators, also known as computer models, are mathematical representations of a physical system implemented in a computer. Simulators have been used to investigate real-world systems in almost all fields of science and technology (Sacks et al. 1989), usually because physical experimentation is either highly expensive or too time-consuming. In a computer experiment, observations are made by running the computer model at various choices of input factors.

Simulators are usually deterministic input-output models, where running the simulator again at the same input values will always give the same outputs. However, the output value is unknown before running the simulator for a particular input set. From a Bayesian perspective, uncertainty about the output of the simulator, also called code uncertainty, can be expressed by a stochastic process. The result is a statistical representation of the simulator, known as a statistical emulator. Statistical emulators have been developed in the 1980s by computer experimenters, e.g. Currin et al. (1988, 1991) and Sacks et al. (1989). O'Hagan

(1978) describes how to use a Gaussian process to represent an unknown function, and Gaussian processes are the principal tool for building an emulator to represent our judgements about the simulator. The Gaussian process emulator can be built using a set of runs of the simulator, known as training data.

Once the emulator has been built, various analyses can be made without more simulation runs. The simplest such analysis is to predict the output at inputs not previously run on the simulator. Welch et al. (1992) present an application of Gaussian processes to screening (input selection) and prediction in computer models. When there is uncertainty on the inputs, Monte Carlo methods applied on the simulator can be very expensive, Oakley and O'Hagan (2002) use the emulator to quantify uncertainty in model outputs induced by uncertainty in inputs. In order to understand how changes in the inputs affect the output, Saltelli, Chan, and Scott (2000) discuss some different measures to quantify sensitivity using the simulator. Oakley and O'Hagan (2004) presented a sensitivity analysis using the emulator where they provided Bayesian inference about sensitivity measures based on variance and regression fitting. Kennedy et al. (2006) present a number of recent applications in which an emulator of a computer code is created using a Gaussian process model. They presented three case studies from the Centre for Terrestrial Carbon Dynamics where sensitivity analysis and uncertainty analysis are illustrated.

Emulators have been used as stochastic approximations of expensive simulators in several areas of science, but in order to build emulators some assumptions and approximations are made. Unless the emulator correctly represents the simulator, inferences made using that emulator will be invalid. Hence, emulators need to be subjected to validation testing. There is a large literature of using emulators to represent expensive simulators, but there has been little research into validating emulators before using them. In this work, we propose some numerical and graphical diagnostics for Gaussian process emulators that take into account the computer model uncertainty.

In Section 2, we review the principal ideas of Gaussian process emulation. In Section 3, we briefly describe some methods that have been proposed for validating computer models, and then we present some numerical and graphical diagnostics for Gaussian process emulators. In Section 4, we demonstrate the diagnostic tools in synthetic and real examples. Where the emulator is failing to represent the simulator adequately, the diagnostics give warnings that allow the source of the validation problems to be identified.

2 Emulation

An emulator is a stochastic process that represents the simulator, where the simulator is viewed as an unknown mathematical function. Although the computer code is in principle known, its complexity allows the simulator output to be considered an unknown function of its inputs. From a Bayesian point of view Kimeldorf and Wahba (1970) and O'Hagan (1978) use Gaussian processes to describe the behaviour of an unknown mathematical function. In the 1980s, the fundamental idea of building a statistical emulator using Gaussian processes was introduced, in a non-Bayesian framework by Sacks et al. (1989), and by Currin et al. (1988, 1991) within a Bayesian framework.

2.1 Gaussian Process Emulators

We briefly review the principal ideas of the Gaussian process emulator; further discussion and detail may be found in Kennedy and O'Hagan (2001), and O'Hagan (2006). For a frequentist point of view, see also Santner et al. (2003, Section 3.3).

The simulator, represented by $\eta(\cdot)$, is assumed to be a function of a set of inputs denoted by $\mathbf{x} = (x_1, \ldots, x_p) \in \chi_1 \times \cdots \times \chi_p = \chi \subset \Re^p$, with output represented by $y \in \Re$. In order to build a Gaussian process emulator, the uncertainty about the simulator output is described as a Gaussian process with a particular mean function $m(\cdot)$, and a covariance function $V(\cdot, \cdot)$. Formally, if $\eta(\cdot)$ has a Gaussian process distribution then for every $n = 1, 2, \ldots$ the joint distribution of $\eta(\mathbf{x}_1), \ldots, \eta(\mathbf{x}_n)$ is multivariate normal for all $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \chi$. The mean function $m(\cdot)$ can be any function of $\mathbf{x} \in \chi$, but $V(\cdot, \cdot)$ must satisfy the property that every covariance matrix with elements $\{V(\mathbf{x}_i, \mathbf{x}_j)\}$ must be non-negative definite.

First, prior information about $\eta(\cdot)$ is represented by a Gaussian process with mean $m_0(\cdot)$ and covariance function $V_0(\cdot, \cdot)$. Using an hierarchical formulation

$$\eta(\cdot)|\beta, \sigma^2, \psi \sim GP\left(m_0(\cdot), V_0(\cdot, \cdot)\right), \tag{1}$$

where the mean function $m_0(\cdot)$ is given by

$$m_0(x) \quad = \quad h(x)^T \beta, \tag{2}$$

$h(\cdot) : \chi \subset \Re^p \longmapsto \Re^q$ is a known function of the inputs, where q can be different from the input space dimension p, β is an unknown vector of coefficients. The function $h(\cdot)$ should be chosen to incorporate any expert's belief about the form of $\eta(\cdot)$. The covariance function $V_0(\cdot, \cdot)$ is given by

$$V_0(x, x') \quad = \quad \sigma^2 C(x, x'; \psi), \tag{3}$$

σ^2 is an unknown scale parameter, $C(\cdot, \cdot; \psi)$ is a known correlation function with the unknown vector of correlation parameters. The chosen correlation function $C(\cdot, \cdot; \psi)$ should ensure that the covariance matrix of any set of inputs is non-negative definite. In this work, we use the Gaussian correlation function $C(\mathbf{x}, \mathbf{x}'; \psi) = \exp\{-\sum_{k=1}^{p} \left(\frac{x_k - x'_k}{\psi_k}\right)^2\}$, and the parameters $(\psi_1, \psi_2, \ldots, \psi_p)$ are called correlation length parameters.

Suppose $\mathbf{y} = [y_1 = \eta(\mathbf{x}_1), \ldots, y_n = \eta(\mathbf{x}_n)]$ contains n realizations of the simulator at design points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in the input space χ; these data are called the training dataset. According to (1) the distribution of the outputs is multivariate normal as follows:

$$\mathbf{y}|\beta, \sigma^2, \psi \sim N_n\left(H\beta, \sigma^2 A\right), \tag{4}$$

where

$$H = [h(\mathbf{x}_1), \ldots, h(\mathbf{x}_n)]^T \tag{5}$$

and A is the matrix with elements

$$A_{i,j} = C(\mathbf{x}_i, \mathbf{x}_j; \psi). \tag{6}$$

Using standard techniques for conditioning in multivariate normal distributions, it can be shown that

$$\eta(\cdot)|\beta, \sigma^2, \psi, \mathbf{y} \sim GP\left(m_0^*(\cdot), V_0^*(\cdot, \cdot)\right), \tag{7}$$

where

$$\begin{aligned} m_0^*(x) &= h(x)^T\beta + t(x)^T A^{-1}(\mathbf{y} - H\beta) \\ V_0^*(x,x') &= \sigma^2\left[C(x,x';\psi) - t(x)^T A^{-1} t(x')\right] \end{aligned}$$

where $t(x) = (C(x,\mathbf{x}_1;\psi),\ldots,C(x,\mathbf{x}_n;\psi))^T$.

Using a weak prior for (β,σ^2), $p(\beta,\sigma^2) \propto \sigma^{-2}$, combining with (4) using Bayes Theorem, the posterior for (β,σ^2) is a Normal Inverse Gamma distribution, characterised by:

$$\beta|\mathbf{y},\sigma^2,\psi \quad \sim \quad N\left(\widehat{\beta}, \sigma^2\left(H^TA^{-1}H\right)^{-1}\right) \tag{8}$$

where $\widehat{\beta} = \left(H^TA^{-1}H\right)^{-1}H^TA^{-1}\mathbf{y}$, and

$$\sigma^2|\mathbf{y},\psi \quad \sim \quad \mathrm{InvGam}\left(\frac{n-q}{2}, \frac{(n-q-2)\widehat{\sigma}^2}{2}\right) \tag{9}$$

where $\widehat{\sigma}^2 = \dfrac{\mathbf{y}^T\left(A^{-1} - A^{-1}H\left(H^TA^{-1}H\right)^{-1}H^TA^{-1}\right)\mathbf{y}}{n-q-2}$.

