pyVISCOUS: An open-source tool for computationally frugal global sensitivity analysis

Hongli Liu[1], Martyn P. Clark[2], Shervan Gharari[3], Razi Sheikholeslami[4], Jim Freer[5], Wouter J. M. Knoben[6], and Simon M. Papalexiou[7]

[1]Centre for Hydrology, University of Saskatchewan
[2]Centre for Hydrology, University of Saskacthewan Coldwater Laboratory
[3]University of Saskatchewan Coldwater Laboratory
[4]University of Oxford
[5]Univerrsity of Saskatchewan
[6]University of Saskatchewan
[7]University of Calgary

November 26, 2022

Abstract

Sensitivity analysis is used to increase our understanding of the evaluated model and ease model parameter estimation. VISCOUS (VarIance-based Sensitivity analysis using COpUlaS) is a given-data, computationally frugal variance-based global sensitivity analysis framework. Grounded in Copula theory, VISCOUS computes the Sobol sensitivity indices using a probability model that describes the relationship between model inputs (e.g., the perturbations in the model parameters) and outputs (e.g., the model responses given a parameter perturbation). In this technical note, we make three contributions to make the VISCOUS framework easier to understand and apply. First, we provide additional derivations of VISCOUS to connect the VISCOUS framework to recent developments in the data science community. We provide didactic examples with simple test functions in order to help a wider group of modelers understand the underpinnings of the VISCOUS framework. Second, we evaluate the VISCOUS framework using three types of Sobol functions and provide a cautionary note on using VISCOUS to approximate Sobol' sensitivity indices for applications where model inputs are of similar importance. Third, we provide an open-source code of VISCOUS in Python, namely, pyVISCOUS. pyVISCOUS is model-independent and can be applied with user-provided input-output data.

Hosted file

`essoar.10512586.1.docx` available at https://authorea.com/users/555369/articles/605997-pyviscous-an-open-source-tool-for-computationally-frugal-global-sensitivity-analysis

pyVISCOUS: An open-source tool for computationally frugal global sensitivity analysis

Hongli Liu[1], Martyn P. Clark[1], Shervan Gharari[2], Razi Sheikholeslami[3,4], Jim Freer[1], Wouter J. M. Knoben[1], Simon Michael Papalexiou[5]

[1] Centre for Hydrology, University of Saskatchewan, Canmore, Alberta, Canada

[2] Centre for Hydrology, University of Saskatchewan, Saskatoon, Saskatchewan, Canada

[3] Department of Civil Engineering, Sharif University of Technology, Tehran, Iran

[4] School of Geography and the Environment, University of Oxford, Oxford, UK

[5] Department of Civil Engineering, University of Calgary, Alberta, Canada

Corresponding author: Hongli Liu (`hongli.liu@usask.ca`)

Key Points:

- We provide didactic examples and additional background material to make VISCOUS easier to understand and apply for general readers.
- We provide a cautionary note on using VISCOUS to approximate Sobol' sensitivity indices when model inputs are of similar importance.
- We provide an open-source code of VISCOUS in Python, namely, pyVISCOUS.

Abstract

Sensitivity analysis is used to increase our understanding of the evaluated model and ease model parameter estimation. VISCOUS (VarIance-based Sensitivity analysis using COpUlaS) is a given-data, computationally frugal variance-based global sensitivity analysis framework. Grounded in Copula theory, VISCOUS computes the Sobol sensitivity indices using a probability model that describes the relationship between model inputs (e.g., the perturbations in the model parameters) and outputs (e.g., the model responses given a parameter perturbation). In this technical note, we make three contributions to make the VISCOUS framework easier to understand and apply. First, we provide additional derivations of VISCOUS to connect the VISCOUS framework to recent developments in the data science community. We provide didactic examples with simple test functions in order to help a wider group of modelers understand the underpinnings of the VISCOUS framework. Second, we evaluate the VISCOUS framework using three types of Sobol functions and provide a cautionary note on using VISCOUS to approximate Sobol' sensitivity indices for applications where model inputs are of similar importance. Third, we provide an

open-source code of VISCOUS in Python, namely, pyVISCOUS. pyVISCOUS is model-independent and can be applied with user-provided input-output data.

Introduction

Sensitivity analysis (SA) investigates how the uncertainty of model output can be attributed to the different uncertain input factors and their interactions (Pianosi et al., 2016). SA is useful in many ways, such as ranking input factors in order of sensitivity, fixing negligible factors to reduce the dimensionality of parameter estimation problems, determining the region of the input space that has a substantial control on model output (e.g., extreme flows) , and prioritizing data acquisition processes to those aspects where they will have the largest impact on the desired outcomes (Nossent et al., 2011; Razavi and Gupta, 2015; Saltelli et al., 2008; van Griensven et al., 2006). SA can also lend insight into the dominant processes that govern spatiotemporal variability of a system by exploring the full spectrum of its behavior (Demaria et al., 2007; Markstrom et al., 2016; Razavi et al., 2021).

SA methods can be generally classified into local and global methods. Local SA evaluates the effects of the input variations around a specific point in the input space, and global SA evaluates the effects of the input variations across the entire variability space (Pianosi et al., 2016). In SA, there are different methods to define sensitivity indices from the input-output data. The commonly-used SA methods are based on one of the following two approaches: (1) an analysis of derivatives, for example the method of Morris (Campolongo et al., 2007; Morris, 1991; Rakovec et al., 2014), and (2) an analysis of variance, for example the method of Sobol (Homma and Saltelli, 1996; Sobol, 2001).

Variance-based methods are attractive because they are model independent, they measure sensitivity across the whole input space, they can deal with non-linear input-output relationship, they measure interaction effects among input factors, and they handle groups of input factors (Saltelli et al., 2008). The major challenge associated with application of variance-based methods is their computational cost, because they require model evaluations for a considerable number of input samples. Running a model for a large number of input samples may be difficult to achieve if the prediction model is computationally expensive. Therefore, much recent research aims to find efficient numerical algorithms to compute variance-based sensitivity indices. (Hu and Mahadevan, 2019; Sheikholeslami et al., 2019)

To overcome the aforementioned computational bottleneck, Sheikholeslami et al. (2021) developed a computationally frugal global SA framework called VISCOUS (VarIance-based Sensitivity analysis using COpulaS). VISCOUS first uses a Gaussian Mixture Copula Model (GMCM) to approximate the joint probability distribution between the input (e.g., the perturbations in the model parameters) and output data (e.g., the model responses given a parameter per-

turbation); and then computes the Sobol sensitivity indices based on the GMCM probability model. The input-output data are not used to directly calculate the Sobol sensitivity indices but are used to train the GMCM, so the input-output data do not need to follow specific sampling strategies (e.g., as required in the Sobol method). VISCOUS is then a "given-data" sensitivity analysis method, in which the GMCM can be developed using any existing input-output data, and no additional prediction model runs are needed when input-output data are already available. For example, the input-output data can be from the previous model runs generated from other modeling purposes, such as uncertainty propagation and model calibration. Therefore, the benefit of VISCOUS is that it provides useful approximations of the Sobol sensitivity indices without the need to produce an extensive sample of new model simulations that follow the Sobol sampling strategy.

The purpose of this technical note is to make it easier for readers to understand and apply the VISCOUS framework. Our specific objectives are to: (i) provide didactic examples and additional background material to make VISCOUS easier to understand and apply for general readers; (ii) evaluate VISCOUS using three types of Sobol functions and provide a cautionary note on using VISCOUS to approximate Sobol' sensitivity indices for applications where model inputs are of similar importance; (iii) provide an open-source code of VISCOUS in Python, namely, pyVISCOUS.

The reminder of the note is organized as follows. The didactic examples and additional background material are in Sections 2 (Methodology) and 3 (Implementation). VISCOUS is evaluated in Section 4. The pyVISCOUS code is described in Section 5. The note concludes with discussion of potential utility of VISCOUS for different modelling applications.

Methodology of the VISCOUS Framework

This section provides additional background on the VISCOUS framework to connect VISCOUS to recent developments in the data science community and make VISCOUS easier to understand for general readers. In this section we first provide basic knowledge about the copula function, the Gaussian Mixture Model (GMM), and the Gaussian Mixture Copula Model (GMCM). This will prepare the ground for introducing the followed derivations. We then provide step-by-step derivations of the first- and total-order Sobol sensitivity indices using the GMCM, providing details that are not provided in Sheikholeslami et al. (2021). Finally, we present Monte Carlo-based approximations of both the first- and total-order Sobol sensitivity indices, the latter of which is not defined explicitly in Sheikholeslami et al. (2021). These additions are also important to evaluate the VISCOUS framework in Section 4.

The Gaussian Mixture Copula Model (GMCM)

Copula function

Assume a random vector of input-output data, i.e., $[x_1, \dots, x_m, y]$, each has a continuous cumulative distribution function (CDF) $F_{X_i}(x_i)$ and $F_Y(y)$, where $i = [1, .., m]$. According to the probability integral transform theorem, if x is a continuous random variable with CDF $F_X(x)$, then $F_X(x)$ has a uniform distribution on $[0, 1]$. The probability integral transformation is given as:

$$\left[u_{x_1}, \dots, u_{x_m}, u_y\right] = \left[F_{X_1}(x_1), \dots, F_{X_m}(x_m), F_Y(y)\right] \quad (SEQ\ Equation \setminus * ARABIC\ 1)$$

where $[u_{x_1}, \dots, u_{x_m}, u_y]$ is a vector over $[0, 1]^{m+1}$. Each $u \in [0, 1]$ follows the uniform distribution.

According to Sklar's theorem (Sklar, 1959; Tewari et al., 2011), the joint distribution $F_{X,Y}(x_1, \dots, x_m, y)$ of random variables $(x_1, \dots, x_m, y)$ can be expressed as a function of the marginal distributions $u_{x_1}, \dots, u_{x_m}, u_y$:

$$F_{X,Y}(\mathbf{x}, y) = C\left(u_{x_1}, \dots, u_{x_m}, u_y\right) \quad (SEQ\ Equation \setminus * ARABIC\ 2)$$

where $\mathbf{x} = [x_1, \dots, x_m]$, $F_{X,Y}(\mathbf{x}, y)$ is the joint CDF of random variables $(\mathbf{x}, y)$. C is the copula function, $C : [0, 1]^{m+1} \longmapsto [0, 1]$. The copula function C is a joint CDF. It takes the marginal CDFs $\left(u_{x_1}, \dots, u_{x_m}, u_y\right)$ as inputs, thereby connecting the marginal CDFs to the joint CDF (Hu and Mahadevan, 2019).

