Methods for Uncertainty and Sensitivity Analysis. Review and recommendations for implementation in Ecolego

António Pereira
Robert Broed

Dept. of Physics
AlbaNova University Center
Stockholm Center of Physics, Astronomy and Biotechnology
106 91, Stockholm, Sweden

January 2006
Abstract

In this paper we review several uncertainty and sensitivity methods available in the literature and provide recommendations for implementing some of these methods in Ecolego.

Predictive models used in the assessment of protection of the environment from ionising radiation present some extreme characteristics that make uncertainty and sensitivity analysis a non-trivial task. Those characteristics constrain the number and type of methods that are useful. The uncertainty methods selected in this work are based in Monte Carlo techniques or in other words they are probabilistic methods. These are found superior to deterministic methods in the context of radiological risk assessments and environmental protection. The Simple Monte Carlo sampling and the Latin Hypercube sampling were the sampling approaches selected to make uncertainty analysis. In respect to sensitivity analysis, the global methods were preferred over the deterministic or local methods. Global sensitivity analyses (GSA) are sampled-based analyses and again the Simple Monte Carlo and the Latin Hypercube techniques are necessary to allow their implementation in Ecolego.

MatLab codes of the recommended methods were delivered 2004, in a CD to the Swedish Radiation Protection Authority (SSI).
Contents

1. Introduction ................................................................. 7
2. Aim and scope of the work ............................................. 9
3. What is uncertainty and sensitivity of model output? .......... 11
4. Uncertainty analysis ..................................................... 13
   4.1 Simple Random Sampling .......................................... 14
   4.2 Latin Hypercube sampling ......................................... 14
   4.3 Latin Hypercube versus Simple Monte Carlo sampling .... 16
5. Introducing correlations between parameters ..................... 19
   5.1 The Iman and Conover method of rank correlations .......... 19
   5.2 Other methods for generation of parameter correlations .... 21
6. Sensitivity analysis ...................................................... 23
   6.1 Introduction .......................................................... 23
   6.2 Qualitative tests .................................................... 24
   6.3 Quantitative tests ................................................... 24
      6.3.1 Tests based on correlation analysis ......................... 24
      6.3.2 Tests based on regression analysis ......................... 26
      6.3.3 Tests based on partitioning of output data .............. 28
      6.3.4 Sobol sensitivity analysis, FAST and Extended FAST .. 29
7. Summary and Recommendations ...................................... 33
REFERENCES .................................................................... 35
Appendix I ...................................................................... 37
Appendix II ..................................................................... 41
1. Introduction

Environmental models are aimed in general at studying the impact of natural or man-made systems on man and other biota. The goal is then to examine potential adverse effects and to extract risk indicators needed in decision making. The predictions obtained from long-term environmental models are usually flawed by large uncertainties making the analysis of the consequences non-trivial.

This work is focused in uncertainty and sensitivity analysis of results obtained by models of contaminant migration.

There is a vast literature on the application of uncertainty and sensitivity analysis (UA/SA) to a variety of disciplines and fields such as medicine, biology, physics, chemistry, weather forecasting, ecology, etc. The methods of UA/SA share a common base but the specific language of each field and the adaptation of the methods to particular cases make it difficult to exploit progress coming from the different disciplines. Since 1995 an effort has been done to join specialists and practitioners in sensitivity analysis in the framework of a conference series, SAMO (Sensitivity Analysis of Model Output). The reader may consult the proceedings of the SAMO series to get a general view on ongoing progress in the field starting with the first conference (Saltelli and Maravic, Edts. 1996)
2. Aim and scope of the work

The aim of this report is to provide a summary view of what we find to be the most useful methods of uncertainty and sensitivity methods to be implemented in Ecolego, a code for radioecological risk assessment (Avila et. al., 2003).

In this work we focus on some methods of uncertainty and sensitivity analysis (and in particular on parameter-driven UA/Sa). The extreme characteristics of the models for migration of radionuclides in the environment pose constrains on the methods of UA&SA that are useful for use in Ecolego, namely:

- The models are in general complex (results being non-monotonic and non-linear)
- A huge number of input parameters may be used in these models
- Input parameters are associated with large uncertainties (wide range of parameter variations)
- Model parameters are often correlated to each other
- Uncertainties other than parameter-driven, may be present in system modelling (scenario uncertainty, conceptual uncertainty, etc.)
- Spatial variability and time-dependence of parameters need also be taken into account.
- Model predictions may cover huge periods (from ten thousands years to hundred thousands years or more.
3. What is uncertainty and sensitivity of model output?

The input parameters of a model are always affected by uncertainties coming from different sources. For instance, the value of the sorption coefficient of nuclide is not known exactly; in fact only a given range or perhaps some “best” estimators may be specified with good certainty. We say that this parameter has an uncertainty and this uncertainty at the input of the model will propagate through it to the output; therefore the output is also affected by an uncertainty. Models have in general several (many) input parameters that are uncertain and those uncertainties will propagate through the models and affect the output uncertainty. This type of uncertainty is called parameter-driven uncertainty and it is this one (and the related parameter sensitivities) that we will address in this work. There are other types of uncertainty such as conceptual- or model uncertainty and scenario uncertainty that also affect the system predictions. One may also find in the literature other ways of classifying those uncertainties, for example in epistemic and aleatoric uncertainties (Helton, 1994, Hoffman and Hammonds, 1994, Helton and Burmaster, 1996). Obviously if the goal is a total integrated analysis of a system for licensing purposes, the complete spectrum of uncertainties must be taken into account in the uncertainty analysis.

In the context of this work the goal of uncertainty analysis is to ascertain and to quantify the uncertainty in the output results as a consequence of the uncertainty in the input data to the model (models)\(^1\).

Uncertainty analysis is very often followed by sensitivity analysis. It is not unusual that confusion arises between the two analyses. The goal of (parameter-driven) sensitivity analysis is to ascertain which are the model parameters that are important and what is their relative importance. For instance, is parameter A more important than parameter B and this one more important than parameter C? In what order can we rank the parameters according to their importance because of their impact on the output and which parameters have an impact on the output that is so small that we could keep them constant during the whole model calculations? Can we draw conclusions about where to put effort or resources if we need to reduce the total uncertainty of the system predictions or make other priorities?

---

\(^1\) For the application of uncertainty analysis including scenario and model uncertainty, the reader should consult Draper et. al., 1999.
4. Uncertainty analysis

The Monte Carlo method is widely used to study uncertainty analysis of model outputs. In a few words the Monte Carlo method implies that the input parameters of the model are sampled from probability density functions. The variance of the probability density functions of each parameter expresses our uncertainty on the respective input parameter. The model is run in a sequential way tens, hundreds or thousands of times with different sets of sampled parameters each time. It is also possible to run the model in a parallel fashion to speed up the calculations using dedicated tools (Pereira et al., 1997, Mendes and Pereira, 2003). The output is anyhow made of many realisations, and is analysed using statistical methods. Figure 1 shows the modelling of the system with the propagation of the parameter uncertainties.

The primary steps in conducting an uncertainty analysis of parameter uncertainties are:

- list all parameters that can potentially contribute to uncertainty of the model predictions
- identify the underlying parameter distributions either through expert judgement or based on empirical evidence. The mean, standard deviation and ranges of those distributions are ascertained throughout this process
- take into account possible correlations between the varying parameters
- propagate the uncertainties of the input parameters through the model (models): perform probabilistic simulations by sampling the parameters from the distributions to use them as input to the models
- extract the central measures of tendency from the output distribution of the predicted values (mean, standard deviation, etc.), the confidence limits and the intervals in order to quantify the uncertainty on the model prediction.

There are several sampling techniques to extract parameters from probabilistic distributions such as the Simple Monte Carlo sampling (SMC), the Latin Hypercube sampling (LHS) and the Importance sampling (IS). Other statistical techniques for computer experiments are factorial designs such as the complete factorial design and the fractional factorial design. In the next section we describe briefly the two sampling techniques that we recommend be implemented in Ecolego.

