# Chapter 4

# SMUQ: A tool for quantifying the uncertainties of the unobservable quantities

## 4.1 Introduction

It is common practice in engineering to attest the validity of a computational model by merely comparing its outputs with the experiments. Such process has evident shortcomings; it relegates, indeed, the possibility of establishing the consistency between simulations and the real world only to cases in which an observable data is available. The different and more general approach to validation that will be presented in the first part of this chapter (Sec. 4.2) is based on a proper representation of the model uncertainty that follows the recent paper [24] by Oliver *et al.* The methods that have been used in order to solve the related calibration problem are based on the Bayesian Inference and. in particular, on a Delayed Rejection Adaptive Method applied to the Metropolis-Hasting Algorithm. All the adopted methodologies are illustrated in the second part of this unit (Sec.s 6.2, 4.4 and 4.5). Lastly (Sec. 4.6), the FORTRAN code SMUQ (Stochastic Modeling for Uncertainty Quantification) developed by the author of this thesis in order to implement the new strategy is presented in Sec. 4.6.

## 4.2 Predictive Problem Statement

In engineering, a solution to a problem is usually obtained by means of a model that is based on various physical, mathematical or empirical theories. Some of these theories are reliable within a domain of applicability, while others, even if are known to be less accurate, are the best choice in the description of a part of the phenomenon we are

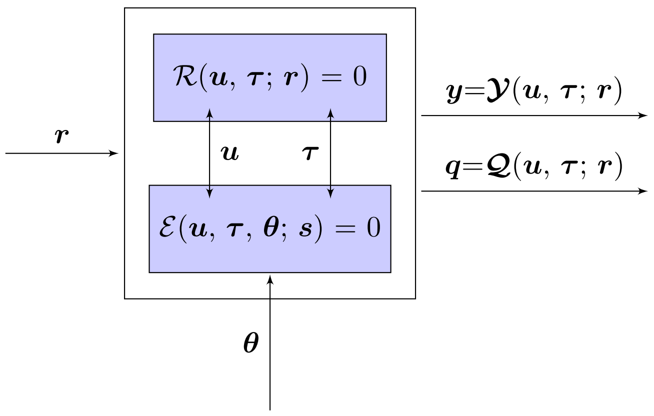

$\mathcal{R}(\boldsymbol{u},\,\boldsymbol{\tau};\,\boldsymbol{r}) = 0$
$\boldsymbol{r}$
$\boldsymbol{u}$
$\boldsymbol{\tau}$
$\mathcal{E}(\boldsymbol{u},\,\boldsymbol{\tau},\,\boldsymbol{\theta};\,\boldsymbol{s}) = 0$
$\boldsymbol{\theta}$
$\boldsymbol{y}{=}\boldsymbol{\mathcal{Y}}(\boldsymbol{u},\,\boldsymbol{\tau};\,\boldsymbol{r})$
$\boldsymbol{q}{=}\boldsymbol{\mathcal{Q}}(\boldsymbol{u},\,\boldsymbol{\tau};\,\boldsymbol{r})$

FIGURE 4.1: *Abstract problem without any model imperfection.*

interested into; this could be due, for example, either to the fact that more rigorous models are not available or to the fact that the computational cost of such meticulous models is not affordable. For this reason a generic prediction could be seen as the result of some reliable theories augmented with "embedded" models [24].
A generic reliable theory can be formulated as:

$$\mathcal{R}(\boldsymbol{u},\boldsymbol{\tau};\boldsymbol{r}) = 0. \tag{4.1}$$

where $\boldsymbol{u}$ is the state of the model, $\boldsymbol{r}$ is a set of scenario variables, $\mathcal{R}$ is the operator related to the theory and $\boldsymbol{\tau}$ represents a quantity that needs to be specified in order to solve the problem.
In most of the practical cases, a biunivocal correspondence between $\boldsymbol{\tau}$ and $(\boldsymbol{r},\,\boldsymbol{u})$ either does not exist or is very hard to be expressed. Therefore, the problem is not closed and the dependence of $\boldsymbol{\tau}$ on other variables need to be made explicit trough an embedded model:

$$\mathcal{E}(\boldsymbol{u},\boldsymbol{\tau},\boldsymbol{\theta};\boldsymbol{s}) = 0, \tag{4.2}$$

where $\boldsymbol{\theta}$ is a set of model parameters and $\boldsymbol{s}$ represent the set of scenario variables, which is either a subset of $\boldsymbol{r}$ $(\boldsymbol{s} \subseteq \boldsymbol{r})$ or an empty set.
Fig. 4.1 shows schematically such composite feature of the model. In spite the fact the observable quantities $\boldsymbol{y}$ and the Quantities of Interest $\boldsymbol{q}$ have all the same characteristic of being the model outputs, in the Figure they are clearly distinguished. This is because the observable quantities are defined to be all the outputs of the model that can be compared with some available data, while no data is accessible for directly inspecting the QoI. The consequences of this difference will be analyzed later in this Section. In any case, $\boldsymbol{y}$ and $\boldsymbol{q}$ are all univocally determined by the model states $\boldsymbol{u}$, the scenario parameters $\boldsymbol{r}$ and the embedded model quantities $\boldsymbol{\tau}$

### 4.2.1 Kennedy and O'Hagan's approach to model validation

The description of the abstract problem discussed above relies on the assumption that the prediction made through the embedded model is not affected by any error. This condition is hard to satisfy. Indeed, failure of the model in delivering an accurate prediction can be due to, firstly, the lack of knowledge on the parameters $\boldsymbol{\theta}$; secondly, the model could be affected by a structural inadequacy, namely its functional form could be not correct. This results in the generation of approximate values for the embedded model variables, $\boldsymbol{\tau}_m \approx \boldsymbol{\tau}$, and the error so created will be propagated to the observable quantities and to the QoIs.
Kennedy and O'Hagan proposed a solution for taking into account "the difference between the true value of the real world process and the code output at the true values of the inputs" [30]; they opted for adding the outputs of some stochastic models to the computational results, as in Fig. 4.2:

$$\boldsymbol{y} = \boldsymbol{\mathcal{Y}}(\boldsymbol{u}, \boldsymbol{\tau}_m; \boldsymbol{r}) + \boldsymbol{\delta}_y(\boldsymbol{\alpha}; \boldsymbol{r}), \tag{4.3}$$

where $\boldsymbol{\alpha}$ is a set of hyperparameters and $\boldsymbol{\delta}_y$ is the output of a stochastic model ($SM_1$), created to represent the error in predicting the observable quantities due to the structural uncertainty. The usefulness of such stochastic description will be clarified later in this chapter.
What needs to be underlined in this context, however, is the fact that such characterization of the model error is unable to propagate the structural uncertainty to the quantities of interest. In fact, while it is possible to take advantage of the data for "tuning" the hyperparameters $\boldsymbol{\alpha}$ so that $\boldsymbol{y} \sim \boldsymbol{D}$, it is not possible to do the same for $\boldsymbol{\beta}$; there is no available data, indeed, for comparing the QoIs.
For this reason, we need a second stochastic model ($SM_2$) in representing the consequences of the model inconsistency on the QoI:

$$\boldsymbol{q} = \mathcal{Q}(\boldsymbol{u}, \boldsymbol{\tau}_m; \boldsymbol{r}) + \boldsymbol{\delta}_q(\boldsymbol{\beta}; \boldsymbol{r}). \tag{4.4}$$

### 4.2.2 Oliver's approach to model validation

Kennedy and O'Hagan's approach clearly has some limitations. First, $\boldsymbol{\beta}$ is a set of hyperparameters that needs to be calibrated together with $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$, but, differently from the latter, no data is accessible for such operation. Second, even assuming that

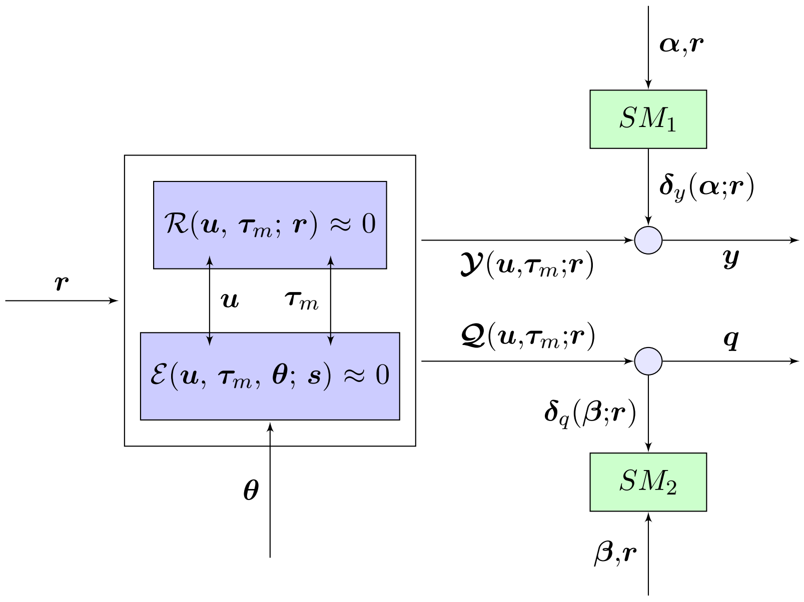
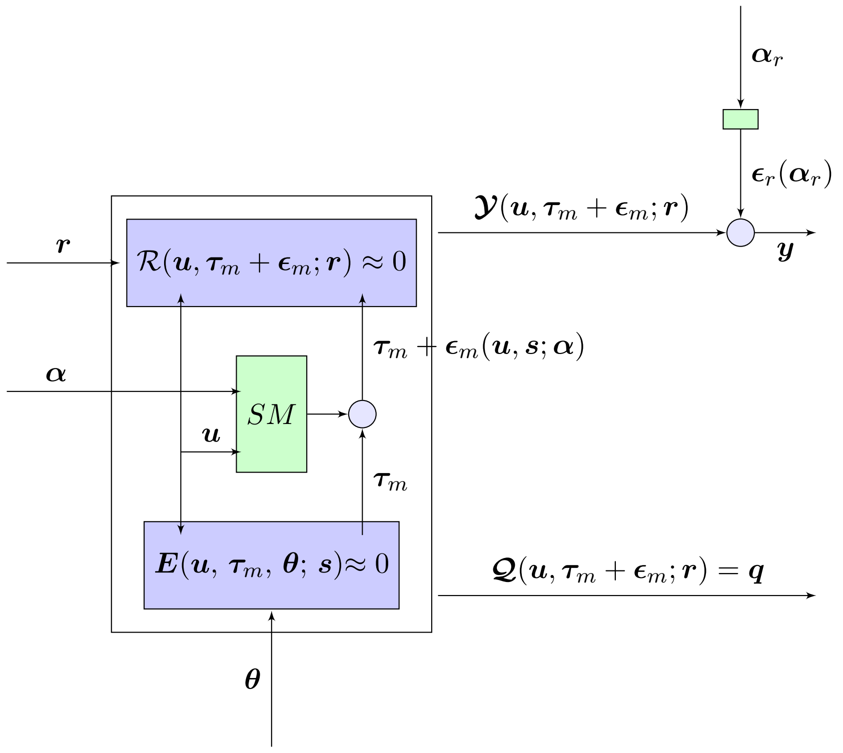

