Journal de la Société Française de Statistique
Vol. 160 No. 2 (2019)

Taking into account input uncertainties in the Bayesian calibration of time-consuming simulators

Titre: Prise en compte des incertitudes sur les entrées de simulation pour le calage de codes numériques coûteux

Guillaume Perrin[1] and Cédric Durantin[1]

Abstract:

The consideration of experimental uncertainties is a key element in the quantification of uncertainties and prediction by simulation. While particular attention is paid to experimental uncertainties on simulation outputs, little work is done on uncertainties on simulation inputs, arguing that they are negligible or small enough to be aggregated with uncertainties on outputs via Taylor development. However, these uncertainties on inputs are not always low and, depending on the structure of the code, linearization around them is not always possible. The objective of this work is therefore twofold. First, it introduces a general Bayesian framework for integrating input uncertainties into the calibration of code parameters. It then proposes several approaches to effectively solve this inference problem, depending on the regularity of the code and the type of inputs considered. The advantages and disadvantages of the different methods are finally illustrated on an analytical example, as well as on a ballistic problem.

Résumé : La prise en compte des incertitudes expérimentales est un élément clé de la quantification des incertitudes et de la prévision par la simulation. Bien qu'une attention particulière soit accordée aux incertitudes expérimentales sur les sorties de simulation, peu de travaux s'intéressent aux incertitudes concernant les entrées de simulation, sous pretexte qu'elles sont négligeables ou suffisamment petites pour être agrégées avec les incertitudes sur les sorties par développement de Taylor. Toutefois, ces incertitudes sur les entrées ne sont pas toujours faibles et, selon la structure du code, la linéarisation autour de celles-ci n'est pas toujours possible. L'objectif de ce travail est donc double. Premièrement, il introduit un cadre bayésien général permettant l'intégration des incertitudes sur les entrées pour le calage de paramètres du code. Il propose ensuite plusieurs approches pour résoudre efficacement ce problème d'inférence, en fonction de la régularité du code et du type d'entrées considérées. Les avantages et les inconvénients des différentes méthodes sont finalement illustrés sur un exemple analytique, ainsi que sur un problème balistique.

Keywords: Bayesian framework, uncertainty quantification, statistical inference, input uncertainty, kernel density estimation
Mots-clés : calibration bayésienne, quantification des incertitudes, inférence statistique, méthode des noyaux
AMS 2000 subject classifications: 62F15, 62F86, 62G07, 60G15, 68T05

1. Introduction

Simulation currently plays a major role in the design, optimization and certification of complex systems. To predict the behavior of these systems, computer codes are introduced, which are generally based on two types of inputs. On the one hand, system parameters refer to the quantities used to define the system and its environment, such as the dimensions of the system, the materials of which it is made, the temperature, the pressure, etc. On the other hand, the calibration

[1] CEA/DAM/DIF, F-91297, Arpajon, France.
E-mail: guillaume.perrin2@cea.fr; cedric.durantin@cea.fr

 ISSN: 2102-6238

parameters correspond to the code constants that must be set for the code to be executed. These quantities are considered universal, in the sense that they are supposed not to depend on the parameters characterizing the system, and generally refer to physical law parameters or numerical thresholds.

For the code to be predictive, these calibration parameters must be identified from experimental data representative of the system's operating conditions. To make this estimate robust to potential experimental uncertainties, a Bayesian formalism is generally considered (Kennedy and O'Hagan (2001)), which consists of modeling calibration parameters by random quantities and seeking their posterior distribution according to available data.

Sampling methods such as Markov Chain Monte-Carlo (MCMC) approaches are generally considered to construct approximations of this latter distribution (see Rubinstein and Kroese (2008); Tian et al. (2016) for more details). To solve the inference problem, the code must therefore be evaluated a very large number of times (Berliner (2001); Kaipio and Somersalo (2004)), which can be prohibitive from a computational point of view when using simulators that take time. To overcome this problem, a classical approach is to build surrogate models (also called emulators) for real code to speed up the MCMC procedure. Such alternative models can be used to adapt the distribution of proposals, as done in Rasmussen (2003); Fielding et al. (2011); Conrad et al. (2016, 2018). Therefore, the number of code evaluations per MCMC step can be significantly reduced, while asymptotically sampling the exact posterior distribution. Alternatively, the real code can be directly replaced by its emulator in the MCMC procedure. In this case, we obtain samples associated with an approximation of the true posterior distribution since the probability has been modified, but the cost to solve the inference problem is usually much lower. More details on this second approach can be found in Marzouk and Najm (2009); Marzouk and Xiu (2009); Wan and Zabaras (2011); Li and Marzouk (2014); Tsilifis et al. (2017) when polynomial representations are considered for substitution models, in Kennedy and O'Hagan (2001); Santner et al. (2003); Higdon et al. (2008); Bilionis and Zabaras (2015); Sinsbeck and Nowak (2017); Damblin et al. (2013); Perrin (2019) when the substitution modeling is based on the Gaussian process regression, or in Higdon et al. (2003); Chen and Schwab (2015) when the substitution models correspond to code evaluations at different resolution levels.

In this work, we do not consider model discrepancy. This does not mean that the computer codes we are interested in are perfect representations of reality, but that once their calibration parameters have been correctly identified, the predictions they provide are sufficiently accurate for their intended uses. Therefore, the main source of uncertainty for the estimation of calibration parameters is experimental uncertainties, which can be divided into two groups: uncertainties on code inputs and uncertainties on code outputs. Whereas the output experimental uncertainties play a central role in the formerly cited papers, few words are said about experimental input uncertainties. Indeed, they are often considered negligible or small enough to be directly mixed with experimental output uncertainties through Taylor expansion. However, these experimental input uncertainties are not always low and it is not always possible to linearize the code around them.

Therefore, depending on the magnitude and complexity of the experimental input uncertainties, the objective of this paper is to present methods that can be used to effectively address input uncertainties when we are in one of the former problematic cases.

The outline of this work is as follows. Section 2 presents the theoretical framework for the

 ISSN: 2102-6238

Bayesian procedure we consider. Section 3 introduces several approximations that can be used to efficiently solve the inference problem when confronted to time-consuming simulators. At last, the advantages and the limits of the different approximations are discussed when analyzing two examples in Section 4.

2. Bayesian calibration

We are interested in predicting the behavior of a system $\mathscr{S}$ using a time-consuming computer code g, such that:

$$g : \left\{ \begin{array}{ccc} \mathbb{X} \times \mathbb{B} & \to & \mathbb{R} \\ (\mathbf{x}, \mathbf{b}) & \mapsto & g(\mathbf{x}; \mathbf{b}) \end{array} \right. . \tag{1}$$

Here, $\mathbf{x} \in \mathbb{X}$ gathers the N_x parameters that are used to characterize system $\mathscr{S}$, whereas $\mathbf{b} \in \mathbb{B}$ gathers the N_b calibration parameters, which are *a priori* unknown, but need to be specified for the code to be run. For the sake of simplicity, only real-valued computer codes are considered in this paper. The true value of $\mathbf{b}$, written β, is supposed to be a real-valued vector such that $\mathbb{B}$ is a subset of $\mathbb{R}^{N_b}$. On the contrary, there is *a priori* no restriction on the nature of the components of $\mathbf{x}$, which can be scalar or functions for instance.

The prior information about β is a density over $\mathbb{B}$, which is denoted by f_β. It is assumed to be known from expert judgement (Marin and Robert (2007)). To identify β, we have access to N measured values of the system output, $(y_n)_{1 \leq n \leq N}$, such that for all $1 \leq n \leq N$,

$$y_n = g(\mathbf{x}_n^{\text{true}}; \beta) + \varepsilon_n. \tag{2}$$

Here, $\varepsilon_n \in \mathbb{R}$ is the output measurement uncertainty, while $\mathbf{x}_n^{\text{true}}$ gathers the true values of the system parameters for the n^{th} measurement. This true value being unknown, it is modeled by a random quantity, whose prior mean is written $\mathbf{x}_n$, and whose random deviation towards this mean is denoted by $\zeta_n \in \mathbb{X}$:

$$\mathbf{x}_n^{\text{true}} = \mathbf{x}_n + \zeta_n, \quad 1 \leq n \leq N. \tag{3}$$

According to Eq. (2), the simulator is supposed to have no model discrepancy. As no computer code can virtually be said to be a perfect representation of reality, this only means that the predictions provided by the calibrated code are accurate enough (compared to the measurement error) for its intended use. In particular, it can exploit error compensation to maximize its predictive capabilities, and can therefore not necessarily perfectly fit reality. Denoting by $\mathbf{y} = (y_1, \ldots, y_N)$ the vector of observations, it comes:

$$\mathbf{y} = \mathbf{g}(\zeta; \beta) + \varepsilon, \tag{4}$$

with $\mathbf{g}(\zeta; \beta) = (g(\mathbf{x}_1 + \zeta_1; \beta), \ldots, g(\mathbf{x}_N + \zeta_N; \beta))$, $\zeta = (\zeta_1, \ldots, \zeta_N)$ and $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_N)$.

In this work, we moreover assume that:
– the mean values $\mathbf{x}_1, \ldots, \mathbf{x}_N$ are known,

 ISSN: 2102-6238

– β, ε and ζ are statistically independent,
– ε admits a **known** density, denoted by f_ε,
– it is possible to generate (at a negligible computational cost) independent realizations of ζ.

Using the Bayes' formula, we deduce that for all $\mathbf{b}$ in $\mathbb{B}$:

$$f_{\beta|\mathbf{y},\zeta}(\mathbf{b}) \propto f_\beta(\mathbf{b}) \times f_\varepsilon(\mathbf{y} - \mathbf{g}(\zeta;\mathbf{b})), \tag{5}$$

$$f_{\beta|\mathbf{y}}(\mathbf{b}) \propto f_\beta(\mathbf{b}) \times \mathbb{E}\left[f_\varepsilon(\mathbf{y} - \mathbf{g}(\zeta;\mathbf{b}))\right]. \tag{6}$$

In the general case, this posterior distribution is not explicit, and sampling procedures, such as a Monte Carlo (MC) within MCMC method, have to be used to generate samples that are approximately distributed according to $f_{\beta|\mathbf{y}}$. In that case, the number of code evaluations required to get one sample of $\beta|\mathbf{y}$ can be very high. When each code evaluation is time-consuming, additional assumptions and/or simplifications are needed, which will be detailed in the next section. In particular, the Gaussian case will rely on the following assumption.

Assumption 1. *ε and ζ are supposed to be centered and normally distributed:*

$$\varepsilon \sim \mathcal{N}(\mathbf{0}, \Sigma_\varepsilon), \quad \zeta \sim \mathcal{N}(\mathbf{0}, \Sigma_\zeta), \tag{7}$$

where Σ_ε and Σ_ζ are two symmetric positive-definite matrices.

3. Approximated methods

In this work, the following assumption is considered.