Integrating β out from the product of (7) and (8), it can be shown that

$$\eta(\cdot)|\mathbf{y},\sigma^2,\psi \sim GP\left(m_1(\cdot), V_1^*(\cdot,\cdot)\right) \tag{10}$$

where

$$\begin{aligned} m_1(x) &= h(x)^T\widehat{\beta} + t(x)^TA^{-1}(\mathbf{y} - H\widehat{\beta}), && (11) \\ V_1^*(x,x') &= \sigma^2\left[C(x,x';\psi) - t(x)^TA^{-1}t(x')\right. \\ &+ \left.\left(h(x) - t(x)^TA^{-1}H\right)(H^TA^{-1}H)^{-1}\left(h(x') - t(x')^TA^{-1}H\right)^T\right]. && (12) \end{aligned}$$

The Gaussian process emulator is obtained by integrating σ^2 out from the product of (9) and (10), and is given by

$$\eta(\cdot)|\mathbf{y},\psi \sim \text{Student Process}\left(n-q, m_1(\cdot), V_1(\cdot,\cdot)\right) \tag{13}$$

where

$$V_1(x,x') = \frac{\widehat{\sigma}^2}{\sigma^2}V_1^*(x,x'). \tag{14}$$

The hyperparameter vector ψ is unknown. Setting a prior for it as $p(\psi)$, it can be shown that

$$\begin{aligned} p(\psi|\mathbf{y}) &\propto p(\psi)\int\int p(y|\beta,\sigma^2,\psi)p(\beta,\sigma^2)d\beta d\sigma^2 \\ &\propto p(\psi)|A|^{-\frac{1}{2}}|H^TA^{-1}H|^{-\frac{1}{2}}(\hat{\sigma}^2)^{-\frac{n-q}{2}} && (15) \end{aligned}$$

where A and $\hat{\sigma}^2$ are functions of ψ.

A fully Bayesian analysis would now integrate out the hyperparameter ψ from the product of the densities in (13) and (15). However, the posterior distribution (15) is a highly intractable function of ψ. Kennedy and O'Hagan (2001) proposed to derive a plausible estimate of the hyperparameter vector ψ and then use the estimate as if it were the true value of ψ. The new emulator is the same as (13) with the estimated values for A, $\widehat{\beta}$, and $\widehat{\sigma}^2$ calculated using the estimated value of ψ. We assume here that this approach is used.

2.2 Possible problems with Gaussian process emulators

Although the Gaussian process is a very flexible class of distributions to represent prior knowledge about the computer model, the Gaussian process emulator (13) can give poor predictions of simulator outputs for at least two basic reasons. First, the assumption of a stationary Gaussian process with particular mean and covariance structures may be inappropriate. Second, even if these assumptions are reasonable there are various parameters to be estimated, and a bad or unfortunate choice of training dataset may suggest inappropriate values for these parameters. In the case of the correlation length parameters, where we condition on fixed estimates rather than integrating over the full posterior distribution, we may also make a poor choice of estimate to condition on.

The simulator is represented by a Gaussian process, so joint normality of the simulator outputs has to be a reasonable assumption. In particular, the emulator asserts that it is very unlikely for the true simulator output to be more than two or three predictive standard deviations from the predictive mean, and that it is no more likely to be above the predictive mean than below it. In this context transformations may be useful.

In addition to the assumption of normality, specific forms are assumed for the mean and the covariance functions. If the assumed form of the mean in (2) is wrong, because inappropriate regressors have been used in $h(\cdot)$, or if the coefficients β have been poorly estimated, then the emulator predictions may be systematically too low or too high in some regions of the input space.

In (3), stationarity is assumed for the covariance function, implying that we expect the simulator output to respond with similar degrees of smoothness at all points in the input space. It is assumed that there is a common variance σ^2 and the correlation function depends only on $(\mathbf{x} - \mathbf{x}')$. Hence, either the unequal variance problem or a correlation structure that depends on spatial position $(\mathbf{x}, \mathbf{x}')$ instead of the difference $(\mathbf{x} - \mathbf{x}')$ only are failures of the stationarity assumption. In practice, simulators may respond much more rapidly to changes in the inputs at some parts of the space than others. In case of such non-stationarity, credible intervals of emulator predictions can be too wide in regions of low responsiveness or too narrow in regions where the response is more dynamic.

Finally, although the form of the covariance function may be appropriate, we may estimate the parameters σ^2 and ψ poorly. When we have incorrect estimation of the variance (σ^2), the credible intervals of the emulator predictions are systematically too wide or too narrow. Poor estimation of the correlation parameters (ψ) leads to credible intervals that are too wide or too narrow in the neighbourhood of the training data points.

In the next section, we present some diagnostics that can be useful to identify problems in emulator predictions. These diagnostics are based on statistical comparisons between new (validation) runs of the simulator and their respective predictions.

3 Diagnostics for Validating Gaussian Process Emulators

Emulators have been used as stochastic approximations of expensive simulators in several areas of science, but in order to build emulators some assumptions and approximations are made. Unless the emulator correctly represents the simulator, inferences made using that emulator will be invalid. Hence, the emulators need to be subjected to validation testing.

In computer science and engineering, the process of checking if the computer model represents the real process that it was intended to simulate is generally divided into two steps, a verification and a validation step, V&V. Verification is the process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model. Validation is the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model (Oberkampf and Trucano 2000). There are several different perspectives and approaches for V&V, such as philosophical theories about validation, statistical techniques, software practices, etc. Some reviews of the literature about V&V are given in Balci and Sargent (1984), Kleijnen (1995), Roache (1998).

Where Gaussian process emulators have been used, there is some literature that considers validation of computer models by comparing model predictions to observed data. Bayarri et al. (2007) present a six-step process for computer model validation based on Bayesian and likelihood methodology. Rougier et al. (2007) presented a 'leave-one-out' diagnostic, where they remove one element from the training data and try to predict it. They repeat this procedure for all elements and plot credible intervals for each element. They also presented a diagnostic where they leave out more than one element. Kennedy and O'Hagan (2001) use Quantile-quantile plots of the standardized residuals of their calibration model, and use the root-mean-square errors of the predictions in order to compare different models. Goldstein and Rougier (2006) present a diagnostic based on the deviation between real observations and Bayes linear predictions of the simulator for the same inputs.

However, V&V methods are concerned with comparing computer models with reality, and have assumed independent errors in the observations, whereas the work we present here focuses on validating emulators as surrogates for expensive computer models. Hence, our validation process is based on comparing the Gaussian process emulator with the simulator. Emulator predictions are not independent and it is important for diagnostics to take account of the correlations. Santner et al. (2003, page 109) use the empirical root mean squared prediction error or the integrated mean squared prediction error, but without reference to whether this matches the uncertainty, including correlations, expressed in the emulator itself.

3.1 Diagnostics for linear models with correlated residuals

Validating an emulator can be compared with criticising linear models with dependent errors, since the Gaussian process is a limit case of the general linear model. However, the Gaussian process emulator is modelling a deterministic function, hence the predictions for observed values used to build the model are perfect, so residuals can only be obtained using cross validation methods, or, more appropriately, a new data set. Therefore, the diagnostics used in general linear models need to be adapted in order to be applied on the computer model framework.

In the context of general linear models, Haslett and Hayes (1998) presented a general theory for residuals, where they describe the marginal and the conditional residuals. The marginal residuals are the errors between the observed values and the fitted values, whereas the conditional residuals are the errors between the predictive values for observed values not used to build the model. Fraccaro et al. (2000) presented graphical diagnostics for marginal and conditional residuals in a time series regression context. The authors decompose the estimated variance matrix of the residuals using a Cholesky decomposition, and rotate them in order to have uncorrelated residuals with unit variance. Houseman et al. (2004) used the Cholesky decomposition of the inverted covariance matrix to rotate the residuals of linear mixed models and time series. The authors provided a Quantile-quantile plot of these uncorrelated errors providing asymptotic properties of the empirical cumulative distribution and pointwise standard errors.

The Gaussian process emulators model deterministic functions, indicating that there are no marginal errors as defined by Haslett and Hayes (1998). Specifically, emulator predictions for outputs used to build the emulator will exactly equal the respective outputs, with zero variances. We can only obtain prediction errors for simulator runs not used to build the emulator, the conditional errors in Haslett and Hayes (1998). Another important feature that should be considered in the diagnostics for validating emulators is the error correlation due to a spatial correlation structure.

Graphical diagnostics presented by Fraccaro et al. (2000), who plot uncorrelated conditional errors against the data order, and by Houseman et al. (2004), who use a QQ-plot for the uncorrelated conditional errors, will be used in this work. However, the Cholesky decomposition depends on the data order. In Fraccaro et al. (2000) the data are indexed by time, but in computer modelling the data order is typically arbitrary. Therefore, it is necessary to present some decomposition methods that are invariant to the data order. In this work, we propose to use the eigen and the pivoted Cholesky decompositions to build the uncorrelated errors, as will be discussed in Section 3.4.

In order to validate an emulator, our diagnostics will be based on comparisons between emulator predictions and simulator runs for a new data set. Let $\mathbf{X}^* = (\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_m^*)$ denote a non-observed set of inputs, called validation input data. The simulator outputs for the validation input data are given by $\mathbf{y}^* = \eta(\mathbf{X}^*)$ where $\mathbf{y}^* = (y_1^*, \ldots, y_m^*)$, and $\eta(\mathbf{X}^*) = (\eta(\mathbf{x}_1^*), \ldots, \eta(\mathbf{x}_m^*))$. The validation input data should be selected to cover the whole of that part of the input space over which we wish to use the emulator. Otherwise, we might validate an emulator which does not represent the simulator for a particular non-observed subset of the input space.

A general diagnostic $D(\cdot)$ is a function of the validation data output, and we propose to compare the observed $D(\mathbf{y}^*)$ with the reference distribution of $D(\eta(\mathbf{X}^*))$ conditioned on the training data. If $D(\mathbf{y}^*)$ lies in an appropriately chosen region with a small probability, then this suggests that there is a conflict between the emulator and the simulator. The test region(s) for $D(\cdot)$ should be chosen so that $D(\mathbf{y}^*)$ falling in the region is associated with a particular failure in the construction of the emulator. If across a range of such diagnostics there are no indications of conflict, then we can suppose that the emulator is representing the simulator accurately.

