Emulation of multivariate simulators using thin-plate splines with application to atmospheric dispersion

Veronica E. Bowman * David C. Woods †

October 26, 2016

Abstract

It is often desirable to build a statistical emulator of a complex computer simulator in order to perform analysis which would otherwise be computationally infeasible. We propose methodology to model multivariate output from a computer simulator taking into account output structure in the responses. The utility of this approach is demonstrated by applying it to a chemical and biological hazard prediction model. Predicting the hazard area which results from an accidental or deliberate chemical or biological release is imperative in civil and military planning and also in emergency response. The hazard area resulting from such a release is highly structured in space and we therefore propose the use of a thin-plate spline to capture the spatial structure and fit a Gaussian process emulator to the coefficients of the resultant basis functions. We compare and contrast four different techniques for emulating multivariate output: dimension-reduction using (i) a fully Bayesian approach with a principal component basis, (ii) a fully Bayesian approach with a thin-plate spline basis, assuming that the basis coefficients are independent, and (iii) a "plug-in" Bayesian approach with a thin-plate spline basis and a separable covariance structure; and (iv) a functional data modeling approach using a tensor-product (separable) Gaussian process. We develop methodology for the two thin-plate spline emulators and demonstrate that these emulators significantly outperform the principal component emulator. Further, the separable thin-plate spline emulator, which accounts for the dependence between basis coefficients, provides substantially more realistic quantification of uncertainty, and is also computationally more tractable, allowing fast emulation. For high resolution output data, it also offers substantial predictive and computational advantages over the tensor-product Gaussian process emulator.

This paper will appear in the Journal of Uncertainty Quantification.

1 Introduction

The simulation of scientific and engineering systems via complex mathematical models has become a common method of gaining knowledge about processes where physical experimentation is infeasible or unaffordable. Encapsulated in computer codes or simulators, many of these models require substantial computing time to evaluate the response for a given set of inputs. For even moderately expensive simulators, the computational resources required to perform, for example, Monte Carlo inference may be prohibitive in practice. Hence, building an emulator or surrogate for the computer model trained on a, usually small, set of simulator evaluations has become standard practice; see for example, the seminal paper of Sacks et al. [29], Kennedy et al. [19], who presented a number of case studies of such computer experiments, and the book-length treatments of Santner et al. [30], Fang et al. [9] and Forrester et al. [10]. Computationally cheap emulators allow for real-time decision making and greater scientific understanding through, for example, sensitivity and uncertainty analyses.

*Defence Science and Technology Laboratory, Salisbury, SP4 0JQ, UK (vbowman@mail.dstl.gov.uk).

†Corresponding author. Southampton Statistical Sciences Research Institute, University of Southampton, Southampton, SO17 1BJ, UK (d.woods@southampton.ac.uk).

Statistical modelling is a common method for constructing emulators. Essentially, simulator output is treated as a realisation of a stochastic process, and regression models are fitted to the data in order to approximate the relationship between the simulator inputs and the outputs. The most common emulator is the Gaussian process (e.g. [25]), a smooth non-parametric interpolator. An emulator based on a statistical model allows for prediction of the simulator at untested inputs and quantification of the associated uncertainty. For deterministic simulators, as considered in this paper, this uncertainty is a result of incomplete knowledge of the simulator across the whole input space and approximation error from the regression model.

Modern applications increasingly involve highly multivariate simulators, with each run of the simulator producing data from a curve, surface or other high-dimensional output structure. The standard approach is to perform dimension reduction on the multivariate output using a set of appropriate basis functions and then use scalar emulation methods, such as the Gaussian process, on the basis coefficients. We delay our discussion of the related statistical literature to Section 2.

Motivated by a simulator of chemical and biological dispersion, the contribution of this paper is to propose the use of thin-plate splines as basis functions for multivariate data with spatial structure, and to develop the necessary methodology for their application. For a two-dimensional output, a thin-plate spline basis provides a spatial mapping using the proximity of data to a set of knots (see Section 3). Splines have also recently been employed for computer experiment modelling [6] as an alternative to Gaussian process models. We implement both a fully Bayesian thin-plate spline emulator using Markov chain Monte Carlo and also a "plug-in" emulator (e.g. [20]) with a separable covariance structure [27] and correlation parameters estimated using a validation data set. We provide a detailed comparison of these competing methodologies with the application of a functional data model using a Gaussian process defined on both the input and output domains.

Dispersion simulators have widespread application in environmental monitoring, civilian emergency planning and military applications, for example in the protection against terrorism threats. Available simulators range from quite simple Gaussian plume models (e.g. [7]), through Gaussian puff models (e.g. [32]) to computationally expensive Lagrangian models (e.g. [18]).

In this paper, the problem of emulating a multivariate simulator built from a Gaussian puff model is considered. This model accounts for both the effects of an urban environment and the underlying terrain features. For each run of the simulator, the response of interest is the dosage, defined as the integrated concentration over time, measured at a large number of points in a two-dimensional geographical domain. Inputs to the simulator include meteorological variables (such as wind speed and direction) and variables related to the source of the dispersion (such as location and size of release). While relatively fast (~ 1 minute per run for a complete spatial output grid) and simple to compute, the dispersion model is employed in the optimization of sensor placements. Therefore computationally inexpensive surrogates are required that can provide accurate high-resolution prediction on the spatial output grid.

Previously, a number of authors have considered the univariate problem of *calibrating* (relatively simple) dispersion models using actual dispersion data obtained under a single, but uncertain, set of meteorological and source conditions; see, for example, [31], [20], [23] and [26]. Typically, the aim of such work is to solve the inverse problem of identifying unknown features of the source.

The remainder of this paper is organised as follows. In Section 2, we introduce and describe the multivariate emulator, dimension reduction techniques for multivariate outputs and separable covariance structures. The necessary methods for thin-plate spline emulation are developed in Section 3, and applied to an illustrative dispersion model in Section 4. In that section, we also compare our methodology to applying dimension reduction via principal components and to a tensor-product, separable Gaussian process model. We explore the effectiveness of the methodology for high-resolution prediction. In Section 5, we compare dimension-reduction and functional data modelling on an artificial environmental example. We conclude in Section 6 with some discussion and areas for future research. The appendices provide mathematical and computational details of the methods.

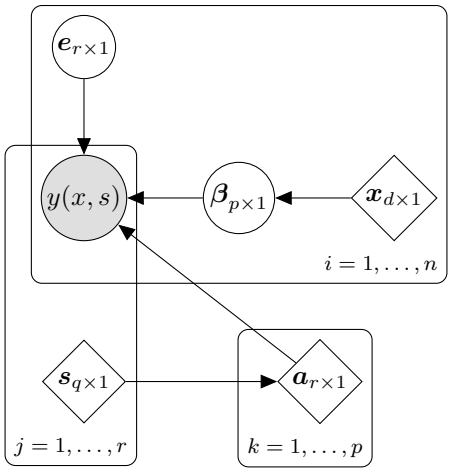


Figure 1: Representation of model (1) as a directed graph with edges representing dependencies, the shaded node representing the observable output, unshaded circular nodes representing unobservable features of the model, and diamond nodes representing deterministic model inputs. Subscripts indicate size of vectors and plates indicate the sizes of data sets.

2 Multivariate emulation

Let $\boldsymbol{x}_i = (x_{1i}, \ldots, x_{di})^{\mathrm{T}} \in \mathcal{X} \subseteq \mathbb{R}^d$ be the vector of input values at which the ith run of the simulator is performed, i.e. the ith input point ($i = 1, \ldots, n$), and let $\boldsymbol{Y}(\boldsymbol{x}_i) = [Y(\boldsymbol{x}_i, \boldsymbol{s}_1), \ldots, Y(\boldsymbol{x}_i, \boldsymbol{s}_r)]^{\mathrm{T}}$ be the vectorised output from this run. The vector $\boldsymbol{s}_j = (s_{1j}, \ldots, s_{qj})^{\mathrm{T}} \in \mathcal{S}$ locates the jth output in the q dimensional output domain $\mathcal{S} \subseteq \mathbb{R}^q$ ($j = 1, \ldots, r$). For example, for a two-dimensional simulator, $q = 2$ and $\boldsymbol{s}_j = (s_{1j}, s_{2j})^{\mathrm{T}}$ which may be the geographical coordinates of the response. We denote the complete output grid as an $r \times q$ matrix $S = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_r)^{\mathrm{T}}$.

We achieve dimension reduction through the assumption of a linear model for the response:

$$\boldsymbol{Y}(\boldsymbol{x}_i) = \sum_{k=1}^{p} \boldsymbol{a}_k(S)\beta_k(\boldsymbol{x}_i) + \boldsymbol{e}_i\,. \tag{1}$$

In (1), $\boldsymbol{a}_1(S), \ldots, \boldsymbol{a}_p(S)$ are a set of $r \times 1$ basis vectors which are assumed independent of $\boldsymbol{x}$ but which may depend on the output grid S, the coefficients $\boldsymbol{\beta}(\boldsymbol{x}) = [\beta_1(\boldsymbol{x}), \ldots, \beta_p(\boldsymbol{x})]^{\mathrm{T}}$ are functions of the input variables $\boldsymbol{x}$ and $\boldsymbol{e}_i$ is a r-vector of errors resulting from the basis function approximation. Figure 1 summarizes this hierarchical model and demonstrates the dependence of $\boldsymbol{\beta}(\boldsymbol{x}_1), \ldots, \boldsymbol{\beta}(\boldsymbol{x}_n)$ on $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, and $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_p$ on $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_r$.

