Chapter 2

SimDec algorithm and guidelines for its usage and interpretation

Mariia Kozlova, Robert J. Moss, Pamphile Roy, Abid Alam, and Julian Scott Yeomans

Abstract

Simulation Decomposition (SimDec) provides an approach for analyzing computational model behaviour and supporting decision-making. At its core is a visualization of model output decomposed by its most influential input variables, where the noteworthy variables have been identified by the calculation of global sensitivity indices. This chapter explains the overall SimDec algorithm, establishes guidelines for its effective implementation, provides the essential building blocks for understanding and interpreting its results, outlines a novel method for calculating sensitivity indices, describes the open-source packages, and discusses various nuances for implementing effective SimDec studies.

1 Introduction

Computational models are frequently created to replicate the behaviour of a system of interest. These "artificial" models can provide facsimiles of such disparate complex systems as societies, economies, ecosystems, scientific endeavours, engineering projects, and investment opportunities. Computational systems can be simulated to study how they behave under different circumstances. The results can provide a better level of understanding of what might be done to succeed, how functioning can be improved, and/or how to ameliorate potential system degradation. The nature of these types of "intentions" has generally been encompassed within the field referred to as *prescriptive analytics*.

In practice, however, the tools available for prescriptive analytics in computational modelling (such as sensitivity analysis and uncertainty analysis) have often been limited, used passively, and/or appended to an analysis to simply "check the box". A lacklustre interpretation of results – how far the probability distribution stretches, what are the most influential variables, are higher-order interactions present in the model, etc. – does not provoke

DOI: 10.4324/9781003453789-3

the consideration of appropriate corrective initiatives, such as: What are the alternatives? What can be done to achieve the desired outcome range? Are there any readily identifiable alternate manoeuvres? What shielding from risk is truly possible?

Simulation Decomposition (SimDec) provides an alternative, methodological pathway to redirect the analysis of computational models towards actionability. In essence, SimDec intelligently maps various multivariable scenarios onto the distribution of an output. This mapping exposes actionable insights that are critical for decision-making, including:

- How to achieve the desired output range while avoiding undesirable ones (which combinations of input variable states result in desirable output ranges)?
- What is the extent of control that can be exercised over the system under the uncertainty conditions present (how much overlap is there between different scenarios)?
- Are there specific scenarios that reduce the levels of uncertainty more than others (what is the relative width of the scenarios)?
- What is the nature of interactions within the model (does an input variable influence the output only under certain circumstances)?

SimDec can facilitate the overall interactivity of many decision-making processes and can prove remarkably powerful in revealing the underlying nature of model behaviour. It also provides a useful tool for assisting in model creation and for ensuring that models function in their intended way. This chapter provides guidelines for those seeking a better understanding of the algorithm behind SimDec (Section 2), wishing to learn how to interpret SimDec results (Section 3), or in using any of the available open-source packages (Sections 4 and 5).

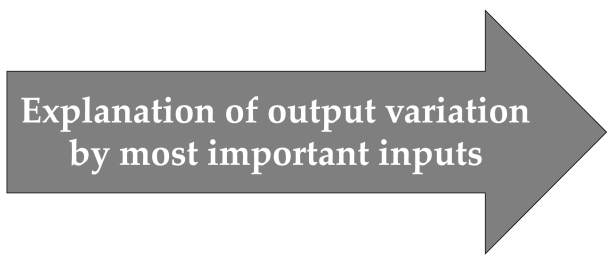
2 SimDec algorithm

Fundamentally, the SimDec algorithm maps multivariable scenarios onto a distribution of model output in an intelligent fashion, thereby enabling a visualization of the critical cause–effect relationships inherent within the model. As its name implies, SimDec decomposes all data (from simulation runs or measured dataset) into automatically created scenarios constructed from the combinations of ranges (states) of the influential input variables. SimDec is a fully automatic procedure that returns a visualization depicting model behaviour that explains the most important sources of variation within the output (see Figure 2.1).

The algorithm consists of two fundamental parts: (1) a computation of sensitivity indices (Kozlova et al., 2023) that discern which influential

Simulation data

#	Y	X_2	X_3
1	-13	86	2
2	338	47	2
3	385	41	2
4	15	78	1
5	61	10	1
...	…	…	…
1000	199	45	1



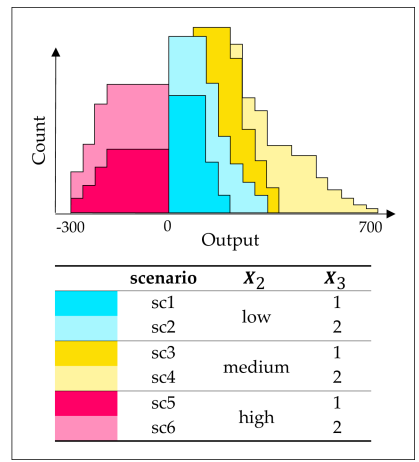


Figure 2.1 Core idea behind SimDec. (colour image is accessible via the link)

variables should be used in the decomposition, followed by (2) a core visualization of the decomposed output distribution (Kozlova & Yeomans, 2022).

2.1 Sensitivity indices computation

The sensitivity indices computed at the beginning of the SimDec procedure are global variance–based indices. The *first-order index* S_{X_i} is computed as a variance of the expectation of Y conditioned on X_i weighted by the total variance of Y.

$$S_{X_i} = \frac{\mathrm{Var}(\mathbb{E}(Y \mid X_i))}{\mathrm{Var}(Y)} \tag{1}$$

The *second-order indices* have a similar formulation, but the expectation of Y is now conditioned on a pair of input variables, and the corresponding first-order effects are deducted to capture the pure excess effect (i.e. it would equal zero in a purely additive model, indicating no interaction).

$$S_{X_iX_j} = \frac{\mathrm{Var}(\mathbb{E}(Y \mid X_i, X_j))}{\mathrm{Var}(Y)} - S_{X_i} - S_{X_j} \tag{2}$$

A *combined* (or *closed*) index is created to aggregate the first- and second-order effects by adding the first-order index to the sum of all second-order effects related to this input variable. The "halving" is necessary to enforce a condition that the sum of all combined effects equals to 1 when the full variability of the output is explained.

$$S^c{}_{X_i} = S_{X_i} + \sum_{\substack{j=1 \\ i \neq j}}^{K_{\text{inputs}}} \frac{S_{X_iX_j}}{2} \tag{3}$$

Equations (1) and (2) can be determined using various mathematical methods and are abstractions of classical Sobol' (1993) expressions. The sensitivity indices used in SimDec are computed via an innovative binning approach (Kozlova et al., 2023) in which the conditional expectation of Y to X (s) is determined by binning the X (s) and then calculating the averages of Y in those bins. Kozlova et al. (2023) have shown that indices calculated by this binning approach possess consistently high accuracy, even on very small datasets. These sensitivity indices can be calculated without any modifications to the algorithm for either simulated data or any measured dataset (i.e. containing input and output variables).

2.2 Decomposition procedure

Figure 2.2 presents a step-by-step illustration of the decomposition procedure on a stylized example model.

The decomposition algorithm requires six steps. Step 1 and step 2 are used to determine which input variables to select for the visualization. Steps 3–6 construct the explicit visual decomposition of the model.

1. **Compute sensitivity indices.** The combined (first- and second-order) effects are calculated.
2. **Order and select inputs.** A threshold of cumulative importance is established. Inputs are selected (in decreasing order of their sensitivity index values) until the cumulative sum of the indices equals or exceeds the threshold. (The threshold could be thought of as establishing "how much explanation of the output variability should be captured by the selected variables".)
3. **Divide inputs into states.** The automatic procedure breaks down the numeric range of each selected input into two or three states – with the same number of observations in each. Three states are chosen if there are two or fewer variables selected in step 2. Otherwise, two states are formed. If an input variable consists of five or fewer unique values, the number of states created corresponds to the number of values (i.e. each value corresponds to exactly one state).
4. **Form scenarios.** All combinations of all states of the selected input variables form an exhaustive set of scenarios. This association is used to construct the legend in the visualization.
5. **Map simulated outputs to scenarios.** Each output value from the dataset is matched to a specific scenario based on the corresponding values of its inputs and the association created in step 4. (The scenario allocations enable a subsequent visualization of the data. As with "classic simulation", the visualization in SimDec is a histogram depicting the distribution of output values.)
6. **Colour-code the output distribution.** The simple output histogram is converted into a stacked histogram (which preserves the original shape), with the scenarios from step 4 as the series. The colour-coding follows a specific rule: the states of the most important variable are assigned distinct primary colours, while all other partitions assume shades of these main colours.

