# UQLibrary: A Robust Implementation of Sensitivity of Identifiability Analysis in Python.

Harley Hanes

ARTICLE INFO

*Keywords*:
Sensitivity Analysis
Identifiability Analysis
Morris Screening
Sobol Analysis

ABSTRACT

We discuss `UQLibrary`, a Python implementation of sensitivity and identifiability analysis. `UQLibrary` aims to alleviate the need for a package which employs a diverse set of sensitivity and identifiability methods while allowing for parellelization to increase efficiency on computationally expensive problems. We present background on each sensitivity and identifiability analysis method used. We discuss the requirements and structure of `UQLibrary`.We also present results from heat diffusion and Ishigami test cases, showing the importance of using non-uniform parameter distributions in global sensitivity analysis.

## 1. Introduction

Sensitivity analysis is a set of methods for quantifying how variability in model parameters, POIs, affects the variability of model outputs, QOIs. Sensitivity analysis can classify which inputs to any blackbox model have the greatest effect on a desired set of outputs and which can be ignored. It has a wide range of applications including constructing reduced order models, identifying how to best focus data collection, and improving design of model optimization. Sensitivity analysis generally falls into two categories, local and global. Local sensitivity analysis quantifies the change in a model output due to perturbations of parameters about a base value, $\boldsymbol{\theta}^*$ [3]. Local sensitivity results can also be utilized in identifiability analysis to classify reduced sets of parameters and formulate reduced order models [7]. In contrast, global sensitivity analysis quantifies how variability in model responses is apportioned to variability in model inputs over a distribution [23]. Global sensitivity analysis is more robust since it measures sensitivity across a broader range of parameter values, but requires assuming a distribution from which parameter values are sampled. Whether a method is one-at-a-time (OAT) or not is another important classification. As the term suggests, OAT methods change only one parameter value at a time when computing sensitivity metrics, which may limit knowledge on sensitivity as parameters often have some correlation. Thus, non-OAT methods can be the most robust for sensitivity analysis but often more computationally expensive and prone to error from mis-estimation of parameter distributions [10].

We developed a new sensitivity and identifiability analysis package for Python called `UQLibrary` [11]. Popular packages currently exist in Python for sensitivity analysis, such as SALib and DAKOTA, but there are currently gaps in usefulness between these packages that require an alternate package [1, 14]. DAKOTA is one of the most robust uncertainty quantification packages currently available and as such includes some sensitivity analysis methods. However, DAKOTA currently only has implemented Sobol analysis. While a very powerful method, Sobol analysis has stringent assumptions, such as parameters' distributions must be independent, and requires a large number of function evaluations making it intractable in many cases. SALib has a wide range of sensitivity analysis methods implemented, but lacks parallelization of model evaluations. Sensitivity analysis often requires making thousands of model evaluations which can make analysis of models that do not significantly benefit from parallelization on their own computationally intractable. Both DAKOTA and SALib also lack any implementation of identifiability analysis and require all parameters to be sampled from uniform distributions. This is often an unrealistic assumption and we detail in Section 4.3 the errors that can occur when limiting analysis to uniform distributions.

`UQLibrary` aims to improve on sampling and computational efficiency over other sensitivity analysis packages while providing a diverse set of methods so that any problem can have a suitable implementation. We will discuss specific sensitivity and identifiability analysis methods `UQLibrary` implements. We then present how `UQLibrary` is formulated including dependencies, major classes and function, and parellelization. We then present results from two test cases and demonstrate the need for non-uniform parameter sampling in global sensitivity analysis. Finally, we discuss future steps for the project.

ORCID(s):

## 2. Methodology

Sensitivity analysis and identifiability analysis are diverse fields with numerous algorithms available. Algorithms are generally grouped into two classes; local and global. Local methods measure sensitivity or identifiability only at a particular parameter value while global methods measure over a range of parameter values. Global methods allow better capturing of uncertainty in parameters by expressing them with distributions rather than fixed values, but local methods are computationally more efficient and give better insight to behavior around a particular parameter set. In `UQLibrary`, we focus on four methods; local sensitivity analysis (LSA), parameter subset selection (PSS), analysis of variance (ANOVA), and Morris screening.