Assumption 2. *The experimental measurements are statistically independent: for all $1 \leq i \neq j \leq N$, (ζ_i, ε_i) is independent of (ζ_j, ε_j).*

Under this assumption, which is often verified in practice, we have:

$$\mathbb{E}\left[f_\varepsilon(\mathbf{y} - \mathbf{g}(\zeta;\mathbf{b}))\right] = \prod_{n=1}^{N} \mathbb{E}\left[f_{\varepsilon_n}(y_n - g(\mathbf{x}_n + \zeta_n;\mathbf{b}))\right], \tag{8}$$

where for all $1 \leq n \leq N$, f_{ε_n} is the density of the n^{th} component of ε. Hence, the evaluation of $f_{\beta|\mathbf{y}}$ in each $\mathbf{b} \in \mathbb{B}$ only requires the estimation of N integrals in dimension N_x, when an integral in dimension $N \times N_x$ was needed for the direct estimation of $\mathbb{E}\left[f_\varepsilon(\mathbf{y} - \mathbf{g}(\zeta;\mathbf{b}))\right]$:

$$f_{\beta|\mathbf{y}}(\mathbf{b}) \propto f_\beta(\mathbf{b}) \times \prod_{n=1}^{N} \mathbb{E}\left[f_{\varepsilon_n}(y_n - g(\mathbf{x}_n + \zeta_n;\mathbf{b}))\right]. \tag{9}$$

The following of this section presents different assumptions that can be introduced to allow efficient evaluations of $f_{\beta|\mathbf{y}}$ (up to a multiplicative constant) in each $\mathbf{b} \in \mathbb{B}$, which is a prerequisite for using MCMC approaches.

 ISSN: 2102-6238

3.1. Linear approximation of the code

Input experimental uncertainties are most of the time assumed to be small, such that it is possible to linearize the code output around them.

Assumption 3. *The computer code can be linearized around the input uncertainties, such that for all* $(n,\mathbf{b})$ *in* $\{1,\dots,N\}\times\mathbb{B}$,

$$g(\mathbf{x}_n+\zeta_n;\mathbf{b}) \approx g(\mathbf{x}_n;\mathbf{b}) + \delta^x(\mathbf{x}_n,\mathbf{b})^T\zeta_n + \varepsilon_n, \quad \delta^x(\mathbf{x}_n,\mathbf{b}) := \frac{\partial g}{\partial \mathbf{x}}(\mathbf{x}_n;\mathbf{b}). \tag{10}$$

Assumption 3 strongly reduces the number of code evaluations required for the estimation of $f_{\beta|\mathbf{y}}$. Indeed, for each $(\mathbf{b},n)$ in $\mathbb{B}\times\{1,\dots,N\}$, once quantities $g(\mathbf{x}_n;\mathbf{b})$ and $\delta^x(\mathbf{x}_n,\mathbf{b})$ have been computed, quantity $\mathbb{E}\left[f_{\varepsilon_n}(y_n - g(\mathbf{x}_n+\zeta_n;\mathbf{b}))\right]$ can be estimated at (almost) no additional cost. For instance, if $\zeta_n(\theta_1),\dots,\zeta_n(\theta_M)$ denote M independent realizations of ζ_n,

$$\mathbb{E}\left[f_{\varepsilon_n}(y_n - g(\mathbf{x}_n+\zeta_n;\mathbf{b}))\right] \approx \frac{1}{M}\sum_{m=1}^{M} f_{\varepsilon_n}\left(y_n - g(\mathbf{x}_n;\mathbf{b}) - \delta^x(\mathbf{x}_n,\mathbf{b})^T\zeta_n(\theta_m)\right). \tag{11}$$

In the Gaussian case, that is to say if Assumption 1 is fulfilled, we obtain:

$$\mathbb{E}\left[f_{\varepsilon_n}(y_n - g(\mathbf{x}_n+\zeta_n;\mathbf{b}))\right] = \phi(y_n - g(\mathbf{x}_n;\mathbf{b});\sigma^2_{\varepsilon_n,\zeta_n}(\mathbf{b})), \tag{12}$$

where $\sigma^2_{\varepsilon_n} = (\Sigma_\varepsilon)_{n,n}$ is the variance of ε_n, Σ_{ζ_n} is the covariance matrix of ζ_n, $\sigma^2_{\varepsilon_n,\zeta_n}(\mathbf{b}) := \sigma^2_{\varepsilon_n} + \delta^x(\mathbf{x}_n,\mathbf{b})^T\Sigma_{\zeta_n}\delta^x(\mathbf{x}_n,\mathbf{b})$, and ϕ is given by the following expression:

$$\phi(u;v) = \frac{1}{\sqrt{2\pi v}}\exp\left(-\frac{u^2}{2v}\right), \quad u\in\mathbb{R}, v>0. \tag{13}$$

3.2. Non-linear approximation of the code

As presented in the former section, the linear approximation allows a strong reduction of the computational cost associated with the inference of the posterior distribution of the calibration parameters. However, when confronted with simulators that take a long time, this calculation cost may still be too high. Indeed, in the positive configuration when each code evaluation also provides partial derivatives δ^x, each MCMC step requires at least N code evaluations for the computation of $f_{\beta|\mathbf{y}}$. In the negative configuration, this computational cost can be multiplied by a factor N_x+1 if finite differentials are used to numerically estimate the values of δ^x. To circumvent this problem, a first approach consists in assuming that there exists a reference value for β, denoted by β^*, such that for all $1\le n\le N$,

$$y_n \approx g(\mathbf{x}_n;\beta^*) + \delta^\beta(\mathbf{x}_n,\beta^*)^T(\beta-\beta^*) + \delta^x(\mathbf{x}_n,\beta^*)^T\zeta_n + \varepsilon_n, \tag{14}$$

$$\delta^\beta(\mathbf{x}_n,\beta^*) := \frac{\partial g}{\partial \beta}(\mathbf{x}_n;\beta^*). \tag{15}$$

 ISSN: 2102-6238

Here, the linearization is only assumed in the vicinity of β^*. Hence, once quantities $g(\mathbf{x}_n;\beta^*)$, $\delta^x(\mathbf{x}_n,\beta^*)$ and $\delta^x(\mathbf{x}_n,\beta^*)$ have been estimated, no additional code evaluation is needed for the computation of $\prod_{n=1}^{N}\mathbb{E}\left[f_{\varepsilon_n}(y_n-g(\mathbf{x}_n+\zeta_n;\mathbf{b}))\right]$ in each $\mathbf{b}\in\mathbb{B}$.

Even if iterative procedures can be proposed to sequentially adapt the value of β^*, this assumption is often too strong. As an alternative, it is generally proposed to directly construct a non-linear surrogate model of mapping $(\mathbf{x},\mathbf{b})\mapsto g(\mathbf{x};\mathbf{b})$ to speed up the MCMC procedure (see Liu et al. (2009)). In this work, we focus on the Gaussian Process Regression (GPR) surrogate model, as it provides a formalism that is particularly adapted to Bayesian calibration. It is based on Assumption 4.

Assumption 4. *Mapping $(\mathbf{x},\mathbf{b})\mapsto g(\mathbf{x};\mathbf{b})$ is a particular realization of a Gaussian stochastic process, $(\mathbf{x},\mathbf{b})\mapsto\gamma(\mathbf{x};\mathbf{b})$, whose mean and covariance functions are denoted by μ_g and C_g respectively:*

$$\mu_g(\mathbf{x},\mathbf{b})=\mathbb{E}\left[\gamma(\mathbf{x};\mathbf{b})\right],\quad C_g((\mathbf{x},\mathbf{b}),(\mathbf{x}',\mathbf{b}'))=Cov\left(\gamma(\mathbf{x};\mathbf{b}),\gamma(\mathbf{x}';\mathbf{b}')\right). \tag{16}$$

In this work, we assume that functions μ_g and C_g have been identified from a finite set of $T\geq 1$ code evaluations using one of the classical procedures presented in Rasmussen (2003); Santner et al. (2003). Hence, they respectively correspond to the mean function and the covariance function of a conditioned Gaussian process. The relation between the code outputs and the experimental results becomes:

$$\mathbf{y}=\Gamma(\zeta;\beta)+\varepsilon, \tag{17}$$

$$\Gamma(\zeta;\beta)=(\gamma(\mathbf{x}_1+\zeta_1;\beta),\ldots,\gamma(\mathbf{x}_N+\zeta_N;\beta)). \tag{18}$$

Equation (6) can finally be rewritten as:

$$f_{\beta|\mathbf{y}}(\mathbf{b})\propto f_\beta(\mathbf{b})\times\mathbb{E}\left[f_\varepsilon(\mathbf{y}-\Gamma(\zeta;\mathbf{b}))\right]. \tag{19}$$

It is important to notice that the expectation in Eq. (19) is carried out over the variability of Γ and ζ. And even if the generation of independent realizations of $\Gamma(\zeta;\beta)$ is fast, it poses some difficulties. Indeed, even though the elements of $(\zeta_n,\varepsilon_n)_{1\leq n\leq N}$ are statistically independent, there is no reason for $\gamma(\mathbf{x}_m+\zeta_m;\beta)+\varepsilon_m$ and $\gamma(\mathbf{x}_n+\zeta_n;\beta)+\varepsilon_n$ to be still independent for any $1\leq n\neq m\leq N$. This *a priori* prevents us from writing the N-dimensional integral associated with this expectation as a product of N integrals as it is done in Eq. (8). This means that the estimation of $\mathbb{E}\left[f_\varepsilon(\mathbf{y}-\Gamma(\zeta;\mathbf{b}))\right]$ in each value of $\mathbf{b}\in\mathbb{B}$ is likely to require huge sets of independent realizations of ζ and $\Gamma(\zeta;\mathbf{b})$. To circumvent this problem, it is interesting to notice that if

- Assumption 2 is valid,
- ε is a centered Gaussian random vector whose covariance matrix is a diagonal matrix denoted by Σ_ε,
- the metamodeling uncertainties are small compared to the experimental errors, in the sense that

$$(\Sigma_g(\zeta,\mathbf{b})+\Sigma_\varepsilon)^{-1}\approx\mathbf{D}(\zeta,\mathbf{b})^{-1}, \tag{20}$$

 ISSN: 2102-6238

$$\det\left(\Sigma_g(\zeta,\mathbf{b})+\Sigma_\varepsilon\right)\approx\det\left(\mathbf{D}(\zeta,\mathbf{b})\right), \tag{21}$$

where $\Sigma_g(\zeta,\mathbf{b})$ is the covariance matrix of $\Gamma(\zeta;\mathbf{b})|\zeta$,

$$\Sigma_g(\zeta,\mathbf{b}):=\begin{bmatrix} C_g((\mathbf{x}_1+\zeta_1,\mathbf{b}),(\mathbf{x}_1+\zeta_1,\mathbf{b})) & \cdots & C_g((\mathbf{x}_1+\zeta_1,\mathbf{b}),(\mathbf{x}_N+\zeta_N,\mathbf{b})) \\ \vdots & \ddots & \vdots \\ C_g((\mathbf{x}_N+\zeta_N,\mathbf{b}),(\mathbf{x}_1+\zeta_1,\mathbf{b})) & \cdots & C_g((\mathbf{x}_N+\zeta_N,\mathbf{b}),(\mathbf{x}_N+\zeta_N,\mathbf{b})) \end{bmatrix}, \tag{22}$$

and $\mathbf{D}(\zeta,\mathbf{b}):=\mathrm{diag}(\Sigma_g(\zeta,\mathbf{b})+\Sigma_\varepsilon)$, then we obtain:

$$\mathbb{E}\left[f_\varepsilon(\mathbf{y}-\Gamma(\zeta;\mathbf{b}))\right]\approx\prod_{n=1}^{N}\mathbb{E}\left[\phi\left(y_n-\mu_g(\mathbf{x}_n+\zeta_n;\mathbf{b});(\Sigma_g(\zeta,\mathbf{b})+\Sigma_\varepsilon)_{nn}\right)\right], \tag{23}$$

where function ϕ is defined by Eq. (13).