By computing the derivatives of Equation (2), we get the joint probability density function (PDF), $f_{X,Y}(\mathbf{x}, y)$, expressed as:

$$f_{X,Y}(\mathbf{x}, y) = \frac{C\left(u_{x_1}, \dots, u_{x_m}, u_y\right)}{\partial u_{x_1} \cdots \partial u_{x_m} \bullet \partial u_y} \bullet \frac{\partial u_{x_1}}{x_1} \dots \frac{\partial u_{x_m}}{x_m} \bullet \frac{\partial u_y}{y}$$

$$= \frac{C\left(u_{x_1}, \dots, u_{x_m}, u_y\right)}{\partial u_{x_1} \cdots \partial u_{x_m} \bullet \partial u_y} \bullet \frac{\partial F_{X_1}(x_1)}{x_1} \dots \frac{\partial F_{X_m}(x_m)}{x_m} \bullet \frac{\partial F_Y(y)}{y}$$

$$= c\left(u_{x_1}, \dots, u_{x_m}, u_y\right) \bullet f_{X_1}(x_1) \cdots f_{X_m}(x_m) \bullet f_Y(y) \quad (SEQ\ Equation \setminus * ARABIC\ 3)$$

where c is the copula density, $f_X(x)$ and $f_Y(y)$ are the marginal PDFs of variables x and y, respectively.

Gaussian Mixture Model (GMM)

GMM is a parametric PDF represented as a weighted sum of Gaussian probabilistic densities. If each Gaussian probabilistic density represents a cluster, then a GMM is a clustering algorithm that identifies the probability that each data belongs to each cluster (Singh, 2019; Xu and Jordan, 1996).

Assume that the input variables $\mathbf{z}_x$ and z_y are in normal space and follow the standard normal distribution. Their joint PDF is expressed by a GMM as:

$f_{Z_X,Z_Y}\left(\mathbf{z}_x\ ,z_y\right)=\sum_{k=1}^{K}\lambda_k \bullet \phi(\mathbf{z}_x,z_y|\ _k,\ _k)$ (*SEQ Equation* * *ARABIC* 4)

where $\mathbf{z}_x$ is a m-dimensional vector, $\mathbf{z}_x$= $[z_{x_1},\ldots,z_{x_m}]$. K is the total number of Gaussian components (or clusters). λ_k is the weight of the k^{th} Gaussian component. $\lambda_k > 0$ and $\sum_{k=1}^{K}\lambda_k = 1$. ϕ is the PDF of a $(m+1)$-variate Gaussian distribution with mean $_k$ and covariance $_k$.

$$_k=\left[\ _{k,z_x}\ ,\mu_{k,z_y}\right]\ (\ \textit{SEQ Equation}\ \backslash *\ \textit{ARABIC}\ 5)$$

$$_k=\begin{bmatrix} _{k,z_xz_x} & _{k,z_xz_y} \\ _{k,z_yz_x} & \sigma^2_{k,z_y}\end{bmatrix}\ (\ \textit{SEQ Equation}\ \backslash *\ \textit{ARABIC}\ 6)$$

where $_{k,z_xz_x}$ is the variance of and covariance between $\mathbf{z}_x$, $_{k,z_xz_y}$ is the covariance between $\mathbf{z}_x$ and z_y, and $_{k,z_xz_y}\ =\ _{k,z_yz_x}$. σ^2_{k,z_y} is the variance of z_y.

The GMM parameters (, ,) are usually estimated by the expectation-maximation (EM) algorithm. The details of the EM algorithm are in **Appendix A1**. The number of Gaussian components K can be selected from a set of GMMs using the Bayesian Information Criterion (BIC).

Further, the conditional PDF of z_y given $\mathbf{z}_x$, $f_{Z_Y|Z_X}\left(z_y|\mathbf{z}_x\right)$ can be derived by:

$$f_{Z_Y|Z_X}\left(z_y|\mathbf{z}_x\right)=f_{X,Y}\left(\mathbf{z}_x,z_y\right)/f_X\left(\mathbf{z}_x\right)\ (\ \textit{SEQ Equation}\ \backslash *\ \textit{ARABIC}\ 7)$$

where

$$f_{Z_X}\left(\mathbf{z}_x\right)=\sum_{k=1}^{K}\lambda_k \bullet \phi(\mathbf{z}_x|\ _{k,z_x},\ _{k,z_xz_x})\ (\ \textit{SEQ Equation}\ \backslash *\ \textit{ARABIC}\ 8)$$

Gaussian Mixture Copula Model (GMCM)

There are various forms of copula functions. Traditional copula functions are usually defined for bivariate problems, and only a few functions, such as the Gaussian copula and the student's t copula, are well-studied for the multivariate high-dimensional cases (Hu and Mahadevan, 2019). In GMCM, the Gaussian copula function is adopted to model the correlation of random variables.

A Gaussian copula, C^G, has the form of:

$$C^G(u)=\Phi_R(\Phi^{-\mathbf{1}}(u_{x_1}),\ldots,\Phi^{-\mathbf{1}}(u_{x_m}),\Phi^{-\mathbf{1}}(u_y))\ (\ \textit{SEQ Equation}\ \backslash *\ \textit{ARABIC}\ 9)$$

where Φ_R is the joint CDF of a multivariate normal distribution with mean vector zero and covariance matrix equal to the correlation matrix $R\in[-1,1]^{(m+1)\times(m+1)}$. Φ^{-1} is the inverse CDF in the standard normal distribution.

The Gaussian copula in Equation (9) operates on $(\Phi^{-\mathbf{1}}(u_{x_1}),\ldots,\Phi^{-\mathbf{1}}(u_{x_m}),\Phi^{-\mathbf{1}}(u_y))$. For ease of notation, let $\mathbf{z}=\left[z_{x_1},\ldots,z_{x_m},z_y\right]=[\Phi^{-\mathbf{1}}\left(u_{x_1}\right),\ldots,\Phi^{-\mathbf{1}}\left(u_{x_m}\right),\Phi^{-\mathbf{1}}\left(u_y\right)]$.

$\mathbf{z}$ is called the standardized vector. Each element of $\mathbf{z}$ follows the standard normal distribution. Importantly, $F_{x_i}(x_i) = u_{x_i} = \Phi\left(z_{x_i}\right)$, $F_y(y) = u_y = \Phi(z_y)$, where $i = [1, 2, .., m]$.

Based on Equation (2), we compute the derivatives of Equation (9) to get the joint PDF of $(\mathbf{x}, y)$, $f_{X,Y}(\mathbf{x}, y)$:

$$f_{X,Y}(\mathbf{x}, y) = \frac{\partial^{m+1}\Phi_R\left(\Phi^{-1}(u_{x_1}),\ldots,\Phi^{-1}(u_{x_m}),\Phi^{-1}(u_y)\right)}{\Phi^{-1}(u_{x_1})\bullet\ldots\bullet\Phi^{-1}(u_{x_m})\bullet\Phi^{-1}(u_y)} \bullet \frac{\Phi^{-1}(u_{x_1})}{\partial u_{x_1}} \cdots \frac{\Phi^{-1}(u_{x_m})}{\partial u_{x_m}} \bullet \frac{\Phi^{-1}(u_y)}{u_y} \bullet \frac{\partial u_{x_1}}{x_1} \ldots \frac{\partial u_{x_m}}{x_m} \bullet \frac{\partial u_y}{y} \quad (SEQ\ Equation \setminus * ARABIC\ 10)$$

where $\frac{\partial^{m+1}\Phi_R\left(\Phi^{-1}(u_{x_1}),\ldots,\Phi^{-1}(u_{x_m}),\Phi^{-1}(u_y)\right)}{\Phi^{-1}(u_{x_1})\bullet\ldots\bullet\Phi^{-1}(u_{x_m})\bullet\Phi^{-1}(u_y)}$ is the Gaussian copula density $c^G(u)$. Since $\frac{\Phi^{-1}(u)}{\partial u} = \frac{1}{\phi(\Phi^{-1}(u))} = \frac{1}{\phi(z)}$, where ϕ is the PDF of the standard normal distribution, Equation (10) can be expressed as:

$$f_{X,Y}(\mathbf{x}, y) = c^G(u) \times \frac{1}{\phi(z_{x_1})} \bullet \cdots \frac{1}{\phi(z_{x_m})} \bullet \frac{1}{\phi(z_y)} \times f_{X_1}(x_1) \cdots f_{X_m}(x_m) \bullet f_Y(y) \quad (SEQ\ Equation \setminus * ARABIC\ 11)$$

In GMCM, the Gaussian copula density $c^G(u)$ of Equation (11) is approximated by the GMM. As such, GMM is combined with the copula. Equation (11) is approximated as:

$$f_{X,Y}(\mathbf{x}, y) \approx f^{\mathrm{GMM}}(\mathbf{z}) \times \frac{1}{\phi(z_{x_1})} \bullet \cdots \frac{1}{\phi(z_{x_m})} \bullet \frac{1}{\phi(z_y)} \times f_{X_1}(x_1) \cdots f_{X_m}(x_m) \bullet f_Y(y) \quad (SEQ\ Equation \setminus * ARABIC\ 12)$$

where $f^{\mathrm{GMM}}(\mathbf{z})$ is an estimated GMM. Note that $f^{\mathrm{GMM}}(\mathbf{z}) = \sum_{k=1}^{K} \lambda_k \bullet \phi(\mathbf{z}|\ _k,\ _k)$ according to Equation (4).

In addition, based on Equations (12) and (8), we can get the marginal PDF of variable $\mathbf{x}$, $f_X(\mathbf{x})$:

$$f_X(\mathbf{x}) \approx f_{\mathrm{GMM}}(z_{x_1}, \ldots, z_{x_m}) \times \frac{1}{\phi(z_{x_1})} \bullet \cdots \frac{1}{\phi(z_{x_m})} \times f_{X_1}(x_1) \cdots f_{X_m}(x_m) \quad (SEQ\ Equation \setminus * ARABIC\ 13)$$

where $f_{\mathrm{GMM}}(z_{x_1}, \ldots, z_{x_m})$ is derived from $f^{\mathrm{GMM}}(\mathbf{z})$ based on Equation (8). From Equations (12) and (13), we see that GMCM operates on both the observed variables $(x_1, \ldots, x_m, y)$ and the standardized variables $(z_{x_1}, \ldots, z_{x_m}, z_y)$.

Estimation of variance-based sensitivity indices using GMCM

This section provides derivations of the first- and total-order Sobol sensitivity indices in the VISCOUS framework. Assume a hydrologic model:

$$y = H(x_1, x_2, \ldots, x_m) \quad (SEQ\ Equation \setminus * ARABIC\ 14)$$

where a total of m input factors are evaluated in sensitivity analysis. According to Saltelli et al. (2008), variance of model response (y) is decomposed into

partial variances: first-order variance (V_i), second-order variance (V_{ij}), …, until m-order variance ($V_{\mathrm{i..m}}$).

$V(y) = \sum_{i=1}^{m} V_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} V_{\mathrm{ij}} + ... + V_{\mathrm{i..m}}$ (*SEQ Equation* * *ARABIC* 15)

where $V(y)$ is the variance of the model response y.

The first-order sensitivity index (S_i) is calculated as:

$S_i = \frac{V_i}{V(y)} = \frac{V(E(y|x_i))}{V(y)}$ (*SEQ Equation* * *ARABIC* 16)

where V_i is the first-order variance which represents the contribution to the variance of response y due to an individual input factor x_i. The first-order sensitivity index is also called the main effect sensitivity index.