Local sensitivity analysis approximates the rate of change of each quantity of interest with respect to each parameter at a particular parameter location [3]. Parameter subset selection is a local identifiability method for determining a subset of parameters which influence responses [21]. ANOVA is a global sensitivity analysis method, usually measured by Sobol indices, which quantifies how model variance is attributed to variance in each parameter or combination of parameters [24]. Finally, Morris screening is a OAT quasi-global sensitivity analysis method that averages local effects over the distribution of parameter values [20]. These methods will provide tools for identifying influential parameters in CFD simulations and reduced-order models. We will identify their uses, caveats, and implementations within `UQLibrary`. For notation purposes, we define the function $f_i(\boldsymbol{\theta})$ to be the $i^{th}$ QOI value computed at the POI values $\boldsymbol{\theta}$.

### 2.1. Local Sensitivity Analysis

Local sensitivity analysis focuses on approximating the instantaneous rate of change of QOI with respect to each POI. To do this we compute the local sensitivity matrix,

$$\mathcal{X}_{ij}(\boldsymbol{\theta}^*) = \frac{\partial f_j}{\partial \theta_i}(\boldsymbol{\theta}^*) \,, \tag{1}$$

where $\boldsymbol{\theta}^*$ is a set of nominal parameter values [23]. Though the computation of partial derivatives is not unique to sensitivity analysis, local sensitivity indices provide a low computational cost characterization of behavior around critical values and are further used in other algorithms such as parameter subset selection.

### 2.2. Parameter Subset Selection

Parameter subset selection (PSS) is an algorithm which groups the set of parameters into identifiable and unidentifiable sets [21]. To select parameter subsets, PSS computes the Fisher information matrix which is defined by the symmetric positive definite matrix $\mathcal{F}(\boldsymbol{\theta}^*) = \mathcal{X}^T \mathcal{X}$ where $\mathcal{X}$ is the sensitivity matrix defined in Equation 1. The singular values of $\mathcal{F}$ are then computed, normalized so the maximum singular value is 1, and a value of statistical identifiability, $\eta$, is set. Note that `UQLibrary` uses $\eta = 10^{-4}$ as the base value. If the smallest singular value is less than $\eta$, the parameter with the largest magnitude in the singular vector corresponding to the smallest singular value is determined to be non-identifiable, that parameter is removed from $\mathcal{X}$, and the process is repeated until no singular values remain below the threshold.

Parameter subset selection is useful in models with many parameters since the variance in only a few of them may determine most of the model variance. Classifying which parameters are identifiable is often an end goal since it directly classifies which aspects of the model can be fixed to improve computational efficiency. Additionally, reducing the parameter dimension can increase the accuracy and efficiency of global sensitivity analysis by reducing the dimension of integral approximation. Parameter subset selection also has the significant advantage that it accounts for parameter correlation, which many other methods do not.

### 2.3. Morris Screening

Morris screening is a one-at-a-time quasi-global sensitivity analysis method, which approximates the local derivatives at parameter values drawn from an assumed distribution [20]. The primary metrics Morris screening calculates are the mean and and standard deviation of sensitivity, $\mu_i^*$ and $\sigma_i$, given by,

$$d_i^j = \frac{f(\xi, \theta^j + \Delta e_i) - f(\xi, \theta^j)}{\Delta} \,, \quad \mu_i^* = \frac{1}{N}\sum_{j=1}^{N} |d_i^j| \,, \quad \sigma_i = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N}(d_i^j - \mu_i)^2}, \tag{2}$$

for a parameter $i$ with $N$ parameter samples, a step-size of $\Delta$, and where $e_i$ is the unit vector for parameter $i$. Selection of $\Delta$ determines how finely the parameter space is searched around each parameter sample $\theta^j$, where very small values are provide a finite-difference derivative approximation whereas large values instead quantify large-scale function variations [23]. Like ANOVA, Morris screening requires assuming parameter distributions. However, OAT parameter variation means Morris screening is less dependent on an error in those assumptions since it only uses distributions for sampling parameter values, not the calculation of variance [23]. Therefore Morris screening is a reliable way to approximate the average variability of an output to a parameter over a distribution when Sobol' analysis is not applicable.

