Emulation of Numerical Models with Over-specified Basis Functions

Avishek Chakraborty†, Derek Bingham and Bani Mallick

Abstract: We propose a Bayesian approach to statistical inference in emulation and calibration of a numerical model with a basis-function representation. Numerical models are commonly used now-a-days as a surrogate for real-life events or experiments which, due to either rarity of occurrence or associated expenses, can be observed only a limited number of occurrences of this event. The goal of our modeling effort is to match the numerical model output with the ones collected from actual events. In addition to unknown parameters in the field experiments, the input-output relationship is also treated as unknowns, allowing a full treatment of all possible sources of uncertainty. To combine these field observations with a collection of runs from the numerical model, we employ a fully model-based approach using the Kennedy-O'Hagan framework of computer model validation. Compared to the conventional Gaussian process emulators, we have achieved greater efficiency with a basis-function representation in terms of smaller predictive uncertainty, greater control over model sparsity and shorter computational time. We have developed an efficient estimation scheme that allows the user to treat the number of functions and their inclusion probabilities as unknown quantities and to estimate them via a reversible jump Monte Carlo sampler. The performance of the model depends on how efficiently we can choose the basis functions so that they can span an wide range of response functions. Here, we use a generalized polynomial chaos expansion - depending on the stochastic model for the input, an optimal class of orthonormal polynomial functionals is chosen from the Askey scheme to represent the emulator. We show that this hierarchical specification provides prior support for a large family of response functions. Along with several simulation examples focusing on different model characteristics, our approach is also implemented with a real dataset from a series of laboratory experiments related to astrophysics.

Keywords and phrases: calibration, computer models, emulators, generalized polynomial chaos, random inputs, reversible jump Markov chain Monte Carlo.

*This work was funded by the Predictive Sciences Academic Alliances Program in DOE/NNSA-ASC via grant DEFC52- 08NA28616
†Corresponding Author, email: avishekc@stat.tamu.edu
1. Introduction

Using deterministic computer models (or simulators) to explore physical systems is common in many scientific disciplines. An important application of simulators, and the application of interest for this paper, is model calibration (Kennedy and O’Hagan, 2001) where realizations of the computer model and field observations are used to estimate parameters that govern the physical system and also to build a predictive model. Such applications arise in a diverse range of applications such as environmental science (Bliznyuk et al., 2008), heat transfer (Higdon et al., 2008a) and hydrology (Confesor and Whittaker, 2007).

The seminal article of Kennedy and O’Hagan (2001) first outlined a fully-model based approach to use simulation output for calibration of unknown features in a field experiment. Their model (hereafter referred to as the Kennedy-O’Hagan model) uses multiple runs from a simulator at designed input configurations to build a predictive model for the input-output relationship. Conditional on the knowledge of that relationship, the model searches for likely values of the calibration parameters that can produce outputs similar to the experimental data. In a hierarchical approach, these steps are iterated sequentially so that the uncertainty in learning one of them is always factored into the other.

The Kennedy-O’Hagan model uses a Gaussian process (GP) to model the simulator output since only limited numbers of model runs were available. The role of GP here is that of an Emulator - an statistical surrogate for the actual simulator. The main reasons for using a GP in this context are that the estimated response surface interpolates the observed outputs at the training points and the model provides a foundation for statistical inference in a deterministic setting. The latter point is most relevant to our work. After conditioning on the outputs and statistical model parameters, the uncertainty at unsampled inputs arises from the different sample paths the GP can take - a type of model uncertainty.

A related problem that has received recent interest is that of propagating the distribution of uncertainty inputs across a computation model. The object of interest is in this context the distribution of outputs. The most common approach is that the use of polynomial chaos (Xiu and Karniadakis, 2002) expansions. The so-called non-intrusive approach is most related to
the problems we consider insofar as the numerical model is treated as a black-box. This method relies on the simulator output at a set of collocation points in the input domain which can be used to determine the expansion coefficients (e.g., see Xiu (2007)).

In this article, we replace the GP in the Kennedy and O’Hagan model with a basis function expansion. In the context of computer model calibration, Higdon et al. (2008b) used an expansion of univariate GPs to model the high-dimensional computer outputs. Bliznyuk et al. (2008) used radial basis functions as a model for emulator. In current article, instead of choosing a fixed set of basis functions, we explicitly use the nature of stochasticity in the input (i.e., prior distributions) to select the appropriate family of functions. This can be achieved with the generalized polynomial chaos expansions (gPC; Xiu and Karniadakis, 2003) that were originally used for variance propagation. The form of the polynomials is chosen to create an orthogonal family with respect to the input (joint) distribution which implies faster convergence for the emulator to the true simulation function. The model we propose in Section 3 uses a set of outputs collected from the numerical model at known input configurations, specifies a stochastic model for the transformation process and employs a Markov chain Monte Carlo (MCMC) scheme to learn the posterior predictive distribution of the unknown input for the field experiment. This two-step hierarchical model facilitates an exhaustive treatment of all possible sources of uncertainty.

In comparison to the traditional GP prior in Kennedy and O’Hagan (2001), the proposed approach adds to the flexibility of the computer model framework in multiple ways: it can be used to model non-Gaussian output processes, can accommodate continuous and discrete inputs, uses the information on the nature of randomness of the input to construct the functional form of the emulator and it is robust with respect to computation. Basis function-based emulators have a natural computational advantage over a GP because the former uses a linear mean structure to capture the input-output relationship in the data whereas the latter uses its covariance matrix, much more difficult to handle with large sample size (Kaufman et al., 2011). There is another aspect of the model where this article adds to the literature. Previous work on gPC-based uncertainty quantification was mostly based
on use of a fixed-dimensional, truncated version of the series but, in Section 4, we present a reversible jump Markov chain Monte Carlo (RJMCMC; Richardson and Green, 1997) sampler to adaptively select the constituent polynomials. This eliminates the need of using any subjective truncation allowing the data to control sparsity and order of the expansion, enabling us to quantify the uncertainty due to use of a finite dimensional gPC. Finally, for computer model emulation, the GP acts as a prior on the function space for the simulator response surface. The proposed approach can be considered in the same way. By proposing an oversized set of basis functions, we can control the complexity of the emulator by using a stochastic selection algorithm that can include or exclude a particular basis function based on its contribution to the likelihood of the observed data. The uncertainty in our model arises from the possible set of basis functions that represent the simulator, conditional on the observed data.

The paper is outlined as follows. In Section 2, we introduce the GP that is traditionally used in computer experiments. In Section 3, a new emulator is proposed and in Section 4 the variable dimension emulator is fully developed, along with an appropriate MCMC implementation. We demonstrate the performance of the proposed method on simulated examples (Section 5) and on a real-world application (Section 6). We conclude the paper with comments and future directions in Section 7.

2. Background: Emulation of computer models with GP

The traditional approach for using stochastic emulators to model the output from deterministic simulators was proposed by Sacks et al. (1989). Consider one such simulator with input \(x\) and output \(y\). The aim of the emulator is to predict the output at every possible input configuration, with associated estimates of uncertainty. Suppose that the computer model has been exercised only a finite number of times, say \(n\). At the run \(i = 1, 2, \ldots, n\), denote the input vector by \(x_i\) and the output by \(y_i\); call them training inputs. At any unsampled input \(x_0\), we want to know the likely values of the model output \(y_0\). Sacks et al. (1989) proposed using a random function, \(f\), to represent the input, so that \(y = f(x)\) and the covariance of
f is related to smoothness of the response. This is achieved by using a GP prior on f - at two different input configurations x and x’, the covariance between the functional values is specified using a stationary correlation function ρ, scale σδ and correlation parameter νδ such that \( \text{Cov}(f(x), f(x')) = \sigma^2_δ ρ(x, x'; νδ) \). Typically the function ρ is chosen to be the product of univariate correlation functions - one for each input. In this case, ρ has a separable stationary form, i.e., \( ρ(x, x'; νδ) = \prod_{j=1}^{p} ρ_i(x_j - x'_j; νδ(i)) \) when the input is p-dimensional. The one-dimensional correlation functions \( \{ρ_i(d; νδ(i)) : i = 1, 2, ..., p, \} \) are usually chosen of the form \( \exp(-νd^α) \), where α = 1 and α = 2 corresponds to the exponential and Gaussian correlation functions, respectively, with decay parameter ν.