The total-order variance (S_{Ti}) is calculated as:

$S_{\mathrm{Ti}} = 1 - S_{\sim i} = 1 - \frac{V(E(y|\mathbf{x}_{\sim i}))}{V(y)}$ (*SEQ Equation* * *ARABIC* 17)

where $V\left(E(y|\mathbf{x}_{\sim i})\right)$ represents the total contribution to the variance of response due to non-$\mathbf{x}_i$ (i.e., $\mathbf{x}_{\sim i}$). $\mathbf{x}_{\sim i}$ is a random vector of all input variables fixed except x_i. The total-order sensitivity index is also called the total effect sensitivity index.

From Equations (16) and (17), we see that the calculation of conditional expectations, $E(y|x_i)$ and $E\left(y|\mathbf{x}_{\sim i}\right)$, is the cornerstone of the variance-based sensitivity analysis. The following takes $E(y|x_i)$ as an example and explains how it is computed. The computation of $E\left(y|\mathbf{x}_{\sim i}\right)$ follows the same logics except replacing the scalar x_i with the vector $\mathbf{x}_{\sim i}$.

If the input factor x_i is fixed to a generic value $\tilde{x}_i$, the resulting conditional expectation of y is:

$E\left(y|x_i = \tilde{x}_i\right) = \int_{\Omega Y} y \bullet f_{Y|X}\left(y|x_i = \tilde{x}_i\right) \mathrm{dy}$ (*SEQ Equation* * *ARABIC* 18)

In GMCM, the conditional PDF, $f_{Y|X}\left(y|\mathbf{x}\right)$, is approximated using GMMs and is calculated by dividing Equation (12) by Equation (13):

$\frac{f_{X,Y}(\mathbf{x},y)}{f_X(\mathbf{x})} \approx \frac{f_{\mathrm{GMM}}\left(z_{x_1},...,z_{x_m},z_y\right)}{f_{\mathrm{GMM}}\left(z_{x_1},...,z_{x_m}\right)} \bullet \frac{1}{\phi(\Phi^{-1}(u_y))} \bullet f_Y(y)$ (*SEQ Equation* * *ARABIC* 19)

Therefore, Equation (18) is approximated by:

$E\left(y|x_i = \tilde{x}_i\right) \approx \int_{\Omega Y} y \bullet \frac{f_{\mathrm{GMM}}\left(z_{x_1},...,z_{x_m},z_y\right)}{f_{\mathrm{GMM}}\left(z_{x_1},...,z_{x_m}\right)} \bullet \frac{1}{\phi(\Phi^{-1}(u_y))} \bullet f(y)\mathrm{dy}$ (*SEQ Equation* * *ARABIC* 20)

Since $f_Y(y)dy = \ F_Y(y) = du_y$, Equation (20) becomes:

$E\left(y|x_i = \tilde{x}_i\right) \approx \int_0^1 y \bullet \frac{f_{\mathrm{GMM}}\left(z_{x_1},...,z_{x_m},z_y\right)}{f_{\mathrm{GMM}}\left(z_{x_1},...,z_{x_m}\right)} \bullet \frac{1}{\phi((\Phi^{-1}(u_y)))} du_y$ (*SEQ Equation* * *ARABIC* 21)

To drop the dependence upon the specific value $\tilde{x}_i$, the variance of $E\left(y|x_i\right)$ is estimated by integrating $E\left(y|x_i=\tilde{x}_i\right)$ over the probability density function of $\tilde{x}_i$, expressed as:

$V\left(E\left(y|x_i\right)\right)=\int_{\Omega x_i}E^2\left(y|x_i=\tilde{x}_i\right)d\tilde{x}_i-[\int_{\Omega x_i}E\left(y|x_i=\tilde{x}_i\right)d\tilde{x}_i]^2$ (*SEQ Equation* * *ARABIC* 22)

From this, we see that two loops are needed in the estimation of $V\left(E\left(y|x_i\right)\right)$. The inner loop is to integrate over du_y to compute $E\left(y|x_i=\tilde{x}_i\right)$. The outer loop is to integrate over $d\tilde{x}_i$ to eliminate the dependence on the value of $\tilde{x}_i$.

Monte Carlo-based approximation of variance-based sensitivity indices

This section explains the Monte Carlo approximations of the first-order and the total-order sensitivity indices, respectively. In Monte Carlo-based approximation, the estimated GMM, $f_{\mathrm{GMM}}(\bullet)$, is used to generate samples $\left(\tilde{z}_1,\ldots,\tilde{z}_m,\tilde{z}_y\right)$. Since two loops are needed to compute $V\left(E\left(y|x_i\right)\right)$ and $V\left(E\left(y|\mathbf{x}_{\sim i}\right)\right)$, two rounds of sampling are conducted. Details are explained below.

First-order sensitivity index

Computing the first-order sensitivity index of x_i requires including the standardized data (z_{x_i},z_y) in the GMM inference. The GMM inference is done through the EM algorithm (see **Appendix A1**). With the inferred GMM $f_{\mathrm{GMM}}(\bullet)$, the first round of sampling is conducted to generate N_1 Monte Carlo samples from $f_{\mathrm{GMM}}(\bullet)$. See $\widetilde{\mathbf{Z}}_1$ in Equation (23). $\widetilde{\mathbf{Z}}_1$is used for the outer loop to eliminate the dependence on the value of $\tilde{x}_i$.

$$\widetilde{\mathbf{Z}}_1=\begin{pmatrix}\tilde{z}_{1,x_i} & \tilde{z}_{1,y}\\ \tilde{z}_{2,x_i} & \tilde{z}_{2,y}\\ \vdots & \vdots\\ \tilde{z}_{N_1,x_i} & \tilde{z}_{N_1,y}\end{pmatrix},\widetilde{\mathbf{Z}}_2=\begin{pmatrix}\tilde{z}_{r_1,x_i} & \tilde{z}'_{1,y}\\ \tilde{z}_{r_1,x_i} & \tilde{z}'_{2,y}\\ \vdots & \vdots\\ \tilde{z}_{r_1,x_i} & \tilde{z}'_{N_2,y}\end{pmatrix},\ r_1=[1,\ldots,N_1].$$

(*SEQ Equation* * *ARABIC* 23)

The second round of sampling is repeated N_1 times by looping through each sample of $\widetilde{\mathbf{Z}}_1$. Per iteration, N_2 Monte Carlo samples are firstly generated from $f_{\mathrm{GMM}}(\bullet)$, and then all values of $\mathbf{z}_{x_i}$ are replaced by a sample of $\widetilde{\mathbf{Z}}_1$. See $\widetilde{\mathbf{Z}}_2$in Equation (23). The first column of $\widetilde{\mathbf{Z}}_2$ is replaced by $\tilde{z}_{r_1,x_i}$ of $\widetilde{\mathbf{Z}}_1$. $\widetilde{\mathbf{Z}}_2$ is used for the inner loop to integrate over du_y to compute $E\left(y|x_i=\tilde{x}_i\right)$. N_1 and N_2 can be but do not have to be equal ($N_1=N_2$ in our study).

Given a fixed value $\tilde{z}_{x_i}=\tilde{z}_{r_1,x_i}$, the conditional expectation in Equation (21) is approximated by:

$$E\left(y\middle|x_i=\tilde{x}_{r_1,i}\right)\approx\frac{1}{N_2}\sum_{r_2=1}^{N_2}F_Y^{-1}\left(u_{r_2,y}\right)\bullet\frac{f_{\mathrm{GMM}}\left(\left(\tilde{z}_{r_1,x_i},\tilde{z}_{r_2,y}\right)\right)}{f_{\mathrm{GMM}}\left(\tilde{z}_{r_1,i}\right)}\bullet\frac{1}{\phi\left(\tilde{z}_{r_2,y}\right)}$$

(*SEQ Equation* * *ARABIC* 24)

where $(\tilde{z}_{r_1,x_i}, \tilde{z}_{r_2,y})$ is the r_2^{th} sample of $\widetilde{\mathbf{Z}}_2$, $r_2 = [1, \dots, N_2]$. $\tilde{z}_{r_2,y}$ denotes the standardized value of y in the standard normal distribution. With $\tilde{z}_{r_2,y}$, the PDF and CDF of $\tilde{z}_{r_2,y}$ in the standard normal distribution (i.e., $\phi(\tilde{z}_{r_2,y})$ and $\Phi(\tilde{z}_{r_2,y})$) can be computed. $F_Y^{-1}\left(u_{r_2,y}\right)$ is the inverse CDF of $u_{r_2,y}$ in the original space of y, $\tilde{y}_{r_2} = F_Y^{-1}\left(u_{r_2,y}\right)$. Similarly, $\tilde{x}_{r_2,i} = F_{X_i}^{-1}\left(u_{r_2,x_i}\right)$.

Eliminating the dependence on the value of $\tilde{x}_i$, the variance of $E\left(y|x_i\right)$ in Equation (22) is approximated by:

$$V\left(E\left(y|x_i\right)\right) \approx \frac{1}{N_1}\sum_{r_1=1}^{N_1} E^2\left(y\middle|x_i = \tilde{x}_{r_1,i}\right) - \left[\frac{1}{N_1}\sum_{r_1=1}^{N_1} E\left(y\middle|x_i = \tilde{x}_{r_1,i}\right)\right]^2$$

(*SEQ Equation* * *ARABIC* 25)

With Equations (24) and (25), and Equation (16), the first-order sensitivity index can be computed.

Total-order sensitivity index

Computing the total-order sensitivity index of x_i requires including the standardized data $\left(\mathbf{z}_{\sim x_i}, z_y\right) = (z_{x_1}, .., z_{x_{i-1}}, z_{x_{i+1}}, \dots, z_{x_m}, z_y)$ in the GMM inference. Similarly, with the inferred GMM $f_{\text{GMM}}(\bullet)$, the first round of sampling is conducted to generate N_1 Monte Carlo samples from $f_{\text{GMM}}(\bullet)$. See $\widetilde{\mathbf{Z}}_1$in Equation (26). $\widetilde{\mathbf{Z}}_1$is used for the outer loop to eliminate the dependence on the value of $\tilde{x}_i$.