3.2 Individual prediction errors

Individual prediction errors for the validation data are given by the differences between the observed simulator outputs and the predictive mean output at the same inputs, i.e. $(\mathbf{y}_i^* - E[\eta(\mathbf{x}_i^*)|\mathbf{y}])$, for $i = 1, 2, \cdots, m$.

We can consider each standardised prediction error

$$D_i^I(\mathbf{y}^*) = \frac{y_i^* - E[\eta(\mathbf{x}_i^*)|\mathbf{y}]}{\sqrt{V[\eta(\mathbf{x}_i^*)|\mathbf{y}]}} \tag{16}$$

as a diagnostic. If the emulator can represent properly the simulator, the standardized prediction errors have standard Student-t distributions, conditional on the training data and the estimated correlation parameters ψ. In practice, the number of training data is generally large enough so that the degrees of freedom is large and we can consider these to be standard normally distributed. Hence, individual large errors, with absolute values larger than 2, say, indicate a conflict between the simulator and the emulator. An isolated outlier of this kind might be ignored, or might indicate a local problem just around the inputs for that validation data point. This can be investigated further by obtaining a few more validation data runs in that vicinity.

If there are a larger number of large standardised errors, this would indicate a more systematic problem. If large errors of the same sign arise in some part of the input space, this suggests an inappropriate choice of mean function or poor estimation of β. It may also be indicative of a failure of the stationarity assumption.

If large errors arise primarily in validation points that are close to training data points, this indicates that one of more of the correlation parameters have been over-estimated, so that the emulator predictions are too strongly influenced by nearby training data points. If there are no such obvious patterns to the occurrence of large errors, then the problem may lie in poor estimation of the σ^2 parameter.

It should be noted that patterns of unexpectedly small standardised errors may indicate conflicts complementary to those discussed above. For instance, if validation points close to training data points give unexpectedly small standardised errors, this suggests under-estimation of correlation parameters. Graphical displays can be powerful ways of spotting patterns of large or small errors, and are discussed in Section 3.5.

3.3 Mahalanobis distance

Although the collection of individual standardised errors $D^I(\mathbf{y}^*)$ provide a range of useful diagnostics, it is also important to be able to summarise them in a single diagnostic.

Hills and Trucano (1999, 2001) use a χ^2-test to compare the simulator output with the real process in a V&V approach. The same idea can be used as a diagnostic to compare emulator predictions with the simulator outputs under the same inputs. Their diagnostic is given by

$$D_{\chi^2}(\mathbf{y}^*) = \sum_{i=1}^{m} D_i^I(\mathbf{y}^*)^2. \tag{17}$$

For a large training dataset, i.e. when $n \to \infty$, and with independence among the output values, the distribution of $D_{\chi^2}(\eta(\mathbf{X}^*))$ converges to a chi-squared distribution with m degrees of freedom. However, the independence assumption is too strong. For example, if the simulator is a smooth function, then similar outputs are expected when the elements are close to each other in the input space. This correlation is captured by the emulator in the covariance function (11).

A natural extension of (17) allowing the correlation among the outputs is the Mahalanobis distance between the emulator and the simulator outputs at the validation inputs set, given by

$$D_{MD}(\mathbf{y}^*) = \left(\mathbf{y}^* - E\left[\eta(\mathbf{X}^*)|\mathbf{y}\right]\right)^T \left(V\left[\eta(\mathbf{X}^*)|\mathbf{y}\right]\right)^{-1} \left(\mathbf{y}^* - E\left[\eta(\mathbf{X}^*)|\mathbf{y}\right]\right) , \tag{18}$$

where the elements of the predictive mean vector $E\left[\eta(\mathbf{X}^*)|\mathbf{y}\right]$ and the predictive covariance matrix $V\left[\eta(\mathbf{X}^*)|\mathbf{y}\right]$ for the Gaussian process emulator are given respectively by (11) and (14). Extreme values (large or small) for the observed Mahalanobis distance ($D_{MD}(\mathbf{y}^*)$) indicate a conflict between the emulator and simulator.

Under Gaussian process emulator assumptions, the distribution of $D_{MD}(\eta(\mathbf{X}^*))$ conditional on the training data and an estimate of the correlation parameter ψ is a scaled F-Snedecor distribution with m and $n-q$ degrees of freedom,

$$\frac{(n-q)}{m(n-q-2)} D_{MD}(\eta(\mathbf{X}^*))|\mathbf{y}, \psi \sim F_{m,n-q}. \tag{19}$$

Proof: Using (11) and (14), (18) can be rewritten as follows

$$D_{MD}(\mathbf{y}^*) = \frac{Z}{W}$$

where $Z = (y^* - m_1(\mathbf{X}^*))^T \left(V_1^*(\mathbf{X}^*)\right)^{-1} (y^* - m_1(\mathbf{X}^*))$ and $W = \hat{\sigma}^2/\sigma^2$. By (10), Z conditional on the training data, σ^2 and ψ follows a chi-squared distribution with m degrees of freedom. And, by (9), $(n-q-2)W$ conditional on the training data and an estimate for ψ follows a chi-squared distribution with $(n-q)$ degrees of freedom. It is straightforward to show that Z and W are independent random variables. Hence, $\frac{Z/m}{(n-q-2)W/(n-q)}$ follows a F-Snedecor distribution with degrees of freedom m and $n-q$. □

As previously mentioned, an unexpectedly large or small value of $D_{MD}(\mathbf{y}^*)$ indicates a conflict between the emulator and the simulator. If such a problem arises, it is important to explore individual errors, to look for patterns of large or small values, so as to identify the most likely cause of the problem. We now consider alternative ways to decompose the Mahalanobis distance into individual diagnostics for this purpose.

3.4 Variance decompositions

Individual prediction errors (16) are correlated, which introduces some risks in interpreting them. Also, looking at individual errors may not effectively identify some conflicts between the emulator and simulator. For instance, two errors may not individually be large, but if they have opposite signs when they are strongly positively correlated, then this suggests a conflict. Let $\mathbf{G}$ be a standard deviation matrix such that $V[\eta(\mathbf{X}^*)|\mathbf{y}] = \mathbf{G}\mathbf{G}^T$. Then the vector of transformed errors

$$D_{\mathbf{G}}(\mathbf{y}^*) = \mathbf{G}^{-1}(\mathbf{y}^* - E[\eta(\mathbf{X}^*)|\mathbf{y}]) \tag{20}$$

are uncorrelated and have unit variances. If the normality assumption made for the outputs is reasonable, the distribution of each of these errors is a standard Student-t with $(n-q)$ degrees of freedom. We can consider these as an alternative set of diagnostics. As with the errors $D^I(\mathbf{y}^*)$, we look for individual large transformed errors and patterns of large and small values. However, the structure of $\mathbf{G}$ will give different interpretations to such patterns. This approach is similar to that of Houseman et al. (2004), who use rotated residuals for linear models with correlated errors.

Another property of this diagnostic is that $D_{MD}(\mathbf{y}^*) = D_{\mathbf{G}}(\mathbf{y}^*)^T D_{\mathbf{G}}(\mathbf{y}^*)$. That is, the sum of squares of the elements of $D_{\mathbf{G}}(\mathbf{y}^*)$ is the Mahalanobis distance, and hence we can interpret these diagnostics as decomposing $D_{MD}(\mathbf{y}^*)$.

There are many ways to decompose a positive definite matrix into the product of a square root matrix and its transpose. The natural choices are the Cholesky decomposition and the eigen decomposition, Golub and van Loan (1996). The eigen decomposition is very popular, but the Cholesky decomposition is computationally cheaper and we shall see that it is more intuitive to interpret than the eigen decomposition.

Eigen decomposition. When $\mathbf{G}$ is the eigen decomposition matrix, we denote the elements of the vector $D_{\mathbf{G}}(\mathbf{y}^*)$ by $D_i^E(\mathbf{y}^*)$, and call them eigen errors. When a large $D_i^E(\mathbf{y}^*)$ is identified, further information may be gained by studying which individual errors are given the largest weights in the linear combination. If the weights single out as important a particular individual error, then this error should be studied as suggested in Section 3.2. If the weights emphasize a subset of the individual prediction errors, then it might indicate a problem in the region of the input space around the inputs of the validation data points associated, indicating a possible non-stationarity problem.

Cholesky decomposition. The Cholesky decomposition is the special case where $\mathbf{G}^T$ is the unique upper triangular matrix $\mathbf{R}$ such that $V[\eta(\mathbf{X}^*)|\mathbf{y}] = \mathbf{R}^T\mathbf{R}$, and we denote the elements of the vector $D_{\mathbf{G}}(\mathbf{y}^*)$ by $D_i^C(\mathbf{y}^*)$, and call them Cholesky errors. Then $\mathbf{G}^{-1}$ is also a triangular matrix, and $D_i^C(\mathbf{y}^*)$ is the unique linear combination of the first i validation errors such that its predictive variance is the conditional variance of the i-th validation error given the preceding $i-1$ errors. Although this has the benefit of producing a set of uncorrelated transformed errors that are still linked to the individual validation points (in contrast to the eigen decomposition), the decomposition is not invariant to how we order the validation points, and patterns of high or low values have no obvious interpretation.