## 2.4. Analysis of Variance

Analysis of Variance (ANOVA) is a global sensitivity analysis method that quantifies the proportion of the total variance in the model outputs apportioned to each parameter or combination of parameters within their joint distributions [22]. `UQLibrary` performs ANOVA by calculating first-order and total Sobol' indices for each parameter. First-order Sobol' indices, $S_i$, quantify the variance in $f$ apportioned only to changes in parameter $i$, while the total Sobol' indices, $S_{T_i}$, quantify the variance in $f$ apportioned to parameters $i$ or any combination of two-parameter changes involving parameter $i$ [23]. In Sobol' analysis, if elements of $\boldsymbol{\theta}$ are independently distributed, $f(\boldsymbol{\theta})$ can be expressed as a hierarchical expansion,

$$f(\boldsymbol{\theta}) = f_0 + \sum_{i=1}^{p} f_i(\theta_i) + \sum_{1\le i<j\le p} f_{ij}(\theta_i,\theta_j) + \dots\,, \tag{3}$$

of conditional expected values. Each conditional expected value of order $n$ is defined by the expected value of $f$ conditioned on $n$ parameters, and subtracting the lower-order conditional expected values of each of those $n$ parameters [23]. Conditional expected values of order zero, one, and two are given by

$$f_0 = \mathbb{E}[f(\boldsymbol{\theta})]\,, \quad f_i(\theta_i) = \mathbb{E}[f(\boldsymbol{\theta})|\theta_i] - f_0\,, \quad f_{ij}(\theta_i,\theta_j) = \mathbb{E}[f(\boldsymbol{\theta})|\theta_i,\theta_j] - f_i(\theta_i) - f_j(\theta_j) - f_0\,. \tag{4}$$

To compute Sobol' indices of different orders, corresponding conditional expected values are integrated, and then normalized by the total function variance $D = var(f(\boldsymbol{\theta}))$. The first-order and total Sobol' indices respectfully are defined by,

$$S_i = \frac{1}{D}\int f_i^2(\theta)d\theta_i\,, \quad S_{T_i} = S_i + \frac{1}{D}\sum_{j=1}^{p}\int\int f_{ij}^2(\theta_i,\theta_j)d\theta_i d\theta_j\,. \tag{5}$$

We note that calculating Sobol' indices requires solving high dimension integrals over the sampling space for parameters which can be computationally intensive [15]. The efficiency of the integral approximations are improved by Saltelli approximation, which significantly reduces the number of function evaluations required for integral approximation compared to Monte Carlo sampling by using low-discrepancy sampling [22]. An essential caveat for utilizing Sobol' analysis is that it requires assuming independently-distributed parameters, which often is either unverifiable or not the case [23]. We note that extensions of the theory accommodate dependent parameters, but they require knowledge of the parameter distribution and can yield negative values which are difficult to interpret. Additionally, even with Saltelli approximation, accurate approximation of the integrals requires far more function evaluations than Morris screening.

## 2.5. Parameter Sampling

Most global sensitivity analysis relies on approximating an integral over the distribution of the parameters. Due to the potentially large number of parameters in a model, global sensitivity analysis usually does not use interpolatory quadrature methods which can become computationally intractable at high dimensions [19]. As a result, sampling from parameter distributions and applying Monte Carlo quadrature, shown in Equation 6, is usually the most basic way to approximate these integrals [20, 24].

$$\int_{\Omega} f(\theta)\, d\theta = \frac{1}{N}\sum_{i=1}^{N} f(\theta_i) \tag{6}$$

However, accuracy for Sobol analysis and Morris screening is greatly improved when utilizing quasi-Monte Carlo sampling [5]. Quasi-Monte Carlo sampling utilizes the same quadrature rule as Monte-Carlo, but instead samples parameters from a quasi-random low-discrepancy sample. The quasi-random samples approximate a uniform distribution by iteratively selecting points so that their distance from the existing points is maximized. This generates a sample that has a more uniform density over the domain than a random sample. The quasi-random samples can then be mapped to sample any other distributions by applying inverse cumulative distribution functions.

## 3. Implementation

UQLibrary is separated into five modules; __init__, __main__, lsa, gsa, and examples. The initialization module contains the primary class definitions and functions to be called by the user along with all functions for displaying and saving results. The main module runs the Ishigami example problem and shows the packages expected outputs. A set of example problems with constructed UQLibrary.Model and UQLibrary.Options variables are contained in examples and can be drawn using the function UQLibrary.examples.get_example.