A useful feature of GP emulators is the interpolation property. If we use \((x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\) to fit the random function with a GP prior, the estimated function exactly passes through \(y_i\) as \(x_i\) for each of \(i = 1, 2, ..., n\). There is zero uncertainty in predicting the output at already tried input configurations. The most important feature, however, is that for new new inputs \(x_0\), the uncertainty is quantified by the variance of \(f(x_0)|f(x_1), f(x_2), ..., f(x_n)\). In practice, we observe that as we move away from the training inputs, the prediction uncertainty increases - intuitively, one would expect this. We shall revisit this property in the context of the proposed approach in Section 5.2.

3. Emulating and calibrating computer models using basis functions

In this section, new methodology for emulating and calibrating computer models is proposed. To foreshadow, we will use a set of over-specified basis functions to represent the simulator response in place of the computer model itself. The approach uses only a sub-set of the basis functions and predictions are made by averaging over the posterior distribution of the collections of basis functions and associated model parameters.

The aim of model calibration is to combine field observations and simulator trials to estimate parameters that impact the physical process and also to form a predictive model for the physical system. There are two types of input variables in a calibration experiment: (i) calibration parameters that are unknown in the field, are to be estimated, but where a
value must be specified to run the computational model; and (ii) *design variables* that are adjustable or measurable in the physical system. Begin by considering the case where the input is only a vector of calibration parameters \( \zeta = (\zeta_1, \zeta_2, ..., \zeta_q)^T \) (i.e., no design variables).

Denote the physical system mean response by \( G_R(\zeta) \) and the computer model response as \( G(\zeta) \). Assume for the moment that the numerical model exactly describes the system dynamics and thus \( G_R \equiv G \). This assumption will be relaxed later in this section. In principle, if the value of \( \zeta \) were known in the physical system, the computer model could be run at this input value to acquire the process mean.

Suppose only a set of \( n \) evaluations of the simulator are available for the calibration exercise, with inputs \( \zeta_i^{(c)}, i = 1, \ldots, n \), chosen by an experimental design (e.g., Latin hypercube design (McKay et al., 1979)). To represent the computer model output at unsampled inputs, we propose to write \( G \) in terms of a collection of basis functions over the range of \( \zeta \). We specify the functional dependence between the simulator output \( y_i^{(c)} \) at the input \( \zeta_i^{(c)} \) as:

\[
y_i^{(c)} = \beta_0 + \sum_j \beta_j \Psi_j(\zeta_i^{(c)}).
\]

This specification implies that the collection of functions \( \{\Psi_j(\zeta)\} \) must be chosen in a way such that the true process \( G \) belongs to the span of the large set of functions, with certain regularity assumptions (i.e., in that case we can write \( G \) as a linear combination of \( \Psi_j \)'s where the coefficients are given by \( \{\beta_1, \beta_2, \ldots\} \)). In principle, infinitely many of the \( \Psi_j \)'s could be specified to model the response process, however, in practice this is not possible. Instead, we use only a finite number \( k \) of such functions. For example, if a response surface is exactly linear in its inputs, choosing \( \Psi_j \) as a first order polynomial in \( \zeta \) would suffice. Similarly, for quadratic relationship between \( y \) and \( \zeta \), using \( \Psi_j \)'s that include up to second order interactions in components of \( \zeta \) will be adequate. However, when the response function is a more complicated function of the inputs, using a larger set of basis functions is likely to be necessary to satisfactorily approximate the true response. We use white noise \( \epsilon \) to represent any high frequency variation that could be actually present in the response, but is unlikely to be detected by the model, and is not taken care of by the \( k \) basis functions we
have included in the model. The stochastic model is now:

\[ y_i^{(c)} = \beta_{k0} + \sum_{j=1}^{k} \beta_{kj} \Psi_j(\zeta_i^{(c)}) + \epsilon_i^{(c)}, \quad i = 1, 2, ..., n, \]

\[ \epsilon_i^{(c)} \sim N(0, \sigma^2). \] (3.1)

where \( \beta_k = (\beta_{k0}, \beta_{k1}, \cdots, \beta_{kk})^T \) is the vector of \((k+1)\) regression coefficients. In particular, \( \beta_{kj} \), for \( j \leq k \), stands for the coefficient of \( j \)-th basis function when \( k \) basis functions are used to model the response. The variance of the residual effect is represented by \( \sigma^2 \). We assign conjugate prior distributions - Gaussian and Inverse-Gamma, respectively - to these location and scale parameters, respectively. We assume a priori that \( \{\zeta_i\} \) are independent random variables, so if \( \mathcal{F}_j \) denotes the range of \( \zeta_j \), \( j = 1, 2, ..., q \), then the prior for \( \zeta_i \), defined over the q-dimensional hypercube \( \mathcal{F} = \otimes \mathcal{F}_j \), is of the form \( \pi(\zeta) = \prod \pi_i(\zeta_i) \). For model calibration in this setting, we want to estimate the real-world value \( \zeta \). To infer this, we have a vector of physical observations, \( y^{(o)} \). Learning \( \zeta \) from \( y^{(o)} \) can be done through the posterior predictive distribution of \( \zeta \) which is determined by:

\[
\pi_k(\zeta | y^{(o)}, y_{1:n}^{(c)}) = \pi(\zeta | y^{(o)}, y_{1:n}^{(c)}, k) \propto \pi(y^{(o)} | \zeta, y_{1:n}^{(c)}, k) \pi(\zeta) \\
\propto \left[ \int_{\beta_k, \sigma^2} \pi(y^{(o)} | \zeta, \beta_k, \sigma^2, k) \pi(\beta_k, \sigma^2 | y_{1:n}^{(c)}, k) d\beta_k d\sigma^2 \right] \pi(\zeta). \] (3.2)

We now relax the assumptions in the beginning of the Section. First, the computer model output most often does not exactly represent the system mean response and thus we need to allow for discrepancy between the experiment and the simulation. This can be done by embedding the proposed model within the well-established framework of Kennedy and O’Hagan (2001). Here, we connect the model for simulation with the field measurements. Estimation and inference from this model leads to comprehensive uncertainty quantification. Second, we also consider the more practical scenario where design inputs that are explicitly controlled, or measured, by the experimenters are included. Let \( \mathbf{x} \) stand for a \( p \)-dimensional design variables from an input space \( \mathcal{X} \). Multiple field experiments are carried out at different values of \( \mathbf{x} \), but \( \zeta \) is same (and unknown) for all of them. For the real-world application in Section
6, gas pressure and laser energy represent the design inputs but intrinsic characteristics of the experimental materials constitute $\zeta$. Now, $X \times F$ denotes the support for $(x, \zeta)$. We use $x_i^{(r)}$ and $x_i^{(c)}$ to denote values of $x$ at $i$-th experiment and simulation, respectively. If $m$ field observations are available, the joint model for $y_i^{(c)}$ and $y_i^{(o)}$ is specified as:

$$
y_i^{(c)} = f(x_i^{(c)}, \zeta_i^{(c)}) + \epsilon_i^{(c)}, i = 1, 2, ..., n,$$
$$y_i^{(o)} = f(x_i^{(r)}, \zeta) + \delta(x_i^{(r)}) + \epsilon_i^{(r)}, i = 1, 2, ..., m.
$$ (3.3)

Here, $f$ is the emulator mean, $\delta(\cdot)$ is the “discrepancy process” that accounts for the systematic difference between the mean of simulator and field observations and $\epsilon_i^{(r)}$ is the observation error. We specify the prior distributions as follows:

$$
\epsilon_i^{(c)} \sim N(0, \sigma^2), \delta(x_i^{(r)}) \sim GP(0_m, C(\sigma_5^2, \nu_5)), \epsilon_i^{(r)} \sim N(0, \sigma_r^2), \beta_k | k \sim N(0, \sigma_\beta^2 I_k).
$$ (3.4)

where $C(\sigma_5^2, \nu_5)$ can be specified as in Section 2. The prior distribution for $\zeta$ is decided based on the prior scientific knowledge (and previous studies, if available) about the range of input $\zeta$.