Pivoted Cholesky decomposition. By permuting the validation data set, we obtain different Cholesky decompositions. In order to have the benefits of both the eigen and Cholesky decompositions, we propose to permute the data so that the first element is the one with the largest variance, the second element is the one with the largest predictive variance conditioned on the first element, and so on. We then denote the elements of the vector $D_{\mathbf{G}}(\mathbf{y}^*)$ by $D_i^{PC}(\mathbf{y}^*)$, and call them pivoted Cholesky errors. This permutation can be obtained by applying the pivoted Cholesky decomposition, which returns a permutation matrix $\mathbf{P}$ and the unique upper triangular matrix $\mathbf{R}$ such that $\mathbf{P}^T V[\eta(\mathbf{X}^*)|\mathbf{y}]\mathbf{P} = \mathbf{R}^T\mathbf{R}$. So $\mathbf{G} = \mathbf{P}\mathbf{R}^T$. More details about the numerical analysis of the pivoted Cholesky decomposition can be found in Higham (2002).

A group of unusually large or small pivoted Cholesky errors in the first part of the sequence suggest poor estimation of σ^2 or non-homogeneity, while a number of unusually large or small errors in the latter part of the sequence indicate poor estimation of ψ or an inappropriate correlation structure. In addition, we have the benefit that each of the $D_i^{PC}(\mathbf{y}^*)$s is associated with a particular validation data point, which makes it easy to investigate individual large errors. The pivoted Cholesky decomposition will therefore be our choice in the examples of Section 4.

3.5 Graphical methods

Graphical displays are efficient ways to investigate the adequacy of the emulator predictions and to check some assumptions made to build the emulator (13). We propose some graphical methods using both the individual standardized errors (16) and the uncorrelated standardized errors (20).

Plot of the individual errors against the emulator's predictions. In this graphical diagnostic, we search for patterns suggesting a problem in the mean function. For example, if for some particular ranges of the output the errors are systematically positive (or negative) this indicates a misspecification of the mean function or poor estimation of the coefficients. Heteroscedasticity of the individual errors suggests that the simulator should be studied as a non-stationary process. Large absolute individual errors might suggest that the predictive variance is too small, and individual errors very close to zero might suggest a too large variance.

In addition to plotting individual errors $D^I(\mathbf{y}^*)$ in this way, we can plot the uncorrelated standardized errors obtained by the Cholesky or pivoted Cholesky decomposition, because each error can be mapped to one emulator prediction. However, we are then less likely to see groups of systematic deviations indicating problems with the mean function. Consider, for instance, a group of positive individual errors in some part of the plot. These may arise from validation points that are relatively close together in the input space. The first of these points to be plotted in the Cholesky or pivoted Cholesky sequences $D_i^C(\mathbf{y}^*)$ or $D_i^{PC}(\mathbf{y}^*)$ may show up in the plot as a large error, but the subsequent ones are conditioned on the first and may appear normal.

We cannot plot the uncorrelated standardized errors obtained by eigen decomposition in this way.

Plot of the errors against the index. The meaning of the index depends on which error we are plotting. For individual errors $D^I(\mathbf{y}^*)$ and Cholesky errors $D_i^C(\mathbf{y}^*)$, the index i is the validation data order. For eigen errors, the index gives the order of the $D_i^E(\mathbf{y}^*)$s with the largest predictive variance. And for pivoted Cholesky errors, the index is the pivoting order, which gives the order of the $D_i^{PC}(\mathbf{y}^*)$s with the largest conditional predictive variance. For all these graphics, it is expected that the errors should be fluctuating around zero with a constant variance and no special patterns. Too many large errors indicates an under-estimation of the variance. On the other hand, too many small errors indicates an over-estimation of the variance. In both cases, it can also suggest that the simulator is a non-stationary process.

The pivoted Cholesky and the eigen decomposition provide an extra interpretation that we can associate with the correlation structure. In both cases, if we observe either large or very small errors at the beginning of the plot, i.e. on the left-hand side, it indicates a failure of estimation of predictive variance, or non-stationarity. If we observe large (or very small) errors at the end of the plot, however, i.e. on the right-hand side, it indicates that the correlation length parameters were over (under) estimated or the chosen correlation structure is unsuitable.

Quantile-quantile plots. Under the normality assumption, the uncorrelated standardized errors $D_G(\mathbf{y}^*)$ have standard Student-t distributions with $(n-q)$ degrees of freedom. So, the quantile-quantile plot (QQ-plot) using this distribution becomes a natural graphical diagnostic. In a QQ-plot, if the points lie close to the 45-degree line through the origin, the normality assumption for the simulator outputs is a reasonable assumption. If the points cluster around a line with slope less (or greater) than one, the implication is that the predictive variability was over-estimated (or under-estimated).

Curvature in the plot indicates non-normality, while outliers at either end of the plot suggest local fitting problems or non-stationarity.

The interpretation of the QQ-plot using uncorrelated standardized errors is independent of the decomposition method, and is generally informative for both eigen and pivoted Cholesky decompositions. Although the distribution of each individual standardized error, $D_i^I(\mathbf{y}^*)$, is also a standard student-t distribution, the fact that the errors are correlated makes the QQ-plot more difficult to interpret.

Plots of errors against inputs. Plotting the standardized errors against the corresponding values of each input is also helpful. Again, we expect to see a horizontal band containing the errors. These plots are used to identify different behaviour of the errors in some parts of the input space, indicating possible failure of the stationarity assumption. This graphic can also indicate that the relationship between the input and the prediction is not fully represented in the mean function. For example, we can identify a pattern that was not included in the mean function.

Notice that we cannot plot the eigen errors in this way. Whilst it is possible to plot the Cholesky or pivoted Cholesky errors against input values, because of the linking of each error to a validation data point, interpretation is complicated by the conditioning in the same way as when plotting against the emulator mean.

3.6 Other diagnostics

Credible interval diagnostic. Another diagnostic is the proportion of validation outputs which lie in their marginal credible intervals. For each validation element using (13) a $100\alpha\%$ credible interval for the simulator output can be built, denoted by $CI_i(\alpha)$ for $i = 1, \ldots, m$. This diagnostic is given by

$$D_{CI}(\mathbf{y}^*) = \frac{1}{m}\sum_{i=1}^{m} \mathbf{1}(\mathbf{y}_i^* \in CI_i(\alpha)) \tag{21}$$

where $\mathbf{1}(\cdot)$ is an indicator function. We expect that the observed value for $D_{CI}(\mathbf{y}^*)$ should be close to α. However, because the outputs are not independent the reference distribution of $D_{CI}(\cdot)$ is not binomial. The only practical way to compute the reference distribution is by simulation.

The distribution of $D_{CI}(\cdot)$ can be obtained by the following Monte Carlo simulation. Sample a large number of samples from Multivariate Student-t $(n-q, E[\eta(\mathbf{X}^*)|\mathbf{y}], V[\eta(\mathbf{X}^*)|\mathbf{y}])$, then for each sample calculate $D_{CI}(\cdot)$. Therefore, the empirical distribution of the calculated $D_{CI}(\cdot)$s is a good estimate of the distribution of $D_{CI}(\eta(\mathbf{X}^*)$. In particular, the mean and the square of standard deviation are respectively estimates of the expectation and the variance of $D_{CI}(\eta(\mathbf{X}^*)$.

This diagnostic is a supplement to the Mahalanobis distance. For example, we can have many unusually large and unusually small errors and yet still have an acceptable value for $D_{MD}(\mathbf{y}^*)$. The $D_{CI}(\mathbf{y}^*)$ diagnostic offers a way to identify this kind of heterogeneity.

Predictive density diagnostic. It is worth mentioning here another interpretation of the Mahalanobis distance. Since the distribution of validation outputs is a multivariate Student-t distribution with mean $m_1(\mathbf{X}^*)$, covariance matrix $V_1(\mathbf{X}^*)$ and $n-q$ degrees of freedom, the density function itself can be a diagnostic.

$$D_{PD}(\mathbf{y}^*) = K\left[1 + \frac{1}{n-q} D_{MD}(\mathbf{y}^*)\right]^{-\frac{m+n-q}{2}} \tag{22}$$

where $K = \frac{\Gamma\left(\frac{m+n-q}{2}\right)}{\Gamma\left(\frac{n-q}{2}\right)}(n-q)^{-m}\pi^{-\frac{m}{2}}|V_1(\mathbf{X}^*)|^{-\frac{1}{2}}$. A small value for this diagnostic would indicate a conflict between the emulator and simulator. Note, however, that $D_{PD}(\mathbf{y}^*)$ is just a decreasing function of the Mahalanobis distance $D_{MD}(\mathbf{y}^*)$, and so small values of the predictive density correspond directly with large values of the Mahalanobis distance.

4 Examples

In this section, we use two data sets to illustrate the proposed diagnostics for Gaussian process emulators. The first example is an artificial model with two inputs, while the second example is a real model of reflectance for a homogeneous plant canopy. In both examples, the prior relationship between the output and the inputs is represented by the mean function (2) with $h(\mathbf{x})^T = (1, \mathbf{x}^T)$. This envisages a linear trend in response to each input, which is widely used for Gaussian process emulation. Although computer models are in practice almost certainly nonlinear, we often have little prior knowledge of what form the nonlinearity will take.