$\boldsymbol{\alpha}$,$\boldsymbol{r}$
$SM_1$
$\boldsymbol{\delta}_y(\boldsymbol{\alpha};\boldsymbol{r})$
$\mathcal{R}(\boldsymbol{u},\, \boldsymbol{\tau}_m;\, \boldsymbol{r}) \approx 0$
$\boldsymbol{\mathcal{Y}}(\boldsymbol{u},\boldsymbol{\tau}_m;\boldsymbol{r})$
$\boldsymbol{y}$
$\boldsymbol{r}$
$\boldsymbol{u}$
$\boldsymbol{\tau}_m$
$\boldsymbol{\mathcal{Q}}(\boldsymbol{u},\boldsymbol{\tau}_m;\boldsymbol{r})$
$\boldsymbol{q}$
$\mathcal{E}(\boldsymbol{u},\, \boldsymbol{\tau}_m,\, \boldsymbol{\theta};\, \boldsymbol{s}) \approx 0$
$\boldsymbol{\delta}_q(\boldsymbol{\beta};\boldsymbol{r})$
$SM_2$
$\boldsymbol{\theta}$
$\boldsymbol{\beta}$,$\boldsymbol{r}$

FIGURE 4.2: *Kennedy-O'Hagan's representation of the model uncertainty.*

some strategy for obtaining $\boldsymbol{\beta}$ can be implemented, there is no way for testing $\boldsymbol{\delta}_q(\boldsymbol{\beta};\boldsymbol{r})$; in order to validate the model, in fact, only the observable quantities $\boldsymbol{y}$ are compared to data and $\boldsymbol{\beta}$ does not affect $\boldsymbol{y}$ in any way. Thus, a different approach is required.
Oliver *et al.* proposed in their paper "to take advantage of the structure of the composite model to introduce model uncertainty representation that can be informed and tested using data" [24]. The key point of such approach is to represent the structural error at its actual source, so that the uncertainty modeling can influence both the observed quantity and the QoI simultaneously (Fig. 4.3). The composite structure of the model allows to attribute the structural discrepancy only to $\boldsymbol{\tau}_m$; if some of the origins of such discrepancy (e.g., the physical and mathematical approximations and phenomena lack of knowledge) are known, one can represent the recognize sources of uncertainty through a stochastic model characterized by some hyperparameters $\boldsymbol{\alpha}$. In this way, the error that in Kennedy and O'Hagan's approach has been simply added to the output can be now partially moved upstream to augment the embedded physical model:

$$\boldsymbol{\tau} \approx \boldsymbol{\tau}_m(\boldsymbol{u},\boldsymbol{\theta};\boldsymbol{s}) + \boldsymbol{\epsilon}_m(\boldsymbol{u},\boldsymbol{\alpha};\boldsymbol{s}). \tag{4.5}$$

The method for obtaining $\boldsymbol{\epsilon}_m$ is problem-dependent and "is driven by physical knowledge about the nature of error as well as practical considerations necessary to make computations with the model tractable" [24]. Anyway, such stochastic contribution will be propagated downstream to both $\boldsymbol{y}$ and $\boldsymbol{q}$; because of that, one can acquire information about $\boldsymbol{\alpha}$ from data and then transfer that knowledge directly to the QoI through the prediction. The only contribution that does not affect the Quantity of Interest is $\epsilon_r$, the residual part of the error, generated through the Kennedy and O'Hagan's stochastic



$\boldsymbol{\alpha}_r$
$\boldsymbol{\epsilon}_r(\boldsymbol{\alpha}_r)$
$\boldsymbol{\mathcal{Y}}(\boldsymbol{u}, \boldsymbol{\tau}_m + \boldsymbol{\epsilon}_m; \boldsymbol{r})$
$\boldsymbol{y}$
$\boldsymbol{r}$
$\mathcal{R}(\boldsymbol{u}, \boldsymbol{\tau}_m + \boldsymbol{\epsilon}_m; \boldsymbol{r}) \approx 0$
$\boldsymbol{\tau}_m + \boldsymbol{\epsilon}_m(\boldsymbol{u}, \boldsymbol{s}; \boldsymbol{\alpha})$
$\boldsymbol{\alpha}$
$SM$
$\boldsymbol{u}$
$\boldsymbol{\tau}_m$
$\boldsymbol{E}(\boldsymbol{u}, \boldsymbol{\tau}_m, \boldsymbol{\theta}; \boldsymbol{s}) \approx 0$
$\boldsymbol{\mathcal{Q}}(\boldsymbol{u}, \boldsymbol{\tau}_m + \boldsymbol{\epsilon}_m; \boldsymbol{r}) = \boldsymbol{q}$
$\boldsymbol{\theta}$

FIGURE 4.3: *Representation of the model uncertainty proposed by Oliver et al.*

term added to the output, governed by the hyperparameters $\alpha_r$. Such uncertainty takes into account the physical and mathematical aspects of the phenomena of interest that we have not been able to capture through the stochastic model used for augmenting the embedded model. The more this last stochastic model is physically-based, the more the Kennedy and O'Hagan's is reduced, the more the prediction of the QoI is reliable.

At this point, it is important to motivate the choice of a stochastic representation for the model inadequacy even if the physical phenomenon being described is deterministic. We should start with underlying the different nature of the hyperparameters $\boldsymbol{\alpha}$ and the parameters $\boldsymbol{\theta}$, which is consequence of the different hierarchical levels which they belong to. In the most simple terms, while a single set of $\boldsymbol{\theta}$ truly represent the parameters of a particular model, a set of $\boldsymbol{\alpha}$ can be used to generate $n$ different sets of parameters.

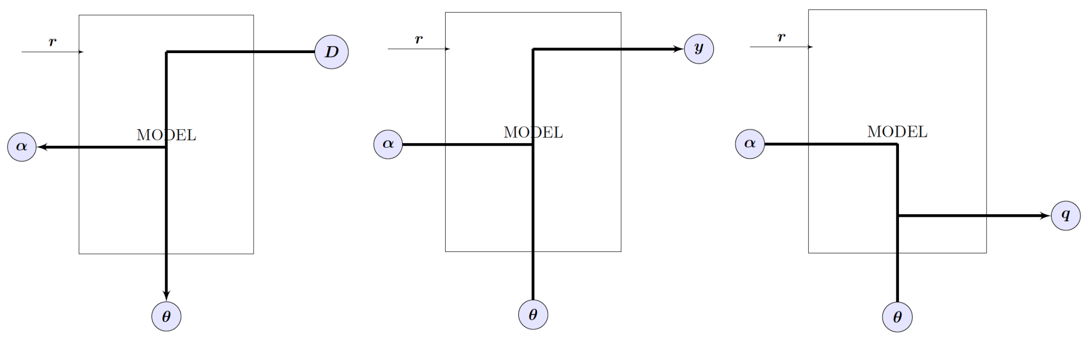
For example, assuming that a particular parameter $\gamma$ appears in our model, we can decide to characterize it in a deterministic way; this would bring us to calibrate directly the value of $\gamma$ (that is, $\gamma \in \boldsymbol{\theta}$). On the other hand, we could represent $\gamma$ in a stochastic way by means of a normal distribution: $\gamma = \mathcal{N}(\mu_\gamma, \sigma_\gamma)$. In such case, we must calibrate the hyperparamters $\mu_\gamma$ and $\sigma_\gamma$ instead of calibrating directly $\gamma$ (that is, $\mu_\gamma, \sigma_\gamma \in \boldsymbol{\alpha}$); it is clear that from a single couple of values of such mean and such standard deviation one can generate $n$ different values for the parameter $\gamma$. Thus, the operation of tuning used for computing $\gamma$ has a different meaning compared to the connotation of calibrating the hyperparameters $\mu_\gamma$ and $\sigma_\gamma$; this is because the former case is based on the assumption

that a unique true value of the parameters set has to be determined.
In the approach that we are considering for representing the model error, we are separating the contribution due to the uncertainty on the parameters from the the contribution that comes from the structural inadequacy; such goal is achieved by considering $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ on different hierarchical levels. The stochastic model $\boldsymbol{\epsilon}_m$ represents the connection between such levels, and it is what allows a proper representation of the structural inadequacy: even in the hypothetical situation where we know the best values for all $\boldsymbol{\theta}$, we would still be able to describe the uncertain nature of the QoIs thanks to the several values of parameters that have been generated by the stochastic model from a single set of $\boldsymbol{\alpha}$.

In this paragraph it has been pointed out that a proper representation of the structural error as in Fig. 4.3 is worthy in order to make a reliable prediction. The next two Sections, therefore, will be focused on providing a methodology that takes advantage on such representation for quantifying the uncertainty in the prediction of the Quantities of interest.

## 4.3 Parameters Sensitivity Analysis and Parameters Calibration

As mentioned before, the available data plays an essential role in the model validation; indeed, the comparison of such experimental results with the different outcomes of the model allows to extrapolate information about the uncertainty on the QoIs.
Fig. 4.3 highlighted all the factors that could affect the computational output: the scenarios variables $\boldsymbol{r}$, the embedded model parameters $\boldsymbol{\theta}$ and the stochastic model hyperparameters $\boldsymbol{\alpha}$. That means that, once the functional forms are defined for the embedded and the stochastic models, there is no way for obtaining different outcomes $\boldsymbol{y}$ but changing at least one of the values of $\boldsymbol{r}$, $\boldsymbol{\theta}$, $\boldsymbol{\alpha}$. Moreover, $\boldsymbol{r}$ is dictated by the scenario that one is interested into and it is assumed to be known without any error; that means that our degrees of freedom are restricted to parameters and hyperparameters. The computational model, therefore, can have different outcomes only varying $\boldsymbol{\theta}$, $\boldsymbol{\alpha}$.
With this in mind, the problem of making reliable predictions could be seen as the problem of finding the proper ranges in which each one of the parameters / hyperparameters can vary without generating outputs $\boldsymbol{y}$ that are too different from the data values $\boldsymbol{D}$. Problems of this type are called calibration problems.
When an output should be considered too different from the data will be clarified in the following Sections; what should pointed out here, anyway, is the basic assumption



(a) Calibration (b) Validation (c) Prediction

FIGURE 4.4: *The process of making reliable predictions for unobservable quantities.*

that has been made: if $\boldsymbol{y}$ is the output of the model that has a particular set of $\boldsymbol{\theta}, \boldsymbol{\alpha}$ as inputs and $\boldsymbol{y}$ is "comparable" to $\boldsymbol{D}$, then the QoIs $\boldsymbol{q}$ obtained with the same set of parameters will be "similar" to their true values for the same scenario conditions, and the prediction must be considered reliable. Such statement relies on the hypothesis that through the creation of the stochastic model we encapsulated all the main sources of uncertainty that can affect $\boldsymbol{y}$ and $\boldsymbol{q}$.