![Figure 1](image)

*Figure 1* – The Monte Carlo method. The output of the model is also a probability distribution. The uncertainties are visualised by the arrows of the distributions representing the variances of those PDFs.
**4.1 Simple Random Sampling**

Simple Random Sampling (SRS) also called simple or crude Monte Carlo sampling is the straightforward way to perform calculations to study the propagation of uncertainties associated to the input parameters (parameter-driven uncertainties) through the model and to the output. The parameters are sampled independently from the respective probability density functions. It is also possible to introduce pair-wise correlations between parameters although in this case the sampling design is not any more a random sampling in the true sense of the word.

**4.2 Latin Hypercube sampling**

Latin Hypercube sampling (LHS) is an alternative to simple random sampling (SRS) that is used whenever one needs to reduce the number of samples in large simulations. It gives “good” estimators of model response.

The LHS method (Iman et. al., 1980, Iman and Shortencarier, 1984) is similar to certain fractional designs such as Latin squares.

We follow below the description (Liebetrau and Doctor, 1987) of the original LHS approach. The idea of LHS is to divide the range of each input variable (parameter) into a certain number of equiprobable intervals. Sampling the parameters from these intervals makes it possible to cover the parameter space in an effective way. This makes it also possible to sample from the tails of the distributions accelerating the convergence of the calculations if the output is sensitive to those tails.

Assume that each of the k components \( X_j \) of the vector \( X \) is partitioned in \( N \) intervals. Denote these intervals as \( I_{ij}, I_{2j} \ldots I_{Nj} \) for \( j=1,2, \ldots k \) (k is the number of parameters). For each parameter \( X_j \) the interval is divided so that

1. It partitions the range of \( X \) in \( N \) intervals
2. \( P_{ij} = \text{Prob}\{ X_j \in I_{ij} \} = 1/N \) \( i=1,2,\ldots N \)

The set of all Cartesian products created from the intervals is:

\[ \{I_{n1} \otimes I_{n2} \otimes \ldots \otimes I_{nk} : n_j = 1,2, \ldots N; j=1,2 \ldots k \} \]

This set maps a partition \( P \) of the parameter-input space into \( N^k \) cells. The “location” of each cell in \( P \) is identified by the coordinate vector \( n = (n_1, n_2, \ldots, n_k) \). A LHS design of size \( N \) consists of \( N \) cells randomly selected from \( P \). This selection is done by taking randomly one of the \( N! \) permutations of the integers \( \{1, 2, \ldots, N\} \) which is written in the first column of the \( n \times K \) “design” matrix \( D \). A second permutation is then generated and written in the second column, etc. The vector of integers in each row of the “design” matrix \( D \) identifies a cell in \( P \). The \( N \) cells identified by \( D \) is the LHS design.

Figure 2 shows a \( k=2 \) and \( N=6 \) (2 parameters divided into 6 equiprobable intervals) generation of two possible LHS designs. After we have obtained the “design” \( D \), we start sampling the input vector \( X \) in each of the \( N \) cells identified by the design. For each variable \( X_j \) (in our example the two variables \( X_1 \) and \( X_2 \)) we proceed in the following way:
• One observation is generated in each of the \( N (= 6) \) intervals \( I_{ij} \) that partitions the range of \( X_j \)
• The \( N \) observations of \( X_j \) are arranged in column \( j \) in the sample matrix \( X_D \) so that the ranks are identified in the column \( j \) of the “design” matrix \( D \)

The algorithm described above has been implemented as a Fortran program (Iman and Shortencarier, 1984).

![Diagram of Latin Hypercube approach for sampling parameters from probability distributions.](image)

**Figure 2** The Latin Hypercube approach for sampling parameters from probability distributions.

Correlations can be introduced in LHS between ranks of parameters using the Iman and Conover method (section 5.3). We recommend to use in Ecolego the modern version of that implementation (Wyss, 1998). This one is based on the division of the PDFs in a number of equiprobable intervals that are equal to the number of the realisations used in the Monte Carlo calculations. If the user decides, for instance, to make 300 LHS realisations, all PDFs used to sample the parameters will be divided in 300 intervals and every parameter is sampled exactly once from each interval.
In general it is a good practice to check that spurious correlations are not introduced during the LHS sampling procedure. One should also observe, that the formulas used to compute some distribution estimators differ from the classical ones used in SRS.

An advantage of the SRS is that it obtains unbiased estimators and the statistics in the analysis of the results is straightforward. The number of realisations (samples) may be high if there are many variable parameters making the calculations time consuming. Obviously it depends also on the complexity of the model.

Due to the easy access to modern and fast computers, some researchers claim that, for high number of realisations the advantages of the reduction of the number of realisations obtained by LHS does not compensate the complexity in the sampling and in the analysis.

4.3 Latin Hypercube versus Simple Monte Carlo sampling

The advantage of the LHS versus SRS is that a lower number of realisations are needed to obtain a reasonable convergence of the results.

To illustrate the relationship between number of probabilistic realisations, the convergence of the mean and standard deviation, and the choice of sampling technique (LHS and SRS), a relatively simple example was tested by using Ecolego. The example model consists of a few compartments (accumulators), where the transfer to the last compartment is dependent on the only stochastic parameter used in the model. This parameter is assigned a normal distribution. A number of simulations were performed to study the relationship between $N$, the number of realisations, and the choice of sampling method. Since the parameter in question is assigned a symmetric normal distribution, the output that we study will also be symmetric (because of its linear dependency of the parameter). Therefore we can use the mean and standard deviations as direct indicators of the rate of convergence of the two sampling methods.

The number of realisations for each sampling technique was:

$N = 20, 50, 100, 250, 500, 750, 1000, 2000, 3000$, and finally $5000$

In the two plots (Fig.3 and Fig. 4), the mean and standard deviation is shown respectively, and it is clearly seen that the use of SRS requires a larger number of realisations in order to get the mean and standard deviations to converge. The mean value using LHS ($5.36 \times 10^9$), converges at 100 samples while in the case of SRS we observe an oscillatory behaviour around that value. The standard deviation ($2.05 \times 10^9$) converges even more rapidly for LHS.
**Figure 3** Mean vs. number of realisations, N.

**Figure 4** Standard deviation vs. number of realisations, N.
<table>
<thead>
<tr>
<th>N</th>
<th>Mean (SRS)</th>
<th>Mean (LHS)</th>
<th>Std (SRS)</th>
<th>Std (LHS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>4.66 x 10^9</td>
<td>5.29 x 10^9</td>
<td>1.96 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
<tr>
<td>50</td>
<td>5.45 x 10^9</td>
<td>5.37 x 10^9</td>
<td>2.36 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
<tr>
<td>100</td>
<td>5.33 x 10^9</td>
<td>5.36 x 10^9</td>
<td>2.13 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
<tr>
<td>250</td>
<td>5.42 x 10^9</td>
<td>5.36 x 10^9</td>
<td>1.92 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
<tr>
<td>500</td>
<td>5.21 x 10^9</td>
<td>5.36 x 10^9</td>
<td>2.03 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
<tr>
<td>750</td>
<td>5.32 x 10^9</td>
<td>5.36 x 10^9</td>
<td>2.02 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
<tr>
<td>1000</td>
<td>5.39 x 10^9</td>
<td>5.36 x 10^9</td>
<td>2.07 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
<tr>
<td>2000</td>
<td>5.34 x 10^9</td>
<td>5.36 x 10^9</td>
<td>2.04 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
<tr>
<td>3000</td>
<td>5.37 x 10^9</td>
<td>5.36 x 10^9</td>
<td>2.10 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
<tr>
<td>5000</td>
<td>5.33 x 10^9</td>
<td>5.36 x 10^9</td>
<td>2.06 x 10^9</td>
<td>2.05 x 10^9</td>
</tr>
</tbody>
</table>

However there are situations in which the statistical indicators obtained using LHS do not converge more rapidly than those obtained with SRS. It depends of the model and input parameters. But in general, if low probability-high consequence tails in the input distributions are present, LHS is superior then SRS.
5. Introducing correlations between parameters

When we use the SRS Monte Carlo method we assume that the parameters are independent from each other and therefore the probability distributions from which they are sampled should be uncorrelated. However, in nature, many phenomena are correlated to others and therefore it may be necessary to introduce pairing correlations between model parameters. The method used for this purpose is non-parametric (Iman and Conover, 1982).