The stacked histogram colouring logic for SimDec can be used in other types of visualizations. For example, box plots can be used when some scenarios contain very little data, the shape of the distribution is inconvenient for visualization, or the data sample is too small to enable the creation of a meaningful histogram (see Section 5.3).

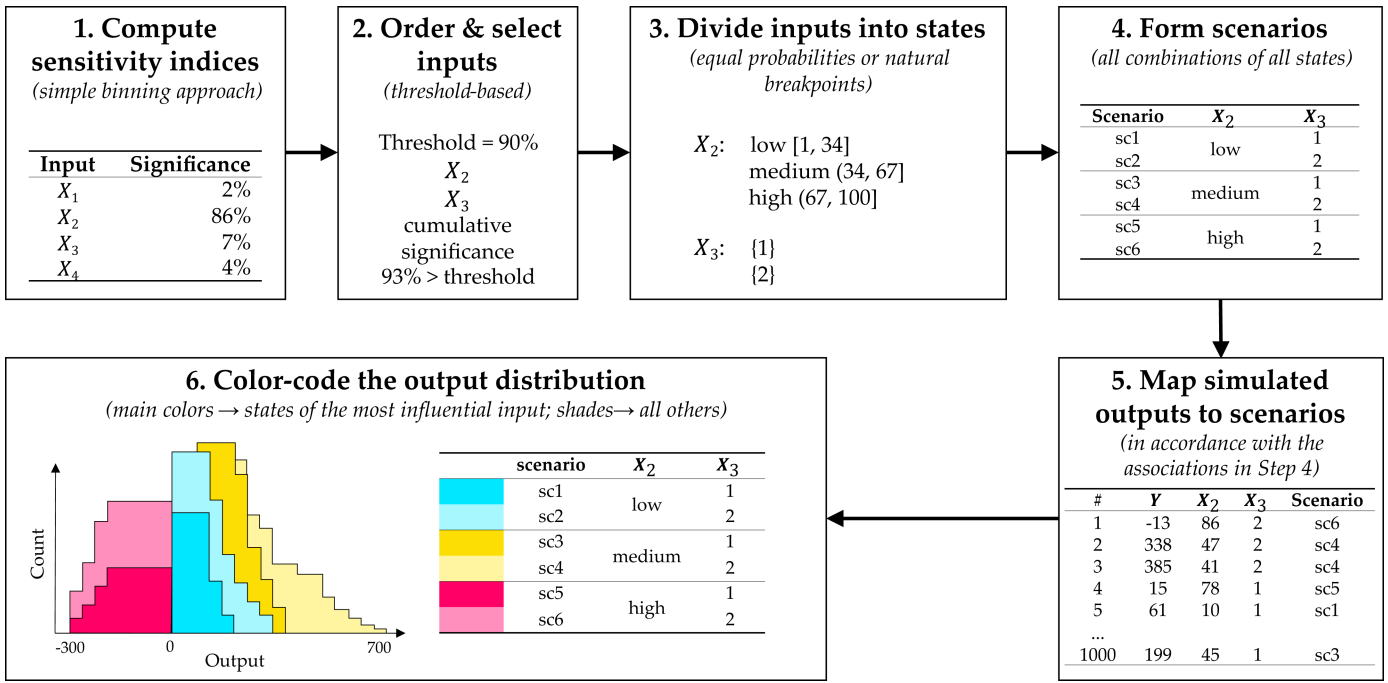


Figure 2.2 SimDec algorithm illustrated on an example model.[1] (colour image is accessible via the link)
Source: Kozlova et al. (2024).

3 How to read SimDec

The various concepts needed to construct an effective interpretation of the SimDec results include an understanding of:

- What do the sensitivity indices actually mean?
- How to read a histogram?
- How to judge the degree of influence of one input on the output using SimDec?
- How to read joint effects of several inputs on the output using SimDec?

3.1 Sensitivity indices

The sensitivity indices used in SimDec are global variance–based indices. *Global* means that they are computed when everything is changing simultaneously (as opposed to one-at-a-time analysis), and *variance-based* means that the index shows how much variability/variation of the output is explained. The simple binning algorithm (Kozlova et al., 2023) used in SimDec computes three types of indices: first-order (or main) effects, second-order (or interaction) effects, and combined (or closed) indices. The combined indices aggregate the first- and second-order effects and provide the default measures used to identify the most influential input variables selected for decomposition.

The first-order indices indicate how much each *input variable* contributes *individually* to the variance of the output. For example, in a situation of $Y=X$, the first-order index of X would be 1.0 (as it explains 100% of the variability). In an additive model $Y=X_1+X_2$, where X_1 and X_2 have identical distributions (or numeric ranges), both inputs would have first-order indices of 0.5 (as, for this model, each explains 50% of the variability). An input variable can have close to 0 influence, if its numeric range is small compared to other variables or if the model mechanics dictate that there is little impact from it.

The second-order indices describe how much *a pair of input variables* contributes to the variance of the output on top of their individual influence. These indices would necessarily be zero for additive models. For example, in $Y=X_1+X_2$, the second-order index for the pair X_1X_2 would be equal to 0. Second-order indices can be positive if the input variables are multiplied in the model or possess a more complex interaction (see 3.4.1, "Interactions"). A positive second-order index means that the pair of variables affects the output synergistically (i.e. together they produce more influence than simply a sum of their individual effects). Second-order indices can assume negative values, which indicate an overlapping effect of these variables (i.e. a correlation or dependency) in the model. For example, if X_1 and X_2 assume the same values in every single simulation run of the model $Y=X_1+X_2$, their second-order effect would be –1.0, denoting a full overlap of their effects. In situations

where both correlations and interactions are present, the second-order index takes the sign of whichever effect is more pronounced (Kozlova et al., 2023).

Combined sensitivity indices are calculated for every input variable as the sum of their first-order index and a halved sum of their second-order indices with all other input variables. The halving is needed to avoid double-counting when summing up all the indices. In the previous example of $Y = X_1 + X_2$, with X_1 and X_2 taking identical values in every simulation run, the first-order effects of both will be 1.0 (since each input separately explains the full variability of the output), the second-order effect of this pair of inputs would be –1.0, the combined index for each input is then 0.5 [= 1.0 + (–1.0)/2], and the sum of the combined indices is 1.0 [= 0.5 + 0.5]. The final summation value of 1.0 means that, overall, the entire variance of the output is explained by these two input variables.

The sum of the combined indices provides a good estimation of whether the selected input variables fully explain the variation of the model output. If the sum is lower than 1.0, it might indicate that there is unaccounted randomness occurring within the model (e.g. if some input variables are not registered for the analysis, there might be some stochasticity coming from the model mechanics or its environment). An alternate explanation might involve the existence of considerable third-order effects. However, third-order effects are a rare phenomenon. A sum of combined indices considerably higher than 1.0 indicates a significant overlapping of information content in the model/system and is a common attribute from the analysis of different model layers (e.g. aggregates of random input variables) or empirical data.

One should bear in mind that sensitivity indices represent approximate estimates and are prone to numeric noise, especially in small-sample/high-number-of-variables situations. Under such circumstances, all indices (especially the second-order ones) can be affected by noise (e.g. many indices would hold values in the 0.01–0.02 range). The focus of analysis should be redirected onto those input variables with indices over 0.05, while all these low-value-effect variables can be safely discarded.

In SimDec, the sensitivity indices prove instrumental in helping to select which inputs to choose for the decomposition, while any actual reporting of their calculated values remains optional.

3.2 Probability distribution/histogram

A histogram provides a representation of the distribution of numerical data. Its horizontal axis shows the range of the variable of interest, and its vertical axis denotes the ***count*** (also called ***frequency***) or the ***probability*** (if the count has been divided by the total number of data points). One could think of creating a histogram as the deliberate act of distributing cubes with numbers (data points) across baskets (bins) that designate the specified number range. Figure 2.3 demonstrates an example of such an action on a small scale.

