gsa-module Documentation

Release 0.9.0

Damar Wicaksono

Jul 06, 2017

Contents

Welcome to the `gsa-module` package documentation, a python3 package to conduct global sensitivity analysis of model output.

If you're just getting started with `gsa-module`, please start with *gsa-module Basics*. If you're already familiar with it, refer to *User's Guide* and *gsa-module Modules reference documentation* for further detail and reference. If you're curious with some theories and detail of implementation behind global sensitivity analysis methods check the *Theory and Implementation*. Finally, if you're thinking to make modification or extend the capability of the package perhaps it is a good idea to check the *Developer's Guide*.

CHAPTER 1

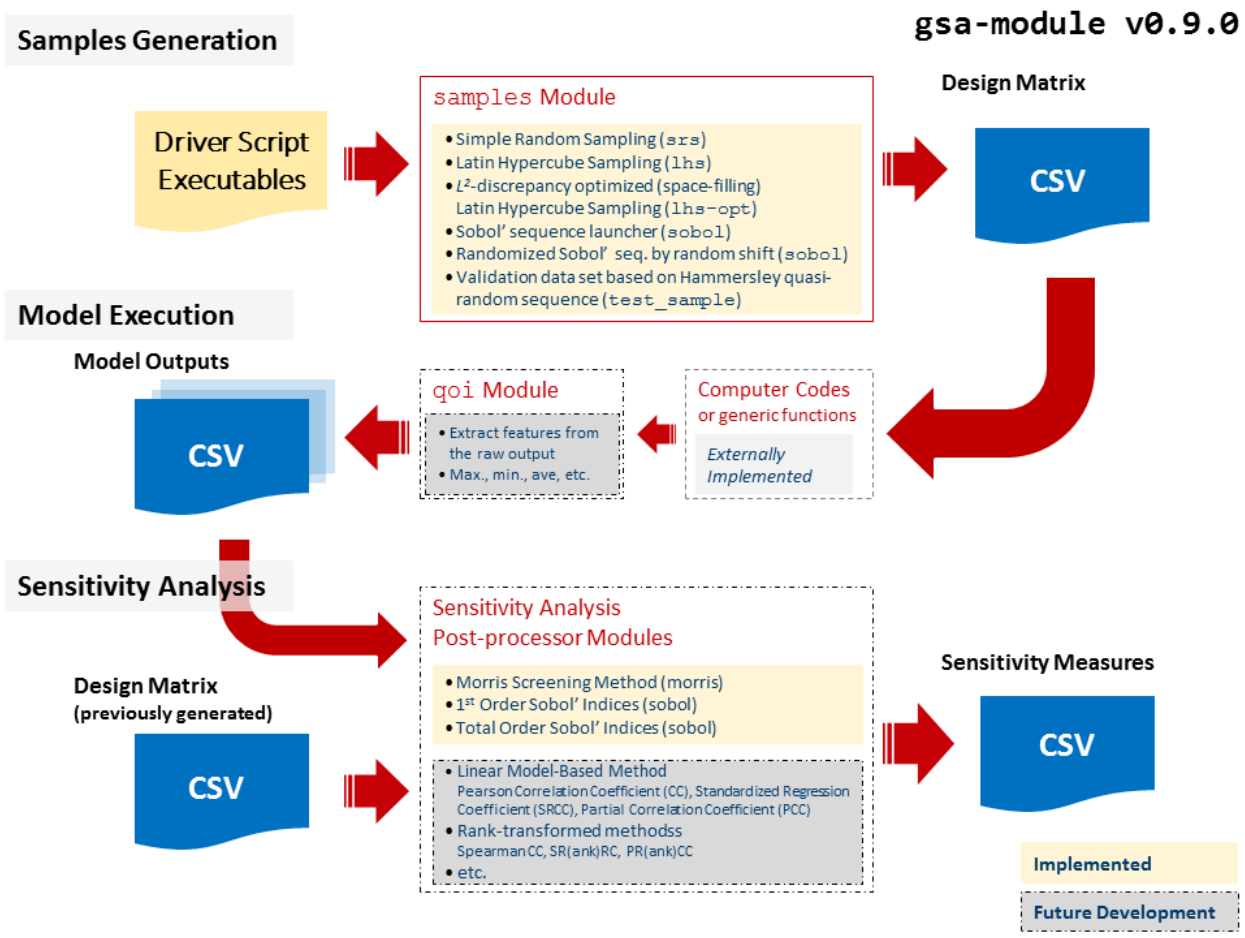
gsa-module Documentation

gsa-module Basics

Introduction

gsa-module is a Python3 package implementing several global sensitivity analysis methods for computer/simulation experiments. The implementation is based on a black-box approach where the computer model (or any generic function) is externally implemented to the module itself. The module accepts the model outputs and the design of experiment (optional, only for certain methods) and compute the associated sensitivity measures. The package also includes routines to generate normalized design of experiment file to be used in the simulation experiment based on several algorithms (such as simple random sampling or latin hypercube) as well as simple routines to post-processed multivariate raw code output such as its maximum, minimum, or average.

The general calculation flow chart involved in using the gsa-module can be seen in the figure below.



List of Features

The following is the main features of the current release (v0.9.0):

- Capability to generate design of computer experiments using 4 different methods: simple random sampling (srs), latin hypercube sampling (lhs), sobol' sequence, and optimized latin hypercube using either command line interface gsa_create_sample or the module API via import gsa_module
- Sobol' quasi-random number sequence generator is natively implemented in Python3 based on C++ implementation of Joe and Kuo (2008).
- Randomization of the Sobol' quasi-random number using random shift procedure
- Optimization of the latin hypercube design is done via evolutionary stochastic algorithm (ESE)
- Generation of separate test points based on a given design using Hammersley quasi-random sequence
- Capability to generate design of computer experiments for screening analysis (One-at-a-time design), based on the trajectory design (original Morris) and radial design (Saltelli et al.)
- Capability to compute the statistics of elementary effects, standardized or otherwise both for trajectory and radial designs. The statistics (mean, mean of absolute, and standard deviation) are used as the basis of parameter importance ranking.
- Capability to estimate the first-order (main effect) Sobol' sensitivity indices using two different estimators (Saltelli and Janon).

- Capability to estimate the total effect Sobol' sensitivity indices using two different estimators (Sobol-Homma and Jansen).
- All estimated quantities are equipped with their bootstrap samples

Installing `gsa-module`

Obtaining and installing `gsa-module` is simple. First is to download the current version the current version hosted in bitbucket and install it to your machine locally. `gsa-module` is written in python3 and can be installed using `pip`:

```
> git clone https://bitbucket.org/lrs-uq/gsa-module
> cd gsa_module
> pip install .
```

Verifying the installation can be done by invoking:

```
> python
>>> import gsa_module as gsa
>>> gsa.__version__
'0.9.0'
```

If you want to modify the package on the fly without re-installing it everytime to check the effect use the `-e` (editable mode) when invoking `pip`:

```
> pip install -e .
```

Tutorial 1: Parameter Screening of the Sobol-G Test Function

This page is still under preparation

Tutorial 2: Sobol' Indices of the Borehole Function

This page is still under preparation

User's Guide

General Purpose Design of Experiment

Successful installation of `gsa-module` will give access to two executables in the path useful to create various designs for computer experiment. The command line utility `gsa_create_sample` can be invoked from the terminal to generate the design (or *design matrix file*) using the following command:

```
> gsa_create_sample -n <number of samples/points> \
                    -d <number of dimensions/variables> \
                    -m <method of generation {srs, lhs, sobol, lhs-opt}> \
                    -s <random seed number> \
                    -o <design matrix output filename> \
                    -sep <delimiter for the design matrix file> \
                    -dirnumfile <direction number file, Sobol' only> \
                    -excl_nom <exclude nominal file, Sobol' only> \
                    -rand <randomize the Sobol' sequence, Sobol' only> \
                    -nopt <number of optimization iterations, Sobol' only>
```

Brief explanation of these parameters can be shown by invoking:

```
> gsa_create_sample --help
```

The table below lists the complete options/flag in detail with their respective default values.

No.	Short Name	Long Name	Type	Required	Description	Default
1	-h	–help	flag	No	Show help message	False
2	-n	–num_samples	integer	Yes	The number of samples/design points	None
3	-d	–num_dimensions	integer	Yes	The number of dimensions	None
4	-m	–method	string	No	The method to generate sample {srs, lhs, sobol, lhs-opt}	srs
5	-s	–seed_number	integer	No	The random seed number	None
6	-o	–output_file	string	No	The output filename	see below
7	-sep	–delimiter	string	No	The delimiter for the file {csv, tsv, txt}	csv
8	-dirnumfile	–direction_numbers	string	No	The path to Sobol' sequence generator	None
9	-excl_nom	–exclude_nominal	flag	No	Exclude the nominal point {0.5} in the design	False
10	-rand	–randomize_sobol	flag	No	Random shift the Sobol' sequence	False
11	-nopt	–num_iterations	integer	No	The maximum bumber of optimization iterations	100
12	-V	–version	flag	No	Show the program's version number and exit	False

Note that options number 8-9 are valid only for Sobol' design, while option number 11 is valid only for Optimized LHS. Without specifying the output filename explicitly, the design matrix file will be produced with the following naming convention:

```
> <method>_<num_samples>_<num_dimensions>.csv
```

Example

For example, upon executing the following command in the terminal:

```
> gsa_create_sample -n 20 -d 5 -m sobol
```

a csv file with filename `sobol_20_5.csv` is produced in the current working directory. The first 5 lines (out of 20) the file are as follow:

```
0.000000e+00,0.000000e+00,0.000000e+00,0.000000e+00,0.000000e+00
5.000000e-01,5.000000e-01,5.000000e-01,5.000000e-01,5.000000e-01
7.500000e-01,2.500000e-01,2.500000e-01,2.500000e-01,7.500000e-01
2.500000e-01,7.500000e-01,7.500000e-01,7.500000e-01,2.500000e-01
3.750000e-01,3.750000e-01,6.250000e-01,8.750000e-01,3.750000e-01
```

In each line, the listed values correspond to the input parameters at which the model is to be evaluated. The values are normalized between 0 to 1.

Morris Screening Method

Successful installation of `gsa-module` will give access to two executables in the path useful to carry out Morris screening analysis on model outputs:

1. `gsa_morris_generate`: executable to generate Morris (One-at-a-Time, OAT) design
2. `gsa_morris_analyze`: executable to compute the statistics of the elementary effects given model inputs/outputs as files

A more theoretical background of the method can be found in the implementation section of this documentation.

Generating Morris Design (Sample)

The first step in conducting sensitivity analysis by Morris screening method is to generate One-at-a-Time design. The Morris design generator driver script can be invoked from the terminal using the following command:

```
> gsa_morris_generate -r <number of blocks/replications> \
                      -d <number of input dimensions> \
                      -o <design matrix output filename> \
                      -sep <delimiter for the output file {csv, tsv, txt}, default =␣
↪csv> \
                      -ss <sampling scheme {trajectory or radial}> \
                      -p <trajectory scheme only, number of levels> \
                      -s <trajectory scheme only, random seed number> \
                      -sobol <radial scheme only, the fullpath to Sobol' sequence␣
↪generator executable> \
                      -dirnum <radial scheme only, the fullpath to Sobol' sequence␣
↪generator direction numbers file>
```

Brief explanation on this parameter can be shown using the following command:

```
> gsa_morris_generate --help
```

By default the naming convention of the output file (if not explicitly specified is):

```
<sampling scheme>_<number of replications>_<number of input dimensions>_<number of␣
↪levels, trajectory only>.csv
```

In general, the larger the number of replications the more accurate the sensitivity measures are. On the other hand, a large number of levels (in trajectory design) increases the granularity in the input parameter space exploration. However, this only makes sense if there is large number of replications otherwise there is big risk of using unbalance design (bias in some part of input parameter space)

Example

As an example, consider the following command:

```
> gsa_morris_generate -r 10 -d 4 -p 6
```

The above command will generate a `csv` file with the name of `trajectory_10_4_6.csv` containing *50* rows and *4* columns. *50* rows are obtained from the $replicates \times (inputs + 1)$ formula and it corresponds to the number of model evaluations, while the number of columns corresponds to the number of input dimensions. In OAT design only one parameter is changed between perturbation and in the case of trajectory scheme there is no base point per se as the perturbations are carried out one parameter at a time from the last perturbed point. In the example above the size of grid jump ($\Delta = 2/3$) is locked to the number of levels.

Another example with more explicit specification of arguments:

```
> gsa_morris_generate -r 10 -d 6 -ss radial \
                      -o test_radial \
                      -sep txt \
                      -sobol ./path_to_sobol_gen/sobol_gen.x \
                      -dirnum ./path_to_sobol_gen/dirnum.txt
```

The above command will generate a space separated file `test_radial.txt` containing *70* rows and *6* columns. The radial OAT design is generated using the specified Sobol' sequence generator. In the radial design, multiple base points are generated for different replications. The perturbation per parameter in each replication is relative to the base point. Furthermore, number of level is not required to be specified as the size of grid jump differs from parameter to parameter and from replication to replication.

Executing Model

Following the design philosophy of `gsa-module` the model executions are implemented outside the module itself. The most important thing to remember is that the OAT design generated using `gsa_morris_generate` is normalized (between [0,1]). If the actual model has a different scale of parameters or different probability distribution, the proper transformation of the design point is to be carried out prior to the model evaluation. Note also that the results of the execution should be saved inside a text file with rows corresponding to the results of each model execution.

In general, the number of model evaluations, both for trajectory and radial scheme, are related to the number of replications (r) and the number of input dimensions (k):

$n_{runs} = r \times (k + 1)$

Analyzing the I/O of Morris Experimental Runs

The last step in conducting the Morris screening analysis is to compute the statistics of the elementary effects for each input. The minimum requirements for this computation are the design file and its corresponding model output. If necessary, the rescaled design file can also be specified to compute the standardized version of the elementary effects. The driver script to analyze the inputs/outputs of Morris experimental run can be invoked from the terminal using the following command:

```
> gsa_morris_analyze -in <the normalized inputs file> \
                     -ir <the rescaled inputs file> \
                     -o <the model/function outputs file> \
                     -output <the results of the analysis output file> \
                     -mc <Verbose model error checking> \
```

Brief explanation on this parameter can be shown using the following command:

```
> gsa_morris_analyze --help
```

By default, the naming convention of the results of the analysis output file is:

```
<normalized inputs filename>-<model outputs file>.csv
```

The `-mc` flag is to verbosely give report on the model specification consistency. This includes:

1. Number of input dimensions in the design file
2. Number of blocks/replications in the design file
3. Total number of runs
4. Type of design

5. Number of levels and grid jump size (trajectory scheme only)
6. Rescaled inputs if specified

This information (except number 6) is directly inferred from the content of the normalized design file.

Example

As an example, consider that a 4-parameter model was evaluated according to the OAT design in the file `trajectory_10_4_10.csv`. The output of the model was saved inside a file `4paramsFunction.csv`.

To compute the statistics of the elementary effects of this I/O pair, invoke the following command:

```
> gsa_morris_analyze -in ./trajectory_10_4_10.csv -o ./4paramsFunction.csv -mc
```

The flag `-mc` will result in verbose reporting of the model specification:

```
Number of Input Dimensions    = 4
Number of Blocks/Replications = 10
Total Number of Runs          = 50
Type of Design                = trajectory
Number of levels (trajectory) = 10 (Delta = 0.5556)
Rescaled Inputs               = None
```

The results of the analysis is saved inside the file `trajectory_10_4_10-4paramsFunction.csv` with the following contents:

```
# mu, mu_star, std_dev, std_mu, std_mu_star, std_std_dev
9.738333e+01,9.738333e+01,3.452392e+01,0.000000e+00,0.000000e+00,0.000000e+00
6.596656e+01,6.596656e+01,3.181203e+01,0.000000e+00,0.000000e+00,0.000000e+00
3.814122e+01,3.814122e+01,2.275404e+01,0.000000e+00,0.000000e+00,0.000000e+00
2.529044e+01,2.529044e+01,1.261223e+01,0.000000e+00,0.000000e+00,0.000000e+00
```

Each column corresponds to the appropriate sensitivity measure as indicated above. Note that the standardized version of the elementary effects are taken to be zero as the rescaled input file was not specified. The parameter is ordered according to the design matrix file (the first column is the first parameter, etc.)

Sobol' Sensitivity Indices

Sobol' sensitivity indices in `gsa_module` are estimated by Monte Carlo procedure. The output of the model has to be generated using a particular design called the Sobol'-Saltelli design. To estimate the main- and total-effect indices, the number of model evaluations, are related to the number of Monte Carlo samples (N) and the number of input dimensions (k):

$n_{runs} = N \times (k + 2)$

A set of $k + 2$ design matrix files can be simultaneously generated using the following command:

```
> gsa_sobol_generate -n <number of samples> \
                     -d <number of dimensions> \
                     -ss <sampling scheme {srs, lhs, sobol}> \
                     -o <output filename header> \
                     -sep <the delimiter for the files> \
                     -int <include design matrices to estimate 2nd order> \
                     -s <seed number, for SRS and LHS only> \
                     -dirnum <direction number file, Sobol' only>
```

Brief explanation of these parameters can be shown by invoking:

```
> gsa_sobol_generate --help
```

The table below lists the complete options/flag in detail with their respective default values.