In our motivating application, interest is only in prediction at the $\boldsymbol{s}_j$, and hence for each output location we mean-center and standardize the data $[Y(\boldsymbol{x}_1, \boldsymbol{s}_j), \ldots, Y(\boldsymbol{x}_n, \boldsymbol{s}_j)]$ to have variance equal to one. We complete the hierarchical specification of the model through choice of prior distributions. For $\beta_k(\boldsymbol{x})$, we assume the impact of the input variables is modeled using a Gaussian process

$$\beta_k(\boldsymbol{x})|\tau_k, \boldsymbol{\theta}_k \sim \mathrm{GP}\left(0, \tau_k\rho(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}_k)\right) \tag{2}$$

where GP denotes a Gaussian process and $\rho(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$ is a correlation function dependent on input vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$. Scale and correlation length hyper-parameters are labeled τ_k and $\boldsymbol{\theta}_k = (\theta_{1k}, \ldots, \theta_{dk})^{\mathrm{T}}$ respectively. We discuss the choice of joint distribution of $\beta_k(\boldsymbol{x})$ and $\beta_{k'}(\boldsymbol{x}')$ below. For $\boldsymbol{e}_i$ we assume

$$\boldsymbol{e}_i|\sigma^2 \sim N(\boldsymbol{0}_r, I_r\sigma^2)\,, \tag{3}$$

with I_r the $r \times r$ identity matrix and $\sigma^2 > 0$. The choice of a independent multivariate prior distribution for $\boldsymbol{e}_i$ assumes that the regression function $\sum_{k=1}^{p} \boldsymbol{a}_k(S)\beta_k(\boldsymbol{x}_i)$ is detailed enough to capture the vast majority of the variation in the data. The adequacy of this assumption for a particular application can be checked via standard residual diagnostics.

To specify the joint distribution of $\beta_k(\boldsymbol{x})$ and $\beta_{k'}(\boldsymbol{x}')$, $k, k' = 1, \ldots, p$, we consider two simplifying cases.

(i) Independence of $\beta_k(\boldsymbol{x})$ and $\beta_{k'}(\boldsymbol{x}')$ for $k \neq k'$, that is

$$\text{Cov}\left\{\beta_k(\boldsymbol{x}), \beta_{k'}(\boldsymbol{x}')\right\} = \tau_k \rho(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}_k)\mathbb{I}(k = k')\,,$$

with $\mathbb{I}$ the indicator function. Let $\boldsymbol{\beta}_k = (\beta_k(\boldsymbol{x}_1), \ldots, \beta_k(\boldsymbol{x}_n))^{\mathrm{T}}$ and $\boldsymbol{\beta} = (\boldsymbol{\beta}_1^{\mathrm{T}}, \ldots, \boldsymbol{\beta}_p^{\mathrm{T}})^{\mathrm{T}}$. Also, let $\tau_k W_k$ denote the $n \times n$ variance-covariance matrix of $\boldsymbol{\beta}_k$, with ijth entry $\tau_k \rho(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\theta}_k)$. Then, the block diagonal variance-covariance matrix of $\boldsymbol{\beta}$ is given by

$$\text{Cov}\left\{\boldsymbol{\beta}\right\} = \text{diag}\left(\tau_1 W_1, \ldots, \tau_p W_p\right)\,. \tag{4}$$

(ii) A product covariance structure, with

$$\text{Cov}\left\{\beta_k(\boldsymbol{x}), \beta_{k'}(\boldsymbol{x}')\right\} = \tau \rho(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}) \times \phi(\boldsymbol{s}, \boldsymbol{s}'; \boldsymbol{\nu})\,,$$

with $\phi(\boldsymbol{s}, \boldsymbol{s}'; \boldsymbol{\nu})$ a function depending on output locations $\boldsymbol{s}$ and $\boldsymbol{s}'$ and parameters $\boldsymbol{\nu} = (\nu_1, \ldots . \nu_q)^{\mathrm{T}}$. Now, the variance-covariance matrix of $\boldsymbol{\beta}$ has the form

$$\text{Var-Cov}\left\{\boldsymbol{\beta}\right\} = \tau W \otimes V\,, \tag{5}$$

with τW being the $n \times n$ common variance-covariance matrix for $\boldsymbol{\beta}_k$ with ijth entry $\tau \rho(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\theta})$ and V a $p \times p$ scale matrix for $\boldsymbol{\beta}(\boldsymbol{x}_i) = \left[\boldsymbol{\beta}_1(\boldsymbol{x}_i), \ldots, \boldsymbol{\beta}_p(\boldsymbol{x}_i)\right]^{\mathrm{T}}$ with entries defined via ϕ. The exact form of ϕ and construction of V depends on the basis chosen; we discuss the choice of V for our thin-plate spline emulators in Section 3.

Point prediction of the response at a untried input point, $\boldsymbol{x}^\star$, is via

$$\hat{\boldsymbol{Y}}(\boldsymbol{x}^\star) = \sum_{k=1}^{p} \boldsymbol{a}_k(S)\hat{\beta}_k(\boldsymbol{x}^\star)\,, \tag{6}$$

where $\hat{\beta}_k(\boldsymbol{x}^\star)$ is an appropriate summary of the posterior distribution of $\beta_k(\boldsymbol{x}^\star)$. There is no conjugate prior distribution available for unknown $\boldsymbol{\theta}$ and so to obtain the full marginal distribution, numerical methods must be used, such as Markov chain Monte Carlo (MCMC; [22], ch. 10) with a sample from the posterior predictive distribution obtained by plugging samples from $\beta_k(\boldsymbol{x}^\star)|Y$ into (6). Alternatively, a maximum a posteriori (MAP), or other estimator, of $\boldsymbol{\theta}$ can be substituted into the conditional posterior distribution of $\beta_k(\boldsymbol{x}^*)$.

Similar models have been developed and applied by a variety of authors. Campbell et al. [5] considered a variety of basis functions for one-dimensional functional responses, including orthogonal polynomials and data-adaptive choices such as principal components and partial least squares. A wavelet basis was used by Bayarri et al. [1] in a calibration problem with a functional response. Higdon et al. [16] used a principal components basis for an example of cylinder deformation from the Manhattan project. These latter authors had a two-dimensional output grid, consisting of angles and times, and were again concerned with model calibration.

Principal component analysis (PCA; [17]) is a dimension-reduction technique that has been commonly used in the literature for defining a new, orthogonal basis for a set of multivariate data. For a $r \times n$ matrix $Y = [\boldsymbol{Y}(\boldsymbol{x}_1), \ldots, \boldsymbol{Y}(\boldsymbol{x}_n)]$, the principal components are defined through the singular value decomposition of $Y = \Lambda \Sigma \Omega^{\mathrm{T}}$, where Λ is a $r \times n$ orthogonal matrix whose columns are the left singular vectors of Y, Σ is a $n \times n$ diagonal matrix holding the singular values of Y and Ω is an $n \times n$ orthonormal matrix whose columns are the

right singular vectors. A p-dimensional principal component basis $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_p$ is given by the first p columns of $n^{-1/2}\Lambda\Sigma$, with corresponding weights $\beta_k(\boldsymbol{x}_i)$ given by entry (k, i) in $n^{1/2}\Omega$ ($k = 1, \ldots, p$; $i = 1, \ldots, n$). The basis functions are orthogonal and have the property of defining subspaces with the largest variance, see [15], ch.3. The data-dependent nature of the principal components provides a flexible non-parametric modelling approach. However, it does not take the structure of the output grid S into account.

3 Thin-plate regression splines

To provide a set of flexible and data-driven basis vectors $\boldsymbol{a}_1(S), \ldots, \boldsymbol{a}_p(S)$ that maintain the multidimensional spatial structure inherent in S, we use thin-plate regression splines (TPRS) [33]. To define the TPRS basis, we start with the $r \times r$ matrix E having uvth entry $\eta_{lq}(||\boldsymbol{s}_u - \boldsymbol{s}_v||)$, with $||\cdot||$ defined as Euclidean distance and

$$\eta_{lq}(t) = \begin{cases} \frac{(-1)^{l+1+q/2}}{2^{2l-1}\pi^{q/2}(l-1)!(l-q/2)!} t^{2l-q}\log(t) & \text{for } q \text{ even}, \\ \\ \frac{\Gamma(q/2-l)}{2^{2l}\pi^{q/2}(l-1)!} t^{2l-q} & \text{for } q \text{ odd}, \end{cases}$$

with $2l > d$ controlling the smoothness of the spline ($l = 2$ corresponds to a smoothness penalty in terms of second derivatives of the response function, and is adopted in this paper). A thin-plate spline for the ith simulator run is then the solution to

$$\text{minimize} \qquad ||\boldsymbol{Y}(\boldsymbol{x}_i) - E\boldsymbol{\delta}_i - T\boldsymbol{\alpha}_i||^2 + \lambda_i \boldsymbol{\delta}_i^{\mathrm{T}} E \boldsymbol{\delta}_i \qquad \text{subject to } T^{\mathrm{T}}\boldsymbol{\delta}_i = \mathbf{0}, \tag{7}$$

with respect to $\boldsymbol{\delta}_i$ and $\boldsymbol{\alpha}_i$ for $\lambda_i \geq 0$ ($i = 1, \ldots, n$). The matrix T holds basis vectors corresponding to orthogonal polynomials in $\mathbb{R}^q$ of degree less than l. In our applications with $l = 2$ and a regular lattice output grid, T is an $r \times 3$ matrix with jth row $[1 \mid \boldsymbol{s}_j^{\mathrm{T}}]$. The thin-plate spline is therefore the solution to a penalised least squares problem.

To avoid problems of choosing "knot locations" (e.g. a subsample of the output grid) in selecting a regression basis, Wood [33] defined a TPRS basis as a rank m approximation to the spline, with basis vectors given by the columns of $U_m D_m Z_m$ and T, where D_m is an $m \times m$ diagonal matrix holding the m largest eigenvalues of E ordered by absolute value, U_m is an $r \times m$ matrix holding the corresponding eigenvectors and Z_m is an m dimensional orthogonal column basis such that $T^{\mathrm{T}} U_m Z_m = \mathbf{0}$. That is, an approximation to problem (7) with $\lambda_i = 0$ (i.e. unpenalised) is given by

$$\text{minimize} \qquad ||\boldsymbol{Y}(\boldsymbol{x}_i) - U_m D_m Z_m \delta_i' - T\boldsymbol{\alpha}_i||^2, \tag{8}$$

with respect to $\boldsymbol{\delta}_i'$ and $\boldsymbol{\alpha}_i$, where $\boldsymbol{\delta}_i' = Z_m^{\mathrm{T}} U_m^{\mathrm{T}} \boldsymbol{\delta}_i$ and condition $T^{\mathrm{T}} U_s Z_s = \mathbf{0}$ ensures $T^{\mathrm{T}}\boldsymbol{\delta}_i = \mathbf{0}$.

