Methodology

The Uranie team, CEA DES, support-uranie@cea.fr

Feb 13, 2026

CONTENTS:

LIST OF FIGURES

LIST OF TABLES

Abstract This documentation is introducing the theoretical basics upon which the Uranie platform (based on Uranie *v4.11.0*), has been developed at CEA/DES. Since this platform is designed for uncertainty propagation, sensitivity analysis and surrogate model generation, the main methods which have been implemented are introduced and discussed. This document is however not made to give a complete overview of the methodology but more to open the scope of the reader by largely relying on a list of references, without getting to attached on the structure of the Uranie platform itself.

Commit
61cebb3

CHAPTER

ONE

GLOSSARY

Analysis of variance or **ANOVA** (*Analyse de variance*): decomposition of the variance (as a breakdown) to elementary pieces (also know as HDMR, Hoeffding's decomposition, Sobol's decomposition… *c.f. No hypothesis on the model*).

Cumulative distribution function or **CDF** (*Fonction de répartition*): function of a real-valued random variable X which, once evaluated at x, gives the probability that X will take a value less than or equal to x (*c.f. The probability distributions*).

Kriging or **Gaussian process** (*Krigeage ou processus gaussien*): is a family of interpolation methods that uses information about the "spatial" correlation between observations to make predictions with a confidence interval at new locations (*c.f. The kriging method*).

Latin hypercube sampling or **LHS** (*Échantillonage par hypercube latin*): sampling methods that stratifies the probability space by dividing it in equal probabilities (*c.f. Introduction*).

Leave-one-out or **LOO** (*validation croisée un contre tous*): type of cross-validation for which a surrogate model is re-train on the learning database removing just one point, in order to obtain an estimation of this new model on this precise point (*c.f. Adapting the fitting strategy*).

Likelihood (*vraisemblance*): is the hypothetical probability that an event that has already occurred would yield a specific outcome. The concept differs from that of a probability in that a probability refers to the occurrence of future events, while a likelihood refers to past events with known outcomes [Wei].

Low discrepancy sequence: (*Suite à faible discrépance*): sequence for which the discrepancy is low, meaning the proportion of points in the sequence falling into an arbitrary set B is close to proportional to the measure of B (*c.f. QMC method*).

Pareto front (*front de Pareto*): a set of nondominated solutions, being chosen as optimal, if no objective can be improved without sacrificing at least one other objective (*c.f. The pareto concept in a nutshell*).

Pearson coefficient (*Coefficient de Pearson*): it is the linear correlation coefficient (*c.f. The monotone case*).

Principal component analysis or **PCA** (*Analyse en conclusion principale*): the process of computing the principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest (*c.f. Combining these aspects: performing PCA*).

Probability density function or **PDF** (*Densité de probabilité*): function whose value at any given sample (or point) in the sample space (the set of possible values taken by the random variable) can be interpreted as providing a relative likelihood that the value of the random variable would equal that sample (*c.f. The probability distributions*).

Quantile (*Quantile*): the quantile x_p, for a probability $p \in [0, 1]$, is the lowest value of a random variable X so that $P\{X \leq x_p\} = p$ (*c.f. The quantile computation*).

Screening method (*méthode de criblage*): process that extracts, isolates and identifies a compound or group of components in a sample with the minimum number of steps and the least manipulation of the sample (*c.f. Sensitivity analysis*).

Simple random sampling or **SRS** (*Échantillonage simple aléatoire*): independent generation of samples following provided PDFs (*c.f. Introduction*).

Sparse grids: numerical techniques to represent, integrate or interpolate high dimensional functions.

CHAPTER

TWO

BASIC STATISTICAL ELEMENTS

Abstract This chapter is introducing the very basic statistical operations that can be done using the Uranie platform, on a set of points provided by the user, or generated by Uranie itself.

This chapter introduces the various probability laws implemented in Uranie and illustrates, for each every one of them, with a few sets of parameters, the resulting shape of three of their characteristic functions. Some of the basic statistical operations are also described in a second part.

2.1 Random variable modelisation

2.1.1 The probability distributions

There are several already-implemented statistical laws in Uranie, that can be called marginal laws as well, used to described the behaviour of a chosen input variable. They are usually characterised by two functions which are intrinsically connected: the PDF (probability density function) and CDF (cumulative distribution function). One can recap briefly the definition of these two functions for every random variable $X : \Omega \to \mathbb{R}$:

- PDF: if the random variable X has a density f_X, where f_X is a non-negative Lebesgue-integrable function, then

$$P\{a \leq X \leq b\} = \int_a^b f_X(s)ds$$

- CDF: the function $F_X : \mathbb{R} \to [0, 1]$, given by

$$F_X(x) = \int_{-\infty}^{x} f_X(s)ds,\ x \in \mathbb{R}$$

For some of the distributions discussed later on, the parameters provided to define them are not limiting the range of their PDF and CDF: these distributions are said to be infinite-based ones.
It is however possible to set boundaries in order to truncate the span of their possible values. One can indeed define an lower bound L and or an upper bound U so that the resulting distribution range is not infinite anymore but only in $[L, U]$. This truncation step affects both the PDF and CDF: once the boundaries are set, the CDF of these two values are computed to obtain P_L (the probability to be lower than the lower edge) and P_U (the probability to be lower than the upper edge). Two new functions, the truncated PDF $f_X^{[L,U]}$ and the truncated CDF $F_X^{[L,U]}$ are simply defined as

$$f_X^{[L,U]}(x) = \frac{f_X(x)}{P_U - P_L},\ \ F_X^{[L,U]}(x) = \frac{F_X(x) - P_L}{P_U - P_L}.$$

These steps to produce a truncate distribution are represented in Figure 2.1 where the original distribution is shown on the left along with the definition of L (the blue shaded part) and U (the green shaded part). The right part of the plot is the resulting truncated PDF.

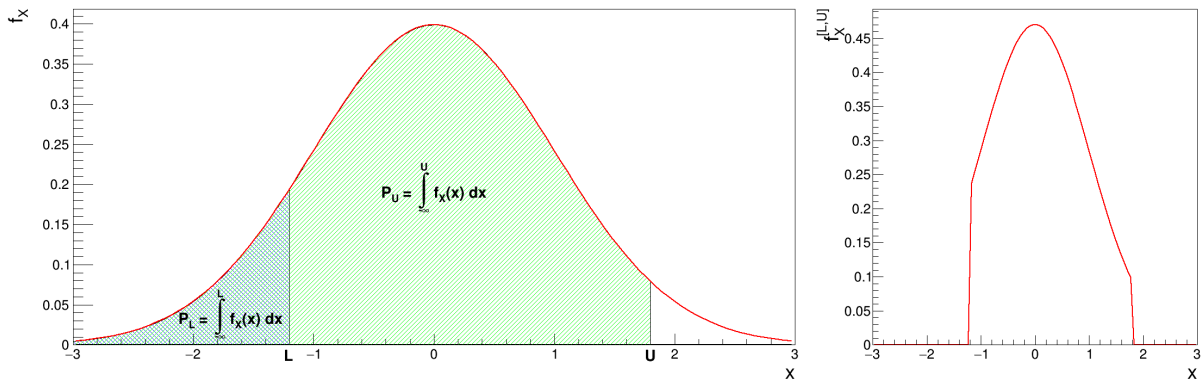

Figure 2.1: Principle of the truncated PDF generation (right-hand side) from the orginal one (left-hand side).

It is possible to combine different probability law, as a sum of weighted contributions, in order to create a new law. This approach, which is further discussed and illustrated in *Composing law*, leads to a new probability density function that would look like

$$f(x) = \sum_{j=1}^{N} \omega_j f_j(x) \quad \text{where} \quad \forall j \in [1, N], \quad \omega_j \in \mathbb{R}^+.$$

These distributions can be used to model the behaviour of variables, depending on chosen hypothesis, probability density function being used as a reference more oftenly by physicist, whereas statistical experts will generally use the cumulative distribution function [App13].

Table 2.1 gathers the list of implemented statistical laws, along with the list of parameters used to define them. For every possible law, a figure is displaying the PDF, CDF and inverse CDF for different sets of parameters (the equation of the corresponding PDF is reminded as well on every figure). The inverse CDF is basically the CDF whose x and y-axis are inverted (it is convenient to keep in mind what it looks like, as it will be used to produce design-of-experiments, later-on).

Table 2.1: List of Uranie classes representing the probability laws

Law	Class Uranie	Parameter 1	Parameter 2	Parameter 3	Parameter 4
Uniform	*TUniformDistribution*	Min	Max		
Log-Uniform	*TLogUniformDistribution*	Min	Max		
Triangular	*TTriangularDistribution*	Min	Max	Mode	
Log-Triangular	*TLogTriangularDistribution*	Min	Max	Mode	
Normal (Gauss)	*TNormalDistribution*	Mean (μ)	Sigma (σ)		
Log-Normal	*TLogNormalDistribution*	Mean (M)	Error factor (E_f)	Min	
Trapezium	*TTrapeziumDistribution*	Min	Max	Low	Up
UniformByParts	*TUniformByPartsDistribution*	Min	Max	Median	
Exponential	*TExponentialDistribution*	Rate (λ)	Min		
Cauchy	*TCauchyDistribution*	Scale (γ)	Median		
GumbelMax	*TGumbelMaxDistribution*	Mode (μ)	Scale (β)		
Weibull	*TWeibullDistribution*	Scale (λ)	Shape (k)	Min	
Beta	*TBetaDistribution*	alpha (α)	beta (β)	Min	Max
GenPareto	*TGenParetoDistribution*	Location (μ)	Scale (σ)	Shape (ξ)	
Gamma	*TGammaDistribution*	Shape (α)	Scale (β)	Location (ξ)	
InvGamma	*TInvGammaDistribution*	Shape (α)	Scale (β)	Location (ξ)	
Student	*TStudentDistribution*	DoF (k)			
GeneralizedNormal	*TGeneralizedNormalDistribution*	Location (μ)	Scale (α)	Shape (β)	

2.1.1.1 Uniform Law

The Uniform law is defined between a minimum and a maximum, as

$$f(x) = \frac{1}{(x_{\text{max}} - x_{\text{min}})} \mathbb{1}_{[x_{\text{min}}, x_{\text{max}}]}(x)$$

The property of the law lies on the fact that all points of the interval $[x_{\text{min}}, x_{\text{max}}]$ have the same probability. The mean value of the uniform law can then be computed as $\mu = \frac{x_{\text{max}} + x_{\text{min}}}{2}$ while its variance can be written as $\sigma^2 = \frac{(x_{\text{max}} - x_{\text{min}})^2}{12}$. The mode is not really defined as all points have the same probability.

Figure 2.2 shows the PDF, CDF and inverse CDF generated for a given set of parameters.

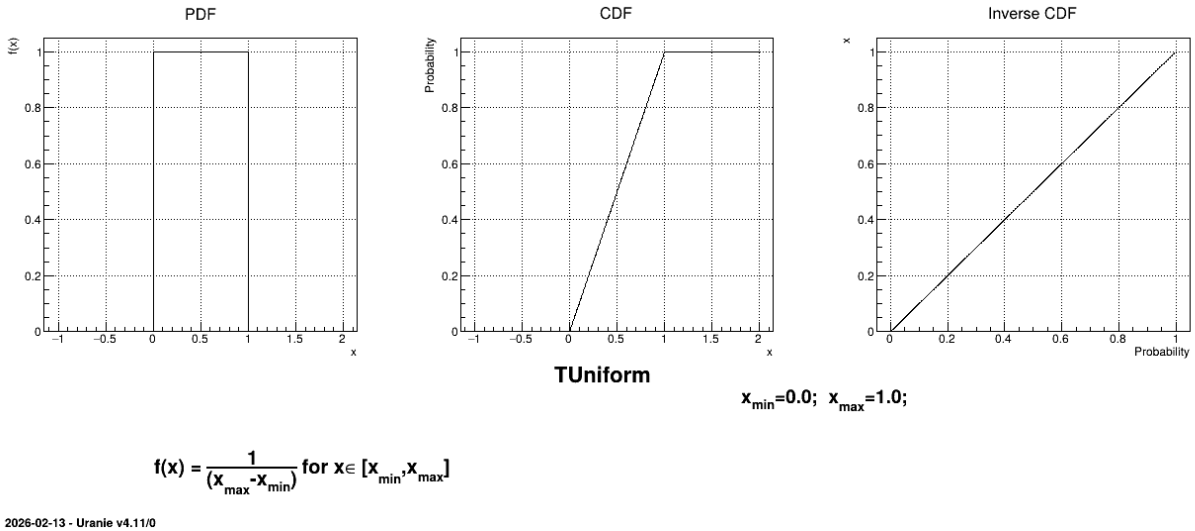

Figure 2.2: Example of PDF, CDF and inverse CDF for Uniform distributions.

2.1.1.2 Log Uniform Law

The LogUniform law is well adapted for variations of high amplitudes. If a random variable x follows a LogUniform distribution, the random variable $\ln(x)$ follows a Uniform distribution, so

$$f(x) = \frac{1}{(x \times \ln(x_{\max}/x_{\min}))} \mathbb{1}_{[x_{\min},x_{\max}]}(x)$$

From the statistical point of view, the mean value of the LogUniform law can then be computed as $\mu = \frac{x_{\max} - x_{\min}}{\ln(x_{\max}/x_{\min})}$ while its variance can be written as $\sigma^2 = \frac{x_{\max}^2 - x_{\min}^2}{2\ln(x_{\max}/x_{\min})} - \left(\frac{x_{\max}-x_{\min}}{\ln(x_{\max}/x_{\min})}\right)^2$. By definition, the mode is equal to $x_{\min}$.

Figure 2.3 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

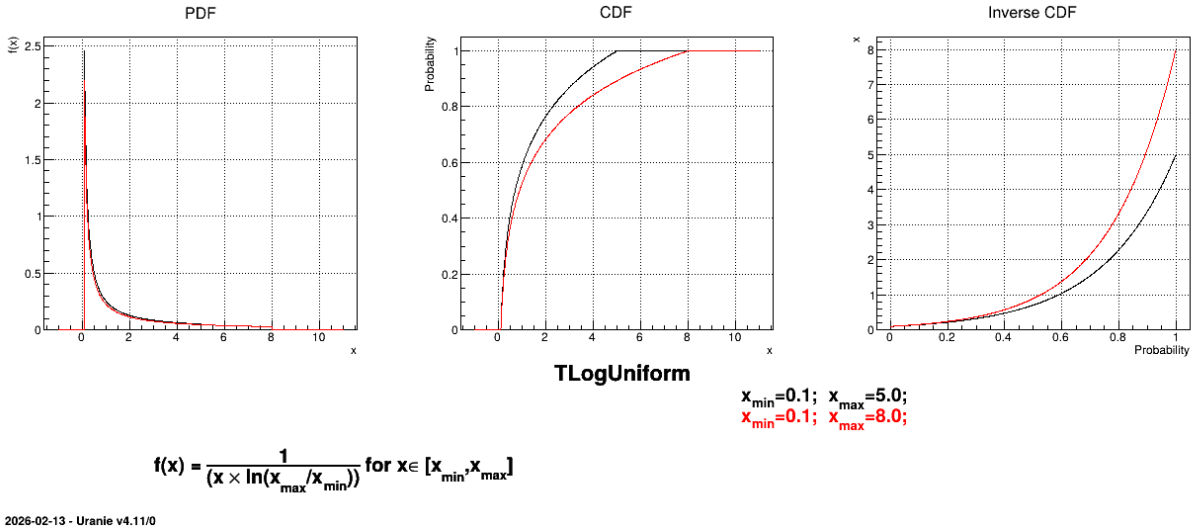

Figure 2.3: Example of PDF, CDF and inverse CDF for LogUniform distributions.

2.1.1.3 Triangular Law

This law describes a triangle with a base between a minimum and a maximum and a highest density at a certain point x_{mode}, so

$$f(x) = \frac{2 \times (x - x_{\text{min}})}{(x_{\text{max}} - x_{\text{min}}) \times (x_{\text{mode}} - x_{\text{min}})} \mathbb{1}_{[x_{\text{min}}, x_{\text{mode}}]}(x)$$

and

$$f(x) = \frac{2 \times (x_{\text{max}} - x)}{(x_{\text{max}} - x_{\text{min}}) \times (x_{\text{max}} - x_{\text{mode}})} \mathbb{1}_{[x_{\text{mode}}, x_{\text{max}}]}(x)$$

The mean value of the triangular law can then be computed as $\mu = \frac{x_{\text{max}} + x_{\text{min}} + x_{\text{mode}}}{3}$ while its variance can be written as $\sigma^2 = \frac{(x_{\text{max}}^2 + x_{\text{min}}^2 + x_{\text{mode}}^2 - x_{\text{max}}x_{\text{min}} - x_{\text{max}}x_{\text{mode}} - x_{\text{mode}}x_{\text{min}})}{18}$.

Figure 2.4 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

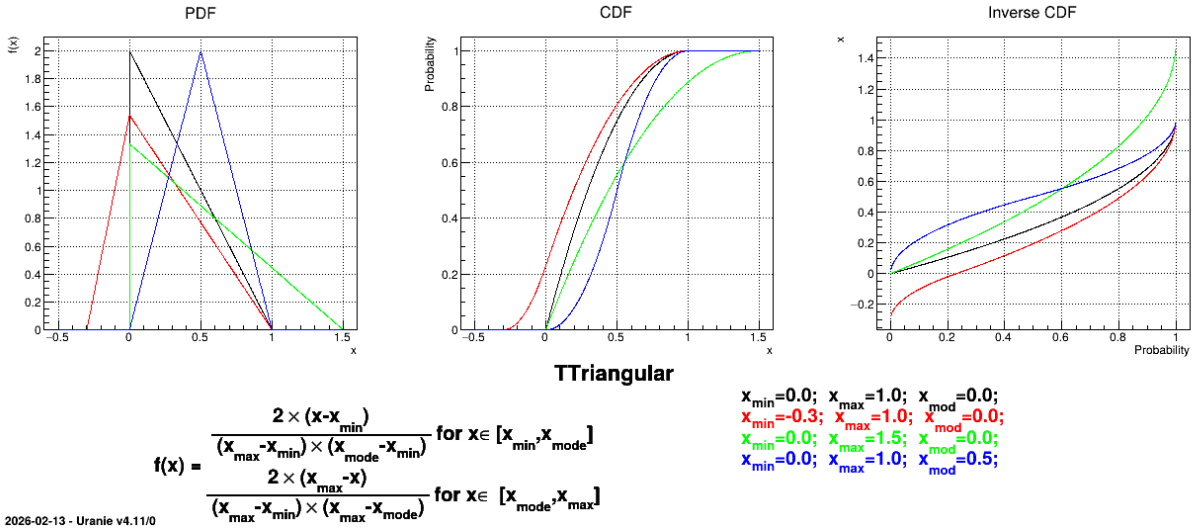

Figure 2.4: Example of PDF, CDF and inverse CDF for Triangular distributions.

2.1.1.4 LogTriangular Law

If a random variable x follows a LogTriangular distribution, the random variable $\ln(x)$ follows a Triangular distribution, so

$$f(x) = \frac{2 \times \ln(x/x_{\text{min}})}{x \times \ln(x_{\text{max}}/x_{\text{min}}) \times \ln(x_{\text{mode}}/x_{\text{min}})} \mathbb{1}_{[x_{\text{min}}, x_{\text{mode}}]}(x)$$

and

$$f(x) = \frac{2 \times \ln(x_{\text{max}}/x)}{x \times \ln(x_{\text{max}}/x_{\text{min}}) \times \ln(x_{\text{max}}/x_{\text{mode}})} \mathbb{1}_{[x_{\text{mode}}, x_{\text{max}}]}(x)$$

Figure 2.5 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

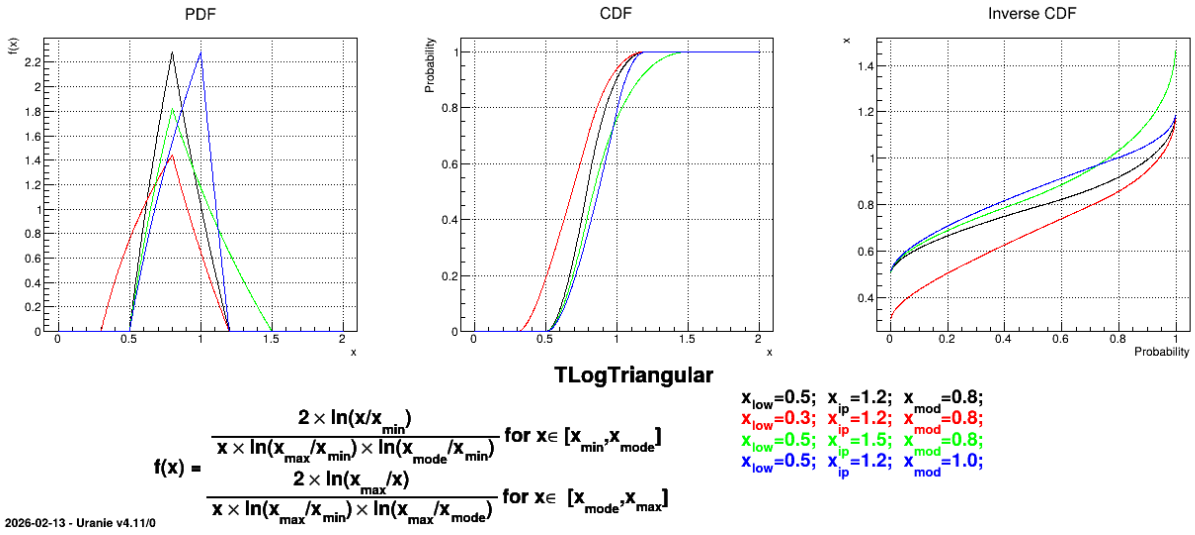

Figure 2.5: Example of PDF, CDF and inverse CDF for LogTriangular distributions.