In practice, the approximations given by Eqs. (20) and (21) have only to be true in the regions of high probability for $\beta|\mathbf{y}$. This can guide the choice of the new points where to evaluate the code for an adaptive enrichment of the surrogate model of g as it is done in Damblin et al. (2013) and Perrin (2019).

3.3. *Approximation of the joint density*

When the linearization of g with respect to ζ is not valid, and when it is not possible to construct an emulator of it at a reasonable computational time, the formalism presented in the two former sections can not be applied to efficiently estimate PDF $f_{\beta|\mathbf{y}}$. This can be due to the fact that ζ lives in a too high-dimensional space (let us think about the case where ζ corresponds to a random field for instance). In that case, it can be more efficient to directly work on the identification of the dependence structure between β and $\mathbf{y}$. To this end, let $\{\zeta^{(1)},\ldots,\zeta^{(M)}\}$, $\{\beta^{(1)},\ldots,\beta^{(M)}\}$ and $\{\rho^{(1)}:=\mathbf{g}(\zeta^{(1)};\beta^{(1)}),\ldots,\rho^{(M)}:=\mathbf{g}(\zeta^{(M)};\beta^{(M)})\}$ be three sets gathering M independent realizations of random vectors ζ, β and $\rho:=\mathbf{g}(\zeta;\beta)$ respectively (the realizations of β being drawn from its prior distribution). From this information, it is indeed possible to approximate the joint density of (ρ,β), which we write $f_{\rho,\beta}$, using any parametric or non-parametric inference technique. However, when the dependence structure associated with the components of (ρ,β) is complex, which is likely to be the case here, the definition of a relevant parametric class to represent $f_{\rho,\beta}$ can become very difficult. In that case, nonparametric approaches are generally preferred to these parametric constructions (Wand and Jones (1995); Scott and Sain (2004)). In particular, we focus in this work on the multidimensional Gaussian kernel-density estimation (G-KDE) method, which approximates the PDF of (ρ,β) as a sum of M multidimensional Gaussian PDFs, which are centered at each available independent realization of (ρ,β):

$$f_{\rho,\beta}(\mathbf{y},\mathbf{b})\approx\widehat{f}_{\rho,\beta}(\mathbf{y},\mathbf{b}):=\frac{1}{M}\sum_{m=1}^{M}\Phi((\mathbf{y},\mathbf{b});(\rho^{(m)},\beta^{(m)}),h^2\mathbf{R}), \tag{24}$$

where:

 ISSN: 2102-6238

– $\mathbf{R}$ is the empirical estimation of the covariance matrix of (ρ,β), whose block decomposition is written as:

$$\widehat{\mathbf{R}} = \begin{bmatrix} \widehat{\mathbf{R}}_{\rho} & \widehat{\mathbf{R}}_{\rho\beta} \\ \widehat{\mathbf{R}}_{\rho\beta}^{T} & \widehat{\mathbf{R}}_{\beta} \end{bmatrix}, \tag{25}$$

– h is a normalisation constant to be identified from the data (see Perrin et al. (2018) for more details about the estimation of h),
– Φ is the multidimensional Gaussian density, such that for any vector $\mathbf{a} \in \mathbb{R}^Q$ and any $(Q \times Q)$-dimensional positive definite matrix $\mathbf{A}$,

$$\Phi(\mathbf{u};\mathbf{a},\mathbf{A}) := \frac{\exp\left(-\frac{1}{2}(\mathbf{u}-\mathbf{a})^T\mathbf{A}^{-1}(\mathbf{u}-\mathbf{a})\right)}{(2\pi)^{Q/2}\sqrt{\det(\mathbf{A})}}, \quad \mathbf{u} \in \mathbb{R}^Q. \tag{26}$$

By Gaussian conditioning, we deduce, for all $\mathbf{b} \in \mathbb{B}$:

$$f_{\rho|\beta=\mathbf{b}}(\mathbf{y}) \approx \frac{\widehat{f}_{\rho,\beta}(\mathbf{y},\mathbf{b})}{\int_{\mathbb{B}} \widehat{f}_{\rho,\beta}(\mathbf{y},\mathbf{b}')d\mathbf{b}'} = \sum_{m=1}^{M} \frac{\alpha_m(\mathbf{b})}{\sum_{m'=1}^{M}\alpha_{m'}(\mathbf{b})}\Phi(\mathbf{y};\mu_m(\mathbf{b}),\mathbf{R}_{\mathbf{y}}), \tag{27}$$

$$\alpha_m(\mathbf{b}) := \exp\left(-\frac{1}{2h^2}\left(\mathbf{b}-\beta^{(m)}\right)^T\mathbf{R}_{\beta}^{-1}\left(\mathbf{b}-\beta^{(m)}\right)\right), \tag{28}$$

$$\mu_m(\mathbf{b}) := \rho^{(m)} + \mathbf{R}_{\rho\beta}\mathbf{R}_{\beta}^{-1}(\mathbf{b}-\beta^{(m)}), \tag{29}$$

$$\mathbf{R}_{\mathbf{y}} := h^2\left(\mathbf{R}_{\rho} - \mathbf{R}_{\rho\beta}\mathbf{R}_{\beta}^{-1}\mathbf{R}_{\rho\beta}^{T}\right). \tag{30}$$

Additionally, if ε is a Gaussian random vector that is independent of β, such that $\varepsilon \sim \mathcal{N}(\mathbf{0},\Sigma_{\varepsilon})$, using Eq. (4), it is straightforward to check that:

$$f_{\mathbf{y}|\beta=\mathbf{b}}(\mathbf{y}) \approx \sum_{m=1}^{M} \frac{\alpha_m(\mathbf{b})}{\sum_{m'=1}^{M}\alpha_{m'}(\mathbf{b})}\Phi(\mathbf{y};\mu_m(\mathbf{b}),\mathbf{R}_y+\Sigma_{\varepsilon}), \tag{31}$$

from which we can deduce an explicit approximation of the posterior PDF of β recalling that:

$$f_{\beta|\mathbf{y}}(\mathbf{b}) \propto f_{\beta}(\mathbf{b}) \times f_{\mathbf{y}|\beta=\mathbf{b}}(\mathbf{y}). \tag{32}$$

The main drawback of this approach comes from the fact that the number of code evaluations that is needed to correctly approximate the PDF of (ρ,β) can quickly become extremely high when the number of experiments increases. However, in the same manner than in Section 3.1 and 3.2, if Assumption 2 is fulfilled, that is, if the components $\varepsilon_1,\ldots,\varepsilon_N$ and $\zeta_1,\ldots,\zeta_N$ are statistically independent, the components of $(\rho|\beta)$ are, by construction, also statistically independent. For all $\mathbf{y} = (y_1,\ldots,y_N) \in \mathbb{R}^N$, it comes:

$$f_{\rho|\beta=\mathbf{b}}(\mathbf{y}) = \prod_{n=1}^{N} f_{\rho_n|\beta=\mathbf{b}}(y_n), \tag{33}$$

 ISSN: 2102-6238

$$f_{\rho_n|\beta=\mathbf{b}}(y_n) = \frac{f_{\rho_n,\beta}(y_n,\mathbf{b})}{\int_{\mathbb{B}} f_{\rho_n,\beta}(y_n,\mathbf{b})d\mathbf{b}}. \tag{34}$$

Hence, even if the dimension of (ρ,β) is $N+N_b$, its PDF can be constructed from the aggregation of N PDFs in dimension $1+N_b$ only, whose identifications are likely to be much easier.

Remarks

- In this section, there is no restriction for the distribution of ζ. This is particularly interesting when confronted to cases where the dimension of ζ is high. Of course, the more complex ζ is, and the more difficult the identification of the PDF of (ρ,β) is likely to be.
- The interest of considering a Gaussian kernel for the approximation of $f_{\rho,\beta}$ is double: it allows explicit derivations of the PDF of $\rho|\beta$, but also of the conditioned PDF of $\beta|\mathbf{y}$ when ε is Gaussian. If ε was not Gaussian, the computation of $f_{\beta|\mathbf{y}}$ would be based on the convolution product between the PDFs of ε and $\rho|\beta$, and there is no denying that other kernels could be more adapted depending on the PDF of ε.

3.4. Input uncertainties and model error

The three former sections are based on the assumption that there exists non-negligible input uncertainties, which are relatively well-known in the sense that it is possible to generate independent realizations of ζ. However, even if there is no information about these input uncertainties, it is crucial not to ignore them. Indeed, in that case, the calibration results could become overconfident around values that are potentially biased, and the uncertainty bands in the model prediction may not be large enough to adequately capture the true value of the quantity of interest we want to predict. To circumvent this problem, conventional Bayesian calibration generally introduces an error term, which is explicitly represented by a function $\mathbf{x} \mapsto \varepsilon^{\text{err}}(\mathbf{x})$, such that for all $1 \leq n \leq N$:

$$y_n = g(\mathbf{x}_n;\beta) + \varepsilon_n + \varepsilon^{\text{err}}(\mathbf{x}_n). \tag{35}$$

This error term is often modeled by a Gaussian process with a covariance function C_{err}, the parameters of which being estimated in a chosen class of covariance kernels, such as the nugget, the squared exponential or the Matern classes (Kennedy and O'Hagan (2001)). This choice of covariance class is not easy, while playing a major role in the calibration results. Indeed, by choosing to make this covariance depend on $\mathbf{x}$, potential error compensations may prevent us from recovering the correct value of β. On the other hand, by proposing covariances independent of $\mathbf{x}$, it is not possible to integrate potential amplifications or reductions of uncertainties by the code. In this case, since the uncertainties explaining the difference between the measurement and the simulation are not correct, there is no reason why the uncertainties on β should be clearly identified.

However, looking at Eqs. (35) and (10), if Assumption 3 is valid, $\varepsilon^{\text{err}}(\mathbf{x}_n) \approx \delta^x(\mathbf{x}_n,\mathbf{b})^T\zeta_n$. Hence, a trade-off between minimizing error compensations and integrating the sensitivity of the quantity of interest to system parameters seems possible by considering covariance functions in the following form:

 ISSN: 2102-6238

$$C_{\text{err}}(\mathbf{x},\mathbf{x}') = \frac{\partial g}{\partial \mathbf{x}}(\mathbf{x};\beta^*)^T \Lambda \frac{\partial g}{\partial \mathbf{x}}(\mathbf{x}';\beta^*)\delta_{\mathbf{x},\mathbf{x}'}, \quad \mathbf{x},\mathbf{x}' \in \mathbb{X}, \tag{36}$$

with $\delta_{\mathbf{x},\mathbf{x}'} = 1$ if $\mathbf{x} = \mathbf{x}'$ and 0 otherwise, β^* a reference value for β in $\mathbb{B}$, and Λ a $(N_x \times N_x)$-dimensional positive definite and symmetrical matrix (which may be diagonal for simplicity) to be identified. By construction, Λ would correspond to the covariance matrix of $\zeta_1,\ldots,\zeta_N$ under the assumption that they are identically distributed.