Hence, we set the first m basis vectors in (1), $\boldsymbol{a}_1(S), \ldots, \boldsymbol{a}_m(S)$, to be the columns of $U_m D_m Z_m$, and the second $p - m$ basis vectors, $\boldsymbol{a}_{m+1}(S), \ldots, \boldsymbol{a}_p(S)$, to be the columns of T. For $i = 1, \ldots, n$, we set

$$\boldsymbol{\beta}(\boldsymbol{x}_i) = \left[\boldsymbol{\beta}_m(\boldsymbol{x}_i)^{\mathrm{T}}, \boldsymbol{\beta}_{p-m}(\boldsymbol{x}_i)^{\mathrm{T}}\right]^{\mathrm{T}} = \left[\left(\boldsymbol{\delta}_i'^{\star}\right)^{\mathrm{T}}, \left(\boldsymbol{\alpha}^{\star}\right)^{\mathrm{T}}\right]^{\mathrm{T}},$$

the solution to (8). We require a constant scale matrix for $\boldsymbol{\beta}(\boldsymbol{x}_i)$ for all $\boldsymbol{x}_i$ in order to construct the separable variance-covariance matrix (5) for the complete parameter vector $\boldsymbol{\beta}$ and to allow prediction for untried input points. Hence, rather than derive the variance-covariance matrix for $\boldsymbol{\beta}(\boldsymbol{x}_i)$ conditional on $\boldsymbol{Y}(\boldsymbol{x}_i)$, we assume $\mathrm{Cov}\,(\boldsymbol{\delta}_i^{\star}) \propto Q$ where $\boldsymbol{\delta}_i^{\star} = U_m Z_m \boldsymbol{\delta}_i'^{\star}$ and Q has uvth entry given by $\rho(\boldsymbol{s}_u, \boldsymbol{s}_v; \boldsymbol{\nu})$ ($u, v = 1, \ldots, r$), a spatial correlation function defined on the output locations. It then follows that $\mathrm{Cov}\{\boldsymbol{\beta}_m(\boldsymbol{x}_i)\} \propto Z_m^{\mathrm{T}} U_m^{\mathrm{T}} Q U_m Z_m$. Further, we assume $\boldsymbol{\beta}_m(\boldsymbol{x}_i)$ and $\boldsymbol{\beta}_{p-m}(\boldsymbol{x}_i)$ are independent, with $\mathrm{Cov}\{\boldsymbol{\beta}_{p-m}(\boldsymbol{x}_i)\} \propto I_{p-m}$, leading to scale matrix

$$V = \begin{pmatrix} Z_m^{\mathrm{T}} U_m^{\mathrm{T}} Q U_m Z_m & \mathbf{0}_{m\times p} \\ \mathbf{0}_{m\times p}^{\mathrm{T}} & I_{p-m} \end{pmatrix}.$$

4 Application to the dispersion simulator

In this section, the methodology from Sections 2 and 3 is applied to an exemplar dispersion simulator. As described in Section 1, the response of interest is dosage (integrated concentration over time). The resultant dosage map is known as a hazard prediction and forms the basis for many modelling systems designed to mitigate hazard such as source-term estimators [26], sensor placement optimizers and training scenarios. The motivating system for this paper is a sensor placement tool which requires rapid hazard predictions over which the probability of detection of a chemical or biological release is numerically maximized via the placement of a set of sensors. For other applications, the methodology could be applied to emulate concentration at a given time or time could be included as an extra dimension to either the input or output (cf [8]).

In the application of interest, it is key that (i) a hazard prediction can be made rapidly, to assess performance of a sensor grid against a particular threat in real time, and (ii) there is accurate prediction of the dosage on a common high-resolution spatial output grid for untried values of the input variables. Our data comes from the Urban Dispersion Model (UDM), an example of a Gaussian puff model. In these simulators, continuous releases are modelled as a serious of instantaneous releases ("puffs") with Gaussian profiles in the "x" and "y" planes (forming a bell shape). The UDM can model dispersion across complex terrain such as urban areas by including the effect of buildings, in addition to underlying terrain (for example, hills or valleys), by tracking the interaction of individual puffs with local geographical features.

There are numerous inputs to a dispersion simulator, including those related to the release (e.g. [4]). For this application, the input space is limited to the $d = 2$ dimensions which, from previous experience, are known to have greatest effect on dosage: mass of the release ($0 \leq x_1 \leq 50$kg) and wind speed ($5 \leq x_2 \leq 30$m/s). For a given terrain, other variables known to affect dosage, such as wind direction and location of release, can be specified post-simulation through rotation and translation operations. Typical simulator responses exhibit a very rapid drop in dosage upwind of the release and a complex downwind gradual reduction in dosage. There is also different behavior in the down-wind ("x") and cross-wind ("y") directions. This behavior is not well described by standard exponential functions; see Figure 2.

Our data results from simulator runs on a "uniform" urban area representative of a medium-rise city environment outside of the central business district. Accurate high-resolution spatial prediction is important for both comparing competing sensor placements in the optimization tool and ensuring assessments of chosen sensor placements are realistic summaries of actual system performance. Errors of only a few tens of meters in sensor placement, perhaps caused by small prediction errors in the modeling, can result in large discrepancies in predicted dosage and therefore lead to substantial underperformance of a sensor system against a specific threat. Any emulator must therefore be capable of producing rapid predictions on a high-resolution spatial grid and even small improvements in predictive performance are practically valuable.

For the ith run, dosage is output on a $k \times k$ grid ($i = 1, \ldots, n$). Hence, $\boldsymbol{s}_j = (s_{1j}, s_{2j})^{\mathrm{T}}$ locates the jth (standardised) geographical position at which dosage is calculated ($j = 1, \ldots, r$). The output grid is common to all runs; if this were not the case, linear interpolation could be used to map the outputs from different simulator runs to a common output grid. The vector $\boldsymbol{Y}(\boldsymbol{x}_i)$ holds the $r = k^2$ dosages for the ith run; we in fact model $\log(\boldsymbol{Y}(\boldsymbol{x}_i) + 1)$. The simulator input values are held in $\boldsymbol{x}_i = (x_{1i}, x_{2i})^{\mathrm{T}}$ ($i = 1, \ldots, n$).

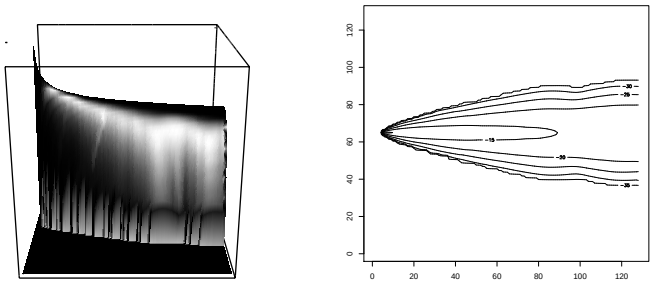
We apply and compare four emulation approaches:

1. A fully Bayesian approach using a principal components basis (PC-GP emulator).

2. A fully Bayesian approach using a TPRS basis and assuming independence of the elements of $\boldsymbol{\beta}(\boldsymbol{x}_i)$, cf (4) (iTPRS-GP emulator).

3. An empirical Bayes approach using a TPRS basis and assuming a separable covariance structure for $\boldsymbol{\beta}$, cf (5) (sTPRS-GP emulator).

4. A empirical Bayes approach using a Gaussian process with a separable covariance structure defined on $\mathcal{X} \times \mathcal{S}$ (sGP emulator).



(a) Perspective Plot (b) Contour Plot

Figure 2: A typical dosage output from the dispersion model output on a logged scale

For emulators 1 and 2, the posterior predictive distribution for $\boldsymbol{Y}$ is obtained using MCMC with convergence assessed graphically using diagnostic plots. For these emulators, we also assume $V = I_p$; that is, the elements of $\boldsymbol{\beta}(\boldsymbol{x}_i)$ are assumed independent. While for emulator 1 this assumption has a heuristic justification via adoption of an orthogonal PC basis, we acknowledge that for emulator 2 it may lead to overconfident prediction intervals and an over-estimate of the effective sample size. However, it substantially reduces the computational burden of the MCMC algorithm and provides a benchmark for emulators 3 and 4. Here, the correlation length parameters $\boldsymbol{\theta}$ and $\boldsymbol{\nu}$ are chosen using validation simulation runs and substituted into the conditional posterior predictive density (see also [27] and [28]).

Emulator 4 models the simulator output directly as a function of both $\boldsymbol{x}$ and $\boldsymbol{s}$, that is

$$Y(\boldsymbol{x}, \boldsymbol{s}) \mid \tau, \boldsymbol{\theta}, \boldsymbol{\nu} \sim \mathrm{GP}\left[0, \tau\rho(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}) \times \rho(\boldsymbol{s}, \boldsymbol{s}'; \boldsymbol{\nu})\right] .$$

The assumption of a separable correlation structure results in a tensor-product variance-covariance matrix for $\boldsymbol{Y} = [\boldsymbol{Y}(\boldsymbol{x}_1), \ldots, \boldsymbol{Y}(\boldsymbol{x}_n)]$. See [27] for a detailed discussion of this model, and extensions.

4.1 Choice of correlation function

In this paper, we use a squared exponential correlation function [16], with parameterization

$$\rho(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}) = \prod_{j=1}^{d} \theta_j^{4(x_j - x_j')^2} . \tag{9}$$

Assuming a standardized wind direction, which can be adjusted post-simulation, the main variation in the response will be axially aligned, making (9) a reasonable choice. A fully Bayesian approach requires a prior distribution for each θ_j and we assume common Beta prior densities with $\pi(\theta_k) \propto \theta_k^{a_\theta - 1}(1 - \theta_k)^{b_\theta - 1}$. A nugget term was added to improve the conditioning of the variance-covariance matrix of $\boldsymbol{\beta}$. Clearly, alternative correlation functions could be employed; we also tried the Matérn correlation function but found for this application it led to less accurate results and did not improve numerical stability of the modeling. The addition of a nugget is a pragmatic method for reducing the sensitivity of the modelling results to underlying assumptions, including the choice of covariance function [12].