No.	Short Name	Long Name	Type	Required	Description	Default
1	-h	–help	flag	No	Show help message	False
2	-n	–num_samples	integer	Yes	The number of samples/design points	None
3	-d	–num_dimensions	integer	Yes	The number of dimensions	None
4	-ss	–sampling_scheme	string	No	The method to generate sample {srs, lhs, sobol}	srs
5	-o	–output_header	string	No	The output filename header	see below
6	-sep	–delimiter	string	No	The delimiter for the file {csv, tsv, txt}	csv
7	-int	–interaction	flag	No	Flag to also generate matrices to estimate 2nd-order indices	False
8	-s	–seed_number	integer	No	The random seed number (only for LHS and Sobol)	None
9	-dirnum	–direction_numbers	string	No	The path to Sobol' sequence generator	None
10	-V	–version	flag	No	Show the program's version number and exit	False

Note that options number 8 is valid only for SRS- and LHS-based samples, while option number 9 is valid only for Sobol-based samples. Without specifying the output filename header explicitly, the design matrices file will be produced with the following naming convention:

```
> <method>_<num_samples>_<num_dimensions>_<matrix_ID>.cs
```

Example

As an example, by invoking the following command in the terminal:

```
> gsa_sobol_generate -n 20 -d 3
```

will produce 5 design matrix files with the following names:

```
.
|
+---
+--- srs_20_3_a.csv
+--- srs_20_3_ab1.csv
+--- srs_20_3_ab2.csv
+--- srs_20_3_ab3.csv
+--- srs_20_3_b.csv
```

Each file has the same structure as the design matrix file produced in *General Purpose Design of Experiment*.

Following the convention of Sobol'-Saltelli design, 2 design of experiments of the same size $N \times k$ are first generated. These are the ones with `matrix_ID` `a` and `b`. Afterward each column of the matrix A is replaced by a column from matrix B. For example, `matrix_ID` `ab1` corresponds to the matrix A whose the first column has been replaced by the first column of matrix B.

The model then has to be evaluated using the parameters values listed in each of these design matrix files.

Theory and Implementation

Sensitivity Analysis: Local vs. Global

An essential part of model development and assessment is properly describing and understanding the impact of model parameter variations on the model prediction. Sensitivity analysis (SA) is an important methodological step in that context[1]. SA is the process of investigating the role of input parameters in determining the model output[2]. It seeks to quantify the importance of each model input parameter on the output.

Various classifications exist in the literature to categorize SA techniques[3][4][5][6][2]. In the review by Ionescu-Bujor and Cacuci[4][5], SA techniques are classified with respect to their scope (local vs. global) and to their framework (deterministic vs. statistical). In the review of SA methods by Iooss and Lemaître[2], and the work by Saltelli et al.[6] and by Santner et al.[7], the statistical framework is implicitly assumed deriving ideas from design of experiment, and the classification is based on the parameter space of interest (local vs. global).

Local analysis is based on calculating the effect on the model output of small perturbations around a nominal parameter value. Often the perturbation is done one parameter at a time thus approximating the first-order partial derivative of the model output with respect to the perturbed parameter. The derivative can be computed through efficient adjoint formulation[8][9] capable of handling large number of parameters.

Besides being numerically efficient, sensitivity coefficients obtained from local deterministic sensitivity analysis have the advantage of being intuitive in their interpretation, irrespective of the method employed[10]. The intuitiveness stems from the aforementioned equivalence to the derivative of the output with respect to each parameter[4] around a specifically defined point (i.e., nominal parameter values). Thus the coefficients can be readily compared over different modeled systems, independently of the range of parameters variations.

The global analysis, on the other hand, seeks to explore the input parameters space across its range of variation and then quantify the input parameter importance based on a characterization of the resulting output response surface. In global deterministic framework [4]_[9]_, the characterization is aimed at the identification of the system's critical points (e.g., maxima, minima, saddle points, etc.). In statistical global methods[6][11], the characterization is aimed at measuring the dispersion of the output based on variance[12][13], correlation[14], or elementary effects[15].

Due to the different characterizations, the global statistical framework can potentially give spurious results not comparable to the results from local method as there is no unique definition of sensitivity coefficient provided by different global methods[10]. In some cases, different methods can give different and inconsistent parameters importance rank-

[1] T.G. Trucano, L.P. Swiler, T. Igusa, W.L. Oberkampf, and M. Pilch, "Calibration, validation, and sensitivity analysis: What's what," Reliability Engineering and System Safety, vol. 91, no. 10-11, pp. 1331-1357, 2006.

[2] B. Iooss and P. Lemaitre, "A review on global sensitivity analysis methods," in Uncertainty Management in Simulation-Optimization of Complex Systems, pp. 101-122, Springer, 2015.

[3] H.C. Frey and S.R. Patil, "Identification and Review of Sensitivity Analysis Methods," Risk Analysis, vol. 22, no. 3, pp. 553–578, 2002.

[4] M. Ionescu-Bujor and D.G. Cacuci, "A Comparative Review of Sensitivity and Uncertainty Analysis of Large-Scale Systems - I: Deterministic Methods," Nuclear Science and Engineering, vol. 147, pp. 189-203, 2004.

[5] D.G. Cacuci and M. Ionescu-Bujor, "A Comparative Review of Sensitivity and Uncertainty Analysis of Large-Scale Systems - II: Statistical Methods," Nuclear Science and Engineering, vol. 147, pp. 204-217, 2004.

[6] A. Saltelli et al., "Global Sensitivity Analysis. The Primer," West Sussex, John Wiley & Sons, 2008.

[7] T.J. Santner, B.J. Williams, and W.I. Notz, "The Design and Analysis of Computer Experiments," Springer, 2003.

[8] D.G. Cacuci, "Sensitivity and Uncertainty Analysis, Volume I: Theory," Chapman & Hall/CRC, 2003.

[9] D.G. Cacuci and M. Ionescu-Bujor, "Sensitivity and Uncertainty Analysis, Data Assimilation, and Predictive Best-Estimate Model Calibration," Handbook of Nuclear Engineering, Springer, pp. 1913-2051, 2010.

[10] S. Razavi and H.V. Gupta, "What do we mean by sensitivity analysis? The need for comprehensive characterization of "global" sensitivity in Earth and Environmental systems models," Water Resources Research, vol. 51, pp. 3070-3092, 2015.

[11] A. Saltelli et al., "Sensitivity Analysis in Practice: a Guide to Assessing Scientific Models," West Sussex, John Wiley & Sons, 2004.

[12] I. M. Sobol, "Global Sensitivity Analysis for nonlinear mathematical models and their Monte Carlo estimates," Mathematics and Computers in Simulation, vol. 55, no. 1-3, pp. 271-280, 2001.

[13] R. Cukier, H. Levine, K. Shuler, "Nonlinear Sensitivity Analysis of Multiparameter Model Systems," Journal of Computational Physics, vol. 26, pp. 1-42, 1978.

[14] J.C. Helton, "Uncertainty and sensitivity analysis techniques for use in performance assessment for radioactive waste disposal," Reliability Engineering & System Safety, vol. 42, pp. 327-367, 1993.

[15] Max D. Morris, "Factorial Sampling Plans for Preliminary Computational Experiments", Technometrics, Vol. 33, No. 2, pp. 161-174, 1991.

ing[68]. Furthermore, the result of the analysis can be highly dependent to the assumed input parameters probability distribution and/or their range of variation[59].

Yet, despite the aforementioned shortcomings, the global statistical framework has three particular attractive features relevant to the present study. First, the statistical method for sensitivity analysis is non-intrusive in the sense that minimal or no modification to the original code is required. In other words, the code can be taken as a black box and the analysis is focused on the input/output relationship[6] of the code. This is the case especially in comparison to adjoint-based sensitivity[16][17] which is a highly efficient and accurate method applicable to a large number of parameters, provided that the code is designed/modified for adjoint analysis.

Second, no a priori knowledge on the model structure (linearity, additivity, etc.) is required. Depending on the model complexity and as the parameter variation range can be large, the linearity or additivity assumption might not hold.

Third and finally, the choice of a statistical framework for sensitivity analysis fits the Monte Carlo (MC)-based uncertainty propagation method widely adopted in nuclear reactor evaluation models cite{Boyack1990, Nutt2004, Wallis2007, Glaeser2008}. The method prescribes that the uncertain model input and parameters (modeled as random variables) should be simultaneously and randomly perturbed across their range of variations. Multiple randomly generated input values are then propagated through the code to quantify the dispersion of the prediction (e.g., peak cladding temperature) which serves as a measure of the prediction reliability. Statistical global sensitivity analysis thus complements the propagation step by addressing the follow-up question on the identification of the most important parameters in driving the prediction uncertainty.

References

Designs of Experiment

This section is under preparation

Morris Screening Method

Screening methods are used to rank the importance of the model parameters using a relatively small number of model executions[1]. However, they tend to simply give qualitative measures. That is, meaningful information resides in the rank itself but not in the exact importance of the parameters with respect to the output. These methods are particularly valuable in the early phase of a SA to identify the non-influential parameters of a model, which then could be safely excluded from further detailed analysis. This exclusion step is important to reduce the size of the problem especially if a more expensive method is to be applied at the next step. In this work, attention was paid to a particular screening method proposed by Morris[2].

Elementary Effects

Consider a model with K parameters, where $\vec{x} = (x_1, x_2, ..., x_K)$ is a vector of parameter value mapped onto the unit hypercube and $y(\vec{x})$ is the model output evaluated at point $\vec{x}$. The elementary effect of parameter *k* is defined as follow[2] ,

$$EE_k = \frac{Y(x_1,x_2,...,x_k+\Delta,...,x_K)-Y(x_1,x_2,...,x_K)}{\Delta}$$

Where Δ, the *grid jump*, is chosen such that $\vec{x} + \Delta$ is still in the specified domain of parameter space; Δ is a value in $\frac{1}{p-1}, \ldots, 1 - \frac{1}{p-1}$ where p is the number of levels that partitions the model parameter space into a uniform grid

[16] D.G. Cacuci and M. Ionescu-Bujor, "Adjoint Sensitivity Analysis of the RELAP5/MOD3.2 Two-Fluid Thermal-Hydraulic Code System - I: Theory," Nuclear Science and Engineering, vol. 136, pp. 59-84, 2000.

[17] M. Ionescu-Bujor and D.G. Cacuci, "Adjoint Sensitivity Analysis of the RELAP5/MOD3.2 Two-Fluid Thermal-Hydraulic Code System - II: Applications," Nuclear Science and Engineering, vol. 136, pp. 85-121, 2000.

[1] A. Saltelli et al., "Sensitivity Analysis in Practice: A Guide to Assessing Scientific Models," John Wiley & Sons, Ltd. United Kingdom (2004).

[2] Max D. Morris, "Factorial Sampling Plans for Preliminary Computational Experiments", Technometrics, Vol. 33, No. 2, pp. 161-174, 1991.

of points at which the model can be evaluated. The grid constructs a finite distribution of size $p^{K-1}[p - \Delta(p-1)]$ elementary effects per input parameters.

The key idea of the original Morris method is in initiating the model evaluations from various "nominal" points $\vec{x}$ randomly selected over the grid and then gradually advancing one grid jump at a time between each model evaluation (one at a time), along a different dimension of the parameter space selected randomly.

Statistics of Elementary Effects and Sensitivity Measure

Consider now that an n_R number of elementary effects (or *replicates*) associated with the k'th parameter have been sampled from the finite distribution of EE_k. The statistical summary of EE_k from the sampled trajectories can be calculated. The first is the arithmetic mean defined as,

$\mu_k = \frac{1}{n_R}\Sigma_{r=1}^{n_R} EE_k^r$

The second statistical summary of interest is the standard deviation of the elementary effect associated with the k'th parameter from all the trajectories,

$\sigma_k = \sqrt{\frac{1}{n_R}\Sigma_{r=1}^{n_R}(EE_k^r - \mu_k)^2}$

The standard deviation gives an indication of the presence of nonlinearity and/or interactions between the *k*'th parameter and other parameters. As a change in a parameter value might have a changing sign on the output and thus result in a cancelation effect (as can be the case for a nonmonotonic function), Campolongo et al.[4] proposed the use of the mean of the absolute elementary effect to circumvent this issue. It is defined as

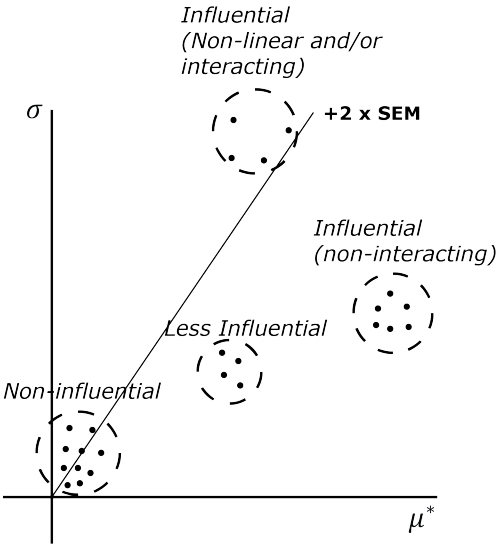
$\mu_k^* = \frac{1}{n_R}\Sigma_{r=1}^{n_R}|EE_k^r|$

The three aforementioned statistical summaries, when evaluated over a large number of trajectories n_R, can provide global sensitivity measures of the importance of the k'th parameter. As indicated by Morris[2] there are three possible categories of parameter importance:

1. Parameters with noninfluential effects, i.e., the parameters that have relatively small values of both μ_k (or μ_k^*) and σ_k. The small values of both indicate that the parameter has a negligible overall effect on the model output.
2. Parameters with linear and/or additive effects, i.e., the parameters that have a relatively large value of μ_k (or μ_k^*) and relatively small value of σ_k. The small value of σ_k and the large value of μ_k (or μ_k^*) indicate that the variation of elementary effects is small while the magnitude of the effect itself is consistently large for the perturbations in the parameter space.
3. Parameters with nonlinear and/or interaction effects, i.e., the parameters that have a relatively small value of μ_k (or μ_k^*) and a relatively large value of σ_k. Opposite to the previous case, a small value of μ_k (or μ_k^*) indicates that the aggregate effect of perturbations is seemingly small while a large value of σ_k indicates that the variation of the effect is large; the effect can be large or negligibly small depending on the other values of parameters at which the model is evaluated. Such large variation is a symptom of nonlinear effects and/or parameter interaction.

Such classification makes parameter importance ranking and, in turn, screening of non-influential parameters possible. However, the procedure is done rather qualitatively, and this is illustrated in the figure below, which depicts a typical parameter classification derived from visual inspection of the elementary effect statistics on the σ_k versus μ_k^* plane.

[4] F. Campolongo, A. Saltelli, and J. Cariboni, "From Screening to Quantitative Sensitivity Analysis. A Unified Approach," Computer Physics Communications, Vol. 192, pp. 978 - 988, 2011.



The notions of influential and non-influential parameters are based on the relative locations of those statistics in the plane. Typically, the non-influential ones are clustered closer to the origin (relative to the more influential ones) with a pronounced boundary such as depicted in the figure. Admittedly, if these statistics are spread uniformly across the plane, the distinction would be more ambiguous (in this case, a more advanced classification such as the ones based on clustering techniques might be helpful). Furthermore, for a parameter with large μ_k and σ_k, the method cannot distinguish between non-linearity effects from parameter interactions on the output.

Design of Experiment for Screening Analysis

There are two available experimental designs for to carry out the Morris screening method in `gsa-module`: the trajectory design and radial OAT design.

Trajectory Design (*Winding Stairs*)

Trajectory design is the original Morris implementation of the design of experiment for screening design[2]. Essentially, it is a randomized one-at-a-time design where each parameter is perturbed once, similar to that of the *winding stairs* design proposed by Jansen et al.[3]. The most important feature of trajectory design is that it does not return to the original base point after perturbation, but continue perturbing another dimension for the last perturbed point. This ensures more efficient parameter space exploration although requires additional user-defined parameter called *level*[4].

A trajectory design is defined by the number of trajectories (*r*), the number of levels (*p*), and the number of model parameters (*k*). Each trajectories evaluate the model *(k + 1)* times so the economy of it in computing the elementary effects statistics is *r * (k+1)* code runs.

A randomized trajectory design matrix is given by b^* ([2],[5]),

$$b^* = (x^* + \frac{\Delta}{2} \times ((2 \times b - j_k) \times d^* + j_k)) \times p^*$$

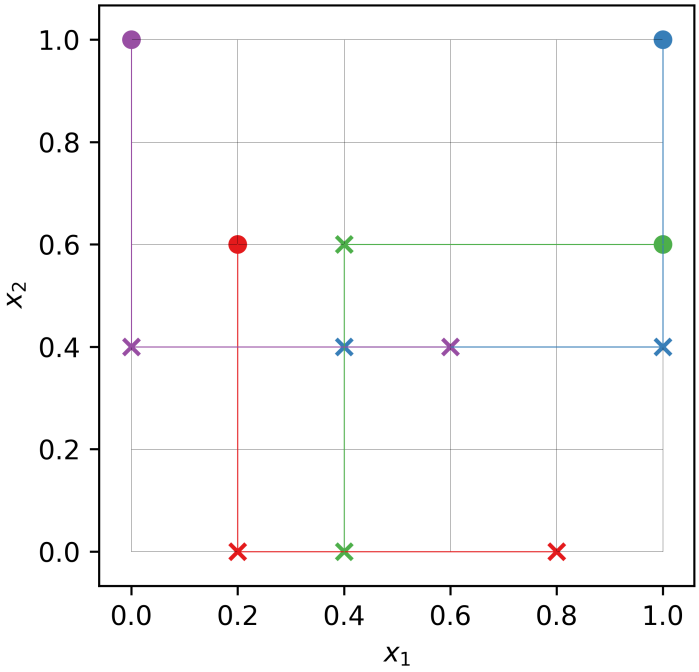
- b: a strictly lower triangular matrix of 1s, with dimension of *(k + 1)-by-k*
- x^*: Random starting point in the parameter space, with dimension of *(k + 1)-by-k* - each row is the same.

[3] Michiel J.W. Jansen, Walter A.H. Rossing, and Richard A. Daamen, "Monte Carlo Estimation of Uncertainty Contributions from Several Independent Multivariate Sources," in Predictability and Nonlinear Modelling in Natural Sciences and Economics, Dordrecht, Germany, Kluwer Publishing, 1994, pp. 334 - 343.