2.1.1.5 Normal law

A normal law is defined with a mean μ (which coincide with the mode) and a standard deviation σ, as

$$f(x) = e^{\frac{-(x-\mu)^2}{2\sigma^2}} \times \frac{1}{\sqrt{2\pi\sigma^2}}$$

Figure 2.6 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

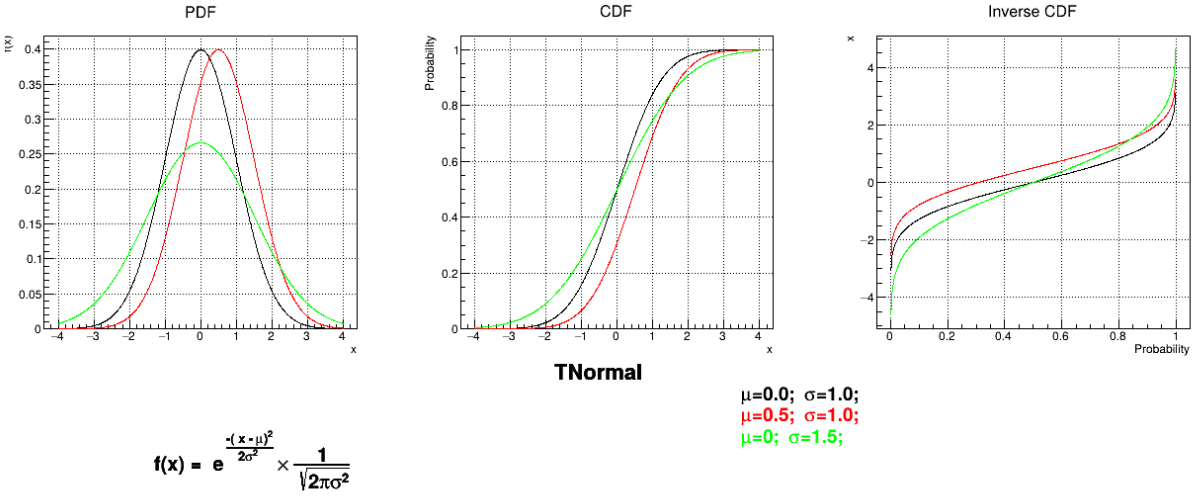

Figure 2.6: Example of PDF, CDF and inverse CDF for Normal distributions.

2.1.1.6 LogNormal law

If a random variable x follows a LogNormal distribution, the random variable $\ln(x)$ follows a Normal distribution (whose parameters are μ and σ), so

$$f(x) = \frac{1}{(x - x_{\min})\sigma\sqrt{2\pi}} \times e^{\frac{-(\ln(x-x_{\min})-\mu)^2}{2\sigma^2}} \mathbb{1}_{[x_{\min},+\infty[}(x)$$

In Uranie, it is parametrised by default using M, the mean of the distribution, E_f, the Error factor that represents the ration of the 95% quantile and the median ($E_f = q_{0.95}/q_{0.50}$) and the minimum $x_{\min}$. One can go from one parametrisation to the other following those simple relations

$$\begin{array}{rcl} M = e^{\mu+\sigma^2/2} + x_{\min} & \Leftrightarrow & \mu = \ln(M - x_{\min}) - \sigma^2/2 \\ E_F = e^{1.645\times\sigma} & \Leftrightarrow & \sigma = \ln(E_f)/1.645 \end{array}$$

The variance of the distribution can be estimated as $\text{Var} = (e^{\sigma^2} - 1)e^{2\mu+\sigma^2} = (e^{(\frac{\ln(E_f)}{1.645})^2} - 1) \times (M - x_{\min})^2$ while its mean is $e^{\mu+\sigma^2/2}$ and its mode is $e^{\mu-\sigma^2}$.

Figure 2.7 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

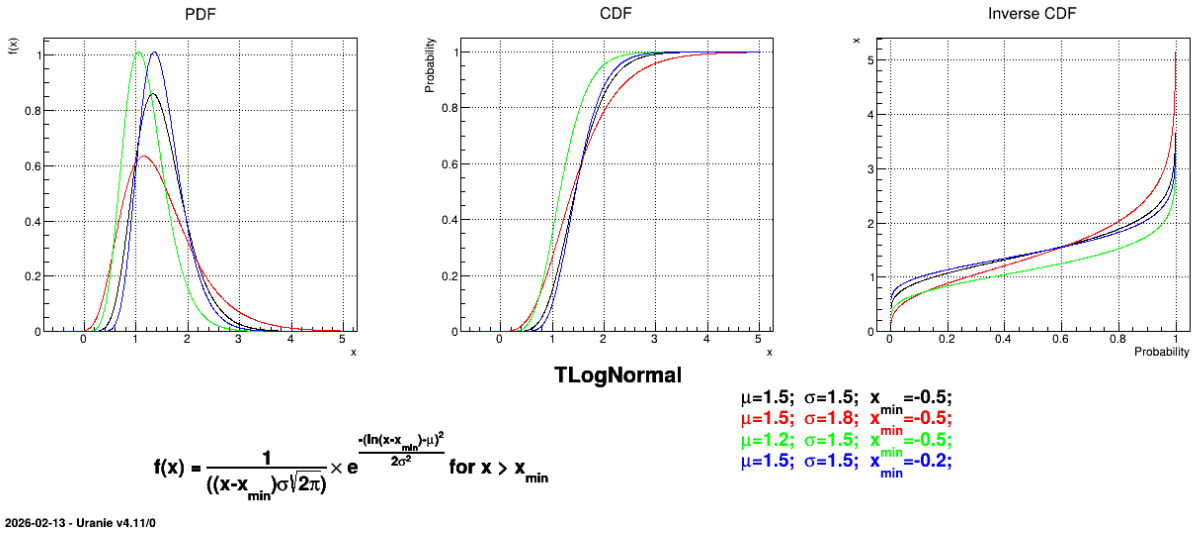

Figure 2.7: Example of PDF, CDF and inverse CDF for LogNormal distributions.

2.1.1.7 Trapezium law

This law describes a trapezium whose large base is defined between a minimum and a maximum and its small base lies between a low and an up value, as

$$f(x) = \frac{2}{(x_{\text{up}} - x_{\text{low}}) + (x_{\max} - x_{\min})} \times Y$$

where $Y = 1$ for $x \in [x_{\text{low}}, x_{\text{up}}]$, $Y = \dfrac{(x - x_{\min})}{(x_{\text{low}} - x_{\min})}$ for $x \in [x_{\min}, x_{\text{low}}]$ and $Y = \dfrac{(x_{\max} - x)}{(x_{\max} - x_{\text{up}})}$ for $x \in [x_{\text{up}}, x_{\max}]$.

For this distribution, the mean can be estimated through $\mu = \frac{1}{3(x_{\max}+x_{\text{up}}-x_{\text{low}}-x_{\min})}\left(\frac{x^3_{\max}-x^3_{\text{up}}}{x_{\max}-x_{\text{up}}} - \frac{x^3_{\text{low}}-x^3_{\min}}{x_{\text{low}}-x_{\min}}\right)$ while the variance is $\sigma^2 = \frac{1}{6(x_{\max}+x_{\text{up}}-x_{\text{low}}-x_{\min})}\left(\frac{x^4_{\max}-x^4_{\text{up}}}{x_{\max}-x_{\text{up}}} - \frac{x^4_{\text{low}}-x^4_{\min}}{x_{\text{low}}-x_{\min}}\right) - \mu^2$. The mode is not properly defined as all probability are equals in $[x_{\text{low}}, x_{\text{up}}]$

Figure 2.8 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

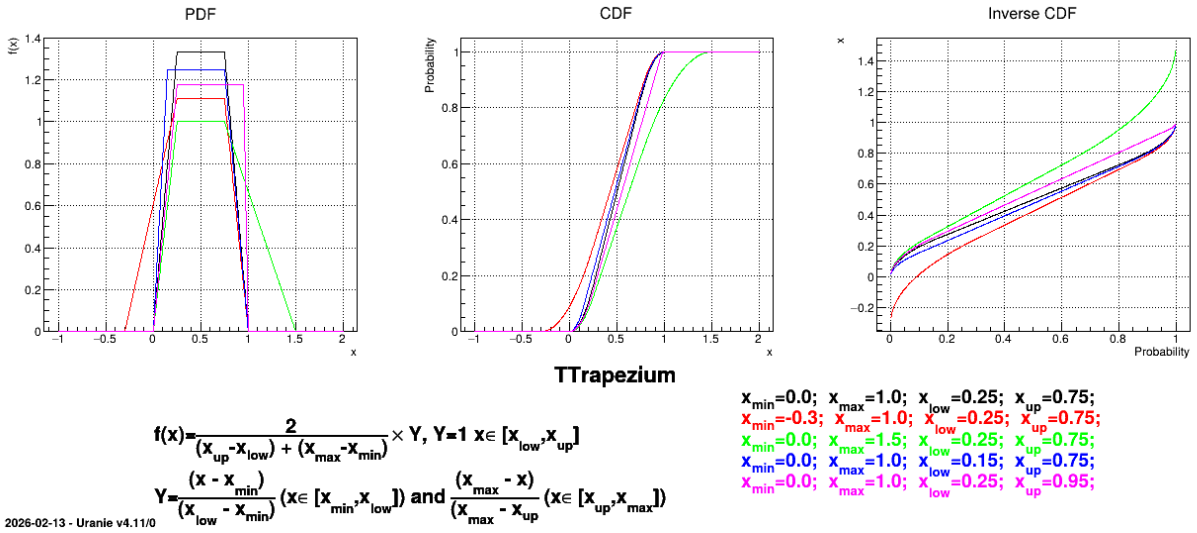

Figure 2.8: Example of PDF, CDF and inverse CDF for Trapezium distributions.

2.1.1.8 UniformByParts law

The UniformByParts law is defined between a minimum and a median and between the median and a maximum, as

$$f(x) = \frac{0.5}{(x_{\text{med}} - x_{\text{min}})} \mathbb{1}_{[x_{\text{min}}, x_{\text{med}}]}(x) \qquad \text{and} \qquad f(x) = \frac{0.5}{(x_{\text{max}} - x_{\text{med}})} \mathbb{1}_{[x_{\text{med}}, x_{\text{max}}]}(x)$$

For this distribution, the mean value is $\mu = 0.25 * (x_{\text{max}} + x_{\text{min}} + 2x_{\text{med}})$ while the variance is $\sigma^2 = \frac{1}{6} * (x_{\text{max}}^2 + x_{\text{min}}^2 + x_{\text{med}}(x_{\text{max}} + x_{\text{min}} + 2x_{\text{med}}))$.

Figure 2.9 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

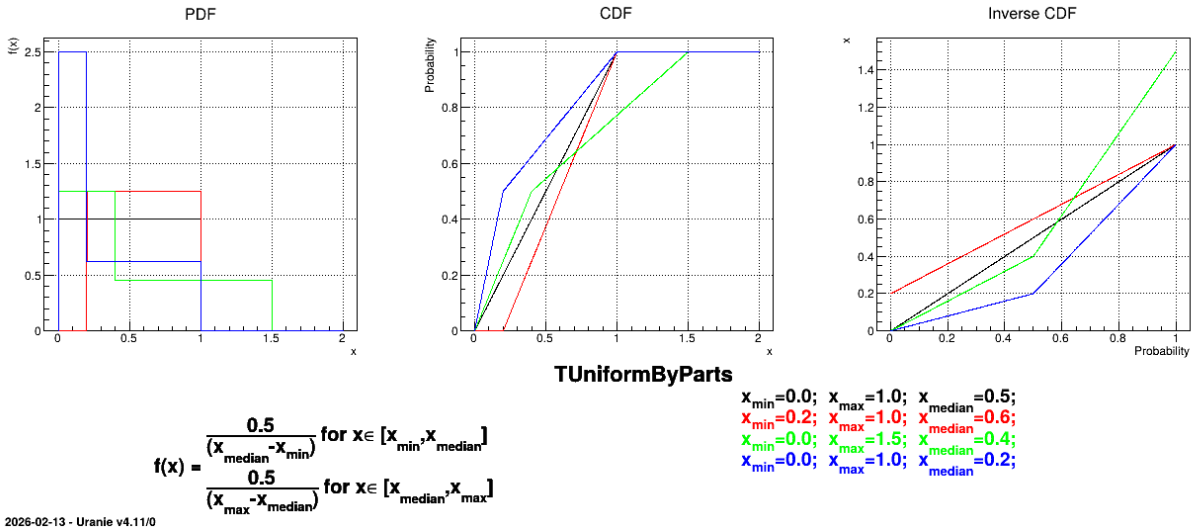

Figure 2.9: Example of PDF, CDF and inverse CDF for UniformByParts distributions.

2.1.1.9 Exponential law

This law describes an exponential with a rate parameter λ and a minimum $x_{\min}$, as

$$f(x) = \lambda \times e^{-\lambda \times (x - x_{\min})} \mathbb{1}_{[x_{\min}, +\infty[}(x)$$

The rate parameter λ should be positive.

The mean value of the exponential law can then be computed as $\mu = \lambda^{-1} + x_{\min}$ while its variance can be written as $\sigma^2 = \lambda^{-2}$. The mode is the chosen minimum value.

Figure 2.10 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

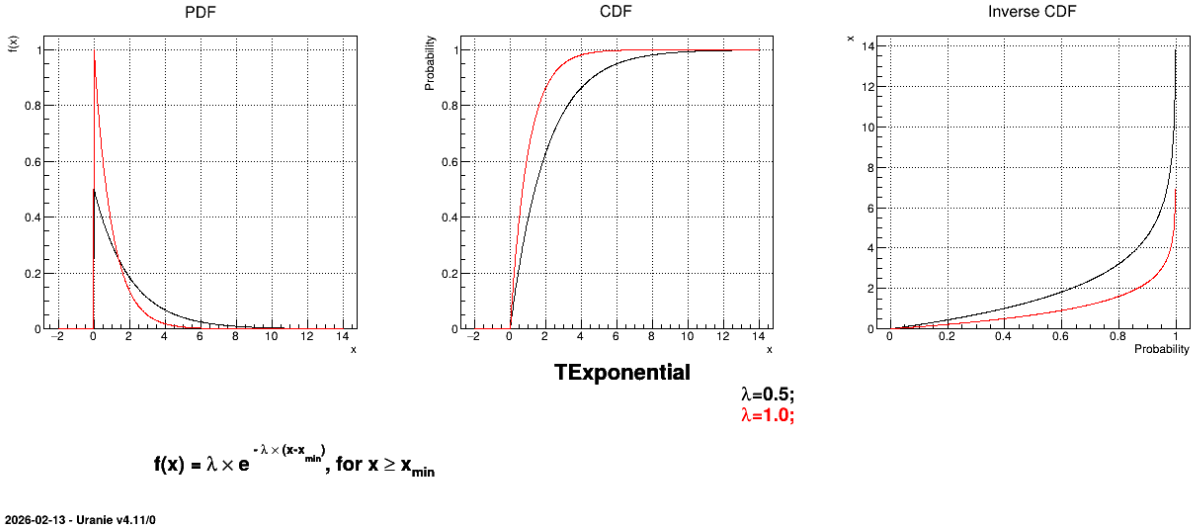

Figure 2.10: Example of PDF, CDF and inverse CDF for Exponential distributions.

2.1.1.10 Cauchy law

This law describes a Cauchy-Lorentz distribution with a location parameter x_0 and a scale parameter γ, as

$$f(x) = \frac{\gamma}{\pi \times (\gamma^2 + (x - x_0)^2)}$$

The mean and standard deviation of this distribution are not properly defined.

Figure 2.11 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

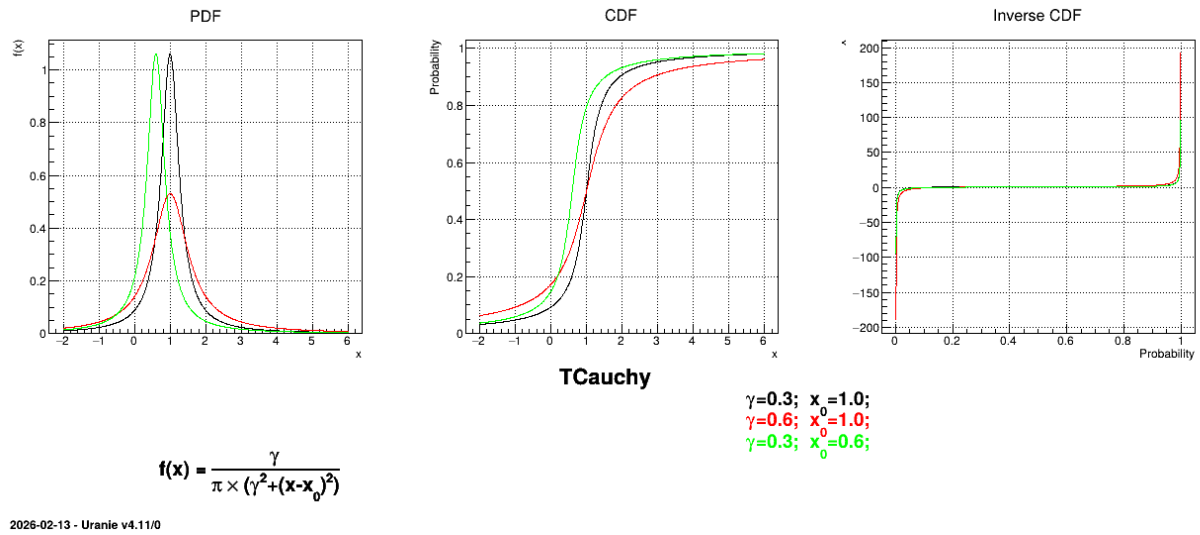

Figure 2.11: Example of PDF, CDF and inverse CDF for Cauchy distributions.

2.1.1.11 GumbelMax law

This law describes a Gumbel max distribution depending on the mode μ and the scale β, as

$$f(x) = z \times \frac{e^{-z}}{\beta}, \text{ where } z = e^{\frac{-(x-\mu)}{\beta}}$$

The mean value of the Gumbel max law can then be computed as mean $= \mu + \beta\gamma$, where γ is the Euler Mascheroni constant and its variance can be written as $\sigma^2 = \frac{\pi^2}{6}\beta^2$

Figure 2.12 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

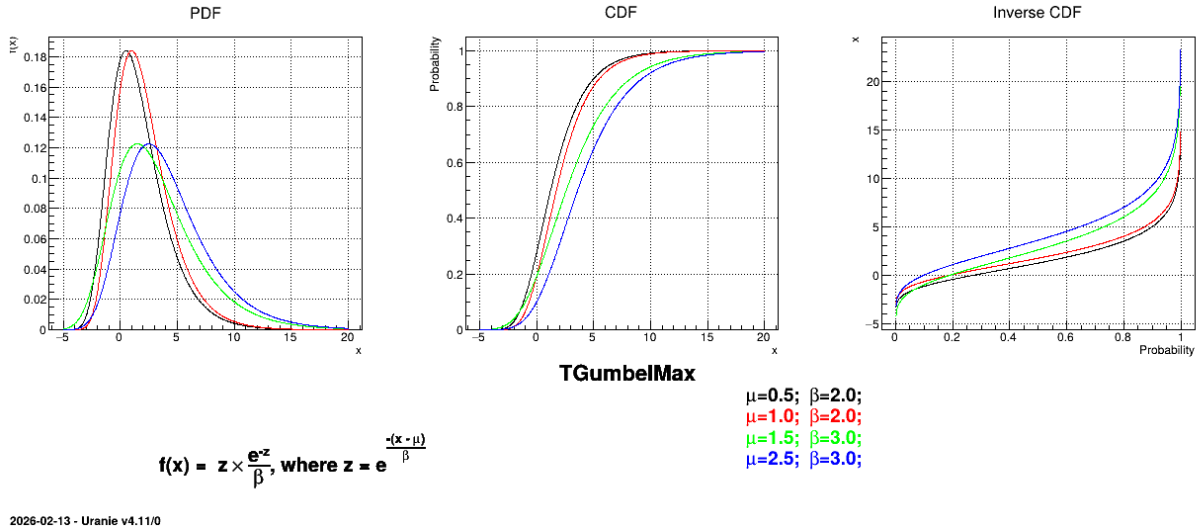

Figure 2.12: Example of PDF, CDF and inverse CDF for GumbelMax distributions.

2.1.1.12 Weibull law

This law describes a weibull distribution depending on the location $x_{\min}$, the scale λ and the shape k, as

$$f(x) = \frac{k}{\lambda} \times \left(\frac{x - x_{\min}}{\lambda} \right)^{k-1} \times e^{-\left(\frac{x-x_{\min}}{\lambda}\right)^k} \mathbb{1}_{[x_{\min}, +\infty[}(x)$$

The mean value of the Weibull law can then be computed as $\mu = \lambda\Gamma(1 + 1/k) + x_{\min}$ while its variance can be written as $\sigma^2 = \lambda[\Gamma(1 + 2/k) - (\Gamma(1 + 1/k))^2]$.

Figure 2.13 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

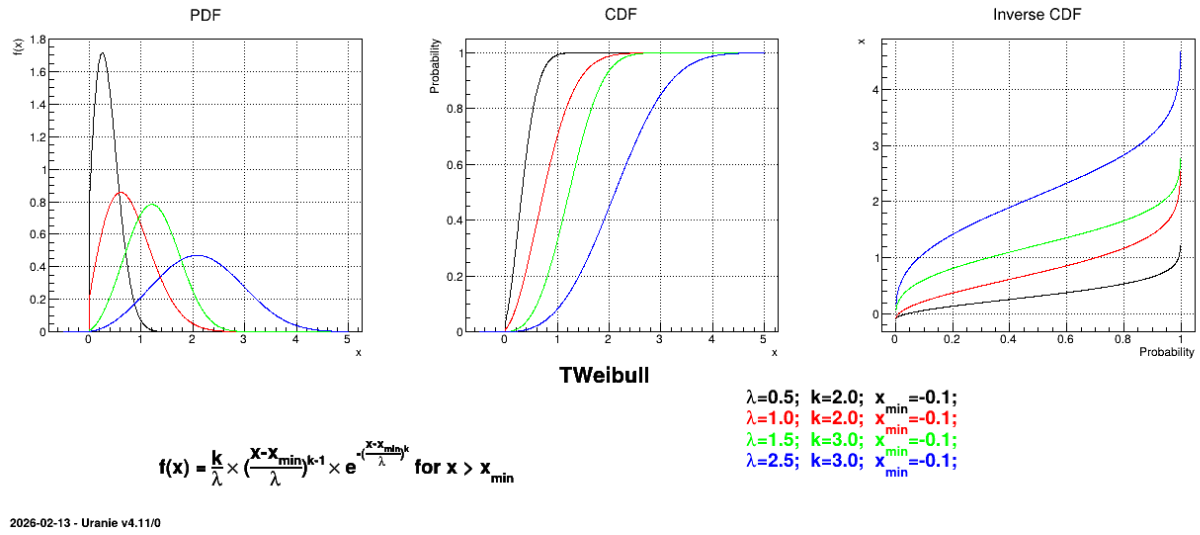

Figure 2.13: Example of PDF, CDF and inverse CDF for Weibull distributions.