4.2 Design of the simulator runs

The emulators were trained using $n = 80$ simulator runs generated as a maximin Latin Hypercube sample [21]. A further 75 simulator runs were generated randomly as a Monte Carlo sample, with 40 of these runs used for validation and choice of some prior hyper-parameters. The remaining 35 runs were used as test cases to assess emulator accuracy. Initially, all simulator runs were generated for a low-resolution $k = 8$

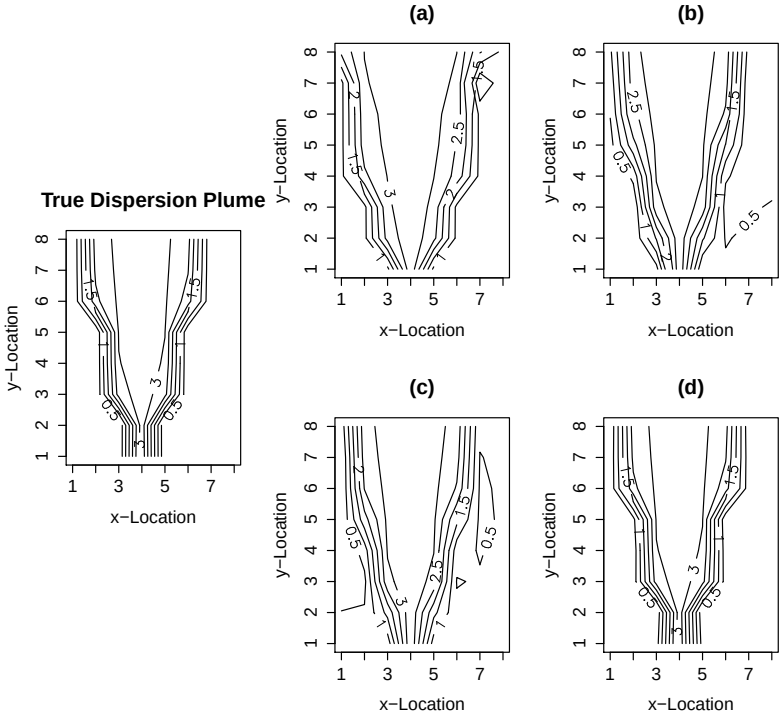


Figure 3: Actual and predicted log dosage for one test run for the (a) PC-GP, (b) iTPRS, (c) sTPRS and (d) sGP emulators.

regular lattice in $\mathcal{S}$ with no boundary points. We assess the more promising emulators on a high-resolution output grid in Section 4.6.

4.3 Model fitting and choice of hyper-parameters

Bayesian inference for the PC-GP and iTPRS-GP emulators proceeded using MCMC and the likelihoods derived by Higdon et al. [16] (for PC-GP) and in Appendix B (for iTPRS-GP). To reduce computational complexity for the iTPRS-GP emulator, we applied the Woodbury matrix identity (e.g. [13], Ch.8) to update the inverse and determinant of the expanded model matrix within the MCMC iterations. This replaces the inversion of a $np \times np$ square matrix with the inversion of a $n \times n$ matrix when sampling each of the nugget, τ_k and $\boldsymbol{\theta}_k$. An outline of this implementation is given in Appendix C. The MCMC was run for 10000 iterations with a burn-in of 1000 which produced acceptable trace plots with no bias from starting location. No thinning is performed as the trace plots showed the chains mixing well. Specification of the prior hyper-parameters for the common Gamma distributions for each τ_k^{-1} is simplified by the standardization of the simulator output, which leads to an expectation of the variance of each $\beta_k(\boldsymbol{x})$ being close to one. Hence, we chose an informative Gamma$(5, 0.2)$ prior distribution, with density $f(\tau^{-1}) \propto \tau^{-4} \exp(-5/\tau)$, again following [16].

Other hyper-parameters for all four emulators, including the numbers of regression basis functions in (1), were chosen to minimise the root mean squared error (RMSE) of prediction on the validation runs. Alternatives include choosing the hyper-parameters to maximise the integrated likelihood but use of the RMSE is consistent with our goal of accurate prediction for untried $\boldsymbol{x}$. For all the dimension-reduced emulators (PC-GP, iTPRS-GP and sTPRS-GP), the number of basis functions p was the main determinant of RMSE. The choice of p controls the trade-off between detailed modeling of the training data and the generalization

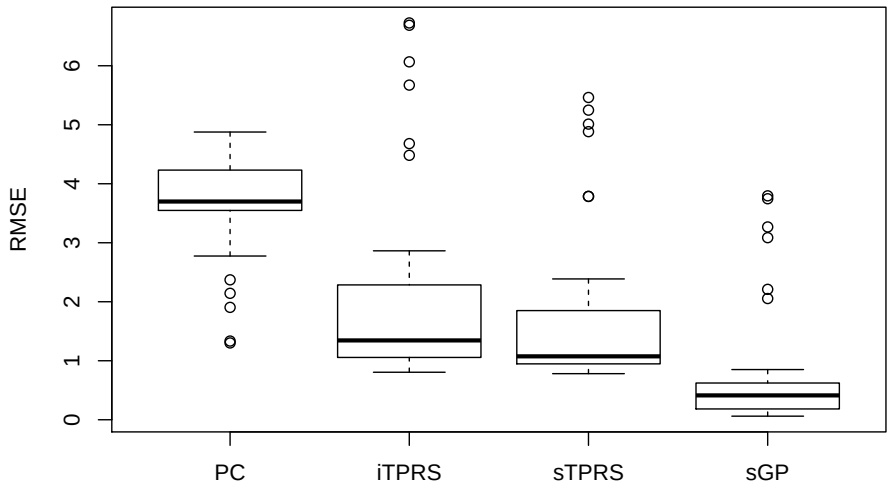


Figure 4: Root mean squared error for each of the PC-GP, iTPRS-GP, sTPRS-GP and sGP emulators, calculated using the posterior predictive mean across the test set.

to untried cases. Increasing p also increases the computational expense of the model fitting. For all three dimension-reduced emulators, we found relatively small values of p minimized the RMSE.

For the PC-GP and iTPRS emulators, these hyper-parameters consisted of a_θ and b_θ, common to the independent Beta prior distributions assumed for the entries of $\boldsymbol{\theta}_k$ (with density $f(\theta) \propto \theta^{a_\theta - 1}(1-\theta)^{b_\theta - 1}$), a_σ and b_σ for the Gamma distribution assumed for σ^{-2}, and the numbers of principal components (PC-GP) and thin plate spline basis functions (iTPRS-GP). We choose $a_\sigma = 2$ to define a Gamma distribution with $\lim_{z\to 0^+} f(z) = 0$ and $\lim_{z\to 0^+} f'(z) = b_\sigma^{-2}$, so that larger values of b_σ result in higher prior density for larger values of σ^{-2}. In general, RMSE was lowest for lower values of b_σ, and also smaller numbers of basis functions. Various shapes of Beta prior density were tested and the results are fairly robust to the choices of a_θ and b_θ. For both emulators, we chose $b_\sigma = 0.01$ and $a_\theta = 1$; for the PC-GP emulator, $p = 3$ and $b_\theta = 3$, and for the iTPRS-GP emulator, $p = 5$ and $b_\theta = 0.1$. Several similar hyper-parameter settings produced comparable RMSE.

For the sTPRS-GP and sGP emulators, correlation parameters $\boldsymbol{\theta} = (\theta_1, \theta_2)^{\mathrm{T}}$ and $\boldsymbol{\nu} = (\nu_1, \nu_2)^{\mathrm{T}}$ are chosen using the validation runs, together with hyper-parameters a_τ and b_τ for the Gamma prior distribution on τ. The numerical results indicated that the RMSE was robust to the choice of a_τ and b_τ, and a Gamma(1,1) prior distribution was chosen, giving near linearly decreasing support between 0 and 1. For the sTPRS-GP emulator, the minimum RMSE was achieved with $p = 5$ and $\boldsymbol{\theta} = \boldsymbol{\nu} = (0.05, 0.05)^{\mathrm{T}}$. For the sGP emulator, the minimum RMSE occurred with $\boldsymbol{\theta} = (0.5, 0.65)^{\mathrm{T}}$ and $\boldsymbol{\nu} = (0.8, 0.4)^{\mathrm{T}}$. For these two emulators, σ^2 is estimated from the mean squared error on the test data.

4.4 Comparison of Emulators

Each of the four emulators were used to predict the logged dosage for the 35 test runs using the posterior predictive mean. All four emulators produced similar spatial dispersion to the true simulator, illustrated in Figure 3. The PC-GP emulator has generally higher RMSE across all of the test runs; see Figure 4.

The median RMSE for the PC-GP emulator (3.70) is considerably greater than the upper quartile of the iTPRS-GP (2.28) or sTPRS-GP (1.85) emulators. The sTPRS-GP emulator performs slightly better than the iTPRS-GP emulator, with lower quartiles of 1.07 for sTPRS-GP and 1.35 for iTPRS-GP. These differences demonstrate the advantage of modeling the within-run correlation between basis functions. For this low-resolution spatial grid, the sGP emulator shows better performance than any of the other emulators, with lower and upper quartiles of 0.19 and 0.61 respectively. In Section 4.6, we investigate if the sGP emulator maintains this advantage over the sTPRS-GP emulator for a high-resolution output grid where the sGP emulator can only be trained on a subset of the output data.

4.5 Uncertainty quantification

The Bayesian approach to emulation allows the uncertainty associated with predictions of the simulator to be quantified and assessed. Realistic and appropriate uncertainty quantification is a key determinant of a good emulator. Prediction uncertainty for the PC-GP and iTPRS-GP emulators can be obtained from the MCMC samples. For the sTPRS emulator, we use (6) with the conditional posterior distribution for $\boldsymbol{\beta}$ and an additive error term corresponding to the approximation error (3). A similar approach is adopted for the sGP emulator [27].

To demonstrate the predicted uncertainty from the four emulators of the dispersion simulation, in Figure 5 we present prediction intervals (posterior predictive mean ± 3 posterior predictive standard errors) for a single test run for each of the PC-GP, iTPRS-GP, sTPRS-GP and sGP emulators; other test runs produce similar results. The sample points in these figures are ordered by row, taking every 16th row and column from Figure 2. They can be related to the output grid in Figure 2 by considering each set of 8 consecutive points as a horizontal transect across the grid. The locations of the points in the output space $\mathcal{S}$ are illustrated in Figure 6. As would be expected in this application, the predictions have higher uncertainty in the more complex central areas of the output grid, with higher predictive means.