[5] A. Saltelli et al., "Global Sensitivity Analysis. The Primer," West Sussex, John Wiley & Sons, 2008, pp. 114

- d^*: a k-dimensional diagonal matrix which each element is either +1 or -1 with equal probability. This matrix determines whether a parameter value will decrease or increase.
- p^*: *k-by-k* random permutation matrix in which each row contains one element equal to 1, all others are 0, and no two columns have 1s in the same position. This matrix determines the order in which parameters are perturbed.
- j_k: *(k + 1)-by-k* matrix of 1s
- Δ: factorial increment in a diagonal matrix of *(k + 1)-by-(k + 1)*

The following is an example of a trajectory design in 2-dimensional input space with 4 trajectories (or *replicates*). The input parameter space is uniformly divided into 6 levels. The filled circles are the random base (nominal) points from which the random perturbation of the same size (i.e., the grid jump) is carried out one-at-a-time.



Radial Design

Radial design is a design for screening analysis proposed in[4]. Similar to trajectory design it is based on an extension of one-at-a-time design. In the implementation of[4], Sobol' quasi-random sequence is used as the basis. Its main advantage over the trajectory design is that the specification of input discretization level by user is no longer required. Furthermore, the grid jump will also be varying from one input dimension to another, and from replicate to replicate incorporating additional possible sources of variation in the method.

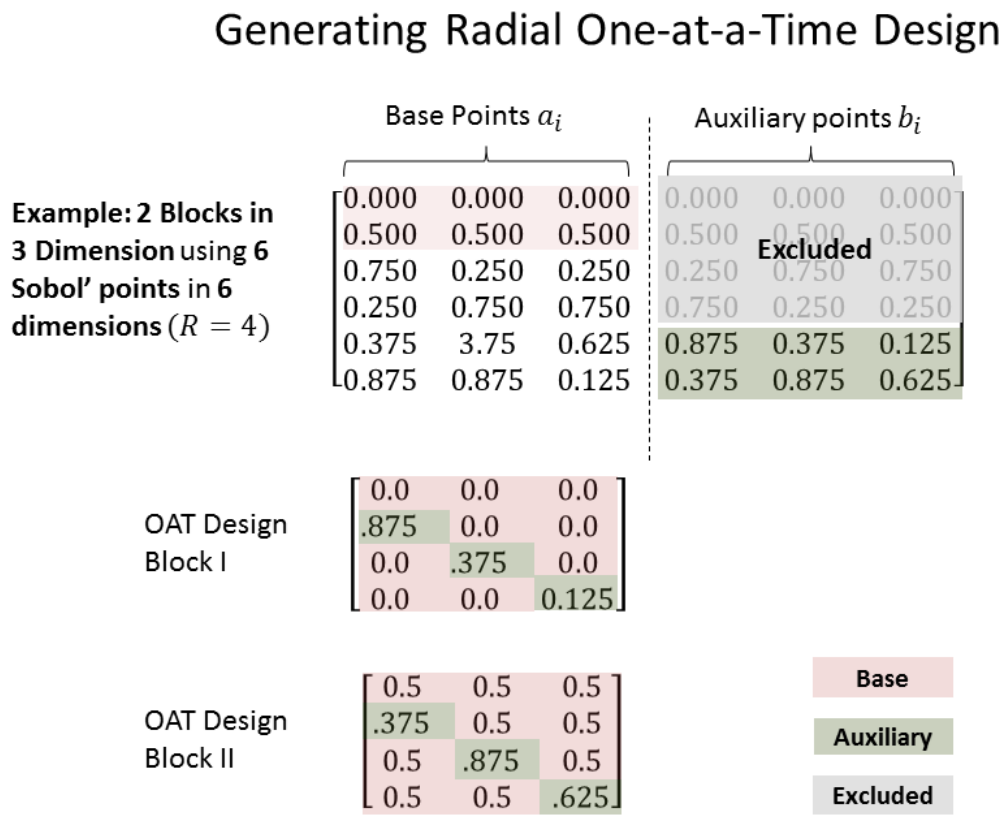
The procedure to generate radial design of *r* replicates is as follow:

1. Generate Sobol' sequence with dimension *(r+R, 2*k)*. *R* is the shift to avoid repetition in the sequence. The value of *R* is recommended to be fixed at *4* following[4], but see *Choosing Shifting Value* below for additional comments.
2. The first half of the matrix up to the *r*-th row will serve as the base points: $a_i = (x_{i,1}, x_{i,2}, \ldots x_{i,k})$; $i = 1, \ldots r$. The second half of the matrix, starting from the *R+1*-th row will serve as the auxiliary points, from

which the perturbed states of the base point are created: $b_i = (x_{R+i,k+1}, x_{R+i,k+2}, \ldots x_{R+i,2k})$; $i = 1, \ldots r$

3. For each row of the base points, create a set of perturbed states by substituting the value at each dimension by the value from the auxiliary points at the same dimension, one at a time. For each base point, there will be additional *k* perturbed points. For instance the 1st perturbed point of the *i*-th base point *a_i* is $a_i^{*,1} = (x_{R+i,k+1}, x_{i,2}, \ldots x_{i,k})$, while the second is $a_i^{*,2} = (x_{i,1}, x_{R+i,k+2}, \ldots x_{i,k})$. In general the *j*-th perturbed point of the *i*-th base point is $a_i^{*,j} = (x_{i,1}, \ldots x_{R+i,k+j}, \ldots x_{i,k})$.
4. A single elementary effect for each input dimension can be computed on the basis of function evaluations at *k+1* points: 1 base point and *k* perturbed points.
5. Repeat the process until the requested *r* replications have been constructed.

An illustration of radial OAT design generation based on Sobol' sequence can be seen in the figure below.



As such the radial design has the same economy as the trajectory design, that is *r * (k+1)* computations for a *k*-dimensional model with *r* replications. The computation of the elementary effect EE_i, however, is slightly different due to the fact that now the grid jump differs for each input dimension at each replication.

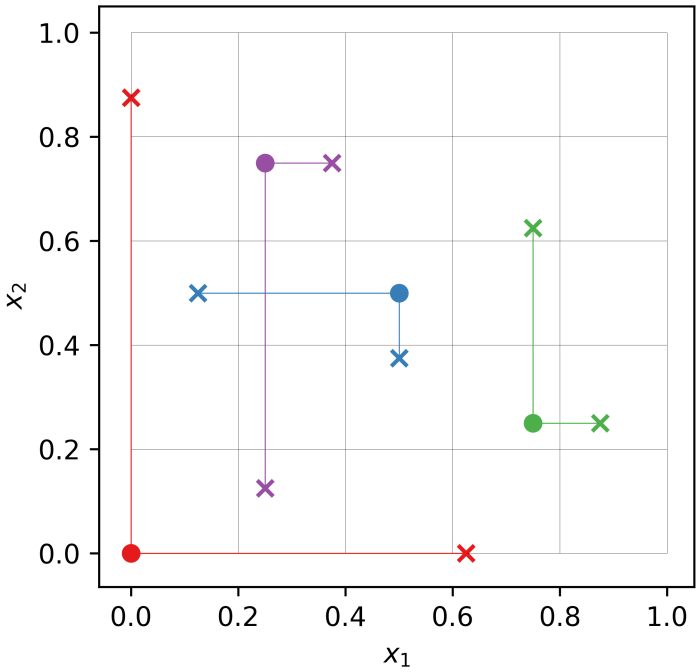
$$EE_j^i = \left| \frac{y(a_i^{*,j}) - y(a_i)}{x_{R+i,k+j} - x_{i,j}} \right|$$

- $y(a_i^{*,j})$: function value at *j*-th perturbed point of the *i*-th replicate.
- $y(a_i)$: function value at the base point of the *i*-th replicate.

- $x_{R+i,k+j}$: the perturbed input at dimension *j* of the *i*-th replicate.
- $x_{i,j}$: the base input at dimension *j* of the *i*-th replicate.

As can be seen the average over many replications of the elementary effect defined above will automatically yield μ^*.

The following is an example of a radial design in 2-dimensional input space with 4 base points (filled circles), located not necessarily in a specific grid. The perturbations are carried out from these base points (crosses). The size of the perturbation differs from input dimension to input dimension and from replicate to replicate.



Choosing Shifting Value

As mentioned the recommendation given by[4] for the value of *R* is *4*. This value reflects the fact that the a sample of Sobol' sequence across dimension tends to repeat values, especially in the first several rows. For example, the first two Sobol' samples used here have the values of *0.0* and *0.5* in all of the dimensions. If such repetition in value happened one or more rows in the Δ matrix will be zero (so is the ΔY vector), and cause the system of linear equation to be under-determined.

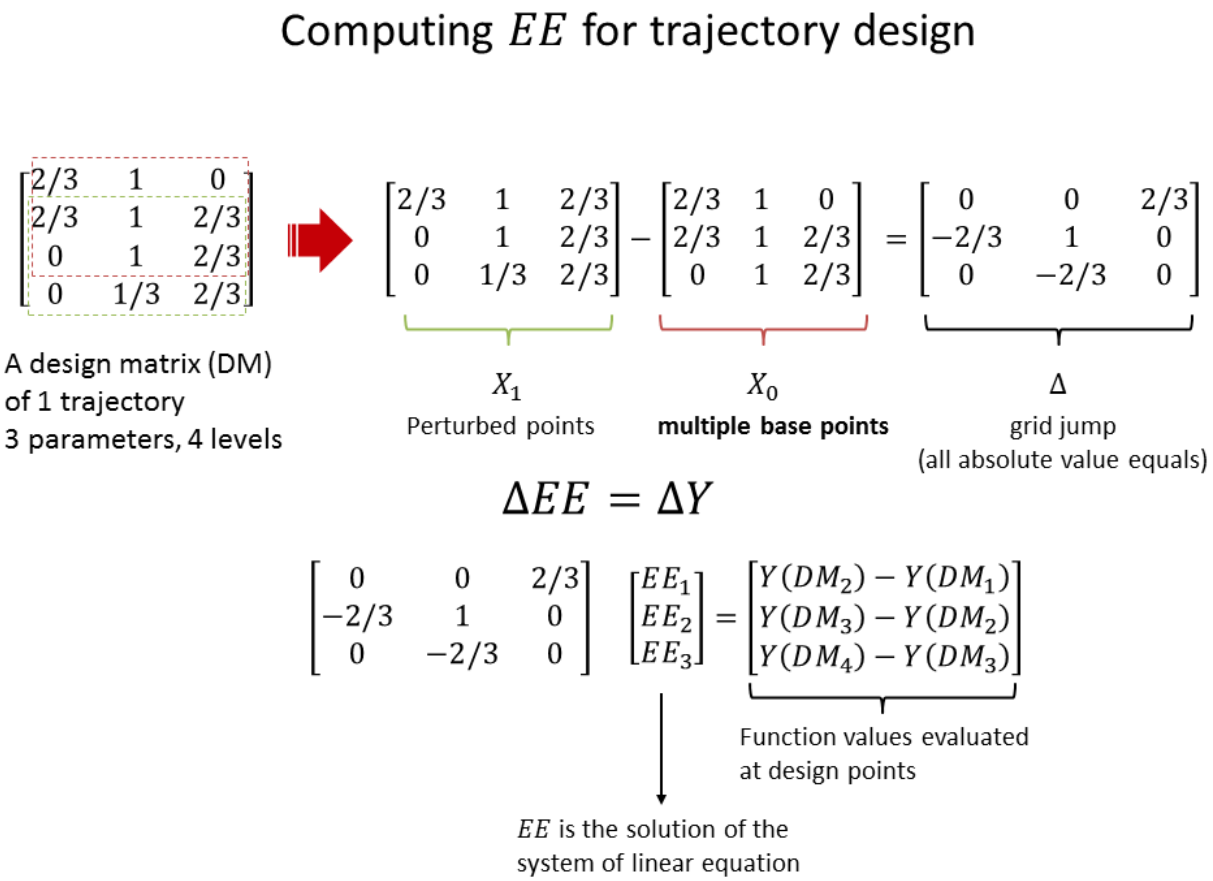
But except for the obvious repetitions of values in different dimensions in the first several samples any other repetitions cannot be excluded to reoccur down the line of samples. As such the value of *R* has to be picked carefully and from our experience this value is highly dependent on the number of samples and/or dimension. Yet, the of the main points of using radial design in the first place was to avoid specifying the number of levels *p*. Choosing *R* for different number of samples and/or dimensions definitely defeat the purpose of using radial design.

A pragmatic solution for this problem, which is adopted here, is to check whether a given auxiliary point has the same value with the base point in one or more dimension, every time a block of one-at-a-time design is generated. If it has then use the next auxiliary point instead. Finally, to replace the missing auxiliary point, an additional point is generated using the Sobol' sequence.

Miscellaneous Topics

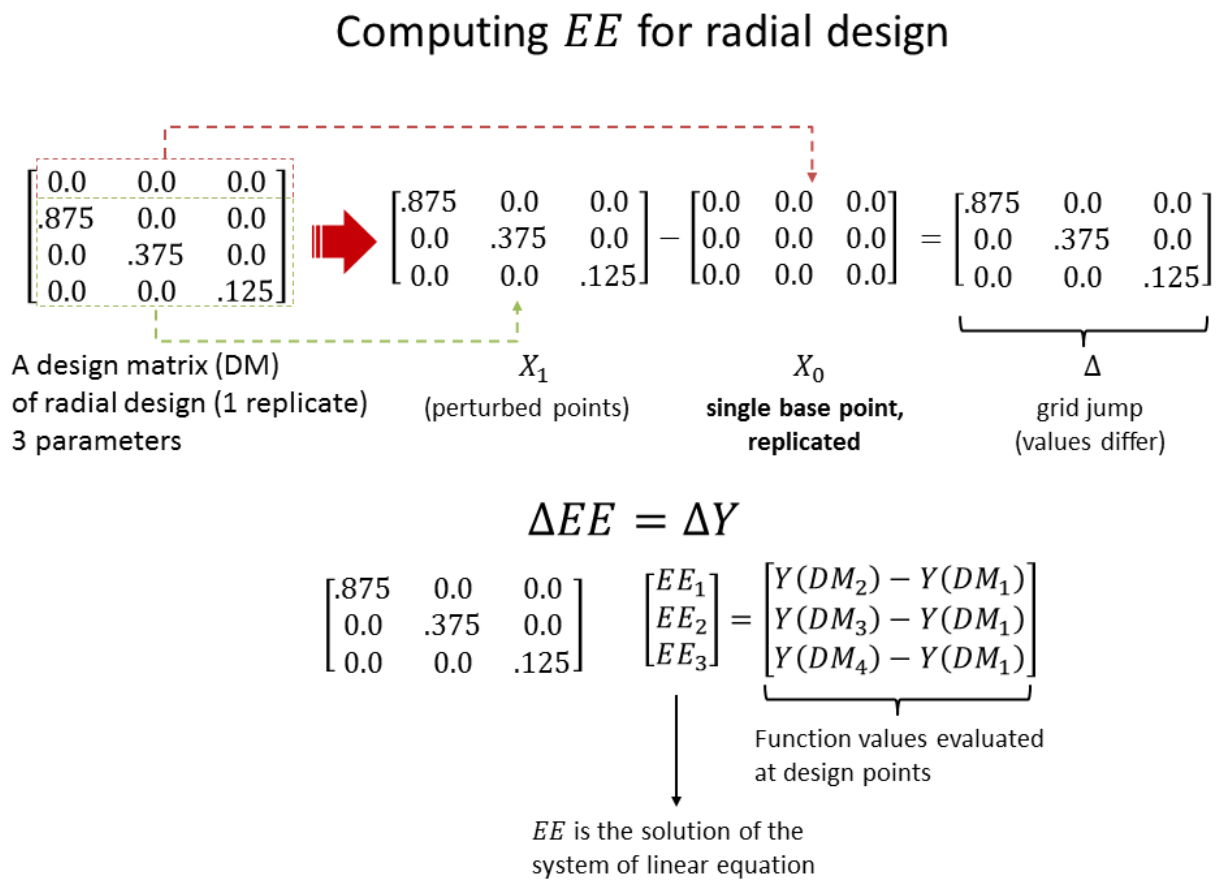
Computation of the Elementary Effect

In `gsa-module`, computing the elementary effect for each replications is achieved by using matrix algebra, which is similar to the implementation in[6]. There is slight difference between the computation of elementary effects for trajectory design and radial design. The following figure illustrate the computation of all the elementary effects of a single replicate for 3-parameter model using trajectory design with 4 levels.



The following figure illustrate the same computation of a single replicate for 3-parameter model using radial design (no number of levels specification needed).

[6] Jon D. Herman, SALib [Source Code], March 2014, https://github.com/jdherman/SALib



The statistics of the elementary effects are eventually computed after the same procedure are repeated for many replications.

Presenting the Results of the Analysis

Standardized Elementary Effect

In the original implementation of Morris method[2], the input parameter is normalized, that is all the parameters values lie between 0, 1. Furthermore, following the suggestion by Saltelli et al.[5], the grid jump size is kept constant for a given number of levels for all parameters. As such, the method is prone to misrank the important parameters if there is a vast difference in the original scale of various parameters (e.g., [0,1] in one parameter, [10,100] in another, etc.). The normalized scale of [0,1] would then be biased to the parameter who has the largest scale of variation. To compare the elementary effect in a common ground taking into account the original scale of variation for each parameter, it is advised in[7] to scale the elementary effect with the standard deviation of the input σ_{x_i} and of the output σ_y,

$$SEE_i = \frac{\sigma_{x_i}}{\sigma_y} \frac{\Delta y}{\Delta x_i}$$

In `gsa-module`, the standardized elementary effect is automatically computed if the rescaled input parameters values are specified. It is used to compute the standard deviation for each of the parameters taking into account the original scale of variation of each.

[7] G. Sin and K. V. Gernaey, "Improving the Morris Method for Sensitivity Analysis by Scaling the Elementary Effects," in Proc. 19th European Symposium on Computer Aided Process Engineering, 2009

Optimized Trajectory Design

References

Sobol' Sensitivity Indices

Variance-based methods for global sensitivity analysis use variance as the basis to define a measure of input parameter influence on the overall output variation[1]. In a statistical framework of sensitivity and uncertainty analysis, this choice is natural because variance (or standard deviation, a related concept) is often used as a measure of dispersion or variability in the model prediction[2]. A measure of such dispersion, in turn, indicates the precision of the prediction due to variations in the input parameters.