The model parameters, thus, can be seen a sort of "accumulators of information" that can be "charged" by the data during the calibration process; such accumulators can then "release" all the information that they stored to both the observable quantities (for validating the model) and to the Quantities of Interest (for making predictions) (Fig. 4.4).

This analogy is also useful for underlining that different parameters could be equipped with different "capacities" of accumulating the information that comes from the data. The model, in fact, could be more sensitive to a particular $\theta_i$ (or to a $\alpha_i$) than to a different $\theta_j$ ($\alpha_j$); moreover, it could happen that a generic $\theta_i$, even if good for storing the information that comes from the data, is not capable to transmit such knowledge to any of the $\boldsymbol{q}$. Sensitivity analysis is the proper tool for assessing which subset of the available $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ has an active role in making a reliable prediction and it can be of vital importance in the first part of the validation process. Indeed, solving an inverse problem is generally complicated and a minimization of the dimensions of the space, where one is looking for a possible solution would consequently reduce such complexity. Model reduction can be achieved by means of ranking $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ based on their correlation with the QoIs and subsequently discarding the unessential parameters and hyperparameters [14].

## 4.4 Calibration and Uncertainty Quantification by means of Bayesian Inference

Calibration belongs to a broader category of problems, the so called inverse problems, that aim at determining the model inputs from which the known observable quantities have been generated. In the last century, different approaches has been developed for solving the inverse problems [31]; one of the most efficient method is the Bayesian Inference, that relies on Bayes' theorem.

### 4.4.1 Bayes' Theorem

Bayes' Theorem can easily be explained starting from the two assumptions that Cox pointed out in 1946 with the purpose of "providing a basis for the principles of probable inference" [32]. The starting point of his paper is the consideration that, in order to have a logical reasoning, the truth of a statement requires an explicit quantification. This can be guaranteed, at least in a transitive manner, by ranking with a real number each of the propositions, so that the most reliable one is associated with the highest value.
Consequently, Cox started to investigate what mathematical rules such real numbers must respect; by means of Boolean logic and algebra, he came to the conclusion that the consistency of reasoning can be satisfied only if all the propositions obey the two fundamental laws of the probability theory, that are the *sum rule* and the *product rule*. The former states that, if one specifies how much a hypothesis X could be true, then he is implicitly defining how much the same hypothesis X could be false. This rule can be expressed mathematically as:

$$p(X|M) + p(\bar{X}|M) = 1, \tag{4.6}$$

where $p(\ )$ is a generic Probability Density Function (PDF), the $\bar{X}$ represents the statement that X is false and M denotes the background information; it is fundamental to explicitly define M because probabilities are never absolute quantities and they always have to be related to the information at hand.
The second rule asserts that: when one specifies how much the proposition Y could be believed to be true and how much the proposition X could be believed to be true given that Y is true, he is also implicitly quantifying how much he believes that both X and Y are true:

$$p(X, Y|M) = p(X|Y, M)\ p(Y|M). \tag{4.7}$$

This two simple but effective rules have a number of different consequences, one of which is Bayes' theorem:

$$p(X|Y,M) = \frac{p(Y|X,M) \times p(X|M)}{p(Y|M)}. \tag{4.8}$$

This equation comes directly from the product rule, simply interchanging X with Y taking into account that $p(X,Y|M)$ can be commutated in $p(Y,X|M)$. Even if it is quite straightforward, this theorem is incredibly powerful: as stated by Sivia [33], "the importance of this property to data analysis becomes apparent if we replace X and Y by hypothesis and data":

$$p(\text{hypothesis} \mid \text{data}, M) p(\text{data} \mid \text{hypothesis}, M) \times p(\text{hypothesis} \mid M). \tag{4.9}$$

The strength of Bayes' theorem, indeed, is the fact that it permits to compute the probability that a hypothesis is correct given the data simply taking advantage of the probability that we would have observed the data if such hypothesis was true; this last quantity is definitely easier to determine.
It is important to note that the inequality in Eq. 4.8 has been replaced with a proportionality in Eq. 4.9. That is motivated by the fact that in a lot of practical applications (e.g., calibration) the omission of $p(\text{data}|M)$ does not represent a big issue, as it will be explained better in the next paragraph.

With the aim of showing the potential of Bayes' theorem in solving a generic inverse problem, we apply Eq. 4.9 to an abstract problem represented as in Subs. 4.2.2 and sketched as in Fig. 4.3.
The hypothesis that must be considered in this context is the fact that, if we prescribe a particular set of values $\boldsymbol{\theta}, \boldsymbol{\alpha}$ to the parameters and to the hyperparameters, the model outcome would be exactly the data ($\boldsymbol{y} = \boldsymbol{D}$). If we want to establish the truth of such hypothesis given the data, we can write:

$$\Pi(\boldsymbol{\theta}, \boldsymbol{\alpha} \mid \boldsymbol{D}, \boldsymbol{M}) \propto \ L(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{M}) \times p(\boldsymbol{\theta}, \boldsymbol{\alpha} \mid \boldsymbol{M}). \tag{4.10}$$

$p(\boldsymbol{\theta}, \boldsymbol{\alpha} \mid \boldsymbol{M})$ represents the probability of finding the proper set of values of parameters and hyperparameters taking advantage only of the knowledge antecedent to the new data acquisition (e.g., literature, old experiments, common sense, ...); for this reason this PDF is called prior distribution. If for each of the $\theta$ parameters and each of the $\alpha$ hyperparameters such information is supposed to be independent from the knowledge available for the other $\boldsymbol{\theta}, \boldsymbol{\alpha}$, the total prior can be computed by:

$$p(\boldsymbol{\theta}, \boldsymbol{\alpha} \mid \boldsymbol{T}) = \prod_{i=1}^{T} p(\theta_i \mid \boldsymbol{M}) \times \prod_{j=1}^{A} p(\alpha_j \mid \boldsymbol{M}). \tag{4.11}$$

The importance of the prior is in the fact that it makes the inverse problem being a process of uploading preexisting information instead of a mere creation of knowledge *ex novo*.

The other PDF on the right-hand side, $L(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{M})$, is called likelihood distribution and it represents the probability of obtaining an output equal to the data ($\boldsymbol{y} = \boldsymbol{D}$), given the model and its uncertainty.

We motivated in the last part of Sec. 4.2, that a stochastic representation must be coupled to the embedded model in order to make reliable predictions; thus, the presence of a second hierarchical level has to be taken into account when we want to compute the likelihood. If $\gamma_k$ is one parameter that originates from $\alpha_{1_k}, \alpha_{2_k}, ..., \alpha_{n_k}$ through the stochastic models $\epsilon_{m_k}(\boldsymbol{u}, \boldsymbol{s}; \alpha_{1_k}, \alpha_{2_k}, ..., \alpha_{n_k})$, a particular value of such $\gamma_k$ has a probability $p(\gamma_k \mid \alpha_r, ..., \alpha_s, \boldsymbol{M})$ of being generated. We can then rewrite the Likelihood in Eq. 4.10 as:

$$L(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{M}) = \int_{\boldsymbol{I_\gamma}} l(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{M}) \times \prod_{k=1}^{p} p(\gamma_k \mid \alpha_{1_k}, ..., \alpha_{n_k}, \boldsymbol{M}) \, d\boldsymbol{\gamma} \tag{4.12}$$

where $l(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{M})$ represents the probability of obtaining an output $\boldsymbol{y}$ such that $\boldsymbol{D} = \boldsymbol{y}$ when the values of $\boldsymbol{\theta}, \boldsymbol{\gamma}$ and $\boldsymbol{M}$ are prescribed. The assumption made here is that the intersection of the various sets of hyperparameters that generate each $\gamma_k$ is an empty set, and this results in the probabilities $p(\gamma_k \mid \alpha_{1_k}, ..., \alpha_{n_k}, \boldsymbol{M})$ being independent of each other.

Eq. 4.10 allows to conclude that Bayes' Theorem allows to solve the calibration problems in a probabilistic way that relies on the combination of two different states of information: the first state is a priori and independent of the measurements, while the second state is based on some observable quantities. Because of such probabilistic approach, the solution will be a Probability Density Function over the parameter space.

### 4.4.2 Markov Chain Monte Carlo

An analytical characterization of the PDF in Eq. 4.10 is possible only in the case in which the distribution has a very simple shape. Moreover, central estimators (e.g., mean and median) and estimators of dispersion (e.g., standard deviation) cannot be useful tools when the PDF is far from being unimodal. "For more general probability distributions, one needs to perform an extensive exploration of the model space. Except for problems with a very small number of dimensions, this exploration cannot be systematic as the number of required points grows too rapidly with the dimension of the space" (Tarantola [31]).

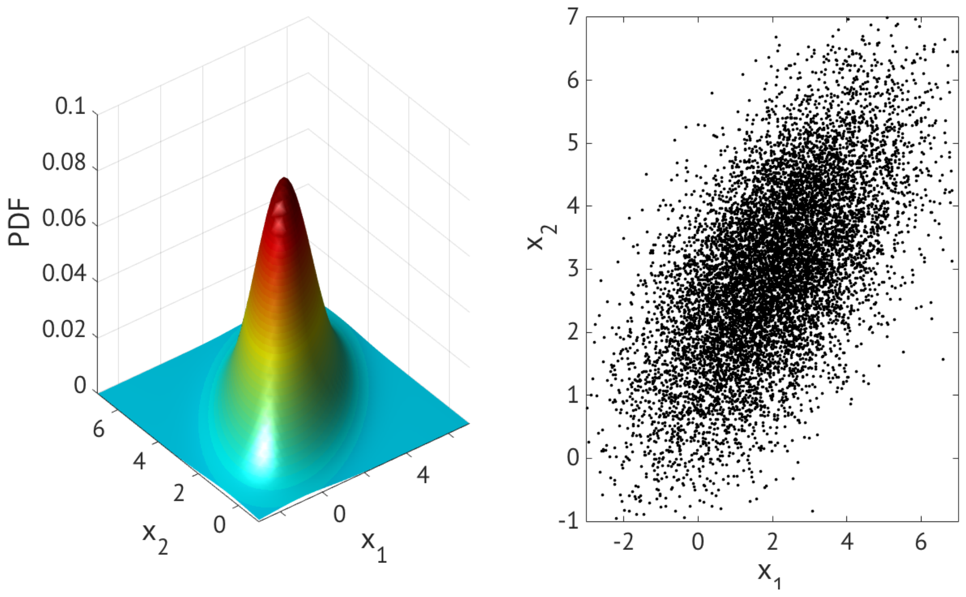


FIGURE 4.5: *10000 uncorrelated points obtained from sampling a multivariate normal distribution with* $\mu = [2, 3]$ *and* $\sigma = [3, 1.5; 1.5, 2]$.

