ETH zürich

UQLab User Manual – Sensitivity Analysis

S. Marelli, C. Lamas, B. Sudret

Chair of Risk, Safety and Uncertainty Quantification
Stefano-Franscini-Platz 5
CH-8093 Zürich

Risk, Safety & Uncertainty Quantification

How to cite this manual

S. Marelli, C. Lamas and B. Sudret, UQLab user manual – Sensitivity analysis, Report # UQLab-V0.9-106, Chair of Risk, Safety & Uncertainty Quantification, ETH Zurich, 2015.

BibTeX entry

```
@TECHREPORT{UQdoc_09_106,
author = {Marelli, S. and Lamas, C. and Sudret, B.},
title = {UQLab user manual – Sensitivity analysis},
institution = {Chair of Risk, Safety \& Uncertainty Quantification, ETH Zurich},
year = {2015},
note = {Report \# UQLab-V0.9-106}
}
```

Document Data Sheet

Document Ref.	UQdoc_09_106
Title:	UQLAB User Manual – Sensitivity Analysis–
Authors:	S. Marelli, C. Lamas, B. Sudret
	Chair of Risk, Safety and Uncertainty Quantification, ETH Zurich, Switzerland
Date:	01/07/2015

Doc. Version	Date	Comments
UQdoc_09_106	01/07/2015	Initial release

Abstract

The UQLAB Sensitivity Analysis module is a powerful and flexible tool for performing sensitivity analysis with a number of different techniques. This user manual includes a review of several advanced sensitivity analysis methods and algorithms available in UQLAB, also serving as an overview of the relevant up-to-date literature. It also provides an in-depth, example-driven user guide to help new users to properly set up and solve sensitivity analysis problems with the techniques described. Finally, a comprehensive reference list of the methods and functions available in the UQLAB sensitivity module is given at the end of the manual.

Keywords: Global sensitivity analysis - Sobol' indices - Morris method - cotter measure - SRC/SRRC coefficients - Polynomial Chaos Expansions

Contents

Chapter 1

Theory

1.1 Introduction

Let $\mathcal{M}(\boldsymbol{x})$ be a mathematical model, depending on a set of input variables $(x_1, ..., x_M)$ described by a random vector $\boldsymbol{X}$. In general, the aim of sensitivity analysis is to describe how the variability of the model response, $y = \mathcal{M}(\boldsymbol{x})$ is affected by the variability of each input variable.
Apart from the interesting information that such an analysis provides to the modeller in a theoretical sense, sensitivity analysis is useful to spot unimportant factors and help reduce the dimension of the problem (*model reduction*), among other usages. This kind of analyses are performed with a *black-box* approach, i.e. only based on the model response evaluations for a certain sample of inputs, selected in such a way that they will maximize the output information about the model structure. Furthermore, it is often the case that each run of the model is expensive in terms of computer time and, therefore, sensitivity methods generally aim at reducing the number of model evaluations as much as possible.

1.2 Problem statement

Numerous approaches are available nowadays to perform sensitivity analysis. Broadly speaking, we can either look at the correlation of the input parameters with the output (correlation-based measures), at the values of partial derivatives of the model or at a given point (local methods) or directly at the distribution of the model output. More specifically, global sensitivity analysis (a.k.a. variance decomposition techniques) aims at decomposing the variance of the model output in terms of contributions of each single input parameter, or combination thereof.
Here, the following approaches will be discussed:

- **Local methods** - They are based on the post-processing of available Monte Carlo samples of the model. They are rather simple as they do not require special sampling strategies. The following methods are available in UQLAB:
  - Input/output correlation

  - Standard regression coefficients

- **Linearisation methods** - They are based on the assumption that the model is linear or can be linearised around a central value, usually referred to as the *nominal value* in the literature. They are simple and usually require few model evaluations. However, they fail to identify non-linear behaviours and higher order interactions between factors. Within this category, we will review the following methods:

  - Perturbation method
  - Cotter method

- **Global methods** - These methods take into account the whole input domain. They may refer to different features of the model output, such as variance or distribution. In the current version of UQLAB the following methods are implemented:

  - Morris' elementary effects
  - Sobol' sensitivity indices

Please note that other authors use a different classification of the aforementioned methods, by identifying the class of so-called *screening methods*. Screening methods aim at providing a qualitative ranking of the importance of the input variables w.r.t. the model response, with relatively low computational costs. Cotter and Morris' methods are commonly grouped in this class. In this context, we could include polynomial chaos expansion-based Sobol' indices (Section 1.5.3) in the class of screening methods.

1.3 Local methods

1.3.1 Input/Output Correlation

Perhaps the most intuitive measure of sensitivity of model response Y to the components of the input random vector $\boldsymbol{X}$ is the component-wise correlation coefficients between the two. Consider a sampling of the input random vector $\mathcal{X} = \left\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, ..., \boldsymbol{x}^{(N)}\right\}$ and the corresponding model responses $\mathcal{Y} = \left\{y^{(1)}, y^{(2)}, ..., y^{(N)}\right\}$. The linear correlation coefficient ρ_i between the i^{th} input and the output is defined as

$$\rho_i \overset{\text{def}}{=} \rho(X_i, Y) = \frac{\mathbb{E}\left[(X_i - \mu_i)(Y - \mu_Y)\right]}{\sigma_i \, \sigma_Y}, \tag{1.1}$$

where $\mu_i \overset{\text{def}}{=} \mathbb{E}\left[X_i\right]$ and $\mu_Y \overset{\text{def}}{=} \mathbb{E}\left[Y\right]$ and σ_i and σ_Y are the corresponding standard deviations. The unbiased Monte Carlo estimator of the sample correlation reads:

$$\widehat{\rho}_i = \frac{\sum_{j=1}^{N}(x_i^{(j)} - \widehat{\mu}_i)(Y^{(j)} - \widehat{\mu}_Y)}{\widehat{\sigma}_i \, \widehat{\sigma}_Y}, \tag{1.2}$$

where the hat notation identifies the sample estimates of the corresponding quantities in Eq. (1.1).

Linear correlation, however, is known to be inaccurate in the presence of strongly non-linear dependence between variables. A more stable estimator that relies on the monotonicity instead of the linearity of the dependence of two variables, is the Spearman's rank correlation index ρ_S.

To calculate Spearman's ρ_S, each component of the input sample and its response are transformed into their rank-equivalents:

$$R_i = \left\{ r_i^{(j)} \in \{1, ..., N\} : r_i^{(j)} > r_i^{(k)} \iff x_i^{(j)} > x_i^{(k)} \; \forall i, j \in \{1, ..., N\} \right\} \tag{1.3}$$

The rank-transformed model response then reads $R_Y = \left\{ r_Y^{(1)}, r_Y^{(2)}, ..., r_Y^{(N)} \right\}$. The Spearman correlation coefficient is defined as the linear correlation coefficients between the ranks:

$$\rho_i^S \stackrel{\text{def}}{=} \rho^S (X_i, Y) = \rho(R_i, R_Y). \tag{1.4}$$

1.3.2 Standard Regression Coefficients

Another simple measure of sensitivity of the model response to each of the input variables is given by the Standard Regression Coefficients (SRC). The model $\mathcal{M}(X)$ is approximated by a linear regression:

$$Y \approx \beta_0 + \sum_{i=1}^{M} \beta_i X_i \tag{1.5}$$

The advantage of this formulation is that, in case of independent variables X_i, the total variance of the model can be estimated by:

$$\widehat{\sigma}_Y^2 = \sum_{i=1}^{M} \beta_i^2 \sigma_{X_i}^2 \tag{1.6}$$

where $\widehat{\sigma}_Y^2$ and $\sigma_{X_i}^2$ are the variances of the response and each input, respectively.

Given a sample of the input $\mathcal{X} = \left\{ \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, ..., \boldsymbol{x}^{(N)} \right\}$ and the corresponding sample of model responses $\mathcal{Y} = \left\{ y^{(1)}, y^{(2)}, ..., y^{(N)} \right\}$, the least-square estimator of the $\boldsymbol{\beta} = \{\beta_1, \beta_2, ..., \beta_M\}$ is given by:

$$\widehat{\boldsymbol{\beta}} = \left(\boldsymbol{A}^{\boldsymbol{T}} \boldsymbol{A} \right)^{-1} \boldsymbol{A}^{\boldsymbol{T}} \mathcal{Y} \tag{1.7}$$

where the matrix $\boldsymbol{A}$ has for columns the M components of the input sampling $\boldsymbol{\mathcal{X}}$.

The SRC indices are then defined as:

$$SRC_i = \frac{\widehat{\beta}_i \sigma_i}{\sigma_Y} \tag{1.8}$$

A straightforward extension to this method consists in substituting both the experimental design samples and their response with the corresponding ranks so as to obtain the standard rank regression coefficients (SRRC). Applying a rank-transform to both Y and $\boldsymbol{X}$ in Eq. (1.5)

yields:

$$r_Y \approx \gamma_0 + \sum_{i=1}^{M} \gamma_i r_i \tag{1.9}$$

where the r_Y are the ranks of the model responses, r_i the ranks of the input coordinates and γ_i the rank-regression coefficients. Eq. (1.7) then becomes:

$$\widehat{\gamma} = \left(\boldsymbol{R^T R}\right)^{-1} \boldsymbol{R^T} R_Y \tag{1.10}$$

where the matrix $\boldsymbol{R}$ has for columns the M components of the rank-transformed experimental design R_i (Eq. (1.4)). The corresponding indices read:

$$SRRC_i = \widehat{\gamma}_i. \tag{1.11}$$

Note that the standard deviations in Eq. (1.8) cancel out after the rank transform because each rank-transformed variable is a permutation of the vector of integer ranks $\{1, 2, \ldots, N\}$.

1.4 Local methods

1.4.1 Perturbation method

The basis of the perturbation method is to substitute the model by its Taylor expansion around a value of interest, usually the mean value of the input distribution. Once the first order expansion is found, it can be used to analytically find the mean and variance of the model, and to develop an importance measure of each of the factors, by looking at their contribution to the model variance in terms of their own initial variance.
Let $\mu_{\boldsymbol{X}} = \{\mu_1, ..., \mu_M\}^\mathsf{T}$ denote the vector containing the means of the marginal distributions of the input factors and $\boldsymbol{C}$ be the covariance matrix. The first order expansion of the model $\mathcal{M}(\boldsymbol{x})$ around the mean value is as follows:

$$\mathcal{M}(\boldsymbol{x}) = \mathcal{M}(\mu_{\boldsymbol{X}}) + \sum_{i=1}^{M} \frac{\partial M}{\partial x_i}|_{x=\mu_{\boldsymbol{X}}} (x_i - \mu_i) + o(\|x - \mu_{\boldsymbol{X}}\|^2)$$

The mean and variance can be estimated from this expression as:

$$\widehat{\mathbb{E}}[\mathcal{M}(\boldsymbol{X})] = \mathcal{M}(\mu_{\boldsymbol{X}}) \tag{1.12}$$

$$\widehat{\mathrm{Var}}\left[\mathcal{M}(\boldsymbol{X})\right] = \nabla\mathcal{M}^T(\mu_{\boldsymbol{X}}) \cdot \boldsymbol{C} \cdot \nabla\mathcal{M}^T(\mu_{\boldsymbol{X}}) \tag{1.13}$$

If the variables are independent, $\boldsymbol{C}$ contains only the variances of the input factors on the diagonal, so the estimate of the total variance in (1.13) simplifies to:

$$\widehat{\mathrm{Var}}\left[\mathcal{M}(\boldsymbol{X})\right] = \sum_{i=1}^{M} \left(\frac{\partial M}{\partial x_i}(\mu_{\boldsymbol{X}})\right)^2 \sigma_i^2 \tag{1.14}$$