This section of the documentation discusses the origin of Sobol' sensitivity indices and the method to estimate them by Monte Carlo simulation.

High-Dimensional Model Representation

Consider once more a mathematical model $f : \mathbf{x} \in [0,1]^D \mapsto y = f(\mathbf{x}) \in \mathbb{R}$. The high-dimensional model representation (HDMR) of $f(\mathbf{x})$ is a linear combination of functions with increasing dimensionality[3],

$$f(\mathbf{x}) = f_o + \sum_{d=1,2,\dots D} f_d(x_d) + \sum_{1 \le d < e \le D} f_{d,e}(x_d, x_e) + \cdots + f_{1,2,\cdots,D}(x_1, x_2, \cdots, x_D) \tag{1.1}$$

where f_o is a constant.

The representation in the above equation is unique given the following condition[4]:

$$\begin{aligned} &\int_0^1 f_{i1,i2,\cdots is}(x_{i1}, x_{i2}, \cdots, x_{is}) d_{x_{im}} = 0 \\ &\quad \text{for} \quad m = 1, 2, \cdots, s; \quad 1 \le i_1 < i_2 < \cdots < i_s \le D; \\ &\text{and} \quad s \in 1, \cdots, D \end{aligned} \tag{1.2}$$

Assume now that $\mathbf{X}$ is a random vector of independent and uniform random variables over a unit hypercube $\{\Omega = \mathbf{x} | 0 \le x_i \le 1; i = 1, \cdots D\}$ such that

$$Y = f(\mathbf{X})$$

where Y is a random variable, resulting from the transformation of random vector $\mathbf{X}$ by function f. Using Eq. (1.2) to express each term in Eq. (1.1), it follows that

$$\begin{aligned} f_o &= \mathbb{E}[Y] \\ f_d(x_d) &= \mathbb{E}_{\sim d}[Y|X_d] - \mathbb{E}[Y] \\ f_{d,e}(x_d, x_e) &= \mathbb{E}_{\sim d,e}[Y|X_d, X_e] - \mathbb{E}_{\sim d}[Y|X_d] - \mathbb{E}_{\sim e}[Y|X_e] - \mathbb{E}[Y] \end{aligned} \tag{1.3}$$

The same follows for higher-order terms in the decomposition. In Eq. (1.3), $\mathbb{E}_{\sim e}[Y|X_e]$ corresponds to the conditional expectation operator, and the $\sim \circ$ in the subscript means that the integration over the parameter space is carried out over all parameters except the specified parameter in the subscript. For instance, $\mathbb{E}_{\sim 1}[Y|X_1]$ refers to the conditional

[1] Dan G. Cacuci and Mihaela Ionescu-Bujor, "A Comparative Review of Sensitivity and Uncertainty Analysis of Large-Scale Systems - II: Statistical Methods," Nuclear Science and Engineering, vol. 147, no. 3, pp. 204-217, 2004.

[2] A. Saltelli et al., "Global Sensitivity Analysis. The Primer," West Sussex, John Wiley & Sons, 2008.

[3] Genyuan Li, Carey Rosenthal, and Herschel Rabitz, "High Dimensional Model Representations," The Journal of Physical Chemistry A, vol. 105, no. 33, pp. 7765-7777, 2001.

[4] I. M. Sobol, "Global Sensitivity Analysis for nonlinear mathematical models and their Monte Carlo estimates," Mathematics and Computers in Simulation, vol. 55, no. 1-3, pp. 271-280, 2001.

mean of Y given X_1, and the integration is carried out for all possible values of parameters in $\mathbf{x}$ except x_1. Note that because X_1 is a random variable, the expectation conditioned on it is also a random variable.

Assuming that f is a square integrable function, applying the variance operator on Y results in

$$\mathbb{V}[Y] = \sum_{d=1}^{D} \mathbb{V}[f_d(x_D)] + \sum_{1 \leq d < e \leq D} \mathbb{V}[f_{d,e}(x_d, x_e)] + \cdots + \mathbb{V}[f_{1,2,\cdots,D}(x_1, x_2, \cdots, x_D)] \tag{1.4}$$

Sobol' Sensitivity Indices

Division by $\mathbb{V}[Y]$ aptly normalizes Eq. (1.4)

$$1 = \sum_{d=1}^{D} S_d + \sum_{1 \leq d < e \leq D} S_{d,e} + \cdots + S_{1,2,\cdots,D}$$

The Sobol' main-effect sensitivity index S_d is defined as,

$$S_d = \frac{\mathbb{V}_d[\mathbb{E}_{\sim d}[Y|X_d]]}{\mathbb{V}[Y]} \tag{1.5}$$

The numerator is the variance of the conditional expectation, and the index is a global sensitivity measure interpreted as the amount of variance reduction in the model output if the parameter X_d is fixed (i.e., its variance is reduced to zero).

A closely related sensitivity index proposed by Homma and Saltelli[5] is the Sobol' total-effect index defined as,

$$ST_d = \frac{\mathbb{E}_{\sim d}[\mathbb{V}_d[Y|\mathbf{X}_{\sim d}]]}{\mathbb{V}[Y]} \tag{1.6}$$

The index, also a global sensitivity measure, can be interpreted as the amount of variance left in the output if the values of all input parameters, *except* x_d, can be fixed.

These two sensitivity measures can be related to the objectives of global SA for model assessment as proposed by Saltelli et al.[6]. The main-effect index is relevant to parameter prioritization in the context of identifying the most influential parameter since fixing a parameter with the highest index value would, *on average*, lead to the greatest reduction in the output variation.

The total-effect index, on the other hand, is relevant to parameter fixing (or screening) in the context of identifying the least influential set of parameters since fixing any parameter that has a very small total-effect index value would not lead to significant reduction in the output variation. The use of total-effect index to identify which parameter can be fixed or excluded is similar to that of the elementary effect statistics of the Morris method, albeit more exact but also more computationally expensive to compute. And finally, the difference between the two indices of a given parameter (Eqs. (1.6) and (1.5)) is used to quantify the amount of all interactions involving that parameters in the model output.

The Sobol'-Saltelli Method

Monte Carlo Integration

In principle, the estimation of the Sobol' indices defined by Eqs. (1.5) and (1.6) can be directly carried out using Monte Carlo (MC) simulation. The most straightforward, though rather naive, implementation of MC simulation to conduct the estimation is using two nested loops for the computation of the conditional variance and expectation appeared in both equations.

[5] Toshimitsu Homma and Andrea Saltelli, "Importance Measures in Global Sensitivity Analysis of Nonlinear Models," Reliability Engineering and System Safety, vol. 52, no. 1, pp. 1-17, 1996.

[6] A. Saltelli et al., "Sensitivity Analysis in Practice: a Guide to Assessing Scientific Models," West Sussex, John Wiley & Sons, 2004.

In the estimation of the main-effect index of parameter x_d, for instance, the outer loop samples values of X_d while the inner loop samples values of $\mathbf{X}_{\sim d}$ (anything else other than x_d). These samples, in turn, are used to evaluate the model output. In the inner loop, the mean of the model output (for a given value of X_d but over many values of $\mathbf{X}_{\sim d}$) is taken. Afterward, in the outer loop, the variance of the model output (over many values of X_d) is taken. This approach can easily become prohibitively expensive as the nested structure requires two N^2 model evaluations *per input dimension* for either the main-effect and total-effect indices, while N (the size of MC samples) are typically in the range of $10^2 - 10^4$ for a reliable estimate.

Sobol'[4] and Saltelli[7] proposed an alternative approach that circumvent the nested structure of MC simulation to estimate the indices. The formulation starts by expressing the expectation and variance operators in their integral form. As the following formulation is defined on a unit hypercube of D-dimension parameter space where each parameter is a uniform and independent random variable, explicit writing of the distribution within the integration as well as the integration range are excluded for conciseness.

First, the variance operator shown in the numerator of Eq. (1.5) is written as

$$\begin{aligned}\mathbb{V}_d[\mathbb{E}_{\sim d}[Y|X_d]] &= \mathbb{E}_d[\mathbb{E}^2_{\sim d}[Y|X_d]] - (\mathbb{E}_d[\mathbb{E}_{\sim d}[Y|X_d]])^2 \\ &= \int \mathbb{E}^2_{\sim d}[Y|X_d]dx_d - \left(\int \mathbb{E}_{\sim d}[Y|X_d]dx_d\right)^2\end{aligned} \tag{1.7}$$

The notation $\mathbb{E}_{\sim\circ}[\circ|\circ]$ was already explained in the previous subsection, while $\mathbb{E}_{\circ}[\circ]$ corresponds to the marginal expectation operator where the integration is carried out over the range of parameters specified in the subscript.

Next, consider the term conditional expectation shown in Eq. (1.7), which per definition reads

$$\mathbb{E}_{\sim d}[Y|X_d] = \int f(\mathbf{x}_{\sim d}, x_d)d\mathbf{x}_{\sim d} \tag{1.8}$$

Note that $\mathbf{x} = \{\mathbf{x}_{\sim d}, x_d\}$.

Following the first term of Eq. (1.7), by squaring Eq. (1.8) and by defining a dummy vector variable $\mathbf{x}'_{\sim d}$, the product of the two integrals can be written in terms of a single multiple integrals

$$\begin{aligned}\mathbb{E}^2_{\sim d}[Y|X_d] &= \int f(\mathbf{x}_{\sim d}, x_d)d\mathbf{x}_{\sim d} \cdot \int f(\mathbf{x}_{\sim d}, x_d)d\mathbf{x}_{\sim d} \\ &= \int\int f(\mathbf{x}'_{\sim d}, x_d)f(\mathbf{x}_{\sim d}, x_d)d\mathbf{x}'_{\sim d}d\mathbf{x}_{\sim d}\end{aligned} \tag{1.9}$$

Returning to the full definition of variance of conditional expectation in Eq. (1.7),

$$\begin{aligned}\mathbb{V}_d[\mathbb{E}_{\sim d}[Y|X_d]] = &\int\int f(\mathbf{x}'_{\sim d}, x_d)f(\mathbf{x}_{\sim d}, x_d)d\mathbf{x}'_{\sim d}d\mathbf{x}_{\sim d} \\ &- \left(\int f(\mathbf{x})d\mathbf{x}\right)^2\end{aligned} \tag{1.10}$$

Finally, the main-effect sensitivity index can be written as an integral as follows:

$$\begin{aligned}S_d &= \frac{\mathbb{V}_d[\mathbb{E}_{\sim d}[Y|X_d]]}{\mathbb{V}[Y]} \\ &= \frac{\int\int f(\mathbf{x}'_{\sim d}, x_d)f(\mathbf{x}_{\sim d}, x_d)d\mathbf{x}'_{\sim d}d\mathbf{x} - \left(\int f(\mathbf{x})d\mathbf{x}\right)^2}{\int f(\mathbf{x})^2 d\mathbf{x} - \left(\int f(\mathbf{x})d\mathbf{x}\right)^2}\end{aligned} \tag{1.11}$$

The integral form given above dispenses with the nested structure of multiple integrals in the original definition of main-effect index. The multidimensional integration is over $2 \times D - 1$ dimensions and it is the basis of estimating

[7] A. Saltelli, "Making best use of model evaluations to compute sensitivity indices," Computer Physics Communications, vol. 145, no. 2, pp. 280-297, 2002.

sensitivity index using MC simulation in this implementation, hereinafter referred to as the Sobol'-Saltelli method. The same procedure applies to derive the total effect-index which yields,

$$\begin{aligned} ST_d &= \frac{\mathbb{E}_{\sim d}[\mathbb{V}_d[Y|\mathbf{X}_{\sim d}]]}{\mathbb{V}[Y]} \\ &= \frac{\int f^2(\mathbf{x})d\mathbf{x} - \int\int f(\mathbf{x}_{\sim d}, x'_d) f(\mathbf{x}_{\sim d}, x_d) d\mathbf{x}'_d d\mathbf{x}}{\int f(\mathbf{x})^2 d\mathbf{x} - \left(\int f(\mathbf{x}) d\mathbf{x}\right)^2} \end{aligned} \tag{1.12}$$

For N number of MC samples and D number of model parameters, MC simulation procedure to estimate the sensitivity indices follows the sampling and resampling approach adopted in[4],[5],[7] described in the following.

Procedures

First, generate two $N \times D$ independent random samples from a uniform independent distribution in D-dimension, $[0, 1]^D$:

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1D} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{ND} \end{pmatrix}; \quad B = \begin{pmatrix} b_{11} & \cdots & b_{1D} \\ \vdots & \ddots & \vdots \\ b_{N1} & \cdots & b_{ND} \end{pmatrix} \tag{1.13}$$

Second, construct D additional design of experiment matrices where each matrix is matrix A with the d-th column substituted by the d-th column of matrix B:

$$A_B^1 = \begin{pmatrix} b_{11} & \cdots & a_{1D} \\ \vdots & \ddots & \vdots \\ b_{N1} & \cdots & a_{ND} \end{pmatrix}$$

$$A_B^d = \begin{pmatrix} a_{11} & \cdots & b_{1d} & \cdots & a_{1D} \\ \vdots & \ldots & \vdots & \ldots & \vdots \\ a_{N1} & \cdots & b_{Nd} & \cdots & a_{ND} \end{pmatrix}$$

$$A_B^D = \begin{pmatrix} a_{11} & \cdots & b_{1D} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & b_{ND} \end{pmatrix}$$

Third, rescale each element in the matrices of samples to the actual values of model parameters according to their actual range of variation through iso-probabilistic transformation.

Fourth, evaluate the model multiple times using input vectors that correspond to each row of A, B, and all the A_B^d.

Fifth and finally, extract the quantities of interest (QoIs) from all the outputs and recast them as vectors. The main-effect and total-effect indices are then estimated using the estimators described below.

Monte Carlo Estimators

For the main-effect sensitivity index, two estimators are considered. One is proposed by Saltelli[7], and the other, as an alternative, is proposed by Janon et al.[8]. The latter proved to be more efficient, especially for a large variation around a parameter estimate[8].

The first term in the numerator of Eq. (1.11) is the same for both estimators and is given by

$$\int\int f(\mathbf{x}'_{\sim d}, x_d) f(\mathbf{x}_{\sim d}, x_d) d\mathbf{x}'_{\sim d} d\mathbf{x}_{\sim d} \approx \frac{1}{N} \sum_{n=1}^{N} f(B)_n \cdot f(A_B^d)_n \tag{1.14}$$

[8] A. Janon et al., "Asymptotic normality and efficiency of two Sobol' index estimators," ESAIM: Probability and Statistics, vol. 18, pp. 342-364, 2014.

where the subscript n corresponds to the row of the sampled model parameters such that $f(B)_n$ is the model output evaluated using inputs taken from the n-th row of matrix B and $f(A_B^d)_n$ is the model output evaluated using inputs taken from the n-th row of matrix A_B^K. The MC estimator for the second term in the numerator and for the denominator differ for the two considered estimators given in Table below.

Estimator	$\mathbb{E}^2[Y] = \left(\int f d\mathbf{x}\right)^2$	$\mathbb{V}[Y] = \int f^2 d\mathbf{x} - \left(\int f d\mathbf{x}\right)^2$
Saltelli[7]	$\frac{1}{N}\sum f(A)_n \cdot f(B)_n$	$\frac{1}{N}\sum f(A)_n^2 - \left(\frac{1}{N}\sum f(A)_n\right)^2$
Janon et al.[8]	$\left(\frac{1}{N}\sum \frac{f(B)_n + f(A_B^d)_n}{2}\right)^2$	$\frac{1}{N}\sum \frac{f(B)_n^2 + f(A_B^d)_n^2}{2} - \left(\frac{1}{N}\sum \frac{f(B)_n^2 + f(A_B^d)_n^2}{2}\right)^2$

The general formula of the main-effect sensitivity index estimator is

$$\widehat{S}_d = \frac{\frac{1}{N}\sum_{n=1}^{N} f(B)_n \cdot f(A_B^d)_n - \mathbb{E}^2[Y]}{\mathbb{V}[Y]} \tag{1.15}$$

where and $\mathbb{E}^2[Y]$ and $\mathbb{V}[Y]$ are as prescribed in the table above.

The general formula of the total-effect sensitivity indices follows the definition of it as given in (1.6). The denominator $\mathbb{E}_{\sim d}[\mathbb{V}_d[Y|\mathbf{X}_{\sim d}]]$ is estimated using the estimators listed in the table below, while $\mathbb{V}[Y]$ is estimated by the Saltelli et al. estimator in table above.

Estimator	$\mathbb{E}_{\sim d}[\mathbb{V}_d[Y\|\mathbf{X}_{\sim d}]]$
Sobol-Homma[5]	$\frac{1}{N}\sum f^2(A)_n \cdot f(A)_n f(AB^d)_n$
Jansen[10]	$\frac{1}{2N}\sum \left(f(A)_n - f(AB^d)_n\right)^2$

The computational cost associated with the estimation of all the main-effect and total-effect indices is $N \times (D+2)$ code runs, where N is the number of MC samples and D is the number of parameters. Compare this to the cost of brute force Monte Carlo of $2 \times D \times N^2$ code runs to estimate all the main-effect and total-effect sensitivity indices.

As an additional comparison, the cost for Morris method to compute the statistics of elementary effect is $N_R \times (D+1)$ code runs, where :math‘N_R‘ is the number of OAT design replications. In either methods, the number of samples :math‘N‘ (in the case of the Sobol’-Saltelli method) and replications N_R (in the case of the Morris method) determines the precision of the estimates. A larger number of samples (and replications) increases the precision. Note, however, that in practice the typical number of Morris replications is between $10^1 - 10^{2}$[10], while the number of gls{mc} samples for the Sobol’ indices estimation amounts to $10^2 - 10^{44}$.