The module lsa contains class definitions and functions for performing local sensitivity analysis and parameter subset selection along with functions for computing derivative approximations. The first class defined in lsa is LsaOptions and which holds hyper-parameter values for derivative approximation and parameter subset algorithm such as finite difference step size and the threshold for identifiability in parameter subset selection. The class LsaResults is also defined in lsa and holds numpy arrays for the local sensitivity indices, relative local sensitivity indices, active parameter subset, and inactive parameter subset computed in lsa. Local sensitivity indices and relative local sensitivity indices are computed using the function get_jacobian, which supports both first order finite difference and complex step derivative approximations. Active and inactive parameter subsets are computed using the function get_active_subset, which computes eigenvalue and eigenvectors of the Fisher information matrix using numpy.linalg.eig [2, 12].

The module gsa contains class definitions and functions for performing Morris screening and Sobol analysis along with functions for sampling parameters. The first class defined in lsa is GsaOptions and which holds hyper-parameter values for the number of samples to use in each method and the step-size for Morris screening. The class GsaResults is also defined in lsa and holds numpy arrays for the Morris and Sobol indices computed in gsa. Morris samples of the parameter space are generated in get_morris_poi_sample by the algorithm found in [23]. Morris indices are then computed using the formula in Equation 2. Sobol indices are computed in the function calculate_sobol according to the Saltelli approximation algorithm [22]. Parameter samples for both algorithms are constructed using sampling functions from numpy.random for random samples and scipy.qmc for quasi-Monte Carlo samples [2, 12, 25, 26]. Quasi-Monte Carlo samples are mapped to other distributions using inverse cumulative distribution functions from scipy [25, 26].

### 3.1. Dependencies

UQLibrary requires the packages; numpy [2, 12], scipy [25, 26], matplotlib [16, 17], mpi4py [8, 9], and tabulate [4]. Most iterable variables in UQLibrary are defined as numpy arrays including POI values, QOI values, and sensitivity indices. Singular value decomposition in parameter subset selection is also performed using numpy. All sampling functions are drawn from scipy including low-discrepancy sampling, distribution sampling functions, and inverse culmulative distribution functions used to map low-discrepency samples to non-uniform distributions. All parallelization in UQLibrary is distributed memory using mpi4py.MPI. Plots of parameter samples and tables of sensitivity results are created using matplotlib and tabulate respectively.

### 3.2. Using UQLibrary

Running UQLibrary requires writing a script which loads UQLibrary as a module and calls the function UQLibrary.run_uq, which takes objects of class UQLibrary.Model and UQLibrary.Options as input. UQLibrary.Options selects which components of UQLibrary to run, how to save or display results, and hyper-parameters for each algorithm.

The class UQLibrary.Model contains all information about the specific test problem to perform sensitivity analysis and identifiability on. The primary component of this class is the function UQLibrary.Model.eval_fcn which the user must define and takes a one or two dimensional numpy array of parameter values and outputs a one or two dimensional array of model outputs. If the input or outputs to UQLibrary.Model.eval_fcn are two dimensional, UQLibrary assumes the first dimension is the number of model evaluations to compute and the second dimension is the parameter values at each sample. UQLibrary.Model.eval_fcn must be able to take one dimensional inputs and two dimensional inputs for global sensitivity analysis methods.

The second class variable of UQLibrary.Model is base_poi which must be provided by the user and holds the nominal parameter values to perform local sensitivity analysis on and to scale Morris screening results to. Additional variables of UQLibrary.Model that do not require user specification are base_qoi, which is results of eval_fcn(base_poi) and is computed when objects of class model are intitialized. The variables n_poi and n_qoi are the number of POIs and QOIs computed from the shape of base_poi and base_qoi. Finally, dist_type is a string selecting the type of distribution to be used such as normal, uniform, quasi-normal, or quasi-uniform while dist_param is a two by n_poi numpy array holding the required parameters for each distribution such as the bounds for a uniform distribution or the mean and standard deviation of a normal distribution. These variables can be defined by the user but otherwise are assumed to be a quasi-uniform distribution over the hyper-cube $[0,1]^{\texttt{n\_poi}}$. Checks are written throughout the initialization of Model to ensure all entered parameters have matching types and shapes to ensure successful computation in run_uq.