As this is a sum of positive terms over the number of variables, each of the terms represents its contribution to the estimate of the total variance. Finally, to construct the perturbation method sensitivity estimates η_i, these terms are divided by the total variance, so that they sum up to one:

$$\eta_i = \frac{\left(\frac{\partial M}{\partial x_i}(\mu_{\boldsymbol{X}})\right)^2 \sigma_i^2}{\widehat{\mathrm{Var}}\left[\mathcal{M}(\boldsymbol{X})\right]}, \qquad i = 1, ..., M. \tag{1.15}$$

1.4.2 Cotter method

The Cotter method (Cotter, 1979) is a simple and low-cost method that allows for ranking input parameters. Let us consider an input vector $\boldsymbol{x} = \{x_1, ..., x_M\}^T \in \mathcal{D}_x$. Each factor X_i is assumed to vary in a known interval described by its *low* and *high* levels $X_i \in [x_i^-, x_i^+]$. Then, a systematic fractional replicate design is carried out, in which the following $2M + 2$ model evaluations are performed:

- One run with all factors at their low levels, say $y_0 = \mathcal{M}(x_1^-, ..., x_M^-)$
- M runs at low levels, switching one factor at a time to its high level: $y_i = \mathcal{M}(x_1^-, ..., x_i^+, ..., x_M^-),\ i = 1, ..., M$
- M runs at high levels, switching one factor at a time to its low level: $y_i = \mathcal{M}(x_1^+, ..., x_i^-, ..., x_M^+),\ i = M + 1, ..., 2M$
- One run with all factors at their high levels, $y_{2M+1} = \mathcal{M}(x_1^+, ..., x_M^+)$

The importance of the factors is then measured by:

$$I_{Cotter}(i) = |C_o(i)| + |C_e(i)|\,, \tag{1.16}$$

where $C_o(i)$ and $C_e(i)$ denote the expectation of the importance of the odd and even order effects involving the factor X_i respectively, and are given by:

$$C_o(i) = \frac{1}{4}\left[(y_{2M+1} - y_{M+i}) + (y_i - y_0)\right], \tag{1.17}$$

$$C_e(i) = \frac{1}{4}\left[(y_{2M+1} - y_{M+i}) - (y_i - y_0)\right]. \tag{1.18}$$

The great advantage of this method is that it only requires $2M + 2$ runs of the model, yet it has the drawback that, if the odd or even orders cancel each other out among dimensions, this can lead to an incorrect importance measure.

1.5 Global methods

1.5.1 Morris method

The Morris method (Morris, 1991) is somehow similar to the Cotter method, but tries to cover more exhaustively the input space. Instead of looking at a single global perturbation,

it creates a grid where multiple different perturbations are possible. The sensitivity measure is then built based on the mean and standard deviation of that set of perturbations.
With no loss of generality, let us assume that the input parameters $\boldsymbol{X}$ vary in an M-dimensional unit hypercube, i.e. $X_i \in [0,1],\ i = 1,...,M$ (in the practical case, these sets can be rescaled to the real input space).
A grid of p values is created for each parameter. It consists of the set:

$$\mathcal{G}_i = \left\{0, \frac{1}{p-1}, \frac{2}{p-1}, ..., 1\right\}. \tag{1.19}$$

The whole input set is then defined by:

$$\boldsymbol{\mathcal{G}} = \mathcal{G}_1 \times ... \times \mathcal{G}_M \tag{1.20}$$

Let us now define a perturbation parameter Δ such that it is a multiple of $\frac{1}{p-1}$,

$$\Delta = \frac{\bar{c}}{p-1},\ \text{for a fixed } \bar{c} \in \{1,...,p-1\}. \tag{1.21}$$

For a given value of $\boldsymbol{x} = \{x_1,...,x_M\}^\mathsf{T} \in \boldsymbol{\mathcal{G}}$, such that $x_i + \Delta \leq 1$, the *elementary effect* for the variable x_i is defined by:

$$d_i(\boldsymbol{x}) = \frac{\mathcal{M}(x_1,...,x_i+\Delta,...,x_M) - \mathcal{M}(\boldsymbol{x})}{\Delta} \tag{1.22}$$

For each factor, there exist a finite number of $p^{M-1}\,(p-\bar{c})$ elementary effects that can be computed. Morris' original approach suggests to estimate the distribution of the elementary factors by randomly sampling r values in $\boldsymbol{\mathcal{G}}$ and computing (1.22) for each of them. Once these values are found, the estimators of the moments of the distribution can be analysed as follows:

- **Mean**, $\widehat{\mu}_i$: A high value for the mean indicates that the input variable X_i is important and, thus, its input values cause variability on the outputs of the model. Low mean would indicate an unimportant factor.
- **Standard deviation**, $\widehat{\sigma}_i$: A high value of the standard deviation indicates that the effect of the variable X_i is significant, either because of a nonlinear behaviour with respect to this variable or because of interactions with other factors.

Note that using this method with samples of r values would require a total of $2Mr$ runs ($2r$ runs per elementary effect).

1.5.1.1 Trajectory-based sampling strategy

Using a *trajectory-based* sampling strategy, the required number of model evaluations can be reduced to $(M+1)r$. The key idea is as follows: since two model evaluations are needed for computing the elementary effect of each factor, one of them can be shared between two

different elementary effects by perturbing the current point properly. This means that only one new model evaluation is needed for each factor after the starting point, leading to $M+1$ model evaluations in total. Let us introduce the following vectors and matrices:

- **Random starting point**, $\boldsymbol{x}^*$
  It is a random point (dimension $M \times 1$) of the grid, constructed in such a way that increasing its coordinates in any dimension will lead to a point that still lies in the grid.

- **Base matrix**, $\boldsymbol{B}$
  It is a $M \times (M+1)$ matrix containing ones in its strict lower triangular part and zeros on the rest.

- **Matrix $\boldsymbol{J_{i,j}}$**
  It denotes a matrix made of ones, with dimension $i \times j$.

- **Random displacements matrix $\boldsymbol{D}$**
  It consists of a diagonal $M \times M$ matrix that contains -1 or 1 on its diagonal, randomly chosen with equal probability.

- **Identity permutation $\boldsymbol{P}$**
  It is a random permutation of the identity matrix of dimension $M \times M$

The trajectory can be represented by a $M \times (M+1)$ matrix $\boldsymbol{B}^*$ that contains in each column a point, starting from the random point $\boldsymbol{x}^*$. Each successive trajectory point is obtained from the previous by sequentially increasing one of its coordinate by Δ. Each coordinate can be increased only once.
A matrix $\boldsymbol{B}'$ satisfying such properties is given by:

$$\boldsymbol{B}' = \left(\boldsymbol{J}_{M+1,1}(\boldsymbol{x}^*)^T + \Delta \boldsymbol{B}\right)^T . \tag{1.23}$$

However, such trajectory is still deterministic except for the starting point. The following formula provides a properly randomized version (Morris, 1991):

$$\boldsymbol{B}^* = \left[\left(\boldsymbol{J}_{M+1,1}(\boldsymbol{x}^*)^T + \frac{\Delta}{2}\left[(2\boldsymbol{B} - \boldsymbol{J}_{M+1,M})\boldsymbol{D}^* + \boldsymbol{J}_{M+1,1}\right]\right)\boldsymbol{P}\right]^T \tag{1.24}$$

1.5.1.2 Modified estimator

As it is based on the mean, the estimator $\widehat{\mu}$ fails to identify the effects of those factors that can cancel each other out (i.e., alternate signs). To avoid this drawback, a new estimator $\widehat{\mu}_*$ was introduced in Campolongo et al. (2007) which, instead of computing the mean of the elementary effects of (1.22), computes the mean of its absolute values:

$$d_i^*(\boldsymbol{x}) = \left|\frac{\mathcal{M}(x_1, ..., x_i + \Delta, ..., x_M) - \mathcal{M}(\boldsymbol{x})}{\Delta}\right| \tag{1.25}$$

This estimator, however, lacks the information about the sign of the effect introduced by the factor; yet combined with the previous $\widehat{\mu}_i$ and $\widehat{\sigma}_i$ it can provide more complete information about the actual importance of each input variable, the sign of its effect and the nature of this effect (linear, non-linear or interactions).

1.5.1.3 Graphical representation

Morris sensitivity analysis provides two estimators for each variable $\hat{\mu}_i$ and $\hat{\sigma}_i$. It is customary to represent them in a scatter-plot, often referred to as a Morris plot. Figure 1 shows a Morris plot for an 8-dimensional problem (see Section 2.3.5 for details on the model).

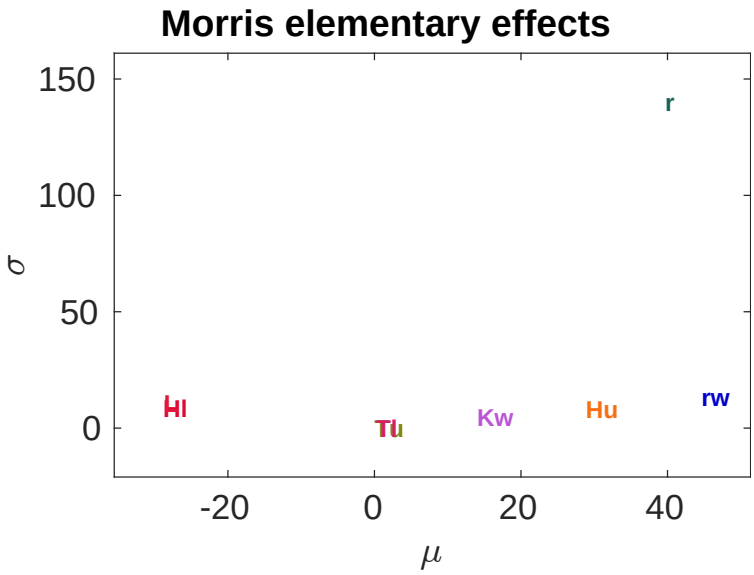

Figure 1: Morris plot for an 8-dimensional problem (see Section 2.3.5 for details). The μ of each index is given by its position on the $x-$axis, while the corresponding σ by that on the $y-$axis.

Each point on the plot is the short name of one of input variables. Its position on the $\mu\sigma$-plane is given by the corresponding $\hat{\mu}_i$ and $\hat{\sigma}_i$. Points close to the origin correspond to unimportant variables. Points far away from the origin in the μ direction are generally important. Points far away from the origin in the σ direction represent variables with significantly non-linear contributions, hence sometimes difficult to properly assess.

1.5.2 Sobol indices

Sobol' indices (Sobol', 1993) are based on the idea of defining the expansion of the computational model into summands of increasing dimension. Analogously, the total variance of the model is described in terms of the sum of the variances of the summands.
Let us assume that the input factors are uniform variables in $[0,1]$ and, therefore, that the support of the input set is $\mathcal{D}_{\boldsymbol{X}} = [0,1]^M$.

The Sobol' decomposition is defined by:

$$f(x_1, ..., x_M) = f_0 + \sum_{i=1}^{M} f_i(x_i) + \sum_{1 \le i < j \le M} f_{ij}(x_i, x_j) + ... + f_{1,2,...,M}(x_1, ..., x_M), \quad (1.26)$$

where the following conditions hold:

i) The term f_0 is constant and equal to the expected value of $f(\boldsymbol{X})$.

ii) The integrals of the summands with respect to their own variables $f_{(i_1,...,i_s)}$ vanish:

$$\int_0^1 f i_1, ..., i_s(x_{i_1}, ..., x_{i_s})\, dx_{i_k} = 0 \quad if\ 1 \le k \le s \quad (1.27)$$

For integrable functions in $\mathcal{D}_{\boldsymbol{X}}$, this expansion is shown to exist and to be unique in Sobol' (1993). It is also shown that all the summands of the expansion can be computed by integrals in a recursive manner as follows:

$$\begin{aligned} f_0 &= \int_{\mathcal{D}_{\boldsymbol{X}}} f(\boldsymbol{x}) d\boldsymbol{x}, \\ f_i(x_i) &= \int_0^1 ... \int_0^1 f(\boldsymbol{x}) d\boldsymbol{x}_{\sim i} - f_0, \\ f_{ij}(x_i, x_j) &= \int_0^1 ... \int_0^1 f(\boldsymbol{x}) d\boldsymbol{x}_{\sim (ij)} - f_0 - f_i(x_i) - f_j(x_j), \end{aligned} \quad (1.28)$$

where $\boldsymbol{x} = \{x_1, ..., x_M\}$ and the notation $\sim$ indicates that factors are excluded, *e.g.*

$$\boldsymbol{x}_{\sim i} = (x_1, ..., x_{i-1}, x_{i+1}, ..., x_M) \quad (1.29)$$

The higher order summands can be constructed in a analogous way. However, the use of Eq. (1.26) is only for computing the variance decomposition, which is presented next.