5.1 The Iman and Conover method of rank correlations

It is not trivial to introduce correlations between parameters either than in the special case of Gaussian distributions. Iman and Conover developed a method that is based on the ranks of the variables instead of using the values of the variables. Substituting the values by their ranks makes the numeric method easier to handle than is the case otherwise, because one needs to make certain operations like inversion of matrices of the parameters values. The reason is that the physical parameters can assume very high values and very small values. The correlation coefficient obtained by using ranks is in general a very good approximation of the Pearson correlation coefficient (this last one is computed based on the values, not on the ranks).

In the following we describe the algorithm used by Iman and Conover:

1. Generate N samples (for example N=50) of a matrix $X$, with k parameters (for example k=4)

$$
X = \begin{bmatrix}
X_1 \\
X_2 \\
\vdots \\
X_N
\end{bmatrix} = \begin{bmatrix}
X_{11} & X_{12} & \ldots & X_{1k} \\
X_{21} & X_{22} & \ldots & X_{2k} \\
\vdots & \vdots & \ddots & \vdots \\
X_{N1} & X_{N2} & \ldots & X_{Nk}
\end{bmatrix}
$$

2. Construct matrix $R$

$$
R = \begin{bmatrix}
R_1 \\
R_2 \\
\vdots \\
R_N
\end{bmatrix} = \begin{bmatrix}
R_{11} & R_{12} & \ldots & R_{1k} \\
R_{21} & R_{22} & \ldots & R_{2k} \\
\vdots & \vdots & \ddots & \vdots \\
R_{N1} & R_{N2} & \ldots & R_{Nk}
\end{bmatrix}
$$

$R_{ij}$ are the ranks of $X_{ij}$ from step 1.

3. Read the correlation matrix $C$ that the user gives as input. We assume as an example the following matrix (with 4 parameters that are correlated pairwise: p1 with p2 and p3 with p4)
4. Compute the Choleski decomposition of $C$ to obtain $P$.

$$C = P \, P'$$

$P'$ is the transpose of $P$

5. Compute the van der Warden score matrix $A$

$$a_{ij} = \vartheta^{-1} \left[ R_{ij} / (50 + 1) \right]$$

6. Compute the matrix $U$ which is the correlation matrix of the score matrix $A$. How to do it? Take the correlations between the columns of $A$ (Pearsson correlations).

7. Decompose the matrix $U$ above. Use Choleski decomposition to get matrix $Q$. $U$ gives $Q$ and $Q'$ (the transpose of $Q$).

8. We have now the matrix $P$ (see step 4 above) and $Q^{-1}$ (equal to the transpose of $Q$, see step 7 above). Then we can compute the matrix $T$ as :

$$T = P \, Q^{-1}$$
9. Compute the transpose score matrix $A^*$:

$$A^* = A^T \quad (A \text{ comes from step 5 and } T \text{ from step 8})$$

10. Compute $R^*$ by replacing the elements of $A^*$ by their ranks

11. Compute the final matrix $X^*$ by rearranging $X$ using the ranks of $R^*$

Matrix $R^*$ should have the desired correlations between the columns. Check it by computing the correlations of the columns of $X^*$.

### 5.2 Other methods for generation of parameter correlations

It is not trivial to generate input parameters sampled from arbitrary probability density functions that are correlated to each other according to predefined correlation values except for the multivariate normal distributions where a theoretical solution is available. This is why the method for rank correlations proposed by Iman and Conover has become so popular.

We describe briefly the procedure to obtain multivariate normal distributions with specified correlation between the marginal distributions (see also Appendix I).

We start from a vector of standard normal random numbers. They can be transformed to the desired form using

$$x = R^T z + \mu$$

In the equation above $\mu$ is a $d \times 1$ vector representing the mean, $z$ is a $d \times 1$ vector of standard normal random numbers and $R$ is a $d \times d$ matrix such that $R^T R = \Sigma$. The matrix $R$ can be obtained by Cholesky factorisation of the covariance matrix $\Sigma$.

Suppose that we want to generate two normal distributions with mean values $-2$ and $4$ and a correlation equal to $0.8$. We use the transpose of the above equation to obtain our two normal distributions with the desired means, standard deviations and correlations:

$$X = Z R^T + \mu$$

where $X$ is an $n \times d$ matrix of the $d$-dimensional random variables and $Z$ is an $n \times d$ matrix of standard random variables. The number of random numbers can be for instance $n=1000$. Denote the mean values by the vector $\mu^T = (-2, 4)$ where $T$ is the transpose operation and where $\Sigma$ is the covariance matrix:

$$\Sigma = \begin{pmatrix} 1 & 0.8 \\ 0.8 & 1 \end{pmatrix}$$
The correlation between the two normal distributions is between values (Pearson correlation) and not between ranks (Spearman correlation). If the distributions are non-normal, for example if we want to obtain samples from a triangular and a log-normal distribution with a specified correlation coefficient we have several alternatives. Either we use the Imam and Conover method if it is sufficient to use rank correlations or we may use two other methods if correlations between values are required. The first one uses a semi-analytic approach (van der Geest, 1998) and the second one, a numerical approach (Pereira and Sundström, 2000). The method of van der Geest requires the solution of integrals that may be non-trivial if the distributions are other than gaussian. The method of Pereira and Sundström is on the other hand computationally demanding. It is based on the principle of entropy and therefore the solver is called correlation entropy solver. The idea is extremely simple and the algorithm can be described as follows:

- Generate your first distribution (the triangular one in our example)
- Generate the second distribution (the log-normal)
- If the desired correlation is positive, order the parameters of both distributions in ascending order. In this way you get a Pearson correlation that is almost equal to +1. If the desired correlation is negative, order the values of one distribution in ascending order and the other in descending order. If you compute the Pearson correlation value you get a coefficient that is almost –1.
- Suppose now that you want a correlation value equal to +0.8. After sorting the two distributions in ascending order, one starts destroying the ordering of the numbers in the distributions by permutation between values within each of them. In this way you increase the entropy (the degree of disorder) and the correlation value decreases from +1 to a lower value, for example to 0.9. Compute the Pearson correlation and if it is not yet 0.8, continue the permutation of parameter values incrementally until you get the desired correlation of 0.8. You specify the error of the correlation by stopping your process when the correlation value falls within an acceptable error band around 0.8.

If the correlation is between several distributions the computational burden of this process increases as it also does for the method of Iman and Conover.
6. Sensitivity analysis

6.1 Introduction

Approaches to sensitivity analysis can be divided into two classes: deterministic and probabilistic sometimes also called local and global sensitivity analyses respectively.

If the model is too complex to be run in a Monte Carlo fashion, then a deterministic approach to sensitivity studies is more common. One may run the model a few times with different parameter combinations varying one at a time for a crude analysis of their impact on the output, or one may use adjoint methods to study the impact of the parameter space through examination of the derivatives of those parameters. In this case, it is possible to obtain simultaneously the results and the influence of the parameters quantified by the information given by the partial derivatives. This can be done for example with the help of ADIFOR a computer code in Fortran 77 (Bischof et. al., 1992). The disadvantage of this method, and of all local methods, is just that it gives only “local information” around a given parameter or set of parameters. Input parameters of environmental models have often a wide range of variation (may be of orders of magnitude). Therefore, the obvious question is - which parameter values shall we choose as a reference? Shall we take the derivatives around the mean of the input parameters or of their mode, or shall we use the values in the middle of the interval of variation for each parameter or around any other value?