$$\widetilde{\mathbf{Z}}_1 = \begin{pmatrix} \tilde{\mathbf{z}}_{1,\sim x_i} & \tilde{z}_{1,y} \\ \tilde{\mathbf{z}}_{2,\sim x_i} & \tilde{z}_{2,y} \\ \vdots & \vdots \\ \tilde{\mathbf{z}}_{N_1,\sim x_i} & \tilde{z}_{N_1,y} \end{pmatrix} = \begin{pmatrix} \tilde{z}_{1,x_1} & \tilde{z}_{1,x_{i-1}} & \tilde{z}_{1,x_{i+1}} & \tilde{z}_{1,x_m} & \tilde{z}_{1,y} \\ \tilde{z}_{2,x_1} & \dots \; \tilde{z}_{2,x_{i-1}} & \tilde{z}_{2,x_{i+1}} & \dots \; \tilde{z}_{2,x_m} & \tilde{z}_{2,y} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{z}_{N_1,x_1} & \tilde{z}_{N_1,x_{i-1}} & \tilde{z}_{N_1,x_{i+1}} & \tilde{z}_{N_1,x_m} & \tilde{z}_{N_1,y} \end{pmatrix},$$

$$\widetilde{\mathbf{Z}}_2 = \begin{pmatrix} \tilde{\mathbf{z}}_{r_1,\sim x_i} & \tilde{z}_{1,y} \\ \tilde{\mathbf{z}}_{r_1,\sim x_i} & \tilde{z}_{2,y} \\ \vdots & \vdots \\ \tilde{\mathbf{z}}_{r_1,\sim x_i} & \tilde{z}_{N_2,y} \end{pmatrix} = \begin{pmatrix} \tilde{z}_{r_1,x_1} & \tilde{z}_{r_1,x_{i-1}} & \tilde{z}_{r_1,x_{i+1}} & \tilde{z}_{r_1,x_m} & \tilde{z}'_{1,y} \\ \tilde{z}_{r_1,x_1} & \dots \; \tilde{z}_{r_1,x_{i-1}} & \tilde{z}_{r_1,x_{i+1}} & \dots \; \tilde{z}_{r_1,x_m} & \tilde{z}'_{2,y} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{z}_{r_1,x_1} & \tilde{z}_{r_1,x_{i-1}} & \tilde{z}_{r_1,x_{i+1}} & \tilde{z}_{r_1,x_m} & \tilde{z}'_{N_2,y} \end{pmatrix},$$

$r_1 = [1, \dots, N_1]$ (*SEQ Equation* * *ARABIC* 26)

The second round of sampling is repeated N_1 times by looping through each sample of $\widetilde{\mathbf{Z}}_1$. Per iteration, N_2 Monte Carlo samples are firstly generated from $f_{\text{GMM}}(\bullet)$, and then all values of $\mathbf{z}_{\sim x_i}$ are replaced by a sample of $\widetilde{\mathbf{Z}}_1$. See $\widetilde{\mathbf{Z}}_2$in Equation (26). $\widetilde{\mathbf{Z}}_2$ is used for the inner loop to integrate over du_y to compute $E\left(y\middle|x_i = \tilde{\mathbf{x}}_{r_1,\sim i}\right)$.

Similar with Equations (24) and (25), $E\left(y\middle|x_i = \tilde{\mathbf{x}}_{r_1,\sim i}\right)$ can be computed by:

$E\left(y\middle|\mathbf{x}_{\sim i}=\tilde{\mathbf{x}}_{r_1,\sim i}\right) \approx \frac{1}{N_2}\sum_{r_2=1}^{N_2} F_y^{-1}\left(u_{r_2,y}\right) \bullet \frac{f_{\mathrm{GMM}}\left((\tilde{\mathbf{z}}_{r_1,\sim x_i},\tilde{z}_{r_2,y})\right)}{f_{\mathrm{GMM}}(\tilde{\mathbf{z}}_{r_1,\sim x_i})} \bullet \frac{1}{\phi(\tilde{z}_{r_2,y})}$
(*SEQ Equation* * *ARABIC* 27)

$V\left(E(y|\mathbf{x}_{\sim i})\right)$ can be approximated by:

$V\left(E\left(y|\mathbf{x}_{\sim i}\right)\right) \approx \frac{1}{N_1}\sum_{r_1=1}^{N_1} E^2\left(y\middle|\mathbf{x}_{\sim i}=\tilde{\mathbf{x}}_{r_1,\sim i}\right) - \left[\frac{1}{N_1}\sum_{r_1=1}^{N_1} E\left(y\middle|\mathbf{x}_{\sim i}=\tilde{\mathbf{x}}_{r_1,\sim i}\right)\right]^2$
(*SEQ Equation* * *ARABIC* 28)

With Equations (27) and (28), and Equation (17), the total-order sensitivity index can be computed.

Didactic examples

This section uses a two-parameter Rosenbrock function to demonstrate the implementation of the VISCOUS framework step-by-step. This example is intended to help users to understand the details of the GMCM probability model, such as the Gaussian components and GMM, and enable users to utilize the VISCOUS framework for their own applications.

Implementation steps

Six steps are involved in the VISCOUS framework according to Sheikholeslami et al. (2021). Taking the first-order sensitivity index of variable x_i as an example, the steps are:

Step 1. Select the to-be-evaluated input and output data based on the goal of sensitivity analysis, i.e., input-output sample data (x_i, y).

Step 2. Calculate the CDF of x_i and y, respectively, using the kernel density estimate to obtain the CDF data (u_{x_i}, u_y).

Step 3. Calculate the inverse CDF of (u_{x_i}, u_y) in the standard normal distribution to obtain the standardized data (z_{x_i}, z_y).

Step 4. Estimate the GMM based on the standardized data (z_{x_i}, z_y) via the EM algorithm in **Appendix A1**.

Step 5. Generate Monte Carlo samples based on the estimated GMM.

Step 6. Calculate the variance-based first-order sensitivity index based on Equations (24), (25), and (16), and the total-order sensitivity index based on Equations (27), (28), and (17).

Demonstration using the Rosenbrock function

The Rosenbrock function, also referred to as the Valley or Banana function, is a popular test problem for uncertainty analysis, sensitivity analysis, and optimization algorithms (Rosenbrock, 1960). In the two-dimensional form, the Rosenbrock function is defined as:

$y = 100(x_2 - x_1^2)^2 + (1 - x_1)^2, \quad x_1, x_2 \in [-2, 2]$ (*SEQ Equation* \ ∗ *ARABIC* 29)

where (x_1, x_2) are the two input variables uniformly distributed in $[-2, 2]$. The global minimum is at $(x_1, x_2) = (1, 1)$, where $y = 0$.

The distribution of the Rosenbrock function is shown in Figure 1. It involves a long steep valley and a gradually sloping floor. The Rosenbrock function in its two-dimensional form enables us to visualize the function itself and the implementation steps of VISCOUS.

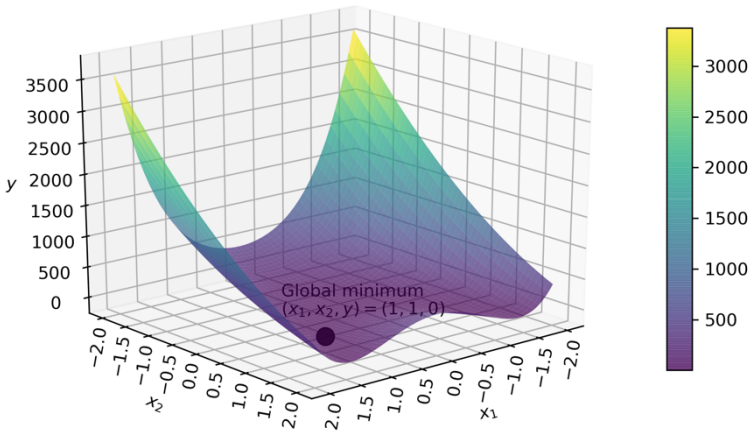

Figure . Distribution of the Rosenbrock function

Determine input-output data and calculate CDFs (steps 1-3)

In step 1, when we compute the first-order sensitivity index of x_1 for the Rosenbrock function, two variables (x_1, y) are included in the VISCOUS framework. Assume the sample size is 10,000. Following steps 1-3 in Section 3.1, we get three sets of data: input-output sample data (x_1, y), CDF data (u_{x_1}, u_y), and standardized data (z_{x_1}, z_y). Figure 2 shows the distribution of each variable among the three data sets.

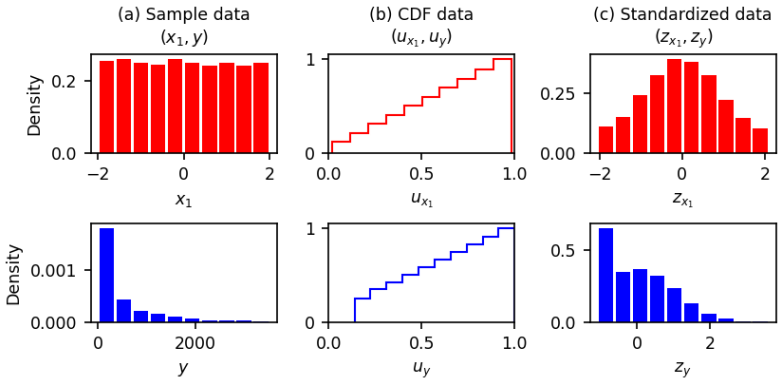

Figure . Histograms of sample data, CDF data, and standardized data.

Estimate the GMM (Step 4)

In step 4, for ease of visualization, we first used two Gaussian components to estimate the GMM ($K = 2$). Two components are unlikely to be sufficient to depict the joint PDF of (z_{x_1}, z_y) in the Rosenbrock problem, but the resulting visualization can help to understand what the Gaussian components are and how they are grouped together to form the GMM that represents the joint PDF of the evaluated input and out variables.

Based on two Gaussian components, the GMM is expressed as:

$f\left(z_{x_1}, z_y\right) = \lambda_1 \phi\left(z_{x_1}, z_y \middle|\ _1,\ _1\right) + \lambda_2 \phi\left(z_{x_1}, z_y \middle|\ _2,\ _2\right)$ (*SEQ Equation \ * ARABIC* 30)

where ϕ is the PDF of a bivariate Gaussian distribution with mean $_k$ and covariance $_k (k = 1, 2)$. Note the GMM inference is based on the standardized data (z_{x_1}, z_y). Figure 3 shows the contour of each Gaussian component and the inferred GMM. The weighted sum of the two bivariate Gaussian distributions (components) makes up the joint PDF of the standardized data (z_{x_1}, z_y). The two components are well separated and of different weights, and the mixture contour closely resemble the component contours.

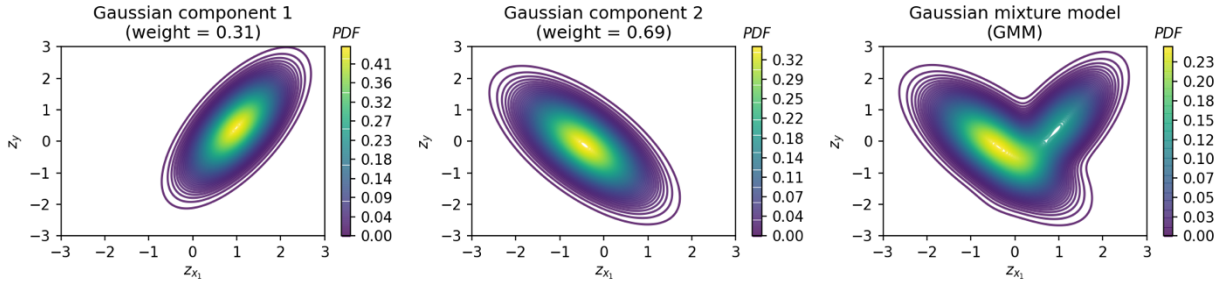

Figure . PDFs of two bivariate Gaussian components and the GMM.