Sampling the posterior overcomes the limitations due to the complexity of such distribution when solving the inverse problem. Sampling a generic PDF p (Fig. 4.5) means obtaining from p a set of independent points such that the probability of finding a point inside any domain $\mathcal{D}$ is equal to the probability of the domain $\mathcal{D}$ itself. In the last seventy years several random methods has been developed for efficiently characterizing a generic PDF by means of samples. A broad category of such methodology is the Monte Carlo (MC) sampling methods [34]; these approaches are based on the remark that high-dimensional spaces are inclined to be very empty; this fact has huge consequences on the effort that one has to address into obtaining some property of the distribution function. For example, computing the area of a circle by means of randomly throwing "darts" in points uniformly distributed over the cartesian plane is much simpler than retrieving the volume of a sphere by the equivalent approach in three dimensions. This is because, the probability of choosing a point randomly in the space and finding such point inside the n-dimensions hypersphere tends to zero very quickly when the number $n$ of the space dimensions increases. Due to the feature that has just been pointed out, solving such high dimensional problems on deterministic grids means wasting a lot of computational time.

The main aspects that differentiates the MC methods are the strategies that they adopt for obtaining efficient random grids.
The approach that has been extensively used in this thesis is a specific type of Monte Carlo methods, known as Markov Chain Monte Carlo (MCMC). Before introducing the

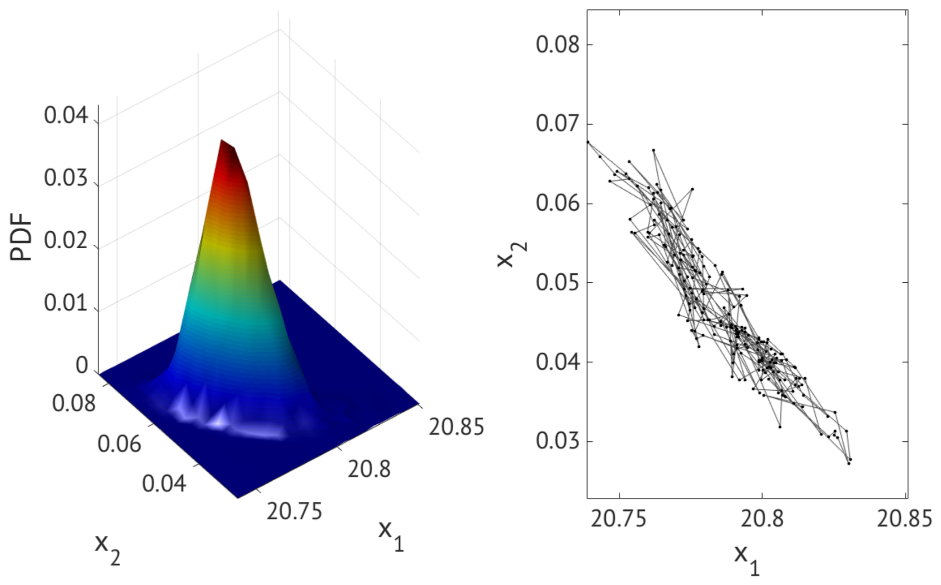


FIGURE 4.6: *A Markov chain random walk (b) for sampling 200 points from a generic PDF (a).*

definition of Markov chain, the notion of Markov process bust be recalled. A random variables X is a Markov process if at each generic time instant t the probability of a transition to a new state $s_j$ depends only on the current state $s_i = X_t$:

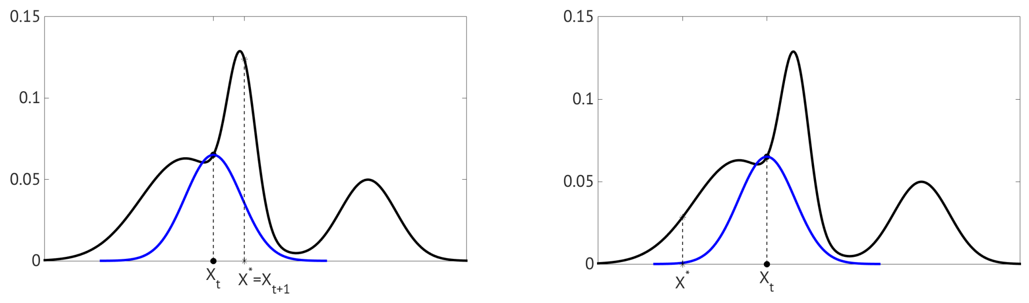
$$P(i \to j) = p(X_{t+1} = s_j \mid X_0 = s_0, ..., X_t = s_i) = p(X_{t+1} = s_j \mid X_t = s_i) \qquad (4.13)$$

A Markov chain is then a sequence of random draws $X_0, ..., X_n$ generated by a Markov process and characterized by a transition probability $P(i \to j)$.
Fig. 4.6 shows a Markov chain created in order to sample a particular PDF. It is possible to notice that such succession of points explores the space by means of a random walk, something similar to a Brownian motion.

### 4.4.3 Metropolis-Hasting Algorithm

As it has been pointed out each point of a Markov chain does not have memory of how the previous points have been collected. Such important feature has been exploited by Metropolis and Hasting in the algorithm that bears their name.
With the aim of sampling a PDF $p(X) = f(X)/C$, where C is a generic normalizing constant and $f(X)$ is another PDF, the Metropolis Algorithm produces a sequence of points with this iterative procedure:

1. Choose an initial value $X_0$ such that: $f(X_0) > 0$.



(a) A first possible jump. (b) Another possible jump.

FIGURE 4.7: *Metropolis Algorithm: Two candidate jumps for the PDF $P(X)$ (black lines) obtained from the current point $X_t$ by means of the same proposal $q(X_i, X_{i+1})$ (blue lines, not normalized). The $X^*$ proposed in (a) will certainly be accepted in the ($\alpha = 1.90$), while the $X^*$ in (b) will be rejected with a probability of 56% ($\alpha = 0.4370$)*

2. Sample a candidate draw $X^*$ from a proposal PDF q with the property of being symmetric (that is, a distribution such that the probability $q(X_i, X_{i+1})$ of moving from a point $X_i$ to a point $X_{i+1}$ is the same of the probability $q(X_{i+1}, X_i)$ of coming back to $X_{i+1}$ from $X_i$); then compute the probability $f(X^*)$ (Fig. 4.7).

3. Compute the ratio between the PDF $p(X)$ at the proposed point $X^*$ and at the current point $X_t$:
$$\alpha = \frac{p(X^*)}{p(X_t)} = \frac{f(X^*)}{C}\frac{C}{f(X_t)} = \frac{f(X^*)}{f(X_t)} \tag{4.14}$$

4. If $\alpha \geq 1$, then accept the candidate point (that is, set $X^* = X_{t+1}$).
If $\alpha < 1$, then accept the proposed point with probability $\alpha$ (e.g., if $\alpha = 0.9$, then nine times out of ten $X^*$ must be accepted to be $X_{t+1}$);
else reject $X^*$ and keep $X_t$ ($X_{t+1} = X_t$).

The Metropolis-Hasting Algorithm is a generalized version of the Metropolis procedure that allows to use an arbitrary (i.e.: not necessarily symmetric) proposal distribution $q(X_i, X_{i+1})$ by setting the acceptance probability for the proposed point as:

$$\alpha = min\left(1\ ,\ \frac{f(X^*)\ q(X^*, X_t)}{f(X_t)\ q(X_t, X^*)}\right) \tag{4.15}$$

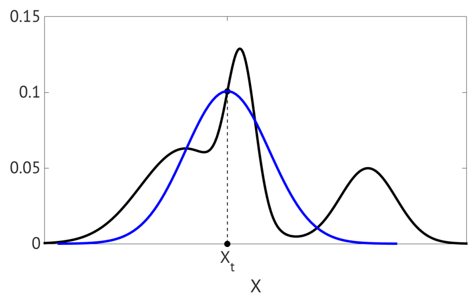
There are two key-ideas inside these algorithms: on the one hand the proposal helps to restrict the domain where to look for a candidate new point; on the other hand, the acceptance rules permits to give more importance to the regions of the space where the PDF is higher. The combination of these two features has important consequences on the performances of the Metropolis-Hasting Algorithms. In fact, it is possible to demonstrate that the Metropolis-Hasting sampling produces a Markov chain whose equilibrium

PDF is the distribution $P(X)$ that we want to characterize. Moreover, it could be mathematically proven that, between all the acceptance-based approaches, the Metropolis-Hasting Algorithms are the most efficient way for sampling a distribution [35]. This means that they generate the highest percentage of accepted proposed values without losing uncorrelation between the points of the sampling.

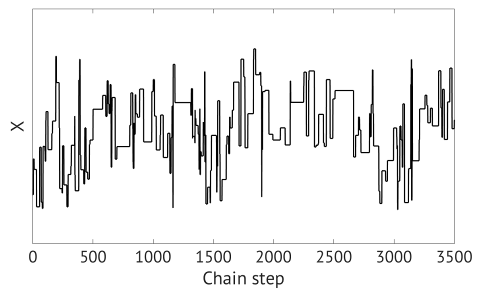
Another aspect makes the Metropolis-Hasting Algorithms to be a powerful tool for solving an inverse problem: thanks to the fact that they need only the ratio between two successive values of the target distribution, such algorithms overcome the complex computation of the constant of proportionality in the case the PDF that we want to be sampled is the posterior in Eq. 4.10.