We propose to model the joint signal shared by the field observations and simulator outputs, $f$, with a basis expansion:

$$f(\cdot) = \mu(\cdot) + \sum_{j=1}^{k} \beta_j \Psi_j,$$

where $\mu$ is the trend function, typically chosen as a constant in computer experiments (Sacks et al., 1989). As before, $\{\Psi_j\}$ is the family of functions in $x$ and $\zeta$. An important aspect of this article is to choose an appropriate class of $\{\Psi_j\}$, which we shall discuss in the following subsection.

3.1. Construction of an optimal collection of basis functions

Consider a system with a single calibration input $\zeta$ and output $y$. Let $\pi(\cdot)$ be the prior distribution for $\zeta$ over its support $F$. The true computer model can be written as a function of the random input: $y = G(\zeta)$. Under the assumption of finiteness for all moments of $\pi$
there exists (Kubo et al., 2007) a sequence of orthonormal polynomials \( \{ \psi_i \}_{i=0}^{\infty} \), such that \( \psi_i \in L^2(F, \pi) \) and
\[
\int_F \psi_i(\zeta) \psi_i'(\zeta) \pi(\zeta) \, d\zeta = \delta_{i,i'},
\]
where \( \delta_{i,i'} = 1 \) if \( i = i' \), 0 otherwise. If the stochastic process \( G \) is assumed to have finite variance, then it follows from Ernst et al. (2012) that \( G \) can be approximated in the \( L_2 \) sense (mean-square) as a linear combination of polynomial functionals \( \{ \psi_i(\zeta) \} \), i.e., there exists \( \beta^0 = \{ \beta^0_i \} \) so that for \( G_k(\zeta) = \sum_{i=0}^{k} \beta^0_i \psi_i(\zeta) \),
\[
\int_F |G(\zeta) - G_k(\zeta)|^2 \pi(\zeta) \, d\zeta \to 0 \text{ as } k \to \infty.
\]
Here, \( G_k \) is called the \( k \)th degree gPC expansion of \( G \) and \( \beta^0 \) is the sequence of polynomial coefficients in the expansion. Using orthonormality of \( \psi_i \)'s we have \( \beta^0_i = \int G(\zeta) G_i(\zeta) \pi(\zeta) \, d\zeta \).

Originally proposed by Wiener (1938), the Hermite polynomial expansion for Gaussian inputs was shown to have \( L_2 \) convergence as above with exponential rate. But, when the true process was non-Gaussian, Xiu and Karniadakis (2003) observed that, for some choices of input distributions, using Hermite expansion still ensures convergence but at a much slower rate. In such cases, they proposed using members from the Askey scheme of hypergeometric orthogonal polynomials (Askey and Wilson, 1985). Each type of polynomials from the Askey scheme form a basis in the Hilbert space determined by their corresponding support so the set of such polynomials converge to any \( L_2 \) functional in the corresponding Hilbert functional space (see Ernst et al. (2012) for relevant theoretical results). Also, orthonormality of polynomials with respect to the input measure implies (Marzouk and Xiu, 2009) that the approximation converges at a rate depending on the smoothness \( \alpha \) of \( G \) so that for some constant \( C \), we have:
\[
\int_F |G(\zeta) - G_k(\zeta)|^2 \pi(\zeta) \, d\zeta \leq Ck^{-\alpha}.
\] (3.6)

If \( \alpha > 1 \), we can show almost sure convergence of \( G_k \) towards \( G \) because of the following lemma. In other words, the lemma implies that when convergence in second moment occurs
at a sufficient fast rate, almost sure convergence also holds. The proof is included in the supplementary document.

Lemma 1. Let $\zeta \sim \pi$ and $f_k(\zeta)$ be a sequence of functions converging in $L_2$ to $f(\zeta)$ at a rate of $k^{-\alpha}$, i.e., $\int |f_k(\zeta) - f(\zeta)|^2 \pi(\zeta)d\zeta < Ck^{-\alpha}$, for some $C > 0$ and $\alpha > 1$. It follows that $f_k(\zeta) \to f(\zeta)$ almost surely with respect to $\pi$.

In calibration experiments, we have a prior knowledge of the stochastic properties (range and likely values) of the calibration parameter. For some commonly used distributions, continuous as well as discrete, optimal choice of orthonormal sequence of polynomials are mentioned in Table 1 (Xiu and Karniadakis, 2002).

<table>
<thead>
<tr>
<th>Input type</th>
<th>Input range</th>
<th>Input Distribution</th>
<th>Orthogonal polynomials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>$(\infty, \infty)$</td>
<td>Gaussian</td>
<td>Hermite</td>
</tr>
<tr>
<td></td>
<td>$(0, \infty)$</td>
<td>Gamma</td>
<td>Lagurre</td>
</tr>
<tr>
<td></td>
<td>$[a, b]$</td>
<td>Beta</td>
<td>Jacobi</td>
</tr>
<tr>
<td></td>
<td>$[a, b]$</td>
<td>Uniform</td>
<td>Legendre</td>
</tr>
<tr>
<td>Discrete</td>
<td>${0, 1, 2, \ldots}$</td>
<td>Poisson</td>
<td>Charlier</td>
</tr>
<tr>
<td></td>
<td>${0, 1, 2, \ldots, N}$</td>
<td>Binomial</td>
<td>Krawtchouk</td>
</tr>
<tr>
<td></td>
<td>${0, 1, 2, \ldots}$</td>
<td>Negative Binomial</td>
<td>Meixner</td>
</tr>
<tr>
<td></td>
<td>${0, 1, 2, \ldots, N}$</td>
<td>Hypergeometric</td>
<td>Hahn</td>
</tr>
</tbody>
</table>

For the $q$-dimensional input $\zeta$ in Section 3, we consider a tensor product of univariate polynomials, i.e., we define a sequence of functions $\Psi_i(\zeta) = \prod_{j=1}^q \psi_{ij}(\zeta_j)$, $i = 0, 1, 2, \ldots$. Here $\{\psi_{ij}\}$ represents the sequence of orthonormal polynomials with respect to the prior distribution $\pi_j(\cdot)$ of $\zeta_j$. Since we have assumed independence between coordinates of $\zeta$, orthonormality of $\{\psi_{ij}\}$ with respect to $\pi_j(\zeta_j)$ implies orthonormality for $\{\Psi_i\}$ with respect to the product measure $\pi$. Eldred et al. (2008) argue that this assumption can be justified if one thinks of $\zeta$ as a transformation of the original input such that each coordinate of $\zeta$ represents a distinct source of randomness. The same assumption of a priori independence among components of $\zeta$ is also used in this article.

Bayesian inference with gPC has been discussed in the context of collocation (Marzouk...
and Xiu, 2009). However, the approach therein results in only a partial uncertainty quantification because, in evaluating the posterior distribution of the unknown input in the field data, the expansion coefficients are approximated with a set of plug-in estimates. This underestimates any uncertainty in determining the coefficients and sparsity of the expansion. The hierarchical model developed in current section, coupled with the adaptive estimation scheme from Section 4, allows for a comprehensive uncertainty quantification.

3.2. Further discussion of the model

Returning to the model in (3.3), we find it useful to add a graphical representation of the dependence among the variables in (3.3) in Figure 1.

For identifiability purposes, it is important that the function $\delta$ is defined over the space of design variables. This implies, $G_R(x, \zeta) - G(x, \zeta)$ can only be a function of $x$. This assumption is reasonable, because the discrepancy function can only be explored by comparing field
observations with the code (or an emulator thereof). Finally, \( \zeta \) being unknown there, it is not feasible to separately identify the function \( \delta \), and \( \zeta \).