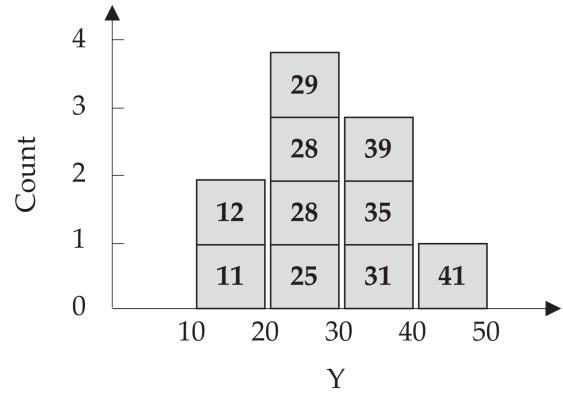


Figure 2.3 A histogram built for an array of Y = {11, 12, 25, 28, 28, 29, 31, 35, 39, 41}. (colour image is accessible via the link)

In Figure 2.3, the Y-axis can be converted from count to probability if divided by the total number of data points, 10. Its labels would be converted from 1 to 0.1 and from 4 to 0.4. The 0.4 mark implies that bins which reach it contain 40% of data (since the bin 20–30 contains 4 cubes, which is 40%). One should note that changing a bin width would also affect the Y-axis of a histogram.

A distribution alone can supply only limited information about the data – its minimum, maximum, shape (where most of the data occurs), together with some additional descriptive statistics. However, an explicit mapping of which input values lead into which specific regions of the output distribution (as provided by SimDec) enables a more definitive exposure of the underlying model behaviour.

3.3 Strength of influence

When decomposing a histogram by a specific single input variable, one can visually perceive its degree of influence. Figure 2.4 demonstrates various single-variable decompositions that project commonly observed scatter plot patterns onto their congruent SimDec visualizations.

If an input variable has no effect on the output, then its states (e.g. low and high) would lie on top of each other in the SimDec histogram, with fully overlapping output ranges. In such a case, the border between the states would be essentially horizontal, and the corresponding sensitivity index would be equal to 0. If an input variable has a strong effect and explains most of the variance of the output, the borders between its states on the SimDec histogram would appear more vertical. Such visualizations have important decision-making implications (e.g. if the high state of X can be fixed by the decision-maker, it would guarantee a certain range of values for Y).

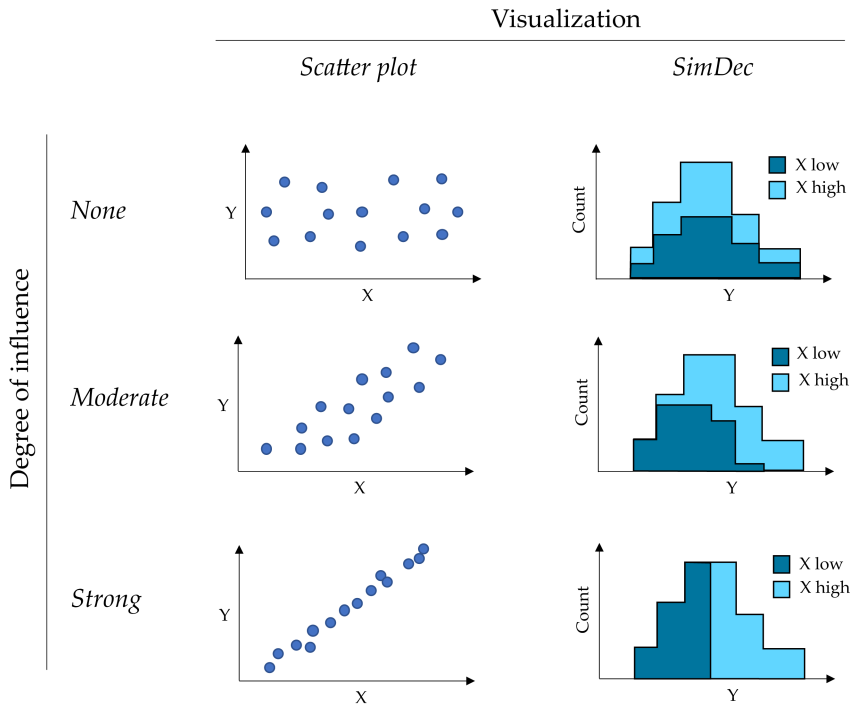


Figure 2.4 Schematic visualization of different degrees of influence of an input variable on a model output in two different visualization types. (colour image is accessible via the link)

The cases in-between possessing low-to-strong effects would display a more "diagonally-appearing" border division between states. The less the states overlap each other, the larger the effect of X on Y. While horizontal displacements of sub-distributions on the SimDec histogram are key to interpreting the results, vertical positionings occur based solely on the technical plotting order of the series in the stacked histogram.

3.4 Joint effects

When two or more input variables are used for decomposition, it becomes possible to examine their joint effects. There are, fundamentally, three ways that a pair of input variables can jointly affect the output:

- The same as the sum of their individual effects (i.e. an absence of correlation or interaction)
- A synergy or extra effect on top of the sum of their individual effects (i.e. interaction)
- A redundancy or overlapping effect (i.e. correlation)

This subsection illustrates how SimDec portrays the different cases behind interactions and correlations in contrast to the no-joint-effect situation.

3.4.1 Interactions

The schematic visualization in Figure 2.5 depicts how different types of interactions of input variables on the output appear in SimDec visualizations.

Figure 2.5A shows how the various sub-distributions (the different colours) of an additive model in which both input variables are equally important would be uniformly shifted. The corresponding second-order effect of such inputs would be equal to zero.

Figure 2.5B illustrates the linear interaction effect that is characteristic of multiplicative models. In the SimDec histogram, the sub-distributions become shifted more-and-more along the horizontal axis. The effect of one input on the output becomes increasingly more magnified with the increasing value of the other input. The sensitivity index computed for the second-order effect of such input variables would be non-zero. The model of an electric aircraft flying range as a function of the capacity of its batteries and the power of its electric motor provides an example of such a linear interaction effect (Kozlova et al., 2021).

In another type of interaction, Figure 2.5C demonstrates how one input variable can switch the direction of influence on the output in different states of the other input variable. Such an effect might occur due to a sign change in a model. The calculated second-order effect would be non-zero. Such an interaction was observed in the carbon footprint model of Kozlova and Yeomans (2022), where, in the case of disposal via landfilling, an increased usage improved the footprint. However, for the case of disposal via incineration, the opposite footprint effect happened. Namely, an increased usage deteriorated the footprint because of accounting for negative carbon emissions.

Figure 2.5D demonstrates that other types of nonlinear interactions can occur in models. For example, an input variable might have no effect on the output in one state of another variable (the red-shaded sub-distributions lying on top of each other) but exhibit a strong effect otherwise (the shifted blue pattern sub-distributions). Such non-linear effects will possess non-zero second-order sensitivity indices. The *crying baby model* of Kozlova et al. (2024) illustrates such an interaction in which the model parameters only affect the output when a particular type of optimization is used.

Figure 2.5 displays one example without interaction (Figure 2.5A) and three cases possessing very different types of interactions (Figures 2.5B–2.5D). In Figures 2.5B–2.5D, the interaction effects are detected by the calculation of non-zero second-order indices. However, it is impossible to ascertain exactly what types of interactions are present without the accompanying SimDec visualizations. Teasing out the nature of the underlying interaction effects in a computational model and understanding the behaviour, in general, is crucial for effective decision-making.

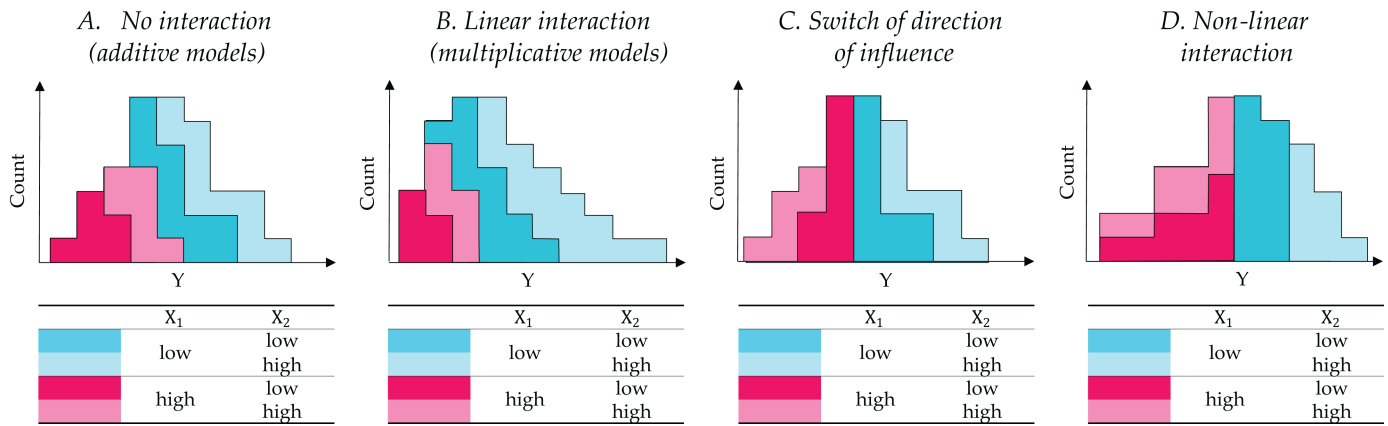


Figure 2.5 Schematic visualization of different types of interactions with SimDec. (colour image is accessible via the link)