We also calculate the frequentist coverage of these prediction intervals using the test data; we would expect coverage of around 99%. From Figure 5, it is clear that excluding within-run correlation in the iTPRS-GP emulator has led to substantial under-estimation of the uncertainty (coverage of 28%), with almost all observed responses lying outside the, very narrow, prediction intervals. The MCMC chains had converged and sufficiently explored the parameter space, and hence the under-estimation of the uncertainty appears to be a consequence of modeling assumptions. The other emulators give more realistic assessment of uncertainty, with coverages of 83% (PC-GP), 98% (sTPRS-GP) and 100% (sGP). The much wider predictive intervals for the sGP emulator plausibly result from the larger effective number of parameters in this emulator (with separate, dependent, Gaussian process models for each output point).

4.6 Prediction on a high-resolution output grid

In our exemplar simulation, the output grid covers a 10km by 10km spatial area. The low-resolution output grid consider up to this point can only accurately assess the performance of sensor locations at around a 1km resolution. For application in a sensor placement tool, much higher resolution ($\sim$10m) is required. To assess the performance of the sTPRS-GP and sGP emulators at a high resolution, we define a 63×63 regular lattice as an output grid (with no boundary points) and attempt to predict the output for untried $\boldsymbol{x}$ on this much finer scale. We choose these two emulators because of their superior performance on the low-resolution output grid.

For this size output grid ($r = 3639$), it is no longer computationally feasible to train the sGP emulator using the whole output grid in realistic time (it requires multiple inversions of a 3696×3696 correlation matrix). However, as the sGP is a model for functional data, instead we train the emulator using an 8×8 sub-grid (actually the same output grid as in the previous sections) and then predict at the interim locations. The sTPRS-GP emulator, however, performs dimension reduction on the output grid and hence can be trained using much larger data sets, with the basis functions defined using the complete output grid. The benefit of this method is two-fold: prediction is actually computationally less expensive and all of the

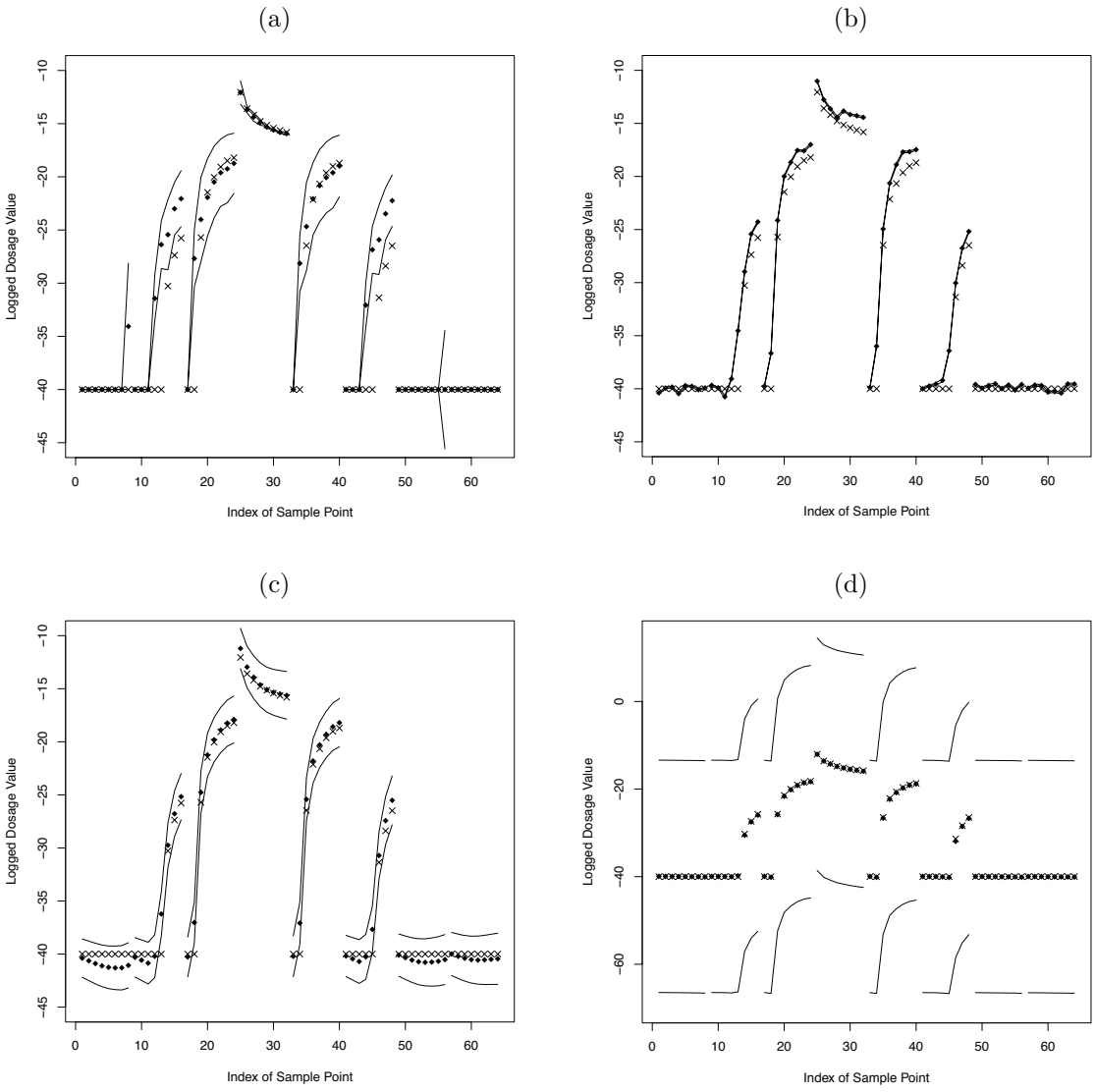


Figure 5: Logged dosage (×), approximate posterior predictive mean (•) and approximate 99% predictive intervals for the $r = 64$ spatial locations for one test run for the (a) PC-GP, (b) iTPRS, (c) sTPRS and (d) sGP emulators. The ordering of the output points corresponds to Figure 6.

output data can be used, providing more accurate prediction across the whole output domain $\mathcal{S}$. The RMSE of the two approaches are shown in Figure 7.

It is clear from this figure that the sTPRS emulator provides substantially more accurate predictions that the sGP emulator. An important advantage here of the sTPRS-GP emulator is that it does not need to use an estimated correlation structure on the output grid for prediction, unlike the sGP emulator. When the output grid displays non-stationary correlation, as with this dispersion example, the sGP emulator struggles to accurately predict at interim locations, see Figure 8 for an illustration. In general, for detailed output domains, e.g. urban locations, using a low-resolution output grid to train the sGP emulator could result in missing important details in the response.

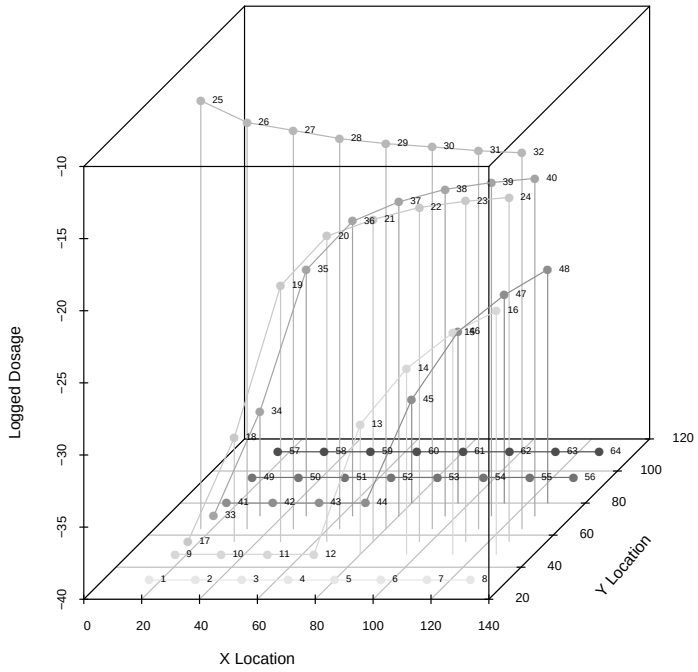


Figure 6: Relationship between the index of the sample output points and the spatial location.

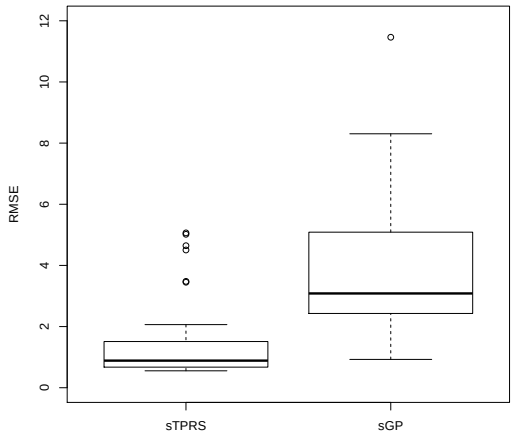
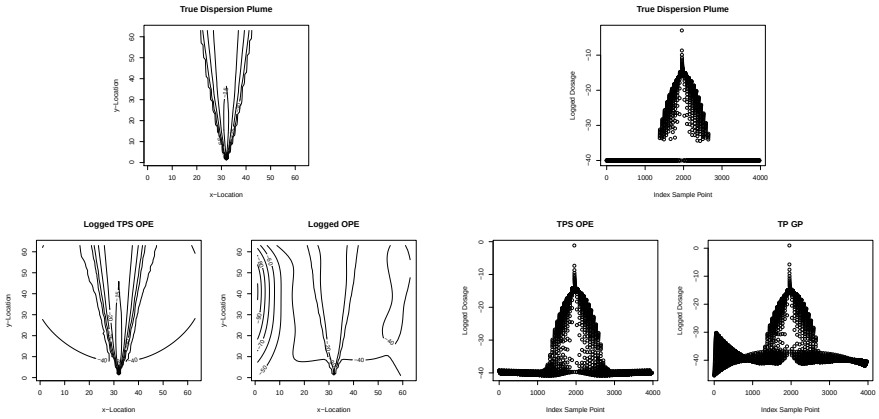


Figure 7: RMSE for the sTPRS-GP and sGP emulators for the high-resolution output grid using the posterior predictive mean across the test set.