The inferred parameter values of the two components are also provided:

$$\lambda_1 = 0.31, \lambda_2 = 0.69$$

$$_1 = \left[\mu_{1,z_{x_1}}, \mu_{1,z_y}\right] = [0.99,\ 0.43], \quad _2 = \left[\mu_{2,z_{x_1}}, \mu_{2,z_y}\right] = [-0.44,\ -0.12]$$

$$_1 = \begin{bmatrix} \sigma^2_{1,z_1} & \text{cov}_{1,z_{x_1}z_y} \\ \text{cov}_{1,z_y z_{x_1}} & \sigma^2_{1,z_y} \end{bmatrix} = \begin{bmatrix} 0.32 & 0.32 \\ 0.32 & 0.73 \end{bmatrix}, \quad _2 = \begin{bmatrix} \sigma^2_{2,z_{x_1}} & \text{cov}_{2,z_{x_1}z_y} \\ \text{cov}_{2,z_y z_{x_1}} & \sigma^2_{2,z_y} \end{bmatrix} = \begin{bmatrix} 0.52 & -0.40 \\ -0.40 & 0.71 \end{bmatrix}$$

In Figure 4, the inferred GMM are investigated in depth. Figure 4(a) shows that when using two Gaussian components, the standardized data (z_{x_1}, z_y) are classified into two clusters. Each Gaussian component represents a cluster. Each Gaussian component is a probabilistic density function, so it identifies the probability that each data belongs to each cluster. In Figure 4(b), the joint PDF of each data point is estimated as the weighted average PDF in the two Gaussian components, which is the GMM. Figure 4(c) shows the 2-dimension histogram or distribution frequency of the standardized data (z_{x_1}, z_y).

When comparing with Figure 4(c), it is found that the 2-component GMM in Figure 4(b) does a very poor job in representing the distribution of (z_{x_1}, z_y). Specifically, the estimated GMM has high probability densities in the lower left part of the (z_{x_1}, z_y) space, while the input samples cluster at the bottom of the z_y axis ($z_y = -1$). Due to this inaccurate representation, we cannot use this estimated GMM for the subsequent Monte Carlo sampling and the sensitivity index estimation.

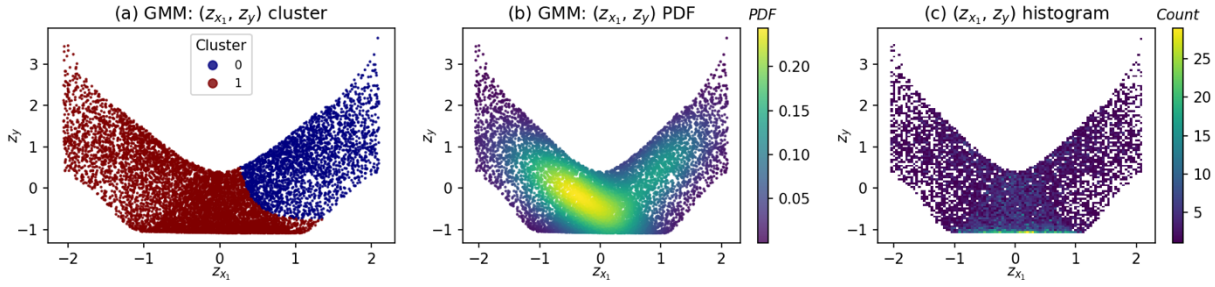

Figure . When using 2 Gaussian components, the GMM clustering (panel a) and joint PDF (panel b) results for (z_{x_1}, z_y), in comparison with the histogram of samples (z_{x_1}, z_y) (panel c).

To get a better GMM estimate, we then used the BIC criterion and selected an optimal Gaussian component number 18. Like Figure 4, Figure 5 shows the

clustering and joint PDF results using the 18-component GMM. When using 18 Gaussian components, the standardized data (z_{x_1}, z_y) are classified into 18 clusters. In Figure 5(b), the estimated probability density (GMM) matches the distribution frequency of (z_{x_1}, z_y) in Figure 5(c). Therefore, the 18-component based GMM better represents the distribution of (z_{x_1}, z_y) than the 2-component based GMM in Figure 4(b). This result highlights the impacts of the number of Gaussian components on GMM performance.

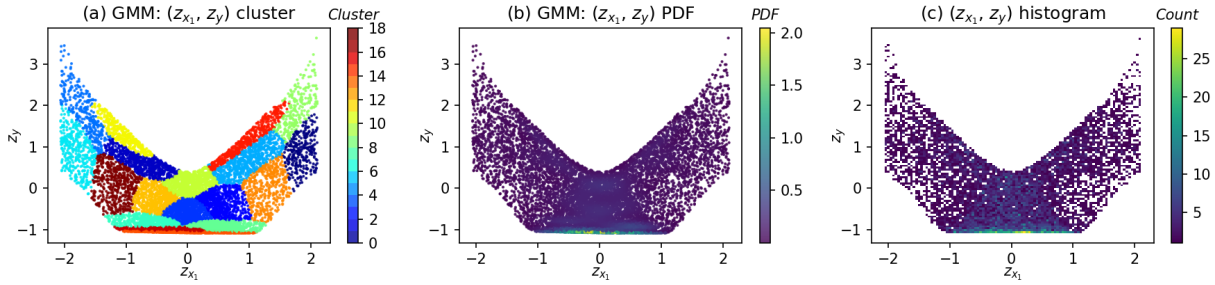

Figure . When using 18 Gaussian components, the GMM clustering (panel a) and joint PDF (panel b) results for (z_{x_1}, z_y), in comparison with the histogram of (z_{x_1}, z_y) (panel c). Note panel c is the same as Figure 4(c).

Monte Carlo sampling and sensitivity estimation (steps 5, 6)

In step 5, we set $N_1 = N_2 = 1000$ and generated Monte Carlo samples $(\tilde{z}_{x_1}, \tilde{z}_y)$ based on the inferred GMM. In step 6, we calculated the first-order sensitivity based on Equations (24), (25), and (16), and calculated the total-order sensitivity using Equations (27), (28), and (17).

To quantify the sampling uncertainty in VISCOUS, we repeated the entire processes 50 times to obtain 50 sets of sensitivity index results. Each experiment uses a different set of input-output sample data with size 10,000; and in the sensitivity index calculation, the Monte Carlo sample sizes are $N_1 = N_2 = 10,000$. For comparison, the same 50 sets of sample data were applied to the Sobol method, getting 50 sets of sensitivity index results. Figure 6 compares the results of the VISCOUS framework and the Sobol method. For both the first-order and total-order sensitivity indices, VISCOUS produces similar median sensitivity indices as Sobol but has a larger sampling uncertainty.

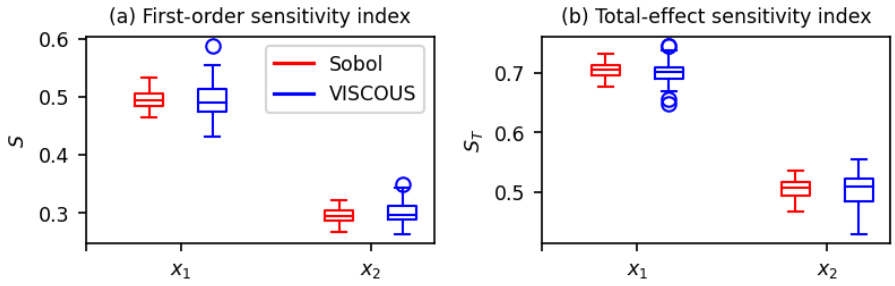

Figure . First-order and total-order sensitivity indices of the Sobol and VISCOUS methods.

Evaluation of the VISCOUS Framework

This section evaluates the VISCOUS framework using three types of Sobol functions and provides a cautionary note on the non-identifiability issue using VISCOUS to approximate sensitivity indices for applications where different input factors have similar importance.

Sobol function

According to Kucherenko *et al.* (2011), model functions can be classified into three types based on their dependence on variables.

- Type A function: Variables are not equally important in terms of sensitivity.
- Type B function: Variables are equally important, and no interaction exists between variables. Therefore, $S_i = S_{\mathrm{Ti}}, \sum S_i = 1$, and $S_i = 1/n$.
- Type C function: Variables are equally important, and interaction exists between variables. Therefore, $S_i < S_{\mathrm{Ti}}$, and $\sum S_i < 1$.

Type A functions are the most common type of functions in practice. For instance, in most water system models, a large proportion of model output variation is often associated with a small proportion of the input factors (Markstrom et al., 2016). In statistics, this is known as the sparsity of factors principle or the Pareto principle (Box and Meyer, 1986). The sparsity of factors principle states that a small subset of factors is often responsible for most of the system output uncertainty.

Type B and C functions have equally important variables. Equal importance means that all variables have the same sensitivity at all orders (i.e., first-order, second-order, …, and total-order). Type B and C functions differ in the interactions between variables. Both types of functions are rare in practice. Examining

the performance of the VISCOUS framework in them enables a more comprehensive evaluation of the strengths and weakness of VISCOUS.

The popular Sobol function is adopted to examine the performance of VISCOUS in all three cases (Kucherenko et al., 2011):

$f(X) = \prod_{i=1}^{n} \frac{|4x_i - 2| + a_i}{1 + a_i}$ (*SEQ Equation* * *ARABIC* 31)

Set $n = 10$, then $(x_1, ..., x_{10})$ are the ten input variables uniformly distributed in $[0, 1]$.We can conveniently get all the three types of function by changing a_i (Table 1) (Kucherenko et al., 2011). In the Type A function, $x_1$and x_2 are equally important, $x_3 ... x_{10}$ are equally important, and $x_1$and x_2 are not the same as sensitive as $x_3 ... x_{10}$. In the Type B and C functions, all ten x variables are equally important.

Table . Configurations of three types of Sobol' function.

Type A	$a_1 = a_2 = 0,\ a_3 = ... = a_n = 6.52$
Type B	$a_i = 6.52$
Type C	$a_i = 0$

Sensitivity indices

Figure 7 shows the first-order and total-order sensitivity indices using the VISCOUS and Sobol methods. As in Figure 6, the Sobol sensitivity indices are the benchmark; and the calculation of each sensitivity index is repeated 50 times to quantify sampling uncertainty. Each experiment uses a different set of input-output sample data with size 100,000; and the Monte Carlo sample sizes are $N_1 = N_2 = 10,000$.

Figure 7(a,c,e) show the first-order sensitivity indices of three functions. VISCOUS produces similar sensitivity results with Sobol in all three functions. Figure 7(b,d,f) show the total-order sensitivity indices of three functions. The sensitivity ranking of different variables is consistent between VISCOUS and Sobol in all three functions. Specifically, in the Type A function, x_1 and x_2 are more sensitive than $x_3, ... x_{10}$.Moreover, x_1 and x_2 are equally important, and $x_3, ... x_{10}$ are equally important. In the Type B and C functions, all x variables are equally important.