3.4.2 Correlations

Even though the states of each input variable are formed to ensure equal numbers of observations in every state (equal area on the SimDec graph), when two or more input variables are combined, some scenarios might contain less data than others. This occurrence is a sign of correlated (or dependent) inputs in the model, which can arise due to deliberately specifying the inputs in a dependent manner (Ahola et al., 2024), the underlying mechanics of the model (Pérez et al., 2024), or an intentional analysis of intermediate outputs (Vinitskaia et al., 2024). Figure 2.6 displays different cases of influential correlated inputs in a model.

Mild correlation (Figure 2.6B) manifests itself in some scenarios having less data than others. Low–low and high–high scenarios have higher probabilities than low–high and high–low ones. This implies that the higher the values of one variable, the more likely it is to see higher values of another variable. An extreme case of such correlation would occur in the total absence of low–high and high–low scenarios (Figure 2.6C). Different types of nonlinear correlations can also be revealed with SimDec. One such example occurs when in one state of one variable no correlation is apparent (high X_1 in Figure 2.6D), but in another state, another variable is only partially present (only low X_2 in low X_1 in Figure 2.6D). All correlation cases presented in Figures 2.6B–2.6D would be readily identifiable due to the negative second-order effects computed via the simple binning procedure.

4 Open-source packages

Open-source SimDec packages have been developed, and all are freely available on GitHub.[2] This enables practitioners to work with a package in their preferred language – currently Python, R, Julia, and Matlab – and all versions operate equivalently on whatever data-set (simulated or otherwise) the user provides. For those who would prefer to circumvent a programming language environment entirely, there is an option to choose between either a web-based dashboard[3] (that works with some given dataset) or an Excel template (that employs a VBA macro to perform the Monte Carlo simulation in a spreadsheet model and then analyze the output). A complete overview of the existing package functionality is presented in Table 2.1.

The Python, R, and Matlab packages all possess identical SimDec functionality. The Julia package only has the visualization functionality and does not compute sensitivity indices. The Python version includes additional functionality that enables the selection of box plot visualization as an alternative to the default histograms. Unfortunately, the Excel template lacks the ability to compute sensitivity indices, thus, only a user-defined selection of input variables for decomposition is possible. The sensitivity indices

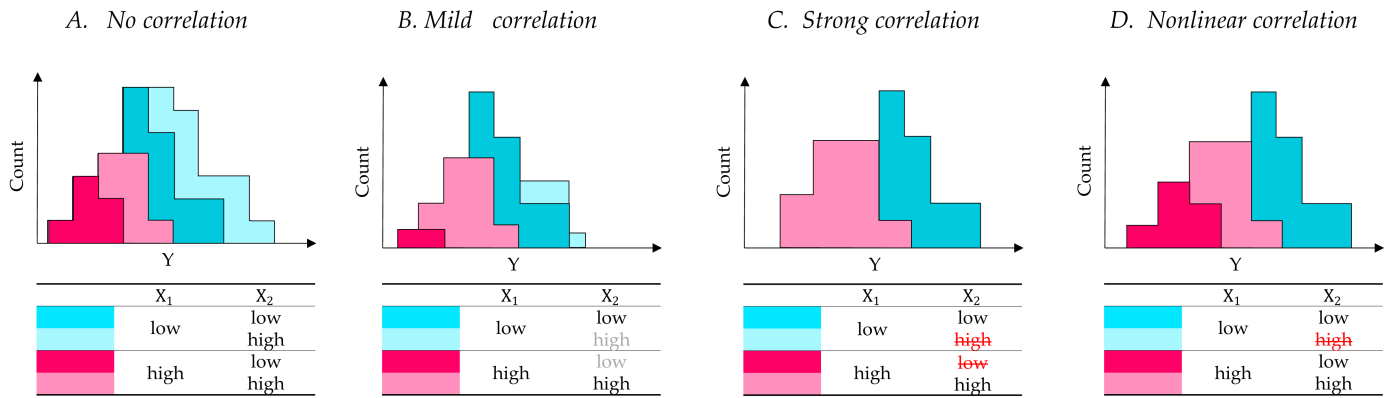


Figure 2.6 Schematic visualization of different cases of correlations with SimDec. (colour image is accessible via the link)

Table 2.1 SimDec open-source packages

Functionality	*Python*	*R*	*Julia*	*Matlab*	*Excel*
Data generation	—	—	—	—	—
Works with a given dataset	✓	✓	✓	✓	×
Runs Monte Carlo simulation of a given model	×	×	×	×	✓
Sensitivity indices	—	—	—	—	—
Computation of sensitivity indices	✓	✓	×	✓	×
Decomposition	—	—	—	—	—
Automatic selection of inputs for decomposition	✓	✓	×	✓	×
Automatic state formation of inputs for decomposition	✓	✓	×	✓	×
Custom selection of input variables for decomposition	✓	✓	✓	✓	×
Visualization	—	—	—	—	—
Stacked histogram visualization	✓	✓	✓	✓	✓
Inbuilt box plot visualization	✓	×	×	✓	×

computed in the Python package may differ slightly from the others because, in Python, the binning is implemented with the available with SciPy's binned_statistic_dd function, whereas in all other packages, a separate binning logic is implemented.

4.1 Python

The Python version of SimDec is publicly developed on GitHub, and its releases are distributed on the Python Package Index (PyPI).[4] It can be installed with:

```
$ pip install simdec
```

The package is composed of three main modules:

- sensitivity_indices: calculate sensitivity indices
- decomposition: perform the SimDec decomposition
- visualization: generate tables and figures

Comprehensive documentation is available on https://simdec.readthedocs.io/. What follows is a short example showing how to use the library. Note that the code is constantly evolving and improving thanks to the feedback received from the community.

```
# load simdec, matplotlib for visualization and pandas
to load data
>>> import matplotlib.pyplot as plot
>>> import pandas as pd
>>> import simdec as sd

# read the data from a CSV file: first column is the output
# other columns are inputs
>>> data = pd.read_csv(fname)
>>> output_name, *inputs_names = list(data.columns)
>>> inputs, output = data[inputs_names],
data[output_name]

# calculate sensitivity indices
>>> indices = sd.sensitivity_indices(inputs=inputs,
output=output)
>>> si = indices.si

# SimDec decomposition itself
>>> res = sd.decomposition(inputs=inputs, output=output,
sensitivity_indices=si)

# based on the number of states, generate a color palette
>>> palette = sd.palette(states=res.states)

# prepare a figure
>>> fig, ax = plt.subplots()

# histogram plot
>>> ax = sd.visualization(bins=res.bins,
palette=palette[::-1], ax=ax)

# table
>>> table, styler = sd.tableau(
. . .  statistic=res.statistic,
. . .  var_names=res.var_names,
. . .  states=res.states,
. . .  bins=res.bins,
. . .  palette=palette,
. . .)
```

This code would produce the visual outputs in Figure 2.7.

Users can manually integrate these functions into their software or analysis, giving them full customization options. The Python package also provides a Dashboard through a Panel web application. It has an easy-to-use graphical user interface (GUI) that runs on a browser. A version deployed

on the cloud has been made publicly available, though users can also run it locally if they want to customize the output or even embed it (see Figure 2.8).