4.1 Two-input toy model

We begin with an artificial model having only two inputs but nevertheless of a challenging form for emulation. We suppose that the simulator encodes the following mathematical function:

$$y = \eta(x_1, x_2) = \left(1 - e^{-\frac{1}{2x_2}}\right)\left(\frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^3 + 500x_1^2 + 4x_1 + 20}\right), \quad x_i \in (0,1), i = 1, 2. \tag{23}$$

The training data are composed of 20 points selected by a Latin Hypercube sampling, McKay et al. (1979). The validation data are composed of 25 points independently selected by another Latin Hypercube sampling. Using the training data, the estimated correlation length parameters (15) are $(\hat{\psi}_1, \hat{\psi}_2) = (0.2421, 0.4240)$, indicating that the simulator is more smooth with respect to the second input than the first. The estimated variance is $\widehat{\sigma}^2 = 3.3316$.

The observed chi-square diagnostic, $D_{\chi^2}(\mathbf{y}^*) = 24.411$, is very close to its expected value $E[D_{\chi^2}(\eta(\mathbf{X}^*))] = 25$, suggesting that the emulator is a good approximation of the simulator. However, this diagnostic ignores the fact that the outputs are correlated. Table 1 presents the observed Mahalanobis distance and credible interval diagnostics, and some statistics of their predictive distributions.

The observed Mahalanobis distance, $D_{MD}(\mathbf{y}^*) = 70.36$, is an extreme value of its theoretical distribution, a scaled F distribution with parameters (25,18). This indicates a conflict between the emulator and the simulator. The observed credible interval diagnostic is less dramatic. We see that 92% (23 out of 25) of the simulator outputs lie in their respective 95% marginal credible intervals built by the emulator. The lower quartile of the distribution of $D_{CI}(\eta(\mathbf{X}^*))$ obtained via simulation, is 0.92, and indeed it is not surprising to have 2 values out of 25 lying outside their 95% intervals.

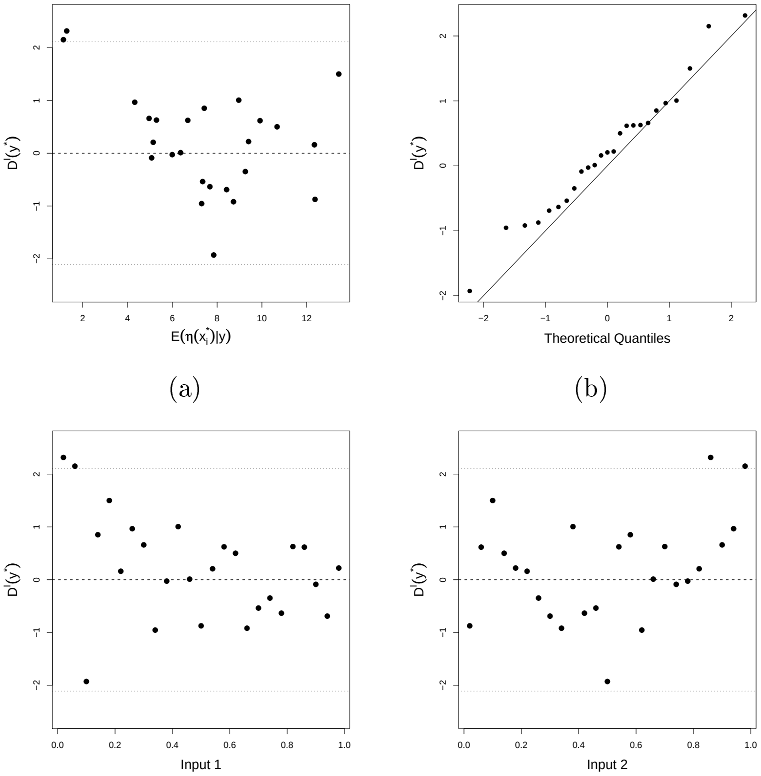
The chi-square and credible interval diagnostics suggest that emulator predictions are marginally satisfactory but the Mahalanobis distance makes it clear that jointly they are far from valid. This may be related to poor estimation of the correlation length parameters or to a non-homogeneity in the input space.

Table 1: The observed Mahalanobis distance and credible interval diagnostics, with some summaries of their predictive distributions, for the toy example.

	Obs.	Expected	Std. dev.	1stQ	Median	3rdQ
$D_{MD}(\cdot)$	70.360	25.000	12.404	16.016	21.778	29.438
$D_{CI}(\cdot)$	0.920	0.951	0.072	0.920	1.000	1.000

Figure 1 presents some graphical diagnostics using the individual standardized errors. Figure 1 (a) presents the individual standardized errors against the emulator predictions given by the expected value. There is no obvious pattern, although the two largest individual errors are associated with small values of the predictions, which might indicate a problem in the mean function or an stationarity problem. Figure 1 (b) is the QQ-plot of the individual errors, and supports the finding of the chi-square and credible interval diagnostics that the predictions appear valid marginally.

In order to check a possible stationarity problem, the individual standardized errors are plotted against each input, Figure 1 (c) and (d). The two large errors are associated with small values of the input 1, and large values of the input 2. This might be connected to a sub-region of the input space not represented in the training data. If possible, new runs of the simulator in this sub-region may improve the emulator. Also, it can be noticed in Figure 1 (c) that the larger the value of input 1, the smaller is the variability of the individual errors. This can indicate non-stationarity, or perhaps an over estimation of the correlation length ψ_1. For the second input, Figure 1 (d), there is no particular pattern to the errors.



(a)

(b)

(c)

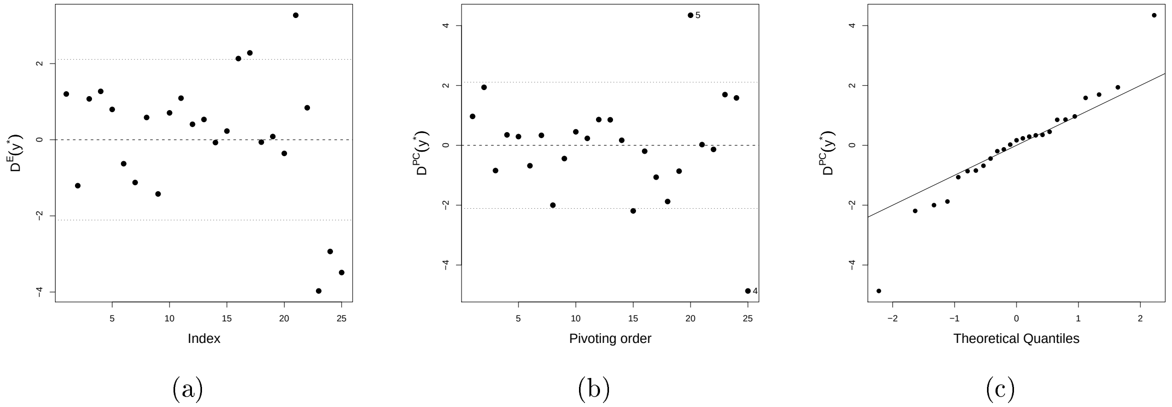
(d)

Figure 1: Graphical diagnostics for the toy example using the individual standardized errors: (a) $D^I(\mathbf{y}^*)$ against the emulator predictions; (b) quantile-quantile plot; (c) $D_i(\mathbf{y}^*)$ against input 1; (d) $D_i(\mathbf{y}^*)$ against input 2.

The diagnostics presented in Figure 1 ignore the correlation structure of the errors, and so do not address the problem found with the Mahalanobis distance. Figure 2 presents graphical diagnostics using the uncorrelated standardized errors. The eigen errors are presented in Figure 2 (a), where we can observe large errors at the end of the plot. This clearly indicates a problem on the correlation structure, which can be either a misspecification of the correlation function, or an over-estimation of the correlation length parameters. When we examine the weights given by the eigen vector for the large errors, we cannot identify a pattern for the components of the 16th, 17th and 23rd eigen errors. However, the components of the 21st eigen value indicate the 5th validation point as a very important point for the size of this eigen error. And all validation points related to the largest weights of the 24th and 25th eigen errors present values for input 1 smaller than 0.5.

The pivoted Cholesky errors are presented in Figure 2 (b). Large errors are again observed at the end of the plot in this diagnostic, suggesting a problem with the correlation structure. However, with this plot it is easier to explore the nature of the problem, because there are only two large values and we can link each one to a single validation data point. Their input vectors are (0.14, 0.58) and (0.18, 0.10). These two points are different from the two large individual standardized errors, but they also are characterized by small values of input 1, suggesting that the emulator predictions are not valid over this part of the input space.

Figure 2 (c) presents the quantile-quantile plot of the pivoted Cholesky errors, where it can be noticed that the points cluster around a line with slope slightly greater than one, so that there may be a small under-estimation of the predictive variability. However, the two outliers, which were the two large values in Figure 2 (b), are the most striking feature of this plot.



(a) (b) (c)

Figure 2: Graphical diagnostics for the toy example using the uncorrelated standardized errors: (a) the eigen errors $D^E(\mathbf{y}^*)$, against the eigenvalue number; (b) the pivoted Cholesky errors $D^{PC}(\mathbf{y}^*)$, against the pivoting order; (c) quantile-quantile plot of the pivoted Cholesky errors.

In summary, the numerical and graphical diagnostics indicate some conflicts between the emulator and the simulator. The conflict seems to be related to poor estimation of the correlation parameters, and perhaps to non-stationarity and the training data not adequately covering a sub-region of input space where X_1 takes values below 0.2.

Since the simulator (23) is a simple function we can quickly run it more times. So, we combine the training data with the validation data, plus another 5 observations randomly selected in a sub region of the input space where both inputs are smaller than 0.5. Using the updated training data, the estimated correlation length parameters are $(\hat{\psi}_1, \hat{\psi}_2) = (0.1764, 0.4116)$, which are smaller than the previous estimates, confirming the suspicion of over-estimation, especially ψ_1. And also the estimated variance is greater than the previous estimate, $\widehat{\sigma}^2 = 4.9389$, supporting the suspicion of under-estimation of the variability of the process.