2.1.1.13 Beta law

Defined between a minimum and a maximum, it depends on two parameters α and β, as

$$f(x) = \frac{Y^{\alpha-1} \times (1 - Y)^{\beta-1}}{B(\alpha, \beta)} \mathbb{1}_{[x_{\min}, x_{\max}]}(x)$$

where $Y = \dfrac{(x - x_{\min})}{(x_{\max} - x_{\min})}$ and $B(\alpha, \beta)$ is the beta function.

Figure 2.14 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

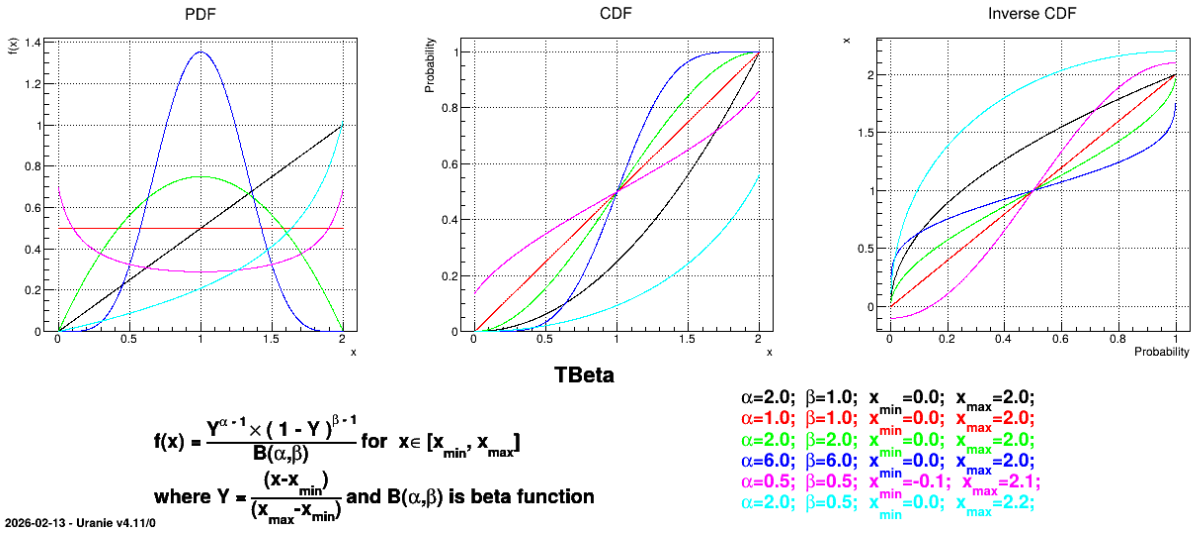

Figure 2.14: Example of PDF, CDF and inverse CDF for Beta distributions.

2.1.1.14 GenPareto law

This law describes a generalised Pareto distribution depending on the location μ, the scale σ and a shape ξ, as

$$f(x) = \frac{1}{\sigma} \times \left(1 + \xi\left(\frac{x-\mu}{\sigma}\right)\right)^{-(1/\xi+1)}$$

In this formula, σ should be greater than 0.

The resulting mean for this distribution can be estimated as $\mu + \sigma/(1-\xi)$ (for $\xi < 1$) while its variance can be computed as $\frac{\sigma^2}{(1-\xi)^2(1-2\xi)}$ (for $\xi < 0.5$).

Figure 2.15 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

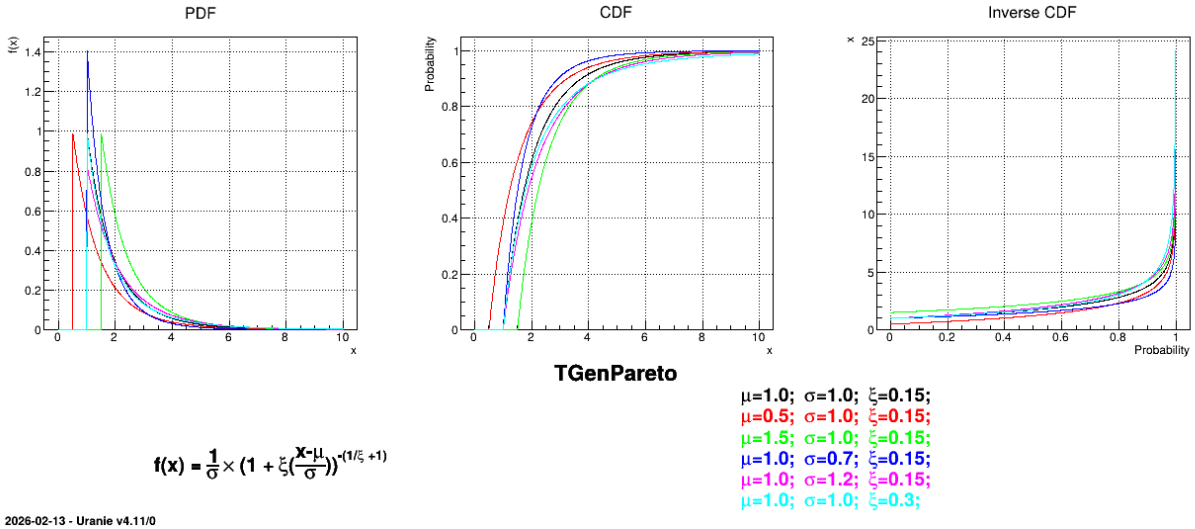

Figure 2.15: Example of PDF, CDF and inverse CDF for GenPareto distributions.

2.1.1.15 Gamma law

The Gamma distribution is a two-parameter family of continuous probability distributions. It depends on a shape parameter α and a scale parameter β. The function is usually defined for x greater than 0, but the distribution can be shifted thanks to the third parameter called location (ξ) which should be positive. This parametrisation is more common in Bayesian statistics, where the gamma distribution is used as a conjugate prior distribution for various types of laws:

$$f(x) = \frac{(x-\xi)^{\alpha-1} e^{-(x-\xi)/\beta}}{\Gamma(\alpha)\beta^{\alpha}} \mathbb{1}_{]\xi,+\infty]}(x)$$

The mean value of the Gamma law can then be computed as $\mu = \alpha\beta + \xi$ while its variance can be written as $\sigma^2 = \alpha\beta^2$.

Figure 2.16 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

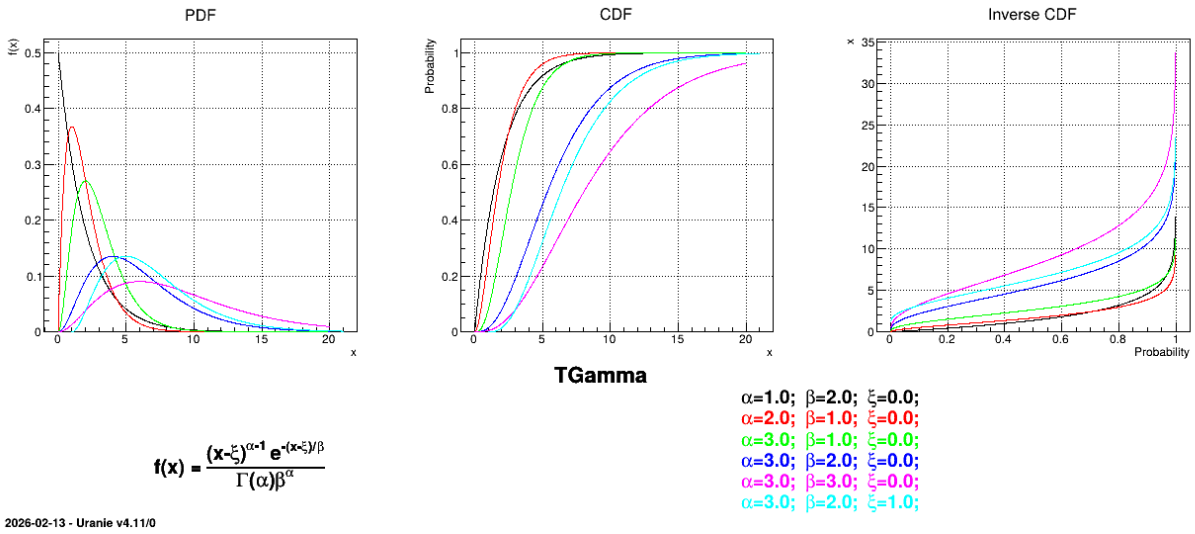

Figure 2.16: Example of PDF, CDF and inverse CDF for Gamma distributions.

2.1.1.16 InvGamma law

The inverse-Gamma distribution is a two-parameter family of continuous probability distributions. It depends on a shape parameter α and a scale parameter β. The function is usually defined for x greater than 0, but the distribution can be shifted thanks to the third parameter called location (ξ) which should be positive.

$$f(x) = \frac{\beta^{\alpha}(x-\xi)^{-\alpha-1} e^{-\beta/(x-\xi)}}{\Gamma(\alpha)} \mathbb{1}_{]\xi,+\infty]}(x)$$

The mean value of the Inverse-Gamma law can then be computed as $\mu = \beta/(\alpha-1) + \xi$ (for $\alpha > 1$) while its variance can be written as $\sigma^2 = \frac{\beta^2}{(\alpha-1)^2(\alpha-2)}$ (for $\alpha > 2$).

Figure 2.17 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

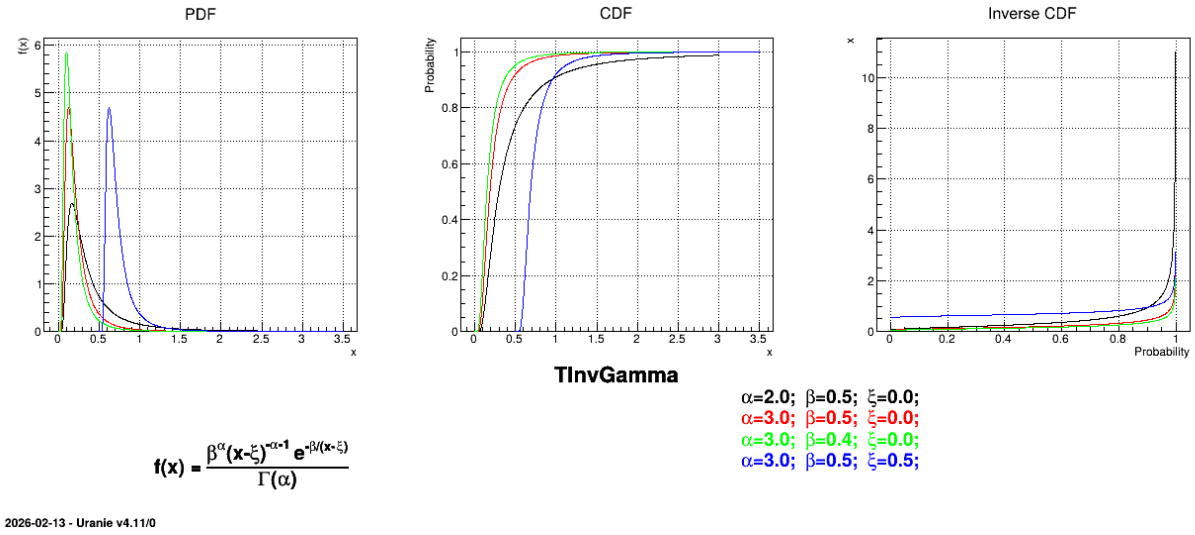

Figure 2.17: Example of PDF, CDF and inverse CDF for InvGamma distributions.

2.1.1.17 Student Law

The Student law is simply defined with a single parameter: the degree-of-freedom (**DoF**). The probability density function is then set as

$$f(x) = \frac{1}{\sqrt{k\pi}} \frac{\Gamma\left(\frac{k+1}{2}\right)}{\Gamma\left(\frac{k}{2}\right)} \left(1 + \frac{t^2}{k}\right)^{-\frac{k+1}{2}}$$

where Γ is the Euler's gamma function.

This distribution is famous for the t-test, a test-hypothesis developed by Fisher to check validity of the null hypothesis when the variance is unknown and the number of degree-of-freedom is limited. Indeed, when the number of degree-of-freedom grows, the shape of the curve looks more and more like the centered-reduced normal distribution. The mean value of the student law is 0 as soon as $k > 1$ (and is not determined otherwise). Its variance can be written as $\sigma^2 = \frac{k}{k-2}$ as soon as $k > 2$, infinity if $1 < k \leq 2$, and is not determined otherwise.

Figure 2.18 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

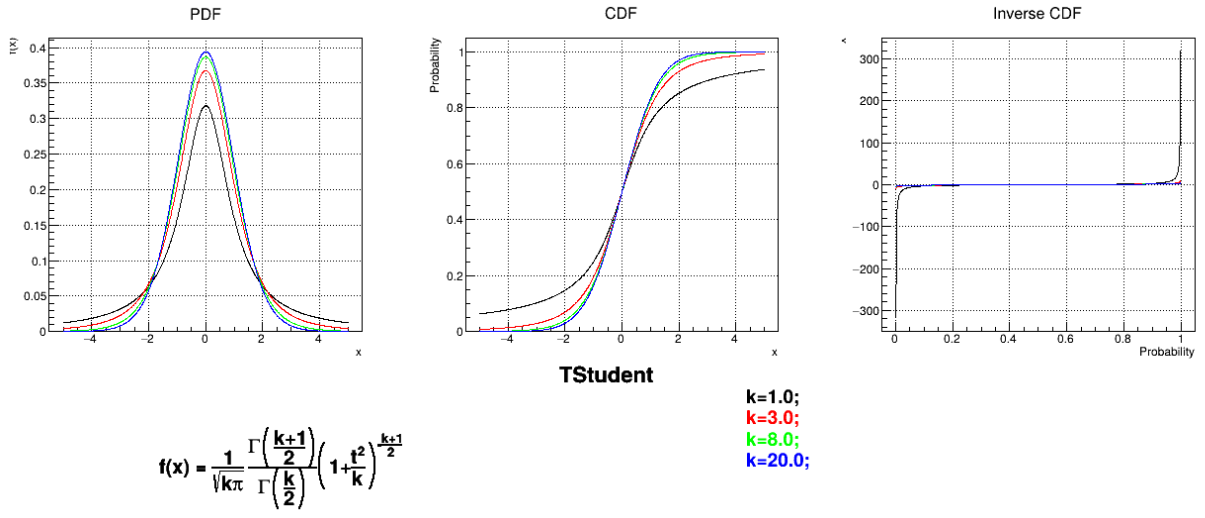

Figure 2.18: Example of PDF, CDF and inverse CDF for Student distributions.

2.1.1.18 Generalized normal law

This law describes a generalized normal distribution depending on the location μ, the scale α and the shape β, as

$$f(x) = \frac{\beta}{2\alpha\Gamma\left(1/\beta\right)} \times e^{-\left(\frac{x-\mu}{\alpha}\right)^{\beta}}$$

The mean value of the generalized normal law is μ while its variance can be written as $\sigma^2 = \frac{\alpha^2\Gamma(3/\beta)}{\Gamma(1/\beta)}$

Figure 2.19 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

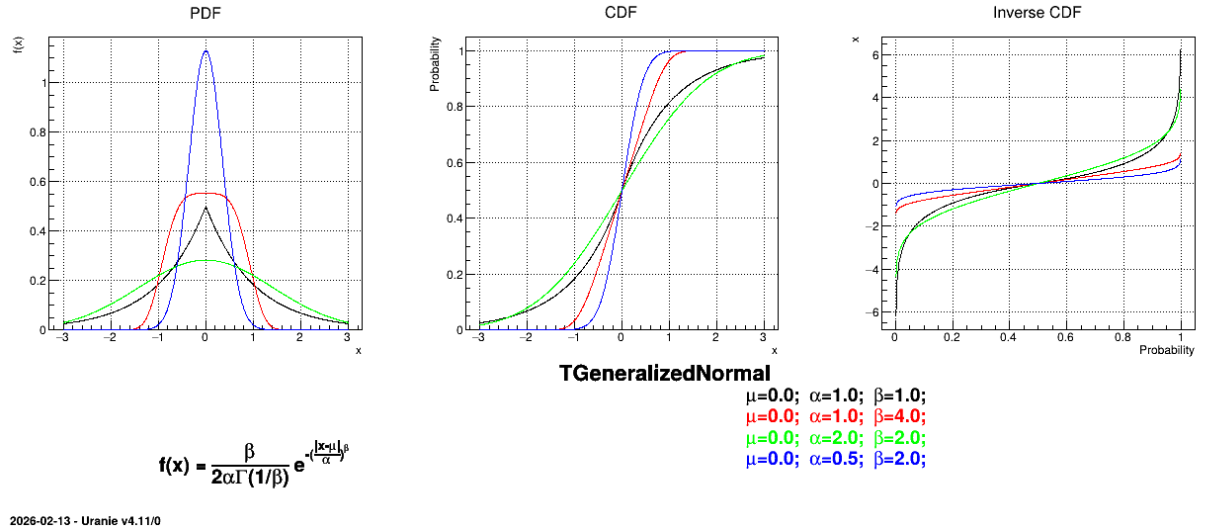

Figure 2.19: Example of PDF, CDF and inverse CDF for GeneralizedNormal distributions.

2.1.1.19 Composing law

It is possible to imagine a new law, hereafter called *composed law*, by combining different pre-existing laws in order to model a wanted behaviour. This law would be defined with N pre-existing laws whose densities are noted $\{f_j\}_{1\leq j\leq N}$, along with their relative weights $\{\omega_j\}_{1\leq j\leq N} \in (\mathbb{R}^+)^N$ and the resulting density is then written as

$$f(x) = \sum_{j=1}^{N} \omega_j f_j(x)$$

The mean value of this newly generated law can be expressed, assuming that all pre-existing laws have a finite and defined expectation denoted $\{\mu_j\}_{1\leq j\leq N}$, as $\mu = \sum_{j=1}^{N} \frac{\omega_j \mu_j}{S}$
where the sum of all weights is $S = \sum_{j=1}^{N} \omega_j$. As for the mean value, the variance of this newly generated law can be expressed, assuming that all pre-existing laws have a finite and defined expectation and variance, as done below in a very generic way.

$$\begin{aligned}
\mathrm{Var}_f &= \mathbb{E}_f(x^2) - (\mathbb{E}_f(x))^2 \\
&= \int x^2 f(x)dx - \left(\int x f(x)dx \right)^2, \text{ with } f(x) = \sum_{j=1}^{N} \omega_j f_j(x) \text{ where } \{\omega_\mathrm{j}\}_{1\leq \mathrm{j} \leq \mathrm{N}} \in (\mathbb{R}^+)^\mathrm{N} \\
&= \int \sum_{j=1}^{N} x^2 \frac{\omega_j f_j(x)}{S} dx - \left(\int \sum_{j=1}^{N} x \frac{\omega_j f_j(x)}{S} dx \right)^2, \text{ where } \mathrm{S} = \sum_{\mathrm{j}=1}^{\mathrm{N}} \omega_\mathrm{j} \\
&= \frac{1}{S} \sum_{j=1}^{N} \omega_j \underbrace{\int x^2 f_j(x)dx}_{\mathrm{Var}_{f_j}(x)+(\mathbb{E}_{f_j}(x))^2} - \frac{1}{S^2} \left(\sum_{j=1}^{N} \omega_j \underbrace{\int x f_j(x)dx}_{\mathbb{E}_{f_j}(x)} \right)^2 \\
&= \frac{1}{S} \sum_{j=1}^{N} \omega_j (\sigma_j^2 + \mu_j^2) - \frac{1}{S^2} \left(\sum_{j=1}^{N} \delta_j \right)^2, \text{ where } \delta_j = \omega_j \mu_j \ \forall j \in [1, N] \\
&= \frac{1}{S} \sum_{j=1}^{N} \omega_j \sigma_j^2 + \sum_{j=1}^{N} \frac{\delta_j^2}{S\omega_j} - \frac{1}{S^2} \left[\sum_{j=1}^{N} \delta_j^2 + 2 \sum_{1\leq i<j\leq N} \delta_i \delta_j \right] \\
&= \frac{1}{S} \sum_{j=1}^{N} \omega_j \sigma_j^2 + \sum_{j=1}^{N} \frac{S - \omega_j}{S^2 \omega_j} \delta_j^2 - \frac{2}{S^2} \sum_{1\leq i<j\leq N} \delta_i \delta_j.
\end{aligned}$$

In the case of unweighted composition, this can be written as $\mathrm{Var}_f = \frac{1}{N} \sum_{j=1}^{N} \sigma_j^2 + \frac{N-1}{N^2} \sum_{j=1}^{N} \mu_j^2 - \frac{2}{N^2} \sum_{1\leq i<j\leq N} \mu_i \mu_j$.

Figure 2.20 shows the PDF, CDF and inverse CDF generated for different sets of parameters.

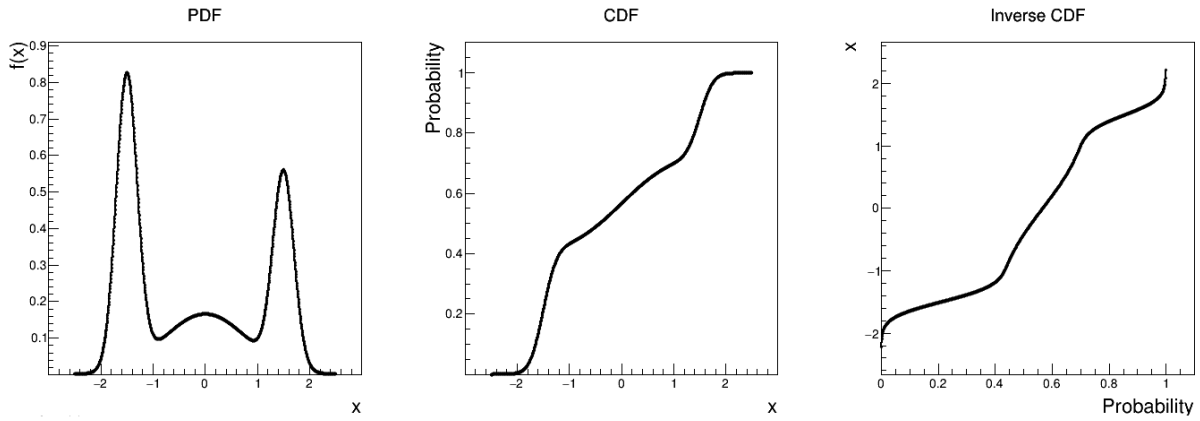

Figure 2.20: Example of PDF, CDF and inverse CDF for a composed distribution made out of three normal distributions with respective weights.