4.2 R

To run the SimDec package in R, the initial step is to install it directly from GitHub. Installing any package requires a loading of the `devtools` package from CRAN, followed by running the `install_github` function to install `SimDec`. The code to execute this process is:

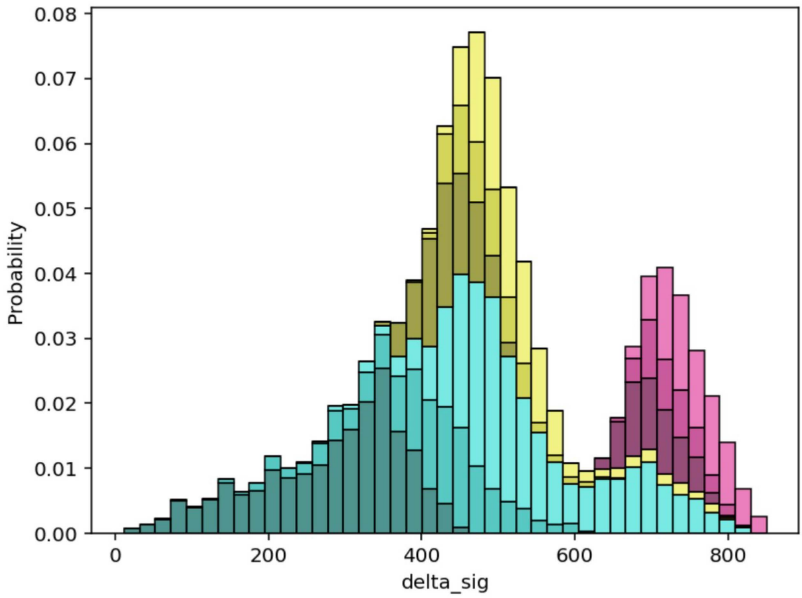
```
> install.packages("devtools")
> library(devtools)
> install_github("Simulation-Decomposition/simdec-R")
> library(SimDec)
```

Once installed and loaded, the package provides two main functions and an example dataset designed to help users familiarize themselves with SimDec. The two functions are (1) `sensitivity_indices` and (2) `simdec_visualization` (see Table 2.2). The example dataset contains 10,000 observations and 5 variables, where variable Y represents the output variable and variables $X1$ through $X4$ are the inputs. Users can access help documentation for the functions and load the example data by executing the following lines of code.

```
> ?sensitivity_indices
> ?simdec_visualization
> data(example_data)
```

To run an automatic SimDec analysis in which the sensitivity indices are used to optimally determine the number and ordering of variables in the decomposition, the following lines of code should be executed, in order to produce a visualization similar to Figure 2.7.

```
> output       <- example_data[,1]
> inputs       <- example_data[,2:5];
> sen          <- sensitivity_indices(output, inputs)
> SI           <- sen$SI
> auto_vis     <- simdec_visualization(output, inputs, SI)
```



sigma_res	R	N°	colour	std	min	mean	max	probability
low	low	9		95.34	11.19	282.08	460.07	0.19
	medium	8		87.76	67.53	407.79	622.35	0.12
	high	7		108.03	237.13	541.32	819.41	0.26
medium	low	6		34.92	350.30	434.90	523.84	0.09
	medium	5		44.39	398.42	485.72	650.98	0.06
	high	4		75.80	414.21	534.19	814.43	0.11
high	low	3		35.43	630.24	703.90	794.81	0.06
	medium	2		33.95	656.34	725.15	816.48	0.04
	high	1		39.51	668.50	755.65	851.00	0.08

Figure 2.7 The output of the SimDec Python package. (colour image is accessible via the link)

```
> auto_vis$simdec_plot
> auto_vis$legend_table
```

The variables for decomposition can also be user-defined (rather than determined automatically), and the colours and appearance of the resulting histogram can be customized manually. For example, to modify the look of the stacked histogram, one could execute the following lines of code.

Figure 2.8 Web-based SimDec dashboard simdec.io. (colour image is accessible via the link)

Table 2.2 Uses and arguments of the SimDec R package functions

Function	*Purpose*	*Inputs*	*Outputs*
sensitivity_ indices.R	Computes sensitivity indices	– input – output	– SI – combined sensitivity indices – FOE – first-order effects – SOE – second-order effects
simdec_ visualization.R	Automatically generates SimDec's stacked histogram visualization	– input – output – SI	– scenarios – vector of scenario indices of size $N_{runs} * 1$ – scen_legend – association between inputs' states and scenarios – boundaries – numeric boundaries of the formed states – simdec_plot – SimDec stacked histogram visualization – legend_table – a legend for the plot

```
> colors           <- c('#8c5eff', '#ffe252', '#0dd189')
> custom_vis       <- simdec_visualization(output,
                    inputs, SI, main_colors = colors)
> custom_vis$simdec_plot
> custom_vis$legend_table
```

A deliberate choice was made during the development process so that the function `sensitivity_indices` would not include any dependencies in order ensure that future "update"-related maintenance issues were minimized. Despite this design decision, the overall SimDec package remains computationally efficient. Conversely, the `simdec_visualization` function does possess several dependencies. However, these dependencies have been carefully chosen to ensure that only regularly maintained packages available through CRAN have been relied upon. Currently, `simdec_visualization` depends on `ggplot2`, `dplyr`, `colorspace`, `gridExtra`, and `kableExtra`. The mandatory arguments for each of the functions are described in detail in Table 2.2, whereas the up-to-date list of optional arguments can be accessed in SimDec R package documentation in GitHub.

4.3 Julia

The scientific computing language Julia is designed for numerical computation with a syntax like Matlab and Python but with the speed of C++ (Bezanson et al., 2017). The lightweight Julia package, SimulationDecomposition.jl, can be installed using the following commands:

```
julia>] # to enter package mode
(@v1.9) pkg> add https://github.com/Simulation-Decompo-
sition/SimulationDecomposition.jl
```

After installation, the user can import the `SimulationDecomposition` package into the Julia REPL. A structure containing the data table and bins is created by first loading the data, selecting the input variables, selecting a target output variable, and number of bins, then calling.

```
julia> using SimulationDecomposition
julia> data = load_data("data_engineering.csv")
julia> inputs = [:Battery, :Motor]
julia> target = :Distance
julia> nbins = 50
julia> simdec = SimDec(data, target, nbins)
```

The coloured histogram and table can then be displayed for further analysis using the functions `plot` and `table`, which result in a similar visualization to that of Figure 2.7.

```
julia> plot(simdec)
julia> table(simdec)
```

An example notebook using the Pluto.jl package can be found in the GitHub repository.[5]

4.4 Matlab

The SimDec Matlab package employs two main functions (see Table 2.3). The Matlab functions[6] must be downloaded and their corresponding folder must be explicitly activated as a path in Matlab. The input data needs to be provided in the form of two variables: `inputs` (of the size $N_{\text{runs}} * K_{\text{inputs}}$) and `output` (of the size $N_{\text{runs}} * 1$).

After the output and the input variables have been formulated in the Matlab workspace, the entire SimDec procedure can be run via these two functions, which will produce a visualization similar to Figure 2.7.

```
% getting the data
Matrix = xlsread ("example_data.xlsx");
output = Matrix(:,1);
inputs = Matrix(:,2:end);

% running SimDec
[SI, FOE, SOE] = sensitivity_indices (output, inputs)
[scenarios, scen_legend, boundaries] = simdec_visualiza-
tion (output, inputs, SI);
```

Several optional arguments are available for the `simdec_visualization.m` function in order to customize a decomposition (see the up-to-date list of optional arguments in the documentation of SimDec Matlab function on GitHub).

The optional arguments are set as in any standard Matlab instance. For example, the following code specifies the names of the variables and changes the colour palette.

```
% custom names and colors
output_name = 'Output';
input_names = {'Input1','Input2','Input3','Input4'};
colors = {'#3F45D0','#DC267F','26DCD1'};
[scenarios, scen_legend, boundaries] = simdec_visualiza-
tion (output, inputs,. . .
     SI,'OutputName',output_name,'InputNames',input_
names,'MainColors',colors);
```

4.5 Excel template

The Excel template is designed to work with spreadsheet models. The template is downloadable via GitHub[7] and contains an example model, a main sheet for the SimDec interface, and a VBA macro that performs the requisite SimDec functionality. The SimDec interface (see Figure 2.9) consists of (1) the Monte Carlo simulation area, (2) a decomposition set-up, and (3) the

Table 2.3 Main functions of the SimDec Matlab package

Function	*Purpose*	*Inputs*	*Outputs*
sensitivity_indices.m	Computes sensitivity indices	– inputs – outputs	– SI – combined sensitivity indices – FOE – first-order effects – SOE – second-order effects
simdec_visualization.m	Automatically creates SimDec stacked histogram visualization	– inputs – outputs – SI	– scenarios – vector of scenario indices of size $N_{runs} * 1$ – scen_legend – association between inputs' states and scenarios – boundaries – numeric boundaries of the formed states – stacked_histogram – object that returns the visualization and the legend

resulting output graphics, together with appropriate summary statistics and a legend.