(a) Contour Plot (b) Index Plot

Figure 8: Contour plot and index logged dosage to compare the sTPRS-GP and sGP emulators.

5 Artificial example

The dispersion example demonstrated the effectiveness of the sTPRS-GP emulator for prediction on dense grids with changing spatial variation. In this section, we use an artificial environmental example to further compare the sTPRS-GP and sGP emulators.

The example is a modified version of a simulator that has previously been used to demonstrate calibration methodology [3]. The simulator models a chemical pollutant spill at two locations into a long, narrow holding channel. We assume the location and time of both spills is known and fixed but that the mass and diffusion rate at each spill location may be inputs to the simulator. For the pollutant concentration at location s_1 and time s_2 (so $q = 2$), the simulator has the simple closed form

$$Y(\boldsymbol{s}, \boldsymbol{x}) = \frac{x_1}{\sqrt{4\pi x_2 s_2}} \exp\left(\frac{-s_1^2}{4x_2 s_2}\right) + \frac{x_3}{\sqrt{4\pi x_4 (s_2 - \tau)}} \exp\left(\frac{-(s_1 - L)^2}{4x_4(s_2 - \tau)}\right) \mathbb{I}(s_2 > \tau)\,, \quad (10)$$

where x_1, x_2 and x_3, x_4 are the mass and diffusion rate of spilled pollutant at spills 1 and 2, respectively. The first spill is at (location, time) $= (0, 0)$; the second is at $(L, \tau) = (1.505, 30.1525)$.

We generated data from simulator (10) for four scenarios, corresponding to $d = 1, \ldots, 4$ variables, starting with varying just the first mass x_1, then the first mass and diffusion rate (x_1, x_2), and so on. Variables held fixed were set to the mid-points of their ranges, $7 \leq x_1, x_3 \leq 13$ and $0.02 \leq x_2, x_4 \leq 0.12$. We adapted code from `http://www.sfu.ca/~ssurjano` and used four $n = 80$ run maximin Latin Hypercube designs with $q = 1, \ldots, 4$ to generate the training data, and separate samples from continuous distributions to generate validation and test data sets, each with 10 runs.

We took the most effective emulators from the dispersion example, sTPRS-GP and sGP, and applied them to the four scenarios. For each scenario, the validation data was used to choose the various prior hyper-parameters for each emulator, including the number, p, of functions in the thin-plate regression spline basis. The sTPRS-GP emulator was estimated using an output grid with $r = 2500$ location/time points, whereas the sGP emulator was estimated using a much smaller grid with $r = 100$ points. Fitting the sGP emulator to the larger output grid was computationally infeasible. In each case, a $r = 2500$ output grid was used for the validation and test sets. All output grids were regular two-dimensional lattices with no boundary points. We again scaled the training, validation and test data sets to have concentration with mean zero and standard deviation 1 at each output point.

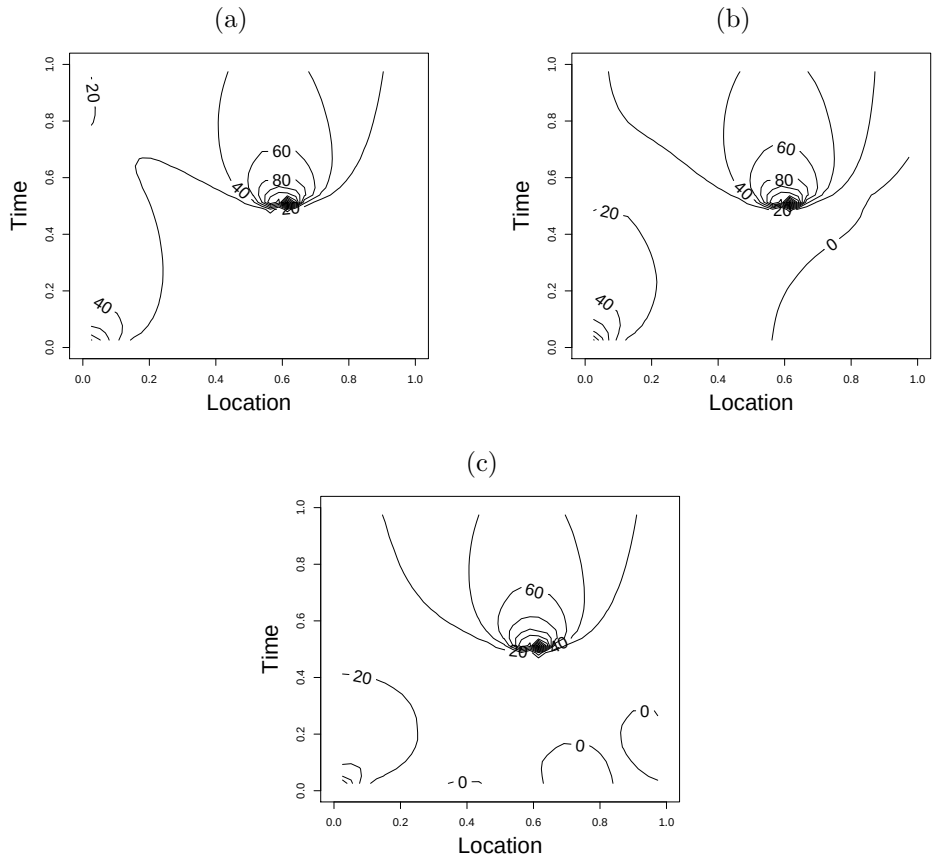


Figure 9: Contour plots for the artificial environment example showing (a) the true concentration surface from the simulator and the predicted concentration surface from (b) the sTPRS-GP emulator and (c) the sGP emulator.

Figure 9 shows some typical output from this modeling exercise when $d = 4$. In this case, both the sTPRS-GP and sGP emulators capture the basic features of the true concentration surface. However, the sGP emulator, which was estimated using the much smaller output grid, has overestimated the mass and/or diffusion at both pollutant spills, resulting in higher predicted concentrations than the sTPRS-GP, which better captures the true concentration. Here, the sTPRS-GP has average root mean squared error 7% smaller than the TP-GP.

For fewer input variables ($d < 4$), the sGP emulator had a predictive advantage in terms of RMSE, see Figure 10. For $d = 1, 2, 3$, the improvement in average RMSE for the sGP was between 2% and 26%, with the sTPRS-GP performing comparatively poorest for $d = 3$. While there seems to be little pattern in performance here, it is plausible that the improved performance of the TPRS-GP emulator for $d = 4$ is due to (i) greater variation in the response across the output grid, particularly when there are high levels of concentrated mass at the two sources (high mass, low diffusion rate) and (ii) more detailed changes in the response surface occurring between runs. Both these situations occur more often when both diffusion rates are varied.

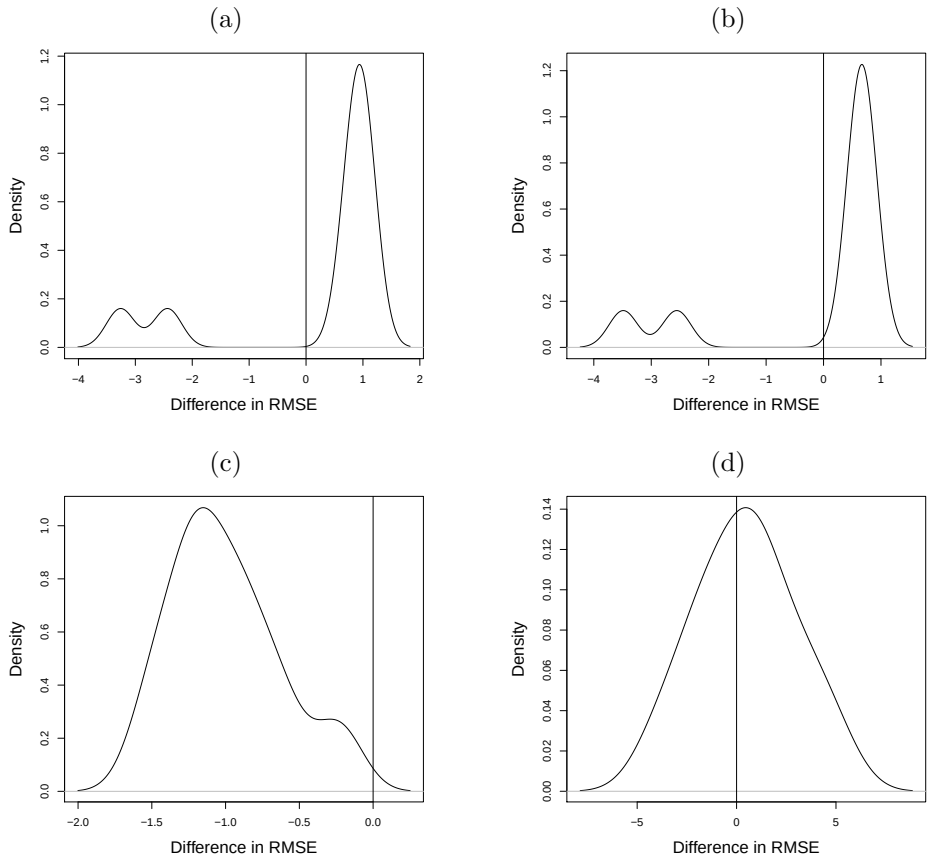


Figure 10: Density plots of the difference in test set RMSE between the sGP and sTPRS-GP emulators for the environmental simulator with (a) one, (b) two, (c) three and (d) four variables.

6 Discussion and future work

We have presented the emulation of multivariate simulators with two-dimensional output structure. Thin-plate regression splines were demonstrated to be an effective method for dimensional-reduction, that resulted in substantially increased prediction accuracy over the use of principal components. When a separable covariance structure is assumed for the basis coefficients, a computationally feasible emulator results that provides realistic measures of uncertainty. For the separable emulator, we have adopted a plug-in approach using the posterior predictive distribution conditional on the correlation parameters. Clearly, there is the potential for this approach to under-estimate the posterior uncertainty. However, the results in Section 4.5 suggest that it is more important to account for the within-run correlations resulting from the non-orthogonal thin-plate spline basis functions.