2.2 Statistical treatments and operations

There are many different kinds of operations that can be applied on an existing set of data (disregarding their origin, *i.e.* whether they come from experiments, simulations…). They are all listed below and the main ones are described in more details in the following subsections. For the sake of simplicity, the input variable is called x leading to an output variable called y. The dataset contains N points and i is be an iterator that goes from 0 to $N-1$. In few words, here what's easily calculable with Uranie:

- The normalisation of variable, in *Normalising the variable*
- The ranking of variable, in *Computing the ranking*
- The elementary statistic computation, in *Computing the elementary statistic*
- The quantile estimation, in *The quantile computation*
- The correlation matrix determination, in *Correlation matrix*

2.2.1 Normalising the variable

The normalisation function can be called to create new variables, for every requested normalisation, whose range and dispersion depend on the chosen normalisation method. Up to now, there are four different ways to perform this normalisation:

- centered-reducted: the new variable values are computed as $\tilde{x} = \dfrac{x - \mu_x}{\sigma_x}$
- centered: the new variable values are computed as $\tilde{x} = x - \mu_x$
- reduced to $[-1, 1]$: the new variable values are computed as $\tilde{x} = 2.0 \times \dfrac{x - x_{\text{Min}}}{x_{\text{Max}} - x_{\text{Min}}} - 1.0$
- reduced to $[0, 1]$: the new variable values are computed as $\tilde{x} = \dfrac{x - x_{\text{Min}}}{x_{\text{Max}} - x_{\text{Min}}}$

2.2.2 Computing the ranking

The ranking of variable is used in many methods that are focusing more on monotony than on linearity (this is discussed throughout this documentation when coping with regression, correlation matrix, *c.f.* for instance *The monotone case*). The way this is done in Uranie is the following: for every variable considered, a new variable is created, whose name is constructed as the name of the considered variable with the prefix "Rk_". The ranking consists in assigning to each variable entry an integer, that goes from 1 to the number of patterns, following an order relation (in Uranie it is chosen so that 1 is the smallest value and N is the largest one).

2.2.3 Computing the elementary statistic

When considering an existing set of points, it exists a method to determine the four simplest statistical notions: the minimum, maximum, average and standard deviation.

The minimum and maximum are trivially estimated by running over all the possible values. The average and standard deviation are estimated on the fly, using the following recursive formulae (where ζ_i represents the value of ζ using all data points up to i for $i = 1, \ldots, n_S$):

- average: μ_{x_0} is set to 0 and then

$$\mu_{x_i} = \mu_{x_{i-1}} \times \frac{i}{i+1} + \frac{x_i}{i+1}$$

- standard deviation: σ_{x_0} is set to 0 and then, for i strictly greater than 0,

$$\sigma_{x_i} = \sigma_{x_{i-1}} \times \frac{i-1}{i} + \frac{i+1}{i} \times \frac{(x_i - \mu_{x_i})^2}{i}$$

2.2.4 The quantile computation

There are several ways of estimating the quantiles implemented in Uranie. This part describes the most commonly used and starts with a definition of quantile.

A quantile x_p, as discussed in the following parts, for p a probability going from 0 to 1, is the lowest value of the random variable X leading to $P\{X \leq x_p\} = p$. This definition holds equally if one is dealing with a given probability distribution (leading to a theoretical quantile), or a sample, drawn from a known probability distribution or not (leading to an empirical quantile). In the latter case, the sample is split into two sub-samples: one containing pN points, the other one containing $(1-p)N$ points.

2.2.4.1 Empirical computation

For a given probability p, the corresponding quantile q is given by:

$$q = (1-p)x_k + px_{k+1}$$

where x_k is the k-Th smallest value of the variable set-of-value (whose size is N).

The way the index k is computed depends on how conservative one wants to be, but also on the case under consideration. For discontinuous cases, one can choose amongst the following list:

- $k = \lfloor p \times N \rfloor$; if $p \times N = k$, $q = x_k$. $q = x_{k+1}$ otherwise.
- $k = \lfloor p \times N \rfloor$; if $p \times N = k$, $q = 1/2 \times (x_k + x_{k+1})$. $q = x_{k+1}$ otherwise.
- $k = \lfloor p \times N - 0.5 \rfloor$; if $p \times N - 0.5 = k$ and k is even, $q = x_k$. $q = x_{k+1}$ otherwise.

For piece-wise linear interpolations, the estimation of k can be done in Uranie amongst the following cases:

- $k = \lfloor p \times N \rfloor$
- $k = \lfloor p \times N - 0.5 \rfloor$
- $k = \lfloor p \times (N+1) \rfloor$

- $k = \lfloor p \times (N-1) + 1 \rfloor$
- $k = \lfloor p \times (N + 1/3) + 1/3 \rfloor$, approximately median unbiased.
- $k = \lfloor p \times (N + 1/4) + 3/8 \rfloor$, approximately unbiased if x is normally distributed.

2.2.4.2 Wilks-quantile computation

The Wilks quantile computation is an empirical estimation, based on order statistic which allows to get an estimation on the requested quantile, with a given confidence level β, independently of the nature of the law, and most of the time, requesting less estimations than a classical estimation. Going back to the empirical way discussed in *Empirical computation*: it consists, for a 95% quantile, in running 100 computations, ordering the obtained values and taking the one at either the 95-Th or 96-Th position (see the discussion on how to choose k in *Empirical computation*). This can be repeated several times and will result in a distribution of all the obtained quantile values peaking at the theoretical value, with a standard deviation depending on the number of computations made. As it peaks on the theoretical value, 50% of the estimation are larger than the theoretical value while the other 50% are smaller (see Figure 2.21 for illustration purpose).

Wilks computation on the other hand request not only a probability value but also a confidence level. The quantile x_p^β represents the x_p quantile given the p probability but this time, the value is provided with a $\beta\%$ confidence level, meaning that $\beta\%$ of the obtain value is larger than the theoretical quantile. This is a way to be conservative and to be able to quantify how conservative one wants to be. To do this, the size of the sample must follow a necessary condition:

$$n > \frac{\ln(1-\beta)}{\ln p}$$

This is the smallest sample size to get an estimation, and, in most cases, the accuracy reached (for a given sample size) is better than the one achieved with the simpler solution provided above. It is also possible to increase the sample size to get a better description of the quantile estimation.

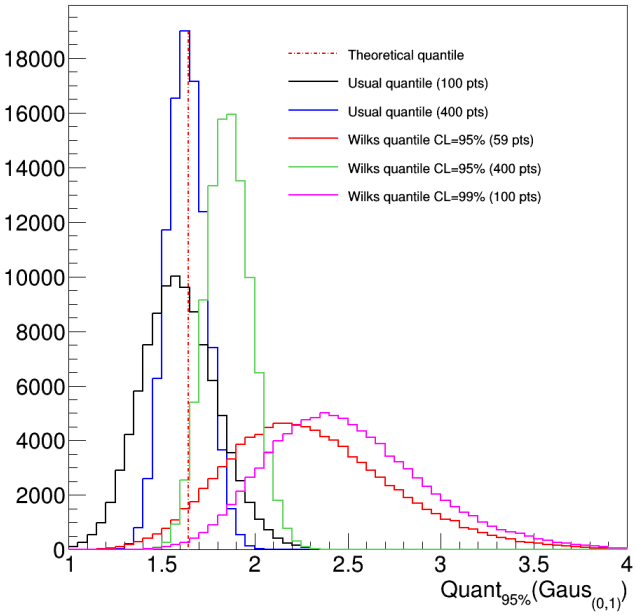

Figure 2.21: Illustration of the results of 100000 quantile determinations, applied to a reduced centered gaussian distribution, comparing the usual and Wilks methods. The number of points in the reduced centered gaussian distribution is varied, as well as the confidence level.

Figure 2.21 shows a simple case: the estimation of the value of the 95% quantile of a centered-reduced normal distribution. The theoretical value (red dashed line) is compared to the results of 100000 empirical estimation, following the simple recipe (black and blue curves) or the Wilks method (red, green and magenta curves). Several conclusions can be drawn:

- The simpler quantile estimation average is slightly biased with respect to the theoretical value. This is due to the choice of k, discussed in *Empirical computation* which can lead to under or over estimation of the quantile value. The bias becomes smaller with the increasing sample size.
- The standard deviation of the distributions (whatever method is considered) is becoming smaller with the increasing sample size.
- When using the Wilks method, the fraction of event below the theoretical value is becoming smaller with the increasing confidence level.

2.2.5 Correlation matrix

The computation of the correlation matrix can be done either on the values (leading to the Pearson coefficients) or on the ranks (leading to the Spearmann coefficients).

Correlation matrices are computed in a 3 steps procedure detailed below:

- An overall M matrix is created and filled, every line being a new entry while every column is a variable
- This matrix is centred and reduced: for every variable under consideration μ_x is subtracted and the results is divided by σ_x.
- The resulting correlation matrix is obtained from the product ${}^tM \times M$

2.3 Combining these aspects: performing PCA

This part is introducing an example of analysis that combines all the aspects discussed up to now: handling data, perform a statistical treatment and visualise the results. This analysis is called PCA for *Principal Component Analysis* and is often used to

- gather event in a sample that seem to have a common behaviour;
- reduce the dimension of the problem under study.

There is a very large number of articles, even books, discussing the theoretical aspects of principal component analysis (for instance one can have a look at [Jol11]).

2.3.1 Theoretical introduction

2.3.1.1 Purpose

The principle of this kind of analysis is to analyse a provided ensemble, called hereafter $\mathcal{D}$, whose size is n_S, and which can be written as

$$\mathcal{D} = \{\mathbf{x}^i\}_{i=1,\dots,n_S}$$

where $\mathbf{x}^i$ is the i-Th input vector, written as $\mathbf{x}^i = (x_1^i, \dots, x_{n_X}^i)$ where n_X is the number of quantitative variable. It is basically a set of realisation of n_X random variables whose properties are completely unknown.

The aim is then to summarise (project/reduce) this sample into a smaller dimension space q (with $1 \leq q \leq n_X$) these q factors being chosen in order to maximise the inertia and being orthogonal one to another[1]. By doing so, the goal is to be able to reduce the dimension of our problem while loosing as few information as possible.

[1] As a reminder, the dispersion of a quantitative variable is usually represented with its variance (or standard deviation), the inertia criteria is, for multi-dimension problems, the sum of all the variable's variance.

2.3.1.2 Implementation in a nutshell

If one calls $\mathbf{X}$ the original sample whose dimension is (n_X, n_S), the idea behind PCA is to find the projection matrix $\mathbf{P}$, whose dimension is (n_X, n_X), that would re-express the data optimally as a new sample, called hereafter $\mathbf{Y}$, with the same dimension (n_X, n_S). The rows of $\mathbf{P}$ are forming a new basis to represent the column of $\mathbf{X}$ and this new basis will later become our principal component directions.

Now recalling the aim of PCA, the way to determine this projection matrix is crucial and should be designed as to

- find out the best linear combinations between variables so that the minimum number of rows (principal components) of $\mathbf{P}$ are considered useful to carry on as much inertia as possible;
- rank the principal component so that, if not satisfy with the new representation, it would be simple to add an extra principal component to improve it.

This can be done by investigating the covariance matrix $\mathbf{C_X}$ of $\mathbf{X}$ that, by definition, describes the linear combination between variables and that could be computed from the centered matrix sample $\mathbf{X_C}$[1] as

$$\mathbf{C_X} = \frac{1}{n_S - 1}\mathbf{X_C}{\mathbf{X_C}}^T$$

If one consider the resulting covariance matrix $\mathbf{C_Y}$, the aim is to maximise the signal measured by variance (diagonal entries that represents the variance of the principal components) while minimising the covariance between them. As the lowest covariance value reachable is 0, if the desired covariance matrix $\mathbf{C_Y}$ would append to be diagonal, this would mean our objectives are achieved. From the very definition of the covariance matrix, one could see that

$$\mathbf{C_Y} = \frac{1}{n_S - 1}\mathbf{Y_C}{\mathbf{Y_C}}^T = \frac{1}{n_S - 1}(\mathbf{PX_C})(\mathbf{PX_C})^T = \frac{1}{n_S - 1}\mathbf{PX_C}{\mathbf{X_C}}^T\mathbf{P}^T = \mathbf{PC_X}\mathbf{P}^T$$

As $\mathbf{C_X}$ is symmetric, it is orthogonal diagonalisable, and can be written $\mathbf{C_X} = \mathbf{ESE}^T$. In this equation, $\mathbf{E}$ is an orthonormal matrix whose columns are the orthonormal eigenvectors of $\mathbf{C_X}$, and $\mathbf{S}$ is a diagonal matrix which has the eigenvalues of $\mathbf{C_X}$. Given this, if we choose $\mathbf{P} = \mathbf{E}^T$, this leads to

$$\mathbf{C_Y} = \mathbf{PC_X}\mathbf{P}^T = \mathbf{E}^T\mathbf{ESE}^T\mathbf{E} = \mathbf{S}$$

At this level, there is no unicity of the $\mathbf{S}$ matrix as one can have many permutations of the eigenvalues along the diagonal, as long as one changes $\mathbf{E}$ accordingly.

Finally, an interesting link can be drawn between this protocol and a very classical method of linear algebra, already mentioned in other places of this document, called the Singular Value Decomposition (*SVD*[2]) leading to

Equation 2.1: General form of a SVD

$$\mathbf{X_C}^T = \mathbf{U\Sigma V}^T \text{ where } \mathbf{X_C}^T(n_S, n_X),\ \mathbf{U}(n_S, n_S),\ \mathbf{V}(n_X, n_X) \text{ and } \mathbf{\Sigma}(n_S, n_X) \tag{2.1}$$

In this context $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices (also known as respectively the *left singular vectors* and *right singular vectors* of $\mathbf{X_C}^T$) while $\mathbf{\Sigma}$ is a diagonal matrix storing the singular values of $\mathbf{X_C}^T$ in decreasing order. The last step is then to state the linear algebra theorem which says that the non-zero singular values of $\mathbf{X_C}^T$ are the square roots of the nonzero eigenvalues of $\mathbf{X_C}{\mathbf{X_C}}^T$ and ${\mathbf{X_C}}^T\mathbf{X_C}$ (the corresponding eigenvectors being the columns of respectively $\mathbf{V}$ and $\mathbf{U}$).

Gathering all this, one can see that by doing the SVD on the centered original sample matrix, the resulting projection matrix can be identified as $\mathbf{P} = \mathbf{V}^T$ and the resulting covariance matrix will be proportional to $\mathbf{\Sigma}^2$. The final interesting property is coming from the SVD itself: as $\mathbf{\Sigma}^2$ gathers the eigenvalues in decreasing order, it assures the unicity of the transformation and give access to the principal component in a hierarchical way.

---

[1] The centered matrix is defined as $\mathbf{X_C} = \mathbf{X} - \bar{\mathbf{x}}^T\mathbf{1_{n_S}}$ where $\bar{\mathbf{x}}$ is the vector of mean value for every variable and $\mathbf{1_{n_S}}$ is a vector of 1 whose dimension is n_S.

[2] SVD is applied to matrix whose number of rows should be greater than its number of columns.

2.3.1.3 Limitation of PCA

From what has been discussed previously it can appear very appealing, but there are few drawbacks or at least limitations that can be raised:

- This method is very sensitive to extreme points: correlation coefficient can be perturbed by them.
- In the case of non-linear phenomenon, the very basic concept of PCA collapses. Imagine a simple circle-shaped set of points, there are no correlation between the two variables, so no smaller space can be found using linear combinations.
- Even if the PCA is working smoothly, one has to be able to find an interpretation of the resulting linear combinations that have been defined to create the principle component. Moreover, it might not be possible to move along on more refined analysis, such as sensitivity analysis for instance.

CHAPTER

THREE

THE SAMPLER MODULE

Abstract This part is a description of the method used to produce the design-of-experiments, which is a crucial part in uncertainty propagation but also for sensitivity analysis and *surrogate model* generation.

3.1 Introduction

The Uranie-sampling module is used to produce design-of-experiments knowing the expected behaviour of the input variables for the problem under consideration. The framework of our approach can be illustrated in the following schematic view:

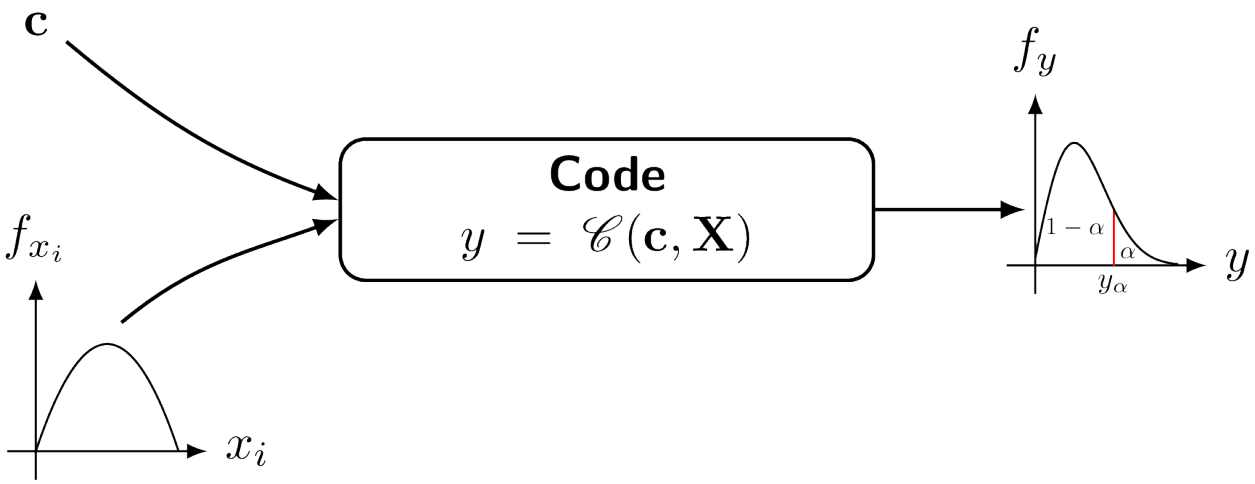

Figure 3.1: Schematic view of the input/output relation through a code

- We will denote as $\mathcal{C}$ the studied computational code which, generally, has two types of inputs:
  - The **constant** parameters which are gathered in the vector $\mathbf{c} \in \mathbb{R}^{n_C}$. They represent constants.
  - The **uncertain** parameters which are gathered in the vector $\mathbf{X} \in \mathbb{R}^{n_X}$

    It shall be noticed that these parameters are supposed to be **uncertain** either because of a lack of knowledge on their actual value or because of their intrinsic random nature.
- The result of the code $\mathcal{C}$ for a given set of parameters $(\mathbf{c}, \mathbf{X})$ gives the vector $y \in \mathbb{R}^{n_Y} = \mathcal{C}(\mathbf{c}, \mathbf{X})$ which contains all the output variables of the analysis.

Most of the time, the code $\mathcal{C}$ implies solving equations with partial derivatives in more or less complex configurations. The most well-known method to handle this is to generate a sample, as representative as possible of the behaviour of all input variables, in order to fully cover the input parameter phase space. This is the definition of a design-of-experiments

which can be generated in many different ways (depending on the analysis purpose, the laws under consideration…). The rest of this section mainly introduces concepts used throughout this documentation; more details can be found either in the user manual or in references.

Different methods exist to obtain a design-of-experiments from uncertain parameters which can be classified into two categories:

1. **stochastic** methods (see *The Stochastic methods*). These methods consist in using a random number generator to produce new samples. This is also called Monte-Carlo.
2. **deterministic** methods (see *QMC method*). Two distinct calls with the same parameters will always give the same point in a design-of-experiments. Some of these methods (those discussed below) are sequences which are sometimes called quasi-Monte Carlo (qMC).

3.2 The Stochastic methods

3.2.1 Introduction

In these methods the knowledge (or mis-knowledge) of the model is encoded in the choice of probability law used to describe the inputs x_i, for $i \in [0,\, n_X]$. These laws are usually defined by:

- a range that describes the possible values of x_i
- the nature of the law, which has to be taken in the list of pre-defined laws already presented in *The probability distributions*

A choice has frequently to be made between two implemented methods of drawing:

SRS (Simple Random Sampling):
This method consists in independently generating the samples for each parameter following its own probability density function. The obtained parameter variance is rather high, meaning that the precision of the estimation is poor leading to a need for many repetitions in order to reach a satisfactory precision. An example of this sampling when having two independent random variables (uniform and normal one) is shown in Figure 3.3-left. In order to get this drawing, the variable are normalised from 0 to 1 and a random drawing is performed in this range. The obtained value is computed calling the inverse CDF function corresponding to the law under study (that one can see from Figure 2.2 until Figure 2.18).

LHS (Latin Hypercube Sampling):
this method [MBC00] consists in partitioning the interval of each parameter so as to obtain segments of equal probabilities, and afterwards in selecting, for each segment, a value representing this segment. An example of this sampling when having two independent random variables (uniform and normal one) is shown in Figure 3.3-right. In order to get this drawing, the variable are normalised from 0 to 1 and this range is split into the requested number of points for the design-of-experiments. Thanks to this, a grid is prepared, assuring equi-probability in every sub-space. Finally, a random drawing is performed in every sub-range. The obtained value is computed calling the inverse CDF function corresponding to the law under study (that one can see from Figure 2.2 until Figure 2.18).

The first method is fine when the computation time of a simulation is "satisfactory". As a matter of fact, it has the advantage of being easy to implement and to explain; and it produces estimators with good properties not only for the mean value but also for the variance. Naturally, it is necessary to be careful in the sense to be given to the term "satisfactory". If the objective is to obtain quantiles for extreme probability values α (*i.e.* α = 0.99999 for instance), even for a very low computation time, the size of the sample would be too large for this method to be used. When a computation time becomes important, the LHS sampling method is preferable to get robust results even with small-size samples (*i.e.* N_{calc} = 50 to 200) [HD02]. On the other hand, it is rather trivial to double the size of an existing SRS sampling, as no extra caution has to be taken apart from the random seed.