A new validation data set using a Latin Hypercube sampling was selected, and the Mahalanobis distance and credible interval diagnostics are presented in Table 2. The Mahalanobis distance is still higher than expected, but now suggests much less conflict with the simulator. The credible interval diagnostic again indicates that the emulator predictions for the validation data are individually reasonable.

Table 2: The observed Mahalanobis distance and credible interval diagnostics, with some summaries of their predictive distributions, for the toy example after new training data.

	Obs.	Expected	Std. dev.	1stQ	Median	3rdQ
$D_{MD}(\cdot)$	51.129	30	10.230	23.172	28.958	36.324
$D_{CI}(\cdot)$	0.933	0.950	0.058	0.933	0.967	1.000

Figure 3 (a) presents the individual standardized errors. There is no obvious pattern, and although there are some errors outside the credibility bounds, they are not large enough to suggest a serious conflict. Figure 3 (b) suggests that the emulator is underestimating the highest output values, and so is perhaps not adequately capturing the peak of the output surface.

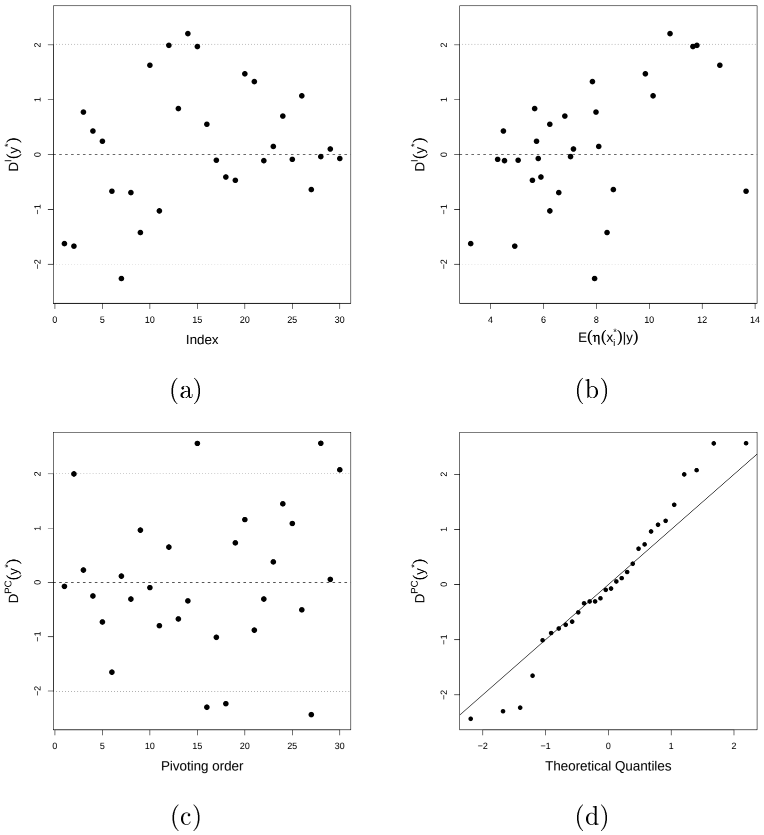
The pivoted Cholesky errors are presented in Figure 3 (c). Although there are no very large errors, there are too many outside the bounds. The QQ-plot of the pivoted Cholesky errors, Figure 3 (d), confirms that the emulator's predicted variability is a bit smaller than the observed variability.

Although the diagnostics indicate that there may still be some validation problems, the emulator built with the updated training data does improve the predictions. At this point we may make a final build of the emulator using all the simulator runs (the original training data, the original validation data, the extra 5 points and the final validation data). We will expect that the final emulator will validate well, in view of the very limited problems found in the second round of validation testing, and not seek to test it again.

Figure 4 shows the steady improvement of the emulator. Figure 4 (a) shows the emulator mean plotted against the two inputs, based on just the original training data. Figures 4 (b) and (c) show how the emulator evolves as we add the original validation data and the extra 5 points, and then when we add in the additional validation data. The training data for each emulator and also the validation data used to build the diagnostics are illustrated on each graphic. Figure 4 (d) presents the true value of the simulator (a plot that will not generally be available to us with a real simulator). Figure 4 (d) shows that this is a difficult function to emulate, which is very sensitive to input 1 when its value is small. The original emulator does not capture this behaviour, but after adding the original validation data and 5 more points the correct shape is emerging. The final emulator in Figure 4 (c) does a very good job of representing the simulator.

4.2 Nilson-Kuusk Model

In this section, a real dataset is used as example for the proposed diagnostics. The simulator was built based on the Nilson-Kuusk model, which is a reflectance model for a homogeneous plant canopy. The simulator



(a) (b)

(c) (d)

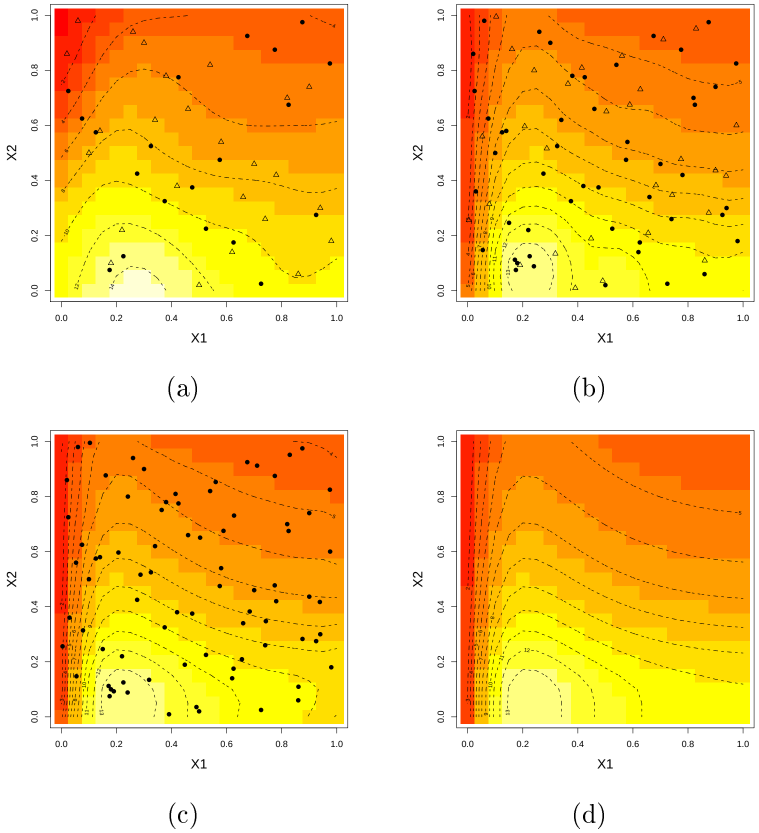
Figure 3: Graphical diagnostics for the toy example after new training data: individual standardized errors $D^I(\mathbf{y}^*)$ against (a) the validation data order, and (b) the emulator predictions; (c) pivoted Cholesky errors $D^{PC}(\mathbf{y}^*)$ against the pivoting order; (d) quantile-quantile plot of pivoted Cholesky errors.

has 5 inputs, the solar zenith angle, the leaf area index, relative leaf size, the Markov clumping parameter and one parameter called λ. For more details of this model, and for the real meanings of these inputs and the outputs, see Nilson and Kuusk (1989) and Kuusk (1996). For our analysis, the inputs were rescaled to make all the input values lie between 0 and 1, and the inputs are referenced as input 1 to 5.

The training data and the validation data contain 150 and 100 points, respectively, and are supplied as example data with the GEM-SA software (*http://ctcd.group.shef.ac.uk/gem.html*). The training and the validation data were selected using independent Latin Hypercube designs. The estimated correlation length parameters are $\hat{\psi} = (0.4607, 1.1183, 2.7996, 2.2590, 0.1470)$. We see that the correlation dies out fastest for input 5, indicating that the model output responds most strongly (and most non-linearly) to this input. Input 3, in contrast, has a high correlation length, suggesting that the output responds very smoothly to changes in this input. The estimated standard deviation is $\widehat{\sigma}^2 = 0.0068$.

The Mahalanobis distance and credible interval diagnostics are presented in Table 3. Both diagnostics point to a major discrepancy between the emulator and the simulator. They also suggest that the variability of the process is greater than was estimated, or the stationarity assumption is too strong.

Figure 5 (a) presents the individual standardized errors against the order of the validation data. Many



(a)

(b)

(c)

(d)

Figure 4: Predictive mean of the Gaussian Process Emulator built with (a) the original training data; (b) the updated training data; (c) all observations as the training data; (d) The two dimensional toy model evaluated over the input space. Training data (●) and validation data (△).