When run_uq is run, it outputs a variable of class UQLibrary.Results which holds numpy arrays of all the outputs from methods that were specified to be performed in UQLibrary.Options. These results can also be printed to the console or saved in a text file using tables constructed with tabulate [4]. If Sobol analysis was applied, scatterplots of POI samples and their correlation to QOIs can also be plotted using matplotlib [16, 17].

### 3.3. Parellelization

Model computations in UQLibrary can be parallelized using the distributed memory MPI implementation of mpi4py [8, 9]. Since UQLibrary relies on distributed memory, any data required for performing model computations must be accessible by every thread. To parallelize model computations, whenever a POI sample with multiple samples would be given to model.eval_fcn(), UQLibrary separates that sample into portions distributed to each thread and then recollects the QOIs results into a single numpy array. When run on every thread UQLibrary.run_uq will automatically perform low cost computations and parameter sample generation on the base thread and broadcast results to all other threads. Only the base thread prints, plots, and saves results but every thread outputs a variable of class UQLibrary.Results from UQLibrary.run_uq.

The module

## 4. Results

To analyze UQlibrary outputs, we first consider the example of steady-state temperature distribution, $T_s(x)$ along an aluminum rod with a heat source at one end, which is given by the ODE in Equation 7. We consider the case of a rod with height and width $a = b = .95cm$ and length $L = 70cm$, ambient temperature $T_{amb} = 21.19\deg C$, thermal convectivity $k = 2.37\frac{W}{cm\cdot C}$, a convective heat transfer coefficient $h \sim \mathcal{N}(0.00191, 1.4482\cdot 10^{-5})$, and source flux $\Phi \sim \mathcal{N}(-18.4, 0.1450)$ [23].

$$\begin{cases} \frac{d^2T_s}{dx^2} = \frac{2(a+b)h}{ab}\frac{h}{k}\left(T_s(x) - T_{amb}\right) \\ \frac{dT_s}{dx}(0) = \frac{\Phi}{k} \\ \frac{dT_s}{dx}(L) = \frac{h}{k}\left(T_{amb} - T_s(L)\right) \end{cases} \tag{7}$$

The ODE has solution,

$$T_s(x,\theta) = T_s(x,\theta) = c_1(\theta)e^{-\gamma x} + c_2(\theta)e^{\gamma x} + T_{amb}, \tag{8}$$

where $\gamma = \sqrt{\frac{2(a+b)h}{abk}}$ and

$$c_1(\theta) = -\frac{\Phi}{k\gamma}[\frac{e^{\gamma L}(h+k\gamma)}{e^{-\gamma L}(h-k\gamma) + e^{\gamma L}(h+k\gamma)}],\ c_2(\theta) = \frac{\Phi}{k\gamma} + c_1(\theta).$$

We aim to analyze the sensitivity of QOI $[T_s(x = 55), \theta]$ with respect to POIs $\theta = [\Phi, h]$. After defining the required model and options objects required for the package, we call UQlibrary.run_uq(model, options) to get the package outputs.

| Parameter | Sensitivity Index | Relative Sensitivity Index |
|---|---|---|
| $\Phi$ | -0.34405 | 0.23003 |
| $h$ | -7396.74 | 0.51335 |

**Table 1**
Sensitivity indices for heat transfer through an insulated rod. Relative sensitivities reveal that temperature is only $h$ is twice as sensitive to $\Phi$ relative to their nominal values though $h$'s sensitivity index is over $20,000$ times larger than $\Phi$'s.

| | | $S_i$ | $S_{T_i}$ |
|---|---|---|---|
| Uniform Distribution | $\Phi$ | .1780 | .1782 |
| | $h$ | .8227 | .8218 |
| Normal Distribution | $\Phi$ | 0.1777 | .1782 |
| | $h$ | 0.8238 | .8216 |

**Table 2**
Global sensitivity indices for heat transfer through an insulated rod. We were unable to achieve convergence past the first decimal place for first order indices ($S_i$) for normal distributions, but total effect indices ($S_{T_i}$) indicate that uniform and normal distributions produce the same sensitivity indices for this problem.