1.5.2.1 Variance

Following the previous results, the total variance of $f(\boldsymbol{X})$ is defined as:

$$D = \int_{\mathcal{D}_{\boldsymbol{X}}} f^2(\boldsymbol{x}) d\boldsymbol{x} - f_0^2, \quad (1.30)$$

and the partial variances are computed as:

$$D_{i_1,...,i_s} = \int_0^1 ... \int_0^1 f^2_{i_1,...,i_s}(x_{i_1}, ..., x_{i_s}) dx_{i_1} ... dx_{i_s} \quad 1 \le i_1 < ... < i_s \le M;\ s = 1, ..., M \quad (1.31)$$

since the summands have zero mean due to Eq. (1.27). The partial variances have the property that they sum up to the total variance. This leads to a natural definition of the sensitivity measures:

$$S_{i_1,...,i_s} = \frac{D_{i_1,...,i_s}}{D}, \quad (1.32)$$

which represent the relative contribution of each group of variables $\{x_{i_1}, \ldots, x_{i_s}\}$. The first-order Sobol' indices S_i represent the effect of each input variable alone, while the 2-terms interaction effects, *e.g.* $S_{ij},\ i \neq j$, account for the effects of the interactions of the variables i and j that cannot be decomposed into the contributions of i and j separately. Higher order interactions can be interpreted analogously.
The variances described in (1.30) may be computed by means of Monte Carlo simulation using the following estimators:

$$\widehat{f_0} = \frac{1}{N}\sum_{n=1}^{N} f(\boldsymbol{x}^{(n)}), \tag{1.33}$$

$$\widehat{D} = \frac{1}{N}\sum_{n=1}^{N} f^2(\boldsymbol{x}^{(n)}) - \widehat{f_0}^2, \tag{1.34}$$

$$\widehat{D}_i = \frac{1}{N}\sum_{n=1}^{N} f(x_i^{(n)}, \boldsymbol{x}_{\sim i}^{(n)}) f(x_i^{(n)}, \boldsymbol{x}'^{(n)}_{\sim i}) - \widehat{f_0}^2, \tag{1.35}$$

where $\boldsymbol{x}'$ denotes a realization of $\boldsymbol{X}$ independent of $\boldsymbol{x} = \left\{x_i^{(n)}, \boldsymbol{x}_i^{(n)}\right\}^{\mathsf{T}}$, and the subscript $\boldsymbol{x}_{j,\sim i}$ indicates the $j-th$ realization of $\boldsymbol{x}$ which does not contain the factor i.

1.5.2.2 Total Sobol' indices

The total Sobol' index of input variable X_i, denoted S_i^T, is the sum of all the Sobol' indices involving this variable:

$$S_i^T = \sum_{\{i_1,\ldots,i_s\} \supset i} S_{i_1,\ldots,i_s}. \tag{1.36}$$

Since this definition is not practical for actual computations (it would result in computing each index separately), the idea of computing indices of groups of variables is developed.

Let us separate the components of the input vector $\boldsymbol{x} \in [0,1]^M$ into two groups, namely $\mathbf{v} = \{i_1, ..., i_s\} \subset \{1, ..., M\}$ and $\mathbf{w} = \sim\mathbf{v}$, such that $y = f(\boldsymbol{x}) = f(\boldsymbol{x}_\mathbf{v}, \boldsymbol{x}_\mathbf{w})$. The Sobol' index with respect to $\mathbf{v}$ can thus be written as:

$$S_\mathbf{v} = \frac{\mathrm{Var}\left[\mathbb{E}\left[Y|\boldsymbol{X}_\mathbf{v} = \boldsymbol{x}_\mathbf{v}\right]\right]}{\mathrm{Var}\left[Y\right]} \tag{1.37}$$

For clarity, from now on, all the sums that have no superscript or subscript are considered to be $\sum = \sum_{i=1}^{N}$. Let us now denote by $y^\mathbf{v} = f(\boldsymbol{x}_\mathbf{v}, \boldsymbol{x}'_\mathbf{w})$, where $\boldsymbol{x}'_\mathbf{w}$ is a sample of $\boldsymbol{X}_\mathbf{w}$ independent of $\boldsymbol{x}_\mathbf{w}$. Analogously $\boldsymbol{X}'_\mathbf{w}$ denotes an independent copy of $\boldsymbol{X}_\mathbf{w}$ and $Y_\mathbf{v} = f(X_\mathbf{v}, X'_\mathbf{w})$. Following Janon et al. (2013), we can rewrite the index in Eq. (1.37) in terms of sample covariance as follows:

$$S_\mathbf{v} = \frac{\mathrm{Cov}\left[Y, Y_\mathbf{v}\right]}{\mathrm{Var}\left[Y\right]}, \tag{1.38}$$

which leads to a first estimator, hereafter referred to as the *Homma estimator*, initially pro-

posed by Saltelli and Homma (1996)

$$S_{\mathbf{v}} = \frac{\frac{1}{N}\sum y_i y_i^{\mathbf{v}} - (\frac{1}{N}\sum y_i)(\frac{1}{N}\sum y_i^{\mathbf{v}})}{\frac{1}{N}\sum y_i^2 - (\frac{1}{N}\sum y_i)^2}. \tag{1.39}$$

A similar estimator shown to perform better in most cases, hereafter referred to as the "Janon estimator", is described in Janon et al. (2013):

$$T_{\mathbf{v}} = \frac{\frac{1}{N}\sum y_i y_i^{\mathbf{v}} - \left(\frac{1}{N}\sum\left[\frac{y_i+y_i^{\mathbf{v}}}{2}\right]\right)^2}{\frac{1}{N}\sum\left[\frac{(y_i)^2+(y_i^{\mathbf{v}})^2}{2}\right] - \left(\frac{1}{N}\sum\left[\frac{y_i+y_i^{\mathbf{v}}}{2}\right]\right)^2}. \tag{1.40}$$

It is clear from the above expressions that for computing the effect of one group of variables, only one estimator is needed. Let us now denote by $S_{\sim i}$ the sensitivity measure of all the factors excluding factor i, i.e. $S_{\sim i} = S_{\mathbf{v}}$, where $\mathbf{v} = \{1, ..., i-1, i+1, ..., M\}$.
This index accounts for the sum of all the single effects and interactions between all variables except those that include factor i. By the definition of the indices in Eq. (1.32) we know that they sum to one. Therefore the total Sobol' index of variable X_i can be computed as:

$$S_i^T = 1 - S_{\sim i}, \tag{1.41}$$

and the term $S_{\sim i}$ can be estimated using one of the estimators in (1.39) or (1.40).

1.5.2.3 Higher order indices

To compute higher order indices, we can use the definition of the total Sobol' index in Eq. (1.36), and follow the same reasoning used in Eq. (1.41). For example, the second order index of (X_i, X_j) is the total effect of these two variables (denoted by $S_{\{ij\}}$) minus the single effect of each of them:

$$S_{ij} = S_{\{ij\}} - S_i - S_j. \tag{1.42}$$

If the lower order indices are known, for each new index only one integral must be computed, (in this case, the estimator of $S_{\{ij\}}$) and it can be done using one of the estimators in Eq. (1.39) or Eq. (1.40).

1.5.2.4 Confidence intervals

The bootstrap method provides a simple and relatively inexpensive way of estimating confidence intervals on the estimations of the Sobol' indices (Efron, 1979). In its simplest form, bootstrap consists of resampling with replacement new points from the original sample set without additional sampling of the original model. The estimator of interest $\widehat{\theta}$ is then computed for each of the B bootstrap samples. From the resulting collection of estimators, namely $\widehat{\Theta} = \{\widehat{\theta}_1, \widehat{\theta}_2, ..., \widehat{\theta}_B\}$, empirical quantiles can be calculated to get confidence intervals on $\widehat{\Theta}$.

1.5.3 PCE-based Sobol indices

Sobol' indices are traditionally evaluated by Monte Carlo simulation which make them difficult to use when computationally expensive models $\mathcal{M}$ are considered. In order to bypass this problem Sudret (2008) has proposed an original post-processing of polynomial chaos expansions for sensitivity analysis. As it will be demonstrated below, the Sobol' decomposition of a truncated PC expansion:

$$\widehat{y} = \mathcal{M}^{PC}(\boldsymbol{x}) = \sum_{\boldsymbol{\alpha}\in\mathcal{A}} \widehat{y}_{\boldsymbol{\alpha}}\, \Psi_{\boldsymbol{\alpha}}(\boldsymbol{x}) \tag{1.43}$$

can be established analytically. For more information about polynomial chaos expansions please refer to the UQLAB User Manual – Polynomial Chaos Expansions .

The truncated expansion in Eq. (1.43) can be rearranged so as to reflect the decomposition into summands of increasing order. For any non-empty set $\mathbf{u} \subset \{1,\dots,M\}$ and any finite truncation set $\mathcal{A} \subset \mathbb{N}^M$ we define:

$$\mathcal{A}_{\mathbf{u}} = \{\boldsymbol{\alpha} \in \mathcal{A} :\ k \in \mathbf{u} \Leftrightarrow \alpha_k \neq 0, k = 1,\dots,M\} \tag{1.44}$$

In other words $\mathcal{A}_{\mathbf{u}}$ contains all the multi-indices within the truncation set $\mathcal{A}$ which have non zero components $\alpha_k \neq 0$ if and only if $k \in \mathbf{u}$. The sum of the associated PC expansion terms form a function which depends *only* on input variables $\boldsymbol{x}_{\mathbf{u}}$. The reordering of Eq. (1.43) reads:

$$\mathcal{M}_{\mathcal{A}}(\boldsymbol{x}) = y_0 + \sum_{\substack{\mathbf{u}\subset\{1,\dots,M\}\\ \mathbf{u}\neq\varnothing}} \sum_{\boldsymbol{\alpha}\in\mathcal{A}_{\mathbf{u}}} \widehat{y}_{\boldsymbol{\alpha}}\, \Psi_{\boldsymbol{\alpha}}(\boldsymbol{x}) \tag{1.45}$$

The Sobol' decomposition described in Eq. (1.26) can also be written as follows, by introducing the generic index set $\mathbf{v} \overset{\text{def}}{=} \{i_1,\dots,i_k\} \subset \{1,\dots,M\}$ and denoting by $\boldsymbol{x}_{\mathbf{v}}$ the subvector of $\boldsymbol{x}$ obtained by extracting the components labelled by the indices in $\mathbf{v}$:

$$f(\boldsymbol{x}) = f_0 + \sum_{\substack{\mathbf{v}\subset\{1,\dots,M\}\\ \mathbf{v}\neq\varnothing}} f_{\mathbf{v}}(\boldsymbol{x}_{\mathbf{v}}) \tag{1.46}$$

As discussed in Section 1.5.2, the Sobol decomposition is *unique*, thus the comparison of Eqs. (1.45) and (1.46) readily provides the expression of each summand for the PC expansion:

$$f_{\mathbf{v}}(\boldsymbol{x}_{\mathbf{v}}) = \sum_{\boldsymbol{\alpha}\in\mathcal{A}_{\mathbf{v}}} \widehat{y}_{\boldsymbol{\alpha}}\, \Psi_{\boldsymbol{\alpha}}(\boldsymbol{x}) \tag{1.47}$$

Due to the orthonormality of the polynomial chaos basis the variance of the truncated PC expansion reads ($Y_{\mathcal{A}} = \mathcal{M}^{PC}(\boldsymbol{X})$):

$$\mathrm{Var}\left[Y_{\mathcal{A}}\right] = \sum_{\substack{\boldsymbol{\alpha}\in\mathcal{A}\\ \boldsymbol{\alpha}\neq\mathbf{0}}} \widehat{y}_{\boldsymbol{\alpha}}^2 \tag{1.48}$$

$$\mathrm{Var}\left[f_{\mathbf{v}}(\boldsymbol{x}_{\mathbf{v}})\right] = \sum_{\substack{\boldsymbol{\alpha}\in\mathcal{A}_{\mathbf{v}}\\ \boldsymbol{\alpha}\neq\mathbf{0}}} \widehat{y}_{\boldsymbol{\alpha}}^2 \tag{1.49}$$

and the associated Sobol' indices in Eq. (1.37) is just the ratio of the above two quantities.