In Figure 3.2, we present two samples of size N_{calc} = 8 coming from these two sampling methods for two random variables U_1 according to a gaussian law, and U_2 a uniform law. To make the comparison easier, we have represented on both figures the partition grid of equiprobable segments of the LHS method, keeping in mind that it is not used by the SRS method.

These figures clearly show that for LHS method each variable is represented on the whole domain of variation, which is not the case for the SRS method. This latter gives samples that are concentrated around the mean vector; the extremes of distribution being, by definition, rare.

Concerning the LHS method (right figure), once a point has been chosen in a segment of the first variable U_1, no other point of this segment will be picked up later, which is hinted by the vertical red bar. It is the same thing for all other variables, and this process is repeated until the N_{calc} points are obtained. This elementary principle will ensure that the domain of variation of each variable is totally covered in a homogeneous way. On the other hand, it is absolutely not possible to remove or add points to a LHS sampling without having to regenerate it completely. A more realistic picture is draw in Figure 3.3 with the same laws, both for SRS on the left and LHS on the right which clearly shows the difference between both methods when considering one-dimensional distribution.

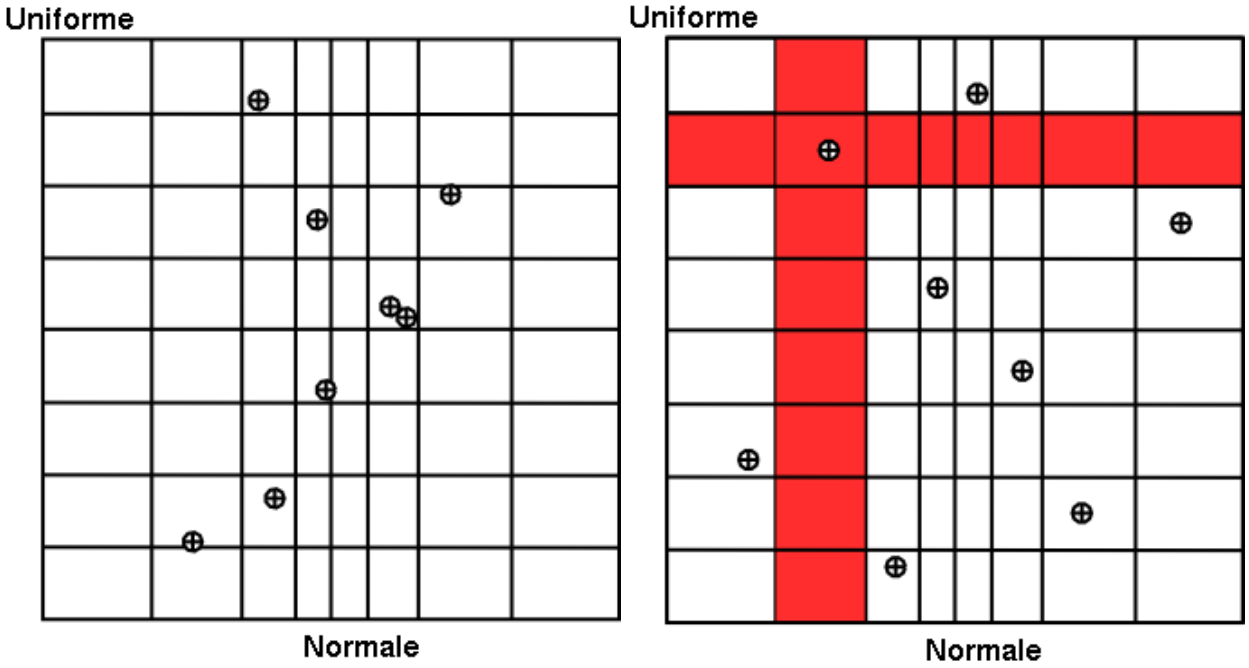

Figure 3.2: Comparison of the two sampling methods SRS (left) and LHS (right) with samples of size 8.

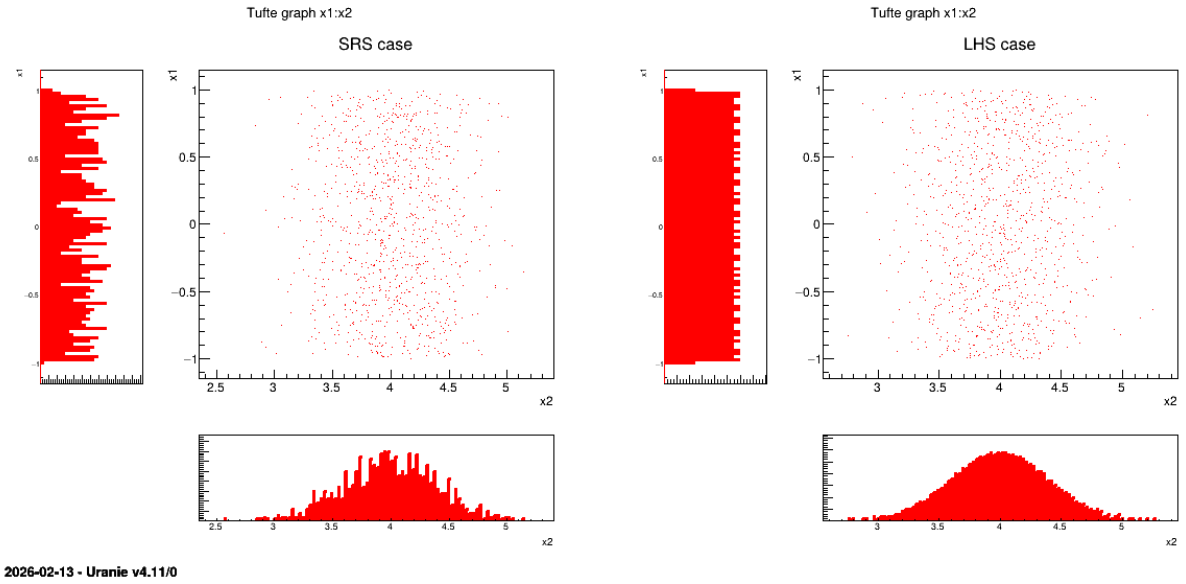

Figure 3.3: Comparison of deterministic design-of-experiments obtained using either SRS (left) or LHS (right) algorithm, when having two independent random variables (uniform and normal one)

There are two different sub-categories of LHS design-of-experiments discussed here and whose goal might slightly differs from the main LHS design discussed above:

- the maximin LHS: this category is the result of an optimisation whose purpose is to maximise the minimal distance between any sets of two locations. This is discussed later-on in *The maximin LHS*.
- the constrained LHS: this category is defined by the fact that someone wants to have a design-of-experiments fulfilling all properties of a Latin Hypercube Design but adding one or more constraints on the input space definition (generally inducing correlation between varibles). This is also further discussed in *The constrained LHS*.

Once the nature of the law is chosen, along with a variation range, for all inputs x_i, the correlation between these variables has to be taken into account. It is doable by defining a correlation coefficient but the way it is treated from one sampler to the other is tricky and is further discussed in the next section.

3.2.2 Correlating samples drawn from different marginals

This section is introducing the way Uranie classes are introducing correlation between random variables when considering either the Pearson of Spearman correlation coefficients. The idea is to better explain the expected behaviour while remaining at this level of correlation description (not going deep into the copula notion).

3.2.2.1 Notation convention

Let start by discussing the definition of a correlation matrix that connect (or not) a variable with the others. For a given problem with n_X variables, the covariance between two variables (denoted $\mathrm{Cov}(X_i, X_j)$) and their linear correlation (denoted $\rho_{X_i X_j}$) can be estimated as

Equation 3.1: Covariance and correlation between two variables

$$\mathrm{Cov}(X_i, X_j) = \mathbb{E}\left[(x_i - \mu_i)(x_j - \mu_j)\right] \quad \text{and} \quad \rho_{X_i X_j} = \frac{\mathrm{Cov}(X_i, X_j)}{\sigma_{X_i}\sigma_{X_j}} \tag{3.1}$$

In the equation above μ and σ are respectively the mean and standard deviation of the random variable under consideration. The coefficients that should be provided by the user are the correlation one, called the Pearson ones (as they've been

estimated using values of the random variables, but this is further discussed at the end of this section and also in *Theoretical aspects*). The idea is to gather all these coefficients in matrix, called hereafter the correlation matrix, that can be written as

$$\mathbf{C}(n_X, n_X) = \begin{pmatrix} 1 & \rho_{X_1X_2} & \cdots & \rho_{X_1X_{n_X}} \\ \rho_{X_2X_1} & 1 & \cdots & \rho_{X_2X_{n_X}} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{X_{n_X}X_1} & \rho_{X_{n_X}X_2} & \cdots & 1 \end{pmatrix}$$

Depending on the reference, one can discuss either the correlation matrix ($\mathbf{C}$) or the covariance matrix ($\mathbf{C_o}$). Going from one to the other is trivial if one defines $\mathbf{D}$ the diagonal matrix of dimension n_X whose coefficients are the standard deviation of the random variables, then

$$\mathbf{C} = \mathbf{D}^{-1}\,\mathbf{C_o}\,\mathbf{D}^{-1} \quad \Leftrightarrow \quad \mathbf{C_o} = \mathbf{D}\,\mathbf{C}\,\mathbf{D}$$

3.2.2.2 Correlation / de-correlation

Let's assume we have a random drawing $\mathbf{X}(n_S, n_X)$, where every column is the drawing of a given random variable of size n_S. On can then compute the following matrix $\mathbf{T} = n_S^{-1}\mathbf{X}^T\mathbf{X}$ which is the correlation matrix (respectively covariance matrix) of our sample, if the columns have been centered and reduced (respectively only centered). If n_S were to be infinite, we would be able to state that the resulting empirical correlation of the drawn marginal would asymptotically be the identity matrix of dimension n_X, noted $\mathbf{1}_{n_X}$.

The next step is then to correlate the variable so that $\mathbf{T}$ is not the identity anymore but the target correlation matrix $\mathbf{C}^*$. Knowing that when one multiplies a matrix of random samples by a matrix $\mathbf{W}$ to get $\mathbf{Y} = \mathbf{WX}$, the resulting variance is estimated as [PP12]

Equation 3.2: Simple correlation / de-correlation principle

$$\begin{aligned} \mathbf{C}^* = \mathrm{Var}[\mathbf{Y}] &= \mathbf{W}\mathrm{Var}[\mathbf{X}]\mathbf{W}^T \\ &= \mathbf{W}\mathbf{W}^T, \quad \text{when} \quad \mathrm{Var}[\mathbf{X}] \to \mathbf{1}_{n_X} \end{aligned} \tag{3.2}$$

This leads to the fact that the transformation matrix that provides such a correlation matrix in the end should satisfy the last line of previous equation which is the definition of the Cholesky decomposition of an hermitian positive-definite matrix (W being a lower triangular matrix)

Equation 3.3: General form of a Cholesky decomposition with lower triangular matrix

$$\mathbf{C}^* = \mathbf{W}\mathbf{W}^T \tag{3.3}$$

These steps are the one used to correlate the variables in both in the `TBasicSampling` and `TGaussianSampling` classes. Let's call this method the simple decomposition.

The implementation done in the `TSampling` class, is far more tricky to understand and the aim is not to explain the full concept of the method, called the Iman and Conover method [IC82]. The rest of this paragraph will just provide insight on what's done specifically to deal with the correlation part which is different from what's been explained up to now. The main difference is coming from the underlying hypothesis, written in the second line of Equation 3.2: in a perfect world, for a given random drawing of uncorrelated variables the correlation matrix should satisfy the relation $\mathbf{T} = \mathbf{1}_{n_X}$. This is obviously not the case[1], so one of the proposal to overcome this is to perform a second Cholesky decomposition, on the drawn sample correlation matrix, to get the following decomposition: $\mathbf{T} = \mathbf{K}\mathbf{K}^T$. As $\mathbf{K}$ is lower triangular, it is rather trivial to invert, we can then consider to transform the generated sample using this relation: $\mathbf{Y} = \mathbf{X}(\mathbf{K}^{-1})^T\mathbf{W}^T$. If one consider that these multiplication does not change the fact that columns are centered and reduced, then one can write the following equations

$$\begin{aligned} n_S^{-1}\mathbf{Y}^T\mathbf{Y} &= n_S^{-1}\mathbf{W}\mathbf{K}^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{K}^{-1})^T\mathbf{W}^T \quad \text{but as} \quad n_S^{-1}\mathbf{X}^T\mathbf{X} = \mathbf{T} = \mathbf{K}\mathbf{K}^T \\ &= \mathbf{W}\left(\mathbf{K}^{-1}\mathbf{K}\right)\left(\mathbf{K}^T(\mathbf{K}^{-1})^T\right)\mathbf{W}^T \\ &= \mathbf{W}\mathbf{W}^T \\ &= \mathbf{C}^* \end{aligned}$$

[1] Just considering statistical fluctuation should convince people, see for instance the discussion about Fisher's z-transformation and confidence interval on correlation factors developped later on in *Getting a confidence-interval estimation*

Thanks to this procedure (and many more technicalities such as, for instance, working with Spearman coefficient to be able to handle correlation with stratified samples) the resulting correlation matrix is designed to be as close as possible to the target one.

The final part of this discussion is a limitation of both methods: relying on Cholesky decomposition to decompose the target correlation matrix. If one considers the case where $\mathbf{C}^*$ is a singular matrix, then two important points can be raised:

- this case means that one or more variables can be completely defined thanks to the others. The number of properly defined variable can then be estimated by the rank of the correlation matrix. This situation can occur, as in some complicated problem variables can be highly-intricated leading to this kind of situation.
- with this kind of correlation matrix, the Cholesky decomposition is not doable anymore so both methods are meant to stop brutally.

In order to overcome this situation we propose to use a workaround based on the Singular Value Decomposition (*SVD*) which leads to, knowing that $\mathbf{C}^*$ is real symmetric, $\mathbf{C}^* = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^T$. This writing emphasise the connection between SVD and eigenvalue decomposition (for a more general form and SVD, see for instance Equation 2.1). In this context, $\mathbf{U}(n_X, n_X)$ is an unitary matrices while $\mathbf{\Sigma}(n_X, n_X)$ is a diagonal matrix storing the singular values of $\mathbf{C}^*$ in decreasing order. In our case, where the correlation is singular, it means that one or more of the singular values are very close or equal to 0. By rewriting our decomposition as below

$$\mathbf{C}^* = \mathbf{U}\mathbf{\Sigma}^{1/2}\mathbf{\Sigma}^{1/2}\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}^{1/2}(\mathbf{U}\mathbf{\Sigma}^{1/2})^T$$

one can redefine the matrix $\mathbf{W} = \mathbf{U}\mathbf{\Sigma}^{1/2}$ and get the usual formula discussed above (see Equation 3.3). This decomposition can then be used instead of the Cholesky decomposition in both method (as it is either for the simple form or along with another Cholesky to decompose the correlation matrix of the drawn sample, in our modified Iman and Conover algorithm).

The usage of an SVD instead of a Cholesky decomposition for the target correlation matrix relies on the underlying hypothesis that the *left singular vectors* ($\mathbf{U}$) can be used instead of the *right singular vectors* ($\mathbf{V}$) in the general SVD formula shown for instance in Equation 2.1. This holds even for the singular case, as the only differences seen between both singular vector basis arise for the singular values close to zero. Since in this method we are always using the singular vector matrix weighted by the square roots of the singular values, these differences are vanishing by construction.

3.2.3 The maximin LHS

3.2.3.1 Introduction

Considering the definition of a LHS sampling, introduced in *Introduction*, it is clear that permutating a coordinate of two different points, will create a new sampling. If one looks at the x-coordinate (corresponding to a normal distribution) in Figure 3.2, one could put the point in the second equi-probable range, in the sixth one, and move the point which was in the sixth equi-probable range into the second one, without changing the y-coordinate. The results of this permutation is a new sampling with the interesting property of remaining a LHS sampling. A follow-up question can then be: what is the difference between these two samplings, and would there be any reason to try many permutations ?

This is a very brief introduction to a dedicated field of research: the optimisation of a design-of-experiments with respect to the goals of the ongoing analysis. In Uranie, a new kind of LHS sampling has been recently introduced, called maximin LHS, whose purpose is to maximise the minimal distance between two points. The distance under consideration is the **mindist** criterion: let $D = [\mathbf{x}_1, \cdots, \mathbf{x}_N] \subset [0,1]^d$ be a design-of-experiments with N points. The mindist criterion is written as:

$$\min_{i,j} ||\mathbf{x}_i - \mathbf{x}_j||_2 \tag{3.4}$$

where $||.||_2$ is the euclidian norm. The designs which maximise the mindist criterion are referred to as maximin LHS, but generally speaking, a design with a large value of the mindist criterion is referred to as maximin LHS as well. It

has been observed that the best designs in terms of maximising (Equation 3.4) can be constructed by minimising its L^p regularisation instead. It is written as

$$\phi_p := \left[\sum_{i<j} ||\mathbf{x}_i - \mathbf{x}_j||_2^{-p} \right]^{\frac{1}{p}}$$

Figure 3.4 shows, on the left, an example of a LHS when considering a problem with two uniform distributions between 0 and 1 but also, on the right, its transformation through the maximin optimisation. The mindist criterion is displayed on top for comparison purpose.

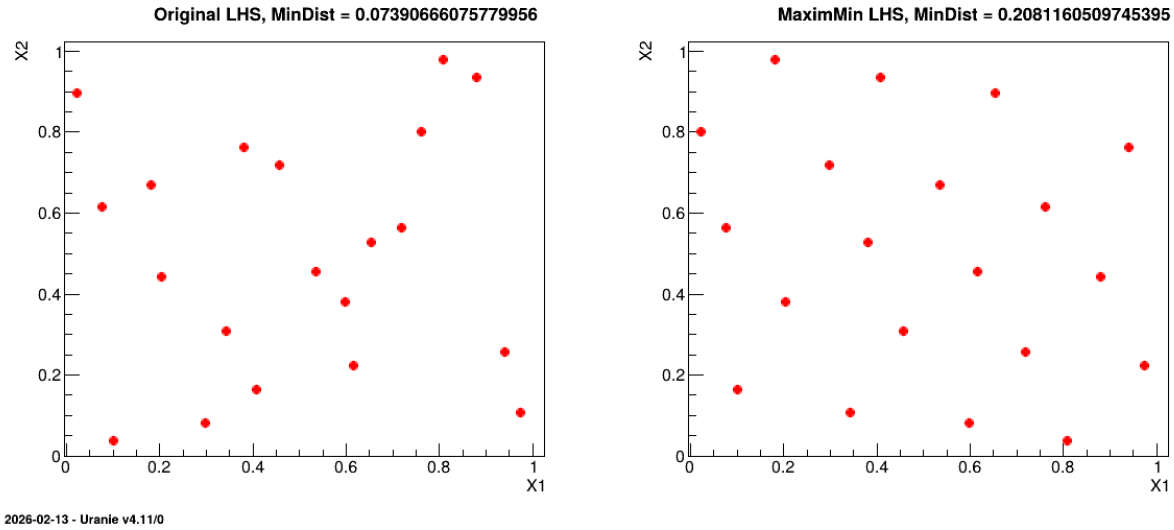

Figure 3.4: Transformation of a classical LHS (left) to its corresponding maximin LHS (right) when considering a problem with two uniform distributions between 0 and 1.

From a theoretical perspective, using a maximin LHS to build a Gaussian process (GP) emulator can reduce the predictive variance when the distribution of the GP is exactly known. However, it is not often the case in real applications where both the variance and the range parameters of the GP are actually estimated from a set of learning simulations run over the maximin LHS. Unfortunately, the locations of maximin LHS are far from each other, which is not a good feature to estimate these parameters with precision. That is why maximin LHS should be used with care. Relevant discussions dealing with this issue can be found in [PM12].

3.2.3.2 The simulated annealing method

The Simulated Annealing (SA) algorithm is a probabilistic metaheuristic which can solve a global optimisation problem. It is here applied to the construction of maximin Latin Hypercube Designs (maximin LHS). The SA algorithm consists in exploring the space of LHS through elementary random perturbations of both rows and columns in order to converge to maximin ones. We have implemented in Uranie the algorithm of Morris and Mitchell [DCI13, MM95], which is driven by the following parameters

- T_0 is the initial temperature
- the decreasing of the temperature is controlled by c
- the number of iterations in the outer loop I
- the number of iterations in the inner loop I_{inner}

It is important to keep in mind that the performances of the simulated annealing method can strongly depend on c and thus changing parametrisation can lead to disappointing results. Below, in Table 3.1, we provide some parametrisation examples working well with respect both to the number of the input variables d and the size of the requested design N.

Table 3.1: Proposed list of parameters value for simulated annealing algorithm, depending on the number of points requested (N) and the number of inputs under consideration (d)

d	$N = 10 \times d$		$N = 20 \times d$		$N = 30 \times d$	
2,3,4	c	0.99	c	0.99	c	0.99
	T_0	0.1	T_0	0.1	T_0	0.1
	I	300	I	300	I	300
	I_{inner}	300	I_{inner}	300	I_{inner}	300
5,6,7	c	0.99	c	0.99	c	0.99
	T_0	0.001	T_0	0.001	T_0	0.001
	I	300	I	300	I	300
	I_{inner}	300	I_{inner}	300	I_{inner}	300
8,9,10	c	0.99	c	0.99	c	0.99
	T_0	0.0001	T_0	0.0001	T_0	0.0001
	I	300	I	300	I	300
	I_{inner}	300-1000	I_{inner}	300-1000	I_{inner}	300-1000

3.2.4 The constrained LHS

3.2.4.1 Introduction

Considering the definition of a LHS sampling, introduced in *Introduction* and also discussed in *Introduction*, it is clear that permutating a coordinate of two different points, will create a new sampling. The idea here is to use this already discussed property to create fulfill requested constraints on the design-of-experiments to be produced. Practically, the constraint will have one main limitation: it should only imply two variables. This limitation is set, so far, for simplicity purpose, as the constraint matrix might blow up and the solutions in term of permutation will also become very complicated (see the heuristic description below in *The heuristic* to get a glimpse at the possible complexity).