Remarks.

- If the code was also replaced by a Gaussian process γ, as it is done in Section 3.2, it would be important to consider well-distinct covariances for γ and ε^{err} for the sake of identifiability.
- If the distribution of ε was unknown, error term ε^{err} could aggregate at the same time input and output experimental uncertainties.

4. Application

The objective of this section is to illustrate on two examples the pros and cons of the different methods presented in the former sections. In each case, the same Metropolis-Hastings algorithms, based on the same parameters, were used to get realizations of the posterior distributions.

4.1. Analytical example

In this section, we are interested in predicting a quantity of interest y^{true}. To this end, we assume that we have access to the following simulator:

$$g : \begin{cases} [0,1]\times\mathbb{R}^2 & \to & \mathbb{R} \\ (x;\mathbf{b}=(b_1,b_2)) & \mapsto & \frac{10 b_1 x}{\exp(b_2\sqrt{x})} \end{cases}. \tag{37}$$

This simulator is perfect in the sense that for all $x \in [0,1]$, $y^{\text{true}}(x) = g(x;\beta^\star = (0.3,-0.4))$. Vector $\beta^\star$ being *a priori* unknown, it is modeled by a random vector β, whose *a priori* PDF is given by:

$$f_\beta(\mathbf{b}) = \begin{cases} 1/4 \text{ if } -1 \le b_1, b_2 \le 1, \\ 0 \text{ otherwise.} \end{cases} \tag{38}$$

The available information is made of $N = 10$ noisy observations gathered in the vector $\mathbf{y} = (y_1,\ldots,y_N)$, which are given by:

$$y_n = y^{\text{true}}(x_n^{\text{true}}) + \varepsilon_n, \quad x_n^{\text{true}} = x_n + \zeta_n, \quad 1 \le n \le N. \tag{39}$$

There, $x_1,\ldots,x_N$ are fixed values of x in $[0,1]$, the input uncertainties $\zeta_1,\ldots,\zeta_N$ correspond to N independent realizations of the Gaussian random variable $\zeta \sim \mathcal{N}(0,0.1)$, whereas the output uncertainties $\varepsilon_1,\ldots,\varepsilon_N$are supposed to be N independent realizations of the Gaussian random variable $\varepsilon \sim \mathcal{N}(0,0.1)$. Assumptions 1 and 2 are therefore valid for this application.

 ISSN: 2102-6238

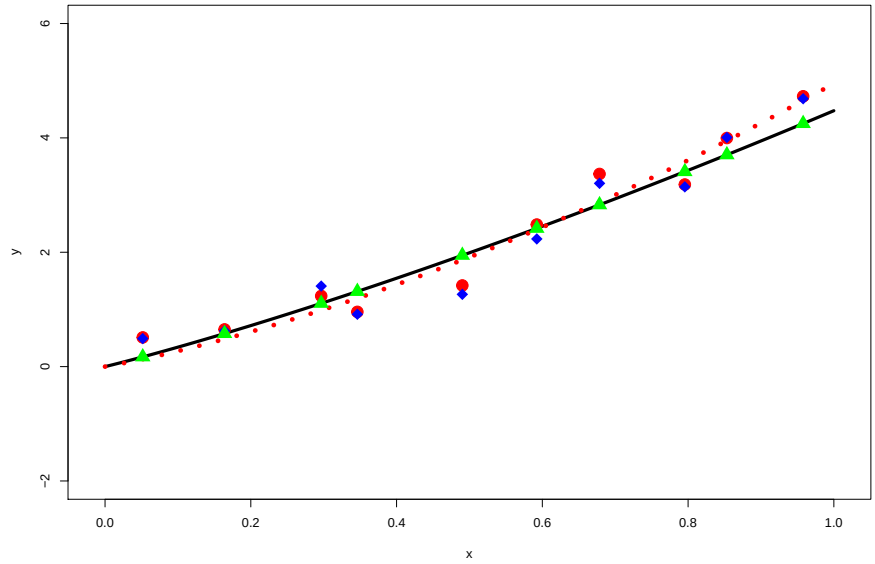


FIGURE 1. *Comparison between the evolutions of* $x \mapsto y^{true}(x)$ *(in black continuous line) and* $x \mapsto g(x;\beta^{MC} = (0.21,-0.88))$ *(in red dotted line). The points* (x_n, y_n) *are represented by red dots, which defines the available information for the identification, whereas the blue diamonds and the green triangles correspond respectively to the points* $(x_n, y^{true}(x_n^{true}))$ *and* $(x_n, y^{true}(x_n))$.

To emphasize the role of these two sources of uncertainties, Figure 1 compares a particular set of observed values of y_n, and the associated values of $y^{\text{true}}(x_n^{\text{true}})$ and $y^{\text{true}}(x_n)$. It also shows the evolution of y^{true} and $x \mapsto g(x;\beta^{\text{MC}})$, where β^{MC} is the least-squares estimate of $\beta^\star$ when neglecting the uncertainties:

$$\beta^{\text{MC}} := \arg\min_{\mathbf{b}\in\mathbb{R}^2} \sum_{n=1}^{N} (y_n - g(x_n;\mathbf{b}))^2. \tag{40}$$

Six configurations are compared for the identification of $f_{\beta|\mathbf{y}}$:

1. $f_{\beta|\mathbf{y}}^{\text{ref}}$ is the Monte Carlo approximation of $f_{\beta|\mathbf{y}}$ based on Eq. (9), and will be seen as the reference for the comparison of the different methods. For each value of $\mathbf{b}$, it is computed from $N \times 10^4$ code evaluations in independent values of ζ to guarantee a satisfying convergence of the expectations.
2. $f_{\beta|\mathbf{y}}^{\text{without}}$ corresponds to the case where the input uncertainties are not taken into account. Using the notations of the former sections, this PDF is simply given by:

$$f_{\beta|\mathbf{y}}^{\text{without}}(\mathbf{b}) \propto f_\beta(\mathbf{b}) \times f_\varepsilon(\mathbf{y} - \mathbf{g}(\mathbf{0};\mathbf{b})), \quad \mathbf{b} \in [-1,1]^2. \tag{41}$$

By construction, each evaluation of $f_{\beta|\mathbf{y}}^{\text{without}}$ requires only N code evaluations.

 ISSN: 2102-6238

3. $f^{\text{lin}}_{\beta|\mathbf{y}}$ is the approximation of $f_{\beta|\mathbf{y}}$ based on the linearization of g presented in Section 3.1. If the gradient function was not directly provided by the code, $N(N_x+1)$ code evaluations would be needed for each evaluation of $f^{\text{lin}}_{\beta|\mathbf{y}}$.
4. $f^{\text{GP}}_{\beta|\mathbf{y}}$ is the approximation of $f_{\beta|\mathbf{y}}$ based on the emulation of function g using a Gaussian process as explained in Section 3.2. Two variants of this approximation are compared :
   – the PDF associated with the case where quantity $\mathbb{E}\left[f_\varepsilon(\mathbf{y}-\Gamma(\zeta;\mathbf{b}))\right]$ is directly approximated using a Monte Carlo approach in dimension N will be denoted by $f^{\text{GP,full}}_{\beta|\mathbf{y}}$. It is based on 10^3 independent realizations of $\Gamma(\zeta;\mathbf{b})$, each of them being associated with the same trajectory of Gaussian process γ ;
   – the PDF associated with the case where this multidimensional integral is approximated by a product of one-dimensional integrals will be denoted by $f^{\text{GP,1D}}_{\beta|\mathbf{y}}$. Once again, it is estimated by a MC approach based on 10^2 independent realizations of $\Gamma(\zeta;\mathbf{b})$. In the following results, each realization of $\Gamma(\zeta;\mathbf{b})$ is also associated with the same trajectory of γ, even if, looking at Eq. (23), it was not necessary.

   In both cases, the number of calls to the simulators that is needed to solve the inference problem is strongly reduced, as once the GP emulator has been constructed, no additional code evaluation is required to compute $f^{\text{GP,full}}_{\beta|\mathbf{y}}$ or $f^{\text{GP,1D}}_{\beta|\mathbf{y}}$. In the following, the number of code evaluations used for the construction of the GP surrogate is T.
5. $f^{\text{KDE}}_{\beta|\mathbf{y}}$ is the approximation of $f_{\beta|\mathbf{y}}$ relying on the nonparametric approximation of the joint PDF of $(\mathbf{g}(\zeta;\beta),\beta)$ described in Section 3.3. In the same manner than for the GP-based approximation, once the approximation of this PDF has been computed, no additional code evaluation is needed to compute $f^{\text{KDE}}_{\beta|\mathbf{y}}$ in any $\mathbf{b}\in\mathbb{B}$. The total number of code evaluations used for the construction of the N kernel density approximations of the PDFs of $(g(\mathbf{x}_n+\zeta_n;\beta),\beta)$ is T.
6. At last, $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$, $f^{\text{mod,diff}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$ are the approximations of $f_{\beta|\mathbf{y}}$ based on the model error formalism presented in Section 3.4. The covariance function chosen for $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$ is a Matern-5/2, it is equal to $(x,x')\mapsto\sigma^2\frac{\partial g}{\partial x}(x,\beta^{\text{MC}})\frac{\partial g}{\partial x}(x',\beta^{\text{MC}})\delta_{x,x'}$ for $f^{\text{mod,diff}}_{\beta|\mathbf{y}}$, while the covariance chosen for $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$ is equal to $(x,x')\mapsto\sigma^2\delta_{x,x'}$. The parameters associated with these three covariance functions are estimated by their maximum likelihood estimators. For this approach, the computational cost is double. It is first given by the number of code evaluations associated with the estimation of β^{MC} and the parameters of the covariance functions. Then, at least N code evaluations are required for each evaluation of $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$, $f^{\text{mod,diff}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$. This second cost could however be reduced by replacing the true code by the surrogate model associated with $f^{\text{GP}}_{\beta|\mathbf{y}}$.