Chapter 2

Usage

In this section a reference problem will be set up to showcase how each of the techniques in Chapter 1 can be deployed in UQLAB.

2.1 Reference problem: the borehole function

The goal of sensitivity analysis is to identify which parameters contribute the most to the variability of the model output. To this end, we will make use of a well known benchmark for sensitivity analysis: the borehole function (see, e.g., `www.sfu.ca/~ssurjano/borehole.html`). It is an analytical 8-dimensional function that models the flow of water through a borehole with uncertain parameters, given by the following equation:

$$f(\boldsymbol{x}) = \frac{2\pi T_u \left(H_u - H_l\right)}{\ln(r/r_w)\left(1 + \frac{2LT_u}{\ln(r/r_w) r_w^2 K_w} + \frac{T_u}{T_l}\right)} \tag{2.1}$$

where the description and ranges of each parameter are given in Table 1.

Table 1: Distributions and descriptions of the parameters of the borehole function in Eq. (2.1). The parameters for Gaussian distributions refer to μ and σ, to those of the associated Gaussian for lognormal distributions and to the bounds for uniform distributions.

Name	Distributions	Parameters	Description
r_w	Gaussian	$[0.1, 0.0161812]$	radius of the borehole (m)
r	Lognormal	$[7.71, 1.0056]$	radius of influence (m)
T_u	Uniform	$[63070, 115600]$	transmissivity of upper aquifer (m^2/yr)
H_u	Uniform	$[990, 1110]$	potentiometric head of upper aquifer (m)
T_l	Uniform	$[63.1, 116]$	transmissivity of lower aquifer (m^2/yr)
H_l	Uniform	$[700, 820]$	potentiometric head of lower aquifer (m)
L	Uniform	$[1120, 1680]$	length of borehole (m)
K_w	Uniform	$[9855, 12045]$	hydraulic conductivity of borehole (m/yr)

An example UQLAB script that showcases all of the sensitivity methods currently available on the borehole function can be found in the example file:

`Examples/Sensitivity/uq_Example_Sensitivity_01_Borehole_Model.m`

2.2 Problem set-up

The model in Eq. (2.1) is implemented as a Matlab `m-file` in:

`Examples/SimpleTestFunctions/uq_borehole.m`

To use it in a UQLAB analysis, we need to configure a MODEL module:

```
MOpts.mFile = 'uq_borehole' ;
myModel = uq_createModel(MOpts);
```

For more details about the configuration options available for a model, please refer to UQLAB User Manual – the MODEL module .

Similarly, the distributions in Table 1 can be implemented in UQLAB as follows:

```
IOpts.Marginals(1).Name = 'rw';
IOpts.Marginals(1).Type = 'Gaussian';
IOpts.Marginals(1).Parameters = [0.10, 0.0161812];

IOpts.Marginals(2).Name = 'r';
IOpts.Marginals(2).Type = 'Lognormal';
IOpts.Marginals(2).Parameters = [7.71, 1.0056];

IOpts.Marginals(3).Name = 'Tu';
IOpts.Marginals(3).Type = 'Uniform';
IOpts.Marginals(3).Parameters = [63070, 115600];

IOpts.Marginals(4).Name = 'Hu';
IOpts.Marginals(4).Type = 'Uniform';
IOpts.Marginals(4).Parameters = [990, 1110];

IOpts.Marginals(5).Name = 'Tl';
IOpts.Marginals(5).Type = 'Uniform';
IOpts.Marginals(5).Parameters = [63.1, 116];

IOpts.Marginals(6).Name = 'Hl';
IOpts.Marginals(6).Type = 'Uniform';
IOpts.Marginals(6).Parameters = [700, 820];

IOpts.Marginals(7).Name = 'L';
IOpts.Marginals(7).Type = 'Uniform';
IOpts.Marginals(7).Parameters = [1120, 1680];

IOpts.Marginals(8).Name = 'Kw';
IOpts.Marginals(8).Type = 'Uniform';
IOpts.Marginals(8).Parameters = [9855, 12045];

myInput = uq_createInput(IOpts);
```

For more details about the configuration options available for an INPUT object, please refer to the to UQLAB User Manual – the INPUT module .

2.3 Sensitivity analysis with different methods

Now all the different sensitivity analyses shall be specified and run, so that the corresponding results can be visualized. The following methods are showcased in this section:

- Input/output correlations: Section 2.3.1
- Standard regression coefficients (SRC/SRRC): Section 2.3.2
- Perturbation method: Section 2.3.3
- Cotter method: Section 2.3.4
- Morris elementary effects: Section 2.3.5
- Sobol' sensitivity indices: Section 2.3.6

2.3.1 Input/output correlation

Specifying an input/output correlation analysis as described in Section 1.3.1 only requires the size of the sample from which to calculate the correlations. To create and run a correlation analysis based on 10,000 points, the following syntax is used:

```
CorrSensOpts.Type = 'uq_sensitivity';
CorrSensOpts.Method = 'Correlation';
CorrSensOpts.Correlation.SampleSize = 10000;
CorrAnalysis = uq_createAnalysis(CorrSensOpts);
```

Once the analysis is performed, a report with the sensitivity results can be printed on screen by:

```
uq_print(CorrAnalysis)
```

The results can be visualized graphically as follows:

```
uq_display(CorrAnalysis)
```

which produces the image in Figure 2.

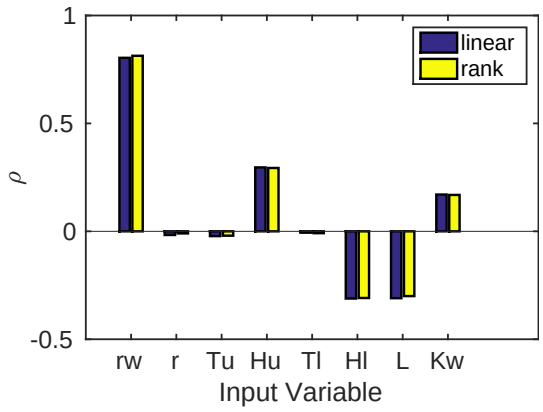

Figure 2: Input/output linear and rank correlation results from the analysis described in Section 2.3.1.

2.3.1.1 Accessing the results

The analysis results can be accessed in the `CorrAnalysis.Results` structure:

```
CorrAnalysis.Results
ans =
  CorrIndices: [8x1 double]
  RankCorrIndices: [8x1 double]
  ExpDesign: [1x1 struct]
  Cost: 10000
  VariableNames: {'rw'  'r'  'Tu'  'Hu'  'Tl'  'Hl'  'L'  'Kw'}
```

In the structure, the `CorrIndices` and `RankCorrIndices` fields are arrays containing the indices in Eq. (1.1) and (1.4), respectively. The `Cost` field represents the number of model evaluations needed to calculate the indices, in this case equal to the sample size. A full description of the format of the output can be found in Table 11.

2.3.1.2 Advanced options

The input/output correlation method is extremely simple. Thus there are only a few options available for configuration. The most relevant are related to the generation of the sample from which the coefficients are computed. In the following, two typical advanced usage scenarios are presented:

- **Specify the sampling scheme**: by default, model realizations are sampled via simple Monte Carlo from the input distributions. It is possible, however, to specify a different sampling scheme (*e.g.* Latin Hypercube Sampling or Sobol' pseudo-random sequences), by adding the following option:

```
% use LHS to sample the inputs
CorrSensOpts.Correlation.Sampling = 'LHS';
```

Any of the sampling schemes supported by the default input module can be specified (see Section 3.2, Page 32 of the UQLAB User Manual – the INPUT module).

- **Manually provide the samples:** if input samples are already available and no additional sampling is needed, it is possible to use them directly for the analysis. In case the samples are available in variables `X_ED` and the corresponding model evaluations in variable `Y_ED`, they can be added as:

```
% manually specify the samples
CorrSensOpts.Correlation.Sample.X = X_ED;
CorrSensOpts.Correlation.Sample.Y = Y_ED;
```

  Note that in this case no `SampleSize` needs to be specified.

For a comprehensive list of the options available to the Correlation method, please see Table 3.

2.3.2 Standard Regression Coefficients

Standard regression coefficients (SRC) and standard rank-regression coefficients (SRRC), described in Section 1.3.2 only require a single configuration option: the size of the sample on which to perform regression.
To create and run a SRC analysis based on a sample of 10,000 points, the following syntax is used:

```
SRCSensOpts.Type = 'uq_sensitivity';
SRCSensOpts.Method = 'SRC';
SRCSensOpts.SRC.SampleSize = 10000;
SRCAnalysis = uq_createAnalysis(SRCSensOpts);
```

Once the analysis is performed, a report with the sensitivity results can be printed on screen by:

```
uq_print(SRCAnalysis)
```

The results can be visualized graphically as follows:

```
uq_display(SRCAnalysis)
```

which produces the image in Figure 3.

2.3.2.1 Accessing the results

The analysis results can be accessed in the `SRCAnalysis.Results` structure:

```
SRCAnalysis.Results
ans =
  SRCIndices: [1x8 double]
  SRRCIndices: [1x8 double]
  Cost: 1000
  VariableNames: {'rw'  'r'  'Tu'  'Hu'  'Tl'  'Hl'  'L'  'Kw'}
```

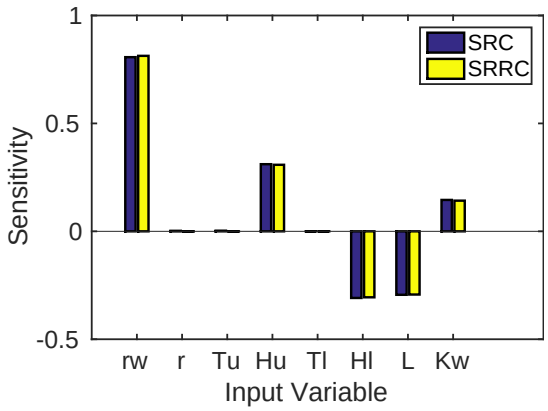

Figure 3: SRC and SRRC sensitivity results from the analysis described in Section 2.3.2.

In the structure, the `SRCIndices` and `SRRCIndices` fields are arrays containing the indices in Eq. (1.8) and (1.11), respectively. The `Cost` field represents the number of model evaluations needed to calculate the indices, in this case equal to the sample size. A full description of the format of the output can be found in Table 12.