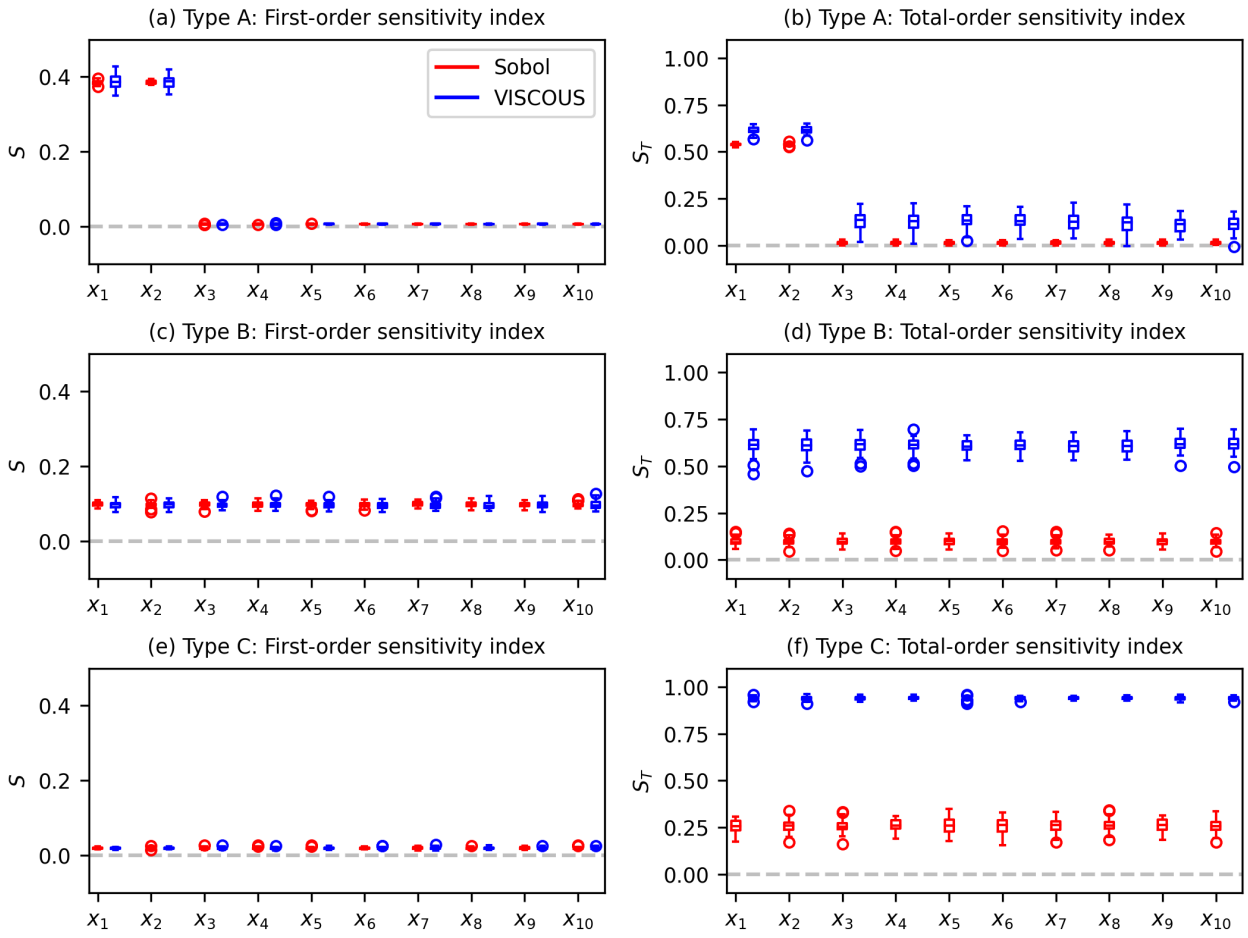

Figure . First-order and total-order sensitivity indices of VISCOUS and Sobol methods for the Type A, B, and C functions.

However, with respect to the values of total-order sensitivity indices, the performance of VISCOUS varies by function type. In the Type A function, VISCOUS results are on average 10% higher than Sobol's. In the Type B and C functions, VISCOUS results are quite different from Sobol's and are unacceptably wrong. For instance, in the Type B function, according to its definition, $S_{\mathrm{Ti}} = S_i = 1/n$ (see Section 4.1). Here $n = 10$, so the true total-order sensitivity indices of the Type B function are $S_{T1} = ... = S_{T10} = 0.1$. However, the VISCOUS results are around 0.6 in Figure 7(d).

Taking Sobol's sensitivity indices as the benchmark, we conclude that, when the number of variables is 10 (n=10), VISCOUS provides a useful approximation of total-order sensitivity index estimates for the Type A function. However, VISCOUS provides incorrect total-order sensitivity index estimates for the Type

B and C functions. The following section tests whether the above conclusion holds with a different number of variables.

Dimensionality effects on total-order sensitivity indices

To investigate the effects of the dimensionality (the number of function variables) on total-order sensitivity indices, we changed the number of function variables to be 4, 6, and 8 and applied the VISCOUS framework to the corresponding functions in all three types. Same as the above, the sensitivity indices of Sobol are the benchmark; the calculation of each sensitivity index is repeated 50 times to quantify the sampling uncertainty. Each experiment uses a different set of input-output sample data with size 100,000, and the Monte Carlo sample sizes are $N_1 = N_2 = 10,000$. Experiments use the same k-means method to generate priors. Results are shown in Figure 8.

For the type A function (the first row of Figure 8), as the number of variables increases from 4 to 8, VISCOUS always produces acceptable total-order sensitivity indices in comparison with Sobol. However, for the Type B and C functions (the second and third rows of Figure 8), as the number of variables increases from 4 to 8, VISCOUS produces progressively worse total-order sensitivity indices in comparison with Sobol. The comparison shows that the performance of the VISCOUS framework in the Type A function is stable, and it can provide acceptable total-order sensitivity estimates. The performance of the VISCOUS framework in the Type B and C functions is not stable, and the total-order sensitivity estimates get worse as the number of variables increases. We will explore why this happens to the Type B and C functions in the next section.

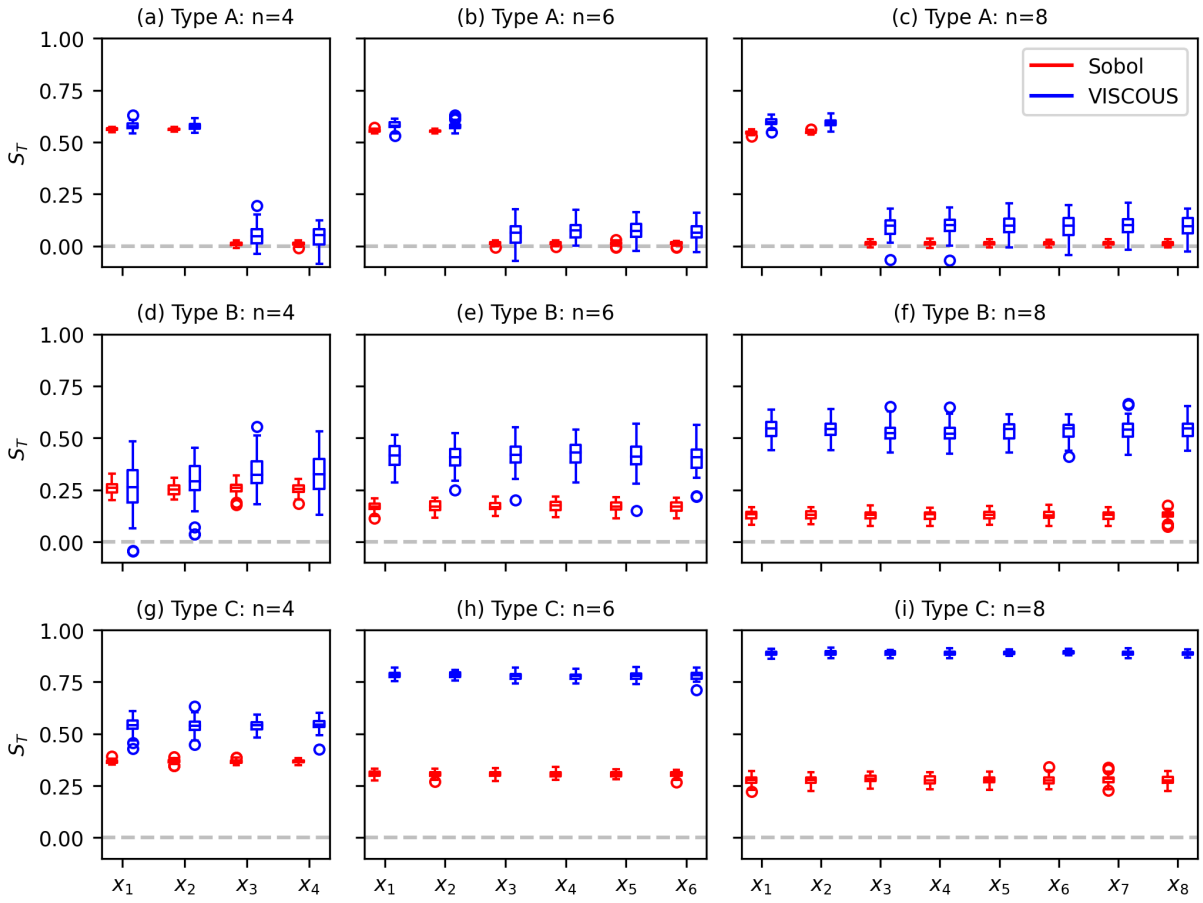

Figure . Total-order sensitivity indices of VISCOUS and Sobol methods for the Type A, B, and C functions when the number of variables is 4,6, and 8, respectively. "n" denotes the number of variables of the corresponding Sobol function.

Non-identifiability in GMM inference

We hypothesize that the inaccuracy of VISCOUS' estimation of the total-order sensitivity index in the Type B and C functions stems from the non-identifiability of the GMM inference. This section first illustrates the non-identifiability phenomenon, then explores the mathematical reason behind it, and provides solutions for obtaining well-posed inferences in the presence of non-identifiability.

Non-identifiability phenomenon in GMM inference

Taking the Type B function as an example, we calculated the total-order sensitivity index of x_1 using VISCOUS. Ten variables $(z_{x_2}, \ldots, z_{x_{10}}, z_y)$ were included in the GMM inference. The input-output sample data size is 100,000. Figure 9 and Figure 10 show the prior and posterior distributions of three GMM components, respectively. This component number is selected among multiple candidates by comparing their BICs (see Section 2.1.2). For readability, only the $z_{x_2}, z_{x_3}, z_{x_4}, z_y$ dimensions are shown in the two figures. Figure 11 details the posterior parameter estimates of the GMM components on all ten dimensions.

In Figure 9, the prior distributions of all three components are almost the same on the $z_{x_2}, z_{x_3}, z_{x_4}$ dimensions (the first three columns of Figure 9); the 1^{st} and 3^{rd} components are also hard to separate on z_y dimension (the last column of Figure 9). Therefore, the prior information is highly exchangeable between the three components.