We conclude this section with a discussion of the implications of using the proposed model in the context of deterministic simulators. The GP is used as an emulator largely because of its ability to interpolate the outputs and because it provides a foundation for statistical inference for deterministic computer models. In the first case, it has been argued that the interpolation property can be relaxed (Gramacy and Lee, 2012). We argue that we do not need the model to interpolate since we know the computer model response for the suite of \( n \) computer model trials, and we really require the emulator to do a good job of prediction at unsampled inputs. The case of statistical inference is more crucial. For the GP, prediction uncertainty at new inputs comes from the possible sample paths the random function can take, conditional on the simulator outputs and correlation function. The proposed emulator also provides a foundation for uncertainty quantification. Here, the sources of predictive uncertainty, conditional on the training data, are (i) selecting an appropriate set of \( k \) polynomials and (ii) the coefficients of the polynomials; and (iii) high frequency variation that we are unlikely to observe due to the sparseness of the sampling of the input space. It should be noted that, gPC is inherently different from a GP with polynomial mean function since, for the former, the form of the polynomial is chosen depending on the type of stochasticity in the input to achieve optimality, whereas in the latter it is user-driven. Also, when a large number of runs are collected from the simulator, using a GP prior gets increasingly expensive due to the associated high-dimensional covariance matrix. As we shall see later, the gPC-based specification offers a simpler and less computationally expensive option. Finally, if we marginalize out \( \{\beta_j\} \), this results in a nonstationary input-dependent covariance model for \( y^{(c)} \) as follows:

\[
\text{Cov}(y^{(c)}_i, y^{(c)}_{i'}) = \sigma^2 1_{i=i'} + \sigma^2 \beta \sum_{j=1}^{k} \Psi_j(x^{(c)}_i, \zeta^{(c)}_i) \Psi_j(x^{(c)}_{i'}, \zeta^{(c)}_{i'}). 
\]

Further flexibility in the nonstationarity structure can be ensured if we let the data choose the polynomials \( \Psi_j \) in an adaptive way allowing for differential treatment for different inputs.
based on their influence on the response. We do exactly that in Section 4.

4. Adaptive selection of chaos terms

In Section 3, much of the uncertainty for the emulator of the mathematical model output has been accounted for. However, the proposed approach still requires a choice for the fixed number of polynomial terms, $k$. Letting $k$ be a parameter to be estimated itself fulfills the goal of a complete uncertainty quantification. To do so, we propose an adaptive scheme that has two general objectives: (i) eliminating the need to use a truncated model and subsequent comparison between nested specifications; (ii) allowing the model to include/delete individual polynomial terms similar to a variable selection method, encouraging sparsity whenever possible. To achieve these aims, the RJMCMC sampler of Richardson and Green (1997) is used to switch between different sets of constituent polynomials (with possibly different sizes). RJMCMC has previously been proposed as a tool for Bayesian variable and model selection problems (e.g., Dellaportas et al. (2002)). It implies from (ii) that, instead of selecting a truncation bound and considering all polynomials within that bound, it is more efficient to let the model choose the most important terms from a much larger pool of polynomials. A priori, one needs to decide on two features: which subset of polynomials to choose from and what is the maximum number of terms that is allowed to enter in the chaos expansion at a time.

There is an immediate theoretical advantage of treating $k$ as a parameter. When we define a hierarchical model, we specify a probability distribution for the observations using parameters and/or processes and then assign prior distributions to those. However, the true system need not resemble the structure of the model. So, it is desirable that the true system response surface lies within the prior support for the model, i.e., there exists a set of realizations of these parameters and processes such that the probability distribution of the observations under the hierarchical model at those values is ‘close’ to the probability distribution under the true data-generating mechanism. For the system at hand, let $(G_0, \sigma^2_0)$ and $(\zeta, G_{R0}, \sigma^2_{r0})$ be the true processes and parameters for the simulation and experiment, respectively, such that
\[y(c) = G_0(x(c), \zeta(c)) + N(0, \sigma_0^2) \text{ and } y(r) = G_{R0}(x(r), \zeta) + N(0, \sigma_{R0}^2).\]

Then, the combined data, denoted by \(Y(f) = \begin{bmatrix} y_{1:m}^{(c)} \\ y_{1:n}^{(r)} \end{bmatrix}\), has a joint distribution that is actually determined by \(\Delta_0 = (G_0, \sigma_0^2, \zeta, G_{R0}, \sigma_{R0}^2)\). But under the proposed model in (3.3), \(\Theta = (\sigma^2, \sigma_r^2, \delta, k, \zeta, \beta_k)\) controls the distribution of \(Y(f)\); so it is important to investigate whether for any specific realization of \(\Delta_0\), there exists a subset in the domain of \(\Theta\) such that \([Y(f)|\Delta_0]\) and \([Y(f)|\Theta]\) are sufficiently close according to some distance measure. Theorem 1 asserts that they are indeed close under some assumptions on the true functions. Its proof is included in the supplementary document.

**Theorem 1.** *(Prior Support)* Assume that

(A1) \(G_0\) has finite variance and is of smoothness \(\alpha > 1\).

(A2) \(X\) is a bounded subset of \(\mathbb{R}^p\), \(F_0(x) = G_{R0}(x, \zeta) - G_0(x, \zeta)\) is a continuous function of \(x\) only and free of \(\zeta\).

(A3) The correlation function of discrepancy process is chosen as \(\rho(x_1, x_2; \nu_\delta) = \prod_{i=1}^p \rho_i(x_{1i} - x_{2i}; \nu_\delta(i))\), where \(\rho_i\) is a nowhere zero continuous, symmetric density function on \(\mathbb{R}\).

Then,
given \(\epsilon > 0\), there exists a set \(D\) such that \(\Theta \in D \implies KL([Y(f)|\Delta_0] || [Y(f)|\Theta]) < \epsilon\), where \(KL\) denotes the Kullback-Leibler distance metric between two probability distributions.

### 4.1. Key MCMC Moves

The implementation of a variable-dimension model in the above setup is now proposed. When \(k\) polynomial terms are included in the model, denote by \(\alpha_k = \{\psi_{ij} : j = 1, 2, ..., p + q; i = 0, 1, ..., k\}\) as the set of univariate polynomials. Marginalizing out \(\beta\) and \(\sigma^2\), the probability distribution, \(p(Y(f)|k, \alpha_k, \ldots)\), can be written in closed form (see supplementary document).

We note that this form of marginal likelihood is convenient for computational purposes as it does not involve \(n \times n\) matrix inversion like the GP. Using a suitable proposal distribution, \(q\), propose a dimension changing move \((k, \alpha_k) \rightarrow (k', \alpha_{k'})\). We consider three types of possible moves (i) **birth**: addition of a basis function, (ii) **death**: deletion of an existing basis function.
and (iii) change : modification of an existing basis function. Thus \( k' \in \{ k-1, k, k+1 \} \). The acceptance ratio for such a move is given by

\[
p_{k \to k'} = \min \left\{ 1, \frac{p(Y^{(f)} | k', \alpha_{k'}, \ldots) p(\alpha_{k'} | k') q((k', \alpha_{k'}) \to (k, \alpha_k))}{p(Y^{(f)} | k, \alpha_k, \ldots) p(\alpha_k | k) q((k, \alpha_k) \to (k', \alpha_{k'}))} \right\}.
\]

We specify a prior for \((k, \alpha_k)\) in the form of \( p(\alpha_k | k) \). A threshold \( k_0 \) is fixed for maximum number of polynomial terms in the model and \((k-1)\) has a Poisson(\(\lambda\)) prior truncated to the right at \( k_0 \). For \( \alpha_k \), one only need specify the order of each \( \psi_{ij} \), say \( h_{ij} \in \{0\} \cup \mathbb{N} \). As mentioned above, we restrict the choice of \( h_{ij} \) to a selected subset of functions \( T_{r,s}, 1 \leq r \leq s \) as follows:

\[
T_{r,s} = \{ \{h_{ij}\}_{j=1}^{p+q} : 0 \leq h_{ij} \leq r, \sum_j h_{ij} \leq s \}. \tag{4.1}
\]

The idea behind specifying \( T_{r,s} \) is to only include main effects of up to order \( r \) and interactions of up to order \( s \). Hence \( s \) denotes the maximum complexity a selected \( \Psi_i \) can have.

When we have a single input \( \zeta \), \( r = s \). We can control the overfitting through \( k_0, \lambda, r \) and \( s \).

Let \( |T_{r,s}| \) denote the number of elements in the set \( T_{r,s} \). For \( k \)-many basis functions present in the gPC expansion, excluding the constant term the remaining \( k-1 \) terms can be chosen in \( \binom{|T_{r,s}|}{k-1} \) different ways. Hence we have,

\[
p(\alpha_k | k) \propto (|T_{r,s}| - k + 1)! (k-1)! . \tag{4.2}
\]

Next, we specify the proposal distribution \( q(\cdot, \cdot) \) for each of the three moves as follows:

(i) First decide on the type of move to be proposed with probabilities \( b_k \) (birth), \( d_k \) (death) and \( c_k \) (change), \( b_k + d_k + c_k = 1 \). We put \( d_k = 0, c_k = 0 \) if \( k = 1 \), \( b_k = 0 \) if \( k = k_0 \).