2.3.2.2 Advanced options

In the following, two typical advanced usage scenarios of the SRC/SRRC method are presented:

- **Specify the sampling scheme**: by default, model realizations are sampled via simple Monte Carlo from the input distributions. It is possible, however, to specify a different sampling scheme (*e.g.*Latin Hypercube Sampling or Sobol' pseudorandom sequences), by adding the following option:

  ```
  % use LHS to sample the inputs
  SRCSensOpts.SRC.Sampling = 'LHS';
  ```

  Any of the sampling schemes supported by the default input module can be specified (see Section 3.2, Page 32 of the UQLAB User Manual – the INPUT module).

- **Manually provide the samples:** if module samples are already available and no additional sampling is needed, it is possible to use them directly for the analysis. In case the samples are available in variables `X_ED` and the corresponding model evaluations in variable `Y_ED`, they can be added as:

  ```
  % manually specify the samples
  SRCSensOpts.SRC.Sample.X = X_ED;
  SRCSensOpts.SRC.Sample.Y = Y_ED;
  ```

  Note that in this case no `SampleSize` needs to be specified.

For a comprehensive list of options available to the SRC method, please see Table 4.

2.3.3 Perturbation method

Perturbation-based sensitivity analysis (Section 1.4.1) does not require any configuration options since it is based on the numerical evaluation of the gradient of the model around the mean value. To create and run a perturbation analysis, the following syntax is used:

```
PerturbationSensOpts.Type = 'uq_sensitivity';
PerturbationSensOpts.Method = 'Perturbation';
PerturbationAnalysis = uq_createAnalysis(PerturbationSensOpts);
```

Once the analysis is performed, a report with the sensitivity results can be printed on screen by:

```
uq_print(PerturbationAnalysis)
```

The results can be visualized graphically as follows:

```
uq_display(PerturbationAnalysis)
```

which produces the image in Figure 4.

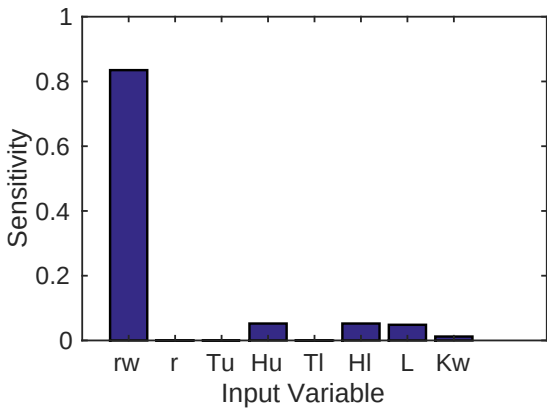

Figure 4: Perturbation-based sensitivity results from the analysis described in Section 2.3.3.

2.3.3.1 Accessing the results

The analysis results can be accessed in the `PerturbationAnalysis.Results` structure:

```
PerturbationAnalysis.Results
ans =
  Mu: 7.093190986636091e+01
  Var: 7.541825411751066e+02
  Sensitivity: [1x8 double]
  Cost: 9
  VariableNames: {'rw'  'r'  'Tu'  'Hu'  'Tl'  'Hl'  'L'  'Kw'}
```

In the structure, the `Mu` and `Var` fields report the estimated mean and variance of the model output, while the `Sensitivity` field is the array containing the indices in Eq. (1.15). The `Cost` field represents the number of model evaluations needed to calculate the indices, in

this case equal to the cost of evaluating partial derivatives of the model. A full description of the format of the output can be found in Table 13.

2.3.3.2 Advanced options

The advanced options for the perturbation method are related to the numerical computation of the gradient of the model. They allow one to:

- **Specify the gradient method**: by default, the gradient is calculated with the "forward" method (to minimize the number of model evaluations). It can be set to any of "Forward","Centered" (more accurate, but more expensive to calculate) or "Backwards" as follows:

```
% use centered gradient calculation
PerturbationSensOpts.Gradient.Method = 'Centered';
```

- **Specify the gradient step mode and its value:** by default, the gradient is calculated numerically in the standardized space (*i.e.* in units of standard deviation), and the finite difference step is set to $h = 10^{-3}$. It is possible, however to specify both the mode and the value as follows:

```
% specify the FD gradient step as absolute (not normalized)
PerturbationSensOpts.Gradient.Step = 'absolute';
% and give it a value of 0.0001
PerturbationSensOpts.Gradient.h = 0.0001;
```

  Note that no `SampleSize` needs to be specified. The `'absolute'` option should be used with caution when the magnitude of the various input parameters varies from one another.

For a comprehensive list of options available to the perturbation method, please see Table 5.

2.3.4 Cotter Measure

The Cotter measure does not require any specific configuration option, because the input bounds (see Section 1.4.2) are set by default to the following values, depending on the input distributions:

- *Uniform, beta and truncated distributions:* $x_i^{\pm}$ match the upper and lower bounds of the corresponding input distribution.
- *Lognormal distribution:* $x_i^- = \max(\mu_i - \sigma_i, 0)$, $x_i^+ = \mu_i + \sigma_i$.
- *Unbounded distributions:* $x_i^- = \mu_i - \sigma_i$, $x_i^+ = \mu_i + \sigma_i$.

where μ_i and σ_i are the mean and standard deviation of variable X_i, respectively, as defined in the INPUT object (see Section 2.2). The Cotter sensitivity analysis is then created as follows:

```
CotterSensOpts.Type = 'uq_sensitivity';
CotterSensOpts.Method = 'Cotter';
CotterAnalysis = uq_createAnalysis(CotterSensOpts);
```

Once the analysis is performed, a report with the sensitivity results can be printed on screen by:

```
uq_print(CotterAnalysis)
```

The results can also be visualized graphically as follows:

```
uq_display(CotterAnalysis)
```

which produces the image in Figure 5.

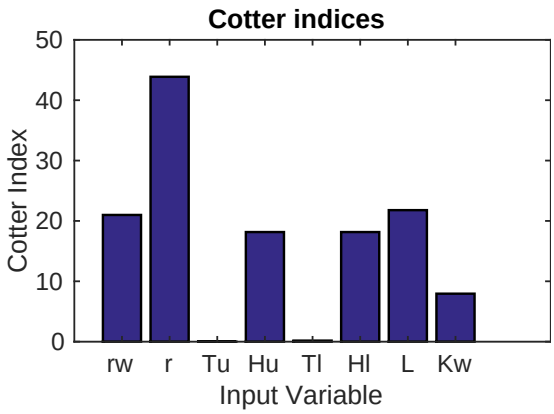

Figure 5: Cotter sensitivity results from the analysis described in Section 2.3.4.

2.3.4.1 Accessing the results

The analysis results can be accessed in the `CotterAnalysis.Results` structure:

```
CotterAnalysis.Results
ans =
  CotterIndices: [1x8 double]
  EvenOrder: [1x8 double]
  OddOrder: [1x8 double]
  Cost: 18
  VariableNames: {'rw'  'r'  'Tu'  'Hu'  'Tl'  'Hl'  'L'  'Kw'}
```

The `CotterIndices` field is the array containing the indices in Eq. (1.16), while the `OddOrder` and `EvenOrder` arrays contain the odd and even order effects in Eq. (1.17) and (1.18), respectively. The `Cost` field represents the number of model evaluations needed to calculate the indices, in this case $(2M + 2)$, where M is the number of input variables.

A full description of the format of the output can be found in Table 14.

2.3.4.2 Advanced options

The Cotter method only allows for the customization of the low and high factor bounds $[x_i^-, x_i^+]$ (see Section 1.4.2). The bounds can be specified in two ways: absolute or relative to the standard deviation of the inputs. The behaviour of the factors can be specified in the `CotterOptions.Factors` structure as follows:

- **Absolute bounds:** they can be specified by setting the 1×2 array `CotterOptions.Factors(ii).Boundaries` that contains lower and upper bounds for each variable. As an example, to set all of the M inputs factors low and high level to 0 and 1, respectively, the following code can be used:

```
for ii = 1:M
  CotterSensOptions.Factors(ii).Boundaries = [0 1];
end
```

- **Relative bounds:** bounds can be specified relative to the standard deviation according to the following equation:

$$x_i^{\pm} = \mu_i \pm k_i \sigma_i \tag{2.2}$$

where k_i is a positive scalar, μ_i and σ_i are the $i-$th factor mean and standard deviations specified as in Section 2.2. There are two ways to specify the k_i:

  - If the `Factors` structure is a 1-element structure and `Factors.Boundaries` is a scalar, all the k_i are set to its value, *e.g.*:

```
CotterSensOptions.Factors.Boundaries = 0.5;
```

    will set all of the $k_{i,\dots,M} = 0.5$.

  - If the `Factors` structure has M elements and each `Factors(ii).Boundaries` is a scalar b_i, the $k_i = b_i$, *e.g.*:

```
for ii = 1:M
  CotterSensOptions.Factors(ii).Boundaries = ii;
end
```

    will manually set each of the $k_i = i$.

For a comprehensive list of options available to the Cotter method, please see Table 6.

2.3.5 Morris Method

The Morris method only requires one configuration option, namely the maximum number of full-model evaluations to be performed to estimate the elementary effects. Hence, a Morris analysis with default settings and a maximum of 10^4 model evaluations can be configured as follows:

```
MorrisSensOpts.Type = 'uq_sensitivity';
MorrisSensOpts.Method = 'Morris';
```

```
MorrisSensOpts.Morris.Cost = 10000;
MorrisAnalysis = uq_createAnalysis(MorrisSensOptions);
```

Note that the `MorrisSensOpts.Morris.Cost` represents the maximum allowed cost. UQLAB will use this constraint conservatively, so that the effective cost is lower than or equal to the specified value. Once the analysis is performed, a report with the sensitivity results can be printed on screen by:

```
uq_print(MorrisAnalysis)
```

The results can also be visualized graphically as follows:

```
uq_display(MorrisAnalysis)
```

which produces the image in Figure 6. The results in Figure 6 are shown with the standard

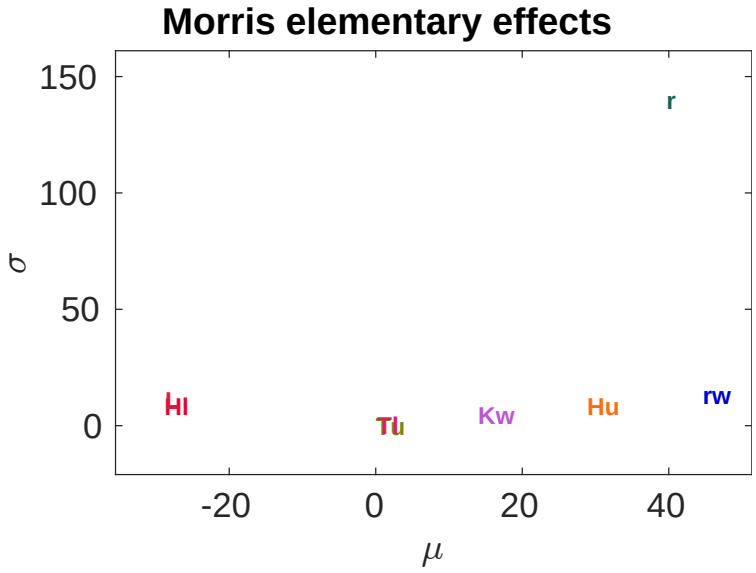

Figure 6: Morris sensitivity results from the analysis described in Section 2.3.5.

Morris format: the x-axis represents the mean value of the elementary effects related to each variable, while the y-axis represents the corresponding standard deviation. Values close to the origin represent unimportant parameters. Parameters far from the origin in the x-direction are important, while a large y-coordinate means that the parameter has a strong degree of non-linearity/interaction.