Detailed instructions for utilizing the template can be found either in a specifically dedicated video tutorial[8] or in Kozlova and Yeomans (2022). The most important distinction between this Excel tool and all remaining SimDec packages is that sensitivity indices are not computed in the template. Any decision to select which variables to use in the decomposition (and their ordering) remains entirely at the discretion of the user.

5 Usage nuances

Several questions come to mind when studying a model with SimDec: How many input variables should be randomized? What should the sample size be? How should the data be sampled? Which variables to choose for decomposition? How to form scenarios for the decomposition? What are the alternatives to stacked histogram visualizations, and when to use them? These questions all are addressed in this section.

5.1 Selection of input variables for decomposition

By default, SimDec uses sensitivity indices to indicate exactly which variables to select for decomposition. The *de facto* method-of-choice is the simple binning approach of Kozlova et al. (2023) and this procedure is incorporated into all SimDec packages. However, any other method for computing sensitivity indices could be used, if preferred. Variance-based methods make more sense, since they straightforwardly translate higher values into more widely dispersed variable states in the histogram. It is also easier to work

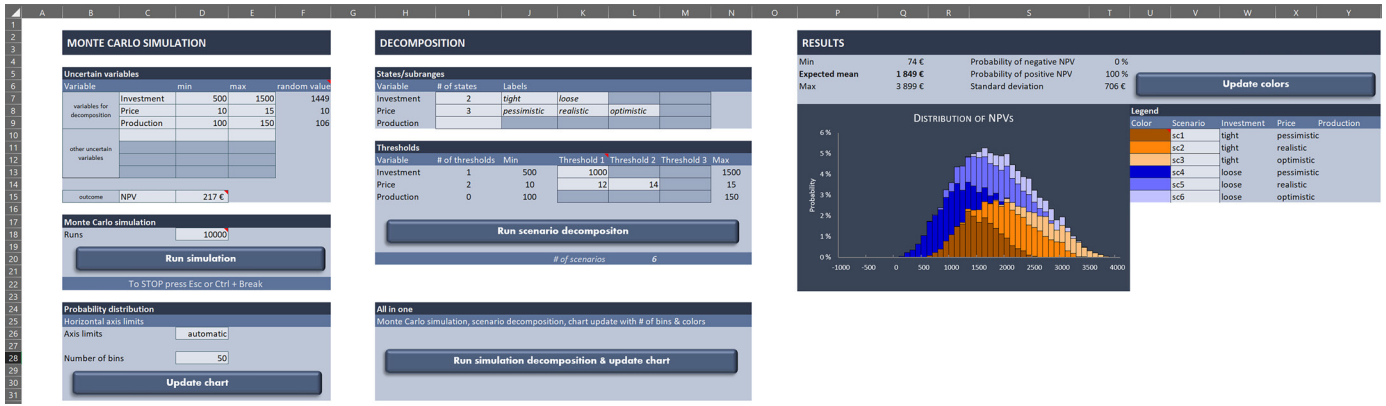


Figure 2.9 SimDec Excel template. (colour image is accessible via the link)

with methods that can operate on the given data (Plischke, 2012; Puy et al., 2024; Kozlova et al., 2023), since the same dataset is later used to build the visualization.

Conversely, selection of input variables for decomposition could also be done manually – whether for exploratory data analysis purposes or to satisfy alternate decision contexts. In complex nonlinear models, exploring the nature of separate interaction effects or the shape of a single-input variable influence on the output can produce additional insights into the model behaviour (Ahola et al., 2024). Some decision problems might dictate the specific choice of variables (e.g. the public policy requirements for project performance in Kozlova et al. (2016)).

5.2 States and scenario formation

One important distinction of SimDec from scenario analysis is that the set of scenarios is not arbitrarily decided upon but results from listing all state combinations in the actual decomposition. However, the choice of the number and numeric boundaries of states is more flexible. By default, SimDec creates three states if two input variables are selected and two states otherwise. An exception occurs when an input variable can assume no more than five unique values, in which case, each value instance becomes its own separate state. The number of states can always be modified in response to the specific needs of the decision context. For example, in one of the SimDec application chapters, a decomposition into nine states is created in order to depict the sub-distributions of nine distinct market opportunities (Myers et al., 2024). It is imperative to supply the corresponding number of HEX codes for the main colours to the function in order to ensure its proper functioning.

For establishing the numeric boundaries between states, the default procedure is to ensure an equal apportionment of data into each state. An alternative, inbuilt option is to choose equally-sized numeric intervals for the states. The two approaches create an identical set of states if the input variables are uniformly distributed. For non-uniformly distributed variables, the "equal-interval" principle allocates different amounts of data into each state. The choice should be based on the specific decision context. For example, use the "equal-amount-of-data" option to reflect the equal probabilities of occurrence of different states for external variables not controlled by the decision-maker. However, equal-sized-interval states would be preferable if the variable is under the decision-maker's control. Custom numeric boundaries have been prescribed to reflect certain key thresholds imposed by a decision-maker, where SimDec can then be used to see whether achieving that threshold (or not) proves beneficial (Kozlova et al., 2016).

However, if the purpose of an analysis is to study the behaviour of the underlying model, then the default state formation method is advised in order to prevent possible visual distortions in the visualization. For example,

an increasing number of data points in states of a uniformly distributed input variable can be confused with linear interaction (see Figure 1.5B), while erroneous boundary setting that results in no data in a state can be confused with correlation (see Figure 1.6C). If empirical data is analyzed, algorithms that detect natural breaking points for defining the state boundaries might provide the judicious choice. Studying the application of SimDec to empirical data provides a potentially fruitful avenue for the direction of future research and development.

5.3 Sample size and sampling strategies

Sample size, in conjunction with the number of randomized input variables, can affect SimDec performance in two ways: (1) accuracy of the computed sensitivity indices, and (2) smoothness of the stacked histogram visualization.

For computing sensitivity indices, the larger the sample size and the fewer the number of variables, the higher the accuracy of resulting estimations. First-order indices converge sooner (Marzban & Lahmer, 2016) than second-order ones (Kozlova et al., 2023). In general, as few as 1,000 data points are sufficient to generate stable and reliable sensitivity indices for models possessing six input variables (Kozlova et al., 2023). Furthermore, quasi-random sampling can be used to improve the accuracy (Kozlova et al., 2023). Increasing noise combined with higher numbers of input variables results in much noisier second-order effects (most of the pairs of input variables show joint 0.01–0.02 effects instead of zero). In such situations, the resulting sum of indices has been observed to overshoot the expected 100%. Nevertheless, even with either a smaller sample size or a larger number of variables, first-order indices have been reliably used to judge the relative importance of input variables. Examples of this range of reliability can be observed for an application with 29 inputs and 1,000 sample (Pellegrino et al., 2024) and for a case of a partial dataset of only 152 points (Pérez et al., 2024).

In a visualization, the more data points there are, the smoother the histogram itself and the more distinct the borders between the scenarios appear (see Figure 2.10). One thousand data points appear to establish the basic minimum amount of data for clear readability. However, if the computational costs are bearable, then 10,000 data points are recommended, as this can produce a very smooth and crisp visualization. An increased number of uncertain variables causes more uncertainty in the model output. This uncertainty results in a larger overlap of consecutive scenarios. The choice for the number of randomized input variables for the simulation (or data analysis) should appropriately reflect the task at hand – a higher number for more realistic modelling of the system and a lower number for studying the key behaviours in the model. Iterative analysis with consequent removal of

less-influential inputs (or the adding of other variables if something important had been missed) enables in-depth exploration of a model behaviour.

It can be observed that different sampling strategies do not produce significant differences (see each column in Figure 2.10). Quasi-random sampling and full factorial designs result in slightly smoother visualizations at sample sizes of 10,000. However, full factorial designs involve rapid escalations in computational costs as the number of input variables increases. Thus, quasi-random sampling can be recommended as the best approach for data generation in SimDec, as this simultaneously improves both the quantitative and visual aspects of the results.

5.4 Alternative visualization types

Figure 2.11 shows that the same information depicted in stacked histograms can also be visualized using box plots. In the figure, each scenario indicated previously with a specific-coloured sub-distribution in the stacked histogram is now represented by a separate box in the box plot. Furthermore, a detailed tracing indicates that each box is located precisely under the correspondingly coloured sub-distribution of the histogram above.