The results are summarized in three groups of figures. Figure 2 first compares the approaches for which each approximation of $f_{\beta|\mathbf{y}}$ needs at least N code evaluations. In this figure, we confirm the fact that there is a risk that the calibrated results be overconfident around biased values when neglecting the input uncertainties. Indeed, PDF $f^{\text{without}}_{\beta|\mathbf{y}}$ is for this example almost a Dirac-like distribution while being centered far from the true value of $\beta^\star$. For this first example, we also see that $f^{\text{lin}}_{\beta|\mathbf{y}}$ is very close to $f^{\text{ref}}_{\beta|\mathbf{y}}$, which is due to the fact that function $\mathbf{x}\mapsto g(\mathbf{x};\mathbf{b})$ is almost linear for each $\mathbf{b}$ in $\mathbb{B}$. The approaches based on the linearization of g and the model error with the

 ISSN: 2102-6238

covariance function $(x,x') \mapsto \sigma^2 \frac{\partial g}{\partial x}(x,\beta^{\mathrm{MC}}) \frac{\partial g}{\partial x}(x',\beta^{\mathrm{MC}}) \delta_{x,x'}$ give also very close results. This was expected as the likelihood on which they are based present very similar expressions. Finally, we observe that considering a generic covariance function, as it is done for $f^{\mathrm{mod,Matern}}_{\beta|\mathbf{y}}$ or $f^{\mathrm{mod,nugget}}_{\beta|\mathbf{y}}$, can either underestimate or overestimate the scattering of $\beta|\mathbf{y}$. This is also not surprising as information is missing for a precise identification of $f_{\beta|\mathbf{y}}$.

Then, Figure 3 presents the results associated with the replacement of the true code by a GP surrogate. The figures on the left are based on $T = 100$ code evaluations, whereas the figures on the right are based on $T = 500$ code evaluations. In both cases, the points where the code is to be called are chosen to fill as much as possible the input domain $\mathbb{X} \times \mathbb{B}$ (see Perrin and Cannamela (2017) for further details about the construction of space-filling designs). Looking at the four figures at the top, we see that for these two values of T, $f^{\mathrm{GP,1D}}_{\beta|\mathbf{y}}$ is much closer to $f_{\beta|\mathbf{y}}$ than $f^{\mathrm{GP,full}}_{\beta|\mathbf{y}}$. This can be explained by the difficulty to approximate an integral in dimension $N = 10$ using a MC approach based on a reduced number of points. This lays stress on the importance of the approximation given by Eq. (23), which writes this N-dimensional integrals as a product of N one-dimensional integrals that is much easier to estimate using samples-based approaches. To quantify the error due to this approximation, the images at the bottom of Figure 3 compare the quantities $q_1(\mathbf{z},\mathbf{b})$ and $q_2(\mathbf{z},\mathbf{b})$ in 100 values of $\mathbf{z}$ and $\mathbf{b}$ randomly chosen in $\mathbb{X}^N$ and $\mathbb{B}$, where:

$$q_1(\mathbf{z},\mathbf{b}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{y}-\mu_g(\mathbf{z};\mathbf{b}))^T \left(\Sigma_g(\mathbf{z},\mathbf{b})+\Sigma_\varepsilon\right)^{-1}(\mathbf{y}-\mu_g(\mathbf{z};\mathbf{b}))\right)}{\det\left(\Sigma_g(\mathbf{z},\mathbf{b})+\Sigma_\varepsilon\right)^{1/2}}, \tag{42}$$

$$q_2(\mathbf{z},\mathbf{b}) = \prod_{n=1}^{N} \frac{1}{\left(\Sigma_g(\mathbf{z},\mathbf{b})+\Sigma_\varepsilon\right)_{nn}{}^{1/2}} \exp\left(-\frac{(y_n-\mu_g(\mathbf{z}_n;\mathbf{b}))^2}{2(\Sigma_g(\mathbf{z},\mathbf{b})+\Sigma_\varepsilon)_{nn}}\right), \tag{43}$$

$$\mu_g(\mathbf{z};\mathbf{b}) := (\mu_g(\mathbf{z}_1;\mathbf{b}),\ldots,\mu_g(\mathbf{z}_n;\mathbf{b})). \tag{44}$$

We therefore verify that by increasing T, we improve the relevance of the GP-based emulator and make the points $(q_1(\mathbf{z},\mathbf{b}),q_2(\mathbf{z},\mathbf{b}))$ be closer to the first bisectrix at the same time, and therefore increase the relevance of $f^{\mathrm{GP,1D}}_{\beta|\mathbf{y}}$.

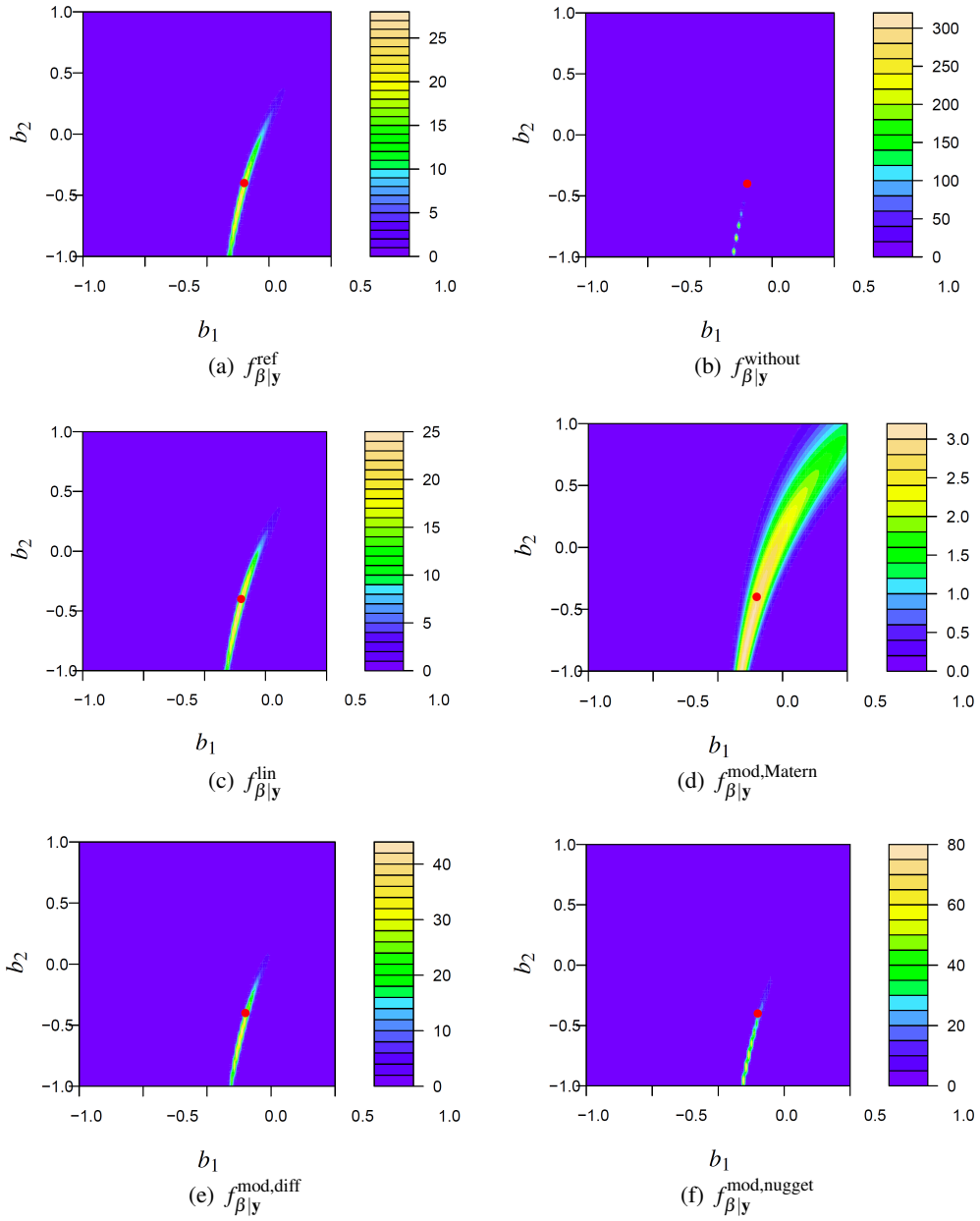
At last, Figure 4 presents the results associated with the KDE-based approach presented in Section 3.3. As explained in Section 3.3, estimating a PDF being often more difficult than approximating a deterministic function, the convergence of $f^{\mathrm{KDE}}_{\beta|\mathbf{y}}$ to $f_{\beta|\mathbf{y}}$ is relatively low compared to the former approaches, in the sense that the value of T needs to be high to get a satisfying approximation.

4.2. *Ballistic example*

The second application deals with the prediction of the trajectory of reconnaissance drones that are catapulted from the ground. To this end, we have access to a model coupling point mechanics and aerodynamics in two dimensions, which is based on the following equations:

$$\begin{cases} m\ddot{\mathbf{p}} = -mg_r\mathbf{e}_z + \|\dot{\mathbf{p}}-\mathbf{s}_w\|\left(\mathbf{e}_x(-\beta_1\alpha_x-\beta_2\alpha_z)+\mathbf{e}_z(-\beta_1\alpha_z+\beta_2\alpha_x)\right), \\ \alpha_x = \langle \dot{\mathbf{p}}-\mathbf{s}_w,\mathbf{e}_x\rangle,\ \alpha_z = \langle \dot{\mathbf{p}}-\mathbf{s}_w,\mathbf{e}_z\rangle, \\ \dot{\mathbf{p}}(t=0)=\mathbf{s}_0,\ \ \mathbf{p}(t=0)=(0,0), \end{cases}$$