When high-resolution prediction was required in the dispersion example, the thin plate regression methodology proved more accurate and computationally feasible than functional data modelling using a separable Gaussian process model. A key reason for the advantage of the TPRS approach is the non-stationary nature of the output data, with differing correlation lengths across the output domain. The artificial example demonstrated that the two methodologies can perform very similarly, although sTPRS-GP emulator was again preferred when non-stationary output variation was observed. An area for future research is to define more fully the scenarios (non-stationary correlation, high-resolution spatial features) in which the TPRS approach has a predictive advantage. Other research could include developing multivariate emulators using Gaussian processes with non-stationary or non-separable covariance structures [11], modeling dispersion

simulators with qualitative inputs (e.g. [24]) such as release type relating to the source term, and building emulators for dynamic responses such as concentration over time. Further research is also required into designing the computer experiments for these multivariate hierarchical problems, including the choice of the simulator inputs and also, in some applications, the selection of output domain locations (cf [2]).

Acknowledgments

This work was supported by the Defense Threat Reduction Agency (grant HDTRA1-08-1-0048), a Dstl Associate Fellowship for V.E. Bowman and an EPSRC Fellowship (EP/J018317/1) for D.C. Woods.

A Development of the conditional posterior density for the sTPRS-GP emulator

For $\boldsymbol{\beta}|\tau, C \sim N(\mathbf{0}_{np}, C\tau)$, a mean-zero Gaussian process,

$$\boldsymbol{\beta}^\star|\boldsymbol{\beta}, \tau \sim N(\boldsymbol{m}, \tau S)\,,$$

with $\boldsymbol{m} = (C^\star)^{\mathrm{T}} C^{-1}\boldsymbol{\beta}$, and $S = C^{\star\star} - (C^\star)^{\mathrm{T}} C^{-1} C^\star$. Here, $C^\star$ is the $np \times p$ matrix of correlations between $\boldsymbol{\beta}$ and $\boldsymbol{\beta}^\star$, and $C^{\star\star}$ is the $p \times p$ matrix of correlations between the elements of $\boldsymbol{\beta}^\star$.

Hence, for a separable correlation structure with with $C = W \otimes V$, point estimates of $\boldsymbol{\beta}^\star$ can be provided by

$$E(\boldsymbol{\beta}^\star|\boldsymbol{\beta}, \tau) \quad = \quad (\boldsymbol{w}^\star \otimes V)^{\mathrm{T}} W \otimes V\boldsymbol{\beta}\,,$$

with $\boldsymbol{w}^\star$ the n-vector of correlations between $\boldsymbol{x}^\star$ and $\boldsymbol{x}$. Similarly, uncertainty in $\boldsymbol{\beta}^\star$ can be assessed via

$$\mathrm{Var}(\boldsymbol{\beta}^\star|\boldsymbol{\beta}, \tau) = \tau \left\{ C^{\star\star} - V \otimes (\boldsymbol{w}^\star)^{\mathrm{T}} \left(W^{-1} \otimes V^{-1}\right) \boldsymbol{w}^\star \otimes V \right\}. \tag{11}$$

In addition,

$$\pi(\tau|\boldsymbol{\beta}) \propto \tau^{-1-(a_\tau+np)/2} \exp\left\{ -(b_\tau + \boldsymbol{\beta}^{\mathrm{T}} C^{-1}\boldsymbol{\beta})/2\tau^{-1} \right\}.$$

That is, $\tau^{-1}|\boldsymbol{\beta} \sim \mathrm{Gamma}(a_\tau + np, (b_\tau + \boldsymbol{\beta}^{\mathrm{T}} C^{-1}\boldsymbol{\beta})^{-1})$. A plug-in estimate of the posterior uncertainty in $\boldsymbol{\beta}$ can be achieved by replacing τ in (11) with its posterior mean, $E(\tau|\boldsymbol{\beta}) = \left(b_\tau + \boldsymbol{\beta}^{\mathrm{T}} W^{-1} \otimes V^{-1}\boldsymbol{\beta}\right)/(a_\tau + np - 2)$.

B Construction of the sampling distribution for the iTPRS-GP emulator

From equation (2),

$$\boldsymbol{\beta}|C \sim N(\mathbf{0}_{np}, C)\,, \tag{12}$$

where C is an $np \times np$ covariance matrix determined by the specified covariance structure and parameter vectors $\boldsymbol{\tau}$ and $\boldsymbol{\theta}$. We specify independent Gamma(a_τ, b_τ) priors for each τ_i and independent Beta(a_θ, b_θ) priors for each θ_i

$$\begin{aligned}
\pi(\tau_i) &\propto \tau_i^{a_\tau - 1} \exp^{-\tau_i / b_\tau} \qquad i = 1, \ldots, pn\,, \\
\pi(\theta_{ik}) &\propto \theta_{ik}^{a_\theta - 1}(1 - \theta_{ik})^{b_\theta - 1} \qquad i = 1, \ldots, pn;\ k = 1 \ldots dp\,.
\end{aligned}$$

We define an nr vector $\tilde{\boldsymbol{Y}}$ to be the concatenation of all n simulation output vectors

$$\tilde{\boldsymbol{Y}} = \mathrm{vec}([\boldsymbol{Y}_1; \ldots; \boldsymbol{Y}_n])\,.$$

Given the precision σ^{-2} of the errors the likelihood is then

$$L(\tilde{Y}|\boldsymbol{\beta}, \sigma^2) \propto \sigma^{-nr} \exp\{-\frac{1}{2}\sigma^{-2}(\tilde{\boldsymbol{Y}} - A\boldsymbol{\beta})^T(\tilde{\boldsymbol{Y}} - A\boldsymbol{\beta})\}\,,$$

where the $nr \times np$ matrix

$$A = [I_n \otimes \boldsymbol{a}_1; \ldots; I_n \otimes \boldsymbol{a}_p]$$

is constructed from the basis vectors $\boldsymbol{a}_k$. A Gamma(a_σ, b_σ) distribution is specified for the error precision σ^{-2}.

We then follow the factorization in [16],

$$\begin{aligned}
L(\tilde{Y}|\boldsymbol{\beta}, \sigma^2) \propto\ & \sigma^{-np} \exp\{-\frac{1}{2}\sigma^{-2}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})^T(A^TA)(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})\} \times \\
& \sigma^{-n(r-p)} \exp\{-\frac{1}{2}\sigma^{-2}\tilde{\boldsymbol{Y}}^T(I - A(A^TA)^{-1}A^T)\tilde{\boldsymbol{Y}}\}\,,
\end{aligned}$$

to define a dimension reduced likelihood and a modified Gamma(a'_σ, b'_σ) prior for σ^{-2}:

$$\begin{aligned}
L(\hat{\boldsymbol{\beta}}|\boldsymbol{\beta}, \sigma^2) &\propto \sigma^{-np} \exp\{-\frac{1}{2}\sigma^{-2}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})^T(A^TA)(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})\}\,, \\
a'_\sigma &= a_\sigma + \frac{n(r-p)}{2}\,, \\
b'_\sigma &= b_\sigma + \frac{1}{2}\tilde{\boldsymbol{Y}}^T(I - A(A^TA)^{-1}A^T)\tilde{\boldsymbol{Y}}\,, \\
\hat{\boldsymbol{\beta}} &= (A^TA)^{-1}A^T\tilde{\boldsymbol{Y}}\,.
\end{aligned}$$

Then the normal-gamma model

$$\tilde{\boldsymbol{Y}}|\boldsymbol{\beta}, \sigma^2 \sim N(A\boldsymbol{\beta}, \sigma^{-2}I_{nr})\,, \qquad \sigma^{-2} \sim \mathrm{Gamma}(a_\sigma, b_\sigma)$$

is equivalent to

$$\hat{\boldsymbol{\beta}}|\boldsymbol{\beta}, \sigma^2 \sim N(\boldsymbol{\beta}, \sigma^2(A^TA)^{-1})\,, \qquad \sigma^{-2} \sim \mathrm{Gamma}(a'_\sigma, b'_\sigma)\,.$$

The likelihood depends on the simulator data only through $\hat{\boldsymbol{\beta}}$; therefore integrating out $\boldsymbol{\beta}$ with respect to its prior distribution (12) gives

$$\begin{aligned}\pi(\sigma^{-2},\boldsymbol{\tau},\boldsymbol{\theta}|\tilde{\boldsymbol{Y}}) \quad \propto \quad & |(\sigma^{-2}A^TA)^{-1}+C|^{-\frac{1}{2}} \times \\ & \exp\{-\frac{1}{2}\hat{\boldsymbol{\beta}}^T([\sigma^{-2}A^TA)^{-1}]+C])^{-1}\hat{\boldsymbol{\beta}}\} \times \\ & (\sigma^{-2})^{a'_\sigma-1}\exp^{-\sigma^{-2}/b'_\sigma} \times \prod_{i=1}^{p}\tau_i^{a_\tau-1}\exp^{-\tau_i/b_\tau} \times \\ & \prod_{i=1}^{p}\left\{\prod_{k=1}^{q_1}\theta_{ik}^{a_\theta-1}(1-\theta_{ik})^{b_\theta-1}\right\}.\end{aligned}$$

This posterior distribution is explored via MCMC using standard metropolis updates. Conditional on the hyper-parameters, the posterior distribution of $\beta_k(\boldsymbol{x})$ is a Gaussian process of fairly standard form [27]. Samples from the unconditional posterior for $\beta_k(\boldsymbol{x})$ can be obtained by substituting samples from the MCMC chain for $\sigma^{-2},\boldsymbol{\tau},\boldsymbol{\theta}|\tilde{\boldsymbol{Y}}$.

The MCMC simulation is hampered by the inversion of the matrix

$$[(\sigma^{-2}A^TA)^{-1}+C],$$

which is of size $np\times np$. However, only part of the matrix is updated at each step of the MCMC, therefore the matrix inversion is carried out once at the start of the chain and then updated via the method described in Appendix C.