Before this algorithm, the solution to be sure to fulfill a constraint was to generate a large sample and apply the constraint as a cut, meaning that no control on the final number of locations in the design-of-experiments and on the marginal distribution shape was possible. From a theoretical perspective, using a constrained LHS is allowing both to have the correct expected marginal distributions and to have precisely the requested number of locations to be submitted to a code or function. This is shown in Figure 3.5, where in a simple case with only three variables uniformly distributed, it is possible to apply three linear constraints: two of them are applied on the (x_0, x_1) plane while the last one is applied on the (x_1, x_2) one.

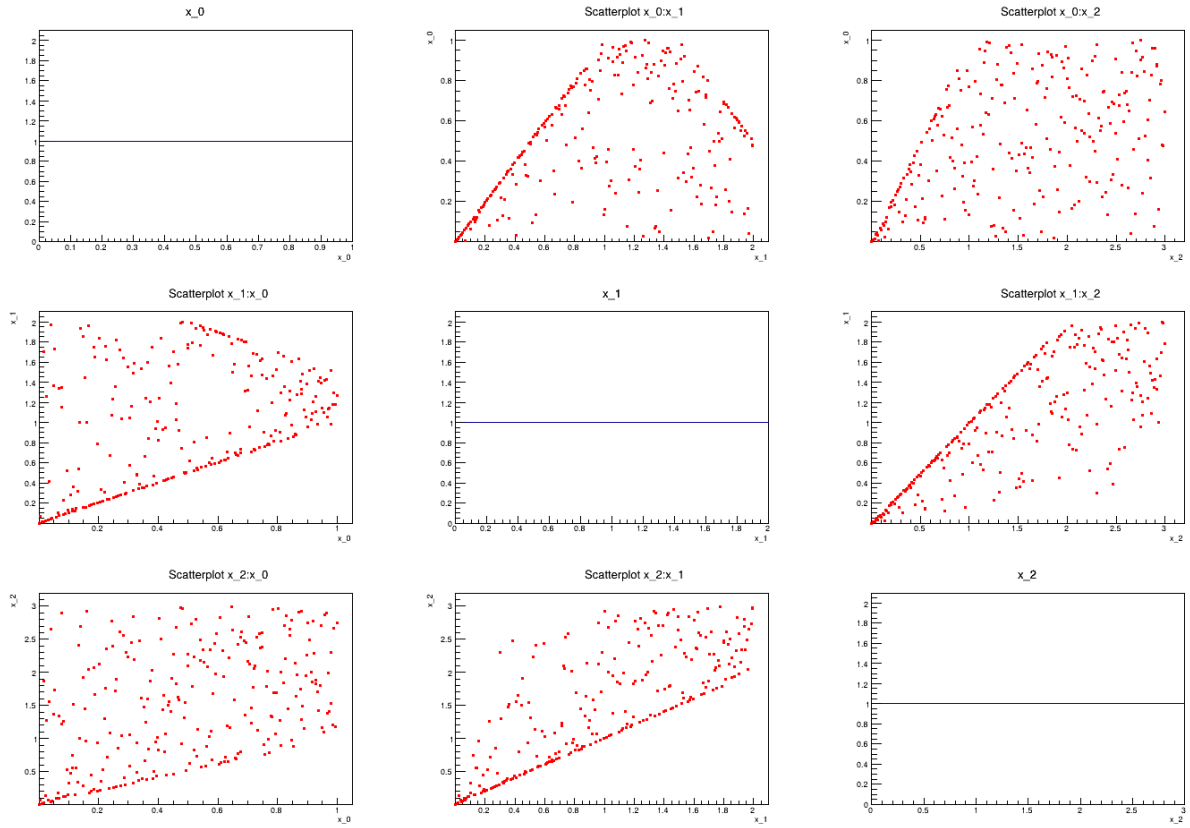

Figure 3.5: Matrix of distribution of three uniformly distributed variables on which three linear constraints are applied. The diagonal are the marginal distributions while the off-diagonal are the two-by-two scatter plots.

3.2.4.2 The heuristic

The idea behing this empirical heuristic is to rely on permutations and to decide on the best permutation to be done thanks to the content of the constraint matrix and also the distributions of solutions along the row and columns. This will be discussed further in the rest of this section but first a focus is done on the definition of a constraint. If one considerers a simple constraint, its implementation can be decompose in the following steps:

1. define it simply with an equation, for instance one wants to reject all locations for which $x_0 < x_1$;
2. define a constraint function that can compute the margin of success, for instance in our simple case $c(x_0, x_1) = x_0 - x_1$;
3. define a characteristic constraint function that only states whether the constraint is fulfilled, for instance in our simple case $\mathbb{1}_c(x_0, x_1) = \begin{cases} 0 & \text{if} x_0 < x_1 \\ 1 & \text{if} x_0 > x_1 \end{cases}$

If formally the constraint function can provide more information when reading it (in terms of margin), one needs to know the way to apply a selection on these results which is not the simplest aspect to provide which explains why in the rest of this section the focus will be put on the characteristic contraint function. In order to illustrate our method, one can start from a provided LHS design-of-experiments, called hereafter $\mathcal{L} = \{\mathbf{x}^i, i = 1, \ldots, n_S\}$ where $\mathbf{x}^i$ is the i-Th location which can be written as $\mathbf{x}^i = (x_1^i, \ldots, x_{n_X}^i)$: it is a sample of size n_S in the n_X input space. From there, the constraint

matrix is indeed defined as done below:

$$\mathbf{C}(n_S, n_S) = \begin{pmatrix} \mathbb{I}_c(x^1_{row}, x^1_{col}) & \mathbb{I}_c(x^1_{row}, x^2_{col}) & \cdots & \mathbb{I}_c(x^1_{row}, x^{n_S}_{col}) \\ \mathbb{I}_c(x^2_{row}, x^1_{col}) & \mathbb{I}_c(x^2_{row}, x^2_{col}) & \cdots & \mathbb{I}_c(x^2_{row}, x^{n_S}_{col}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{I}_c(x^{n_S}_{row}, x^1_{col}) & \mathbb{I}_c(x^{n_S}_{row}, x^2_{col}) & \cdots & \mathbb{I}_c(x^{n_S}_{row}, x^{n_S}_{col}) \end{pmatrix}$$

The constraint matrix is, as visible from the definition above, a (n_S, n_S) matrix which only contains 0 and 1 depending on whether the current configuratio of the (x_{row}, x_{col}) plane fulfill the constraint. This shows why restraining the constraint to only two-variables function is a reasonable approach: for a given configuration the number of permutations will blow up if one will considerer larger dimension constraint. Giving this object, one can introduces more useful objects:

- the constraint row solution vector, that sums up for every row the number of solutions, *i.e.* the number of columns that allow to fulfill the constraint $\{c^i_{row} = \sum_{j=1}^{n_S} \mathbf{C}^{ij}, \forall i \in [1, n_S]\}$;
- the constraint column solution vector, that sums up for every column the number of solutions, *i.e.* the number of row that allow to fulfill the constraint $\{c^i_{col} = \sum_{i=1}^{n_S} \mathbf{C}^{ji}, \forall i \in [1, n_S]\}$;
- the constraint diagonal vector *i.e.* the diagonal of the constraint matrix $\{c^i_{diag}(n_S) = \mathbf{C}^{ii}, \forall i \in [1, n_S]\}$.

Bearing in mind that the objective is to have a constraint fulfilled, the heuristic will use the following criterion as a stopping signal: $c_{stop} = \sum_{i=1}^{n_S} c^i_{diag} = n_S$. This heuristic starts from a provided LHS design-of-experiments for which one can consider three cases:

A
: it cannot fulfill the constraint, meaning that there is no sets of permutation to have $c_{stop} = n_S$;

B
: it can fulfill the constraint with only one set of permutation leading to the only solution $\mathcal{L}^{constr}$;

C
: it can fulfill the constraint with many different sets of permutation leading to a very large number of configurations and so to a very large number of constrained design-of-experiments.

Practically, the heuristic is organised in a step-by-step approach in which the variable used as first argument in the constraint definition will be used as row indicator (it will be called x_{row}) and it is considered fixed. This implies that the permutation will be done by reorganising the second variable, called x_{col}. The heuristic is then described below:

1. The constraint matrix is estimated along with all the objects discussed above $(\mathbf{C}, c_{row}, c_{col}, c_{diag})$.
   - If c_{row} contains one 0, it means that the design-of-experiments cannot fulfill the constaint. If this design-of-experiments, has been provided, the method stops, on the other hand if it was generated on-the-fly, then another attempt is done (up to 5 times).
   - A bipartite graph method is called to check that there can be at least one solution for every row. If not, if the design-of-experiments has been provided, the method stops, on the other hand if it was generated on-the-fly, then another attempt is done (up to 5 times).

   This step allows to sort out the design-of-experiments that will fall into the category A (defined above) from those that might fall either in the B or C ones.
2. If $c_{stop} < n_S$, then all the rows for which the diagonal elements is 0 are kept aside and sorted out by increasing number of solutions over all the columns (their value of c_{row}), which defines $\Omega_{row} = \{i \in [1, n_S], c^i_{diag} = 0\}$. The row considered, whose index will be written k for kept hereafter, is the one with the lowest number of solutions (the most urgent one in a way), so it can be written $k = \min_{c_{row}} \Omega_{row}$
3. For x^k_{row} the chosen value, all the columns that provide a solution are kept aside and sorted out by increasing order[1] of solutions over all rows (their value of c_{col}), which defines $\Omega^k_{col} = \{i \in [1, n_S], \mathbf{C}^{ki} = 1\}$. This sorting provides information on the marging, as the highest values of solutions over the rows means that this value of x_{col} is compatible with many other x_{row} instances. A loop is performed over all these solutions, so that $\forall t \in \Omega^k_{col}$:

[1] the decreasing order has also been tested, but it has show a lower discrepancy in the resulting design-of-experiments.

- By definition we know that $\mathbf{C}^{kk} = 0$ and $\mathbf{C}^{kt} = 1$, so if $\mathbf{C}^{tk} = 1$ then the permutation will only increase the stopping criterion (c_{stop}), as the actual column k, will become the new column t after permutation. In this case, one can move to the permutation step. This can be written, by calling s the actual index of the column (for selected), $s = \min_{c_{col}}\{t \in \Omega^k_{col}, \mathbf{C}^{tk} = 1\}$. From there, one moves to the permutation step.
- If none of the solution t under investigation can satisfy the requirement $\mathbf{C}^{tk} = 1$, then $s = \min_{c_{col}}\{t \in \Omega^k_{col}, t \notin \Omega^k_{col,perm}\}$. In this definition, the ensemble $\Omega^k_{col,perm}$ is the ensemble of column index which has already been used previously (as this is an interative heuristic) in a permutation for the row k under investigation. This precaution has been introduced in order to prevent from having a loop in the permutation process. From there, one moves to the permutation step.
- If $\{t \in \Omega^k_{col}, t \notin \Omega^k_{col,perm}\} = \varnothing$, meaning that all possible solutions have been tested, then the selected column index is the result of a random drawing in the nsemble of solutions, meaning that $s = \text{rand}(\Omega^k_{col})$. From there, one moves to the permutation step.

4. Now that both k and s are known, the permutation will be done, it consists in:
   - changing content of the design-of-experiments, meaning doing $x^k_{col} \leftrightarrow x^s_{col}$;
   - changing the columns in the constraint matrix, meaning doing $\mathbf{C}^{ik} \leftrightarrow \mathbf{C}^{is}, \forall i \in [1, n_S]$;
   - changing the content of the constraint column solution vector, meaning doing $c^k_{col} \leftrightarrow c^s_{col}$;
   - fill once more the constraint diagonal vector (c_{diag}) and compute once more the stopping criterion (c_{stop}) to check whether the permutation process has to be continued. If so, then one moves to the second step.

This presentation has been simplified as, here, there is only one constraint applied to the design-of-experiments. Indeed, when there are more than one constraint, let's call $(\mathbf{C}, c_{row}, c_{col}, c_{diag})$ and $(\mathbf{D}, d_{row}, d_{col}, d_{diag})$ the objects associated to two constraints defined in both planes (x^c_{row}, x^c_{col}) and (x^d_{row}, x^d_{col}), few cautions and extra steps have to be followed as long as both stopping criteria, c_{stop} or d_{stop} are different from n_S depending on the variables in the constraint plane definition:

- if planes (x^c_{row}, x^c_{col}) and (x^d_{row}, x^d_{col}) have no common variable, or if $x^c_{row} = x^d_{row}$ but $x^c_{col} \neq x^d_{col}$ then the constraints can be considered orthogonal;
- if x^c_{col} is the same variable as x^d_{row} then any permutations for the constraint c have to be propagated to the objects of the constraint d, meaning that on top of the list in step 4, the extra steps consist in:
  - changing the rows in the constraint matrix for the constraint d, meaning doing $\mathbf{D}^{ki} \leftrightarrow \mathbf{D}^{si}, \forall i \in [1, n_S]$;
  - changing the content of the constraint row solution vector for constraint d, meaning doing $d^k_{row} \leftrightarrow d^s_{row}$;
  - fill also the constraint diagonal vector (d_{diag}) and compute once more the stopping criterion (d_{stop}) to check whether the permutation process has to be continued.
- if $x^c_{col} = x^d_{col}$, then any permutations from on the constraint can undo a previous one defined from the other one, meaning that one can create a loop that will (thanks to our heuristic definition) will end up into the random permutation area. To prevent this, so far, one can not create a second constraint using the same variable as second argument.

3.3 QMC method

The deterministic samplings can produce design-of-experiments with well defined properties, that can be very useful in specific cases such as:

- to cover at best the space of the input variables
- to explore the extreme cases
- to study combined or non-linearity effect

There are two kinds of quasi Monte-Carlo sampling methods implemented in Uranie: the regular ones and the sparse grid ones. On the first hand, the former can be generated using two different sequences:

1. Sequences of **Sobol** [Sobol67]
2. Sequences of **Halton** [Hal64]

Figure 3.6 shows a comparison of the design-of-experiments obtained with both sequences, along with the ones produced with a basic stochastic sampling, following the LHS and SRS "recipes", all when dealing with two uniform variables. The coverage is clearly more regular in the case of quasi Monte-Carlo sequences which is the origin of their name: low-discrepancy sequences. There are plenty definitions for the notion of discrepancy (see litterature for them) but they all quantify how close the sequence is to a perfect equidistribution of points.

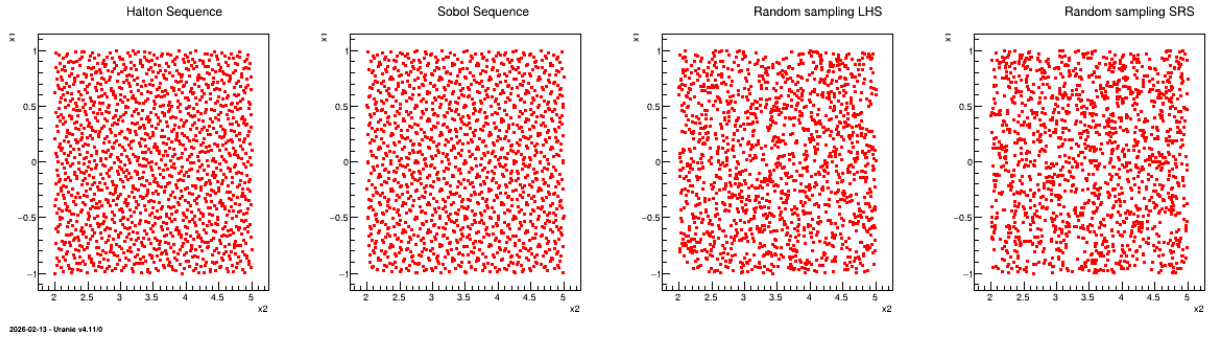

Figure 3.6: Comparison of both quasi Monte-Carlo sequences with both LHS and SRS sampling when dealing with two uniform variables.

On the other hand, the sparse grid sampling can be very useful for integration purposes and can be used in some of the meta-modelling definition, see, for instance, in *The integration method*. In Uranie we can used the Petras algorithm [Pet01] to produce these sparse grids, shown for different levels in Figure 3.7, that can be compared to regular algorithms ones in Figure 3.6 (in both cases, the problem is described with two uniform variables).

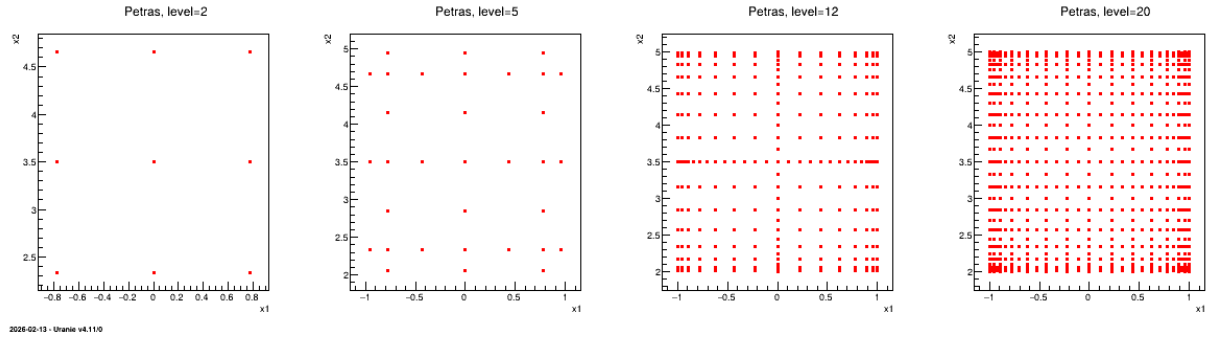

Figure 3.7: Comparison of design-of-experiments made with Petras algorithm, using different level values, when dealing with two uniform variables.

CHAPTER

FOUR

GENERATING SURROGATE MODELS

This is a description of the main meta-modelling methods implemented in the Uranie platform.

4.1 Introduction

This is a test: This part

This part discusses the generation of surrogate models which aim to provide a simpler, and hence faster, model in order to emulate the specified output of a more complex model (and generally time and memory consuming) as a function of its inputs and parameters. The input dataset can either be an existing set of elements (provided by someone else, resulting from simulations or experiments) or it can be a design of experiments generated on purpose, for the sake of the ongoing study.

This ensemble (of size n_S) can be written as

$$\mathcal{L} = \{(\mathbf{x}^i, y^i), i = 1, \ldots, n_S\}$$

where $\mathbf{x}^i$ is the i-th input vector which can be written as $\mathbf{x}^i = (x_1^i \ \ldots \ x_{n_X}^i)$ and the output $y^i = y(\mathbf{x}^i)$.

There are several predefined surrogate-models proposed in the Uranie platform:

- The linear regression, discussed in *The Linear Regression*.

4.1.1 Adapting the fitting strategy

4.2 The Linear Regression

When using the linear regression, one assumes that there is only one output variable and at least one input variable.

4.3 Chaos polynomial expansion

4.3.1 Nisp in a nutshell

4.3.1.1 The integration method

4.4 The kriging method

CHAPTER

FIVE

SENSITIVITY ANALYSIS

5.1 Brief reminder of theoretical aspects

5.1.1 Theoretical aspects

5.1.1.1 The monotone case

5.1.2 No hypothesis on the model

5.2 The regression method

5.2.1 Getting a confidence-interval estimation

CHAPTER

SIX

DEALING WITH OPTIMISATION ISSUES

6.1 Introduction

6.1.1 The pareto concept in a nutshell

CHAPTER

SEVEN

THE CALIBRATION MODULE

7.1 Brief reminder of theoretical aspects

This section presents different calibration methods that are provided to help achieve an accurate estimation of the parameters of a model with respect to data (either from experiment or from simulation). The methods implemented in Uranie are ranging from point estimation to more advanced Bayesian techniques and they mainly differ in the hypotheses they rely on.

In general, a calibration procedure requires an input dataset meaning an existing set of elements (either resulting from simulations or experiments). This ensemble (of size n) can be written as

$$\mathcal{D} = \{(\mathbf{x}^i, \mathbf{y}^i), i = 1, \ldots, n\}$$

where $\mathbf{x}^i$ is the i-th input vector which can be written as $\mathbf{x}^i = (x_1^i \ \ldots \ x_{n_X}^i)$ while $\mathbf{y}^i$ is the i-th output vector which can be written as $\mathbf{y}^i = (y_1^i \ \ldots \ y_{n_Y}^i)$.

These data will be compared with model predictions, where the model is a mathematical function $\mathbf{f}_\theta : \mathbb{R}^{n_X} \rightarrow \mathbb{R}^{n_Y}$. From now on and unless otherwise specified the dimension of the output is set to 1 ($n_Y = 1$) which means that the reference observations and the predictions of the model are scalars (the observations will then be written y and the predictions of the model $f_\theta(\mathbf{x})$).

In addition to the already introduced previously input vector, the model also depends on a parameter vector $\theta \in \Theta \subset \mathbb{R}^p$ which is constant but unknown. The model is deterministic, meaning that $f_\theta(\mathbf{x})$ is constant once both $\mathbf{x}$ and θ are fixed. In the rest of this documentation, a given set of parameter values θ is called a **configuration**.

The standard hypothesis for probabilistic calibration is that the observations differ from the predictions of the model by a certain amount which is supposed to be a random variable as

$$\varepsilon = y - f_\theta(\mathbf{x}) \tag{7.1}$$

where ε is a random variable whose expectation is equal to 0 and which is called *residuals*. This variable represents the deviation between the model prediction and the observation under investigation. It might arise from two possible origins which are not mutually exclusive:

- experimental: affecting the observations. For a given observation, it could be written $\varepsilon_{\text{obs}} = y_{\text{real}} - y$
- modelling: the chosen model f_θ is intrinsically not correct. This contribution could be written $\varepsilon_{\text{model}} = f_\theta^* - f_\theta$

As the ultimate goal is to have $y_{\text{real}} - f_\theta^* = 0$, injecting back the two contributions discussed above, this translates back to Equation 7.1, only breaking down:

$$y - f_\theta = \varepsilon_{\text{obs}} + \varepsilon_{\text{model}}$$

The rest of this section introduces two important discussions that will be referenced throughout this module:

- the distance between observations and the predictions of the models, in *Distances and likelihoods used to compare observations and model predictions*;

- the theoretical background and hypotheses (linear assumption, concept of prior and posterior distributions, the Bayes formulation…) in *Discussing assumptions and theoretical background*.