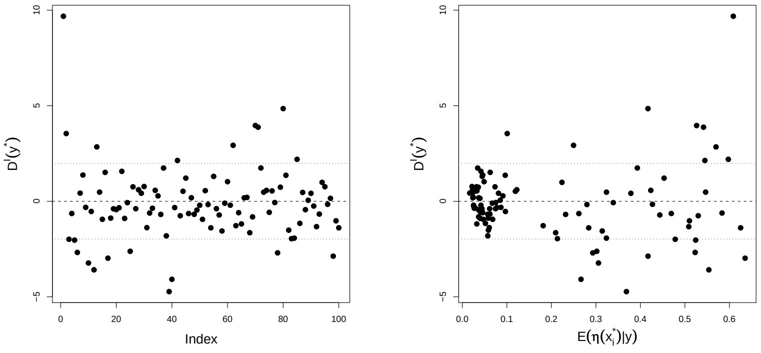
large errors can be observed, but there is no particular pattern. The only very large error suggests a local fitting problem around that validation point, while the remaining large errors indicate either an under-estimation of the variability, or a non-stationarity problem. Figure 5 (b) plots the individual standardized errors against the emulator predictions given by the expected value. The variability of the errors for small values of the predictions is smaller than the error variability for the large values of the predictions, indicating heteroscedasticity.

In order to find whether there is a particular subspace in the input space where the behaviour is different, the individual errors are plotted against each input, Figure 6. There is no clear systematic pattern for the inputs 1 to 4. However, the variability of the errors seems to depend on whether the input 5 is greater than 0.5, or 700 on the original scale. The last panel in Figure 6 plots the model output against input 5 for the combined training and validation data, and shows a clear nonlinearity and change of behaviour in the model for values of input 5 above 700.

The pivoted Cholesky errors are presented in Figure 7 (a). Large errors at the end of the plot indicates a possible over-estimation of the correlation length parameters, although the suggested non-stationarity of the model may be causing these large conditional errors. The QQ-plot of the pivoted errors, Figure 7 (b), indicates that the observed variability is bigger than the estimated, with many large values supporting the

Table 3: The observed Mahalanobis distance and credible interval diagnostics and some summaries of their predictive distributions, Nilson-Kuusk model.

	Obs.	Expected	Std. dev.	1stQ	Median	3rdQ
$D_{MD}(\cdot)$	750.237	100.000	18.593	87.288	98.813	111.964
$D_{CI}(\cdot)$	0.80	0.95	0.0275	0.93	0.95	0.97



(a)

(b)

Figure 5: Graphical diagnostics for the Nilson-Kuusk model using the individual standardized errors: (a) $D^I(\mathbf{y}^*)$ against the validation data order; (b) $D^I(\mathbf{y}^*)$ against the emulator predictions.

suggested non-stationarity problem.

According to the diagnostics, the emulator built with the training data is not a good and valid representation of the simulator. The diagnostics consistently point to the presence of non-stationarity and/or heteroscedasticity, with the model output for values of input 5 above 700 being shifted and more variable than when this input is below 700. Actions to improve the emulator might include adapting the mean function to the apparent shape of the response to input 5, allowing for a different variance when input 5 is above 700, or transforming the output variable to induce more homoscedasticity. Such changes, together with rebuilding the emulator using all 250 data points, should improve the fit substantially, but this should be checked against new validation data. However, we have no access to the simulator code, so such an analysis is not feasible

Instead, 50 validation data points randomly chosen were added to the training data. The points were randomly chosen because if we run the emulator adding extra points in the area where the input space is between 650 and 800, it would have very few data points in that region for validation. The other 50 validation data points were used for the diagnostics. Attempting to improve the emulator, we change the mean function allowing a 4th order polynomial for the input 5, i.e. $h(x)^T = (1, \mathbf{x}^T, x_5^2, x_5^3, x_5^4)$. And for stabilizing the variability over the input space, a log transformation of the output was used. Alternatively, more complex analyses could be used since the diagnostics pointed the presence of non-stationarity in the input space. For instance, for non-stationary model Treed Gaussian process can be used (Gramancy and Lee 2008). Another approach would be using a linear spline with knots at 700 and 775 or 800 for the input 5 as the mean function instead of a fourth order polynomial, but for this example we have opted for the polynomial mean function because of its simplicity.

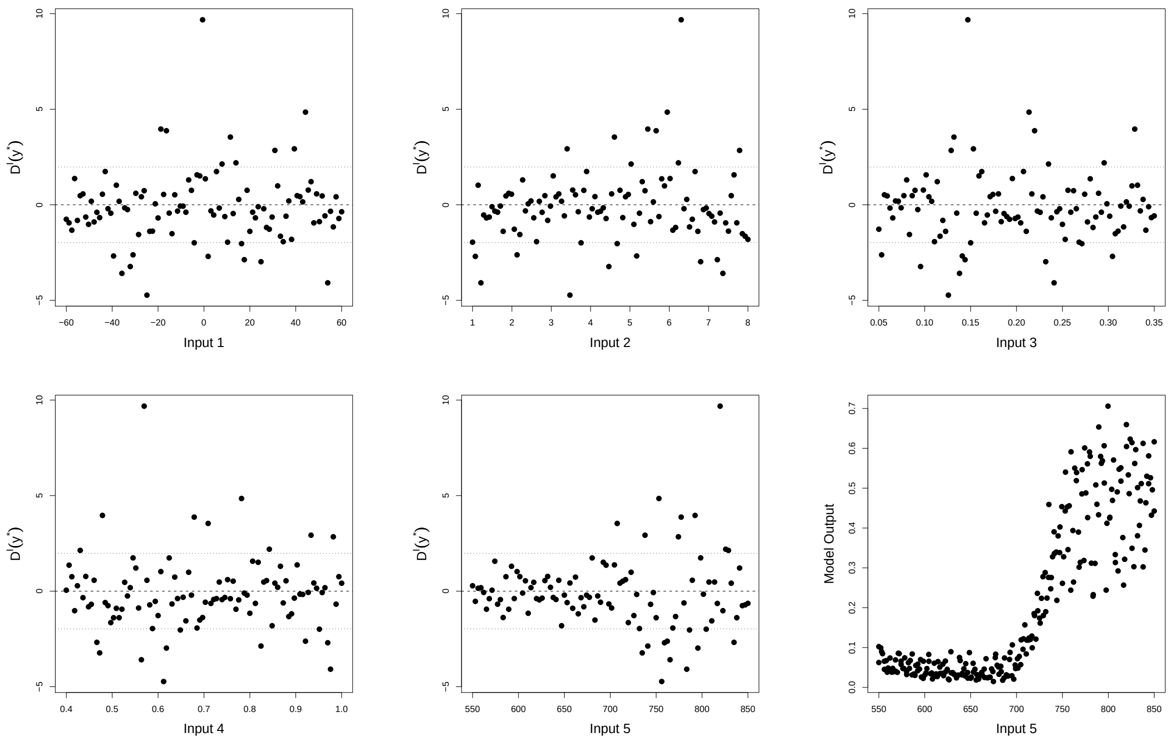


Figure 6: Individual standardized errors for the Nilson-Kuusk model, $D^I(\mathbf{y}^*)$, against the five input variables. Also the combined training and validation model outputs against input 5.

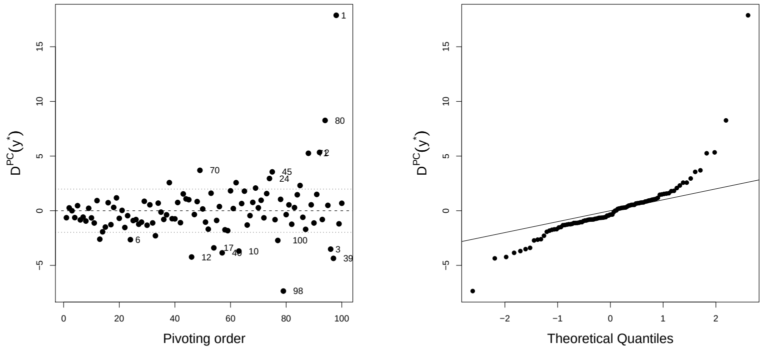
The Mahalanobis distance and credible interval diagnostics are presented in Table 4. Both diagnostics suggest no conflict between the rebuilt emulator and the Nilson-Kuusk model.

Table 4: The observed Mahalanobis distance and credible interval diagnostics and some summaries of their predictive distributions of the updated emulator for the Nilson-Kuusk model.

	Obs.	Expected	Std. dev.	1stQ	Median	3rdQ
$D_{MD}(\cdot)$	63.873	50.000	11.305	43.007	49.968	58.320
$D_{CI}(\cdot)$	0.94	0.951	0.032	0.92	0.96	0.98

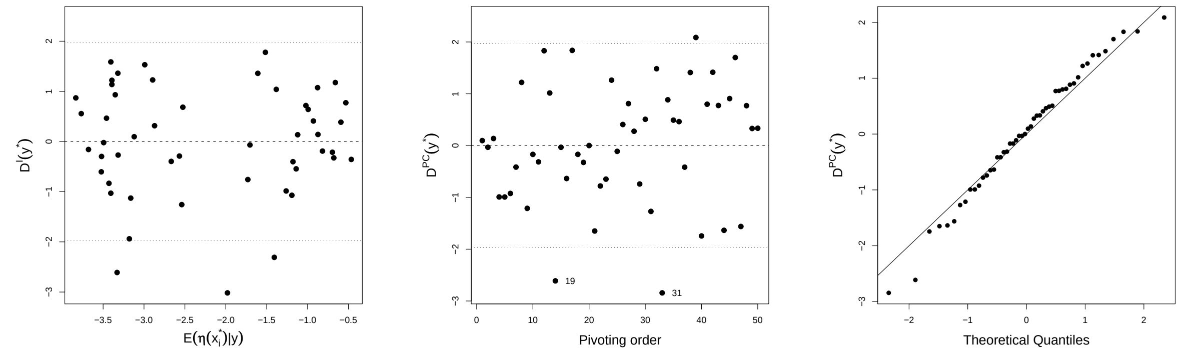
Figure 8 shows as expected that the rebuilt emulator can be a surrogate for the Nilson-Kuusk model. Figure 8 (a) presents the Individual standardized errors against the emulator predictions, and Figure 8 (b) presents the pivoted Cholesky errors against the pivoting order. Both figures indicate no obvious pattern. Figure 8 (c) presents the QQ-plot of the pivoted Cholesky errors. As the points lie close to the 45-degree line, the normality assumption for the log transformed simulator outputs appears to be reasonable.