2.3.5.1 Accessing the results

The analysis results can be accessed in the `MorrisAnalysis.Results` structure:

```
MorrisAnalysis.Results
ans =
  Mu: [1x8 double]
  MuStar: [1x8 double]
  Std: [1x8 double]
```

```
  Cost: 9999
  VariableNames: {'rw'  'r'  'Tu'  'Hu'  'Tl'  'Hl'  'L'  'Kw'}
```

The mean value of the elementary effects for each variable is given in the `Mu` field, while the corresponding standard deviation in the `Std` field (see Eq. (1.22)). The `MuStar` field corresponds instead to the mean of the modified estimator in Eq. (1.25), especially useful for ranking purposes. The `Cost` field represents instead the effective number of model evaluations used to calculate the elementary effects. A full description of the format of the output can be found in Table 15.

2.3.5.2 Advanced options

There are several parameters that can be fine-tuned in a Morris' sensitivity analysis. They include:

- **Min-max boundaries of the parameters:** they are similar to the boundaries for the Cotter measure (see Section 2.3.4.2). They can be specified with the `SensOptions.Factors` field, with the same syntax and the same default values as for the Cotter sensitivity case.
- **Perturbation grid:** the input space between the variable boundaries is explored through a regular grid as defined in Eqs (1.19) to (1.21). Its parameters can be specified as follows:

  ```
  MorrisSensOpts.GridLevels = 6;
  MorrisSensOpts.PerturbationSteps = 2;
  ```

  where `GridLevels` and `PerturbationSteps` correspond to p in Eq. (1.19) and $\bar{c}$ in Eq. (1.21), respectively.

The number of replications for each elementary effect (variable r, in Section 1.5.1) is determined directly from the specified `Cost` according to:

$$cost = r(M+1), \qquad r = \left\lfloor \frac{cost}{M+1} \right\rfloor \tag{2.3}$$

where $cost$ is the value specified in Section 2.3.5.
For a comprehensive list of options available to the Morris' method, please see Table 7.

2.3.6 Sobol indices (MC-based)

The Sobol measure only requires one configuration option, namely the number of model evaluations needed to calculate *each index*. Hence, a Sobol' analysis with default settings (Total + first order indices only, Janon estimator in Eq. (1.40)) and a maximum of $10,000$ model evaluations per dimension can be configured as follows:

```
SobolSensOpts.Type = 'uq_sensitivity';
SobolSensOpts.Method = 'Sobol';
SobolSensOpts.Sobol.SampleSize = 10000;
SobolAnalysis = uq_createAnalysis(SobolSensOptions);
```

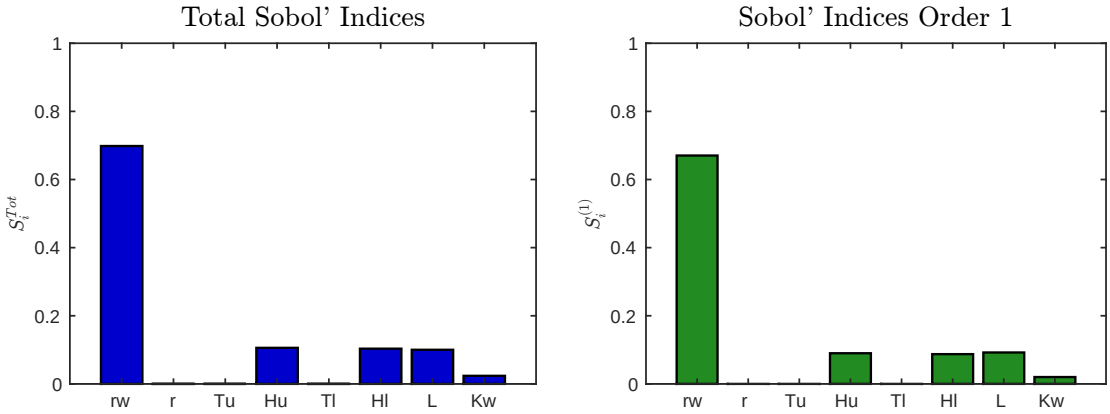

Figure 7: Sobol' sensitivity results from the analysis described in Section 2.3.6. Left: total Sobol' indices; Right: first order sobol' indices.

Note that the `SobolSensOpts.SampleSize` represents the number of model evaluations used *for each index*. This is important as the number of indices that can be calculated increases with the input dimension. The increase is linear for total and first order indices, but increases factorially with the order of the indices. Once the analysis is performed, a report with the sensitivity results can be printed on screen by:

```
uq_print(SobolAnalysis)
```

The results can also be visualized graphically as follows:

```
uq_display(SobolAnalysis)
```

which produces the images in Figure 7.

2.3.6.1 Accessing the results

The analysis results can be accessed in the `SobolAnalysis.Results` structure:

```
SobolAnalysis.Results
ans =
  AllOrders: {[8x1 double]}
  Total: [8x1 double]
  FirstOrder: [8x1 double]
  VarIdx: {[8x1 double]}
  TotalVariance: 813.6991
  Cost: 100000
  ExpDesign: [1x1 struct]
  PCEBased: 0
  VariableNames: {'rw'  'r'  'Tu'  'Hu'  'Tl'  'Hl'  'L'  'Kw'}
```

The `Total` and `FirstOrder` arrays contain the total and first order Sobol' indices, respectively. All the indices (excluding the total Sobol' indices but including the first order ones) are grouped by order in the `AllOrders` *cell array*. The value of the indices of order j are given as column vectors in `AllOrders{j}`. The corresponding list of variables for each order is given in the `VarIdx{j}` array. To access the first order indices and the corresponding list

of variables in our example, *e.g.*:

```
SobolAnalysis.Results.AllOrders{1}
ans =
  0.6439
  -0.0176
  -0.0176
  0.0800
  -0.0177
  0.0846
  0.0667
  0.0061

SobolAnalysis.Results.VarIdx{1}
ans =
  1
  2
  3
  4
  5
  6
  7
  8
```

Note: because the number of indices depends both on M and on the index order, `AllOrders` and `VarIdx` are MATLAB *cell arrays*, hence their contents must be accessed with the `{}` notation.

As in previous cases, `Cost` gives the total cost (in model evaluations) of the analysis. `TotalVariance` is an array that contains the total variance of the model response. The `PCEBased` flag is `0` if the indices are calculated via Sampling as in Section 1.5.2 or `1` if they are calculated via PCE expansion as in Section 1.5.3. Finally, the `ExpDesign` field contains information about the model evaluations used to calculate the Sobol' indices.

For a complete overview of the output format of the Sobol' analysis module, please refer to Table 16.

2.3.6.2 Advanced Options

Several options can be fine-tuned for the calculation of Sobol' indices. The most common customization options are reported below:

- **Specify the maximum Sobol' index order:** the maximum Sobol' index order can be specified with the `Sobol.Order` option:

  ```
  SobolSensOpts.Sobol.Order = 2;
  ```

- **Specify an estimator:** to use the estimator T in Eq. (1.40), set the following:

  ```
  SobolSensOpts.Sobol.Estimator = 'T';
  ```

- **Specify the sampling scheme:** the Monte-Carlo sampling scheme can be specified via

the `Sobol.Sampling` option. Note that all of the options Section 3.2, Page 32 of the UQLAB User Manual – the INPUT module are valid:

```
SobolSensOpts.Sobol.Sampling = 'LHS';
```

- **Calculate confidence intervals:** bootstrap-based confidence intervals can be calculated by UQLAB by specifying the `SobolSensOptions.Boostrap` option. The following code creates a bootstrap resampling with $B = 100$ samples and calculates confidence bounds corresponding to the 0.025 and 0.975 quantiles:

```
% Number of resamplings:
SobolSensOpts.Bootstrap.Replications = 100;
% Calculate the Alpha quantiles
SobolSensOpts.Alpha = 0.05;
```

  If bootstrap is used, an additional `Bootstrap` field is added to `SobolAnalysis.Results`:

```
SobolAnalysis.Results.Bootstrap
ans =
  Total: [1x1 struct]
  FirstOrder: [1x1 struct]
  AllOrders: {[1x1 struct]}
```

  Details on how the Bootstrap results are reported are given in Table 17.

- **Do not store the model evaluations:** by default, the full model evaluations that are run to calculate the Sobol' indices are saved, as they can be very expensive to calculate. However, when higher order indices are calculated from inexpensive models, a very large number of model evaluations may be generated (order of 10^9), hence creating taxing storage needs. It is possible, however, to remove the model evaluations after the index calculation is complete, by setting:

```
SobolSensOpts.SaveEvaluations = false;
```

For a complete reference list of options available for the calculation of Sobol' indices, please refer to Table 8.

2.3.7 PCE-based Sobol' indices

If a polynomial chaos expansion (PCE) metamodel created by UQLAB is available, it is possible to calculate Sobol' indices *analytically* from its coefficients. A comparison between Sobol' indices calculated with Sampling (see Section 1.5.2) and directly from PCE (see Section 1.5.3) for the borehole model is provided in the file:
`Examples/Sensitivity/uq_Example_Sensitiviy_02_Sobol.m`.

The problem is set up as in Section 2.2, but, in addition, a PCE metamodel is created before defining the sensitivity analysis. This can be accomplished as follows (for details on how to

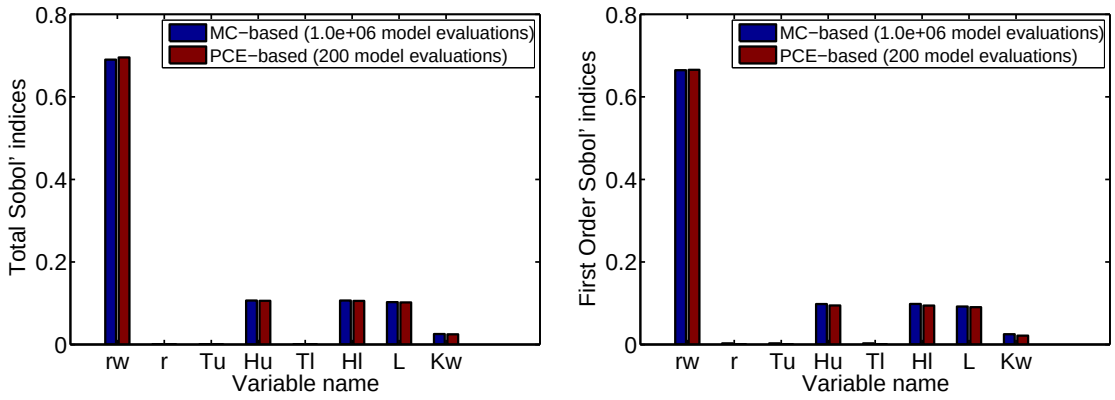

Figure 8: Comparison between MC-based and PCE-based Sobol indices for the borehole model. Left: total Sobol' indices; Right: first order Sobol' indices.

use the UQLAB's metamodelling toolbox, please refer to the Section 2 of the UQLAB User Manual – Polynomial Chaos Expansions). The following code creates a sufficiently accurate PCE of the borehole model based on an experimental design of 200 samples.

```
%  select the 'metamodel tool' in UQLab
PCEOpts.Type = 'uq_metamodel';
%  choose the Polynomial Chaos Expansion module
PCEOpts.MetaType = 'PCE';
%  PCE requires an input to be defined
PCEOpts.Input = myInput;
%  As well as a full model, if the Experimental Design is not provided
PCEOpts.FullModel = myModel;
%  Run 5th degree PCE (default: sparse PCE expansion)
PCEOpts.Degree = 5;
%  Specify the experimental design size (actual cost of the metamodel)
PCEOpts.ExpDesign.NSamples = 200;
%  Calculate the PCE coefficients
myPCE = uq_createModel(PCEOpts);
```

The Sobol' analysis is initialized as in Section 2.3.6.1, but the `SampleSize` is omitted. To create a second order Sobol' analysis the following is sufficient:

```
PCESobol.Type = 'uq_sensitivity';
PCESobol.Method = 'Sobol';
PCESobol.Sobol.Order = 2;
PCESobolAnalysis = uq_createAnalysis(PCESobol);
```

Note that there is no necessity to specify that the Sobol' indices should be calculated from the PCE coefficients: if the current model is of type "PCE metamodel", PCE-based Sobol' indices will be calculated. A comparison between Monte Carlos-based and PCE-based Sobol' indices is shown in Figure 8.