Box plots provide a useful alternative under the following circumstances.

- Some of the scenarios contain very little data and are not visible on the histogram (see an example in Figure 2.12).
- The shape of the distribution is inconvenient for histogram visualizations (for example, the too-skewed distribution in an exponential model).
- The data sample is too small, and the histogram is too dissected (e.g. Figure 2.10, bottom row).

The colour-coding of scatter plots according to the values of another input have occasionally appeared in the literature (see, for example, Palar et al., 2023). However, a multivariable decomposition colouring on the scatter plot did not yield any informative visualizations. Scatter plot visualizations tend to be obscured by the overlapping of the dots in the scenarios and by the need to read the effect of one input variable relative to the dots' location while reading (an)others relative to their colours.

Overlay charts have also been considered as an alternative visualization format and have been incorporated as an optional display in a number of the commercial spreadsheet add-ins. The idea is analogous to the scenario portrayal in a stacked histogram, but instead of the stacking, these "scenarios" are overlayed instead. However, this overlaying approach possesses multiple drawbacks, including problems in visualizing multiple scenarios, simultaneously, which is an essential "must-have" pre-requisite provided by SimDec (Kozlova & Yeomans, 2020). Consequently, stacked histograms (and box

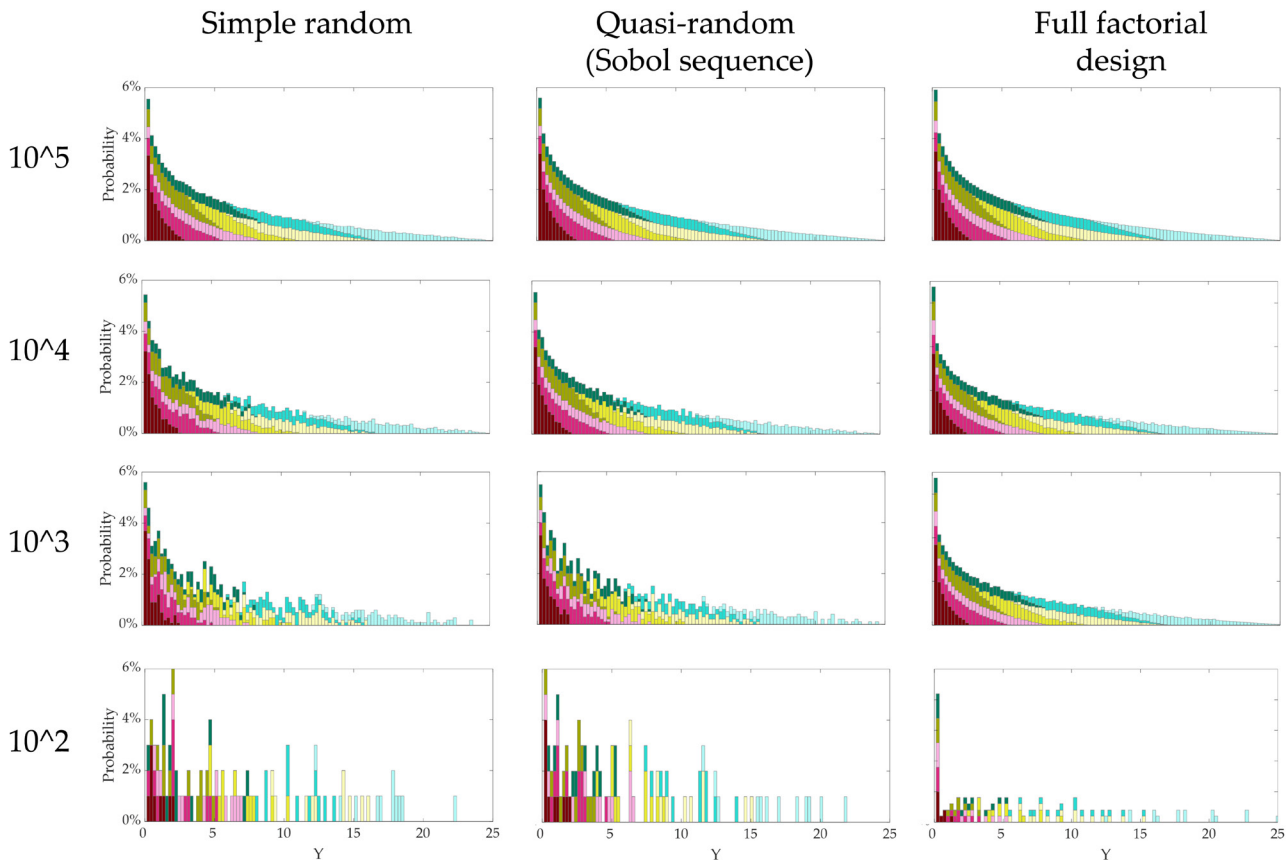
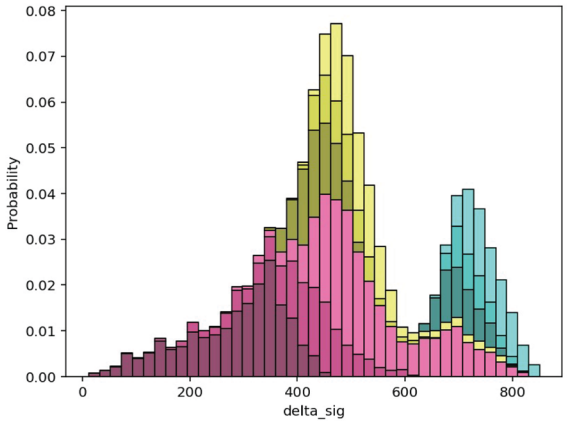
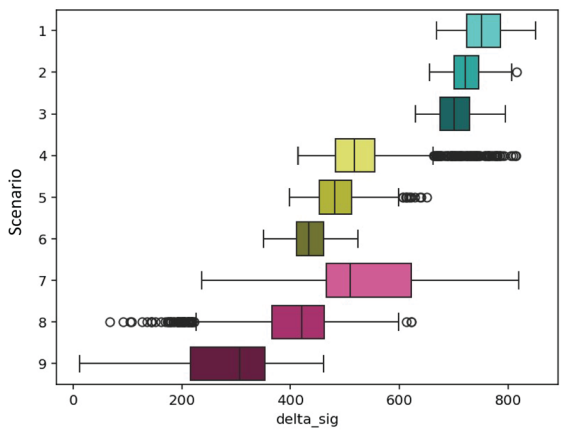


Figure 2.10 Appearance of SimDec histogram depicting $1Y = \begin{cases} X_1(1+X_2X_3) & X_2 < 60 \\ X_1(1+X_4) & X_2 \geq 60, \end{cases}$ model output under different sample sizes (rows) and sampling strategies (columns). (colour image is accessible via the link)

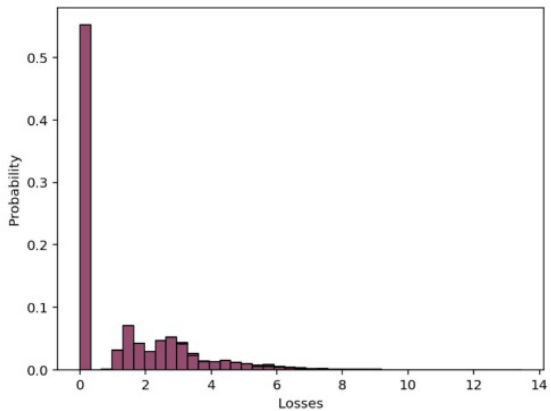




sigma_res	R	N°	colour	std	min	mean	max	probability
low	low	9		95.34	11.19	282.08	460.07	0.19
	medium	8		87.76	67.53	407.79	622.35	0.12
	high	7		108.03	237.13	541.32	819.41	0.26
medium	low	6		34.92	350.30	434.90	523.84	0.09
	medium	5		44.39	398.42	485.72	650.98	0.06
	high	4		75.80	414.21	534.19	814.43	0.11
high	low	3		35.43	630.24	703.90	794.81	0.06
	medium	2		33.95	656.34	725.15	816.48	0.04
	high	1		39.51	668.50	755.65	851.00	0.08

Figure 2.11 Comparison of stacked histogram with box plots produced for the same decomposition of the structural reliability case implemented with SimDec dashboard,[9] where all scenarios are clearly visible on both visualization types. (colour image is accessible via the link)

Source: Ahola et al. (2024).