We have seen in this example that the natural response to the diagnostics in the original emulator fit would have been to include all the validation data in the training sample, to add extra validation points with values of input 5 in the neighbourhood of 700, and then to validate the new emulator with extra validation points. This was not practical because we did not have access to the simulator to make more runs, but we were instead able to produce a valid emulator with only 50 extra training data, randomly selected from the



(a) (b)

Figure 7: Graphical diagnostics for the Nilson-Kuusk model using the the pivoted Cholesky errors: (a) $D^{PC}(\mathbf{y}^*)$ against the pivoting order; (b) Quantile-quantile plot.



(a) (b) (c)

Figure 8: Graphical diagnostics of the updated emulator for the Nilson Kuusk model: (a) Individual standardized errors, $D^I(\mathbf{y}^*)$, against the emulator predictions; (b) Pivoted Cholesky errors, $D^{PC}(\mathbf{y}^*)$, against the pivoting order; (c) Quantile-quantile plot of $D^{PC}(\mathbf{y}^*)$.

original validation sample. This result is encouraging because additional simulator runs can often be costly to obtain.

5 Concluding remarks

In this paper, we have presented a set of diagnostics attempting to validate Gaussian process emulators. We believe that this is a very important step before using the emulator as surrogate for the computer model, because a non-valid emulator can induce wrong conclusions. Our diagnostics focus on comparing emulator outputs with new runs of the computer model, referred to here as validation data, in a way that takes account of the uncertainties and correlations in the Gaussian process emulator predictions.

We proposed two kinds of diagnostics, numerical and graphical. The numerical diagnostics are functions of the validation data outputs, where we compare the observed value of each diagnostic with its induced distribution by the predictive distribution of the emulator outputs. The graphical diagnostics are basically visualisations of the prediction errors, where we consider the individual standardized errors and the uncorrelated standardized errors. The diagnostics are able to indicate whether the emulator and its uncertainty can represent the simulator. And where the emulator fails, the diagnostics can give information about where might be the problem.

However, the interpretations we have suggested for the various diagnostics should be used with care, since in a complex system like a Gaussian process emulator all the elements will interact in determining the diagnostic values. Better understanding of how to read different combinations of diagnostics will come with more experience of their use.

The diagnostics are equally effective when the training dataset is large. In this case, the mean function (11) will typically reproduce the simulator almost perfectly even for a non-stationary simulator, and the predictive variances will be very small throughout the region of input space covered by the training data. Even though the validation data may appear to follow the emulator mean function very closely, the diagnostics may nevertheless find that they are not close enough relative to the small predictive variances, so that standardised errors are not acceptable.

In the case of a large training dataset, computational problems are often encountered due to near-singularity of the predictive variance matrix. The pivoted Cholesky decomposition is then particularly valuable because the ill-conditioned nature of the matrix will emerge with pivoting variances eventually becoming negative due to rounding errors. Such components should be ignored, and the diagnostics based only on the uncorrelated errors produced up to that point.

In practice, "all models are wrong" and no emulator will represent its simulator with perfect validity. The emulator will still be useful, and 'good enough', if any remaining conflicts are small. In particular, an emulator built on a large training dataset (and hence with uniformly small predictive variances) may be deemed adequate even if there are substantive validation failures. It would be useful to have some measures for whether the emulator is 'good enough', but they are likely to depend on the uses to which the emulator will be put.

We have focused on the case in which the simulator gives a single output. It would be relevant to extend the diagnostic tools for multiple output emulators and for dynamic emulators.

We have also focused on validating the emulator as a representation of the simulator, whereas conventional verification and validation methods concern whether the simulator is an adequate representation of reality. Thus, further relevant extensions to these methods would address comparisons between the simulator and observational data, and between the emulator predictions and the observations. Furthermore, in the calibration approach of (Kennedy and O'Hagan 2001) the emulator is 'corrected' to allow for discrepancy between the simulator and reality, and this introduces yet more relevant comparisons for validation testing. It also makes it clear that the assumption in traditional V&V methods that discrepancies between simulator and reality are due simply to independent observation errors is naive. Note that in all of these cases we will generally need to work with relatively few real world observations, and hence with small validation samples.

Acknowledgements

This research is part of the “Managing Uncertainty in Complex Models” (MUCM) project, funded by Research Councils UK under its Basic Technology programme. The authors have benefited from several valuable discussions with other members of the MUCM team. We are also grateful for the helpful comments and suggestions of the editor and referees.

References

Balci, O. and Sargent, R. G. (1984), “A bibliography on the credibility, assessment and validation of simulation and mathematical models,” *Simuletter*, 15, 15–27.

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

Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D. (1988), “A Bayesian Approach to the Design and Analysis of Computer Experiments,” Tech. Rep. ORNL-6498, Oak Ridge National Laboratory.

— (1991), “Bayesian Prediction of Deterministic Functions, with Applications to the Design and Analysis of Computer Experiments,” *Journal of the American Statistical Association*, 86, 953–963.

Fraccaro, R., Hyndman, R. J., and Veevers, A. (2000), “Residual diagnostic plots for checking for model mis-specification in time series regression,” *Australia and New Zealand Journal of Statistics*, 42, 463–477.

Goldstein, M. and Rougier, J. (2006), “Bayes Linear Calibrated Prediction for Complex Systems,” *Journal of the American Statistical Association*, 101, 1132–1143.

Golub, G. H. and van Loan, C. F. (1996), *Matrix computations*, Johns Hopkins University Press, 3rd ed.

Gramancy, R. B. and Lee, H. K. H. (2008), “Bayesian Treed Gaussian Process Models With an Application to Computer Modeling,” *Journal of the American Statistical Association*, 103, 1119–1130.

Haslett, J. and Hayes, K. (1998), “Residuals for the linear model with general covariance structure,” *Journal of the Royal Statistical Society B*, 60, 201–215.

Higham, N. J. (2002), *Accuracy and Stability of Numerical Algorithms*, Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, second Edition. First edition 1996.

Hills, R. G. and Trucano, T. G. (1999), “Statistical Validation of Engineering and Scientific Models: Background,” Tech. Rep. SAND99-1256, Sandia National Laboratory.

— (2001), “Statistical Validation of Engineering and Scientific Models: A Maximum Likelihood Based Metric,” Tech. Rep. SAND2001-1783, Sandia National Laboratory.

Houseman, E. A., Ryan, L. M., and Coull, B. A. (2004), “Cholesky Residuals for Assessing Normal Errors in a Linear Model With Correlated Outcomes,” *Journal of the American Statistical Association*, 99, 383–394.

Kennedy, M. C., Anderson, C. W., Conti, S., and O’Hagan, A. (2006), “Case studies in Gaussian process modelling of computer codes,” *Reliability Engineering & System Safety*, 91, 1301–1309.

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

Kimeldorf, G. S. and Wahba, G. (1970), "A correspondence between Bayesian estimation on stochastic processes and smoothing by splines," *The Annals of Mathematical Statistics*, 41, 495–502.

Kleijnen, J. P. C. (1995), "Verification and Validation of simulation models," *European Journal of Operational Reasearch*, 82, 145–162.

Kuusk, A. (1996), "A computer-efficient plant canopy reflectance model," *Computers and Geosciences*, 22, 149–163.

McKay, M. D., Beckman, R. J., and Conover, W. J. (1979), "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, 21, 239–245.

Nilson, T. and Kuusk, A. (1989), "A reflectance model for the homogeneous plant canopy and its inversion," *Remote Sensing of Environment*, 27, 157–167.

Oakley, J. E. and O'Hagan, A. (2002), "Bayesian inference for the uncertainty distribution of computer model outputs," *Biometrika*, 89, 769–784.

— (2004), "Probabilistic sensitivity analysis of complex models: a Bayesian approach," *Journal of the Royal Statistical Society B*, 66, 751–769.

Oberkampf, W. and Trucano, T. (2000), "Validation Methodology in computational fluid dynamics," Tech. Rep. 2000-2549, American Institute of Aeronautics and Astronautics.

O'Hagan, A. (1978), "Curve fitting and optimal design for predictions," *Journal of the Royal Statistical Society B*, 40, 1–42.

— (2006), "Bayesian analysis of computer code outputs: A tutorial," *Reliability Engineering & System Safety*, 91, 1290–1300.

Roache, P. J. (1998), *Verification and Validation in computer science and engineering*, Albuquerque: Hermosa.

Rougier, J., Sexton, D. M. H., Murphy, J. M., and Stainforth, D. (2007), "Emulating the sensitivity of the HadAM3 climate model using ensembles from different but related experiments," Tech. Rep. 07/04, MUCM, The University of Sheffield, being revised for the Journal of Climate.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989), "Design and Analysis of Computer Experiments," *Statistical Science*, 4, 409–423.

Saltelli, A., Chan, K., and Scott, M. (eds.) (2000), *Sensitivity Analysis*, New York: Wiley.

Santner, T. J., Williams, B. J., and Notz, W. I. (2003), *The Design and Analysis of Computer Experiments*, Springer-Verlag.

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.