References

Developer’s Guide

Guidelines for Contributing

This page is still under preparation

Installation Instructions for Developer

This page is still under preparation

Modifying Code

This page is still under preparation

[10] F. Campolongo, A. Saltelli, and J. Cariboni, “From Screening to Quantitative Sensitivity Analysis. A Unified Approach,” Computer Physic Communications, vol. 182, no. 4, pp. 978-988, 2011.

About gsa-module

Contributors

gsa_module is written and currently maintained by Damar Wicaksono <damar.wicaksono@gmail.com>

Other co-maintainers:

- Gregory Perret <greg.perret@psi.ch>

License

Funding

gsa_module is developed to support sensitivity analysis of model output, particularly from (though not stricted to) thermal-hydraulics system code, under a doctoral research project funded by the Swiss Federal Institute of Technology, in Lausanne; carried out at the Laboratory for Reactor Physics and Systems Behaviour, Paul Scherrer Institut, in Villigen.

Gallery of Applications to Test Functions

Ishigami Function

Ishigami function is a 3-dimensional function introduced by Ishigami and Homma[1],

$$f(\mathbf{x}) = \sin x_1 + a \sin^2 x_2 + b x_3^4 \sin x_1$$

$$x_d \sim U[-\pi, \pi]; d = 1, 2, 3$$

the parameters a and b can be adjusted but have default values of 7 and 0.1, respectively.

[1] T. Homma and A. Saltelli, "Importance measures in global sensitivity analysis of nonlinear models," Reliability Engineering and System Safety, vol. 52, pp. 1-17, 1996.

Analytical Solution

The analytical formulas for the variance terms of the Ishigami function for $\mathbf{X}_d \sim \mathcal{U}[-\pi, \pi]$; $d = 1, 2, 3$ and the given parameter a and b are the following

Marginal Variance

$$\mathbb{V}[Y] = \frac{a^2}{8} + \frac{b\pi^4}{5} + \frac{b^2\pi^8}{18} + \frac{1}{2}$$

Top Marginal Variance

$$\begin{aligned} V_1 &= \mathbb{V}_1[\mathbb{E}_{2,3}[Y|X_1]] \\ &= \frac{1}{2}\left(1 + \frac{b\pi^4}{5}\right)^2 \\ V_2 &= \mathbb{V}_2[\mathbb{E}_{1,3}[Y|X_2]] \\ &= \frac{a^2}{8} \\ V_3 &= \mathbb{V}_3[\mathbb{E}_{1,2}[Y|X_3]] \\ &= 0 \end{aligned}$$

Bottom Marginal Variance

$$\begin{aligned} VT_1 &= \mathbb{E}_{2,3}[\mathbb{V}_1[Y|\mathbf{X}_2, \mathbf{X}_3]] = \frac{1}{2}\left(1 + \frac{b\pi^4}{5}\right)^2 + \frac{8b^2\pi^8}{225} \\ VT_2 &= \mathbb{E}_{1,3}[\mathbb{V}_2[Y|\mathbf{X}_1, \mathbf{X}_3]] = \frac{a^2}{8} \\ VT_3 &= \mathbb{E}_{1,2}[\mathbb{V}_3[Y|\mathbf{X}_1, \mathbf{X}_2]] = \frac{8b^2\pi^8}{225} \end{aligned}$$

The analytical main- and total-effect sensitivity indices can be computed using their respective definition in relation to the variance terms given above.

Morris Screening Results

The function was used to test the implementation of the Morris screening and most precisely that of the two designs of experiment: the trajectory and radial designs (see *Morris Screening Method*).

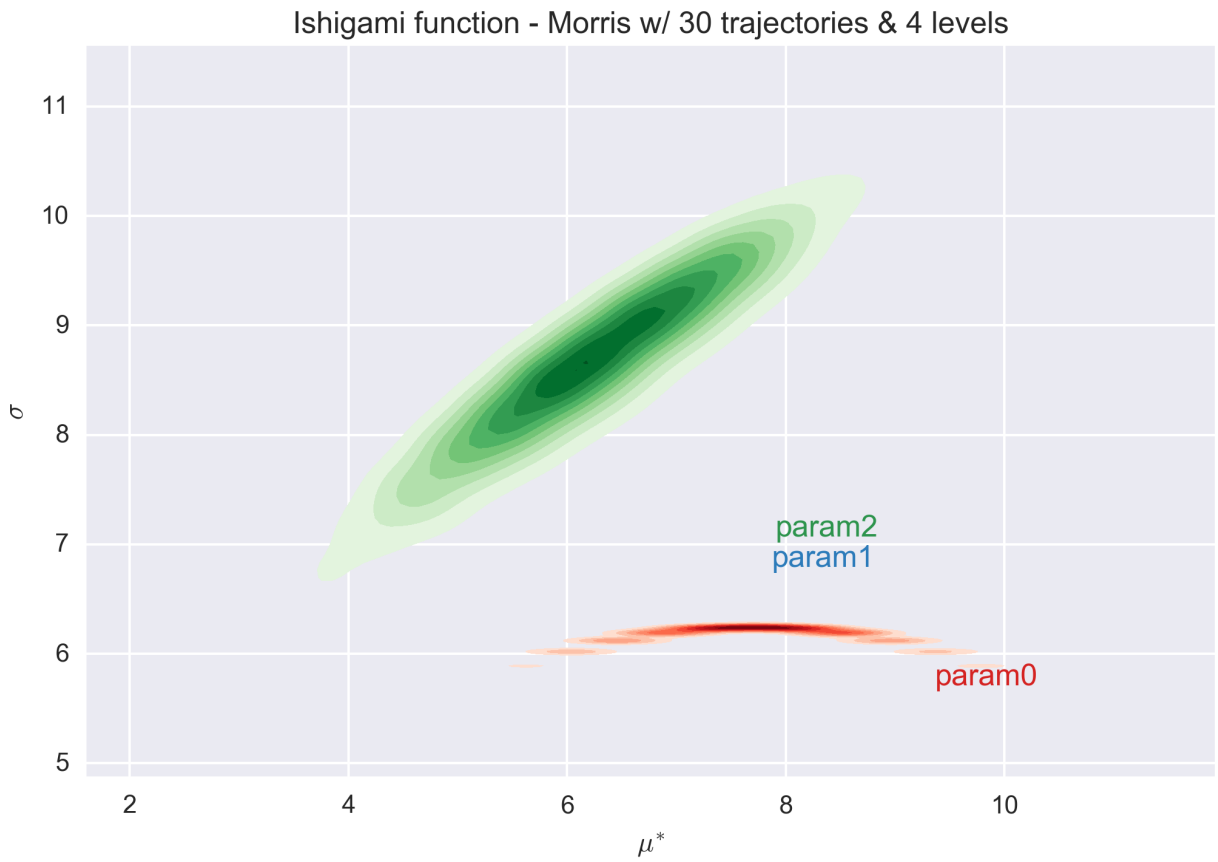
Trajectory sampling design

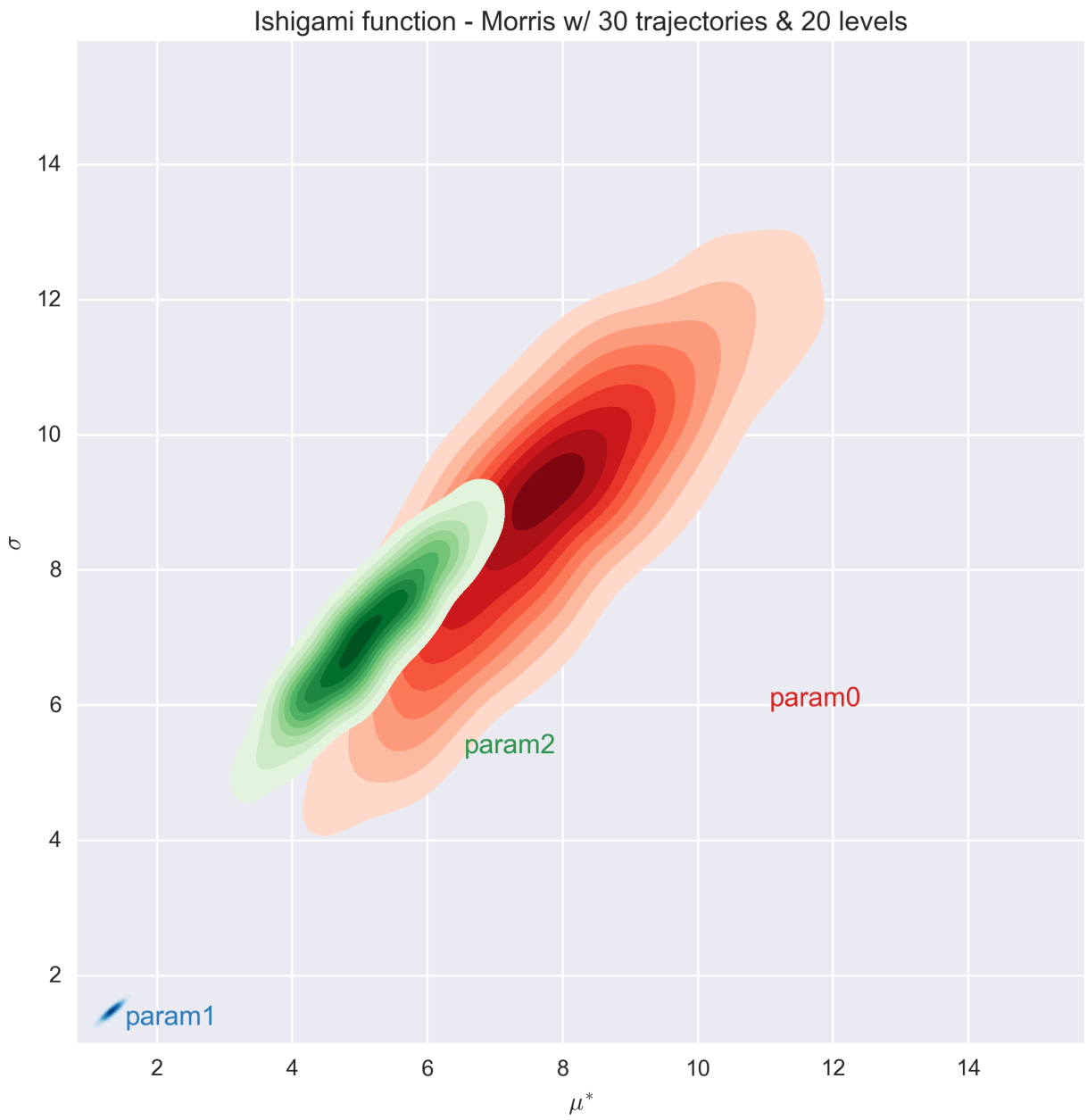
The trajectory effect is the original design proposed by Morris. The design matrix was generated with:

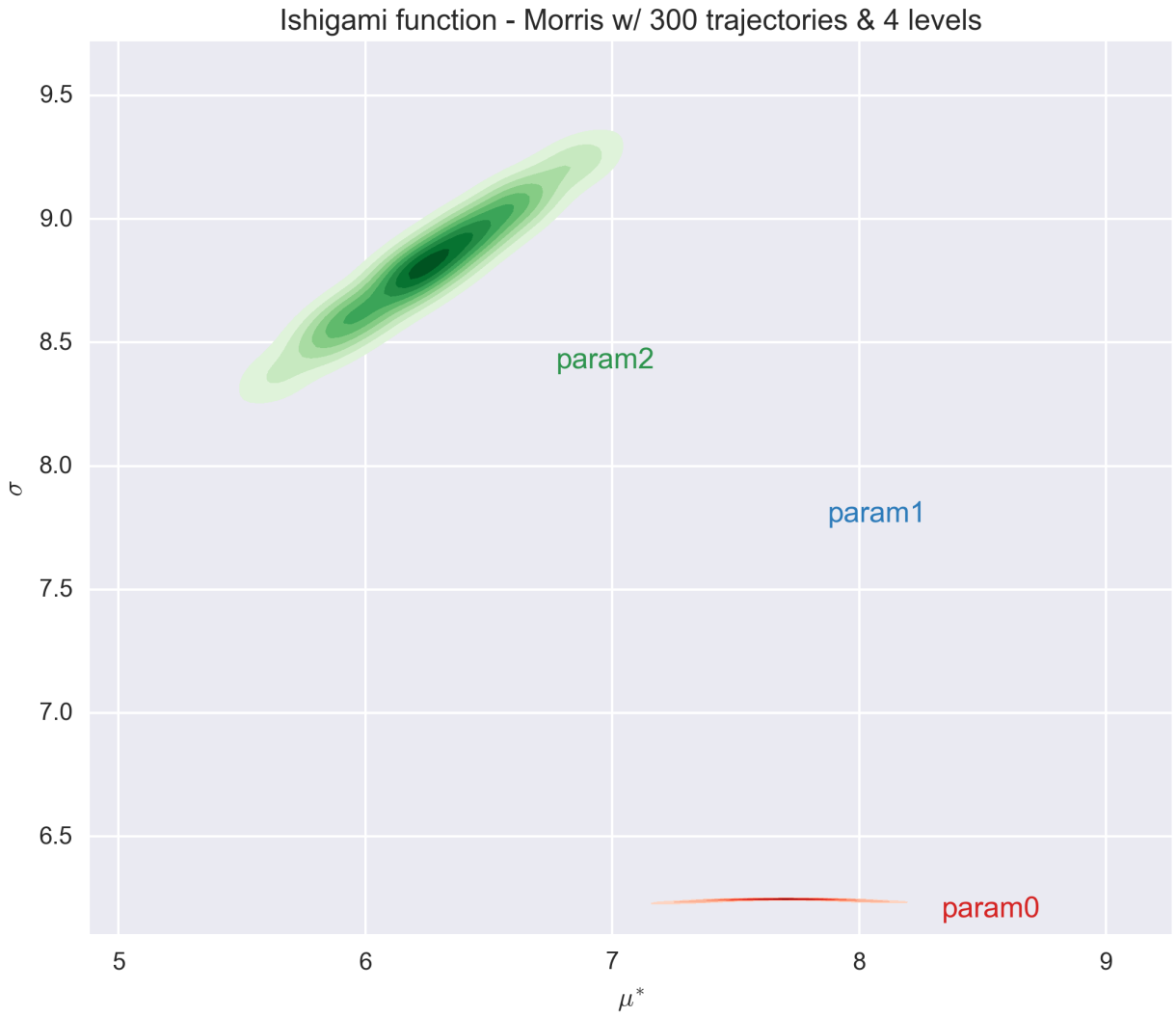
- number of trajectories (*r*) equal to 10, 100 and 1000 times the number of parameter (*k=3*),
- and levels (*p*) equal to 4, 8, 12 and 20.

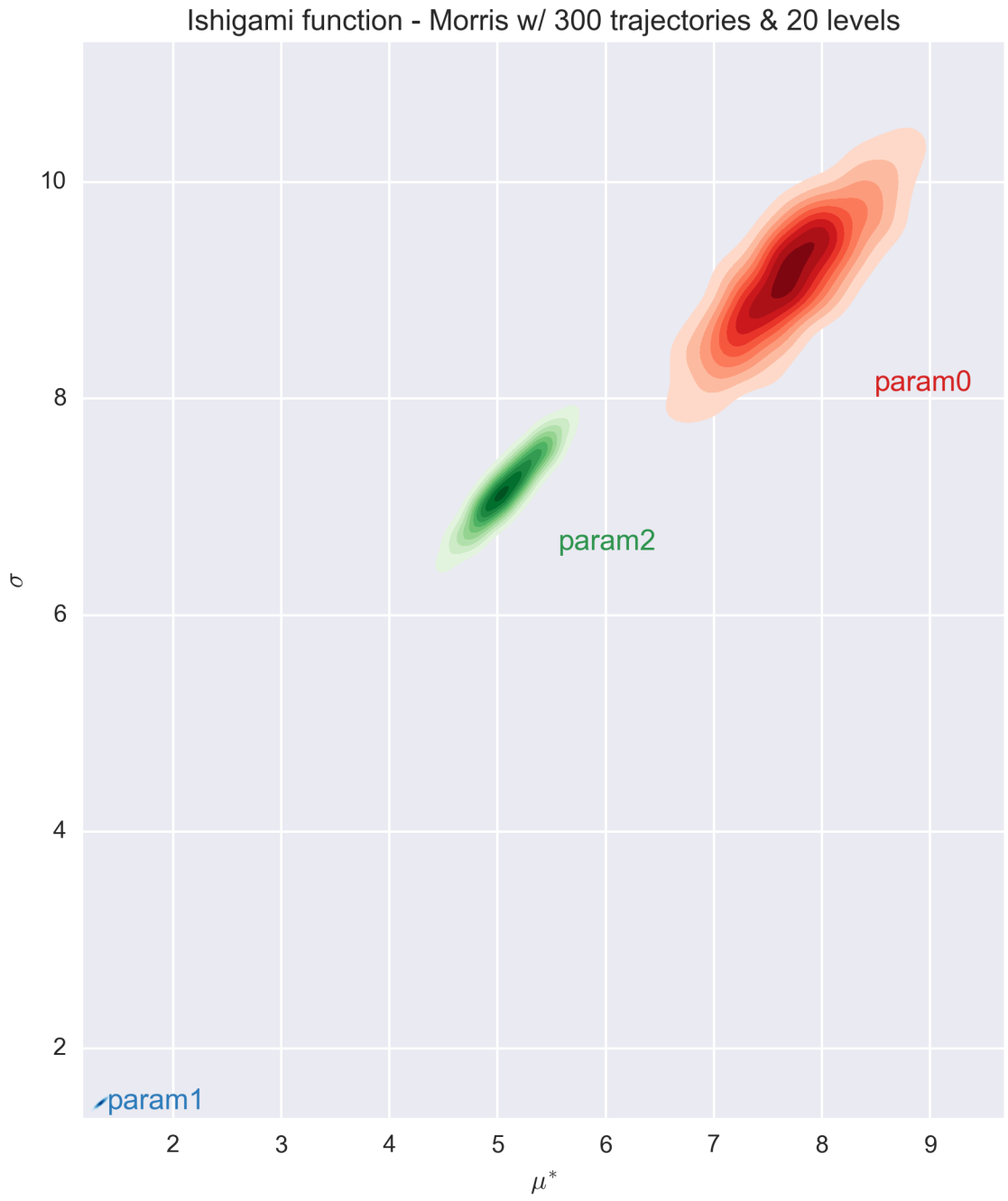
Each generated design was used to evaluate the Morris modified function and the associated elementary effects were calculated.