The construction of a response surface function of a deterministic model (RS method) may also be useful to study local variations due to input parameters. If a sufficient number of runs have been done with different parameter combinations it is possible to construct a response surface and eventually to approximate it with some fitting procedure. This response function can then be analysed with respect to the input parameters used in the fit to study their impact on it.

Probabilistic methods often require simplified models that can used in the context of Monte Carlo simulations. In this last case the parameter values are sampled by probability density functions (PDFs) according to a predefined sampling strategy. After the Monte Carlo simulations have been performed using one of the sampling methods mentioned in the previous section (SRS, LHS, IP, etc.) we can proceed with the post-processing analysis.

One may ask:

a) what is the contribution of the variance of each input distribution to the variance of the output distribution if we adopt a variance-based approach to sensitivity analysis,
b) if a given amount of variation of one input parameter influences strongly the output,
c) which parameter is the one that has the highest influence,
d) how to rank the input parameters in regard to their impact on the output,
e) if there are parameters that do not impact significantly on the output,
f) if it is possible to reduce the output uncertainty by gathering more information on important parameters,
g) Is it possible to disentangle the influence of correlated parameters on the output PDF?

There are several statistical tests to identify the most important parameters and how to rank them. We will look into some of the most commonly used, their advantages and
6.2 Qualitative tests

An analysis should start with an overview of the input distributions, calculations and results. This is done through exploratory data analysis (EDA). Graphics are widely used in EDA. Scatter plots between input parameters tell us for example if some spurious correlations have been introduced during the sampling of parameters from PDFs. Although unusual, this may happen in some cases when using Latin Hypersquare sampling which would require the repetition of the Monte Carlo calculations. Scatter plots between input parameters and outputs are also useful to get an idea of the relationship between a parameter and the output. Is it so, that when a parameter increases in value the output value also increases; or does it decrease? A qualitative picture of the correlation between inputs and output is obtained through those plots. Frequency histograms or density histograms (the previous histograms normalised to unity) give an idea of the shape of the output PDF and if they have long tails. The analyst should also check the histograms of the input variables to get an idea of how well they represent the intended distributions. Is the number of simulations sufficient to avoid discrepancy between the expectation value of the theoretical distribution of a parameter and the mean value obtained from the actual sampling? Cluster plots (a special type of scatter plots) can indicate aggregation of releases for certain parameter values: surface plots, contour plots, bivariate histograms, 3D scatter plots, cumulative plots, scatter matrix plots and histogram matrix plots may also give a rapid overview of the calculation results. The previous qualitative tests and the summary statistics of the output distribution (mean, median, minimum, maximum, standard deviation, etc.) complete the EDA. We recommend to include in Ecolego several of these graphical capabilities.

6.3 Quantitative tests

To perform quantitative tests several statistical tools can be used:

a) correlation analysis (Pearson correlation coefficients, Spearman correlation coefficients, Partial correlation coefficients, Partial rank correlation coefficients),

b) regression analysis (standardised regression coefficients, standardised rank regression coefficients),

c) partitioning of output data (Smirnov, Mann-Witney, Cramér-von Mises and Square rank tests),

d) Other variance based methods as Sobol indices, FAST and Extended FAST.

A more comprehensive description of methods in a) b) and c) may be found in the literature (Conover, 1980, Iman and Helton, 1985, Bradley, 1980, Iman and Shortcarrier, 1984). For information on the variance-based methods mentioned in d), the reader should consult Saltelli et. al,1997. Below we provide a description of the main principles behind the above methods.

6.3.1 Tests based on correlation analysis

The Pearson correlation coefficient is widely used. It is a measure of association between each input considered separately and the output. It works directly on the values of the parameters and of the output. It assumes a linear relationship between input and output and it cannot take into account that the actual parameter may be correlated to another parameter. So
the value given by the linear correlation coefficient or Pearson coefficient $r$ is a first order (linear) contribution of that parameter to the output variance. The fact that $r$ is very sensitive to outliers is a difficulty in the analysis of results. The correlation is given by Eq. 1

$$r = \frac{\sigma_{xy}}{\sigma_x \sigma_y} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \sqrt{\sum (y_i - \bar{y})^2}}$$  \quad i = 1, ..., N \quad (1)$$

where $N$ is the number of samples.

The Spearman rank correlation coefficient $\rho$ (Eq. 2) improves the situation by computing Eq. 1 using ranks instead of values. We obtain:

$$\rho = \frac{\sum_i (R_i - \bar{R})(S_i - \bar{S})}{\sqrt{\sum_i (R_i - \bar{R})^2} \sqrt{\sum_i (S_i - \bar{S})^2}}$$  \quad (2)$$

where $R_i$ and $S_i$ are the ranks of $x_i$ and $y_i$ respectively.

An easier way to compute the above formula is:

$$\rho = 1 - \frac{6T}{N(N^2-1)} \quad (3)$$

$$T = \sum_{i=1}^{N} \left[ R(X_i) - R(Y_i) \right]^2$$

The above formula is valid in the absence of ties. If ties exist, an equivalent formula is:

$$\rho = \frac{\sum_{i=1}^{N} R(X_i)R(Y_i) - N\left(\frac{N+1}{2}\right)^2}{\left(\sum_{i=1}^{N} R(X_i)^2 - N\left(\frac{N+1}{2}\right)^2\right)^{1/2} \left(\sum_{i=1}^{N} R(Y_i)^2 - N\left(\frac{N+1}{2}\right)^2\right)^{1/2}}$$  \quad (4)$$

To quantify our degree of confidence on the influence of a parameter we make the following hypothesis: “no correlation exists between $Y$ and $X_j$”. After computing the Spearman correlation coefficient $\rho$ using $N$ runs we compare its value with the quantiles of the Spearman test distribution. The comparison is made at a certain level of confidence ($\alpha$). The hypothesis of no correlation is rejected if $\rho$ is lower than $W(\alpha/2)$ or higher than $W(1-\alpha/2)$, where the $W$s are the quantiles of the test distribution. In other words the level of significance $\alpha$ is the probability that we reject erroneously the hypothesis or the probability that the test indicates a correlation when the input and output are actually uncorrelated. To
apply the test at a 0.05 significance level, $W(0.025)$ and $W(0.975)$ are extracted from tables or computed (Conover, 1980). For example for $N=500$ runs the Spearman quantiles are $W(0.025)=-0.088$ and $W(0.975)=0.088$. So if the Spearman correlation coefficient $\rho$ is outside the range $(-0.088, +0.088)$ the hypothesis that the variable and the output are not correlated is rejected, this is those entities are actually correlated, and the probability of an erroneous rejection when they are actually uncorrelated is 0.05.

The partial correlation coefficient (PCC) indicates the degree of linear relationship between one parameter and the output, correcting for linear effects of the other variables (parameters). In the presence of pair correlations between parameters the PCC (Eq. 5) performs better than the Pearson correlation coefficient. The partial rank correlation coefficient (PRCC) is the corresponding relation but using ranks instead of values. It is also less sensitive to outliers.

Assuming the output $Y$ and the two variables (parameters) $x_1$ and $x_2$, the partial correlation between $Y$ after removing the influence of $x_2$ from both $x_1$ and $Y$ is:

$$r_{Y_1x_2} = \frac{r_{Yx_1}r_{x_2x_1} - r_{Yx_2}r_{x_1x_2}}{\sqrt{1-r_{x_2x_1}^2} \sqrt{1-r_{x_1x_2}^2}}$$

and by the same token:

$$r_{Y_2x_1} = \frac{r_{Yx_2}r_{x_1x_2} - r_{Yx_1}r_{x_2x_1}}{\sqrt{1-r_{x_1x_2}^2} \sqrt{1-r_{x_2x_1}^2}}$$

with $r_{Yx_i}$ as the linear correlation coefficient between $Y$ and $x_i$ and $r_{Yx_2}$ and $r_{x_1x_2}$ are the analogue definitions for the corresponding entities. The partial rank correlation coefficients (PRCC) are computed as the PCC but using the ranks of the variables.