## 4.1. Local Sensitivity Analysis

The local sensitivity indices, shown in Table 1, reveal that temperature's sensitivity to $h$ is $20,000$ times larger than to $\Phi$, but that its relative sensitivity to $h$ is only approximately twice its relative sensitivity to $\Phi$. The sensitivity indices alone may indicate that changes to $\Phi$ have a minimal effect on the temperature compared to $h$, however, the relative sensitivities clarify that, for perturbations proportional to the parameter's nominal values, $\Phi$ and $h$ affect temperature at the same order of magnitude.

## 4.2. Global Sensitivity

We next analyze the global sensitivity of $\Phi$ and $h$. We first calculate Sobol indices using the normal distributions that the parameters follow, and then compare those results to those calculated using uniform distributions with the same mean and variance. Sobol indices for the heated rod, Table 2, first show that that the temperature $55cm$ away from the heat source is approximately 4× more sensitive to $h$ than to $\Phi$. Additionally, since $S_i \approx S_{T_i}$ for both parameters, there is effectively no interaction non-linear between the parameter. This result is expected since in the formula for the temperature, $\Phi$ only appears as a multiplier of terms involving $h$. Additionally, the Sobol indices from uniformly and normally distributed parameters within .1% and .01% of each other for $S_i$ and $S_{T_i}$ respectively. This highlights that, in this case, the uniform distribution was suitable, and the option of normally distributed parameters provided by our package did not improve results.

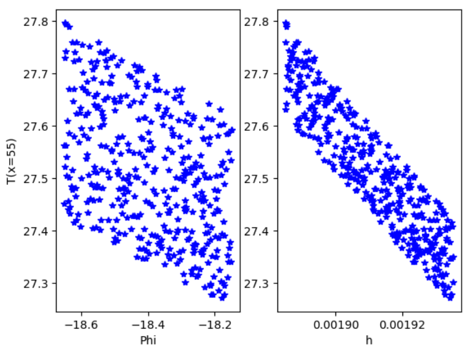
## 4.3. Ishigami Function

To further investigate this we refer to the Ishigami function, Equation 9, a common test problem for sensitivity analysis due to its nonlinear relationships between POIs, $(x_1, x_2, x_3)$, and QOIs, $f(x)$ [18]. Uniform distributions from $-\pi$ to $\pi$ for POIs are usually chosen for this problem, but we investigated how results might change if we assume all parameters are normally distributed with equal mean and variance.

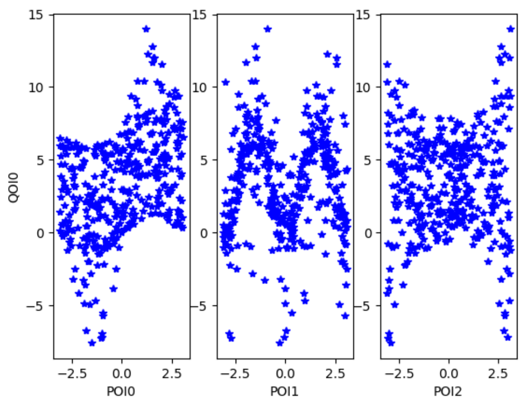
$$f(x) = \sin(x_1) + a\sin^2(x_2) + bx_3^4\sin(x_1) \tag{9}$$

Table 3 shows our Sobol indices for the Ishigami function with both uniform and normally distributed parameters. Our results for the uniform distribution are equivalent to those found by another sensitivity analysis package [14]. Using a normal distribution, the ordering of most sensitive POIs for both first-order and total effects changed. Many changes were found, but the most critical are that $x_2$ was the most sensitive parameter for first-order effects under a uniform distribution but had minimal first and total effects under a normal distribution. Additionally, first-order effects accounted for most of the variance under a uniform distribution since $\sum S_i \approx .75$, but accounted for very little under a normal distribution since $\sum S_i \approx .21$.