The following figures show the σ vs. μ^* plot for the four parameters of the Ishigami function for different sets of *r* and *p* values. Each set of (*r*, *p*) value was repeated 1000 times and a histogram of the results is presented for each parameter in the figures.


Ishigami function - Morris w/ 30 trajectories & 4 levels
11
10
9
8
7
6
5
σ
2
4
6
8
10
μ^*
param2
param1
param0


Ishigami function - Morris w/ 30 trajectories & 20 levels
14
12
10
8
6
4
2
σ
param0
param2
param1
2
4
6
8
10
12
14
μ*


Ishigami function - Morris w/ 300 trajectories & 4 levels
9.5
9.0
8.5
8.0
7.5
7.0
6.5
σ
param2
param1
param0
5
6
7
8
9
μ*



Countrary to the *Modified Morris Function*, the value of the elementary design with a level *p=4* is quite different that the values obtained with the other level values. This remains true for a very large number of trajectories (i.e. 3000). The predictions with a level of 8 or higher are consistent.

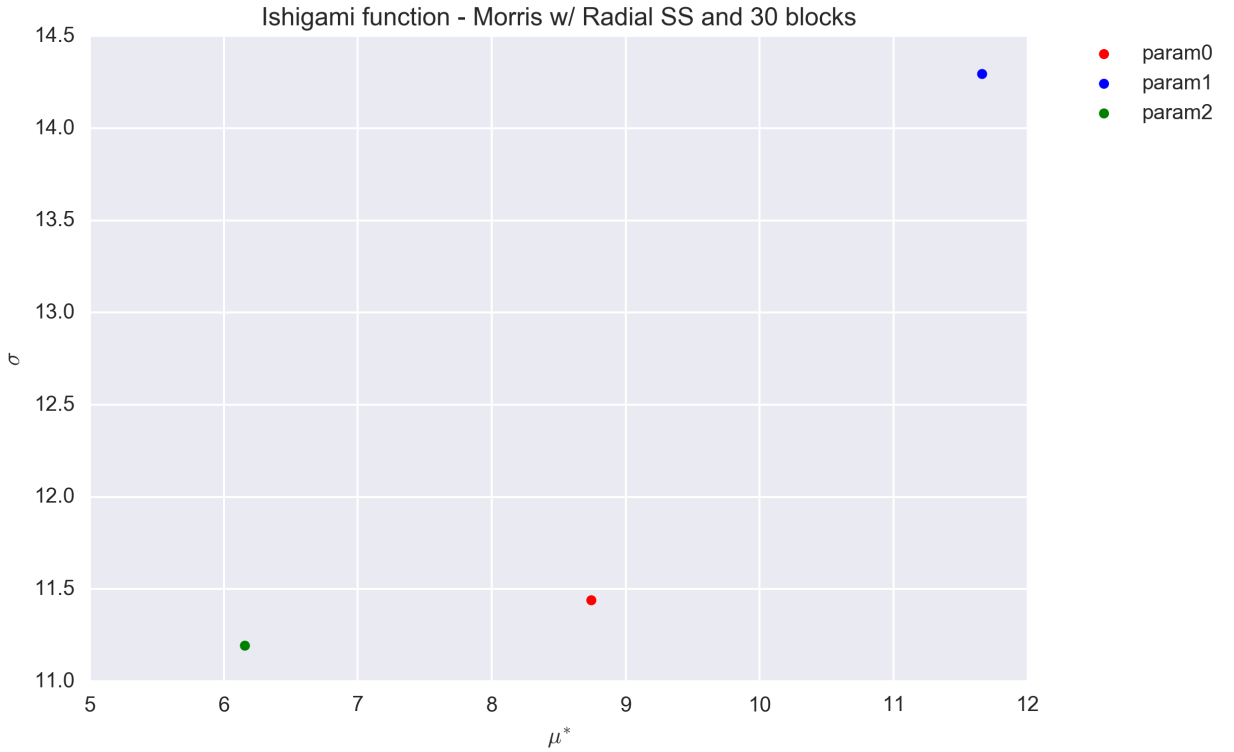
We also observe that a number of trajectories equals to 10 times the number of parameter (i.e. 30) is not sufficient to entirely classified the parameters (as the histograms of parameters 0 and 2 overlap). A minimum number of trajectories of about 100 times the number of parameter is necessary for the Ishigami function to separate the parameters.

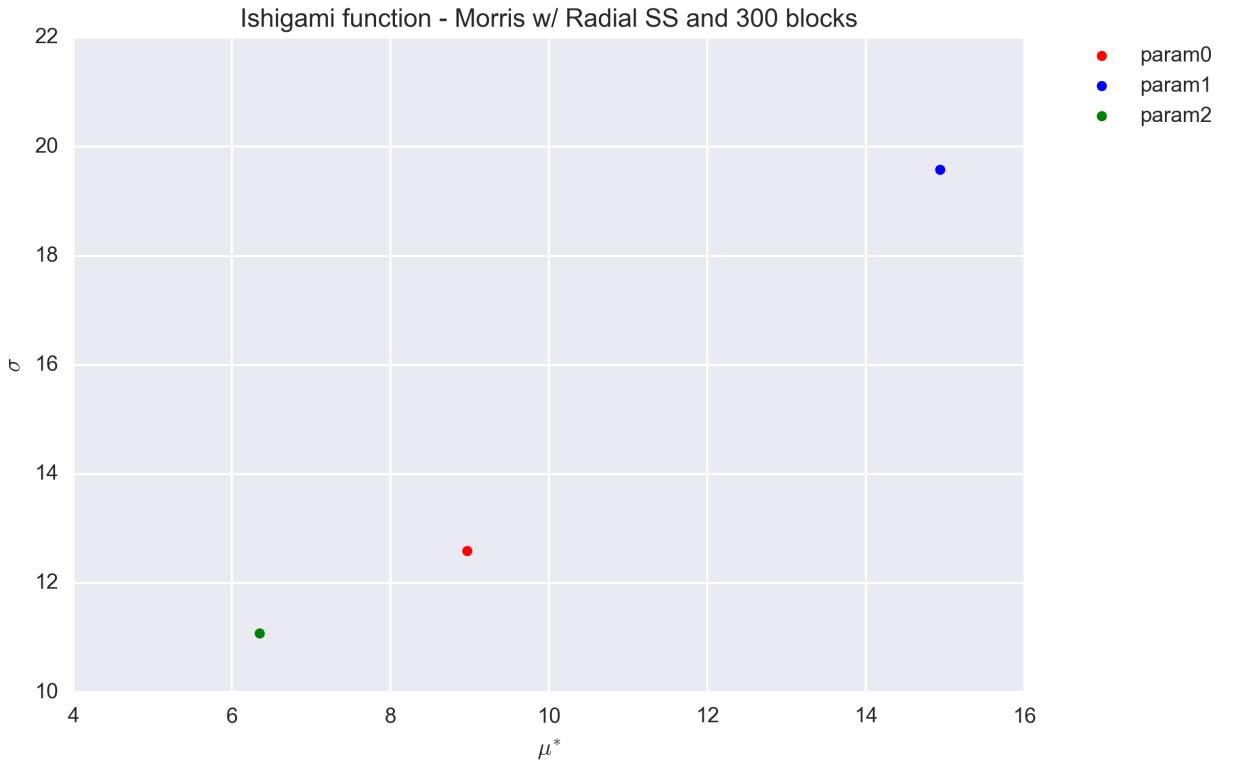
Radial sampling design

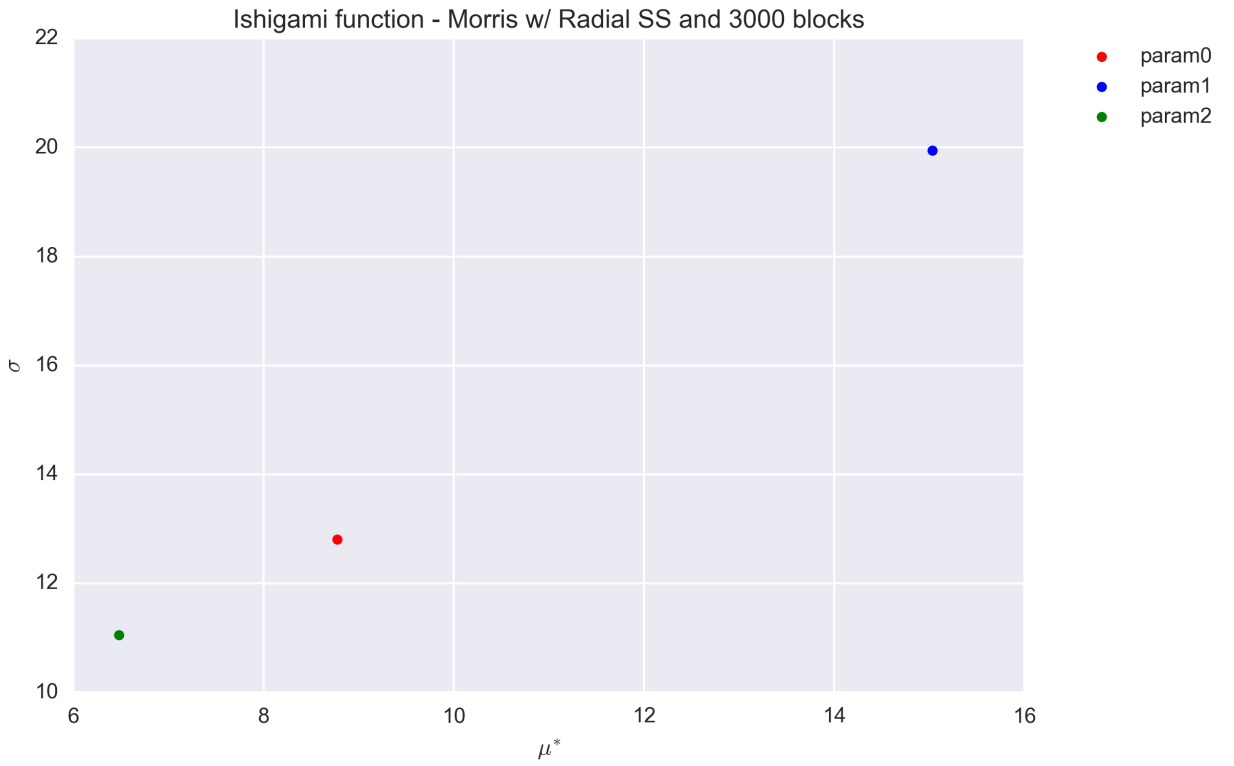
The radial sampling design has been proposed by Campagnolo et al. and is described in more details at *Morris Screening Method*. Only a number of trajectories, here called blocks to differentiate from the previous design, is required. For testing purposes we investigated, as previously, numbers of blocks (r) equal to 10, 100 and 1000 times the number of parameter (k=*3*).

Each generated design was used to evaluate the Ishigami function and the associated elementary effects were calculated.

The following figures show the σ vs. μ^* plot for the three parameters of the Ishigami function and for the three sets of r values. Countrary to the trajectory design, the radial design uses the Sobol generator, which is deterministic. As such no repetitions were performed to investigate the dispersion of the (σ, μ^*) values.







The values are found to be quite stable, even for a block value of *r=30*. They differ, however, significantly from that obtained with the trajectory design (Section *Trajectory sampling design*).

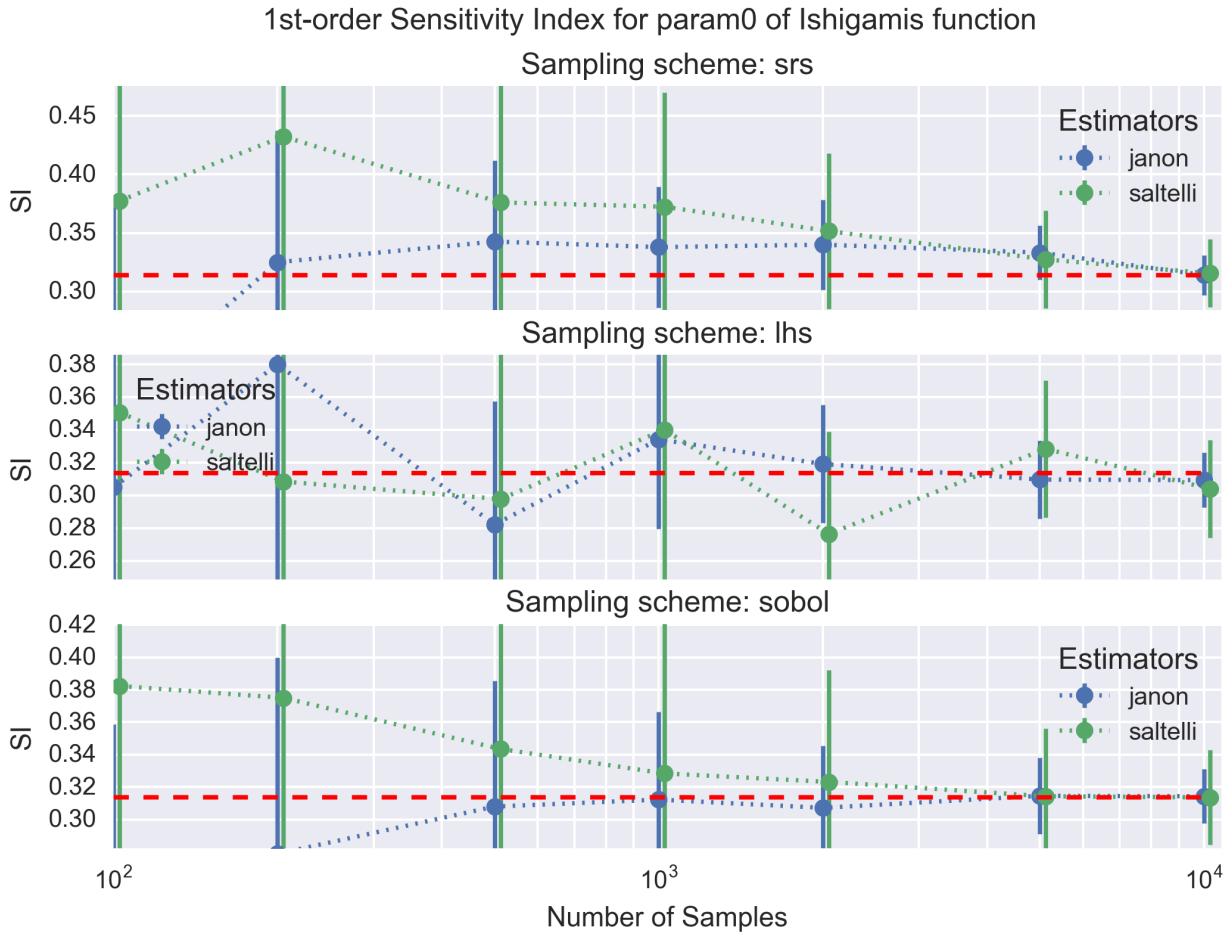
The exact value for the elementary effects were not found in litterature. However, the sensitivity indices are $S_1 =$

0.3138, $S_2 = 0.4424$ and $S_3 = 0^1$. Because μ^* and S quantify the same information, we expect them to be ordered in the same way. Therefore the results obtained with the radial sampling design appear preferable.

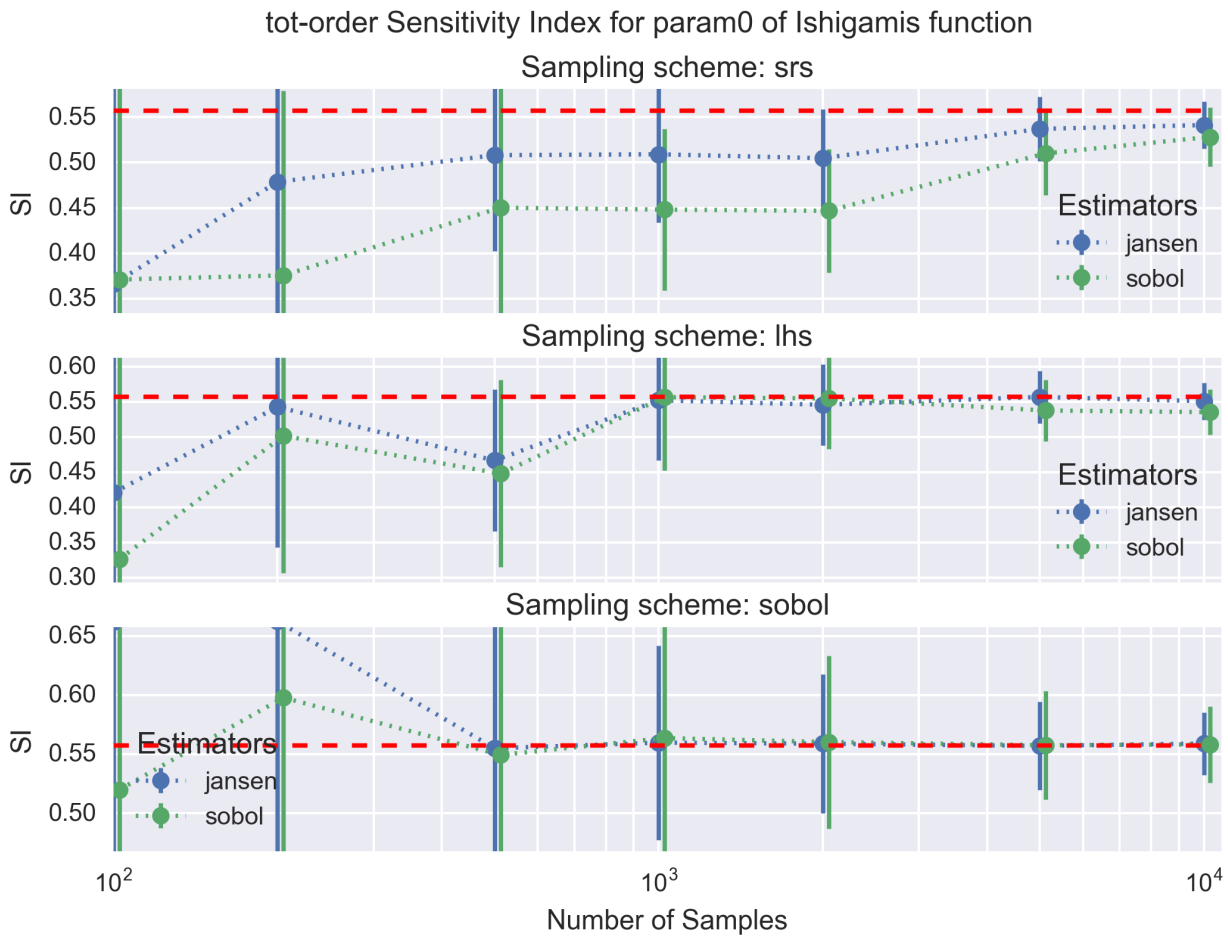
Sobol Sensitivity Indices Results

The function was used to test the implementation of the Sobol sensitivity indices. The main-effect (first order) and total-effect (total order) sensitivity indices are both computed. Both the sampling scheme type and the estimator for the sensitivity indices were tested. The tested sampling schemes are simple random sampling (*srs*), latin-hypercube sampling (*lhs*) and the sobol sampling (*sobol*). The tested estimators are *janon* and *saltelli* for the main-effect *SI* and *jansen* and *sobol* for the total-effect *SI* (see *Sobol' Sensitivity Indices*).