6.3.2 Tests based on regression analysis

Sensitivity analysis may be conducted using a regression model is fitted between the model output and the input parameter values. Denoting the input parameters by $x_m$ and the output of the model by $Y_j$ a multiple linear regression can be constructed from a given random sample:

$$(x_{1,j}, \ldots, x_{m,j}, Y_j) \quad size \ n (j = 1, 2, \ldots, n)$$

$$Y' = a_0 + a_1 x_1 + a_2 x_2 + \ldots + a_m x_m + \epsilon$$

$\epsilon$ is the residual and the coefficients $a_0$, $a_1$, $\ldots$, $a_m$ are obtained according to a certain criterion for example by minimising the sum of the squares of the difference between the $Y_j$’s and the mean value:

$$\Sigma (Y'_j - Y_j)^2$$
where $Y_j$ is the value estimated by the regression model. Stepwise regression is a common procedure to build a regression model to include only the statistically significant variables in the model (Draper and Smith, 1966).

How do we judge if the fitted model is adequate? The coefficient of determination $R^2$ gives us a good measure. $R^2$ indicates the proportion of the variation in $Y$ that is “explained” by the regression model.

The following identity holds

$$\frac{\sum_{k=1}^{N} (Y_k - \bar{Y})^2}{\sum_{k=1}^{N} (Y_k - \bar{Y})^2 - \sum_{k=1}^{N} (Y_k - \bar{Y}_k)^2}$$

and the ratio $R^2$ given by Eq. 6 is therefore the fraction in the uncertainty of $Y$ that can be accounted by the regression model. The coefficients $a_i, i=1,2,...,m$ of the multiple linear regression can also be interpreted as the partial derivatives of $Y$ with respect to the input parameters. They indicate therefore the change in $Y$ associated to a unit of change in a given parameter, all other remaining constant. Therefore the regression coefficients provide a measure of sensitivity of the output to changes in the input parameters.

There is a difficulty associated to certain models used, for example, in environmental physics. The values of the coefficients depend on the units of the parameters and therefore they cannot serve as a measure of their relative importance. To improve the situation it is necessary to standardise the output $Y$ and the parameters $X_j$ and this is done by taking the difference between those entities and their mean values and then by dividing that difference by the respective standard deviations:

$$Y^* = (Y - \bar{Y})/\sigma_Y$$ and $$X_j^* = (X_j - \bar{X}_j)/\sigma_j \quad j=1,2,...,m$$

In this case we get the standardised linear regression model which is more useful for sensitivity analysis than the non-standardised version. That model can be written as:

$$Y^* = a_1x_1^* + a_2x_2^* + ... + a_mx_m^*$$

The standardised regression coefficients indicate how many standard deviation changes in the output are associated with one standard deviation of an input parameter, all others remaining constant.
6.3.3 Tests based on partitioning of output data

The use of sensitivity tests based on partition of data as the Smirnov “two-sample” test (Conover, 1980) exploits the idea of dividing the sample of the parameter $X_j$ into two sub-samples according to the quantiles of the output distribution $Y$.

If the distributions of the two sub-samples mentioned above can be proved to be different then the parameter is an influential one. For the Smirnov test, one takes for example the $x_{ij}$’s corresponding to output $y_i$’s above the 90th quantile of the cumulative distribution $F(Y)$ to be one sub-sample and the remaining $x_{ij}$’s as the other sub-sample. The Smirnov measure is computed according to

$$S(Y, X_j) = \text{Max}_{X_j} \left| F_1(X_j) - F_2(X_j) \right|$$

The Smirnov measure represents the maximum vertical distance between the two cumulative curves. This test is valid if the two sub-samples are random samples, are mutually independent, the random scale is at least ordinal and the random variables are continuous.

The Cramér-von Mises method (Conover, 1980 and references therein) is similar to the Smirnov method:

$$Cr(Y, X_j) = \frac{N_1 N_2}{(N_1 + N_2)^2} \sum_{j=1}^{N_j} \left( F_1(X_j) - F_2(X_j) \right)^2$$

where the squared summation of the difference is computed at each point $x_{ij}$. This statistics depends therefore upon the total area between the distributions.

The Mann-Whitney test is also a two-sample test that looks at the differences between the means of the ranks of the $X_j$ values in the two sub-samples:

$$Mw(Y, X_j) = \sum_{i=1}^{N_1} R(x_{i,j})$$

where the summation is extended to the members of one sub-sample only. By subtracting the mean and dividing by the variance one obtains a more robust formula, because it takes the ties into account (Conover, 1980).

$$Mw(Y, X_j) = \frac{\sum_{i=1}^{N_1} R(x_{i,j}) - N_1 N + 1}{2} \left[ \frac{N_1 N_2}{N(N-1)} \sum_{i=1}^{N} R^2(x_{i,j}) - \frac{N_1 N_2 (N_1 + 1)^2}{4(N-1)} \right]^{1/2}$$

The parametric equivalent of the Mann-Whitney statistics is the Two-sample t-Test. This test assumes that the sample follows a normal distribution. One can use this test on the ranks to compute the Mann-Whitney statistics. The two-sample test statistics is computed according to
where $z_{1,i}$ and $z_{2,i}$ indicate the values of the variables $x_{ij}$ of the two sub-samples (not the ranks).

### 6.3.4 Sobol sensitivity analysis, FAST and Extended FAST

For the sake of completeness we mention three other methods that are closely related to each other: Sobol sensitivity analysis (Sobol, 1993), FAST (Fourier Amplitude Sensitivity Analysis, Saltelli et. al., 1998) and Extended FAST, Saltelli et. al., 1997. These three methods pertain to the class of variance based methods, which in turn are closely related to Fisher’s theory (1935) of Experimental Design (DOE). DOE is used in experimental work where the number of parameters is very restricted in contrast with the case of numerical experiments. Suppose that the analyst wants to design an experiment to study the influence of temperature ($T$), diffusivity ($D$) and velocity of gas flow ($V$) upon the outlet concentration from a catalytic device. Designing the experience carefully it is possible with DOE to study the influence of the three variables by themselves (main interactions) and in combination with each other (higher order interactions). In fact DOE allows to extract a first order fractional variance

\[
V\left( E(Y \mid X_i = x_i^*) \right)
\]

(12)

where

\[
E(Y \mid X_i = x_i^*)
\]

(13)

is the expectation of $Y$ conditioned on $X_i$ having a given $x_i^*$ fixed and $V$ is the variance over all possible values of $x_i^*$.

The total variance is related to the partial variances by:

\[
V(Y) = V\left( E(Y \mid X_i = x_i^*) \right) + E\left( V(Y \mid X_i = x_i^*) \right)
\]

(14)

The first term at the right hand side is used to estimate a sensitivity measure in numerical experiments. It gives the main effect term of $X_i$ on $Y$. A full decomposition of the variance in a sum of terms with increasing dimensionality includes the higher order effects:

\[
V = \sum_i V_i + \sum_{i<j} V_{ij} + \sum_{i<j<m} V_{ijm} + \ldots + V_{12...k}
\]

(15)

with $V_i$ as above (Eq.12) and

\[
V_{ij} = V\left( E(Y \mid X_i = x_i^* , X_j = x_j^* ) \right) - V\left( E(Y \mid X_i = x_i^* ) \right) - V\left( E(Y \mid X_j = x_j^* ) \right) - V\left( E(Y \mid X_i = x_i^* , X_j = x_j^* ) \right)
\]

(16)
and so on.