The former is simply the way to obtain statistics over the n samples of the reference observations when comparing them to a set of parameters and how these statistics are computed when the $n_Y \neq 1$. The latter is a general introduction, partly reminding elements already introduced in other sections and discussing some assumptions and theoretical foundations needed to understand the methods discussed later on.

On top of this description, there are several predefined calibration procedures proposed in the Uranie platform:

- *Using minimisation techniques*
- *Analytical linear Bayesian estimation*
- *Approximate Bayesian Computation techniques (ABC)*
- *Markov chain Monte Carlo approach*
- *CIRCE method*

7.1.1 Distances and likelihoods used to compare observations and model predictions

There are many ways to quantify the agreement between the reference observations and the model predictions, given a parameter vector θ. Depending on the framework adopted (deterministic or Bayesian), different tools are required. In a deterministic setting, a distance is used to measure how far the prediction is from the observation. In contrast, within the Bayesian framework, the analogous concept is the likelihood, a function that evaluates the probability of observing the data given a parameter vector.

Since the number of variables n_Y used to perform the calibration can be greater than one, it might be useful to introduce the coefficients $\{\omega_j\}_{j\in[1,n_Y]}$ to weight the contribution of each variable relative to the others. The following lists the distance functions available in Uranie:

- L1 distance function (sometimes called Manhattan distance): $d(\mathbf{y}, \mathbf{f}_\theta(\mathbf{x})) = \sum_{j=1}^{n_Y} \omega_j \times \left(\sum_{i=1}^{n} |\mathbf{y}_i^j - \mathbf{f}_\theta(\mathbf{x})_i^j| \right)$;
- Least squares distance function: $d(\mathbf{y}, \mathbf{f}_\theta(\mathbf{x})) = \sum_{j=1}^{n_Y} \sqrt{\omega_j \sum_{i=1}^{n} (\mathbf{y}_i^j - \mathbf{f}_\theta(\mathbf{x})_i^j)^2}$;
- Relative least squares distance function: $d(\mathbf{y}, \mathbf{f}_\theta(\mathbf{x})) = \sum_{j=1}^{n_Y} \sqrt{\omega_j \sum_{i=1}^{n} \left(\frac{\mathbf{y}_i^j - \mathbf{f}_\theta(\mathbf{x})_i^j}{\mathbf{y}_i^j} \right)^2}$;
- Weighted least squares distance function: $d(\mathbf{y}, \mathbf{f}_\theta(\mathbf{x})) = \sum_{j=1}^{n_Y} \sqrt{\omega_j \sum_{i=1}^{n} \psi_i^j \times (\mathbf{y}_i^j - \mathbf{f}_\theta(\mathbf{x})_i^j)^2}$, where the coefficients $\{\psi_i^j\}_{i\in[1,n]}$ are associated with the j-th variable and are used to weight each observation with respect to the others;
- Mahalanobis distance function: $d(\mathbf{y}, \mathbf{f}_\theta(\mathbf{x})) = \sum_{j=1}^{n_Y} \sqrt{\omega_j (\mathbf{y}^j - \mathbf{f}_\theta(\mathbf{x})^j)^T \Sigma^{-1} (\mathbf{y}^j - \mathbf{f}_\theta(\mathbf{x})^j)}$ where Σ is the covariance matrix of the observations.

Regarding the likelihood functions already implemented, only the Gaussian log-likelihood for independent parameters is available, as it is the most commonly used. Its expression follows:

- Gaussian log-likelihood for independent parameters: $\textit{log-}\mathcal{L}(\theta|\mathbf{x}, \mathbf{y}) = -\frac{1}{2} \sum_{j=1}^{n_Y} \sum_{i=1}^{n} \left(\log\left(2\pi \left(\sigma_i^j\right)^2\right) + \left(\frac{\mathbf{y}_i^j - \mathbf{f}_\theta(\mathbf{x})_i^j}{\sigma_i^j} \right)^2 \right)$ where the coefficients $\{\sigma_i^j\}_{i\in[1,n]}$ are the standard

deviations of each observation associated to the j-th variable.

If needed, it is still possible to define a custom likelihood (or distance).

These definitions are not orthogonal. Indeed, if $\{\psi_i\}_{i\in[1,n]} = \alpha, \alpha \in \mathbb{R}$, then the least squares function is equivalent to the weighted least squares one. This situation is realistic, as it can correspond to the case where the least squares estimation is weighted with an uncertainty affecting the observations, assuming the uncertainty is constant throughout the data (meaning $\alpha = \sigma^{-2}$). This is called the **homoscedasticity** assumption and it is important for the linear case, as discussed later on.

One can also compare the relative and weighted least squares, if $\alpha = \mathbb{R}$ and $\{\psi_i = (\alpha\% \times y_i)^{-1}\}_{i\in[i,n]}$ these two forms become equivalent (the relative least squares is useful when uncertainty on observations is multiplicative). Finally, if one assumes that the covariance matrix of the observations is the identity (meaning $\Sigma = \mathbf{1}$), the Mahalanobis distance is equivalent to the least squares distance.

Warning

It might seem natural to think that the lower the distance is, the closer our parameters are to the real values. Bearing this in mind would mean thinking that "having a null distance" is the ultimate target of calibration, which is actually dangerous. As for the general discussion in *Generating Surrogate Models*, the risk could be to overfit the set of parameters by "learning" just the set of observations at our disposal as the "truth", not considering that the residuals (introduced in Equation 7.1) might be here to introduce observation uncertainties. In this case, knowing the value of the uncertainty on the observations, the ultimate target of the calibration might be to get the best agreement of observations and model predictions within the model uncertainty, which can be translated into a distribution of the reduced-residuals (that would be something like $\{(y^i - f^i_\theta)/\sigma_{\varepsilon_i}\}_{i\in[1,n]}$ in a scalar case) behaving like a standard normal distribution.

7.1.2 Discussing assumptions and theoretical background

7.1.2.1 Calibration in the context of VVUQ principle

VVUQ is a known acronym standing for "Verification, Validation and Uncertainty Quantification". Within this framework, the calibration procedure of a model, sometimes also called "Inverse problem" [Tar05] or "data assimilation" [ABN16] depending on the assumptions and the context, is an important step of uncertainty quantification. This step should not be confused with validation, even if both procedures are based on a comparison between reference data and model predictions, their definitions are given below [TSI+06]

validation:
process of determining the degree to which a model is an accurate representation of the real world for its intended uses.

calibration:
process of improving the agreement between model calculations and a chosen set of benchmarks by adjusting the parameters implemented in the model.

The underlying question of validation is "What is the confidence level that can be granted to the model given the difference seen between the predictions and physical reality ?" while the underlying question of calibration is "Given the chosen model, which parameter value minimises the difference between a set of observations and its predictions, under the chosen statistical assumptions?".

It sometimes happens that a calibration problem allows an infinite number of equivalent solutions [Han96], which is possible for instance when the chosen model f_θ depends explicitly on an operation of two parameters. The simplest example would be to have a model f_θ depending only on two parameters through the difference $\theta_1 - \theta_2$. In this peculiar case, every couple of parameters (θ_1, θ_2) that would lead to the same difference $\theta_1 - \theta_2$ would provide the exact same model prediction, which means that it is impossible to disentangle these solutions. This issue, also known as parameter identifiability, is crucial as one needs to consider how the chosen model is parameterised [WP97].

Defining a calibration analysis consists in several important steps:

- Specify the set of observations that will be used as reference;
- Specify the model that is supposed to adequately represent the real world;
- Define the parameters to be analysed (either by specifying *a priori* distributions or at least by setting a range). This step requires particular caution concerning identifiability.
- Choose the method used to calibrate the parameters.
- Choose the distance function used to quantify the discrepancy between the observations and the model predictions.

7.1.2.2 Interest in the least square measurement

The least squares distance function introduced in *Distances and likelihoods used to compare observations and model predictions* is widely used when considering calibration issues. This is true whether calibration is performed within a statistical framework or not (see the discussion on uncertainty sources in *Brief reminder of theoretical aspects*). The importance of the least squares approach can be understood by adding an additional assumption on the residuals defined previously. If one considers that the residuals are normally distributed, it implies that one can write

$$\varepsilon_i = \mathcal{N}(0, \sigma_{\varepsilon_i}^2) \quad \text{for} \quad i = 1, \dots, n,$$

where σ_{ε_i} can quantify both sources of uncertainty and whose values are supposed known. The formula above can be used to transform Equation 7.1 into (setting $n_Y = 1$ for simplicity):

$$y_i \sim Y_i|\theta := \mathcal{N}(f_\theta(\mathbf{x}_i), \sigma_{\varepsilon_i}^2) \tag{7.2}$$

This particular case is very interesting, as from Equation 7.2 it becomes possible to write down the probability of the observation set $\mathcal{D}$ as the product of all its component probabilities which can be summarised as such:

$$L(\mathbf{y}|\theta) = \prod_{i=1}^{n} \ell(y_i|\theta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_{\varepsilon_i}} e^{-\frac{1}{2}\left(\frac{y_i - f_\theta(\mathbf{x}_i)}{\sigma_{\varepsilon_i}}\right)^2} \tag{7.3}$$

It is logical to consider that, since the dataset $\mathcal{D}$ has been observed, the probability of this collection of observations must be high. The probability defined in Equation 7.3 can then be maximised by varying θ in order to get its most probable values. This is called the Maximum Likelihood Estimation (MLE) and maximising the likelihood is equivalent to minimising the logarithm of the likelihood which can be written as:

$$\log L(\mathbf{y}|\theta) = -\frac{n}{2} \log 2\pi\sigma_{\varepsilon_i}^2 - \frac{1}{2} \sum_{i=1}^{n} \left(\frac{y_i - f_\theta(\mathbf{x}_i)}{\sigma_{\varepsilon_i}}\right)^2 \tag{7.4}$$

The first part of the right-hand side is independent of θ which means that minimising the log-likelihood is basically focusing on the second part of the right-hand side which essentially corresponds to the weighted least squares distance with the weights set to $\{\psi_i = \sigma_{\varepsilon_i}^{-2}\}_{i\in[i,n]}$. Their values depend on the underlying model assumptions, and this discussion is postponed to another section (this is discussed in *Using minimisation techniques*). More details on least squares concepts can be found in many references, such as [Borck96, PCHS13].

7.1.2.3 Introduction to Bayesian approach

The probability of an event occurring can be seen as the limit of its occurrence rate or as the quantification of a personal judgement or opinion regarding its realisation. This is a difference in interpretation that usually distinguishes the frequentist from the Bayesian perspective. For a simple illustration one can flip a coin: the probability of getting heads, denoted $\mathbb{P}[\text{head}]$ is either the average result of a very large number of experiments (this definition is very factual, but its value depends strongly on the number of experiments) or the personal belief that the coin is well-balanced or not (which is basically an *a priori* opinion that might be based on observations, or not).

Let's call (W, Z) a random vector with a joint probability density $f_{(W,Z)}(w, z)$ and marginal densities written as $f_W(w)$ and $f_Z(z)$. From there, the Bayes' rule states that:

$$f_{W|Z}(w|z) = \frac{f_{Z|W}(z|w) \times f_W(w)}{f_Z(z)} \tag{7.5}$$

where $f_{W|Z}(w|z)$ (respectively $f_{Z|W}(z|w)$) is the conditional probability density of W knowing that z has been realised (and vice-versa respectively). These laws are called conditional laws.

Getting back to our formalism introduced previously, using Equation 7.5 implies that the probability density of the random variable θ given our observations, which is called *posterior* distribution, can be expressed as

$$\pi_{post}(\theta|\mathbf{y}) = \frac{L(\mathbf{y}|\theta)\pi_{prior}(\theta)}{\pi(\mathbf{y})} \propto L(\mathbf{y}|\theta)\pi_{prior}(\theta) \tag{7.6}$$

In this equation, $L(\mathbf{y}|\theta)$ represents the conditional probability of the observations knowing the values of θ, $\pi_{prior}(\theta)$ is the *a priori* probability density of θ, often referred to as **prior**, $\pi(\mathbf{y})$ is the marginal likelihood of the observations, which is constant in our scope (as it does not depend on the values of θ but only on its prior, as $\pi(\mathbf{y}) = \int_\Theta L(\mathbf{y}|\theta)\pi_{prior}(\theta)d\theta$, it consists only of a normalizing factor).

The *prior* law is said to be proper when one can integrate it, and *improper* otherwise. It is conventional to simplify the notations, by writing $\pi(\theta|\mathbf{y})$ instead of $\pi_{post}(\theta|\mathbf{y})$ and also $\pi(\theta)$ instead of $\pi_{prior}(\theta)$. The choice of the prior is a crucial step when defining the calibration procedure and it must rely on physical constraints of the problem, expert judgement and any other relevant information. If none of these are available or reliable, it is still possible to use non-informative priors, in which case calibration relies solely on the data. One can find more discussions on non-informative priors here [Bio15, Jef46].

7.2 Using minimisation techniques

The theory behind this method may seem very basic as it consists mainly in a point estimation of a valid configuration that could be performed without accounting for underlying uncertainty. This view is overly simplistic, since numerical optimisation problems are not simple, even for single-objective optimisation due, for instance, to:

- the regularity of the cost function, the presence of possible local minima, and the computational cost involved;
- the dimensionality of the input space and the potentially complex constraints on the inputs.

These challenges can be addressed through an appropriate problem formulation and the use of a suitable optimisation algorithm. However, these choices are not always straightforward, especially when calibration is confronted with identifiability issues. For further methodological aspects on these issues, see *Dealing with optimisation issues*.

7.3 Analytical linear Bayesian estimation

This method consists mainly of the analytical formulation of the posterior distribution under assumptions: the problem can be considered linear and the prior distributions are normally distributed (or non-informative/flat, as noted at the end of this section).

In the specific case of a linear model, one can then write $f_\theta(\mathbf{x}) = h^T(\mathbf{x})\theta$ where $h(\mathbf{x})$ is the regressor vector. This way of writing the model can include an "hidden virtual" $\theta_0 = 1$ whose purpose is to integrate a constant term into the regression (to describe a pedestal). Using the statistical approach introduced in *Brief reminder of theoretical aspects*, one can also define the covariance matrix of the residuals which will be written hereafter as $\Sigma = \text{diag}(\sigma_{\varepsilon_1}, \ldots, \sigma_{\varepsilon_n})$

From there, one can construct the *design matrix* $H = [h(\mathbf{x}_1), \ldots, h(\mathbf{x}_n)]^T \in M_{n,p}(\mathbb{R})$ whose columns define the subspace onto which the model is projected. With a normal prior, which follows the form $\theta \sim \mathcal{N}(m_\theta, \Sigma_\theta)$ the posterior is also expected to be normal, so it can be written $\pi(\theta|\mathbf{y}) \sim \mathcal{N}(m_\theta^{post}, \Sigma_\theta^{post})$ where its parameters are expressed as

$$m_\theta^{post} = \left(\Sigma_\theta^{-1} + H^T\Sigma^{-1}H\right)^{-1}\left(m_\theta^T\Sigma_\theta^{-1} + \mathbf{y}^T\Sigma^{-1}H\right)^T \tag{7.7}$$

and

$$\Sigma_\theta^{post} = \left(\Sigma_\theta^{-1} + H^T\Sigma^{-1}H\right)^{-1} \tag{7.8}$$

It is also possible, as introduced in *Introduction to Bayesian approach*, to use a non-informative **prior** such as Jeffrey's prior: it is an improper flat prior ($\pi(\mathbf{y}) \propto 1$) [Bio15], whose posterior distribution (in the linear case) is also Gaussian. For this prior, the posterior parameters are equivalent to those obtained with a Gaussian prior, given in Equation 7.7 and Equation 7.8 with all references to Σ_θ removed:

$$m_\theta^{post} = \left(H^T\Sigma^{-1}H\right)^{-1}H^T\Sigma^{-1}\mathbf{y} \text{ and } \Sigma_\theta^{post} = \left(H^T\Sigma^{-1}H\right)^{-1} \tag{7.9}$$

This final form corresponds to the expected results obtained when only considering linear regression within the weighted least squares approach [BF10].

7.3.1 Prediction values

Once both the posterior parameter values and covariances are estimated, it is possible to make a prediction for a dataset not used in the estimation. The central value of the prediction is easy to get, as with any other methods presented in this documentation, since one knows the model and can use the newly estimated posterior mean values of the parameters.

The novel aspect is that a variance can also be estimated for the predicted mean using the posterior covariance matrix of the parameters, Σ_θ^{post}, already introduced in Equation 7.8. This variance represents the uncertainty in each new predicted point due to parameter uncertainty, and it is contained in the covariance matrix Σ_θ^{pred} of dimension (q, q), where q is the sample size under consideration. To obtain the estimate, one needs the new design matrix $H_{pred} = [h(\mathbf{x}_1), \ldots, h(\mathbf{x}_q)]^T \in M_{q,p}(\mathbb{R})$ which then leads to

$$\Sigma_\theta^{pred} = \left(H_{pred}\,\Sigma_\theta^{post}\,H_{pred}^T\right) \tag{7.10}$$

7.4 Approximate Bayesian Computation techniques (ABC)

This section covers methods grouped under the acronym **ABC**, which stands for *Approximate Bayesian Computation*. The core idea is to perform Bayesian inference without explicitly evaluating the model likelihood function. For this reason, these methods are also refered to as **likelihood-free** algorithms [Wil13].

As a reminder, the principle of the Bayesian approach is summarized in the equation $\pi_{post}(\theta|\mathbf{y}) = \frac{L(\mathbf{y}|\theta)\pi_{prior}(\theta)}{\pi(\mathbf{y})} \propto L(\mathbf{y}|\theta)\pi_{prior}(\theta)$, where $L(\mathbf{y}|\theta)$ is the conditional probability of the observations given the parameter values θ, $\pi_{prior}(\theta)$ is the *a priori* probability density of θ (the prior), and $\pi(\mathbf{y})$ is the marginal likelihood of the observations, which is constant here. It does not depend on the values of θ but only on its prior, as $\pi(\mathbf{y}) = \int_\Theta L(\mathbf{y}|\theta)\pi_{prior}(\theta)d\theta$ making it a normalizing factor. For more details, see *Introduction to Bayesian approach* .

7.4.1 Rejection ABC algorithm

Rejection ABC is the simplest version of the ABC approach. Its origins date back to the 1980s [Rub84] and it is possible to see a nice introduction of the rejection algorithm as originally applied to a problem with a finite countable set $\mathcal{V}$ of values in [MPRR12]. In this specific case, it only consisted in two random draws: the parameter values according to their prior then the model prediction according to the parameter values just drawn. If the result was an element of the reference dataset, the configuration was kept.

Things become more complicated when considering continuous sample spaces, since there is no such thing as strict equality when considering stochastic behaviour (without even discussing the numerical issues that have to arise at some points). This implies the need for two important concepts

- a distance metric in the output space, denoted $\rho(.,.)$;
- a tolerance parameter, denoted δ, which determines the accuracy of the algorithm.

Unlike the simpler discrete case quickly introduced above where the aim is to have strict equality between the predictions and the reference data, here the accepted configurations would be those fulfilling the following condition

$$\rho(\mathbf{z}, \mathbf{y}) \leq \delta$$

where θ_T is the configuration under study drawn from the prior $\pi(\theta)$ and $\mathbf{z}$ is the model predictions generated from f_{θ_T} once run on the reference dataset. This was firstly used in the late nineties, as can be seen in [PSPLF99].

> **Warning**
>
> One should recall that the uncertainty model is defined on the residuals, as stated in Equation 7.1 and that residuals are usually considered normally distributed, as in Equation 7.2. Disregarding the origin of these residuals, as discussed in *Brief reminder of theoretical aspects*, if the model one is providing is deterministic, the calibration will focus on a single realisation of the observation without uncertainty consideration. In this case, the model prediction must be modified to include a noise representative of the residuals hypotheses [vdVPS18].

This methodology shows that accepted configurations are not directly sampled from the true posterior distribution $\pi(\theta|\mathbf{y})$ but rather come from an approximation of it that can be written $\pi(\theta|\rho(\mathbf{z}, \mathbf{y}) \leq \delta)$. Two interesting asymptotic regimes can be highlighted:

- when $\delta \to 0$: the algorithm is exact and converges to the true posterior $\pi(\theta|\mathbf{y})$;
- when $\delta \to \infty$: the algorithm ignores the reference data and simply returns the original prior $\pi(\theta)$.

There are many different versions of this kind of algorithm, among which one could find an extra step using summary statistics $S(.)$ to project both $\mathbf{z}$ and $\mathbf{y}$ onto a lower dimensional space. In this version, the configurations kept are drawn from $\pi(\theta|\rho(S(\mathbf{z}), S(\mathbf{y})) \leq \delta)$.

Finally, another possible way to select the best representative sub-sample might be by using a percentile of the analysed and computed set of configurations. Although, mainly recommended for high-dimensional cases (i.e., when n becomes large), this solution might work as long as one keeps an eye on the residuals distribution provided by the *a posteriori* estimated parameters. Indeed, if no threshold is chosen but a percentile is used, the requested number of configurations will always be obtained in the end, but the only way to check whether the uncertainty assumptions are valid is to assess how closely the predictions match the full reference dataset.

7.5 Markov chain Monte Carlo approach

In a Bayesian framework, **Markov Chain Monte Carlo** (MCMC) methods are a powerful tool for calibration. They are especially valuable when the statistical model cannot be solved analytically, such as when the prior distribution has a complex structure or the model is nonlinear. Unlike many classical approaches, MCMC does not require the assumption of Gaussian errors: it remains applicable even when the likelihood is non-Gaussian.

Rather than providing a single "best-fit" solution (as in minimisation techniques), MCMC generates a collection of parameter samples that represent the full posterior distribution (similar to ABC methods). However, these methods also come at the cost of potentially high computational demand, since long sampling chains may be required to achieve convergence and reliable estimates. Users should therefore interpret results as distributions and ensure that convergence diagnostics are checked before drawing conclusions (more details are given in the following sections).