Correlation plots between POIs and QOIs, Figure 1, reveals why distribution selection was critical for the Ishigami function but negligible for the heated rod. Within the POI sampling bounds for the heated rod, the temperature was



(a) Heated rod



(b) Ishigami

**Figure 1:** POI-QOI correlation plots for the Heated rod and Ishigami examples. Distribution selection was negligible for the heated rod likely because the temperature was approximately linearly associated with the POIs within the sampled region, while the highly non-linear relationship between the POIs and Ishigami function changed Sobol indices with distribution selection.

| | | $S_i$ | $S_{T_i}$ |
|---|---|---|---|
| Uniform Distribution | $x_1$ | 0.3139 | 0.5579 |
| | $x_2$ | 0.4424 | 0.4424 |
| | $x_3$ | $1.42 \times 10^{-5}$ | 0.2437 |
| Normal Distribution | $x_1$ | 0.1264 | 0.9109 |
| | $x_2$ | 0.0861 | 0.0860 |
| | $x_3$ | 0.0027 | 0.7878 |

**Table 3**
Global sensitivity indices for Ishigami function. Due to the non-linear relationship between the QOI and POIs, distribution selection significantly impacts Sobol indices.

approximately linearly dependent on the POIs, so adjusting where the domain samples were drawn from did not significantly impact relative variance. However, the Ishigami function is highly nonlinearly dependent on its POIs, so drawing a higher density of samples from near the nominal value, as a normal distribution does, significantly changes the QOI variance with respect to POIs.

## 5. Conclusion

`UQLibrary` is a Python implementation of a diverse set of sensitivity and identifiability analysis methods. We use local, global, one-at-a-time, and variance based methods so that a suitable method is available for most any problem and all methods utilize the same function for the model. We allow for sampling from non-uniform distributions, a feature unavailable in other packages, and demonstrate that limiting analysis to uniform distributions can cause significant errors for nonlinear models. We also allow parallelization of model computations so that global sensitivity analysis of computationally expensive models is feasible.

Future development opportunities for `UQLibrary` include adding more derivative approximation methods and allowing for sampling of parameters from different distribution types in a single model. `UQLibrary` currently only implements finite difference derivative approximation methods in first order finite difference and complex step. Though these methods are suitable for local sensitivity analysis for most problems, accuracy of first-order finite difference is severely limited by catastrophic cancellation and the imaginary parameter values complex step requires usable for all models. Accuracy and efficiency of analyzing some models could be improved by implementing automatic differentiation or adjoint equations. Automatic differentiation tracks all mathematical operations taken within a function and applied chain rule to analytical solve for the derivatives [13]. Adjoint equations can be derived from a number of different models, particularly CFD simulations, and can be used to compute the sensitivity of parameters analytically. When available, this is a more accurate and computationally cheaper approach that can make local analysis of a computationally expensive problem easy [6]. Both of these methods have the ability to significantly increase accuracy and decrease computation time on problems for suitable problems.

As shown in Section 4.3, accurate selection of parameter distributions can greatly influence sensitivity result in nonlinear problems. `UQLibrary` makes advancements by allowing sampling of parameters from non-uniform distributions, but is still limited by requiring all parameters be from the same distribution type, i.e. uniform, normal, beta, ect. Allowing for mixing of distribution types when sampling parameters will further increase the robustness that `UQLibrary` provides over other packages in Python.

## References

[1] Adams, B.M., Bohnhoff, W.J., Dalbey, K.R., Edeida, M.S., Eddy, J.P., Eldred, M.S., Hooper, R.W., Hough, P.D., Hu, K.T., Jakeman, J.D., Kahlil, M., Maupin, K.A., Monschke, J.A., Ridgway, E.M., Rushdi, A.A., Seidl, D.T., Stephens, J.A., Swiler, L.P., Winokur, J.G., 2021. Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analyiss: Version 6.14 user's manual.

[2] Archibald, A., van Beek, B., Harris, C., Oh, D.K., Shoudhury, K., Adel, S., Berg, S., . Numpy v1.20.3. URL: https://github.com/numpy/numpy/releases/tag/v1.20.3.

[3] Arriola, L., Hyman, J., 2009. Sensitivity analysis for uncertainty quantification in mathematical models. Mathematical and Statistical Estimation Approaches in Epidemiology , 195–247doi:10.1007/978-90-481-2313-1_10.

[4] Astanin, S., . Tabulate v0.8.9. URL: https://github.com/astanin/python-tabulate/tree/v0.8.9.

[5] Burhenne, S., Jacob, D., Henze, G., 2011. in: Sampling based on Sobol' sequences for Monte Carlo techniques applied to building simulations, pp. 1816–1823.