The following figure shows the convergence of the main-effect *SI* (first order) with the number of samples for the first parameter (*param0*) of the Ishigami's function. Each panel shows the *janon* and *saltelli* estimators, with their $1 - \sigma$ uncertainties, for a given sampling scheme. The dotted red line is the analytical solution (i.e. the target value).



A similar figure is shown below for the total-effect *SI* (tot-order) for the *jansen* and *sobol* estimators.



All estimators for the main- and total-effect *SI* converge to the analytical solution with a sufficient number of samples (i.e. 10^4 in the worst case). As expected the *sobol* and *lhs* sampling schemes for the design matrix are clearly superior to the simple random scheme (*srs*) as the calculated main- and total-effect *SI* converge faster and with a lower uncertainies; the *sobol* sampling scheme appears to be only slightly better than *lhs*. Finally, comparing the estimators the *janon* and *jansen* estimators show slightly better properties than the *saltelli* and *sobol* estimators. These conclusions remain the same for all input parameter of the Ishigami's function.

From a practical point of view, we advise to use the *sobol* or *lhs* sampling scheme with at least *1000* points. The estimator does not play a significant role.

References

Modified Morris Function

A modified version of test function appeared in Morris' original article[1] is used as a test function in this module. Instead of 20-dimensional function, the modified version is only 4-dimensional and truncated as follows,

$$f(\underline{x}) = \sum_{i=1}^{4} \beta_i x_i + \sum_{i \leq j}^{4} \beta_{i,j} x_i x_j$$

$$\beta_i = [0.05, 0.59, 10.0, 0.21]$$

[1] Max D. Morris, "Factorial Sampling Plan for Preliminary Computational Experiments," Technometrics, vol. 33, no. 2, pp. 161 - 174, 1991.

$$\beta_{i,j} = \begin{bmatrix} 0 & 80 & 60 & 40 \\ 0 & 30 & 0.73 & 0.18 \\ 0 & 0 & 0.64 & 0.93 \\ 0 & 0 & 0.0 & 0.06 \end{bmatrix}$$

The function accepts inputs $\underline{x}$ the component of which are :math: *0 < x_i leq 1*.

Morris Screening Results

The function was used to test the implementation of the Morris screening and most precisely that of the two designs of experiment: the trajectory and radial designs *Morris Screening Method*.

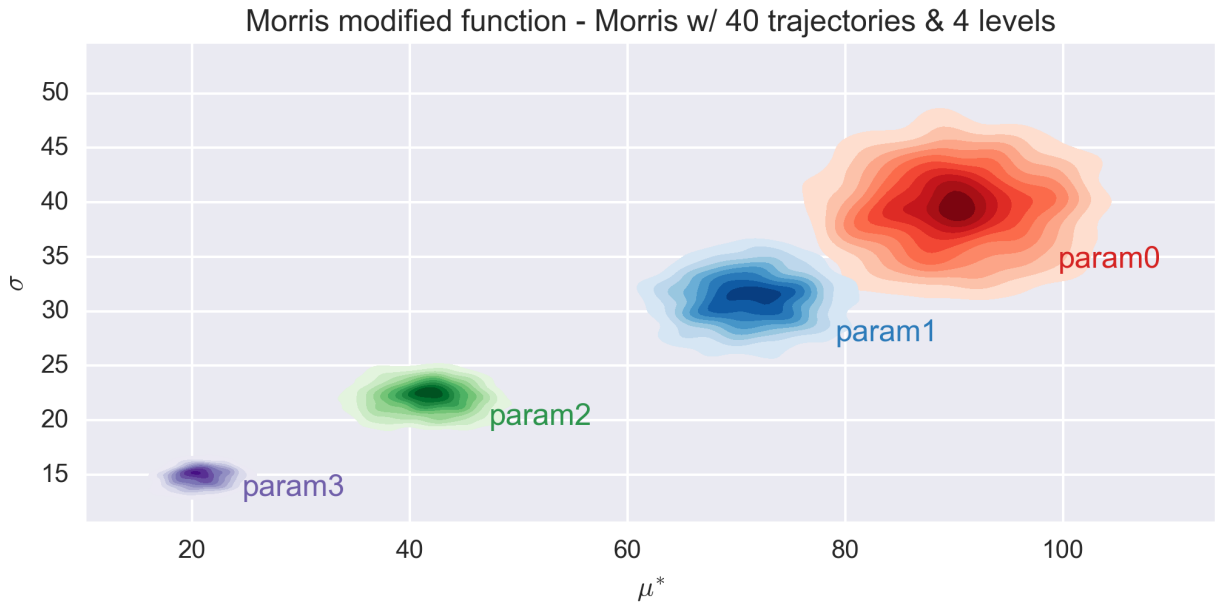
Trajectory sampling design

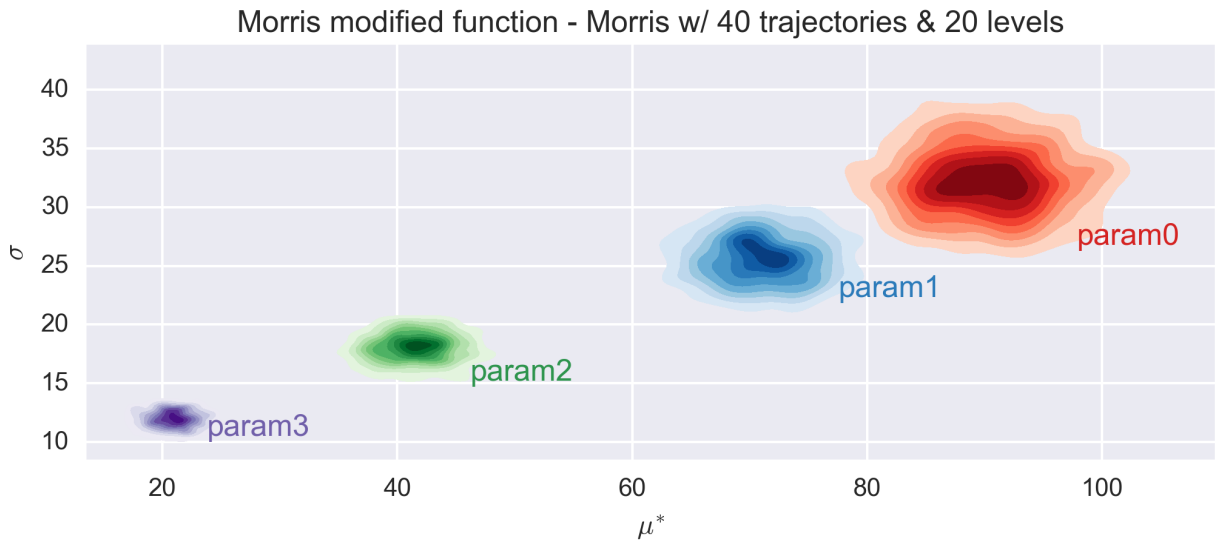
The trajectory effect is the original design proposed by Morris. The design matrix was generated with:

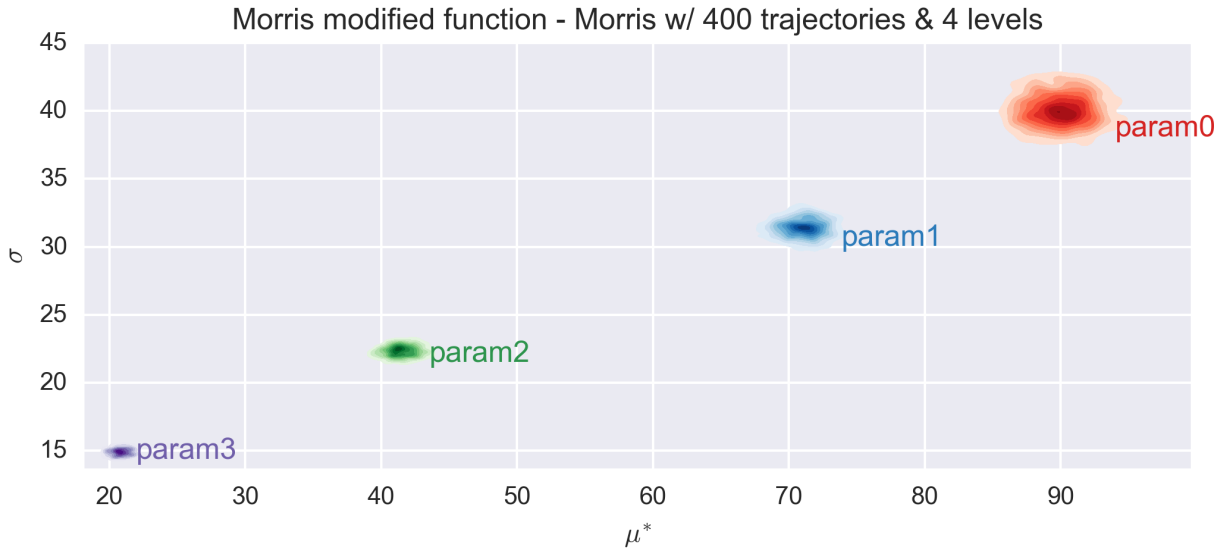
- number of trajectories (*r*) equal to 10, 100 and 1000 times the number of parameter (*k=4*),
- and levels (*p*) equal to 4, 8, 12 and 20.

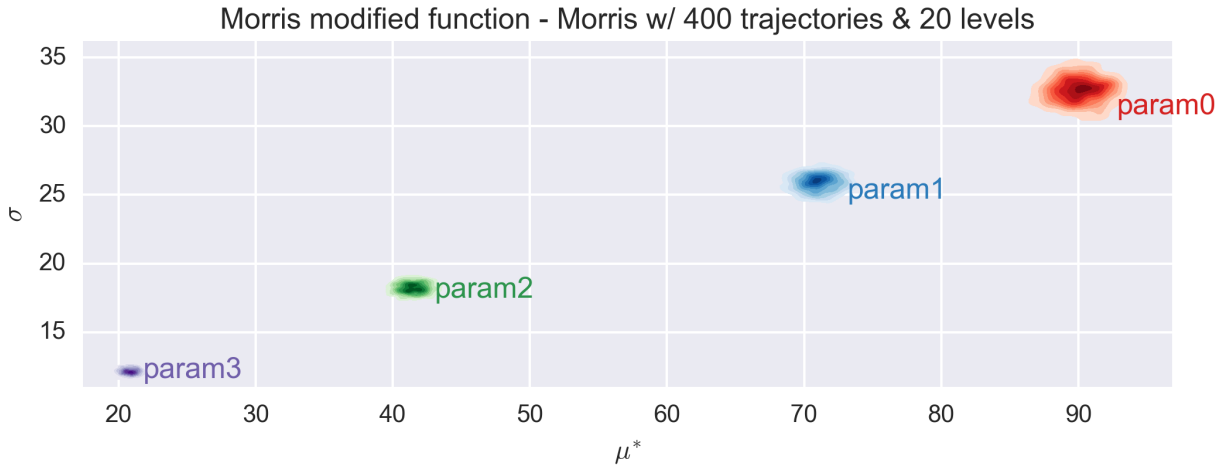
Each generated design was used to evaluate the Morris modified function and the associated elementary effects were calculated.

The following figures show the σ vs. μ^* plot for the four parameters of the Morris modified function for different sets of *r* and *p* values. Each set of (*r*, *p*) value was repeated 1000 times and a histogram of the results is presented for each parameter in the figures.









The elementary effects of the function are similar for all combinations of r and p values and consistent with the

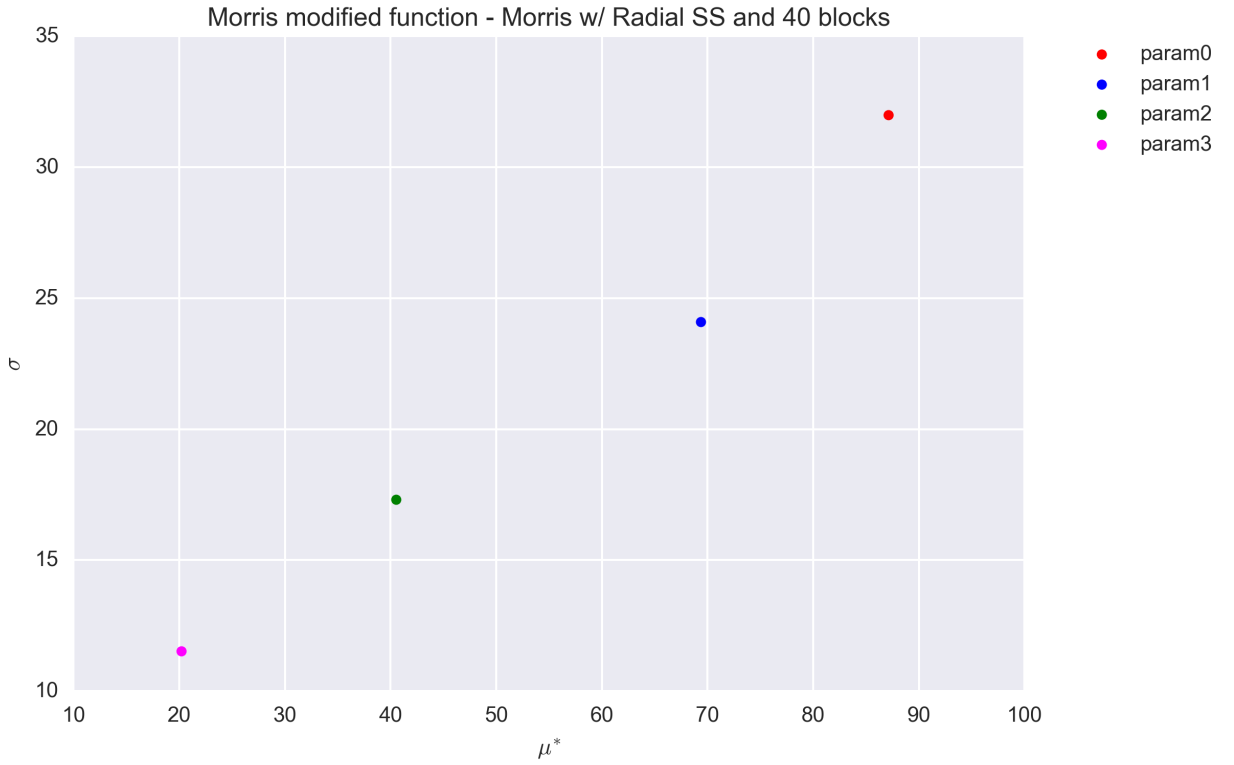
expected values. As expected, the histogram dispersion is reduced when the number of trajectories increases. The classification of the parameters is stable already for an number of trajectories equals to 10 times the nubmer of dimension.