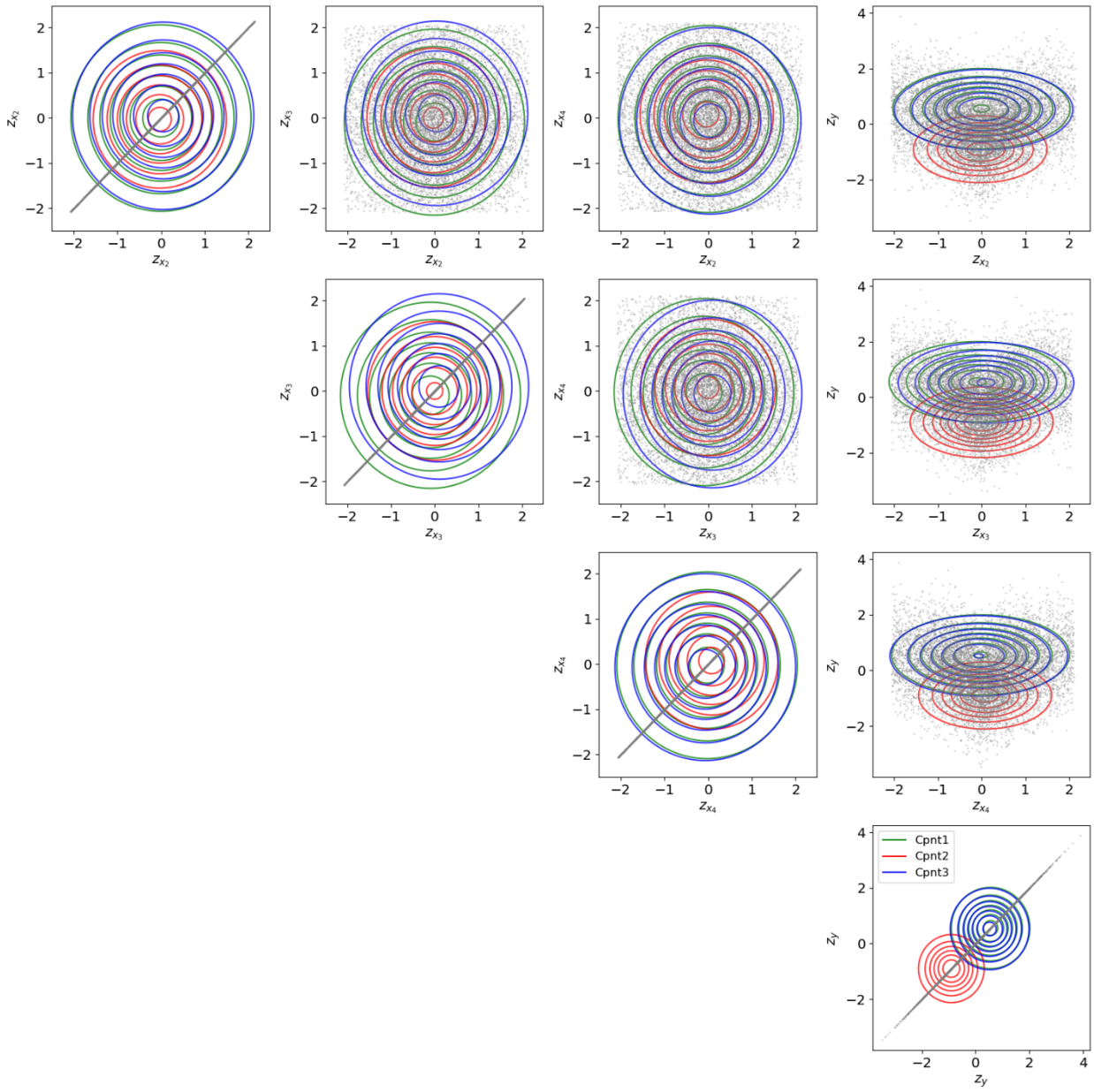

Figure . Prior distributions of three Gaussian components on selected dimensions of the Type B function. The scattered grey dots are the standardized data samples. "Cpnt" is short for component. The prior component weights are $^{\text{prior}} = [0.32,\ 0.37,\ 0.31]$. The elements below the diagonal are symmetrically equal to the elements above the diagonal and are not drawn for simplicity.

After the GMM inference, we get the posterior distributions of the three components (Figure 10). The three components become distinct on the z_{x_4} dimension (the third column of Figure 10). However, the three components are still homogeneous on z_{x_2} and z_{x_3}dimensions, and the 1st and 3rd components are homogenous on z_y dimension. Figure 11 details the similarity of the posterior distributions of the three components over eight of ten dimensions (i.e., $z_{x_2}, z_{x_3}, z_{x_5}, ..., z_{x_{10}}, z_y$). Therefore, the GMM inference result are not skillful as they do not gain much from its inference data in comparison with the prior distributions. The similar phenomena are also found in the Type C function (reported in **Appendix A2**).

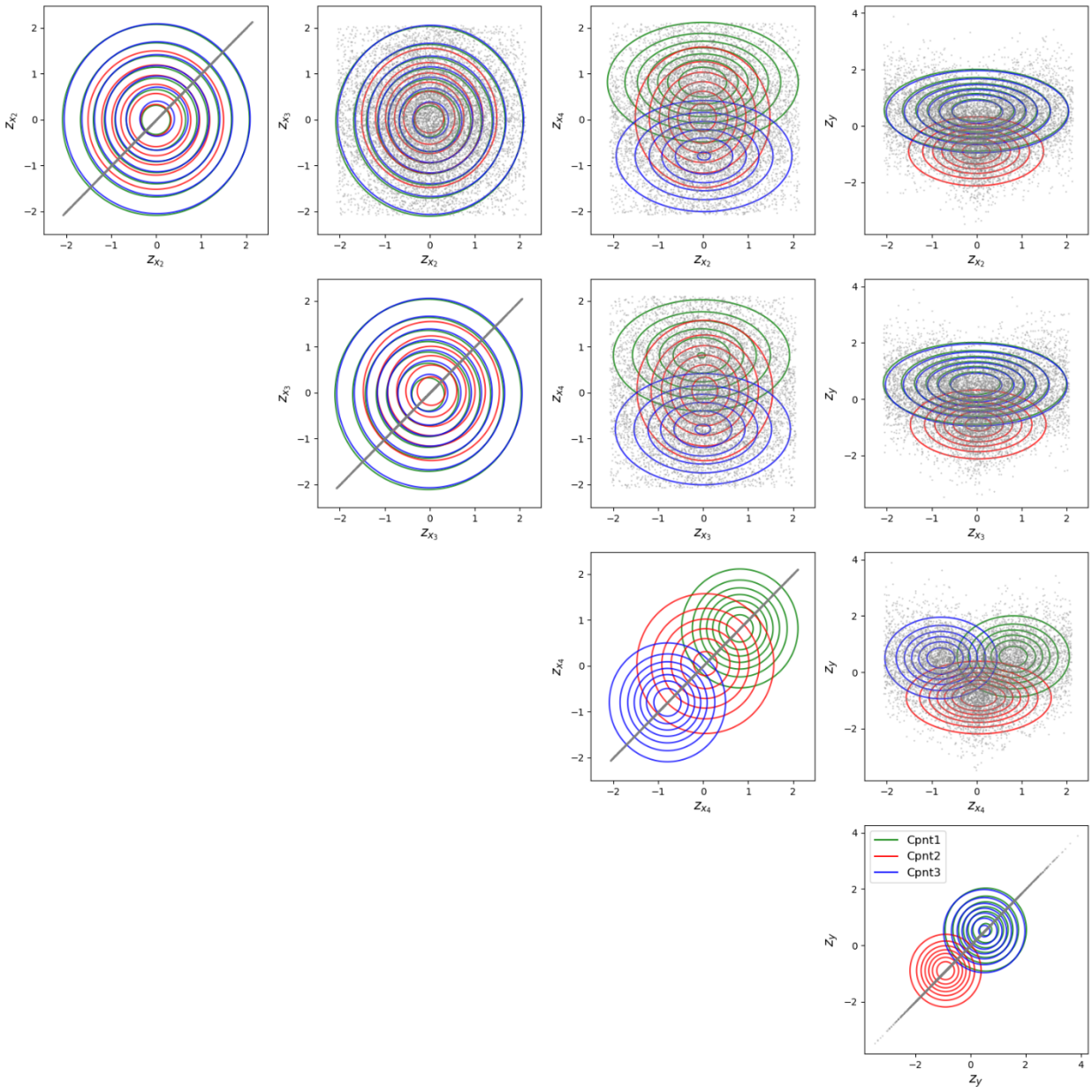

Figure . Posterior distributions of three Gaussian components on selected dimensions of the Type B function. The posterior component weights are $^{\text{posterior}} = [0.30,\ 0.37,\ 0.33]$. Notations are the same as in Figure 8.

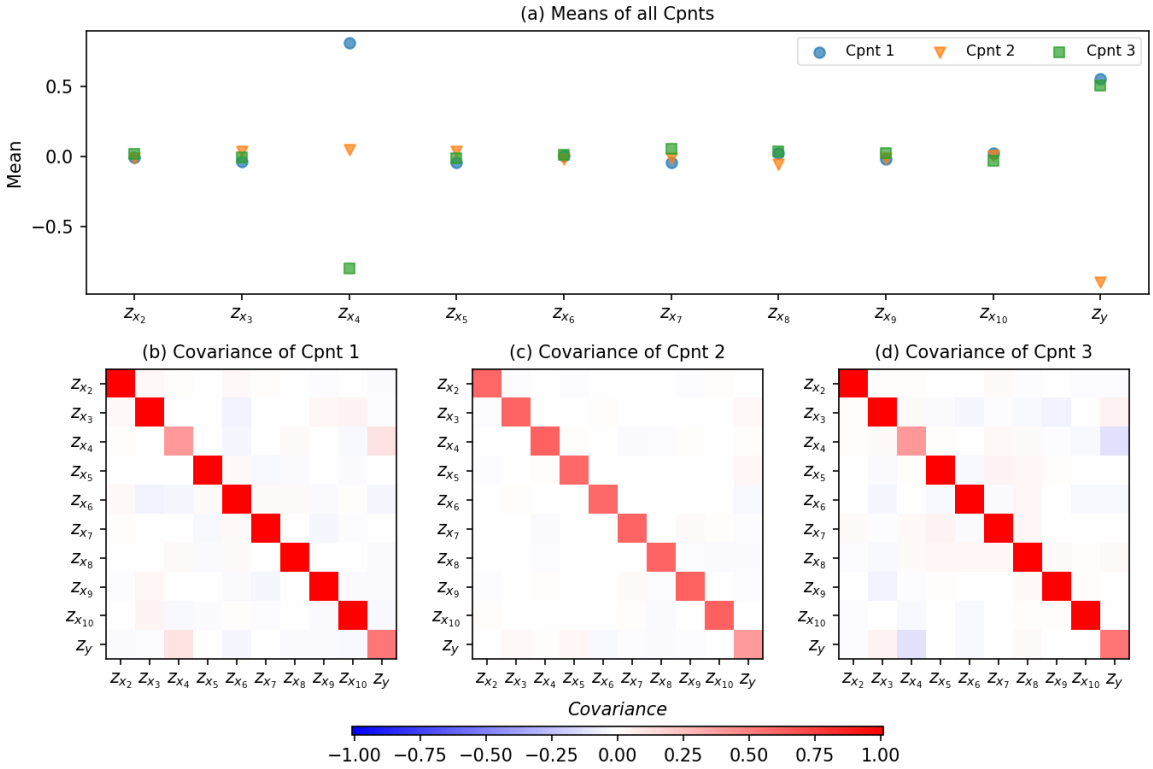

Figure . Posterior mean and covariance estimates of three Gaussian components for the Type B function. "Cpnt" is short for component.