A relevant remark at this point should be made with regard to a strategy for choosing the Proposal function. The Proposals in Fig. 4.8 are represented through three normal distributions with same mean $\mu_q$ but with three different $\sigma_{q_1}$ (Fig. 4.8(a)), $\sigma_{q_2} = \sigma_{q_1}/2$ (Fig. 4.8(c)) and $\sigma_{q_3} = \sigma_{q_1}/4$ (Fig. 4.8(e)).
The consequences that such heterogeneities have on the Markov chains are shown in Fig. 4.8(b), Fig. 4.8(d) and Fig. 4.8(f). It is easy to notice how a relatively high $\sigma_q$ causes a very poorly mixed chain (Fig. 4.8(b)); this must be related to the fact that the domain spanned by the proposal includes some widespread region in which $P(X)$ is zero; this is particularly true when the current value $X_t$ is at the borders of the target PDF. As consequence, several proposed draws $X^*$ will come out with a very low probability $\alpha$ of being accepted and the chain will spend a lot of time stuck in the current value $X_t$.
On the other hand, a relatively low $\sigma_q$ (Fig. 4.8(e)) produces a very slow random walk, due to the fact that the beneficial growth of the acceptance rate is accompanied by the decrease in size of the jump between two successive points. Thus, the proposed value $X^*$ will have a high probability of being accepted, but at the same time it will be very close to the current value $X_t$.
The last two examples underline that the percentage of accepted values is a good estimator for the quality of the chain mixing: low percentages are synonymous of poorly dense chain, while high percentages are symptomatic of a random motion that needs a huge amount of points for exploring the entire support of $p(X)$.
The best compromise for obtaining a well mixed chain (Fig. 4.8(d)) through a Metropolis-Hasting Algorithm is choosing a proposal with a medium variance, as in Fig. 4.8(c); such trade-off permits to obtain an accepting ratio of about 0.5.

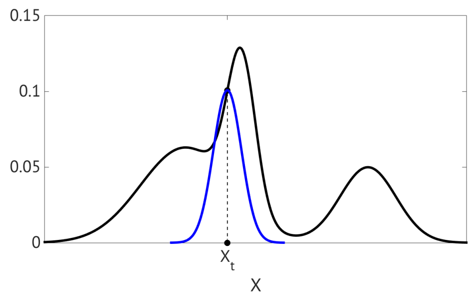
Independently of how one decides to represent the proposal distribution, a poor choice of the algorithm starting point can question the fast convergence of the chain to a stationarity distribution. Moreover, in case of a multimodal target PDF (like the $p(X)$ plotted in black in the previous figures) combined with a narrow proposal, moving from



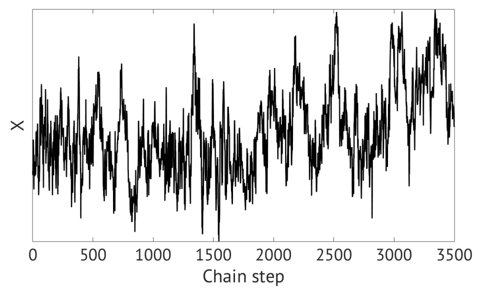
(a) Target (black line) and Proposal (blue line) PDFs.



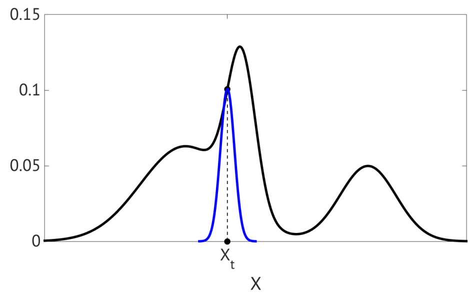
(b) Metropolis-Hasting chain relative to (a).



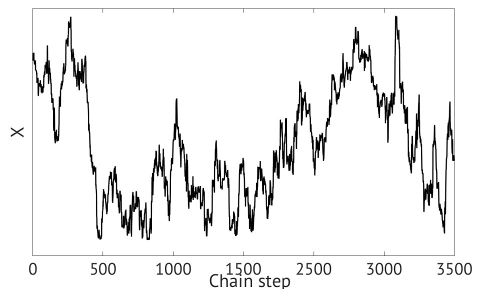
(c) Target (black line) and Proposal (blue line) PDFs.



(d) Metropolis-Hasting chain relative to (c).



(e) Target (black line) and Proposal (blue line) PDFs.



(f) PMetropolis-Hasting chain relative to (e).

FIGURE 4.8: *Metropolis Algorithm:*
*On the left column: three different proposal functions (blue lines):*
$q_1 = \mathcal{N}(\mu_q; \sigma_q = \sigma_{q1})$ *(a),* $q_2 = \mathcal{N}(\mu_q; \sigma_q = \sigma_{q1}/2)$ *(c),*
$q_3 = \mathcal{N}(\mu_q; \sigma_q = \sigma_{q1}/4)$ *(e).*
*On the right column: three different Metropolis-Hasting Markov chains produced by means of three different proposal functions, respectively: (b) from (a), (d) from (c) and (f) from (e).*

one mode region to another one becomes very hard and, as consequence, the chain could be poorly mixed.
Different solutions to such problems have been proposed in the last decades (e.g., Simulated Annealing approach [36]). However, a possible way for simply overcoming such issues is creating multiple chains that start from different highly dispersed initial values [37]. As we will see in the next Section, this technique is computationally particularly convenient when the algorithm has been parallelized on multiple processors and to each one of the cores has been assigned the realization of an independent chain.

All that has been shown in this Section for the generic variable X can be of course generalized for a vector of variables, $\boldsymbol{X} = \{X_1, X_2, ..., X_n\}$, in case the PDF $P(\boldsymbol{X})$ would be defined above a multidimensional space. For example, with the assumption of the proposal function $q_i(X_{i_t})$ for the i-th variable at computational time instant t being independent with respect to all the other n-1 proposals for the variables $X_{j_t}$, one can write the acceptance probability for the proposed point as:

$$\alpha = min\left(1\ ,\ \frac{f(X^*)\ q_1(X_1^*, X_{1_t})\ q_2(X_2^*, X_{2_t})\ ...\ q_n(X_n^*, X_{n_t})}{f(X_t)\ q_1(X_{1_t}, X_1^*)\ q_2(X_{2_t}, X_2^*)\ ...\ q_n(X_{n_t}, X_n^*)}\right) \tag{4.16}$$

and the sampling assumes the form of $n$ random walks, each one in a different dimension. However, as it will be explained in detail in the next Subsection, this could not be the best way for extending the Metropolis-Hasting Algorithms to a multidimensional space.

### 4.4.4 The Delayed Rejection Adaptive Method for MCMC

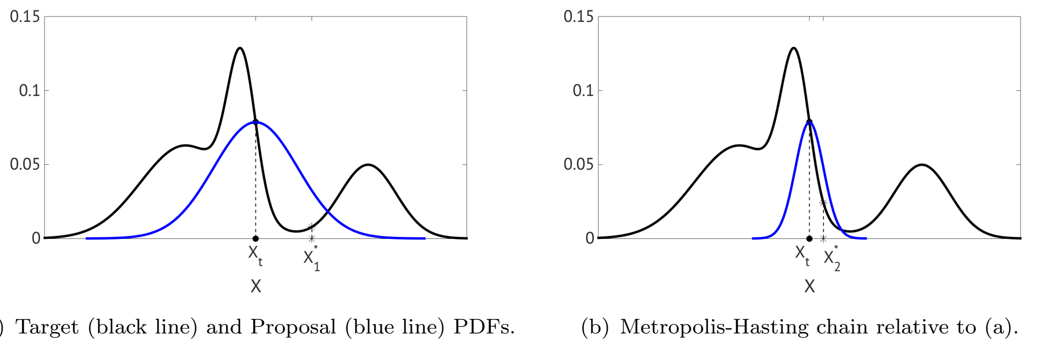
A number of different solutions has been developed for improving the Metropolis-Hasting Algorithm. In 2001 Green and Mira presented the Delayed Rejection Method (DR) [38], an approach that particularly helps when a proper representation of the proposal distribution is hard to find. We have seen from Fig. 4.8 that a Proposal with large standard deviation causes either to be stuck in the same point or to travel a long distance in moving to the new accepted value. While the first (very likely) case has negative consequences on the chain convergence chain, the second event is beneficial in order to have a fast exploration of the parameter space. DR approach follows from such consideration. The main idea is starting to propose a candidate draw $X_i^*$ produced by means of a distribution with a large covariance $\boldsymbol{Q}$; then, upon rejection of such candidate point, instead of keeping $X_{i+1} = X_i$ one should try with a new draw $X_i^{*'}$, obtained from a proposal with a smaller standard deviation $\boldsymbol{Q'} < \boldsymbol{Q}$. Green and Mira motivated this process as follows: "The advantage of DR over these alternatives is that a hierarchy between kernels can be exploited so that kernels that are easier to compute

in terms of CPU time are tried first, thus saving terms on simulation time. Moves that are more “bold” are tried at earlier stages thus allowing the sampler to explore the state space more efficiently following a sor of “first bold” versus “second timid” tennis-service stategy” [39]. Also, the same authors proved in the same paper that this rejection process can be repeated for a prefixed number of stages without nullifying the Markovian property of the chain.
Fig. 4.9 gives a better picture of the DR procedure for a mono-dimensional Metropolis Algorithm. Assume that we fixed the maximum number of stages for which the DR can be repeated in case of rejections (here $n_{DR} = 3$), and the constant $\omega_{DR}$ that will divide the proposal standard deviation at each of the stages. In Fig. 4.9(a) a first candidate point $X^*$ is obtained by a proposal with standard deviation $\sigma_q$ and this value will give in turn an acceptance probability of 10% ($\alpha = 0.1$). Then, supposing that $X^*$ has been rejected, we try again we a new proposal created dividing the old standard deviation by the constant $\omega_{DR}$ ($\sigma_q' = \sigma_q/\omega_{DR}$) and we obtain a candidate $X^{*'}$ with probability of being accepted equal to 30% (Fig. 4.9(b)). Presuming that also this point has been rejected, we will produce a last Proposal; this time ($\sigma_q'' = \sigma_q'/\omega_{DR} = \sigma_q/\omega_{DR}^2$). A point $X^{*''}$ is computed from such distribution (Fig. 4.9(c)) and this draw has a probability of being accepted equal to 59%. At this point, considering that we spent all our attempts ($i_{DR} = n_{DR} = 3$), there are two options: likely, $X^{*''}$ will be confirmed and therefore $X_{t+1} = X^{*''}$; in the unlucky case that also this third point is rejected, we would have to give up and keep the current value $X_t$ as the new one.