2.3.7.1 Accessing the results

The results are stored in the same format as for MC-based Sobol' indices.

```
PCESobolAnalysis.Results
```

```
ans =
  AllOrders: {[8x1 double]}
  Total: [8x1 double]
  FirstOrder: [8x1 double]
  TotalVariance: 804.2971
  VarIdx: {[8x1 double]}
  VariableNames: {'rw'  'r'  'Tu'  'Hu'  'Tl'  'Hl'  'L'  'Kw'}
  PCEBased: 1
```

See Section 2.3.6.1. Note that the `PCEBased` flag is now automatically set to 1 and that there are no `ExpDesign` nor `Cost` fields, because no new model evaluations are run.

2.3.7.2 Advanced Options

Except for the Sobol' order, that is specified as in Section 2.3.6.2, only one option can be specified for PCE-based Sobol' analysis:

- **force Monte-Carlo based estimation of the indices:** for comparison/validation, it is possible to force the calculation of the Sobol' indices through MC estimator as in Section 2.3.6. The `PCEBased` flag is then specified as follows:

```
% Force MC calculation even with a PCE metamodel
SobolSensOpts.Sobol.PCEBased = 0;
```

2.4 Sensitivity analysis of models with multiple outputs

2.4.1 Introduction

If a model has a vector-valued response $\boldsymbol{Y} = \mathcal{M}(\boldsymbol{X}) \in \mathbb{R}^{N_{out}}$, UQLAB will treat each of the N_{out} outputs independently and automatically. Full-model evaluations are of course performed only once, regardless of the number of output components. Hence, no changes in the syntax are needed to run the analyses presented throughout this chapter.
An example analysis calculating Sobol' indices on a multi-component model (deflection of a simply supported beam at multiple points) can be found in:
`Examples/Sensitivity/uq_Example_Sensitivity_04_MultipleOutputs.m`

2.4.2 Accessing multi-output results

Results are stored in the same format as described in the previous section, with the only difference that each output is stored in a different column of the results array. Consider as an example the correlation-based indices results in Section 2.3.6.1. If the number of output components were increased to $N_{out} = 5$, the results structure would read:

```
SobolAnalysis.Results
ans =
  AllOrders: {[8x5 double]}
  Total: [8x5 double]
```

```
  FirstOrder: [8x5 double]
  VarIdx: {[8x1 double]}
  TotalVariance: [1x5 double]
  Cost: 100000
  ExpDesign: [1x1 struct]
  PCEBased: 0
  VariableNames: {'rw'  'r'  'Tu'  'Hu'  'Tl'  'Hl'  'L'  'Kw'}
```

The only difference with respect to the scalar model output case ($N_{out} = 1$) lies in the fact that now all of the result arrays have one column for each output.

Note: the `VarIdx` cell array for Sobol' analysis is independent on the output component, hence its dimension is unchanged, as it corresponds to $\mathbf{v}$ in Eq. (1.37).

Chapter 3

Reference List

How to read the reference list

Structures play an important role throughout the UQLAB syntax. They offer a natural way to group configuration options and output quantities semantically. Due to the complexity of the algorithms implemented, it is not uncommon to employ nested structures to fine-tune inputs/outputs. Throughout this reference guide, we adopt a table-based description of the configuration structures.
The simplest case is given when a field of the structure is a simple value/array of values:

Table X: `Input`			
●	`.Name`	String	A description of the field is put here

which corresponds to the following syntax

```
Input.Name = 'My Input';
```

The columns correspond to name, data type and a brief description of each field. At the beginning of each row a symbol is given to inform as to whether the corresponding field is mandatory, optional, mutually exclusive, etc. The comprehensive list of symbols is given in the following table:

●	Mandatory
□	Optional
⊕	Mandatory, mutually exclusive (only one of the fields can be set)
⊞	Optional, mutually exclusive (one of them can be set, if at least one of the group is set, otherwise none is necessary)

When one of the fields of a structure is a nested structure, we provide a link to a table that describes the available options, as in the case of the `Options` field in the following example:

Table X: `Input`			
●	`.Name`	String	Description
☐	`.Options`	Table Y	Description of the `Options` structure

Table Y: `Input.Options`			
●	`.Field1`	String	Description of `Field1`
☐	`.Field2`	Double	Description of `Field2`

In some cases an option value gives the possibility to define further options related to that value. The general syntax would be

```
Input.Option1 = 'VALUE1' ;
Input.VALUE1.Val1Opt1 = ...;
Input.VALUE1.Val1Opt2 = ...;
```

This is illustrated as follows:

Table X: `Input`			
●	`.Option1`	String	Short description
		`'VALUE1'`	Description of `'VALUE1'`
		`'VALUE2'`	Description of `'VALUE2'`
⊞	`.VALUE1`	Table Y	Options for `'VALUE1'`
⊞	`.VALUE2`	Table Z	Options for `'VALUE2'`

Table Y: `Input.VALUE1`			
☐	`.Val1Opt1`	String	Description
☐	`.Val1Opt2`	Double	Description

Table Z: `Input.VALUE2`			
☐	`.Val2Opt1`	String	Description
☐	`.Val2Opt2`	Double	Description

Note: In the sequel, `double/doubles` mean a real number represented in double precision (resp. a set of such real numbers).

3.1 Create a sensitivity analysis

Syntax

```
myAnalysis = uq_createAnalysis(SOpts)
```

Input

The Struct variable `SOpts` contains the configuration information for a sensitivity analysis. The detailed list of options available is reported in Table 2.

Table 2: `SOpts`

●	`.Type`	`'uq_sensitivity'`	Identifier of the sensitivity module.
●	`.Method`	String	Sensitivity analysis method. The available options are listed below:
		`'Correlation'`	I/O correlation method (Section 2.3.1).
		`'SRC'`	Standard Regression Coefficients (Section 2.3.2).
		`'Perturbation'`	Perturbation Method (Section 2.3.3).
		`'Cotter'`	Cotter Method (Section 2.3.4).
		`'Morris'`,	Morris method (Section 2.3.5).
		`'Sobol'`,	Sobol' indices.(Section 2.3.6 and 2.3.7).
☐	`.FactorIndex`	Logical	Vector with M elements that contains `true` if the factor must be taken into account or `false` if it should be ignored in the computations. Because the total computational costs are related to the number of factors, this option can help save model evaluations.
☐	`.SaveEvaluations`	Logical default: 1	Store or not the performed model evaluations.
		1	Model evaluations are saved.
		0	Model evaluations are not saved.
☐	`.Display`	String default: `'normal'`	Controls the amount of information displayed during execution.
		`'normal'`	Default display level, only the most important information is shown.

		`'quiet'`	Minimum display level, nothing or very little information is shown.
		`'all'`	Maximum display level, all the available information is shown.
☐	`.Gradient`	See Table 5	Define the options for computing the gradient. It applies only to the perturbation method, when `SOpts.Method = 'Perturbation'`.
☐	`.Factors`	See Table 6	Defines the intervals for input parameters. It applies only when `SOpts.Method = 'Cotter'` or `SOpts.Method = 'Morris'`.
☐	`.Morris`	See Table 7	Define the options for the Morris method. It applies only when, `SOpts.Method = 'Morris'`.
☐	`.Sobol`	See Table 8 and Table 10	Define the options for the Sobol' indices. It applies only when, `SOpts.Method = 'Sobol'`.

3.1.1 Input/Output correlation options

For details on the usage, please refer to Section 2.3.1.

Table 3: `SOpts.Correlation`

☐	`.Sampling`	String default: `'MC'`	Sampling method to generate the sample. All methods in Table 6, page 32 of the UQLAB User Manual – the INPUT module are supported.
☐	`.Sample`	`.X`	Manually provided input sampling.
		`.Y`	Manually provided model response.
☐	`.SampleSize`	Double	Number of samples to be generated.

3.1.2 Standard Regression Coefficients (SRC and SRRC) options

For details on the usage, please refer to Section 2.3.2.

Table 4: `SOpts.SRC`			
☐	`.Sampling`	String default: `'MC'`	Sampling method to generate the sample. All methods in Table 6, page 32 of the UQLAB User Manual – the INPUT module are supported.
☐	`.Sample`	`.X`	Manually provided input sampling.
		`.Y`	Manually provided model response.
☐	`.SampleSize`	Double	Number of samples to be generated.

3.1.3 Perturbation Method options

For details on the usage, please refer to Section 2.3.3.

Table 5: `SOpts.Gradient`			
☐	`.Step`	String default: `'relative'`	Defines how the perturbation h used in the approximation is calculated.
		`'relative'`	h is an amount relative to the standard deviation of each variable $h_i = (\texttt{Gradient.h}) \cdot \sigma_i$
		`'absolute'`	h is defined as an absolute value.
☐	`.h`	Double default: 10^{-3}	Value of the difference for the scheme. If `Gradient.Step` is set to `'relative'`, the difference in each coordinate is computed as $h_i = (\texttt{Gradient.h}) \cdot \sigma_i$
☐	`.Method`	String default: `'forward'`	Specifies the type of finite differences scheme to be used.
		`'forward'`	Forward finite differences. Two model evaluations $\mathcal{M}(\boldsymbol{x})$ and $\mathcal{M}(x_1, \ldots, , x_i + h_i, x_{i+1}, \ldots, x_M)$ are used.
		`'backward'`	Same as forward, but the increment is backwards, $\mathcal{M}(\boldsymbol{x} - h)$.

		`'centred'` or `'centered'`	Evaluates the model at two points in the neighbourhood of the point of interest, but not at the point itself. It is more accurate, but it needs more model evaluations.

3.1.4 Cotter Measure options

For details on the usage, please refer to Section 2.3.4.

Table 6: `SOpts.Factors(i)`

●	`.Boundaries`	(1×2) Double	lower and upper boundaries for the i-th variable $[x_i^-, x_i^+]$
		Double default: 1	If `Factors(i).Boundaries` is a single scalar k, the intervals for each factor are as $x_i^{\pm} = \mu_i \pm k\sigma_i$

3.1.5 Morris Measure

For details on the usage, please refer to Section 2.3.5.

Table 7: `SOpts.Morris`

⊕	`.Cost`	Double	Maximum number of evaluations allowed for the Morris method. $Cost = (M+1)r$.
⊕	`.FactorSamples`	Double	Number of replications r for each elementary effect. $Cost = (M+1)r$.
□	`.GridLevels`	Double default: M	Number of points used to discretize each interval where the factors vary.
□	.PerturbationSteps	Double default: 1	Number of grid levels used for computing each elementary effect.

3.1.6 Sobol' indices (sampling based)

For details on the usage, please refer to Section 2.3.6.

Table 8: `SOpts.Sobol (MC-Sampling indices)`			
●	`.SampleSize`	Double	Size of each of the samples to be generated. The total number of evaluations is $N(M+2)$, for a problem of dimension M.
☐	`.Estimator`	String default: `'t'`	The type of estimator used to approximate the Sobol' indices.
		`'sobol'`, `'classic'`	Original estimator presented in Sobol' (1993), based on equations (1.33)-(1.35).
		`'homma'`, `'s'`	The estimator presented in Saltelli and Homma (1996), as described in (1.39).
		`'janon'`, `'t'` Default	Estimator proposed by Janon et al. (2013), described in equation (1.40). It is the default estimator.
☐	`.Order`	Double default: 1	Maximum order of the interactions for which the Sobol' indices are estimated.
☐	`.Sampling`	String	Sampling strategy for generating the initial samples.
		`'halton'`	The samples are created with Halton pseudo-random sampling.
		`'lhs'`	The samples are created with Latin Hypercube sampling.
		`'mc'`	The samples are created with Monte Carlo random sampling.
		`'sobol'`	The samples are created with Sobol' pseudo-random sampling.
☐	`.Bootstrap`	See Table 9	Specify the options for generating bootstrap confidence intervals for the estimates.