Risk B	Risk C	Risk A	N°	colour	std	min	mean	max	probability
low	low	low	8		1.66	0.00	1.22	12.11	0.98
		high	7		1.53	3.09	4.54	8.99	0.01
	high	low	6		1.81	5.39	7.47	12.42	0.00
		high	5		nan	9.64	9.64	9.64	0.00
high	low	low	4		1.77	5.43	7.16	13.47	0.01
		high	3		nan	nan	nan	nan	0.00
	high	low	2		nan	nan	nan	nan	0.00
		high	1		nan	nan	nan	nan	0.00

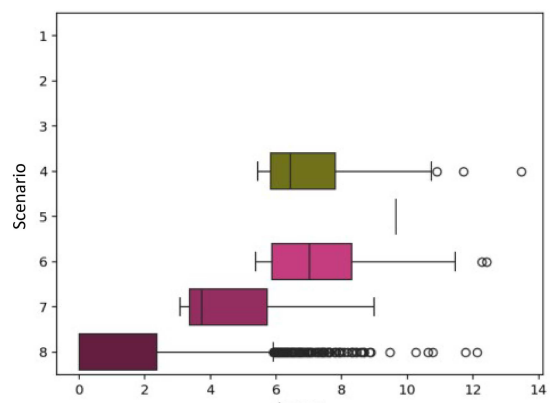


Figure 2.12 Comparison of stacked histogram with box plots produced for the same decomposition of a risk analysis case implemented with SimDec dashboard,[10] where only one out of four existing scenarios is visible on the histogram. (colour image is accessible via the link)

Source: Kokkonen (2023).

plots for cases where these are not sufficient) are explicitly recommended for the visualization of decompositions.

6 Conclusion

This chapter has outlined the structure underlying the basic SimDec algorithm and has provided guidelines for its effective usage and interpretation. The fundamental elements for understanding and interpreting the results of SimDec – including both quantification (sensitivity indices) and visualization – have been described. A summary of the various nuances regarding the experimental design for SimDec usage leads to the following recommendations: (1) either simple random sampling or quasi-random sampling should be adopted, if possible; (2) at least 1,000 data items should be used, and more if the number of randomized input variables is more than six; (3) the default automatic decomposition (using the most influential variables identified by the binning sensitivity indices and where states contain equal amounts of data) should be employed, unless dictated otherwise by the problem context; and (4) the stacked histogram should provide the default means for SimDec visualizations.

Open-source SimDec codes have been made readily accessible in Python, R, Julia, and Matlab. In addition, there are options available to run SimDec that do not possess any programming requirements from either a web-based dashboard or by using an Excel template. Finally, readers are invited to join the Sensitivity Analysis community on Discord[11] for further information regarding the applicability and interpretation of results, for updates in the development cycles of the various SimDec software packages, to acquire specific knowledge of other contributions, and to gain experience from networking collaborations with others in the broader SimDec community.

Acknowledgements

This research was supported in part by grant OGP0155871 from the Natural Sciences and Engineering Research Council; by funding from Business Finland, grant # 6713/31/2021; and by grants #220177 and #220178 from Finnish Foundation for Economic Foundation.

Notes

1 $Y = \begin{cases} X_1(1+X_2X_3) & X_2<60 \\ X_1(1+X_4) & X_2 \geq 60, \end{cases}$, where X_1 is uniformly distributed between 30 and 50, X_2 between 1 and 100, X_4 *between –7 and –0.3 and* X_3 *is a binary variable that assumes values either 1 or 2. The model is simulated* 1,000 times, and the simulated data consisting of the output and four input variables of the overall size 1,000 by 5 is fed to SimDec algorithm.
2 https://github.com/Simulation-Decomposition.
3 https://simdec.io/.
4 https://pypi.org/project/simdec.

5 https://github.com/Simulation-Decomposition/SimulationDecomposition.jl.
6 https://github.com/Simulation-Decomposition/simdec-matlab.
7 https://github.com/Simulation-Decomposition/simdec-excel.
8 https://youtu.be/8l6D58fiOxs?si=BfuiYXIaaOwU-Cfd.
9 https://simdec.io/.
10 https://simdec.io/.
11 https://discord.gg/M7XeFzCpRs.

References

Ahola, A., Kozlova, M., & Yeomans, J. S. (2024). Capturing multi-dimensional nonlinear behaviour of a steel structures reliability model – global sensitivity analysis. In M. Kozlova & J. S. Yeomans (Eds.), *Sensitivity analysis for business, technology, and policymaking made easy with Simulation Decomposition*. Routledge.

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. (2017). Julia: A fresh approach to numerical computing. *SAIM Review*, *22*(1), 65–98.

Kokkonen, M. (2023). *Towards improved decision support in financial sector operational risk analysis* [Master's thesis, LUT University]. https://urn.fi/URN:NBN:fi-fe20231030141810

Kozlova, M., Ahola, A., Roy, P., & Yeomans, J. S. (2023). *Simple binning algorithm and SimDec visualization for comprehensive sensitivity analysis of complex computational models* (Working Paper LUT). https://doi.org/10.48550/arXiv.2310.13446

Kozlova, M., Collan, M., & Luukka, P. (2016). Simulation Decomposition: New approach for better simulation analysis of multi-variable investment projects. *Fuzzy Economic Review*, *21*(2), 3–18.

Kozlova, M., Moss, R. J., Yeomans, J. S., & Caers, J. (2024). Uncovering heterogeneous effects in computational models for sustainable decision-making. *Environmental Modelling & Software, 171,* 105898. https://doi.org/10.1016/j.envsoft.2023.105898

Kozlova, M., Nykänen, T., & Yeomans, J. S. (2021). Technical advances in aviation electrification: Enhancing strategic R&D investment analysis through Simulation Decomposition. *Sustainability*, *14*(1), 414.

Kozlova, M., & Yeomans, J. S. (2020). Visual analytics in environmental decision-making: A comparison of overlay charts versus Simulation Decomposition. *Journal of Environmental Informatics Letters*, *4*, 93–100.

Kozlova, M., & Yeomans, J. S. (2022). Monte Carlo enhancement via Simulation Decomposition: A "must-have" inclusion for many disciplines. *INFORMS Transactions on Education*, *22*(3), 147–159.

Marzban, S., & Lahmer, T. (2016). Conceptual implementation of the variance-based sensitivity analysis for the calculation of the first-order effects. *Journal of Statistical Theory and Practice*, *10*, 589–611.

Myers, A., Kozlova, M., & Yeomans, J. S. (2024). Where should we go? Deep tech market entry decisions through the lens of uncertainty. In M. Kozlova & J. S. Yeomans (Eds.), *Sensitivity analysis for business, technology, and policymaking made easy with Simulation Decomposition*. Routledge.

Palar, P. S., Zuhal, L. R., & Shimoyama, K. (2023). Enhancing the explainability of regression-based polynomial chaos expansion by Shapley additive explanations. *Reliability Engineering & System Safety*, *232*, 109045.

Pellegrino, R., Kozlova, M., Brandao, L., & Yeomans, J. S. (2024). Unpacking the role of contextual factors in public support for mitigating revenue risk in public–private partnership projects. In M. Kozlova & J. S. Yeomans (Eds.), *Sensitivity analysis for business, technology, and policymaking made easy with Simulation Decomposition*. Routledge.

Pérez, M. G., Kozlova, M., Bermúdez, S. I., Pérez, J. C., & Yeomans, J. S. (2024). Sensitivity analysis of a superconducting magnet design model. In M. Kozlova & J. S. Yeomans (Eds.), *Sensitivity analysis for business, technology, and policymaking made easy with Simulation Decomposition*. Routledge.

Plischke, E. (2012). How to compute variance-based sensitivity indicators with your spreadsheet software. *Environmental Modelling & Software*, *35*, 188–191.

Puy, A., Roy, P. T., & Saltelli, A. (2024). Discrepancy measures for global sensitivity analysis. *Technometrics*, 1–11. https://doi.org/10.1080/00401706.2024.2304341

Sobol', I. M. (1993). Sensitivity estimates for nonlinear mathematical models. *Mathematical Modeling and Computational Experiment*, *1*, 407.

Vinitskaia, N., Zaikova, A., Kozlova, M., & Yeomans, J. S. (2024). Uncertainty considerations in life cycle assessment of COVID-19 masks: Single-use versus reusable. In M. Kozlova & J. S. Yeomans (Eds.), *Sensitivity analysis for business, technology, and policymaking made easy with Simulation Decomposition*. Routledge.