(ii) For a birth move, choose \( \{h_{k+1,j}\}_{j=1}^{p+q} \) from \( T_{r,s} \) excluding the existing \((k-1)\) basis functions \( \Psi_h, h = 1, 2, \ldots, k-1 \). That can be done with probability \( \frac{1}{|T_{r,s}| - k + 1} \).

(iii) The death move is performed by randomly removing one of the \( k-1 \) existing basis functions (excluding the constant basis function).

(iv) A change move consists of choosing an existing nonconstant basis function randomly and alter the order of polynomials present in it.
From above, we have

\[ q((k, \alpha_k) \rightarrow (k', \alpha_{k'})) = \begin{cases} 
  b_{k} \frac{1}{|T_{r,s}| - k + 1} & k' = k + 1, \\
  d_{k} \frac{1}{k - 1} & k' = k - 1, \\
  c_{k} \frac{1}{|T_{r,s}| - k + 1} & k' = k.
\end{cases} \]

The acceptance ratios for different types of move can be worked out from this. It can also be verified that, in the final form for \( p_{k \rightarrow k'} \), \( |T_{r,s}| \) cancels out from the prior and proposal parts. This implies, one does not need to enumerate \( T_{r,s} \) and only requires a sampling strategy to draw from it. Hence, depending on practical considerations, different choices for \( T_{r,s} \) can be implemented in the current modeling framework.

Set \( k = k', \alpha_k = \alpha_{k'} \) if the move is accepted, leave unchanged otherwise. Subsequently, \( \beta_k \) can be updated using the \( k \)-variate \( t \) distribution with degrees of freedom \( d = n + m + 2a_\sigma \), mean \( \mu_k \), dispersion \( \frac{a_k \Sigma_k}{d} \), whose expressions are given (with derivation) in the supplementary document.

Finally, we need to choose a prior for \( \zeta \). Suppose \( i \)-th component of \( \zeta \) is assigned a prior \( \tilde{\pi}_i(\cdot) \). For the convenience of sampling, discretize its mass on the set of points \( \zeta^0_i = \{ \zeta^0_{l,i} : l = 1, 2, ..., g_i \} \), \( i = 1, 2, ..., m \), so the posterior can be sampled using an inverse cdf method. In the \( m \) dimensional space, the total number of such points is \( \prod_{i}^{m} g_i \). To avoid the burden of computing many probabilities at each iteration, we use conditional distributions one at a time i.e. \( [\zeta_i | \zeta^{(r)}_{-i},...] \), which requires evaluation of \( \sum_{i}^{m} g_i \) many expressions. The corresponding posterior probability at a point \( \zeta^{(0)}_{l,i} \) is given by \( \pi_{l,i} \propto \tilde{\pi}_{l,i} \exp[-Q^TQ/(2\sigma_r^2)] \), where \( Q = y^{(o)}_{1:m} - z_{1:m} - \sum_{h}^{k} \beta_{h} \Psi_{h} [x^{(r)}_{1:m} \zeta^{T} \otimes 1_{m}] \), \( \zeta_i = \zeta^{(0)}_{l,i} \). Alternatively, one can use Metropolis-Hastings sampling algorithm properly tuning it to achieve a desired acceptance rate. Conditional on the values of \( k \) and \( \zeta \), the hierarchical model in (3.3) and (3.4) has a Gaussian structure; hence the location and scale parameters therein can be updated during the MCMC using standard conditional distributions.
4.1.1. Using non-uniform priors for polynomial selection

A key innovation of the proposed approach is the ability to focus on simpler models a priori. The reversible jump step for selecting polynomials \( \alpha_k = \{ \psi_{ij} : j = 1, 2, \ldots, p+q; i = 0, 1, \ldots, k \} \) involves a prior distribution in form of \( p(\alpha_k | k)p(k) \). So far, we have used a truncated Poisson prior for \( k \) and, conditional on \( k \) terms in the expansion, we put a uniform prior in (4.2) on every possible selection of \( k \) terms from \( T_{r,s} \). However, we may emphasize simpler models by putting greater probability, a priori, on lower order polynomials than one consisting of higher order terms. This can be achieved by a modification of our reversible jump parameters, as we show below.

Consider, for example, a set-up where we have a scalar input \( \zeta \). As before, we only allow at most \( k_0 \) many \( \psi_i \) terms in the model at one iteration. The order of each candidate \( \psi_i \), say \( h_i \) is restricted to be at most \( r \), i.e., \( T_r = \{ h_i : 1 \leq h_i \leq r \} \) and \( |T_r| = r \).

The prior distribution is constructed as follows: for any polynomial \( \psi_l \) of degree \( l \), consider an weight function \( w_l = \exp(-\theta a(l)) \), where \( \theta > 0 \) is a constant and \( a(l) \) is an positive increasing function of \( l \) on \( \mathbb{N} \). We shall talk about choice of \( \theta \) and \( a(l) \) later. For a given \( k \), a typical realization of \( \alpha_k \) consists of \( k \) polynomials \( (\psi_{i(1)}, \psi_{i(2)}, \ldots, \psi_{i(k)}) \). We specify a prior on \( \alpha_k | k \) and \( k \) as:

\[
p(\alpha_k | k) \propto \prod_{l=1}^{k} w_{i(l)} = \exp \left[ -\theta \sum_{l=1}^{k} a(i(l)) \right] \\
p(k) \propto \sum_{\alpha_k} p(\alpha_k | k) 1(k \leq k_0)
\]

(4.3)

The motivation for this prior specification is borrowed from Swersky et al. (2012), where the authors discussed putting probability distribution on subsets of a set of indices. Indeed, this specification is also in a similar probability spirit to the effect hierarchy principle (i.e. lower order terms are more likely to active than higher order terms) used in industrial experiments (Wu and Hamada, 2009). So, we focus on simpler models a priori. In general, the normalizing constants of \( p(\alpha_k | k) \) and \( p(k) \) in (4.3) are difficult to evaluate, particularly for large \( k_0 \), since that requires evaluating all possible choices of \( \alpha_k \) out of \( k \) many polynomials and then
repeating that computation for all possible values of $k$ up to $k_0$. However, we are going to use this specification only inside the reversible jump step. So, we need to evaluate the ratio of prior probabilities of the form $\frac{p(\alpha_{k'}|k')p(k')}{p(\alpha_k|k)p(k)}$, which cancels out all normalizing constants from the prior.

We conclude this discussion by interpreting the role of $\theta$ and choice of $a(l)$. Given the form of $a(l)$, $\theta$ controls the rate of decay (in log-scale) of the weight as the order of the polynomial increases. Uniformity of the prior distribution is achieved by putting $\theta = 0$, increasing $\theta$ implies smaller value of prior probability mass function at configurations involving higher-order polynomials. Choice of $a(l)$ controls how strongly we want the probabilities to decay with higher $k$. Using $a(l) = l$ provides a log-linear decay, whereas using $a(l) = \log(l + 1)$ reduces the decay rate as we move to higher order terms. In addition to $k_0$ and $T_{r,s}$, $\theta$ and $a(l)$ can provide further control on model parsimony. We have found that this prior specification of $(k, \alpha_k)$ works well in Section 5.2. Detailed analysis of the effect of different choices of $a(l)$ and possible ways to extend this to multivariate inputs can be explored in future research. These terms, along with the model specification in this section amount to a prior distribution on the function space for the emulator where the analysts can insert information about the degree of complexity and sparsity.