 ISSN: 2102-6238



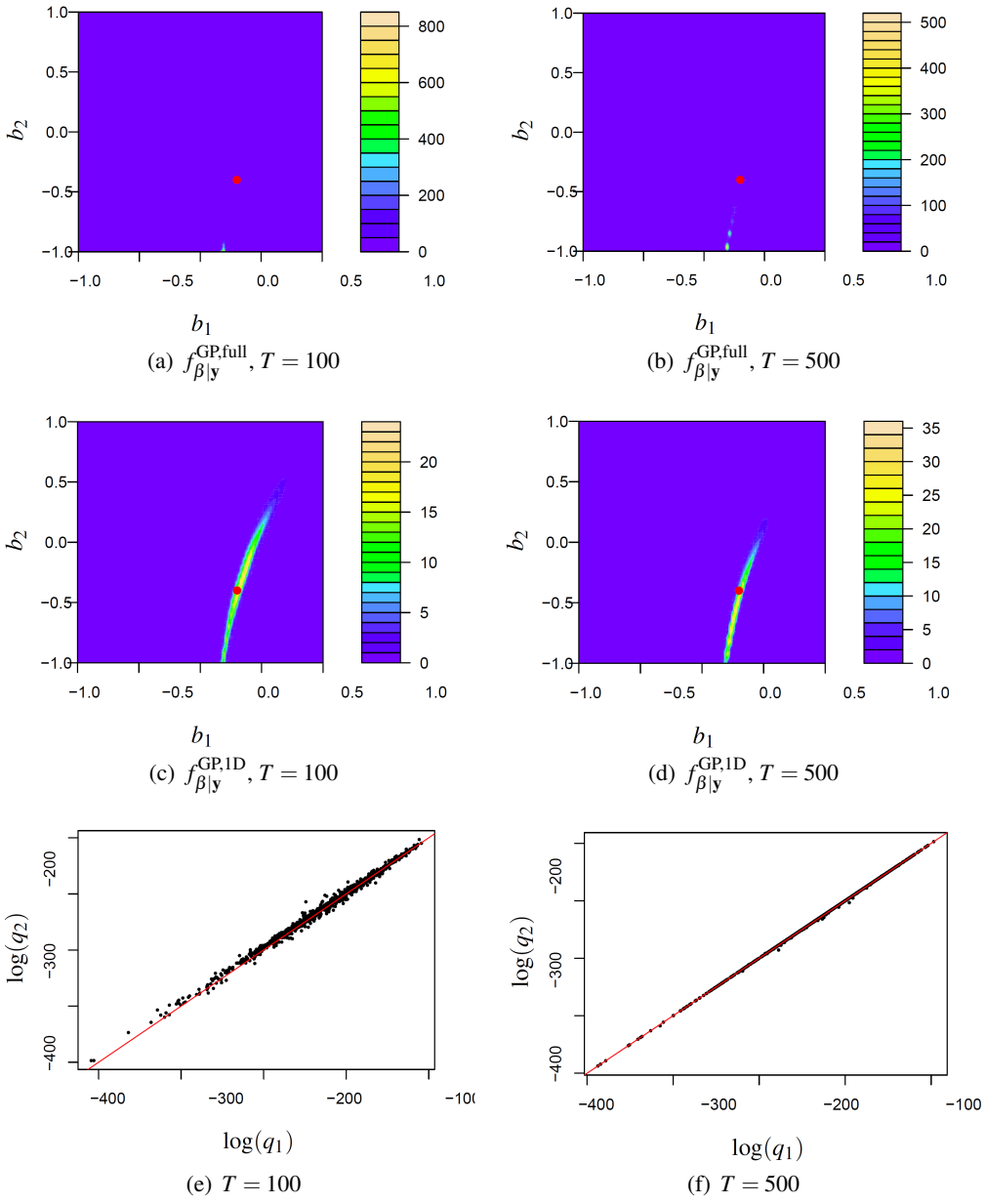
(a) $f^{\text{ref}}_{\beta|\mathbf{y}}$ (b) $f^{\text{without}}_{\beta|\mathbf{y}}$

(c) $f^{\text{lin}}_{\beta|\mathbf{y}}$ (d) $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$

(e) $f^{\text{mod,diff}}_{\beta|\mathbf{y}}$ (f) $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$

FIGURE 2. *Comparison of several approximations of* $f_{\beta|\mathbf{y}}$. *In each figure, the red point corresponds to the value of* $\beta^\star$.

 ISSN: 2102-6238



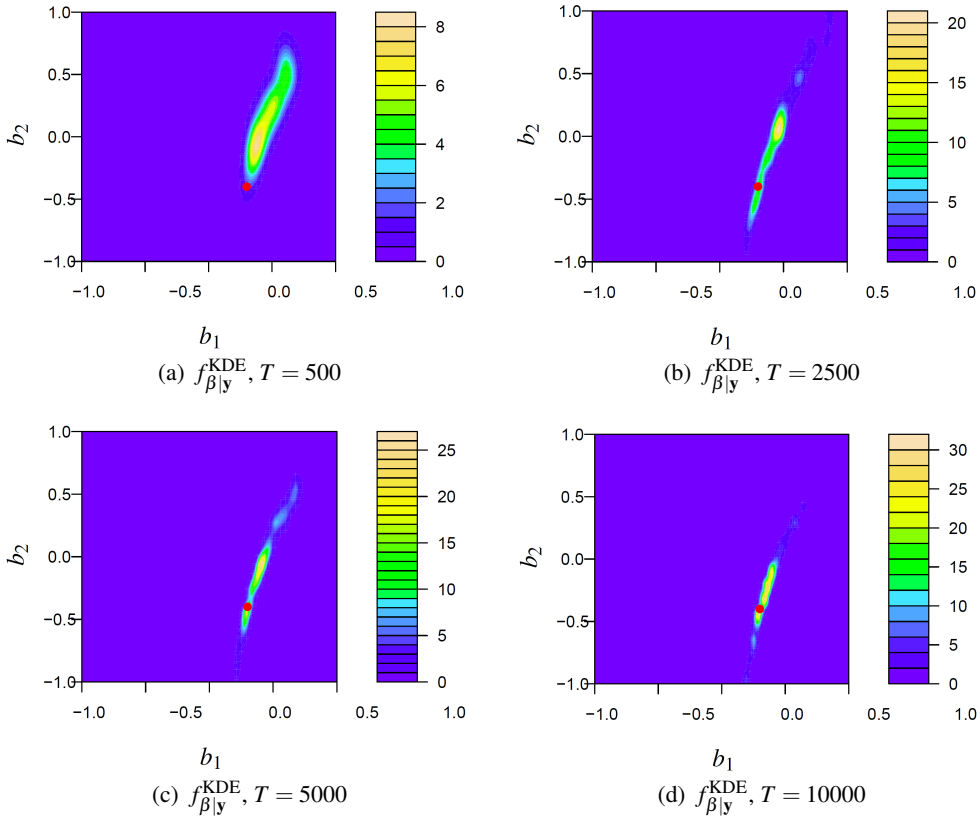
(a) $f^{\text{GP,full}}_{\beta|\mathbf{y}}$, $T = 100$ (b) $f^{\text{GP,full}}_{\beta|\mathbf{y}}$, $T = 500$

(c) $f^{\text{GP,1D}}_{\beta|\mathbf{y}}$, $T = 100$ (d) $f^{\text{GP,1D}}_{\beta|\mathbf{y}}$, $T = 500$

(e) $T = 100$ (f) $T = 500$

FIGURE 3. *Approximations of $f_{\beta|\mathbf{y}}$ when replacing the true code by a GP surrogate. In the four top figures, the red point corresponds to the value of $\beta^\star$. The two bottom figures compares the values of $q_1(\mathbf{z},\mathbf{b})$ and $q_2(\mathbf{z},\mathbf{b})$, which are defined by Eqs. (43) and (44), the red lines being the first bisectrix.*

 ISSN: 2102-6238



(a) $f^{\mathrm{KDE}}_{\beta|\mathbf{y}}$, $T = 500$ (b) $f^{\mathrm{KDE}}_{\beta|\mathbf{y}}$, $T = 2500$

(c) $f^{\mathrm{KDE}}_{\beta|\mathbf{y}}$, $T = 5000$ (d) $f^{\mathrm{KDE}}_{\beta|\mathbf{y}}$, $T = 10000$

FIGURE 4. *Approximations of $f_{\beta|\mathbf{y}}$ using a KDE of the-based approach. In each figure, T is the number of code evaluations that was used to construct approximation $f^{KDE}_{\beta|\mathbf{y}}$, while the red point corresponds to the value of $\beta^\star$.*

 ISSN: 2102-6238

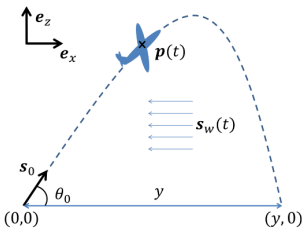


FIGURE 5. *Description of the experimental protocol for the ballistic example.*

n	V_0 (m/s)	θ_0 (rad)	m (kg)
1	logN(59.8, 0.400)	logN(0.614, 0.00523)	logN(1.82, 0.0100)
2	logN(124, 0.400)	logN(0.329, 0.00523)	logN(1.51, 0.0100)
3	logN(146, 0.400)	logN(0.525, 0.00523)	logN(1.31, 0.0100)
4	logN(76.2, 0.400)	logN(0.465, 0.00523)	logN(1.00, 0.0100)
5	logN(101, 0.400)	logN(0.407, 0.00523)	logN(1.76, 0.0100)

TABLE 1. *Description of the input experimental uncertainties. For all a, b, logN(a,b) and $\mathcal{N}(a,b)$ respectively correspond to the lognormal and to the normal distributions with mean a and standard deviation b.*

where (see Figure 5 for a graphical representation of the experimental protocol):

- $\mathbf{p}(t) = p_x(t)\mathbf{e}_x + p_z(t)\mathbf{e}_z$ is the position at time t of the center of inertia of the system,
- g_r is the gravity that is supposed to be constant and known,
- m is the mass of the system,
- $\mathbf{s}_w(t)$ is the wind speed at time t,
- $\mathbf{s}_0 = V_0(\cos(\theta_0)\mathbf{e}_x + \sin(\theta_0)\mathbf{e}_z)$ is the initial speed,
- β_1 and β_2 are respectively the drag and the lift coefficients.

Using an explicit scheme in time, the system of equations defined by Eq. (4.2) can be solved numerically. For given values of $\mathbf{x} = (m, V_0, \theta_0, \{\mathbf{s}_w(t), t \geq 0\})$ and $\beta = (\beta_1, \beta_2)$, let $g(\mathbf{x}; \beta)$ be the landing position of the drone, which corresponds to the horizontal component of $\mathbf{p}(t)$ when $\langle \mathbf{p}(t), \mathbf{e}_z \rangle = 0$. Whereas the components of $\mathbf{x}$ can directly be measured, experiments are carried out to assess the true value of β, which is chosen equal to $\beta^\star = (0.1, 0.05)$ in the following numerical analyses. We assume that the same drone is launched $N = 5$ times for different values of V_0 and θ_0, with different loadings, such that m also varies from one experiment to another. These values of m, V_0, θ_0 are affected by uncertainties, whose characteristics are detailed in Table 1.

To better compare the pros and cons of the different approximations introduced, two cases are

n	y_n (m, no wind)	y_n (m, presence of wind)
1	$\mathcal{N}(320, 1)$	$\mathcal{N}(322, 1)$
2	$\mathcal{N}(862, 1)$	$\mathcal{N}(858, 1)$
3	$\mathcal{N}(1140, 1)$	$\mathcal{N}(1122, 1)$
4	$\mathcal{N}(417, 1)$	$\mathcal{N}(414, 1)$
5	$\mathcal{N}(678, 1)$	$\mathcal{N}(678, 1)$

TABLE 2. *Description of the output experimental uncertainties. For all a, b, $\mathcal{N}(a,b)$ corresponds to the normal distributions with mean a and standard deviation b.*

 ISSN: 2102-6238

moreover distinguished concerning the wind speed. On the one hand, the wind speed is equal to zero: $\mathbf{s}_w(t) = \mathbf{0}$ for all $t \geq 0$. On the other hand, the wind speed is modeled by the following centered Gaussian process:

$$\begin{cases} \mathbf{s}_w(t) = S(t)\mathbf{e}_x, \\ \mathbb{E}\left[S(t), S(t')\right] = 25\exp(-2|t-t'|), \quad t, t' \geq 0. \end{cases} \tag{45}$$

In this example, the wind speed is included in the system parameters as it can be seen as a particular boundary condition. It is expected to play a major role on the relevance of the calibration results associated with the different simplifications introduced in Section 3. In particular, it is introduced on purpose to lay stress on the difficulty for most of the presented methods to take into account functional inputs. The true evolution of the wind speed for each measurement is supposed to be unknown, but we assume it is possible to generate independent realizations of it.