3.1.6.1 Boostrap error estimate on the Sobol' indices

For details on the usage, please refer to Section 2.3.6.2.

Table 9: `SOpts.Bootstrap`			
☐	`.Replications`	Double default: 0	Number of bootstrap replicates to carry out for generating confidence intervals.
☐	`.Alpha`	Double default: 0.05	The bootstrap confidence intervals are constructed with confidence level $1-\alpha$.

3.1.7 PCE-based Sobol' indices

For details on the usage, please refer to Section 2.3.7.

Table 10: `SOpts.Sobol (PCE-based indices)`			
☐	`.PCEBased`	Logical default: `true`	If set to `false`, UQLAB will compute the Monte Carlo estimates of the Sobol' indices, not the analytical ones.
☐	`.Order`	Double default: 1	Maximum order of the interactions for which the Sobol' indices are calculated.

3.2 Accessing the results

Syntax

```
myAnalysis = uq_createAnalysis(SOpts);
```

Output

Each of the sensitivity analyses presented in Chapter 2 produces different results, albeit with an overall similar structure. The details for each one of them are given in the following tables:

- Input/output correlation - Table 11
- Standard regression coefficients - Table 12
- Perturbation - Table 13
- Cotter - Table 14
- Morris - Table 15
- Sobol' Indices - Table 16
- PCE-based Sobol' Indices - Table 20

3.2.1 Input/Output correlation

See also Section 2.3.1.1.

Table 11: `myAnalysis.Results`

`.CorrIndices`	$(M \times N_{out})$ Double	Correlation-based sensitivity indices, according to Eq. (1.2).
`.RankCorrIndices`	$(M \times N_{out})$ Double	Rank-correlation-based sensitivity indices, according to Eq. (1.4).
`.Cost`	Double	Total model evaluations carried out by the method.
`.ExpDesign`	See Table 19	Model evaluations used during the calculation.
`.VariableNames`	$(1 \times M)$ Cell	Cell array with the names of the input variables.

3.2.2 Standard Regression Coefficients

See also Section 2.3.2.1.

Table 12: `myAnalysis.Results`		
`.SRCIndices`	$(M \times N_{out})$ Double	SRC sensitivity indices, according to Eq. (1.8).
`.SRRCIndices`	$(M \times N_{out})$ Double	Rank-correlation-based sensitivity indices, according to Eq. (1.11).
`.Cost`	Double	Total model evaluations carried out by the method.
`.ExpDesign`	See Table 19	Model evaluations used during the calculation.
`.VariableNames`	$(1 \times M)$ Cell	Cell array with the names of the input variables.

3.2.3 Perturbation Method

See also Section 2.3.3.1.

Table 13: `myAnalysis.Results`		
`.Mu`	Double	First order estimate of the model mean.
`.Var`	Double	First order estimate of the model variance.
`.Sensitivity`	$(M \times N_{out})$ Double	Sensitivity estimates of the perturbation method, from equation (1.15).
`.Cost`	Double	Total model evaluations carried out by the method.
`.ExpDesign`	See Table 19	If `SOpts.SaveEvaluations` was set to true, the model evaluations are saved in this struct.
`.VariableNames`	$(1 \times M)$ Cell	Cell array with the names of the input variables.

3.2.4 Cotter Method

See also Section 2.3.4.1.

Table 14: `myAnalysis.Results`		
`.CotterIndices`	$(M \times N_{out})$ Double	Vector containing the value of the Cotter sensitivity measure for each of the factors from equation (1.16).
`.EvenOrder`	$(M \times N_{out})$ Double	Vector containing the value of the even order effect for each of the factors.
`.OddOrder`	$(M \times N_{out})$ Double	Vector containing the value of the odd order effect for each of the factors.
`.Cost`	Double	Total model evaluations carried out by the method.
`.ExpDesign`	See Table 19	If `SOpts.SaveEvaluations` was set to true, the model evaluations are saved in this struct.
`.VariableNames`	$(1 \times M)$ Cell	Cell array with the names of the input variables.

3.2.5 Morris Method

See also Section 2.3.5.1.

Table 15: `myAnalysis.Results`		
`.Mu`	$(M \times N_{out})$ Double	Original Morris sensitivity measure, μ, mean of the elementary effects.
`.MuStar`	$(M \times N_{out})$ Double	Improved Morris sensitivity measure, μ^* (Eq. (1.25)).
`.Std`	$(M \times N_{out})$ Double	Standard deviation of the elementary effects.
`.Cost`	Double	Total model evaluations performed.
`.ExpDesign`	See Table 19	Model evaluations used for the estimation (if `SOpts.SaveEvaluations=1`).
`.VariableNames`	$(1 \times M)$ Cell	Cell array with the names of the input variables.

3.2.6 Sobol' Indices

See also Section 2.3.6.1.

Table 16: `myAnalysis.Results`

`.Total`	$(M \times N_{out})$ Double	Array of total Sobol' indices. Each column corresponds to an output variable.
`.FirstOrder`	$(M \times N_{out})$ Double	Array of first order Sobol' indices. Each column corresponds to an output variable.
`.AllOrders`	Cell	Each element `AllOrders{i}` is a $(M_i \times N_{out})$ array of i-th order Sobol' indices for each of the J outputs.
`.VarIdx`	Cell	Each element `VarIdx{i}` is a $(M_i \times i)$ array with interactions considered in the Sobol' indices in `AllOrders{i}`. For Example, if `VarIdx{2} = [1 2]`, then `AllOrders{2}` contains the second order Sobol' index $S_{1,2}$ (see Eq. (1.37)).
`.TotalVariance`	$(1 \times N_{out})$ Double	Total variance of each output of the model.
`.Cost`	Double	Total model evaluations carried out by the method.
`.Bootstrap`	See Table 17	Results of the bootstrap method.
`.ExpDesign`	See Table 19	If `SOpts.SaveEvaluations` is `true`, the model evaluations are saved in this Struct.
`.VariableNames`	$(1 \times M)$ Cell	Cell array with the names of the input variables.
`.PCEBased`	Logical	If true, the indices are analytical for a PCE meta-model, otherwise they are Monte Carlo estimates.

3.2.6.1 Bootstrap

See also Section 2.3.6.1.

Table 17: `myAnalysis.Results.Bootstrap`		
`.Total`	See Table 18	Struct containing the bootstrap information for the total Sobol' indices.
`.FirstOrder`	See Table 18	Struct containing the bootstrap information for the first order Sobol' indices.
`.AllOrders`	See Table 18	Cell array containing the bootstrap information for all the calculated Sobol' indices, grouped by order. Each element `AllOrders{i}` refers the i-th order indices.

Table 18: `myAnalysis.Results.Bootstrap.Total/FirstOrder/AllOrders{i}`		
`.CI`	$(N_{ind} \times N_{out} \times 2)$ Double	Upper and lower bounds of the confidence interval for each of the N_{ind} indices and N_{out} output variables.
`.ConfLevel`	$(N_{ind} \times N_{out})$ Double	Confidence level for each of the N_{ind} indices and N_{out} output variables.
`.Mean`	$(N_{ind} \times N_{out})$ Double	Means of the bootstrap estimates for each of the N_{ind} indices and N_{out} output variables.

3.2.6.2 Model Evaluations

If the `SOpts.SaveEvaluations` option is set to `true`, the model evaluations will be saved in the `myAnalysis.Results.ExpDesign` structure detailed in Table 19.

Table 19: `myAnalysis.Results.ExpDesign`		
`.X`	$(N \times M)$ Double	Coordinates of the experimental design.
`.Y`	$(N \times N_{out})$ Double	Model responses.
`.Shuffled(ii)`	Struct	Experimental design for higher order indices. (Monte Carlo estimators only.)

3.2.7 PCE-based Sobol' Indices

See also Section 2.3.7.1.

Note: this table is essentially identical to Table 16, with the exclusion of the model-evaluation related fields (`Cost`, `ExpDesign` and `Bootstrap`).

Table 20: `myAnalysis.Results`

`.Total`	$(M \times N_{out})$ Double	Array of total Sobol' indices. Each column corresponds to an output variable.
`.FirstOrder`	$(M \times N_{out})$ Double	Array of first order Sobol' indices. Each column corresponds to an output variable.
`.AllOrders`	Cell	Each element `AllOrders{i}` is a $(M_i \times N_{out})$ array of i-th order Sobol' indices for each of the J outputs.
`.VarIdx`	Cell	Each element `VarIdx{i}` is a $(M_i \times i)$ array with interactions considered in the Sobol' indices in `AllOrders{i}`. For Example, if `VarIdx{2} = [1 2]`, then `AllOrders{2}` contains the second order Sobol' index $S_{1,2}$ (see Eq. (1.37)).
`.TotalVariance`	$(1 \times N_{out})$ Double	Total variance of each output of the model.
`.VariableNames`	$(1 \times M)$ Cell	Cell array with the names of the input variables.
`.PCEBased`	Logical	If true, the indices are analytical from the PCE coefficients, otherwise they are Monte Carlo estimates.

3.3 Printing/Visualizing of the results

UQLAB offers two commands to conveniently print reports containing contextually relevant information for a given object:

3.3.1 Printing the results: uq_print

Syntax

```
uq_print(myAnalysis);
uq_print(myAnalysis,outidx);
```

Description

`uq_print(myAnalysis)` print a report on the results of the sensitivity analysis in object `myAnalysis`. If the model has multiple outputs, only the results for the first output variable are printed.

`uq_print(myAnalysis, outidx)` print a report on the results of the sensitivity analysis in object `myAnalysis` for the output variables specified in the array `outidx`.

Examples:

`uq_print(myAnalysis, [1 3])` will print the sensitivity analysis results for output variables 1 and 3.

3.3.2 Graphically display the results: uq_display

Syntax

```
uq_display(myAnalysis);
uq_display(myAnalysis, outidx);
```

Description

`uq_display(myAnalysis)` create a visualization of the results of the sensitivity analysis in object `myAnalysis`, if possible. If the model has multiple outputs, only the results for the first output variable are visualized.

`uq_display(myAnalysis, outidx)` create a visualization of the results of the sensitivity analysis in object `myAnalysis` for the output variables specified in the array `outidx`.

Examples:

`uq_display(myAnalysis, [1 3])` will display the sensitivity analysis results for output variables 1 and 3.
See also the examples in Part 2.

References

Campolongo, F., J. Cariboni, and A. Saltelli (2007). An effective screening design for sensitivity analysis of large models. *Environmental modelling & software 22*(10), 1509–1518. 7

Cotter, S. (1979). A screening design for factorial experiments with interactions. *Biometrika 0*, 317–320. 5

Efron, B. (1979). Bootstrap methods: another look at the Jackknife. *Ann. Stat. 7*(1), 1–26. 11

Janon, A., T. Klein, A. Lagnoux, M. Nodet, and C. Prieur (2013). Asymptotic normality and efficiency of two Sobol index estimators. *ESAIM: Probability and Statistics*, 1–20. 10, 11, 39

Morris, M. D. (1991). Factorial Sampling Plans for Preliminary Computational Experiments. *Technometrics 33*(2), 161–174. 5, 7

Saltelli, A. and T. Homma (1996). Importance measures in global sensitivity analysis of model output. *Reliab. Eng. Sys. Safety 52*, 1–17. 11, 39

Sobol', I. M. (1993). Sensitivity estimates for nonlinear mathematical models. *Math. Modeling & Comp. Exp. 1*, 407–414. 8, 9, 39

Sudret, B. (2008). Global sensitivity analysis using polynomial chaos expansions. *Reliab. Eng. Sys. Saf. 93*, 964–979. 12