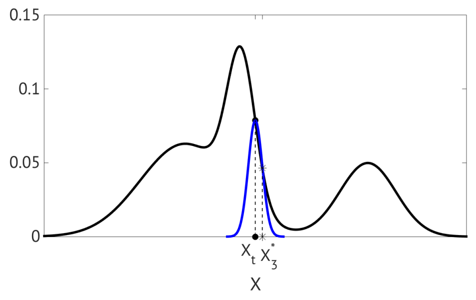
Another improovment of the Metropolis-Hasting Algorithm is achievable through the Adaptive Method (AM) proposed by Haario *et al.* [40]. The key idea of such process is to take advantage of the previous accepted points for correcting the proposal distribution. In other words, the AM modifies the proposal's covariance matrix, so that its principal axis could be aligned to the main directions of the random walk.
This method starts with assuming a generic proposal during an entire non-adaptation period $n_{AM_0}$; this is required for letting the chain take its time with finding the preferential directions of evolution. At the first step grater than $n_{AM_0}$, the proposal is set to be centered at the current position of the Markov chain and to have covariance:

$$\boldsymbol{Q}_n = s_d \left(\mathrm{Cov}(\boldsymbol{X}_0, ..., \boldsymbol{X}_{n-1}) + \epsilon \boldsymbol{I}_d\right) \tag{4.17}$$

where $s_d$ is a scaling parameter (that Gelman suggested to set as $s_d = 2.4^2/d$ with d representing the number of dimensions [37]), $\epsilon$ is a parameter that ensures that $\boldsymbol{Q}_n$ will not become singular and $\mathrm{Cov}(\boldsymbol{X}_0, ..., \boldsymbol{X}_{n-1})$ is the covariance matrix computed using



(a) Target (black line) and Proposal (blue line) PDFs. (b) Metropolis-Hasting chain relative to (a).



(c) Target (black line) and Proposal (blue line) PDFs.

FIGURE 4.9: *Delayed Rejection Method:*
*A first candidate point $X^*$ has been obtained in (a) sampling a normal proposal distribution with $\sigma_q$ and it has been rejected. A new proposal has then been computed with $\sigma'_q = \sigma_q/2$ and it generated a candidate $X^{*'}$ (b). This point has been rejected as well and a new point $X^{*''}$ has been produced from a new distribution with $\sigma''_q = \sigma_q/4$ (c).*

the previous points:

$$\mathrm{Cov}(\boldsymbol{X}_0, ..., \boldsymbol{X}_{n-1}) = \frac{1}{k}\left(\sum_{i=0}^{k} \boldsymbol{X}_i \boldsymbol{X}_i^T - (K+1)\bar{\boldsymbol{X}}_k \bar{\boldsymbol{X}}_k^T\right) \tag{4.18}$$

where $\bar{\boldsymbol{X}}_k = \frac{1}{k+1}\sum_{i=0}^{k} \boldsymbol{X}_i$.
At the following steps of the chains, the matrix $\boldsymbol{Q}$ will then be updated through the points collected in the meantime; the new covariance will be:

$$\boldsymbol{Q} = s_d \left(\mathrm{Cov}(\boldsymbol{X}_0, ..., \boldsymbol{X}_{t-1}) + \epsilon \boldsymbol{I}_d\right), \qquad t > n_{AM_0}. \tag{4.19}$$

Moreover, it can be shown that $C_t$ satisfies the more handy recursive formula:

$$\boldsymbol{Q}_t = \frac{t-2}{t-1}\boldsymbol{Q}_{t-1} + \frac{s_d}{t-1}\left((t-1)\bar{\boldsymbol{X}}_{t-2}\bar{\boldsymbol{X}}_{t-2}^T - t\bar{\boldsymbol{X}}_{t-1}\bar{\boldsymbol{X}}_{t-1}^T + \boldsymbol{X}_{t-1}\boldsymbol{X}_{t-1}^T + \epsilon \boldsymbol{I}_d\right). \tag{4.20}$$

This mechanism of modifying the proposal is shown in Fig. 4.10; if the target distribution is defined above a multidimensional space, this technique allows to start the chain with

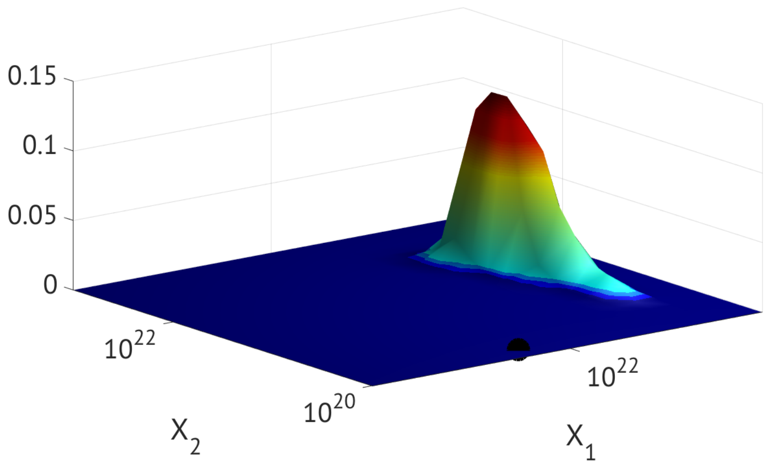
an independent proposal for each of the variables (that is, with a diagonal covariance matrix for representing $\boldsymbol{Q}$ for all the $\boldsymbol{X}_i < n_{AM_0}$). Indeed, under the assumption that $n_{AM_0}$ is large enough, the Adaptive Method itself will take into account the correlation that we are missing by means of learning from the random walk path.
Haario *et al.* proved in the same paper that such form of adaptation is an ergodic process, although it does not generate a Markov chain due to the dependence of the proposed values upon the history of the random walk. In summary, DR and AM are two approaches for adapting the proposal distribution to the target PDF $P(\boldsymbol{X})$. However, while the former aims to reach a local adaptation of $\boldsymbol{Q}$ based on the shape of $P(\boldsymbol{X})$ at the current point $\boldsymbol{X}_i$, the latter has the goal of globally adapting the proposal based on the chain past history.
Considering that in general the target PDF is far from being a single-mode normal distribution, the local shape of the distribution is different from its global aspect. The intuition of Haario *et al.*, proposed in their more recent paper [39], was to create a compromise between these two approaches by combining them into a single method, the DRAM. It is possible to couple the two Algorithms in practice through the following two steps:

- At the first stage of DR a covariance matrix $\boldsymbol{Q}_t$ is generated from the previous draws of the chain.
- At the later stages of DR (in the event in which $i_{DR} \leq n_{DR}$ and such that $\boldsymbol{Q}_t^{i_{DR}-1}$ proposed some value that has not been accepted) $\boldsymbol{Q}_t^{i_{DR}}$ is computed as a scaled version of the proposal at the first stage: $\boldsymbol{Q}_t^{i_{DR}} = \boldsymbol{Q}_t/(\omega_{DR})^{i_{DR}-1}$.

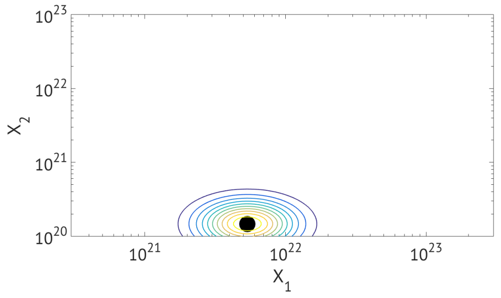
The advantages of this hybrid method compared to the single DR and AM and to the original MH is sumarized in the Figure captured from Haario's paper (Fig. 4.11), that shows the convergence properties of these sampling strategies for a particular case of study proposed in the same article.

## 4.5 Validation

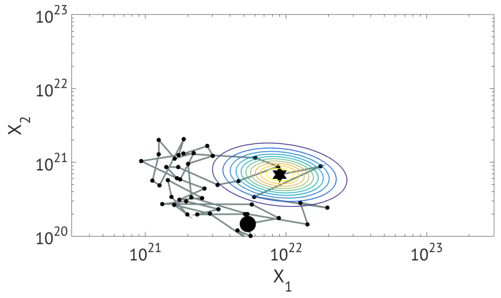
We have shown that a Delayed Rejection Adaptive Method applied to the Markov Chain Monte Carlo sampling can provide a solution to the Calibration inverse problem in terms of a probability distribution defined above the parameter and hyperparameter space $S = \{\boldsymbol{\theta}; \boldsymbol{\alpha}\}$. Through the marginalization of such PDF over each of the parameters, is then possible to find the probability that a particular value of $\theta_i$ or $\alpha_i$ produces an observable output $\boldsymbol{y}$ equal to the data $\boldsymbol{D}$. Moreover, thanks to the stochastic representation that



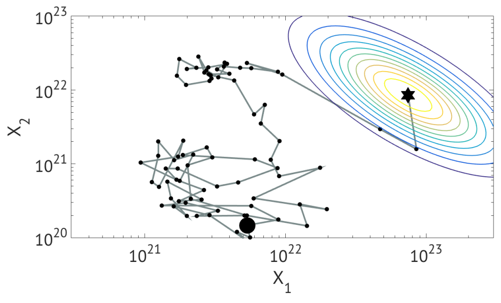
(a)



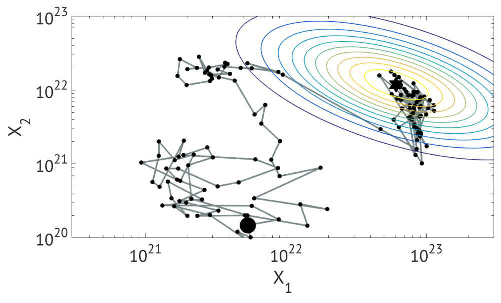
(b)



(c)



(d)



(e)

FIGURE 4.10: *Adaptive Method: A MCMC is started from the black point for sampling the target distribution in (a). The resulting chain and the current proposal distribution (level lines) are shown at four computational times: at the first point (b), at an instant right after the end of the non-adaptation period (c) (note that the covariance matrix is almost diagonal), and at two successive instants (d and e).*