In both cases, there are uncertainties on the landing positions, denoted by y_n, $1 \leq n \leq N$ (see Table 2 for a listing of the considered values of y_n). As these uncertainties only depend on the measurement device, their amplitudes are independent of the presence of wind.

In the same manner than in Section 4.1, six cases are now compared.

1. $f^{\text{ref}}_{\beta|\mathbf{y}}$ is the Monte Carlo approximation of $f_{\beta|\mathbf{y}}$ based on Eq. (9). It is based on the coupling of a Monte Carlo approach for the estimation of the expectation, and a MCMC procedure to generate values of β that can approximately be seen as independent realizations of $\beta|\mathbf{y}$. For this approach, the presence of wind is naturally taken into account, and should only tend to make the uncertainties on the calibration results be higher.
2. $f^{\text{without}}_{\beta|\mathbf{y}}$ corresponds to the case where the input uncertainties are not taken into account.
3. $f^{\text{lin}}_{\beta|\mathbf{y}}$ is the approximation of $f_{\beta|\mathbf{y}}$ based on a linearization of g with respect to $\mathbf{b}$, V_0, θ_0 and m. As the relationship between the values of $\mathbf{b}$ and (V_0, θ_0, m), and the associated value of the landing position is not explicit, the derivatives are estimated using finite differentials. For this approach, everything is made as if there was no wind.
4. $f^{\text{GP,1D}}_{\beta|\mathbf{y}}$ is the approximation of $f_{\beta|\mathbf{y}}$ based on the emulation of function g using a Gaussian process as explained in Section 3.2. The presence of a functional input is however not really adapted to this formalism, as it was explained in Section 3.3. Hence, in the same manner than for the linear case, the wind speed is here neglected. Consequently, the GPR approximation is based on the evaluation of g in a $(T \times 5)$-dimensional space-filling design in $\mathbb{X} \times \mathbb{B}$, where $\mathbb{X} = [50, 150] \times [\pi/10, \pi/5] \times [1, 2]$ denotes the input space of (V_0, θ_0, m) only. For this application, we notice in Table 2 that the variance of y_n is small compared to its mean. This means that a high value of T is likely to be needed for the metamodel error to be smaller than the experimental error, which is a prerequisite for the approximation of the multidimensional integral by a product of one-dimensional integrals to be valid. In practice, the value of T was gradually increased until 10^3, where satisfying results were obtained for this approximation.

 ISSN: 2102-6238

5. $f^{\text{KDE}}_{\beta|\mathbf{y}}$ is the approximation of $f_{\beta|\mathbf{y}}$ relying on the nonparametric approximation of the joint PDF of $(\mathbf{g}(\zeta;\beta),\beta)$ described in Section 3.3. For each $1 \leq n \leq N$, the KDE approximations were based on 200 code evaluations, such that the total computational cost is equivalent to the GPR case. To integrate the potential presence of wind, for each realization of β, a realization of the wind speed was drawn at random to compute the associated realization of $\mathbf{g}(\zeta;\beta)$.
6. At last, $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$, $f^{\text{mod,diff}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$ are the approximations of $f_{\beta|\mathbf{y}}$ based on the model error formalism presented in Section 3.4. Here, as the input uncertainties are aggregated in a global additive error term for the output, the presence of wind should not be a problem.

The results are summarized in Figure 6. In this figure, focusing first on the results associated with the absence of wind, we see that in the same manner than in Section 4.1:

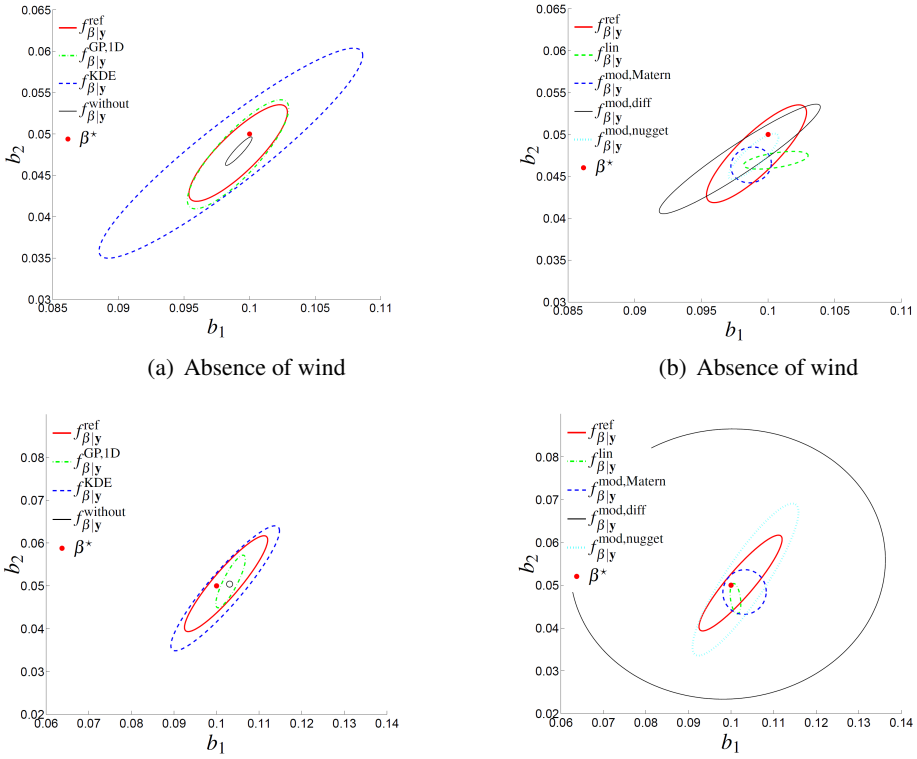
– very interesting results are obtained when considering the GPR approximation of the code response;
– when neglecting the input uncertainties, we obtain results that are overconfident around biased values;
– when trying to directly learn the dependence structure between β and $\mathbf{y}$ using a KDE-based approach, the calibration results are a bit too conservative, in the sense that the area of the 95% confidence ellipse is too big.

However, we notice strong differences for the other cases. First, the results associated with the linearization of the code are much less accurate this time. And not surprisingly, poor results are also obtained by considering an error term whose covariance function depends on this linearization. When introducing general covariance functions (see $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$) to characterize this error term, not really satisfying results are obtained, in the sense that the true value of β does not belong to the 95% confidence ellipses. This may be due to an amplification phenomenon from the input uncertainties to the final output uncertainty, which can also be attributed to the non-linearity of $\mathbf{x} \mapsto g(\mathbf{x};\mathbf{b})$.

Looking at the ellipses associated with $f^{\text{ref}}_{\beta|\mathbf{y}}$, we notice that considering a non zero wind speed adds another source of uncertainty, and therefore increases the uncertainty on the value of β. Without surprise, this new source of uncertainty does not help $f^{\text{without}}_{\beta|\mathbf{y}}$, $f^{\text{lin}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,diff}}_{\beta|\mathbf{y}}$ to be closer to $f^{\text{ref}}_{\beta|\mathbf{y}}$. However, we notice an important increase in the distance between $f^{\text{GP,1D}}_{\beta|\mathbf{y}}$ and $f^{\text{ref}}_{\beta|\mathbf{y}}$. This is also not surprising : by not taking into account the presence of wind, the GP-based calibration underestimates the uncertainty on the value of β. On the contrary, the wind speed variability being integrated in the construction of $f^{\text{KDE}}_{\beta|\mathbf{y}}$, $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$, better results are obtained for these three approaches. Whereas $f^{\text{KDE}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$ seem to correctly capture the true covariance between the components of β while being a bit too conservative, $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$ allows us to construct a 95% confidence ellipse that contains the true value of β.

To go further, Figure 7 compares, for $n \in \{1,...,5\}$, the 95% prediction intervals for $g(\mathbf{x}_n;\beta)$ associated with the different identified posterior distributions of β in the case where the wind speed is not zero. In order to focus on the influence of the uncertainty associated with β only, a fixed realization of the wind speed has been chosen for all the simulations to compute the results

 ISSN: 2102-6238



(a) Absence of wind (b) Absence of wind

(c) Presence of wind (d) Presence of wind

FIGURE 6. *In each figure, the red point indicates the position of* $\beta^\star$*, and the ellipses correspond to* 95% *confidence ellipses, in the sense that the integration of each posterior PDF over the domain corresponding to its ellipse is near 0.95. In each figure, the reference ellipse is represented in red thick continuous line. In the left figures, the ellipses associated with the case where the input uncertainties are not taken into account is represented in black thin continuous line, the GPR-based approximations correspond to the green twodashed line ellipses, and the KDE-based approximations correspond to the blue thick dashed line ellipses. In the right figures, the approximations based on a linearization of the code correspond to the green twodashed line ellipses, and the approximations based on the model error formalism are shown in black thin continuous line for* $f^{mod,diff}_{\beta|\mathbf{y}}$*, in blue thick dashed line for* $f^{mod,Matern}_{\beta|\mathbf{y}}$*, and in cyan dotted line for* $f^{mod,nugget}_{\beta|\mathbf{y}}$*.*

 ISSN: 2102-6238

of Figure 7. In this series of figures, the grey horizontal line corresponds to the value of $g(\mathbf{x}_n;\beta^\star)$, the red dotted lines are the upper and lower bounds of the credible interval associated with $f^{\text{ref}}_{\beta|\mathbf{y}}$, and the other cases correspond to $f^{\text{without}}_{\beta|\mathbf{y}}$, $f^{\text{GP,1D}}_{\beta|\mathbf{y}}$, $f^{\text{KDE}}_{\beta|\mathbf{y}}$, $f^{\text{mod,diff}}_{\beta|\mathbf{y}}$, $f^{\text{lin}}_{\beta|\mathbf{y}}$, $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$. The same conclusions can be made: whereas the results associated with $f^{\text{without}}_{\beta|\mathbf{y}}$, $f^{\text{lin}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,diff}}_{\beta|\mathbf{y}}$ are not relevant for this application, $f^{\text{GP,1D}}_{\beta|\mathbf{y}}$, $f^{\text{KDE}}_{\beta|\mathbf{y}}$, $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$ provide interesting results for the prediction of $g(\mathbf{x}_n;\beta^\star)$. More precisely, as it was previously observed, $f^{\text{GP,1D}}_{\beta|\mathbf{y}}$ tends to underestimate the uncertainties, when $f^{\text{KDE}}_{\beta|\mathbf{y}}$ and $f^{\text{mod,nugget}}_{\beta|\mathbf{y}}$ tend to overestimate them. At last, $f^{\text{mod,Matern}}_{\beta|\mathbf{y}}$ provides results that are a little bit biased compared to the ones of $f^{\text{ref}}_{\beta|\mathbf{y}}$.