[6] Cao, Y., Li, S., Petzold, L., 2002. Adjoint sensitivity analysis for differential-algebraic equations: algorithms and software. Journal of Computational and Applied Mathematics 149, 171–191. URL: https://www.sciencedirect.com/science/article/pii/S0377042702005289, doi:https://doi.org/10.1016/S0377-0427(02)00528-9. scientific and Engineering Computations for the 21st Century - Me thodologies and Applications Proceedings of the 15th Toyota Conference.

[7] Constantine, P.G., 2015. Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies. Society for Industrial and Applied Mathematics, USA.

[8] Dalcin, L., Fang, Y.L.L., . Mpi4py v3.1.3. URL: https://github.com/mpi4py/mpi4py/tree/3.1.3.

[9] Dalcin, L., Fang, Y.L.L., 2021. mpi4py: Status update after 12 years of development. Computing in Science Engineering 23, 47–54. doi:10.1109/MCSE.2021.3083216.

[10] DM, H., 1995. A comparison of sensitivity analysis techniques. Health Phys , 195–204doi:10.1097/00004032-199502000-00005.

[11] Hanes, H., . Uqlibrary v0.1.2. URL: https://github.com/HarleyHanes/UQLibrary-CourseProject, doi:10.5281/zenodo.6363991.

[12] Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., del Río, J.F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., Oliphant, T.E., 2020. Array programming with NumPy. Nature 585, 357–362. URL: https://doi.org/10.1038/s41586-020-2649-2, doi:10.1038/s41586-020-2649-2.

[13] Hascoet, L., Pascual, V., 2013. The tapenade automatic differentiation tool: Principles, model, and specification. ACM Trans. Math. Softw. 39. URL: https://doi.org/10.1145/2450153.2450158, doi:10.1145/2450153.2450158.

[14] Herman, J., Usher, W., 2017. Salib: An open-source python library for sensitivity analysis. The Journal of Open Source Software 2, 97. doi:10.21105/joss.00097.

[15] Homma, T., Saltelli, A., 1996. Importance measures in global sensitivity analysis of nonlinear models. Reliability Engineering and System Safety 52, 1–17. URL: https://www.sciencedirect.com/science/article/pii/0951832096000026, doi:https://doi.org/10.1016/0951-8320(96)00002-6.

[16] Hunter, J.D., . Matplotlib v3.4.3. URL: https://github.com/matplotlib/matplotlib/tree/v3.4.3.

[17] Hunter, J.D., 2007. Matplotlib: A 2d graphics environment. Computing in Science & Engineering 9, 90–95. doi:10.1109/MCSE.2007.55.

[18] Ishigami, T., Homma, T., 1990. An importance quantification technique in uncertainty analysis for computer models, in: [1990] Proceedings. First International Symposium on Uncertainty Modeling and Analysis, pp. 398–403. doi:10.1109/ISUMA.1990.151285.

[19] Lemieux, C., 2009. Monte Carlo and Quasi-Monte Carlo Sampling. Springer, New York, NY, USA.

[20] Morris, M.D., 1991. Factorial sampling plans for preliminary computational experiments. Technometrics 33, 161–174.

[21] Quaiser, T., Mönnigmann, M., 2009. Systematic identifiability testing for unambiguous mechanistic modeling – application to jak-stat, map kinase, and nf- b signaling pathway models. BMC Systems Biology 3. doi:10.1186/1752-0509-3-50.

[22] Saltelli, A., Annoni, P., Azzini, I., Campolongo, F., Ratto, M., Tarantola, S., 2010. Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index. Computer Physics Communications 181, 259–270. URL: https://www.sciencedirect.com/science/article/pii/S0010465509003087, doi:https://doi.org/10.1016/j.cpc.2009.09.018.

[23] Smith, R.C., 2013. Uncertainty Quantification: Theory, Implementation, and Applications. SIAM Computational Science and Engineering.

[24] Sobol, I., 2001. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. Mathematics and Computers in Simulation 55, 271–280. URL: https://www.sciencedirect.com/science/article/pii/S0378475400002706, doi:https://doi.org/10.1016/S0378-4754(00)00270-6. the Second IMACS Seminar on Monte Carlo Methods.

[25] Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors, . Scipy v1.7.1. URL: `https://github.com/scipy/scipy/tree/v1.7.1`.

[26] Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors, 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods 17, 261–272. doi:`10.1038/s41592-019-0686-2`.