7.5.1 Markov chain principle

A usual approach to explain Markov chain theory on a continuous space is to start with a transition kernel $P(x, A)$ where $x \in \mathbb{R}^p$ and $A \in \mathcal{B}$, where $\mathcal{B}$ is the Borel σ-algebra on $\mathbb{R}^p$ [Wak13]. This transition kernel is a conditional distribution function that represents the probability of moving from x to a point in the set A. It is interesting to notice two properties: $P(x, \mathbb{R}^p) = 1$ and $P(x, \{x\})$ is not necessarily zero, meaning that a transition might be possible from x to x. For a single estimation, from a given starting point x_0, this can be summarised as $\mathbb{P}(x_1 \in A|x_0) = P(x_0, A)$. The Markov chain is

defined as a sequence by repeatedly applying this transition kernel a certain number of times, leading to the k-th estimate ($k \geq 1$) $\mathbb{P}(x_k \in A|x_0) = P^k(x_0, A)$ where P^k denotes the k-th iteration of the kernel P [Tie94].

The important property of a Markov chain is the invariant distribution, π^*, which is the only distribution satisfying the following relation

$$\pi^*(dy) = \int_{\mathbb{R}^p} P(x, dy)\pi(x)dx \tag{7.11}$$

where π is the density with respect to the Lebesgue measure of π^* (meaning $\pi^*(dy) = \pi(y)dy$). This invariant distribution is an equilibrium distribution for the chain that is the target of the sequence of transitions, as

$$\lim_{k\to\infty} P^k(x, A) = \pi^*(A)$$

The Markov chain Monte Carlo approach (MCMC) approach works as follows: the invariant distribution is assumed known, since it is the distribution from which we wish to sample, while the transition kernel is unknown and to be determined. This might seem to be "the proverbial needle in a haystack" but the idea is to be able to express the target density using a transition probability kernel $p(x, y)$ (describing the move from x to y) as

$$P(x, dy) = p(x, y)dy + r(x)\delta_x(dy) \tag{7.12}$$

where $\delta_x(dy) = 1$ and 0 otherwise, while $r(x) = 1 - \int_{\mathbb{R}^p} p(x, y)dy$ is the probability that the chain remains at its current location. If the transition part of this function, $p(.,.)$, satisfies the *reversibility condition* (also called *time reversibility*, *detailed balance*, *microscopic reversibility*…)

$$\pi(x)p(x, y) = \pi(y)p(y, x) \tag{7.13}$$

then $\pi(.)$ is the invariant density of $P(x, .)$ [Tie94].

7.5.2 The MCMC algorithms

In the MCMC approach, the *candidate-generating* density is traditionally denoted $q(x, y)$. If this density satisfies the *reversibility condition* in Equation 7.13 for all x and y the search would be over, but this is very unlikely. What's more probable is to find something like $\pi(x)q(x, y) > \pi(y)q(y, x)$ that states that moving from x to y occurs too often (or too rarely).

The proposed way to correct this is to introduce a probability $\alpha(x, y) < 1$ where this $\alpha(x, y)$ is called the *probability of a move* that is injected in the *reversibility condition* to help fulfil it. Without going into too much detail (see [CG95] which nicely discusses this), the probability of move is usually set to

$$\begin{aligned} \alpha(x, y) &= \min\left[\frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}, 1\right], \qquad \text{if } \pi(x)q(x, y) > 0 \\ &= 1, \qquad \text{otherwise} \end{aligned}$$

If the chain is currently at a point $x_k = x$, then it generates a value y accepted as x_{k+1} with the probability $\alpha(x, y)$. If rejected, the chain remains at the current location and another drawing is performed from there.

With this, one can define the off-diagonal density of the MCMC kernel as function $p(x, y) = q(x, y)\alpha(x, y)$ if $x \neq y$ and 0 otherwise and thanks to Equation 7.12, one has the invariant distribution for P [Tie94].

Warning

Two important things to notice here

- the obtained sample is obviously not independent as the (k+1)-th location is taken out from the k-th one. It might thus be useful to thin the obtained sample to ensure that the samples are less correlated (or ideally uncorrelated) with one another thanks to the **lag** hyperparameter.

- the very first drawn locations are usually rejected as part of the **burn-in** (also called **warm-up**) process. As discussed above, the algorithm needs a certain number of iterations to converge through the invariant distribution.

7.5.2.1 Metropolis-Hastings algorithm

The idea here is to choose a family of candidate-generating densities that follow $q(x, y) = q_1(x - y)$ where $q_1(.)$ is a multivariate density [Muller91], a classical choice being q_1 typically chosen as a multivariate normal density. The candidate is indeed drawn as current value plus a noise, hence the name "random walk".

Once the newly selected configuration-candidate is chosen, let's call it θ_T, a comparison is made with the latest retained configuration, denoted θ_k, through the likelihood ratio, which allows to get rid of any constant factors and should look like this once transformed to its log form:

$$\log\left(\frac{L(\mathbf{y}|\theta_T)}{L(\mathbf{y}|\theta_k)}\right) = \frac{1}{2}\sum_{i=1}^{n}\left(\frac{y_i - f_{\theta_k}(\mathbf{x}_i)}{\sigma_{\varepsilon_i}}\right)^2 - \frac{1}{2}\sum_{i=1}^{n}\left(\frac{y_i - f_{\theta_T}(\mathbf{x}_i)}{\sigma_{\varepsilon_i}}\right)^2$$

This result is then compared to the logarithm of a random uniform drawing between 0 and 1 to decide whether one should keep this configuration (as is usually done in Monte Carlo approaches, see [CG95]).

There are a few more properties for this kind of algorithm such as the acceptation ratio, that might be tuned or used as validity check according to the dimension of our parameter space for instance [GRG+96, RGG+97] or the lag definition, sometimes used to thin the resulting sample (whose use is not always recommended as discussed in [LE12]). These subjects being very close to the implementation choices, they are not discussed here.

Convergence is often difficult to diagnose in practice. To increase confidence in the results, it is advisable to initialize several chains in Markov Chain Monte Carlo (MCMC). This approach helps assess convergence and improves the reliability of the results. Since MCMC methods sample from a probability distribution by simulating a chain of dependent draws, a single chain might get stuck in a local mode or fail to explore the distribution fully. Running multiple chains from different starting points allows us to check whether they converge to the same target distribution, thereby increasing confidence that the sampling has stabilized. Additionally, comparing chains provides diagnostics for mixing and convergence, making the inference more robust.

7.5.2.2 Component-wise Metropolis-Hastings

As explained before, the Metropolis–Hastings (MH) algorithm proposes updates for the entire parameter vector at once, drawing candidates from a proposal distribution in the full parameter space. This can be efficient when the proposal is well-tuned and captures correlations between parameters, but in high dimensions it is often difficult to design such a proposal, and acceptance rates may become very low.

In contrast, **Component-wise Metropolis–Hastings** (CMH), also known as the **Metropolis-within-Gibbs** (MwG) method [Tie94], updates one parameter (or a block of parameters) at a time (selected sequentially or randomly) while keeping the others fixed. This usually leads to higher acceptance rates and easier tuning, especially when parameters are weakly correlated. However, CMH may converge more slowly if strong correlations exist between components, since updates proceed dimension by dimension.

In practice, MH is preferred when a good joint proposal distribution is available or when parameters are highly correlated, while CMH is often advantageous in high-dimensional problems where simpler, one-dimensional proposals are easier to construct and tune.

7.5.3 Assessing convergence

Assessing convergence is a crucial step when applying MCMC methods, as a lack of convergence can compromise the validity of the inference. In practice, several complementary techniques are used:

- a burn-in period is typically discarded to remove the influence of the arbitrary starting point before the chain reaches stationarity;

- thinning the chain by keeping only every k-th sample (lag) can reduce autocorrelation between draws;
- visual inspection of trace plots is also a simple but powerful way to verify whether the chains have mixed well and stabilized;
- monitoring the acceptance ratio provides further information about the efficiency of the algorithm: values that are too high may indicate inefficient exploration, while values that are too low suggest poor mixing.

Beyond these practical checks, formal diagnostics are often employed. The Effective Sample Size (ESS) quantifies the number of effectively independent samples, accounting for autocorrelation [Gey92]:

$$ESS = \frac{N}{1 + 2\sum_{k=1}^{\infty} \rho_k}, \tag{7.14}$$

where N is the total number of draws and ρ_k the autocorrelation at lag k. In practice, the sum over k is truncated once the autocorrelation becomes small enough, often smaller than 0.05. A larger ESS indicates more efficient sampling.

The Gelman–Rubin statistic $\widehat{R}$ compares within-chain and between-chain variances across multiple chains [GR92]:

$$\widehat{R} = \sqrt{\frac{\frac{N-1}{N}W + \frac{1}{N}B}{W}}, \tag{7.15}$$

with W the within-chain variance and B the between-chain variance for n_C chains of length N. Values of $\widehat{R}$ close to 1 suggest that the chains have converged to the same target distribution.

Together, these diagnostics and techniques provide complementary evidence that the MCMC algorithm has converged and that its samples can be reliably used for inference.

7.6 CIRCE method

The CIRCE method is a statistical approach proposed as an alternative to expert judgement, designed to determine the uncertainty of parameters in a physical model. Such uncertainties are often difficult to assess because some parameters may not be directly measurable. However, by relying on separate-effect tests (SET) experiments, that are sensitive to the physical model, it becomes possible to infer estimates of these uncertainties.

7.6.1 Main principle of the CIRCE method

As already stated, CIRCE (which stands for "Calcul des Incertitudes Relatives aux Correlations Elementaires") is a statistical method in which the uncertainties are defined through random variables, mean values and standard deviations [DCrecy12, DCrecyB01]. Usually, if one considers P_i the unobserved parameters (i being the number of these parameters, limited here to a certain number q), one can write the following equation

$$P_i = p_i \times P_i^{\text{nom}}$$

Each physical parameter is expressed as a function of a nominal value (P_i^{nom}) and a multiplier coefficient p_i. A relation can be constructed between these multipliers and the parameters considered by CIRCE, as

Equation 7.16: Relation between multipliers and CIRCE parameters.

$$p_i = 1 + \alpha_i \text{ or } p_i = e^{\alpha_i} \tag{7.16}$$

The nominal value of α_i is set to 0 (which implies that the nominal value of the influential physical model is equal to 1). The other inputs needed by the method are the observed data (or responses) hereafter referred to as R_j^{exp} (j being a realisation of the SET experiment), and the corresponding code result R_j^{code}. CIRCE combines the difference between the experimental results and the code predictions ($R_j^{\text{exp}} - R_j^{\text{code}}$) with the derivatives of each code response with respect to each parameter $\frac{\partial R_j^{\text{code}}}{\partial \alpha_i}$. It is also possible to take into account the experimental uncertainties of the response, called

hereafter δR_j^{exp}. This procedure should lead to the estimation, for every α_i parameter, of its mean value b_i (for bias) and its standard deviation, σ_i.

In order to perform this estimation, there are two main hypotheses done by the CIRCE method:

- the linearity between the code response and each parameter α_i. This hypothesis is clearly visible since first-order derivatives are used for the estimation $\frac{\partial R_j^{\text{code}}}{\partial \alpha_i}$. It is further discussed in *The linearity hypothesis*.
- the normality of the α_i parameters. A hypothesis on the PDF of CIRCE parameters is indeed compulsory, leading to the hypothesis of normality or lognormality of the p_i multiplier if the additive or exponential change of variable is used in Equation 7.16. This is further discussed in *The normality hypothesis*.

7.6.1.1 The linearity hypothesis

For every response, the quantity of interest is $R_j^{\text{exp}} - R_j^{\text{code}}$ which can also be written as, if R_j^{real} denotes the real value of the response R_j, $R_j^{\text{exp}} - R_j^{\text{code}} = (R_j^{\text{exp}} - R_j^{\text{real}}) + (R_j^{\text{real}} - R_j^{\text{code}})$. It is the sum of two independent random variables:

- $(R_j^{\text{exp}} - R_j^{\text{real}})$: the experimental uncertainty which follows a centered normal distribution of known standard deviation σ^{exp}.
- $(R_j^{\text{real}} - R_j^{\text{code}})$: which is obtained from a first-order development as

$$R_j^{\text{real}} - R_j^{\text{code}} = \sum_{i=1}^{q} \frac{\partial R_j^{\text{code}}}{\partial \alpha_i} (\alpha_{j,i} - \alpha_i^{\text{nom}})$$

  In this definition, $\alpha_{j,i}$ is the unknown value assigned to the i-th parameter such that $R_j^{\text{code}}(\alpha_{j,1}, \ldots, \alpha_{j,q}) = R_j^{\text{real}}$ ($\alpha_{j,i}$ being different for every response) and α_j^{nom} is the nominal value of this i-th parameter (generally 0).

7.6.1.2 The normality hypothesis

If we collect all the information about the system described so far, the problem can be summarised as

$$R_j^{\text{exp}} - R_j^{\text{code}} = (R_j^{\text{exp}} - R_j^{\text{real}}) + (R_j^{\text{real}} - R_j^{\text{code}}) = e_j + \sum_{i=1}^{q} \frac{\partial R_j^{\text{code}}}{\partial \alpha_i} \times \alpha_{j,i}$$

In this expression, one can discuss the different contributions:

- $R_j^{\text{exp}} - R_j^{\text{code}}$ and $\frac{\partial R_j^{\text{code}}}{\partial \alpha_i}$ are known.
- e_j is a realisation of $\mathcal{N}(0, (\sigma^{\text{exp}})^2)$ where σ^{exp} is also known.
- The $\alpha_{j,i}$ are unknown. The only available information comes from their statistical properties: their bias b_i and their standard deviation σ_i.

Several solutions are possible for the vector α, leading to a needed choice among them. The criterion chosen to do so is the maximum likelihood estimation, which requires a hypothesis on the form of the probability distribution followed by the α_i parameters. The normality assumption is then adopted.

CHAPTER

EIGHT

THE UNCERTAINTY MODELER MODULE

Abstract This chapter presents the features of the UncertModeler module of Uranie - version v4.11.0. The *namespace* of this library is **URANIE::UncertModeler**.

8.1 Tests based on the *Empirical Distribution Function ("EDF tests")*

This part is introducing comparison tests, sometimes called "goodness of fit" tests, which are used as test hypothesis. This idea is to check, when considering a certain variable, whether it is following a predefined law among the list of implemented ones: normal, lognormal and uniform ones. To do so, there are three different tests implemented in Uranie. If one calls $F_n(x)$ the *Empirical Distribution Function* of the law $F(x)$ (*i.e.* the distribution that we'd like to test) and $F_0(x)$ the reference law one wants to compare to, then, for n the number of data in the EDF, these tests are defined as:

Kolmogorov-Smirnov (D) [Kol33]

$$D = \sup|F_0(X_i) - F_n(X_i)|_{i=1,\dots,n}$$

Anderson-Darling (A^2) [AD52]

$$A^2 = n\int \frac{|F_0(x) - F_n(x)|^2}{F_0(x)(1-F_0(x))}dF_0(x) = -n - \frac{1}{n}\sum_{i=1}^{n}(2i-1)\times[\log(F_n(X_i)) + \log(F_n(X_{n+1-i}))]$$

Cramer-VonMises (W^2) [And62]

$$W^2 = n\int |F_0(x) - F_n(x)|^2 dF_0(x) = \frac{1}{12n}\sum_{i=1}^{n}\left(F_0(X_i) - \frac{2i-1}{2n}\right)^2$$

In these three formulas, the $(X_i)_{i=1,\dots,n}$ set represents the ordered data of the random variable x which comes usually as the CDF distribution for convenience.

BIBLIOGRAPHY

[And62] T. W. Anderson. On the distribution of the two-sample cramer-von mises criterion. *Ann. Math. Statist.*, 33(3):1148–1159, 09 1962. URL: http://dx.doi.org/10.1214/aoms/1177704477, doi:10.1214/aoms/1177704477.

[AD52] T. W. Anderson and D. A. Darling. Asymptotic theory of certain goodness of fit criteria based on stochastic processes. *Ann. Math. Statist.*, 23(2):193–212, 06 1952. URL: http://dx.doi.org/10.1214/aoms/1177729437, doi:10.1214/aoms/1177729437.

[App13] W. Appel. *Probabilité pour les non probabilistes*. H & K, Paris, 2013.

[ABN16] Mark Asch, Marc Bocquet, and Maëlle Nodet. *Data assimilation. Methods, algorithms and applications.* SIAM, 2016.

[Borck96] Ake Börck. *Numerical Methods for Least Squares Problems*. Society for Industrial Applied Mathematics, 1996.

[BF10] N. H. Bingham and John M. Fry. *Regression. Linear Models in Statistics*. Springer, 2010.

[Bio15] Christèle Bioche. *Approximation de lois impropres et applications*. PhD thesis, 2015.

[CG95] Siddhartha Chib and Edward Greenberg. Understanding the metropolis-hastings algorithm. *The american statistician*, 49(4):327–335, 1995.

[DCI13] G. Damblin, M. Couplet, and B Iooss. Numerical studies of space filling designs: optimization of Latin hypercube samples and subprojection properties. *Journal of simulation*, 7:276–289, 2013.

[DCrecy12] A. De Crécy. Circe: a methodology to quantify the uncertainty of the physical models of a code. Technical Report, CEA DEN/DANS/DM2S/STMF/LGLS/RT/12-013/A, 2012.

[DCrecyB01] A. De Crécy and P. Bazin. Determination of the uncertainties of the constitutive relationship of the CATHARE 2 code. *M&C 2001*, 2001.

[GRG+96] Andrew Gelman, Gareth O Roberts, Walter R Gilks, and others. Efficient metropolis jumping rules. *Bayesian statistics*, 5(599-608):42, 1996.

[GR92] Andrew Gelman and Donald B Rubin. Inference from iterative simulation using multiple sequences. *Statistical science*, 7(4):457–472, 1992.

[Gey92] Charles J Geyer. Practical markov chain monte carlo. *Statistical science*, pages 473–483, 1992.

[Hal64] J. H. Halton. Algorithm 247: radical-inverse quasi-random point sequence. *Commun. ACM*, 7(12):701–702, December 1964. URL: http://doi.acm.org/10.1145/355588.365104, doi:10.1145/355588.365104.

[Han96] Christian Hansen. *Rank-Deficient and Discrete Ill-Posed Problems*. SIAM, 1996.

[HD02] J. C. Helton and F. J. Davis. Illustration of sampling-based methods for uncertainty and sensitivity analysis. *Risk Analysis*, 22(3):591–622, 2002. URL: http://dx.doi.org/10.1111/0272-4332.00041, doi:10.1111/0272-4332.00041.

[IC82] R. L. Iman and W. J. Conover. A distribution-free approach to inducing rank correlation among input variables. *Communications in Statistics - Simulation and Computation*, 11(3):311–334, 1982. doi:10.1080/03610918208812265.

[Jef46] Harold Jeffreys. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 186(1007):453–461, 1946.

[Jol11] Ian Jolliffe. *Principal component analysis*. Springer, 2011.

[Kol33] A. N. Kolmogorov. Sulla Determinazione Empirica di una Legge di Distribuzione. *Giornale dell'Istituto Italiano degli Attuari*, 4:83–91, 1933.

[LE12] William A Link and Mitchell J Eaton. On thinning of chains in mcmc. *Methods in ecology and evolution*, 3(1):112–115, 2012.

[MPRR12] Jean-Michel Marin, Pierre Pudlo, Christian P Robert, and Robin J Ryder. Approximate bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180, 2012.

[MBC00] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, February 2000. URL: http://dx.doi.org/10.2307/1271432, doi:10.2307/1271432.

[MM95] D. Morris and J. Mitchell. Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43:381–402, 1995.

[Muller91] Peter Müller. *A generic approach to posterior integration and Gibbs sampling*. Purdue University, Department of Statistics, 1991.

[PCHS13] V. Pereyra P. C. Hansen and G. Scherer. *Least Squares Data Fitting with Applications*. Johns Hopkins University Press, 2013.

[PP12] K. B. Petersen and M. S. Pedersen. The matrix cookbook. nov 2012. Version 20121115. URL: http://localhost/pubdb/p.php?3274.

[Pet01] K. Petras. Fast calculation of coefficients in the smolyak algorithm. *Numerical Algorithms*, 26(2):93–109, 2001. URL: http://dx.doi.org/10.1023/A:1016676624575, doi:10.1023/A:1016676624575.

[PSPLF99] Jonathan K Pritchard, Mark T Seielstad, Anna Perez-Lezaun, and Marcus W Feldman. Population growth of human y chromosomes: a study of y chromosome microsatellites. *Molecular biology and evolution*, 16(12):1791–1798, 1999.

[PM12] L. Pronzato and W. Muller. Design of computer experiments: space filling and beyond. *Statistics and Computing*, 22(3):681–701, 2012.

[RGG+97] Gareth O Roberts, Andrew Gelman, Walter R Gilks, and others. Weak convergence and optimal scaling of random walk metropolis algorithms. *The annals of applied probability*, 7(1):110–120, 1997.

[Rub84] Donald B Rubin. Bayesianly justifiable and relevant frequency calculations for the applies statistician. *The Annals of Statistics*, pages 1151–1172, 1984.

[Sobol67] I.M Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86 – 112, 1967. URL: http://www.sciencedirect.com/science/article/pii/0041555367901449, doi:http://dx.doi.org/10.1016/0041-5553(67)90144-9.

[Tar05] Albert Tarantola. *Inverse problem theory*. SIAM, 2005.

[Tie94] Luke Tierney. Markov chains for exploring posterior distributions. *the Annals of Statistics*, pages 1701–1728, 1994.

[TSI+06] Timothy G Trucano, Laura Painton Swiler, Takera Igusa, William L Oberkampf, and Martin Pilch. Calibration, validation, and sensitivity analysis: what's what. *Reliability Engineering & System Safety*, 91(10-11):1331–1357, 2006.

[vdVPS18] Elske van der Vaart, Dennis Prangle, and Richard M Sibly. Taking error into account when fitting models using approximate bayesian computation. *Ecological applications*, 28(2):267–274, 2018.

[Wak13] Jon Wakefield. *Bayesian and frequentist regression methods*. Springer Science & Business Media, 2013.

[WP97] Eric Walter and Luc Pronzato. Identification of parametric models. *Communications and control engineering*, 1997.

[Wei] Eric W. Weisstein. Likelihood. https://mathworld.wolfram.com/Likelihood.html. URL: https://mathworld.wolfram.com/Likelihood.html.

[Wil13] Richard David Wilkinson. Approximate bayesian computation (abc) gives exact results under the assumption of model error. *Statistical applications in genetics and molecular biology*, 12(2):129–141, 2013.