5. Simulation Studies

In this section, multiple simulation studies are conducted to highlight different properties of our model. The first example compares our approach with the usual GP-based emulators with respect to a variety of numerical models. The next study analyzes how predictive uncertainty of the gPC-based emulator varies over training and test input configurations. The third setting deals with posterior distribution of the calibration parameters. Finally, the use of the proposed for a dataset with a discrete input is considered. The real-world application that motivated this work is presented in Section 6.
5.1. Predictive Performance of the Emulator

We use synthetic datasets from four numerical models to validate the proposed approach. For each dataset, we compare the fit from our model to that of the GP based emulator. The first two models have a two-dimensional input \( \zeta = (\zeta_1, \zeta_2)^T \in [-1, 1]^2 \). The response functions are defined as:

\[
y(\zeta) = \exp(\zeta_1 + \zeta_2)
\]

\[
y(\zeta) = a_1 \cos(2\pi \zeta_1) + a_2 \zeta_1^2 + a_3 \zeta_1 \log(1 + \zeta_2).
\]

The next two models use a three-dimensional input \( \zeta = (\zeta_1, \zeta_2, \zeta_3)^T \in [-1, 1]^3 \). The response functions are:

\[
y(\zeta) = b_1 \zeta_1^2 \zeta_2 \zeta_3^2 + b_2 \log(1 + \zeta_1 \zeta_3),
\]

\[
y(\zeta) = c_1 \zeta_1^2 + c_2 \sqrt{1 + \zeta_3}.
\]

The four models described above cover a wide class of functions including logarithmic, exponential, trigonometric, irrational as well as interactions between inputs. They are also fairly smooth and thus the GP should do relatively well.

For each of these four models, we simulate between 400 to 700 observations and leave out 100 of them randomly as a test sets. We fit our model as well as a GP on the remaining points and predict the value of the response over the test set. For comparison purpose, we use constant trend functions in both methods. The GP has been fitted using the “tGP” R package (http://cran.us.r-project.org/) with a separable power exponential correlation function. The MCMC was run for 12,500 iterations, rejecting the first 20% draws and thinning the rest at every 5-th draw. We compare the performance of the proposed model to the GP using the following criteria: (i) absolute predictive bias: average difference between the actual response and its posterior predictive estimate for each point in the test set, (ii) predictive uncertainty: width of 90% posterior credible set for the response value at each point in the test set, (iii) empirical coverage: what proportion of the test responses are contained inside their
90% credible sets and (iv) runtime: how fast the models can be estimated using identical numbers of iterations. We replicate the entire experiment 30 times and, in Table 2 we provide a summary of the performance averaged over the replications:

<table>
<thead>
<tr>
<th>Simulation Details</th>
<th>Method</th>
<th>Abs. predictive bias</th>
<th>Predictive uncertainty</th>
<th>Empirical coverage by 90% credible sets</th>
<th>Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Data size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F1 300</td>
<td>gPC</td>
<td>$2.777 \times 10^{-4}$</td>
<td>0.065</td>
<td>0.981</td>
<td>$\sim 35$ sec</td>
</tr>
<tr>
<td></td>
<td>GP</td>
<td>$1.912 \times 10^{-3}$</td>
<td>0.388</td>
<td>1.00</td>
<td>$\sim 1.5$ min</td>
</tr>
<tr>
<td>F2 400</td>
<td>gPC</td>
<td>$4.818 \times 10^{-4}$</td>
<td>0.067</td>
<td>0.948</td>
<td>$\sim 35$ sec</td>
</tr>
<tr>
<td></td>
<td>GP</td>
<td>$4.290 \times 10^{-3}$</td>
<td>0.572</td>
<td>1.00</td>
<td>$\sim 2.5$ min</td>
</tr>
<tr>
<td>F3 500</td>
<td>gPC</td>
<td>$6.781 \times 10^{-3}$</td>
<td>0.114</td>
<td>0.933</td>
<td>$\sim 40$ sec</td>
</tr>
<tr>
<td></td>
<td>GP</td>
<td>$1.244 \times 10^{-2}$</td>
<td>0.942</td>
<td>1.00</td>
<td>$\sim 4$ min</td>
</tr>
<tr>
<td>F4 600</td>
<td>gPC</td>
<td>$9.281 \times 10^{-4}$</td>
<td>0.084</td>
<td>0.943</td>
<td>$\sim 40$ sec</td>
</tr>
<tr>
<td></td>
<td>GP</td>
<td>$3.500 \times 10^{-3}$</td>
<td>0.929</td>
<td>1.00</td>
<td>$\sim 7$ min</td>
</tr>
</tbody>
</table>

A quick glance at Table 2 reveals better predictive results under gPC for bias, predictive uncertainty as well as runtime - though both approaches perform well. Particularly, we observe considerable reductions in average credible set width ranging from eighty to ninety percent under gPC compared to the one obtained with GP. Importantly, even with significantly shorter prediction ranges, the 90% credible sets have always covered more than 90% of the test responses which is desirable. GP has produced much wider prediction intervals leading to hundred percent coverage in all cases. From the computational perspective, estimation using a gPC-based emulator in these examples has been observed to be about forty to ninety percent faster than a GP emulator. A closer look at those numbers reveals that, as the size of the training dataset increases, the relative savings in runtime also goes up rapidly. It is worth mentioning that the datasets we considered in this simulation study are of moderate size. Hence, with a large dataset of thousands points, the computational ad-
vantage of the basis-function-based emulators over GP will be practically beneficial. This is particularly important for applications where there are moderate to large sample sizes. While the computer models themselves can computationally challenging, experimenters may have moderate to large samples because of super-computing resources (Goh et al., 2013; Kaufman et al., 2011).

5.1.1. Choice of the threshold $k_0$

The choice of $k_0$ (the maximum number of bases used at any iteration) is now investigated. More specifically, we want to know if we keep on increasing $k_0$, whether that would lead to including greater number of polynomial terms in the emulator. For this analysis, we simulate 500 observations from the model (F4) and fit it with 4 different values of $k_0 = 5, 20, 30, 45$. The posterior samples of $k$, the number of non-constant polynomial functions included in the model, are obtained under each value of $k_0$. The corresponding histograms are shown in Figure 2.

![Figure 2](image)

**Figure 2.** Variation in the posterior distribution of number of non-constant functions in the emulator under different choices of the bound $k_0$. The dataset has been simulated using the model in (F4).

We observe that, with $k_0 = 5$, the model puts maximum of its weight at models with 5 basis functions, indicating that even larger values are also likely. With $k_0 = 20$, the decaying right tail-behavior of the posterior distribution indicates that this range is potentially sufficient to capture most of the variability. We decided to increase $k_0$ further to see if this pattern changes when we allow more terms to be included. However, even with $k_0$ set at 30 and 45,
its behavior is essentially unchanged from $k_0 = 20$, that is, it continues to put maximum probability in the range of 4-8 basis functions. Overall, we do not have to be concerned too much about choosing a strict bound as $k_0$.

5.2. Effect of adaptive selection on predictive uncertainty

In this simulation, we want to explore some characteristics of the predictive uncertainty of the proposed approach. First, to emulate the response function well, it should be able to predict the output with low uncertainty at and in the vicinity of observed data points. The predictive variability should increase as we move away from the observed points. We also want to understand the effect of considering a larger pool of basis functions on predictive uncertainty: how the uncertainty changes if we fit an over-specified emulator to the data and whether our adaptive method can reduce that uncertainty by using the reversible jump step for selecting a subset of “necessary” polynomials.

For illustration purposes, we use the following one-dimensional numerical model:

$$y(\zeta) = \zeta^3 + \zeta \sin(3\pi\zeta),$$

where the model input $\zeta$ is chosen uniformly within $[-1,1]$. As our training dataset, we evaluate the function at 12 different values of $\zeta$. We use those 12 input-output pairs to estimate the response function over $[-1,1]$ and compare its predictive uncertainty (measured by the width of 90% posterior credible interval) over a grid. We deliberately choose to work with a small training set in order to magnify the uncertainty estimates so that any change in uncertainty due to location of unsampled inputs, as well as the use of over-specified emulator, can be easily visualized.

First, we fit a fixed-size gPC expansion to the training data using first 40, 50 and 60 Legendre polynomials, respectively. This involves fitting the full-models and no selection routine has not been employed. In Figure 3, we provide the posterior uncertainty estimates over the range of $\zeta$ and the number of basis functions. The uncertainty is so large, the approach has no utility.
Fig 3. Predictive uncertainty (width of 90% posterior credible intervals) of estimates of $y(\zeta)$ over the range of $\zeta$ for a fixed-size emulator consisting of first (left) 40 (middle) 50 and (right) 60 Legendre polynomials. The blue dashed line indicates the curve of the function $y(\zeta)$. The blue stars represent locations of the training data.