5. Conclusion

In this work, two examples were presented to highlight the importance of properly taking into account input uncertainties for the calibration of parametric simulators, in the sense that neglecting these uncertainties can lead to predictions that are too confident around biased values. However, this integration quickly becomes extremely difficult in terms of computation time when confronted with time-consuming simulators. Approximations have therefore been presented in this paper to obtain satisfactory results at a reasonable computation time. For example, if the amplitude of input uncertainties is low, working with an approximate likelihood based on a linearization of the code output presents a good compromise between accuracy and resolution efficiency. Non-linear approximations of code output based on Gaussian Process Regression (GPR) also provide interesting solutions to the inference problem, at a controlled computational cost. However, when the system parameters gather high-dimensional vectors or functions, it may not be possible to construct such approximations. In that case, it has been shown that working directly on the dependency structure between the parameters to be calibrated and the code outputs, using kernel density estimates for example, presents an interesting alternative, even if the associated computational cost is generally higher.

All these developments are based on the assumption that there are significant and relatively well known input uncertainties. If information on these input uncertainties were missing, conventional statistical approaches would capture this discrepancy between model predictions and observational data by adding to the quantities of interest an external error term with an *ad hoc* covariance structure. This is clearly suboptimal, as the error term will not take into account the physical constraints imposed by the model. Proposing covariance functions that are better adapted to the input-output relationship is therefore an interesting perspective for this work. Similarly, the adaptation of the integrated model error representations presented in Sargsyan et al. (2018) to the case of input uncertainties is a possible extension to this work.

References

Berliner, L. M. (2001). Monte carlo based ensemble forecasting. *Statistics and Computing*, 11.

Bilionis, I. and Zabaras, N. (2015). Bayesian uncertainty propagation using gaussian processes. *In: Ghanem R., Higdon D., Owhadi H. (eds) Handbook of Uncertainty Quantification. Springer.*

Chen, P. and Schwab, C. (2015). Sparse-grid, reduced basis bayesian inversion. *Computer Methods in Applied Mechanics and Engineering*, 297:84–115.

 ISSN: 2102-6238

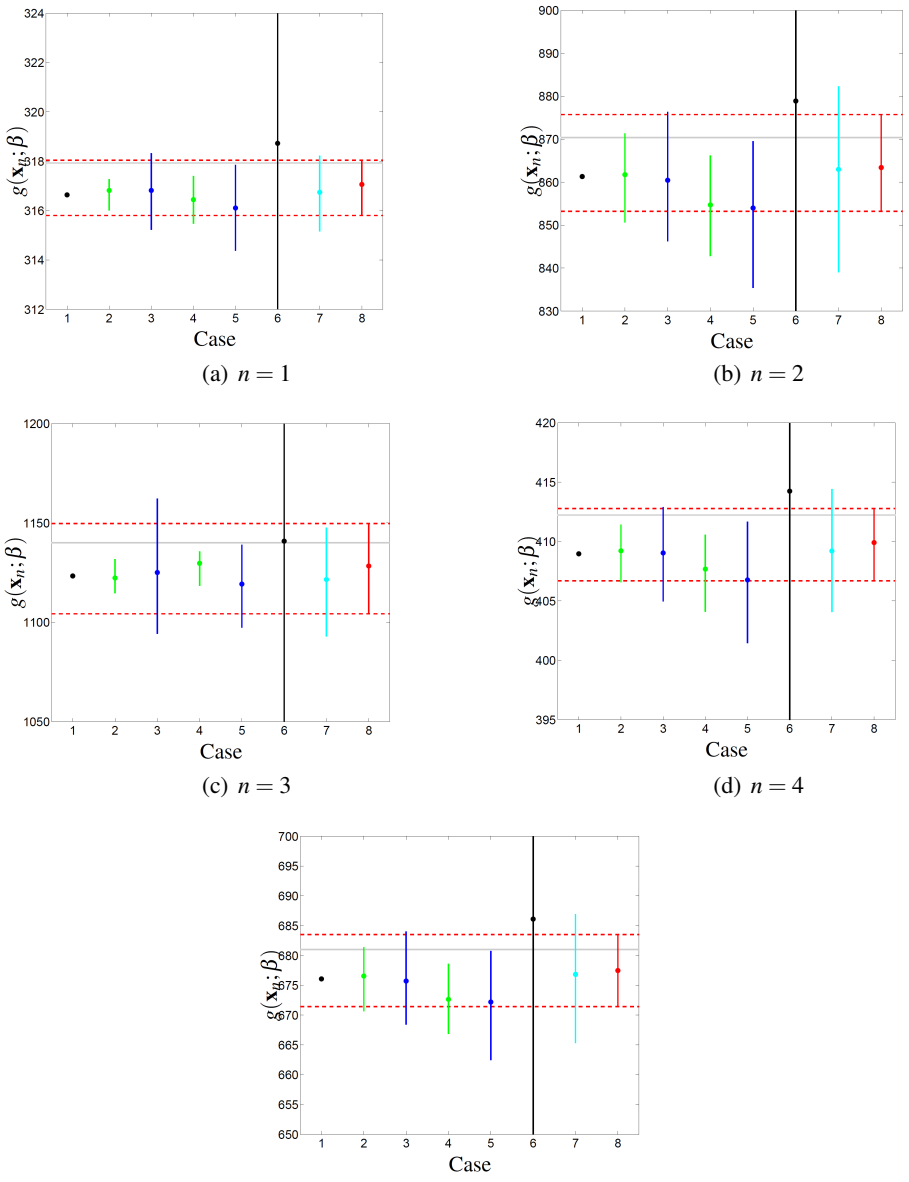


FIGURE 7. *Prediction of the code output in different values of* $\mathbf{x}$ *using several approximations of* $f^{ref}_{\beta|\mathbf{y}}$. *In each figure, the grey continuous line corresponds to the true value of the code output and the two red dashed lines are the upper and lower bound of the* 95% *prediction interval associated with reference posterior PDF* $f^{ref}_{\beta|\mathbf{y}}$. *Then, for each case, the point corresponds to the median of* $g(\mathbf{x}_n;\beta)$ *and the vertical interval defines a* 95% *prediction interval for* $g(\mathbf{x}_n;\beta)$. *Case 1* $\leftrightarrow f^{without}_{\beta|\mathbf{y}}$. *Case 2* $\leftrightarrow f^{GP,1D}_{\beta|\mathbf{y}}$. *Case 3* $\leftrightarrow f^{KDE}_{\beta|\mathbf{y}}$. *Case 4* $\leftrightarrow f^{lin}_{\beta|\mathbf{y}}$. *Case 5* $\leftrightarrow f^{mod,Matern}_{\beta|\mathbf{y}}$. *Case 6* $\leftrightarrow f^{mod,diff}_{\beta|\mathbf{y}}$. *Case 7* $\leftrightarrow f^{mod,nugget}_{\beta|\mathbf{y}}$. *Case 8* $\leftrightarrow f^{ref}_{\beta|\mathbf{y}}$.

 ISSN: 2102-6238

Conrad, P. R., Davis, A., Marzouk, Y. M., Pillai, N. S., and Smith, A. (2018). Parallel local approximation MCMC for expensive models. *SIAM/ASA J. Uncertainty Quantification*, 6(1):39–373.

Conrad, P. R., Marzouk, Y. M., Pillai, N. S., and Smith, A. (2016). Accelerating asymptotically exact MCMC for computationally intensive models via local approximations. *Journal of the American Statistical Association*, 111:1591–1607.

Damblin, G., Barbillon, P., Keller, M., Pasanisi, A., and Parent, E. (2013). Adaptive Numerical Designs for the Calibration of Computer Codes. *SIAM/ASA J. Uncertainty Quantification*, 6(1):151–179.

Fielding, M., Nott, D. J., and Liong, S. Y. (2011). Efficient MCMC schemes for Computationally Expensive Posterior Distributions. *Technometrics*, 53(1):16–28.

Higdon, D., Gattiker, J., Williams, B., and Rightley, M. (2008). Computer model calibration using high-dimensional output. *Journal of the American Statistical Association*, 103(482):570–583.

Higdon, D., Lee, H., and Holloman, C. (2003). Markov chain monte carlo based approaches for inference in computationally intensive inverse problems. *Bayesian Statistics*, 7:181–197.

Kaipio, J. P. . and Somersalo, E. (2004). *Statistics and Computational Inverse Problems*. Springer, New York.

Kennedy, M. and O'Hagan, A. (2001). Bayesian calibration of computer models. *Journal of the royal statistical society*, 63:425–464.

Li, J. and Marzouk, Y. M. (2014). Adaptive construction of surrogates for the bayesian solution of inverse problems. *SIAM Journal on Scientific Computing*, 36:A1163–A1186.

Liu, F., Bayarri, M., and Berger, J. (2009). Modularization in bayesian analysis, with emphasis on analysis of computer models. *Bayesian Analysis*, 4(1):119–150.

Marin, J. M. . and Robert, C. P. (2007). *Bayesian core*. Springer-Verlag, New York.

Marzouk, Y. M. and Najm, H. N. (2009). Dimensionality reduction and polynomial chaos acceleration of bayesian inference in inverse problems. *Journal of Computational Physics*, 228 (6):1862–1902.

Marzouk, Y. M. and Xiu, D. (2009). A stochastic collocation approach to bayesian inference in inverse problems. *Communications in Computational Physics*, 6:826–847.

Perrin, G. (2019). Adaptive calibration of a computer code with time-series output. *submitted to Reliability Engineering and System Safety*.

Perrin, G. and Cannamela, C. (2017). A repulsion-based method for the definition and the enrichment of opotimized space filling designs in constrained input spaces. *Journal de la Société Française de Statistique*, 158(1):37–67.

Perrin, G., Soize, C., and Ouhbi, N. (2018). Data-driven kernel representations for sampling with an unknown block dependence structure under correlation constraints. *Journal of Computational Statistics and Data Analysis*, 119:139–154.

Rasmussen, C. E. (2003). Gaussian processes to speed up hybrid monte carlo for expensive bayesian integrals. *Bayesian Statistics*, 7:651–659.

Rubinstein, R. T. and Kroese, D. (2008). *Simulation and the Monte Carlo method*. John Wiley and Sons, Inc., Hoboken, New Jersey.

Santner, T. J., Williams, B., and Notz, W. (2003). *The design and analysis of computer experiments*. Springer, New York.

Sargsyan, K., Huan, X., and Najm, H. (2018). Embedded model error representation for Baysian model calibration. *arXiv:1801.06768v1*.

Scott, D. W. and Sain, S. R. (2004). Multidimensional Density Estimation.

Sinsbeck, M. and Nowak, W. (2017). Sequential Design of Computer Experiments for the Solution of Bayesian Inverse. *SIAM/ASA J. Uncertainty Quantification*, 5:640–664.

Tian, M., Li, D., Cao, Z., Phoon, K., and Wang, Y. (2016). Bayesian identification of random field model using indirect test data. *Engineering Geology*, 210:197–211.

Tsilifis, P., Ghanem, R. G., and Hajali, P. (2017). Efficient Bayesian Experimentation Using an Expected Information Gain Lower Bound. *SIAM/ASA J. Uncertainty Quantification*, 5:30–62.

Wan, J. and Zabaras, N. (2011). A Bayesian approach to multiscale inverse problems using the sequential Monte Carlo method. *Inverse Problems*, 27.

Wand, M. P. and Jones, M. C. (1995). Kernel Smoothing. *Encyclopedia of Statistics in Behavioral Science*, 60(60):212.

 ISSN: 2102-6238