The Sobol sensitivity indexes make also use of variance decomposition. Sobol decomposes the function $f(X)$ where $X$ are random vectors with their elements defined in a multidimensional unit hypercube space:

$$ R^n = (x \mid 0 \leq x_i \leq 1; \ i = 1,...n) $$

$$ f(x_1,...,x_n) = f_0 + \sum_{i=1}^{n} f_i(x_i) + \sum_{1 \leq i \leq j \leq n} f_{ij}(x_i,x_j) + \ldots + f_{1,2,...n}(x_1,...x_n). $$

Sobol used a decomposition of $f$ on multiple Fourier-Haar series and he has shown that the above described decomposition is unique (Sobol, 1990). The total variance $D$ of $f(x)$ is defined as:

$$ D = \int_{R^n} f^2(x)dx - f_0^2 $$

(17)

and the partial variances as:

$$ D_{i_1...i_s} = \int_{0}^{1} ... \int_{0}^{1} f^2(x_1,...,x_s)dx_{i_1}...dx_{i_s} $$

(18)

with $1 \leq i_1 \leq \ldots \leq i_s \leq n$ and $s=1,...,n$.

Sobol shows (Sobol, 1990) that:

$$ D = \sum D_i + \sum_{1 \leq i \leq j \leq n} D_{ij} + \ldots + D_{1,2,...n} $$

(19)

The sensitivity indexes can therefore be defined as $S_{i_1,...i_s}$

$$ S_{i_1,...i_s} = \frac{D_{i_1,...i_s}}{D} $$

(20)

The sum of all terms in Eq. 20 is equal to 1.

In a few words the Sobol’s total sensitivity index $TSI$ is a sum of all sensitivity indexes that involves the actual parameter. For the case of three parameters, we have for parameter 1:

$$ TSI_1 = S_i + S_{i2} + S_{i3} $$

(21)

where $TSI_i$ is the total sensitivity index for parameter 1, $S_i$ is the first order sensitivity index for parameter 1, $S_{i2}$ is the second order sensitivity index for the couple of parameters 1 and 2, i.e. the interaction between 1 and 2 (second order interaction), $S_{i3}$ describes the interaction between 1 and 3 (second order interaction), and $S_{i23}$ the third order interaction. Sobol uses a
series of quasi-random numbers (the LPτ series) to compute the multidimensional integrals involved in the terms.

The FAST method (Saltelli et. al., 1997) uses a search algorithm for optimising the combination of the random parameters by scanning of the hypercube space using a theorem of Quantum Mechanics (Weyl, 1938). The optimisation is based on the fact that Weyl’s theorem reduces the multidimensional Monte Carlo integrals to one-dimensional. The Extended FAST method refines the FAST method and allows the computation of the first order indexes as well as the total indexes of Sobol (Saltelli et. al., 1997).

Table 3 lists the methods mentioned above, their advantages and disadvantages. We recommend the inclusion of the listed methods in Ecolego. However, there is a very limited experience, if any, in applying the last three methods (Sobol indexes, FAST and Extended FAST) to radioecological modelling, probably due to the lack of readily available codes\(^2\). Therefore one should be aware that their inclusion in Ecolego would require a considerable effort in time and resources.

\(^2\) The FAST method is implemented in SIMLAB 1.1. This software for UA/SA can be freely downloaded from http://webfarm.jrc.cec.eu.int/uasa/
<table>
<thead>
<tr>
<th>Method</th>
<th>Advantages</th>
<th>Disadvantages</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scatter plots</td>
<td>Visualises the data and uncovers non-linearities</td>
<td>Qualitative.</td>
<td>Affected by correlations</td>
</tr>
<tr>
<td>Standardised regression</td>
<td>Works for each time point</td>
<td>Difficult to cope with parameters covering orders of magnitude.</td>
<td>May be done stepwise</td>
</tr>
<tr>
<td>coefficient</td>
<td>And also for output regardless of time</td>
<td>Assume linear models.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Does not cope with non-monotonic results.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Affected by correlations.</td>
<td></td>
</tr>
<tr>
<td>Standardised rank regression</td>
<td>Works for each time point</td>
<td>Assume linear models.</td>
<td></td>
</tr>
<tr>
<td>coefficient</td>
<td>And also for output regardless of time</td>
<td>Affected by correlations.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>May be done stepwise.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>More robust to non-linear models than the previous one.</td>
<td></td>
</tr>
<tr>
<td>Pearson correlation coefficient</td>
<td>Works for each time point</td>
<td>Difficult to cope with parameters covering orders of magnitude.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>And also for output regardless of time</td>
<td>Assume linear models.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Does not cope with non-monotonic results.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Affected by correlations.</td>
<td></td>
</tr>
<tr>
<td>Spearman correlation coefficient</td>
<td>Works for each time point</td>
<td>Assumes linear models.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>And also for output regardless of time</td>
<td>Affected by correlations.</td>
<td></td>
</tr>
<tr>
<td>Partial Correlation Coefficient</td>
<td>Works for each time point and also for output regardless of time</td>
<td>Difficult to cope with parameters covering orders of magnitude.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not affected by pair-wise correlations</td>
<td>Assume linear models.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Does not cope with non-monotonic results.</td>
<td></td>
</tr>
<tr>
<td>Partial Rank Correlation</td>
<td>Works for each time point</td>
<td>Assumes linear models.</td>
<td></td>
</tr>
<tr>
<td>Coefficient</td>
<td>And also for output regardless of time</td>
<td>Does not cope with non-monotonic results.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not affected by pair-wise correlations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sobol indices, Fast and</td>
<td>Works for each time point</td>
<td>Does not cope with correlations.</td>
<td>Do not assume linear models</td>
</tr>
<tr>
<td>Extended Fast</td>
<td>And also for output regardless of time</td>
<td>Affected by correlations.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Computes first order effects</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>And total effects</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
7. Summary and Recommendations

In this work we summarise some of the most commonly used methods for UA/SA that we recommend be included in Ecolego. Sensitivity analyses is best made interactively. However, it is desirable to develop a programming tool that goes routinely through the principal steps that the analyst should run through: exploratory data analysis, correlation analysis, variance-based analysis with certain methods, etc. On the other hand, it will always exist aspects of the UA/SA analysis that are best done step by step using for example a commercial statistical package. The summary and presentation of the results is a very important aspect of the whole analysis.

Variance-based methods of sensitivity analysis are still the most powerful. To use such methods the probabilistic approach is needed. There are still difficulties in the conduction of sensitivity analysis whenever the output distributions are non-monotonic, the relationship between input and output is non-linear and correlations between parameters are present. There is today no single method that can address this issue coherently. Different methods are bounded by special assumptions and give at the end of the day results that are in general different, when it comes to the ranking of the most influential parameters. The recommended approach is therefore to use several methods before drawing any conclusions on the most influential parameters.
REFERENCES


Appendix I

Correlations and multivariate distributions

In the example below we generate a multivariate normal distribution with a specified positive correlation between the marginal distributions (equal to 0.8). It is observed that running the code three times we obtain different values for the product correlation coefficient: 0.811, 0.8161 and 0.8086. Sometimes it may be required to calculate the standard deviation of the correlation coefficient or, of any other estimator. This may be done using a bootstrap method. But it is not always possible to partition a sample. For instance sub-sample of a LHS sample is not a LHS sample in contrast with sub-samples of Simple Monte Carlo samples. In that case it is usually recommended to repeat the LHS runs several times to obtain the standard deviation of the estimator in question.

Example: Bivariate normal distribution with correlated parameters using MatLab:

```matlab
clear all
n=500;
mu=[-2,3];
covar=[1 0.8 ; 0.8 1];
d=length(mu);
% Get Cholesky factorization of covariance.
R=chol(covar);
% Generate the standard normal random variables.
Z=randn(n,d);
X=Z*R + ones(n,1)*mu;
x1=[X(1:n,1)];
x2=[X(1:n,2)];
mean(X)
corrcoef(X)
```

Results from three runs:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0000</td>
<td>0.8111</td>
</tr>
<tr>
<td>0.8111</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0000</td>
<td>0.8161</td>
</tr>
<tr>
<td>0.8161</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0000</td>
<td>0.8086</td>
</tr>
<tr>
<td>0.8086</td>
<td>1.0000</td>
</tr>
</tbody>
</table>
**The Iman and Conover method**

Suppose that we have a matrix with 3 variables and 1000 samples. In this example we use Matlab to generate such a matrix by writing \( X = \text{rand}(1000,3) \), which generates a matrix with 3 uniformly distributed variables.