Now, we want to see, if the proposed selection procedure can significantly reduce the uncertainty of prediction compared to the ones in Figure 4. We again consider first 40, 50 and 60 Legendre polynomials, but this time, during the MCMC, we adaptively select the functions, i.e., we put a prior on the subsets of polynomials by using $\theta = 0.05$ and $a(l) = \log(l + 1)$ in (4.3). We can view the fixed-size model used to create Figure 3 as a threshold for this approach where all of the candidate basis functions gets included in the emulator. The uncertainty estimates obtained under this model are presented in Figure 4.

Fig 4. Predictive uncertainty (width of 90% posterior credible intervals) of estimates of $y(\zeta)$ over the range of $\zeta$ for a variable-size emulator chosen from first (left) 40 (middle) 50 and (right) 60 Legendre polynomials. The blue dashed line indicates the curve of the function $y(\zeta)$. The blue stars represent locations of the training data.
The foremost contrast between Figures 3 and 4 is their difference in y-scales. With overspecified emulators, we observed in Figure 3 an uncertainty range of ±150 at test points. In fact, when plotted in that wide range, the training points appear to imitate a constant response. However, the same set of functions, when filtered through our adaptive selection algorithm, could predict the response at new points with uncertainty ranging between ±10 in Figure 4, i.e., more than 90% reduction in uncertainty has been achieved. The uncertainty in Figure 3 arises from estimation of basis coefficients. The sources of uncertainty in Figure 4 are the selection of a subset of basis functions and estimation of the coefficients of selected functions.

For both figures, the uncertainty increases as we move from left to right. For Figure 3, this is due to increase in number of coefficients to be estimated whereas, in Figure 4, using a larger collection of functions to select, leads to greater uncertainty in selection - that is, there are more sets of basis functions to explain the observed data. For both of the figures, the uncertainty attains local minimums at observed input values. This simulation study highlights the effectiveness of our adaptive selection scheme in reducing the predictive uncertainty. In most practical applications, we are likely to have more training points compared to what we have used here so it is reasonable to expect more accurate predictive performance in those situations.

5.3. Posterior modality of the unknown input

We carry out another simulation study with a different objective. If $\zeta$ is the unknown value of the input which has produced an observation $y^{(o)}$, then can we always retrieve $\zeta$ from our model? To explore this, start with the following numerical model:

$$y(\zeta) = \frac{a}{b + (\zeta - c)^2},$$

where $a, b, c$ are constants and the model input $\zeta$ is uniformly chosen within $[-1, 1]$. Notice that, if $0 < c < 1$, $y$ is not a monotone function of $\zeta$, but it is monotone if $c < 0$ or $c > 1$.

Now, consider a field experiment with calibration input $\zeta$ which is governed by the above numerical model. The output from any experimental run is measured with a small Gaussian
noise of scale 0.8. Suppose, 10 runs of the field experiment are conducted under identical situation and the outputs are available to the user. The interest is to infer about the unknown value of the calibration input, say $\zeta_0$, for those runs. The user can access a simulator that generates $y$ at a chosen value of $\zeta$ using the above numerical model.

The inference proceeds as follows. First, the user evaluates the simulator at 197 values of $\zeta$ chosen uniformly within its range. Without any controlled input $x$, this problem resembles the structure of (3.1). So, the user can utilize (3.2) for the posterior predictive distribution of $\zeta_0$. (Notice that in (3.2) we had $k$ fixed, but we can now vary $k$ as in Section 4.1.)

We repeat the above simulations under two different values of $c$ at 1.5 and 0.08. We choose $\zeta_0 = 0.356$ in both cases. Notice that, with $c = 1.5$, $y$ is a strictly one-to-one function of $\zeta$. However, with $c = 0.08$, two values of $\zeta$ can produce identical output. The posterior densities of $\zeta_0$ for the two different choices of $c$ are presented below in Figure 5.

![Figure 5](image_url)

**Fig 5.** Posterior density estimates for $\zeta_0$ under (left) $c = 1.5$ and (right) $c = 0.08$. The blue dashed line corresponds to the true value of $\zeta_0 = 0.356$.

When $c = 1.5$, the posterior of $\zeta_0$ includes the true $\zeta_0$ within its high-probability region. However, with $c = 0.08$, $\zeta_0 = -0.196$ and 0.356 correspond to identical $y_0$. So, the posterior shows bimodality around those values of $\zeta_0$. Hence, the ability of the model to identify the actual value of the input depends on whether $G(\zeta)$ is an one-to-one function of $\zeta$. Similar
restriction on identifiability exists when $\zeta$ is a vector. (i.e., if there exists two input combinations $\zeta_{01} \neq \zeta_{02}$ such that $G(\zeta_{01})$ and $G(\zeta_{02})$ are close, then the posterior of $\zeta$ can not separate one from the other unless informative prior is used.) It depends on our objective whether this lack of identifiability is a concern. If we want an emulator for $G$ to predict the response, then this is not a problem at all because this does not impact the contribution of $\zeta$ in the emulator (quantified by $G(\zeta)$, which is identical at $\zeta_{01}$ and $\zeta_{02}$). But, if identifying the actual configuration $\zeta_0$ is crucial, then informative prior for $\zeta_0$ needs to be used.

5.4. Model with discrepancy and discrete input

We present a simulation study for computer model validation specifically in a scenario where GP based emulator can not be employed. We start with the following numerical model:

$$y = \beta_0 + \beta_1 x^2 + \beta_2 \log(1 + \zeta^2),$$

where we draw $x$ using a uniform distribution on $[-1, 1]$ and $\zeta$ from a Poisson(4) distribution. We simulate 270 observations from this model using different combinations of $x$ and $\zeta$. Next, 35 additional observations are generated with a "real life" discrepancy $\delta(x) = \beta_3 (1 + \sin(2\pi x))^2$ creeping into the above model. A measurement error with scale parameter 0.05 is also added to these "real" observations to match the setup of the shock experiment dataset in Section 6.

Because $\zeta$ takes discrete values, a Gaussian process prior on the input cannot be used. (See Zhou et al. (2011) for a direction on how to define a GP prior when there are qualitative outputs in the computer model.) However, we can still use the gPC representation in (3.5). The gPC is constructed using Legendre chaos for $x$ and Charlier chaos for $\zeta$. We use two different models - with and without a model for discrepancy function as in (3.3). For validation purpose, we randomly remove a set of 15 observations from the data and for each of them constructed the posterior mean, interval estimates and coverage proportion under the two models. In Figure 6, we present the predictive performance of the models.
Figure 6 illustrates the predictive efficiency of the model with discrepancy in terms of smaller bias and shorter uncertainty ranges without any reduction in coverage proportion. This is expected as a discrepancy function has indeed been introduced in the simulation step for the "real" data. If we do not use a model for $\delta(\cdot)$ when there is indeed a discrepancy between the real process and the numerical model, the scale parameter $\sigma_r$ of the error term $\epsilon^{(r)}$ gets large to account for the additional patterns in the field data. As a result, the prediction intervals get wider. In Table 3, we present the useful summary statistics.

### Table 3

Effect of accounting for discrepancy on predictive diagnostics

<table>
<thead>
<tr>
<th>Method</th>
<th>Prediction criteria</th>
<th>absolute bias</th>
<th>uncertainty</th>
<th>empirical coverage by 90% credible sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>gPC with discrepancy</td>
<td></td>
<td>0.260</td>
<td>2.602</td>
<td>1.00</td>
</tr>
<tr>
<td>gPC without discrepancy</td>
<td></td>
<td>2.434</td>
<td>7.804</td>
<td>0.80</td>
</tr>
</tbody>
</table>
6. Analysis of radiative shock experimental dataset

We now move on to the radiative shock experiments that play an important role in modern-day high-energy-density physics (HEDP) and astrophysical studies. The experimental research is connected to the question of how radiative shocks alter the emission that we see from supernovae, and how they affect the evolution of young supernova remnants (Chevalier, 1997). It is possible to generate and observe radiating shock waves in the laboratory that can be viewed as a scaled experiment for understanding astrophysical shock waves. In these experiments, a disk of beryllium (Be) is placed at the end of a xenon (Xe) filled tube. A laser is discharged onto the Be disk; the energy deposition of the laser causes a layer of Be to ablate thereby accelerating a shock. This shock travels through the Be and breaks-out of the disk into the Xe gas where radiography is able to capture images of the shock structure. These experiments are expensive in terms of cost, time and amount of planning required. Also, in addition to the controlled inputs of the experiment (Be disk thickness and the laser energy) there are other factors, which can potentially influence the outcome, but can not be measured (i.e., calibration parameters).