C Reduction of Computational Burden

Inversion of the $np\times np$ matrix $[(\sigma^{-2}A^TA)^{-1}+C]$, see Appendix B, is computationally intensive when using a non-orthogonal basis such as a thin plate spline. To overcome this problem and make the MCMC updates feasible, we note that any update of the parameters $\boldsymbol{\tau}$, $\boldsymbol{\theta}$, or the nugget only change an $n\times n$ sub-matrix of the covariance matrix C. This sub-matrix depends on the parameter being updated, and hence care is required in locating the correct segment. Once the inverse of the whole matrix has been performed for the update of σ^{-2} the proceeding inverses can be computed using the Woodbury formula.

$$(D+PQ)^{-1}=D^{-1}-D^{-1}P(I+QD^{-1}P)^{-1}QD^{-1},$$

where D, P and Q all denote matrices of the correct size, and I is an identity matrix. For our problem, $D=[(\sigma^{-2}A^TA)^{-1}+C]$ and is size $np\times np$, P is size $np\times n$ and Q is size $n\times np$. In order to make use of this result, we need to find a PQ equal to the difference in the D matrix resulting from the update.

Let

$$P=\begin{pmatrix}\mathbf{0}\\P_1\\\mathbf{0}\end{pmatrix},\quad Q=\begin{pmatrix}\mathbf{0} & Q_1 & \mathbf{0}\end{pmatrix}$$

where each $\mathbf{0}$ denotes a matrix of the correct dimension to position the change in the D matrix for the current update. Note that in the first updates, the initial $\mathbf{0}$ matrix will be of size 0×0 and in the last updates the final $\mathbf{0}$ matrix will be of size 0×0. Then

$$PQ=\begin{pmatrix}\mathbf{0} & \mathbf{0} & \mathbf{0}\\\mathbf{0} & P_1Q_1 & \mathbf{0}\\\mathbf{0} & \mathbf{0} & \mathbf{0}\end{pmatrix}.$$

We obtain P_1Q_1 by performing LU decomposition on the difference between the current D matrix and the proposed D matrix, a simple $np \times np$ subtraction. Using block notation and remembering that D and therefore D^{-1} are symmetric, let

$$D^{-1} = \begin{pmatrix} D_{11}^{-1} & D_{12}^{-1} & D_{13}^{-1} \\ D_{12}^{-T} & D_{22}^{-1} & D_{23}^{-1} \\ D_{13}^{-T} & D_{23}^{-T} & D_{33}^{-1} \end{pmatrix}.$$

Then $QD^{-1}P$ can be replaced by $Q_1D_{22}^{-1}P_1$, and inversion of $(I + QD^{-1}P)^{-1}$ by inversion of $X = (I + Q_1D_{22}^{-1}P_1)^{-1}$, an $n \times n$ matrix.

Finally simple block multiplication of the defined matrices, taking account of symmetry, results in

$$\begin{aligned} (D+PQ)^{-1} &= D^{-1} - \begin{pmatrix} D_{11}^{-1} & D_{12}^{-1} & D_{13}^{-1} \\ D_{12}^{-T} & D_{22}^{-1} & D_{23}^{-1} \\ D_{13}^{-T} & D_{23}^{-T} & D_{33}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & P_1XQ_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \times \\ &\qquad \begin{pmatrix} D_{11}^{-1} & D_{12}^{-1} & D_{13}^{-1} \\ D_{12}^{-T} & D_{22}^{-1} & D_{23}^{-1} \\ D_{13}^{-T} & D_{23}^{-T} & D_{33}^{-1} \end{pmatrix} \\ &= D^{-1} - \begin{pmatrix} D_{12}^{-1}P_1XQ_1D_{12}^{-T} & D_{12}^{-1}P_1XQ_1D_{22}^{-1} & D_{12}^{-1}P_1XQ_1D_{23}^{-1} \\ D_{22}^{-1}P_1XQ_1D_{12}^{-T} & D_{22}^{-1}P_1XQ_1D_{22}^{-1} & D_{22}^{-1}P_1XQ_1D_{23}^{-1} \\ D_{23}^{-T}P_1XQ_1D_{12}^{-T} & D_{23}^{-T}P_1XQ_1D_{22}^{-1} & D_{23}^{-T}P_1XQ_1D_{23}^{-1} \end{pmatrix}. \end{aligned}$$

This can be computed efficiently using bespoke multiplication routines. However, for large iteration cycles it is recommended that a full inversion be performed approximately once every 100 steps to ensure that small numerical errors do not appreciate. A similar derivation can be performed using the matrix determinant lemma [14].

References

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

[2] S. Biedermann, H. Dette, and D. C. Woods, *Optimal design for additive partially nonlinear models*, Biometrika, 98 (2011), pp. 449–458.

[3] N. Bliznyuk, D. Ruppert, C. Shoemaker, R. Regis, S. Wild, and P. Mugunthan, *Bayesian calibration and uncertainty analysis for computationally expensive models using optimization and radial basis function approximation*, Journal of Computational and Graphical Statistics, 17 (2008), pp. 270–294.

[4] V. E. Bowman and D. C. Woods, *Weighted space-filling designs*, Journal of Simulation, 7 (2013), pp. 249–263.

[5] K. Campbell, M. D. McKay, and B. J. Williams, *Sensitivity analysis when model outputs are functions*, Reliability Engineering and System Safety, 91 (2006), pp. 1468–1472.

[6] A. Chakraborty, B. K. Mallick, R. G. Mcclarren, C. C. Kuranz, D. Bingham, M. J. Grosskopf, E. M. Rutter, H. F. Stripling, and R. P. Drake, *Spline-based emulators for radiative shock experiments with measurement error*, Journal of the American Statistical Association, 108 (2013), pp. 411–428.

[7] R. H. Clarke, *The first report of a working group on atmospheric dispersion: A model for short and medium range dispersion of radionuclides released to the atmosphere*, Tech. Rep. NRPB-R91, National Radiological Protection Board, Harwell, 1979.

[8] S. Conti and A. O'Hagan, *Bayesian emulation of complex multi-output and dynamic computer models*, Journal of Statistical Planning and Inference, 140 (2010), pp. 640–651.

[9] K. Fang, R. Li, and A. Sudjianto, *Design and Modelling for Computer Experiments*, Chapman and Hall, Boca Raton, 2006.

[10] A. I. J. Forrester, A. Sobester, and A. J. Keane, *Engineering Design via Surrogate Modelling*, Wiley, Chichester, 2008.

[11] T. E. Fricker, J. E. Oakley, and N. M. Urban, *Multivariate Gaussian process emulators with nonseparable covariance structures*, Technometrics, 55 (2013), pp. 47–56.

[12] R. B. Gramacy and H. K. H. Lee, *Cases for the nugget in modeling computer experiments*, Statistics and Computing, 22 (2012), pp. 713–722.

[13] F. A. Graybill, *Matrices with Applications in Statistics*, Wadsworth, Belmont, CA, 1969.

[14] D. A. Harville, *Matrix Algebra from a Statisticians Perspective*, Springer, New York, 1997.

[15] T. Hastie, R. Tibshirani, and J. H. Friedman, *Elements of Statistical Learning*, Springer, New York, 2nd ed., 2009.

[16] D. Higdon, J. Gattiker, B. J. Williams, and M. Rightly, *Computer model validation using high-dimensional output*, Journal of the American Statistical Association, 103 (2008), pp. 570–583.

[17] I. T. Jolliffe, *Principal Component Analysis*, Springer, New York, 2nd ed., 2002.

[18] A. R. Jones, D. J. Thomson, M. Hort, and B. Devenish, *The U.K. Met Office's next-generation atmospheric dispersion model, NAME III*, in Air Pollution Modeling and its Application XVII, C. Borrego and A.-L. Norman, eds., New York, 2007, Springer, pp. 508–589.

[19] M. C. Kennedy, C. W. Anderson, and A. O'Hagan, *Case studies in Gaussian process modelling of computer codes*, in Sensitivity Analysis of Model Output, Los Alamos National Laboratory, 2005, pp. 476–485.

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

[21] M. D. Morris and T. J. Mitchell, *Exploratory designs for computer experiments*, Journal of Statistical Planning and Inference, 43 (1995), pp. 381–402.

[22] A. O'Hagan and J. J. Forster, *Bayesian Inference*, vol. 2B of Kendall's Advanced Theory of Statistics, Arnold, London, 2nd ed., 2004.

[23] K. Politis and L. Robertson, *Bayesian updating of atmospheric dispersion after a nuclear accident*, Journal of the Royal Statistical Society C, 53 (2004), pp. 583–600.

[24] P. Z. G. Qian, H. Wu, and C. F. J. Wu, *Gaussian process models for computer experiments with qualitative and quantitative factors,*, Technometrics, 50 (2008), pp. 383–396.

[25] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*, MIT Press, Cambridge, MA, 2006.

[26] P. Robins, V. E. Rapley, and N. Green, *Realtime sequential inference of static parameters with expensive likelihood calculations*, Journal of the Royal Statistical Society C, 58 (2009), pp. 641–662.

[27] J. C. Rougier, *Efficient emulators for multivariate deterministic functions*, Journal of Computational and Graphical Statistics, 17 (2008), pp. 827–843.

[28] J. C. Rougier, S. Guillas, A. Maute, and A. D. Richmond, *Expert knowledge and multivariate emulation: the thermosphere-ionosphere electrodynamics general circulation model (tie-gcm)*, Technometrics, 51 (2009), pp. 414–424.

[29] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, *Design and analysis of computer experiments (with discussion)*, Statistical Science, 4 (1989), pp. 409–435.

[30] T. J. Santner, B. J. Williams, and W. I. Notz, *The Design and Analysis of Computer Experiments*, Springer, New York, 2003.

[31] J. Smith and S. French, *Bayesian updating of atmospheric dispersion models for use after an accidental release of radioactivity*, Journal of the Royal Statistical Society D, 42 (1993), pp. 501–511.

[32] R. I. Sykes, S. F. Parker, D. S. Henn, C. P. Cerasoli, and L. P. Santos, *PC-SCIPUFF Version 1.2pd Technical Documentation*, Tech. Rep. ARAP 718, ARAP Group, Princeton, NJ, 1998.

[33] S. N. Wood, *Thin plate regression splines*, Journal of the Royal Statistical Society B, 65 (2003), pp. 95–114.