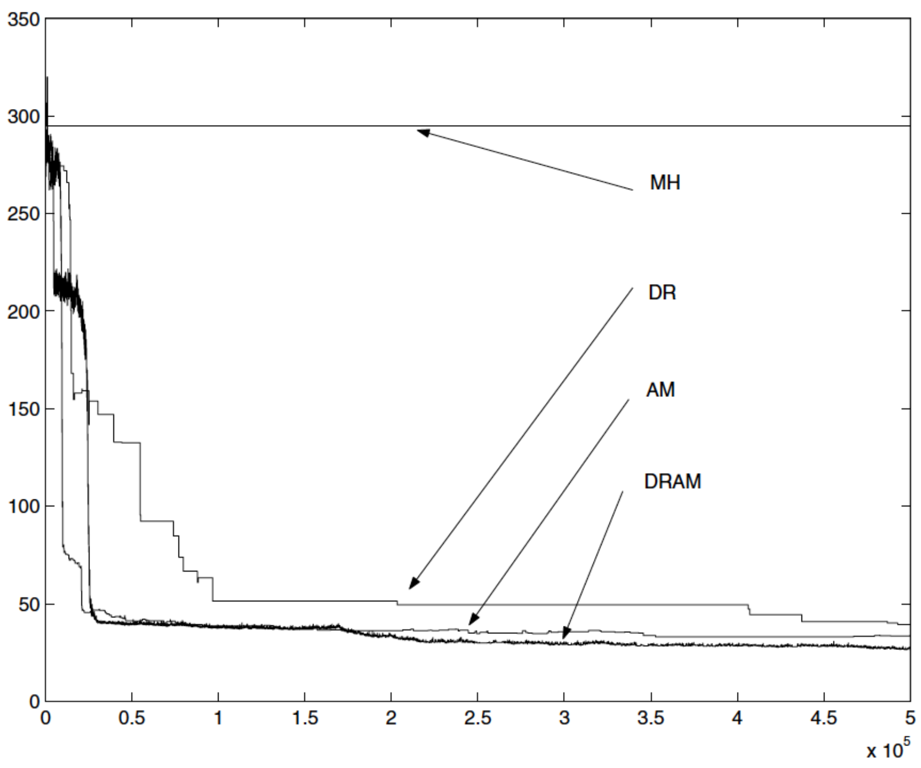


Figure 4.11: *Convergence of chain by MH, DR, AM and DRAM, from a case of study in Haario et al.'s paper [39]. The sum of the residual squares between data and output is plotted against simulation time*

has been used for characterizing the structural inadequacy and thanks to the fact that such stochastic modeling has been moved upstream, we are able in this way to propagate the parameter uncertainty to the quantities of interest $\boldsymbol{q}$.

Unfortunately, the calibration process is only able to guarantee that the parameters are learning as much as possible from the data; however, this could be not enough: we have to explicitly check that the calibrated model is able to replicate the experiment with acceptable discrepancies. In case any of the available data is not a probable outcome, we can assess that such model is not capable of making reliable predictions with that particular scenario condition.

At this point, it is clearly imperative to define a proper metric for establishing when the discrepancy between output and data can be considered acceptable. For this purpose, Oliver *et al.* [24] suggested to use the Highest Posterior Density (HPD) credibility regions.

A $\beta - HPD$ credible region is defined as "the set for which the probability of belonging to S is $\beta$ and the probability density for each point in S is grater than that of the points

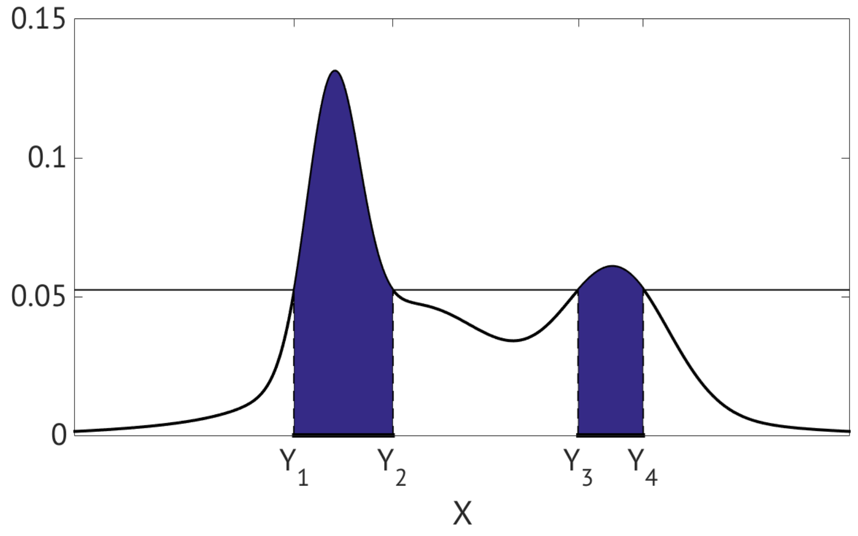


FIGURE 4.12: $50\% - HPD$ *regions (area in blue) for the PDF in black.*

outside S". That is:

$$\beta = \int_S p(\boldsymbol{y} \mid \boldsymbol{D}, \boldsymbol{M}) d\boldsymbol{y}, \qquad where\ S = \Big\{ \boldsymbol{y} : p(\boldsymbol{y}|\boldsymbol{D}.\boldsymbol{M}) \geq p(\boldsymbol{Y}|\boldsymbol{D}.\boldsymbol{M}) \Big\}. \tag{4.21}$$

where $\boldsymbol{Y}$ is a particular value from the outcomes.

For example, if we want to compute the High Density Posterior credibility regions with a $\beta = 50\%$ for the probability density shown in Fig. 4.12, we draw an horizontal line extended for the entire domain and tangent to such PDF at its maximum. Then, we start to pull down the horizontal line until the area under the function and between the points in which it intersects the PDF is equal to $\beta$. The intervals $(Y_1, Y_2)$ and $(Y_3, Y_4)$ will be our $50\% - HPD$ region.

Assume that we solved our calibration problem and we computed the posterior distribution for all the important parameters and hyperparameters. Moreover, say that we already propagated such uncertainty to the observable quantities by solving $n$ direct problems using the $n$ sets of parameters of the Markov chain that we computed during the calibration process. Thus, we have a posterior distribution for the observable quantities, $\Pi(\boldsymbol{y})$.

We now want to validate our model; in other words, we want to show that the $\boldsymbol{y}$ that we obtained are good predictions of the data $\boldsymbol{D}$. We can thus use the Highest Posterior Density credibility region in their reverse sense: instead of fixing $\beta$ and computing the HPD intervals, we can fix the intervals to contain the data and to be at the same time as smallest as possible and we can than compute the interval probability $\beta$.

If we define a parameter $\chi$ to be:

$$\chi = 1 - \beta, \tag{4.22}$$

when $\chi = 1$ the most probable model output coincides with the data. On the other hand, $\chi = 0$ means that the model is unable to produce the data as outcome. Thus we can set a reasonable tolerance $\bar{\chi}$, so that if $\chi < \bar{\chi}$ the data cannot be considered a probable outcome and the model is invalidated.

The fact that the model has not failed the validation process does not guarantee by itself that such model is capable of making reliable prediction. For example, as already mentioned in the previous Sections, one has to be sure that the sthocastical model that has been created for representing the structural inadequacy is upstream enough and that it is capable of affecting the QoIs' computation. Such dependency can easily be proven assessing the sensitivity of the quantity of interest with respect to the hyperparameters.

## 4.6 SMUQ: Stochastic Modeling for Uncertainty Quantification

Reached the stage of delineating a proper way for quantifying the uncertainty and validating the model, the author of this thesis implemented all the methodology described in this chapter in a single FORTRAN library, that at a later time has been named Stochastic Modeling Uncertainty Quantification (SMUQ).
Initially, writing *ex novo* a UQ code was merely for educational purposes. Indeed, a significant number of UQ softwares has been released in the last years by authoritative Departments, Institutions and Agencies (e.g., DAKOTA by Sandia National Laboratories [41] and UQTools by NASA [42]). Moreover, some of them (e.g., QUESO by the PECOS Center, University of Texas at Austin [25]) are based on methodologies similar to the ones that has been presented in this chapter. On the other hand, a mechanical use of an already packed tool would have likely meant losing sight at the physical essence of the problem to solve. This could have been particularly dangerous for the approach to the validation problem that we are going to use, due to the fact that the stochastic representation of the model inadequacy has to be based on strong physical reasons.
Once that the first results started to come out, however, SMUQ showed to perform well, also compared to the other softwares: it has been able to reproduce the results obtained through different UQ codes and it also turned out to be very user-friendly in solving a wide range of UQ problems. This is due to the fact that SMUQ treats the external model as a black box, as it will be explained in the next lines.

Although a complete description of the implemented code's operating principles is not one of the aims of this thesis, it is useful to point out some of its key-ideas and features,

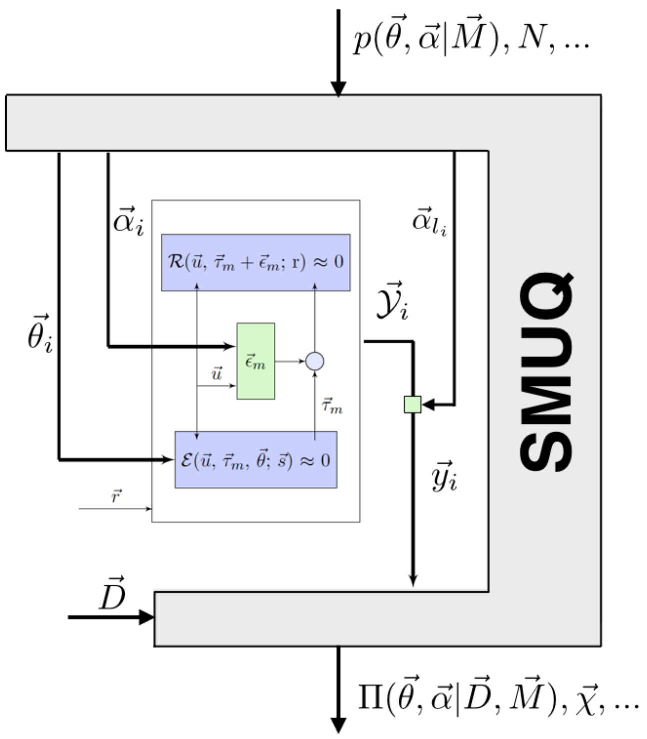

$p(\vec{\theta}, \vec{\alpha}|\vec{M}), N, ...$
$\vec{\alpha}_i$
$\vec{\alpha}_{l_i}$
$\mathcal{R}(\vec{u}, \vec{\tau}_m + \vec{\epsilon}_m; r) \approx 0$
$\vec{\mathcal{Y}}_i$
$\vec{\theta}_i$
$\vec{\epsilon}_m$
$\vec{u}$
$\vec{\tau}_m$
$\mathcal{E}(\vec{u}, \vec{\tau}_m, \vec{\theta}; \vec{s}) \approx 0$
$\vec{y}_i$
$\vec{r}$
SMUQ
$\vec{D}$
$\Pi(\vec{\theta}, \vec{\alpha}|\vec{D}, \vec{M}), \vec{\chi}, ...$