**Example:**

\[
X = \begin{bmatrix}
    \text{Sample} & \text{Variable 1} & \text{Variable 2} & \text{Variable 3} \\
    1 & 0.2067 & 0.4955 & 0.4866 \\
    2 & 0.3069 & 0.6343 & 0.9638 \\
    3 & 0.8528 & 0.7366 & 0.9751 \\
    \vdots & \vdots & \vdots & \vdots \\
    N=1000 & 0.9346 & 0.7839 & 0.4449
\end{bmatrix}
\]

Check the original rank correlation between the three variables:

\[
C_{\text{original}} = \begin{bmatrix}
    1 & -0.0917 & 0.0343 \\
    -0.0917 & 1 & 0.0122 \\
    0.0343 & 0.0122 & 1
\end{bmatrix}
\]

The correlation between the variables should be near zero apart from spurious correlations that are statistically non-significant, which we is indeed the case.

Next we specify the desired rank correlation (Spearman’s correlation) that we would like to introduce between the variables. We do this by entering the correlation coefficients in a matrix form:

\[
C = \begin{bmatrix}
    1 & 0.2 & 0.7 \\
    0.2 & 1 & 0.1 \\
    0.7 & 0.1 & 1
\end{bmatrix}
\]

where the element \( C_{ij} \) corresponds to the rank correlation between variables \( i \) and \( j \).

Now we apply the Iman-Conover method by using the following function call:

\[
X_{\text{correlated}} = \text{imancon}(X,C)
\]

\( X_{\text{correlated}} \) is a matrix of the same size as \( X \), and with the same numbers in the variables as \( X \) but with the row-order permutated to achieve the desired rank correlation. We can verify the resulting Spearman rank correlation by:

\[
C_{\text{new}} = \text{spearman}(X_{\text{correlated}})
\]
which for the matrix in our example returns:

\[
C_{\text{correlated}} = \begin{bmatrix}
1 & 0.1803 & 0.6824 \\
0.1803 & 1 & 0.0871 \\
0.6824 & 0.0871 & 1
\end{bmatrix}
\]

The final matrix \( C_{\text{correlated}} \) is indeed approximately equal to \( C \).

It is worth mentioning that a requirement for the Iman-Conover method to work is that the desired correlations between variables are valid. For instance it is not theoretically possible to have a positive correlation between variables 1 and 2, and 2 and 3, if at the same time we have specified a negative correlation between variables 1 and 3. If this requirement is not met then the Cholesky-decomposition of matrix \( C \) will not be non-negative, which is a requirement for the algorithm to function.
Appendix II

Code implementation in Ecolego

In this appendix we include some of the code implemented in Ecolego during the actual phase of the project.

Pearson’s correlation coefficient

```matlab
function r=pearson(a,b);
if length(a)~=length(b)
    disp('vectors must have same length!');
else
    if size(a)~=size(b);
        b=b';
    end
end
var_a=(a-mean(a));
var_b=(b-mean(b));
r=sum(var_a.*var_b)./sqrt(sum(var_a.^2).*sum(var_b.^2));
```

Spearman rank correlation coefficient

```matlab
function r=spearman(a0,b0);
a=get_ranks(a0);
b=get_ranks(b0);
if length(a)~=length(b)
    disp('vectors must have same length!');
else
    if size(a)~=size(b);
        b=b';
    end
end
var_a=(a-mean(a));
var_b=(b-mean(b));
r=sum(var_a.*var_b)./sqrt(sum(var_a.^2).*sum(var_b.^2));
```

Iman-Conover rank correlation method

```matlab
% X = matrix, where columns are vectors with samples to apply correlation to
R=get_ranks(X); % get rank indices of values in X
% C = desired correlation matrix between columns of X
% calculate cholesky decomposition of C
P=chol(C);
% calculate van der Warden scores
N=size(X);n=N(1); % get length of input vectors
A=invnormal(R/(n+1));
U=corrcoef(A);
Q=chol(U);
Q=Q';
T=P*Q';
Ascore=A*T;
Rscore=get_ranks(Ascore);
xfinal=x(rank2ind(R,Rscore));
```
Random Generators, Simple Random Sampling

Uniform

function r=uniform(a,b,n);
    if a<b
        r=a+rand(1,n)*(b-a);
    else
        disp('b must be > a!');
    end

Example: uniform(-1,1,10000)
LogUniform*

function r=loguniform(a,b,lo,hi,n);
a=log(a);
b=log(b);
r=zeros(n,1);
if a<b
    for i=1:n
        r(i)=inf;
        while r(i)>=hi | r(i)<=lo
            r0=a+rand*(b-a);
            r(i)=exp(r0);
        end
    end
else
    disp('b must be > a!');
end
r=r';

Example: loguniform(1,10,-inf,inf,10000)
Normal

function r=normal(mu,sigma,lo,hi,n)
r=inf*ones(1,n);
if (sigma>0)
   r(1)=inf;
   for i=1:n
      while (r(i)>=hi) | (r(i)<=lo)
         r0=randn(1,1);
         r(i)=mu+sigma*r0;
      end
   end
else
   disp('b must be > 0');
end

Example: normal(0,1,-inf,inf,10000)
LogNormal**

**Note:**
Mean and standard deviation in input argument is not for logarithmic values, instead the mean and standard deviation can be calculated for the lognormal distribution by the following formulas, where $\mu$ = mean and $\sigma$ = standard deviation:

Mean: $e^{\mu + \sigma^2/2}$
Variance: $e^{(2\mu + \sigma^2)} \times (e^\sigma - 1)$

```matlab
function r=lognormal(mu,sigma,lo,hi,n)
    if (sigma>0)
        r=inf*ones(1,n);
        for i=1:n
            p=1;
            p1=p;
            p2=p;
            r2=hi;
            while (r(i)>=hi) | (r(i)<=lo)
                r0=randn(1,1);
                r(i)=exp(mu+sigma*r0);
            end
        end
    else
        disp('b must be > 0');
    end
end
```

**Example:** `lognormal(0,1,10,10000)`

![Graph of lognormal distribution](image)
Triangular

function r=triangular(Min,Max,Mode,lo,hi,n);
r=inf*ones(1,n);
if (Min<Max) & (Min<=Mode) & (Mode<=Max)
    for i=1:n
        while r(i)>=hi | r(i)<=lo
            md=(Mode-Min)/(Max-Min);
            p=rand;
            if p<md
                r(i)=Min+sqrt((Max-Min)*(Mode-Min)*p);
            else
                r(i)=Max-sqrt((Max-Min)*(Max-Mode)*(1-p));
            end
        end
    end
else
    disp('condition: Min <= Mode <= Max');
end

Example: triangular(0,10,3,-inf,inf,10000)
function r=logtriangular(MinR,MaxR,ModeR,lo,hi,n);
    MinR=log(MinR);
    MaxR=log(MaxR);
    ModeR=log(ModeR);
    r=inf*ones(1,n);
    if (MinR<MaxR) & (MinR<=ModeR) & (ModeR<=MaxR)
        for i=1:n
            while r(i)<=lo | r(i)>=hi
                md=(ModeR-MinR)/(MaxR-MinR);
                p=rand;
                if p<=md
                    r0=MinR+sqrt((MaxR-MinR)*(ModeR-MinR)*p);
                else
                    r0=MaxR-sqrt((MaxR-MinR)*(MaxR-ModeR)*(1-p));
                end
                r(i)=exp(r0);
            end
        end
    else
        disp('condition: Min <= Mode <= Max');
    end

Example: logtriangular(1,10,2,1,10,10000)
Random Generators, Latin Hypercube Sampling