The underlying mathematical model (Drake, 2006) has been implemented as a computer model – an adaptive mesh refinement Eulerian radiation hydrodynamics code that computes the shock behavior. The subpart of the simulator considered in this article is a preprocessor: a two-dimensional Lagrangian radiation hydrodynamics code (HYADES), referred to as H2D, that computes the laser energy deposition and system evolution for the first nanosecond. We consider 4 inputs including the controlled inputs \( (x) \): Be disk thickness, laser energy and calibration inputs \( (\zeta) \): electron flux limiter (related to heat transport by the electrons) and Be gamma constant (related to the compressibility of Be). A previous analysis with 1D output, McClarren et al. (2011) found these inputs to have a significant effect on the output. An additional input, specific to the two-dimensional solution is also considered – the wall opacity of the plastic tube, which controls how strongly radiation is absorbed. A set of 104 runs of this preprocessor was conducted to cover this 5 dimensional input space. Additionally, we consider results from 8 laboratory experiments. The range of measurement
inaccuracy ($\pm 10$ picoseconds), added with another timing error of $\pm 50$ picoseconds for the laser firing mechanism, is converted into a zero mean Gaussian error, with scale determined using the $3\sigma$ criterion. All the 5 inputs in this problem are known to lie within bounded intervals with independent uniform prior distributions. The associated optimal choice, same for all inputs, is the Legendre chaos. Location-scale adjustments were applied to the input data to shift their values within $[-1, 1]$.

We first evaluate how gPC performs against the conventional GP-based model. For all subsequent analysis, we choose a linear trend function $\mu$ in $x$ and $\zeta$ and, for the class of polynomials $T_{r,s}$ in (4.1), we choose $r = 4, s = 8$. The covariance function $C(\cdot, \cdot)$ for the GP model was chosen to be stationary anisotropic with an exponential form separable in its arguments, i.e., $C(a, b) = \sigma^2 \exp(-\sum_{i=1}^{p+q} \nu_i |a_i - b_i|)$.

The inference proceeds as follows: we leave out 11 randomly chosen observations as our test sample, fit the model with the remaining points as training data, then construct point estimate and 90\% predictive interval for each point in the test sample. This completes one trial. Randomizing selection of the test samples, we perform 40 such trials, sufficient to cover all the simulator runs. Below, in Figure 7 and Table 4, we provide the summary measures of predictive performance for both the models - median absolute bias, predictive uncertainty (width of 90\% posterior credible set) and empirical coverage rates.

<table>
<thead>
<tr>
<th>Method</th>
<th>Prediction criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>absolute bias</td>
</tr>
<tr>
<td>GP</td>
<td>3.85597</td>
</tr>
<tr>
<td>gPC</td>
<td>2.9951</td>
</tr>
</tbody>
</table>

It is evident from all of the above diagnostics that use of the gPC representation contributes to smaller bias as well as significantly shorter predictive intervals without sacrificing the expected coverage rate for future predictions. So, we proceed to fit the simulator and experimental outputs simultaneously using the full hierarchical model from Section 4. There,
Fig 7. Model performance on (left) average absolute error and (right) width of 90% credible interval for prediction of test samples for the H2D output

the unknown input $\zeta$ has 3 components - Be gamma, wall opacity and. We report the empirical summaries of their posterior distributions below, in Table 5. The corresponding visual representation through posterior histograms is presented in Figure 8.

**Table 5**

*Posterior summaries for components of $\zeta$*

<table>
<thead>
<tr>
<th>Summary statistics</th>
<th>Be gamma</th>
<th>wall opacity</th>
<th>flux limiter</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>1.424</td>
<td>1.005</td>
<td>0.0579</td>
</tr>
<tr>
<td>90% credible interval</td>
<td>(1.400,1.461)</td>
<td>(0.727,1.275)</td>
<td>(0.0503,0.0714)</td>
</tr>
</tbody>
</table>

The extent of nonuniformity in the posterior reflects whether the data have strong enough evidence to identify the “correct” value that prevails in nature. Here, the posterior distributions of Be gamma in the above model looks to be the most informative, as it is concentrated only over a small subset of its prior range. Flux limiter is reasonably well constrained, but little is learned about the wall opacity.

It is of interest to know if the H2D numerical model is a good substitute of the actual experiment (adjusted for the measurement error). In Section 3, we have noted that the discrepancy process $\delta(x^{(r)})$ captures any potential discrepancy between the two systems. So,
we decide to fit the model with and without including the $\delta$ function. Here, we follow the leave-one-out validation procedure, i.e., at any time we remove all measurements that belong to a particular experiment. Models are fitted to remaining data (simulator and experimental outcomes together) and similar to the above, point- and interval-estimates for the leave-one-out data points are obtained from posterior draws, which were converted to estimates of bias and uncertainty as before. In Table 6, we present the diagnostics.

<table>
<thead>
<tr>
<th>Method</th>
<th>Prediction criteria</th>
<th>absolute bias</th>
<th>uncertainty</th>
</tr>
</thead>
<tbody>
<tr>
<td>gPC with discrepancy</td>
<td></td>
<td>22.1831</td>
<td>72.3235</td>
</tr>
<tr>
<td>gPC without discrepancy</td>
<td></td>
<td>19.2354</td>
<td>62.6619</td>
</tr>
</tbody>
</table>

Adding discrepancy term does not improve the bias and, as expected, it increases the uncertainty of prediction because it involves the additional uncertainty in estimation of the discrepancy process. Hence, we can conclude that the H2D code adequately represents the dynamics of an actual shock breakout experiment so, in future, it can be relied upon for estimating the shock breakout times in the explored experimental ranges instead of performing costly laboratory experiments.
7. Discussion

We have presented a full Bayesian approach to emulate and calibrate the parameters of a dynamical system using chaos-based expansions. This specification explicitly utilizes the stochastic information about the input to capture a wide range of output processes outside the Gaussian family. Even for a Gaussian setup, when the simulator response is not well represented by a stationary GP, a nonstationary covariance function is required for the GP emulator. However, the usual nonstationary covariance models are often complex and subject to computational difficulties. As we have shown in the end of Section 3, with an adaptive basis function expansion, if we integrate out the expansion coefficients, we can obtain (an input-dependent marginal nonstationary covariance pattern in the emulator. Conjugacy in the hierarchical specification enables us to perform model search efficiently. Priors can be specified encouraging sparsity in the expansion still allowing for inclusion of higher order terms whenever required. Inference on the emulator, a stochastic surrogate of the forward model, as well as the unknown inputs are drawn from a fully-model based posterior distribution that accounts for possible sources of uncertainty. We showed the tractability of our model using both univariate and multivariate examples. In summary, The present work attempts to introduce basis-function-dependent methods in the increasingly important domain of calibration and uncertainty quantification for physical experiments. We anticipate that the development of a fully model based approach along with efficient computational scheme would motivate further application in other fields of stochastic models as well.

It is worth noting that we can view the Gaussian process model as an infinite dimensional regression. For example, if we take the Karhunen-Loéve expansion (Loéve, 1978) of a Gaussian process $f$ with input $x$ then, with known covariance parameters, it can be written as: $f(x) = \sum_{k=1}^{\infty} w_k e_k(x)$ where the the basis functions $e_k(x)$ are orthogonal and the coefficients $\{w_k\}$ are independent, mean-zero normal random variables with variance of $w_k$ equal to the $k$-th largest eigenvalue $\lambda_k$ (assume they are arranged in decreasing order). From this, we can see that the variability in the emulator comes from the uncertainty in the coefficients after conditioning on the data. That is, the source of variability at unsampled inputs comes from
model uncertainty. It is a similar argument that we use to provide a foundation for statistical inference for the methodology we have proposed.

Acknowledgement

The authors acknowledge the contribution of Carolyn Kuranz, Michael Grosskopf, Erica Rutter, Ryan McClarren and Soma Dhavala.

References