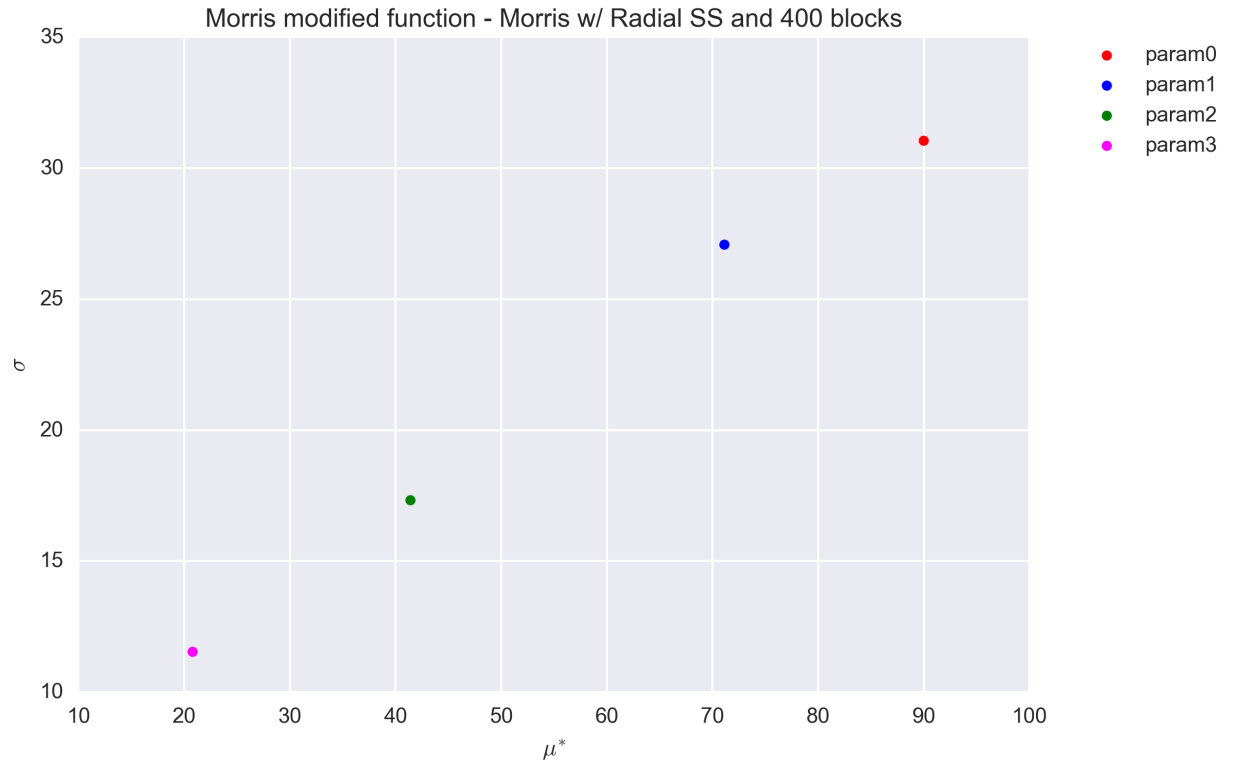
Radial sampling design

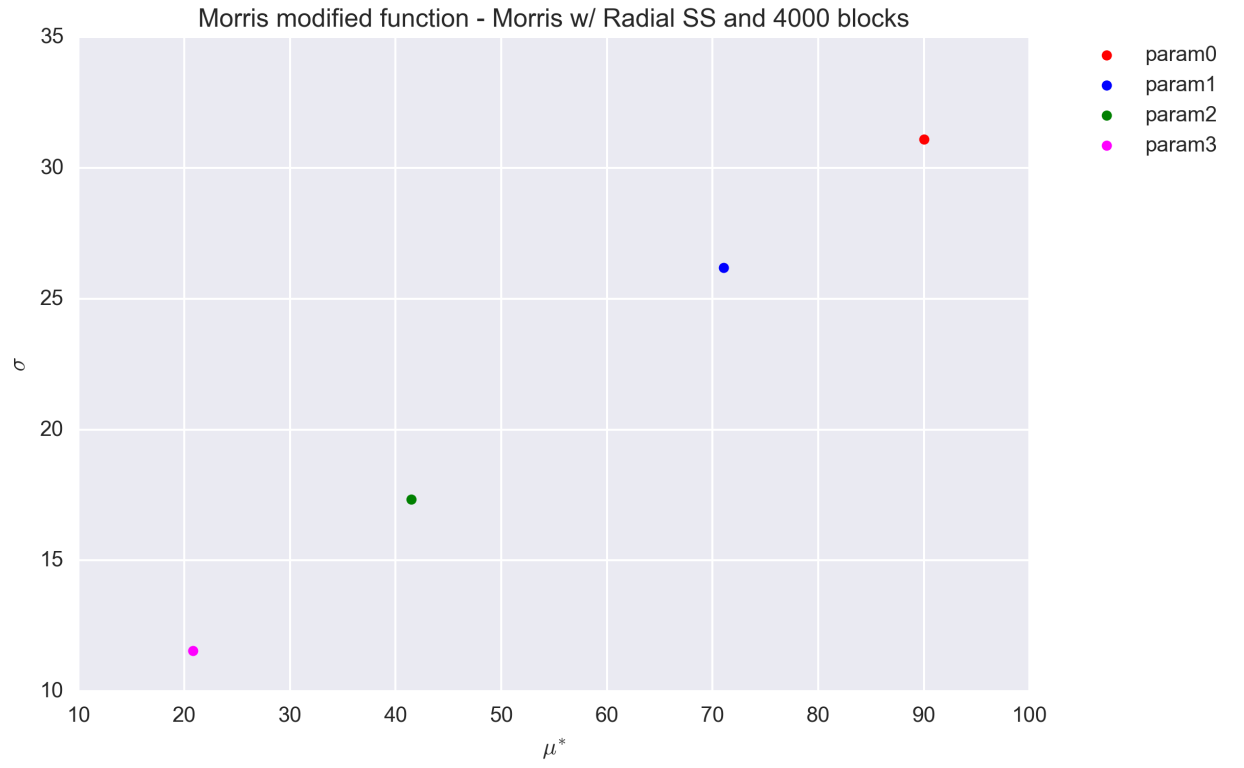
The radial sampling design has been proposed by Campagnolo et al. and is described in more details at *Morris Screening Method*. Only a number of trajectories, here called blocks to differentiate from the previous design, is required. For testing purposes we investigated, as previously, numbers of blocks (r) equal to 10, 100 and 1000 times the number of parameter (k=*4*).

Each generated design was used to evaluate the Morris modified function and the associated elementary effects were calculated.

The following figures show the σ vs. μ^* plot for the four parameters of the Morris modified function and for the three sets of r values. Countrary to the trajectory design, the radial design uses the Sobol generator, which is deterministic. As such no repetitions were performed to investigate the dispersion of the (σ, μ^*) values.







The values are found to be quite stable, even for a block value of *r=40* and are in agreement with that obtained with the trajectory design (Section *Trajectory sampling design*). A better quanitification of the convergence to the reference values of σ, μ^* is shown in the table below. The only possible exception is for σ of parameter 1 (x_2).

Parameter	Statistics	Ref	r=40	r=400	r=4000
Param0	μ^*	90.05	87.11	89.97	90.02
Param0	σ	31.08	31.99	31.05	31.09
Param1	μ^*	71.05	69.36	71.11	71.06
Param1	σ	28.86	24.10	27.09	26.18
Param2	μ^*	41.47	40.50	41.38	41.47
Param2	σ	17.75	17.31	17.33	17.33
Param3	μ^*	20.83	20.19	20.76	20.82
Param3	σ	11.54	11.53	11.55	11.55

References

Sobol-G^* Function

Sobol-G^* is a modified version of Sobol-G function proposed in[1] initially to avoid excluding singular value at $\mathbf{x} = \{0.5\}$. The function reads,

$$G^*(\mathbf{x}, \mathbf{a}, \boldsymbol{\alpha}, \boldsymbol{\delta}) = \prod_{d=1}^{D} g_i^*$$
$$g_d^* = \frac{(1+\alpha_d) \cdot |2(x_d + \delta_d - I[x_d + \delta_d]) - 1|_d^\alpha + a_d}{1 + a_d}$$

where $\mathbf{x} \in [0,1]^D$; $a_d \in \mathbb{R}^+$ determines the first-order importance of parameter-d; $\delta \in [0,1]^D$ is the shift parameter; $\boldsymbol{\alpha} \in \mathbb{R}^{D+}$ is the curvature parameter; and $I[\circ]$ is the integer part of the number.

The parameters $\mathbf{a}$ and α can be adjusted. The particular test function used in this module is G_2^* (see pp. 265 in[1]) where

$$\alpha_d = 1;\ d = 1, 2, \cdots, 10$$
$$a_d = (0, 0.1, 0.2, 0.3, 0.4, 0.8, 1, 2, 3, 4)$$
$$\delta_d \sim \mathcal{U}[0,1];\ d = 1, 2, \cdots, 10$$

Analytical Solution

The analytical formulas for the variance terms of the Sobol-G^* function for $\mathbf{X}_d \sim \mathcal{U}[0,1];\ d = 1, 2, \cdots, 10$ are the following

Marginal Variance

$$\mathbb{V}[G_2^*] = \left[\prod_{d=1}^{D}(1 + V_d)\right] - 1$$

where V_d is the top marginal variance given below,

Top Marginal Variance

$$V_d = \mathbb{V}_d[\mathbb{E}_{\sim d}[G_2^*|X_d]] = \frac{\alpha_d^2}{(1+2\alpha_d)(1+a_d)^2}$$

Bottom Marginal Variance

$$VT_d = \mathbb{E}_{\sim d}[\mathbb{V}_d[G_2^*|\mathbf{X}_{\sim d}]] = V_d \prod_{e \neq d}(1 + V_e)$$

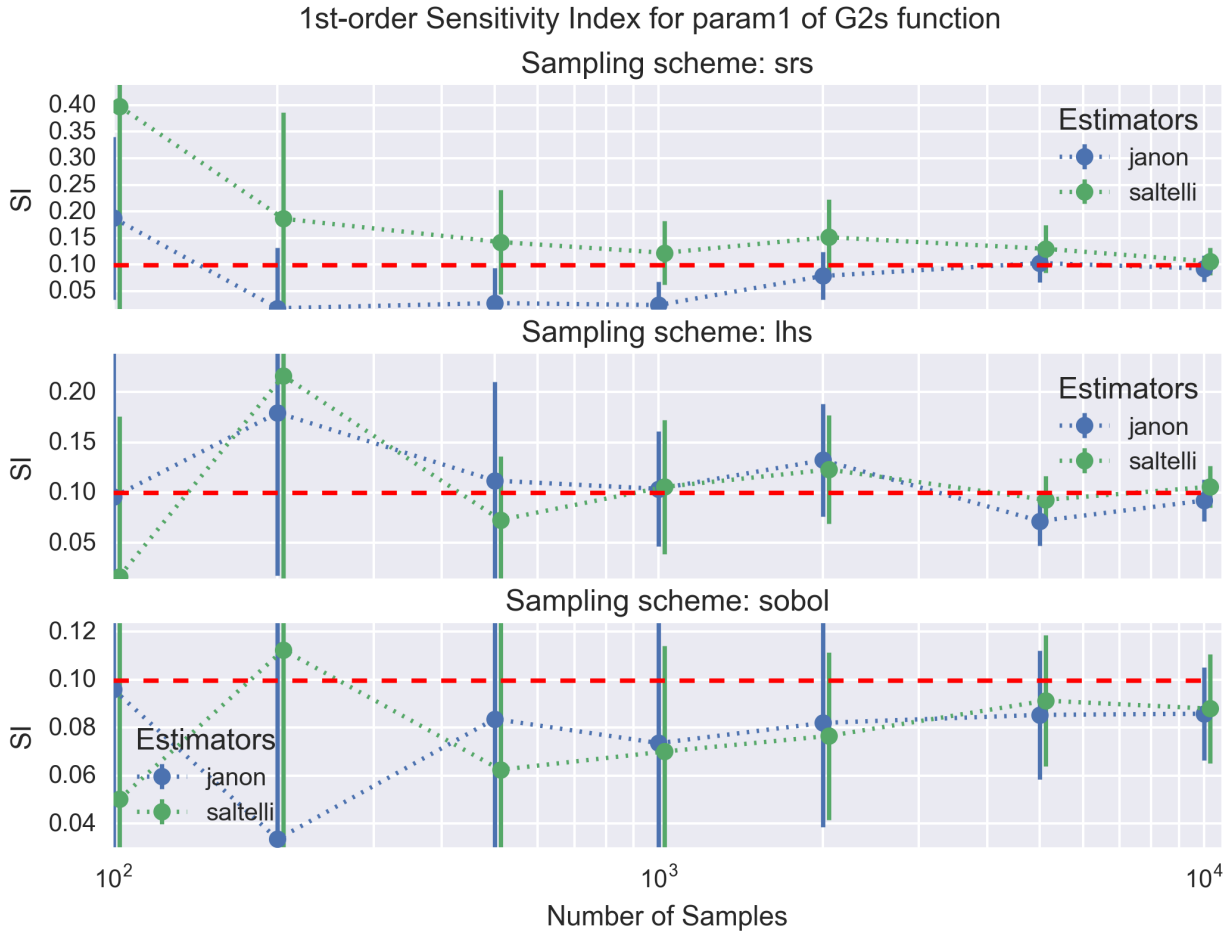
The analytical main- and total-effect sensitivity indices can be computed using their respective definition in relation to the variance terms given above.

[1] A. Saltelli et al., "Variance based sensitivity analysis of model output. Design and estimator for the total sensitivity index," Computer Physics Communication, vol. 181, no. 2, pp. 259 - 270, 2010.

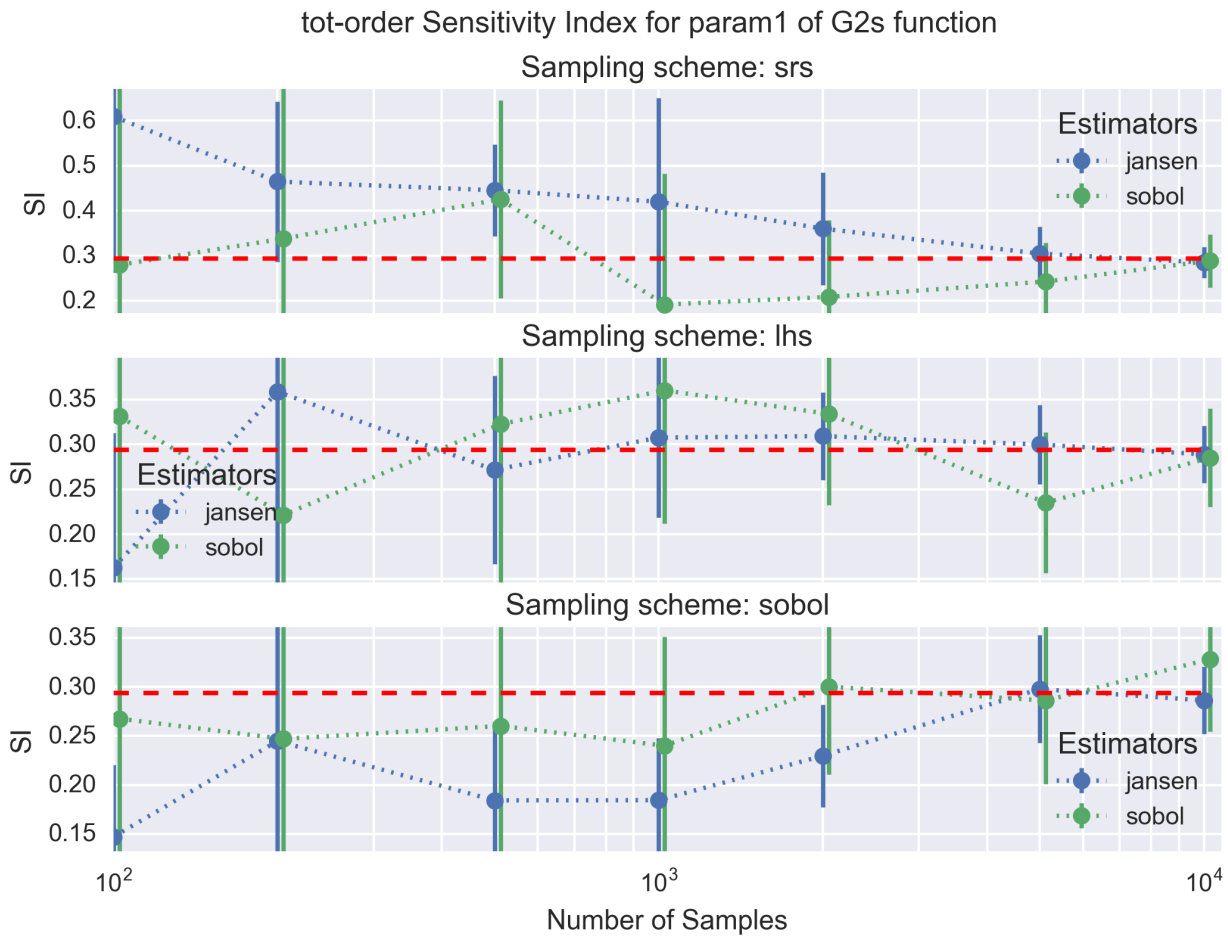
Sobol Sensitivity Indices Results

The function was used to test the implementation of the Sobol sensitivity indices. The main-effect (first order) and total-effect (total order) sensitivity indices are both computed. Both the sampling scheme type and the estimator for the sensitivity indices were tested. The tested sampling schemes are simple random sampling (*srs*), latin-hypercube sampling (*lhs*) and the sobol sampling (*sobol*). The tested estimators are *janon* and *saltelli* for the main-effect *SI* and *jansen* and *sobol* for the total-effect *SI* (see *Sobol' Sensitivity Indices*).

The following figure shows the convergence of the main-effect *SI* (first order) with the number of samples for the second parameter (*param1*) of the G_2 function. Each panel shows the *janon* and *saltelli* estimators, with their $1 - \sigma$ uncertainties, for a given sampling scheme. The dotted red line is the analytical solution (i.e. the target value).



A similar figure is shown below for the total-effect *SI* (tot-order) for the *jansen* and *sobol* estimators.



All estimators for the main- and total-effect *SI* converge to the analytical solution with a sufficient number of samples (i.e. 10^4 in the worst case). As expected the *sobol* and *lhs* sampling schemes for the design matrix are clearly superior to the simple random scheme (*srs*) as the calculated main- and total-effect *SI* converge faster and with a lower uncertainies; the *sobol* sampling scheme appears to be slightly better than *lhs*. Finally, comparing the estimators the *janon* and *jansen* estimators show slightly better properties than the *saltelli* and *sobol* estimators. The better properties of the estimators and sampling schemes illustrated for *param1* can vary slightly from parameter to parameter. The convergence of the calculation, however, remains. This confirms the good implementation of the *SI* calculation routines.

References

`gsa-module` Modules reference documentation

`samples` Package

`gsa_module.samples.cmdln_args`

`gsa_module.samples._hammersley`

`gsa_module.samples.srs`

gsa_module.samples.lhs

gsa_module.samples.lhs_opt

gsa_module.samples.sobol

morris Package

sobol Package

test_functions Package

CHAPTER 2

Indices and tables

- genindex
- modindex
- search