FIGURE 4.13: *Schematic representation of SMUQ's working strategy in solving the Calibration and Validation processes: once the input parameters for the Bayesian Inference, for the Markov Chain and for the Delayed Rejection Adaptive Method are specified, SMUQ is able to quantify the model uncertainty and to establish the reliability of the model predctions taking advantage of the available data* $\boldsymbol{D}$.

in order to give a better understanding of the results proposed in the next chapters.
The main goal of such code is to compute the necessary quantities for quantifying the uncertainty and validating the model. Whith this ambition, one should have in mind what suggested by Tarantola in his book: "This is essentially what must be proposed: to look at a large number of randomly generated models (first, of the prior distribution, and then, of the posterior distribution) in order to intuitevely apprend the basic properties of the probability distribution, followed by calculation of the probabilities of all interesting events. Sometimes, it may nevertheless be necessary to estimate some moments (mean, covariance, etc.) of distribution. Of course, they can also be evaluated using the samples." [31]. For this reason, SMUQ provides in outputs quantities like the posterior distribution of the parameters and hyperparameters $\Pi(\boldsymbol{\theta}, \boldsymbol{\alpha}|\boldsymbol{D}, \boldsymbol{M})$, the posterior distribution of the observable quantities $\Pi(\boldsymbol{y}|\boldsymbol{D}, \boldsymbol{M})$ and the probabilties connected to the High Posterior Density regions $\boldsymbol{\chi}$.
As it has already been explained, the code is based on Bayesian Inference for solving the inverse problem and it generates a Markov chain by means of a Delayed Rejection

Adaptive Method applied to the Metropolis-Hasting Algorithm. Therefore, SMUQ requires that all the functions and all the parameters associated with such composition of methodologies have been specified. These functions and parameters will thus represent the inputs of the code.

For this to be clarified, we will briefly describe SMUQ's working strategy, that is represented schematically in Fig. 4.13. Assuming that a data $\boldsymbol{D}$ is available and the scenario inputs $\boldsymbol{r}$ for the external model have been specified, SMUQ computes recursively the observable quantities $\boldsymbol{y}_i$ through the external model; at each of the iterations, however, the values of the parameters $\boldsymbol{\theta}_i$ and of the hyperparameters $\boldsymbol{\alpha}_i$ have been properly changed by the DRAM process. This repeated mechanism produces a Markov chain $\{(\boldsymbol{\theta},\boldsymbol{\alpha})_1,(\boldsymbol{\theta},\boldsymbol{\alpha})_2,...,(\boldsymbol{\theta},\boldsymbol{\alpha})_{N_{MC}}\}$, where the length of chain $n_{MC}$ must have been specified in the SMUQ's input, and such succession is a set of samples of the parameters posteriors.

SMUQ is capable of running in parallel on different processors and this is made possible by creating a different recursive process on each of the computer cores. Each of this processes generates a different chain and the final total set of samplings, then, will be a collage of such successions. The advantages of this feature are crucial: the parallelization reduces considerably the computational time, permits to reduce the autocorrelation of the total chain and allows to explore deeply and faster the entire parameters space. This last point is due to the fact that all the processors receive the same initial inputs, except for the chain initial points (Subsec. 4.4.3); each of the processors, indeed, produces a first draw by means of an independent sample of the prior distribution. This requires to provide a proper representation of such PDF, based on the knowledge available about the parameters.

For example, if the only information that we have about a particular $\theta_i$ (or about a particular $\alpha_j$) is the range in which it is restricted to vary, then the box function can be a good description of the prior:

$$p(\theta_i \mid \boldsymbol{M}) = \begin{cases} 1, & \text{if } \theta_i \in \Big(\theta_{\mathrm{min}_i}, \theta_{\mathrm{max}_i}\Big), \quad \forall\, i \leq T \\ 0, & \text{otherwise} \end{cases} \tag{4.23}$$

or

$$p(\alpha_j \mid \boldsymbol{M}) = \begin{cases} 1, & \text{if } \alpha_j \in \Big(\alpha_{\mathrm{min}_j}, \alpha_{\mathrm{max}_j}\Big), \quad \forall\, j \leq A \\ 0, & \text{otherwise.} \end{cases} \tag{4.24}$$

In other cases, a normal distribution can be more appropriate:

$$p(\theta_i \mid \boldsymbol{M}) \propto \exp\Big(-\frac{1}{2}\big(\frac{\theta_i - \mu_{\theta_i}}{\sigma_{\theta_i}}\big)^2\Big) \tag{4.25}$$

or

$$p(\alpha_j \mid \boldsymbol{M}) \propto \exp\Big(-\frac{1}{2}\big(\frac{\alpha_j - \mu_{\alpha_j}}{\sigma_{\alpha_j}}\big)^2\Big) \tag{4.26}$$

Sometimes, nonetheless, a combination of the former and the latter PDFs is the best choice and the prior must be chosen to be a Gaussian function in which the tails are cut off.

All these probability distributions have been implemented in SMUQ. Once one specifies for each of the parameters and hyperparameters the shape of the distribution, the ranges of variation in case of box function and the mean and the standard the distribution in case of a Gaussian, SMUQ samples the priors and produces the initial point for each of the processors. Moreover, under the assumption of uncorrelation between the PDFs in Eq.s 4.23 - 4.26, the code computes the values of the joint priors at such starting points, as in Eq. 4.11.

The next step is obtaining the posterior at ($\boldsymbol{\theta}_1$ and $\boldsymbol{\alpha}_1$). In order to do that, we have to define the proper representations for the likelihood. Following what suggested by different autors [33] [35], a Gaussian function is an appropriate representation of the likelihood for a wide range of problems. Assume now that the data is available in $N_D$ different points and that, in the same nodes, we computed the model outputs; we can write:

$$l(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{M}) = \frac{1}{\sqrt{2\pi\sigma_l^2}} \exp\Big(-\frac{1}{2}\sum_{k=1}^{N_D}\big(\frac{y_k - D_k}{\sigma_l}\big)^2\Big) \tag{4.27}$$

At this point, it is possible to compute the likelihood $L(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{M})$ as shown in Eq. 4.12; the integration can be realized by means of a Montecarlo integration [43]:

$$\begin{aligned} L(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{M}) &= \int_{-\infty}^{+\infty} l(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{M}) \times \prod_{k=1}^{p} f(\gamma_k \mid \alpha_{1_k}, ..., \alpha_{n_k}, \boldsymbol{M})\ d\boldsymbol{\gamma} = \\ &= \int_{-\infty}^{+\infty} l(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{M})\ g(\boldsymbol{\gamma})\ d\boldsymbol{\gamma} = \\ &= \mathbf{E}_{g(\boldsymbol{\gamma})} \Big|\ l(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{M})\ \Big| = \\ &\approx \sum_{z=1}^{G} \frac{l(\boldsymbol{y} = \boldsymbol{D} \mid \boldsymbol{\theta}, \bar{\boldsymbol{\gamma}}_z \boldsymbol{M})}{N_\gamma} \end{aligned}$$

where $\bar{\boldsymbol{\gamma}}_z$ represent a value of the vector $\boldsymbol{\gamma}$ obtained by sampling the probability distribution $g(\boldsymbol{\gamma}) = \prod_{k=1}^{p} f(\gamma_k \mid \alpha_{1_k}, ..., \alpha_{n_k}, \boldsymbol{M})$ with one of the G different draws. From such equation it is clear that SMUQ needs the standard deviation $\sigma_l$, the probability $g(\boldsymbol{\gamma})$ and the number of draws G to be specified in order to compute the value of likelihood at some generic $\boldsymbol{\theta}_i$ and $\boldsymbol{\alpha}_i$, given a data $\boldsymbol{D}$.

Once the values of the prior and of the likelihood have been obtained, with a simple

multiplication it is possible to find the posterior distribution $\Pi(\boldsymbol{\theta_1}, \boldsymbol{\alpha_1} \mid \boldsymbol{D}, \boldsymbol{M})$.
At this point, a candidate set of values for the parameters and for the hyperparameters is computed by means of a proposal PDF. Such distribution can be represented at this step by a multi-dimensional Gaussian function with a mean $\boldsymbol{\mu}_q$ and a diagonal covariance matrix $\boldsymbol{Q} = \text{diag}(\sigma_{q_1}, \sigma_{q_2}, ..., \sigma_{q_T}, \sigma_{q_{T+1}}, ..., \sigma_{q_{T+A}})$. With the same procedure explained above, it is possible to determine the value of the posterior at $\boldsymbol{\theta}^*, \boldsymbol{\alpha}^*$.
Following the steps of the Metropolis-Hasting Algorithm, the ratio

$$\alpha_1 = \frac{\Pi(\boldsymbol{\theta}^*, \boldsymbol{\alpha}^* \mid \boldsymbol{D}, \boldsymbol{M})}{\Pi(\boldsymbol{\theta_1}, \boldsymbol{\alpha_1} \mid \boldsymbol{D}, \boldsymbol{M})} \tag{4.28}$$

will establish if the set $\boldsymbol{\theta}^*, \boldsymbol{\alpha}^*$ will be a new point of the Markov chain. In the case such set is rejected there are two possibilities: if SMUQ is set to use a DR method (i.e: if $n_{DR} > 1$), new $\boldsymbol{\theta}^*, \boldsymbol{\alpha}^*$ will be generated by a $\boldsymbol{Q}^{'} = \boldsymbol{Q}/\omega_{DR}$; otherwise, the chain will not move from the first point and $\boldsymbol{\theta}_2 = \boldsymbol{\theta}_1, \boldsymbol{\alpha}_2 = \boldsymbol{\alpha}_1$.
If one defined $n_{MC}$ to be the length of the Markov chain, SMUQ will repeat the process described above $n_{MC}$ times in order to sample the posterior distribution of the parameters. In other words, the software will run the external code in a loop of $(n_{MC} \times G)$ repetitions, changing every time the model parameters and hyperparameters and eventually modifying the proposal covariance matrix $\boldsymbol{Q}$ by means of the Adaptive Method. Once the posterior distribution of the parameters has been defined, the model uncertainty is quantified. Moreover, while SMUQ repeats such process, it also stores the outputs $\boldsymbol{y}_i$. Such vectors represent a sample of the observable quantities' posterior distribution and they are used to compute the probabilities $\boldsymbol{\chi}$ connected to the High Posterior Density regions by means of Eq.s 4.21 and 4.22.

Chap. 6 will present the results of the application of SMUQ to a particular uncertainty quantification problem and that will be useful for better clarifying the code working strategy and its potential.