Uniform

function r=lhs_uniform(a,b,n);
if a<b
   a1=linspace(a,b,n+1);
dl=(b-a)/n;
a2=a1(1:end-1);
a3=randperm(length(a2));
a2=a2(a3);
r0=rand(size(a2));
r=a2+r0*dl;
else
    disp('Condition: Min < Max');
end

Example: lhs_uniform(-1,1,10000)
LogUniform*

function r=lhs_loguniform(a,b,lo,hi,n);
if lo<a
    lo=a;
end
if hi>b
    hi=b;
end
lo=log(lo);
hi=log(hi);
if lo<hi
    a1=linspace(lo,hi,n+1);
dl=(hi-lo)/n;
a2=a1(1:end-1);
a3=randperm(length(a2));
a2=a2(a3);
r0=rand(size(a2));
r=exp(a2+r0*dl);
else
    disp('Condition: Min < Max')
end

Example: lhs_loguniform(1,10,-inf,inf,10000)
function r=lhs_normal(my,sigma,lo,hi,nr);
pt=linspace(0,1,nr+1);
pt(1)=[];
pt(end)=[];
xt=invnormal(pt,0,1);
if ~isinf(lo)
zmin=(lo-my)/sigma;
Fmin=interp1(xt,pt,zmin,'linear','extrap');
else
 Fmin=0;
end
if ~isinf(hi)
zmax=(hi-my)/sigma;
Fmax=interp1(xt,pt,zmax,'linear','extrap');
else
 Fmax=1;
end

pny=linspace(Fmin,Fmax,nr+1); % probability intervals
%pny=(Fmax-Fmin)*pny+Fmin;
n=randperm(nr); % randomly permute vector of integers from 1:nr, used for selecting bin
dp=pny(2)-pny(1); % fixed probability interval, since first index equals zero
px=pny(n)+(dp)*rand(1,nr);
r = interp1(pt,xt,px,'linear','extrap'); % use interpolation within each bin
r=sigma*r+my; % transform from standard to general normal distribution

Example: lhs_normal(0,1,-inf,inf,10000)
**LogNormal**

**Note:**
Mean and standard deviation in input argument is not for logarithmic values, instead the mean and standard deviation can be calculated for the lognormal distribution by the following formulas, where $\mu = \text{mean}$ and $\sigma = \text{standard deviation}$:

Mean: $e^{(\mu + \sigma^2/2)}$

Variance: $e^{(2\mu + \sigma^2)}(e^{\sigma^2} - 1)$

```matlab
function r = lhs_lognormal(my, sigma, lo, hi, nr);
    pt = linspace(0, 1, nr + 1);
    pt(1) = [];
    pt(end) = [];
    xt = invnormal(pt, 0, 1);
    if ~isinf(lo); lo = log(lo); zmin = (lo - my)/sigma;
        Fmin = interp1(xt, pt, zmin, 'linear', 'extrap');
    else
        Fmin = 0;
    end
    if ~isinf(hi); hi = log(hi); zmax = (hi - my)/sigma;
        Fmax = interp1(xt, pt, zmax, 'linear', 'extrap');
    else
        Fmax = 1;
    end
    pny = linspace(Fmin, Fmax, nr + 1); % probability intervals
    n = randperm(nr); % randomly permuted vector of integers
    dp = pny(2) - pny(1); % probability interval length
    px = pny(n) + (dp)*rand(1, nr);
    r = interp1(pt, xt, px, 'linear', 'extrap'); % use interpolation within each bin
    r = sigma*r + my; % transform from standard to general normal distribution
    r = exp(r);
end
```

**Example:** `lhs_lognormal(0,1,10,10000)`

![Example Graph](image)
function r=lhs_triangular(Min,Max,Mode,lo,hi,n);
if (Min<Max) & (Min<=Mode) & (Mode<=Max)
if lo<Min
    lo=Min;
end
if hi>Max
    hi=Max;
end
if lo<Mode
    Fmin=(lo-Min)^2/((Max-Min)*(Mode-Min));
else
    Fmin=1-(Max-lo)^2/((Max-Min)*(Max-Mode));
end
if hi<Mode
    Fmax=(hi-Min)^2/((Max-Min)*(Mode-Min));
else
    Fmax=1-(Max-hi)^2/((Max-Min)*(Max-Mode));
end
md=(Mode-Min)/(Max-Min);
p=linspace(Fmin,Fmax,n+1);
dp=p(2)-p(1);
p=p(1:end-1);
p=rand(1,n)*dp+p;
p=p(randperm(n));
i1=find(p<=md);
i2=find(p>md);
r(i1)=Min+sqrt((Max-Min)*(Mode-Min)*p(i1));
r(i2)=Max-sqrt((Max-Min)*(Max-Mode)*(1-p(i2)));
else
    disp('condition: Min <= Mode <= Max');
end

Example: lhs_triangular(0,10,3,-inf,inf,10000)
LogTriangular***/***/

function r=lhs_logtriangular(Min,Max,Mode,lo,hi,n);
if lo<Min;lo=Min;end
if hi>Max;hi=Max;end
Min=log(Min);Max=log(Max);Mode=log(Mode);
lo=log(lo);
hi=log(hi);
if (Min<Max) & (Min<=Mode) & (Mode<=Max)
    if lo<Mode
        Fmin=(lo-Min)^2/((Max-Min)*(Mode-Min));
    else
        Fmin=1-(Max-lo)^2/((Max-Min)*(Max-Mode));
    end
    if hi<Mode
        Fmax=(hi-Min)^2/((Max-Min)*(Mode-Min));
    else
        Fmax=1-(Max-hi)^2/((Max-Min)*(Max-Mode));
    end
    md=(Mode-Min)/(Max-Min);
p=linspace(Fmin,Fmax,n+1);
dp=p(2)-p(1);
p=p(1:end-1);
p=rand(1,n)*dp+p;
p=randperm(n));
i1=find(p<=md);
i2=find(p>md);
    r(1)=Min+sqrt((Max-Min)*(Mode-Min)*p(i1));
    r(i2)=Max-sqrt((Max-Min)*(Max-Mode)*(1-p(i2)));
else
    disp('condition: Min <= Mode <= Max');
end
r=exp(r);

Example: lhs_logtriangular(1,10,2,1,10,10000)
Dependent functions:

**get_ranks**: converts values to rank values in a vector/matrix

```matlab
function r = get_ranks(u);
sz = size(u);
if sz(1) == 1
    u = u';
end
n = length(u(:,1));
n2 = 1:n;
n2 = n2';
[i, j] = sort(u);
r = u;
for i2 = 1:n
    for j2 = 1:length(u(1,:))
        r(j(:,i2), i2) = n2;
    end
end
```

**sortbyrank**: sort vector/matrix by rank indices

```matlab
function r = sortbyrank(matr, rnk);
if size(matr) ~= size(rnk)
    disp('Different size of original matrix and rank matrix')
else
    out = matr;
sz = size(matr);
cols = sz(2);
rows = sz(1);
    for i2 = 1:cols
        x = matr(:, i2);
x2 = zeros(rows, cols);
xr = get_ranks(x);
rnkx = rnk(:, i2);
rx = rnk(:, i2);
    for j2 = 1:rows
        out(j2, i2) = x(find(rnkx(j2) == xr));
    end
end
r = out;
```

**invnormal**: get inverse of the normal cumulative distribution function

```matlab
function z = invnormal(p, mu, sigma)
z = (-sqrt(2)*sigma).*erfcinv(2*p) + mu;
```

Explanations:

For all generating functions, the input argument \( n \) is the number of desired samples.
Input arguments \( \text{lo} \) and \( \text{hi} \) are the truncation boundaries, \(-\text{inf}\) and \(\text{inf}\) can be used as arguments when no truncation is desired. No truncation is necessary for uniform distribution.

* In distributions where log-function are used, the natural logarithm is used.
** For log-normal distribution, mean and standard deviation is given for the non-logarithmic normal-parameters are given for the
*** For the logtriangular distribution, the natural logarithm of Min, Max and Mode is the corresponding values for the non-logarithmic distribution.