Grouped component parameters in GMM inference

The phenomenon shown above is called non-identifiability. Non-identifiability is the inability to infer some or all parameters of interest from the available data (Renard et al., 2010; Wagener et al., 2001). This section explains the reason behind the non-identifiability of GMM and therefore VISCOUS.

In GMM, the likelihood of all samples is expressed by:

$$P(Z|\ ) = \prod_{n=1}^{N} \sum_{k=1}^{K} \lambda_k \mathcal{N}\left(\mathbf{z}_n|\ _k,\ _k\right) \quad (32)$$

where $\ = [\ ,\ ,\ ]$ is the parameter vector, N is the total number of input samples. $\mathbf{z}_n = [\mathbf{z}_{n,x}, z_{n,y}]$ is the n$^{\text{th}}$ standardized input sample.

Consider a simple example of the non-identifiability of GMM with two components. The likelihood is:

$$P(Z|\ ) = \prod_{n=1}^{N} \left[\lambda_1 \mathcal{N}\left(\mathbf{z}_n|\ _1,\ _1\right) + \lambda_2 \mathcal{N}\left(\mathbf{z}_n|\ _2,\ _2\right)\right] \quad (33)$$

Assuming the two Gaussian components are independent, the likelihood function can be re-parameterized as:

$P(Z|\ ) = \prod_{n=1}^{N} \mathcal{N}\left(\mathbf{z}_n|\ ,\ \right)$ (34)

where,

$_{\text{EM}}$=$\lambda_1\ _1 + \lambda_2\ _2$ (35)

$_{\text{EM}}$=$\lambda_1^2\ _1 + \lambda_2^2\ _2$ (36)

The re-parameterized likelihood function depends on the weighted sum of $_1$ and $_2$., not on the individual $_1$ and $_2$. Therefore, $_{\text{EM}}$ and $_{\text{EM}}$ are identifiable and its inference are well posed, but the individual $_1$ and $_2$ are not identifiable.

However, VISCOUS needs well-posed inference on both grouped parameters ($_{\text{EM}}$, $_{\text{EM}}$) and individual component parameters $_k$. This is because to compute the conditional expectations in variance-based sensitivity indices, both the joint and the marginal distributions of the GMM are needed (see Equations (24) and (27)). The next section will provide a solution for getting well-posed inference on individual component parameters.

Non-exchangeable priors in GMM inference

The strength of the prior information determines whether the inference on individual component parameters is well-posed or ill-posed in the presence of non-identifiability (Renard et al., 2010). According to Renard et al. (2010), an inference result is well-posed if the associated posterior has the following properties: (1) it integrates to unity; (2) it is informative, and (3) it depends on reasonably continuously on the inference data.

Inference using the non-exchangeable and precise priors yields well-posed individual component parameters. Non-exchangeable priors mean that the priors for one component are distinctly different from the priors for all other components:

$\mu_k \neq \mu_{k'}.\ Or,\ \left[\mu_{k,x}\ , \mu_{k,y}\right] \neq \left[\mu_{k',x}\ , \mu_{k',y}\right]$ (37)

$\Sigma_k \neq \Sigma_{k'}.\ Or,\ \left[\Sigma_{k,xx}\ \Sigma_{k,xy}\ \Sigma_{k,yx}\ \sigma_{k,y}^2\ \right] \neq \left[\Sigma_{k',xx}\ \Sigma_{k',xy}\ \Sigma_{k',yx}\ \sigma_{k',y}^2\ \right]$ (38)

where k and k' represent two different Gaussian components of the GMM ($k,\ k^{'} \in [1, .., K],\ k \neq\ k'$). Otherwise, if $\mu_k = \mu_{k'}$ and $\Sigma_k = \Sigma_{k'}$, then the two priors are exchangeable between the k^{th} and k'^{th}components (see Figure 9).

The inherent challenge in generating non-exchangeable priors exists in equally important variables. In the Type B and C functions, the equally important variables have the same distribution and same interaction with other variables (including y), the prior information on these equally important variable dimensions is very similar or even the same (see Figure 9). if the sample data induce exchangeable priors and cannot discriminate between components, then the sample data cannot discriminate between the individual component parameters. In this case, it is hard for any inference algorithm to explicitly discriminate these component parameters.

In the literature, there are two main approaches to generating non-exchangeable and precise prior information. The first solution is to create strong constraints on the prior component means and covariances. Univariate problems can follow Bartolucci (2005), multivariate problems can follow Zio et al. (2007), or use a hierarchical prior (Malsiner-Walli et al., 2017; Teh et al., 2006). The second approach is *ad hoc* and includes two steps. It first estimates multiple Gaussian components, and then merges these components according to some criteria. Example criteria include the closeness of the means (Li, 2005), the modality of the obtained mixture density, the degree of overlapping measured by misclassification probabilities, and the entropy of the resulting partition (Malsiner-Walli et al., 2017). Note this work used the k- means, not one of the above approaches to generate the priors of Gaussian component parameters. Applying the above approaches to generate the priors is out of scope of this study.

Even with the above methods of generating non-exchangeable priors, high dimensionality poses another challenge to the GMM inference. An unconstrained GMM with K components and D dimensional data involves $K \times D \times D + K \times D + K$ parameters. In detail, K covariance matrices each of size $D \times D$, K mean vectors of length D, plus a component weight vector of length K. This can be a problem for high-dimensional problems because it can quickly become impossible to find a sufficient amount of sample data to make good inferences regardless of their prior distributions. This partly explains why VISCOUS produces progressively worse total-order sensitivity indices as the number of variables in the Type B and C functions increases (see Figure 8).

Code Availability

Access to the VISCOUS source code (pyVISCOUS) and examples are available in a public repository at https://github.com/h294liu/pyviscous.git. pyVISCOUS can be installed with pip or from source. We also provide example notebooks for the Rosenbrock function and three Sobol functions (Type A, B, and C). Each example notebook contains code and documentation on how to generate input-output data, set up and run VISCOUS, and evaluate the sensitivity index results.

Conclusions

The purpose of this technical note is to make the VISCOUS (VarIance-based Sensitivity analysis using COpUlaS) framework easier to understand and apply. In this note, we make three contributions. First, provide didactic examples and additional background material to make VISCOUS easier to understand and apply for general readers. Second, we evaluate VISCOUS using three types of Sobol functions and investigate the non-identifiability property for functions where different model inputs are of similar sensitivity. Third, we provide an open-source code of VISCOUS in Python, namely, pyVISCOUS.

This contribution extends the introductory VISCOUS paper of Sheikholeslami et al. (2021) in several ways: basic knowledge about the copula function, the Gaussian Mixture Model (GMM), and the Gaussian Mixture Copula Model (GMCM); step-by-step derivations of the first- and total-order Sobol sensitivity indices using the GMCM for didactic test problems; Monte Carlo-based approximations for both the first- and total-order Sobol sensitivity indices.

Our evaluation of VISCOUS using three types of Sobol functions shows that VISCOUS provides the same sensitivity ranking as the Sobol method for both first-order and total-order sensitivity indices (see Figure 7). For first-order sensitivity indices, VISCOUS provides accurate estimates. For total-order sensitivity indices, when all variables are not equally important (Type A functions), VISCOUS provides a useful approximation of the Sobol total-order sensitivity indices. When all variables are equally important (Type B and C functions), VISCOUS does not perform well and produces progressively worse total-order sensitivity indices as the number of variables increases (see Figure 8).

Type A functions are the most common type of functions in practice, since a small subset of factors is often responsible for most of the system output uncertainty. Therefore, VISCOUS is suitable for most system models. The advantage of VISCOUS lies in that it provides useful approximations of the Sobol sensitivity indices by using any existing input (e.g., the perturbations in the model parameters) and output data (e.g., the model responses given a parameter perturbation). As a "given-data" sensitivity analysis framework, VISCOUS does not require the input data follow specific sampling strategies, it also does not need additional model runs when input-output data are available. For example, the input-output data can be from the previous model runs generated from other modeling purposes, such as uncertainty propagation and model calibration.

VISCOUS' inaccurate estimates of total-order sensitivity indices stem from the non-identifiability of the GMM inference. In the GMM inference, the individual Gaussian component parameters (e.g., mean and covariance) are grouped and thus cannot be identified. In the presence of non-identifiability, obtaining well-posed inferences on individual component parameters requires non-exchangeable and precise prior information. VISCOUS currently uses the k-means method to generate priors. The k-means method works well with Type A functions but not with Type B and C functions. Future work is needed to incorporate the method of creating non-exchangeable priors into VISCOUS, so VISCOUS can handle the functions with equally important variables (i.e., Type B and C functions).

Future work is also needed to better understand the applicability of VISCOUS for different applications. For example, further research is needed to understand the number of model evaluations that are needed for VISCOUS to provide reliable sensitivity estimates. We gladly invite discussion and collaboration with others interested in related issues of sensitivity and uncertainty analysis for computationally expensive models. We seek collaborations to assess pyVISCOUS' effectiveness in large samples of model types and study locations across a variety of hydroclimatic and environmental regimes. This will further help us test,

improve, and modify the proposed sensitivity analysis framework.

Acknowledgements

This work was completed as part of the Core Modelling theme in the Global Water Futures program. Access to the VISCOUS source code (pyVISCOUS) and examples is available in a public repository at https://github.com/h294liu/pyviscous.git.

Conflict of interest statement

The authors have no conflicts of interests to declare.

Appendix

Appendix A1. Expectation-Maximization (EM) algorithm in GMM inference

With the likelihood function in Equation (31), the average log-likelihood of all samples is:

$$\log\left(P(Z|\ )\right) = \sum_{n=1}^{N} \log\left(\sum_{k=1}^{K} \lambda_k \mathcal{N}\left(\mathbf{z}_n|\ _k,\ _k\right)\right) \text{ (A1)}$$

where $= [\ ,\ ,\ ]$ is the parameter vector. is the component weight vector. and are the mean vector and covariance matrix of a Gaussian component. N is the total number of input samples, $\mathbf{z}_n = [\mathbf{z}_{n,x}, z_{n,y}]$ is the n^{th} set of samples in the standard space.

The EM algorithm is to maximize the average log-likelihood, that is, to solve the following optimization problem:

$$\max \log\left(P\left(Z|\ \right)\right) = \max \sum_{n=1}^{N} \left[\log \sum_{k=1}^{K} \lambda_k \mathcal{N}\left(\mathbf{z}_n|\ _k,\ _k\right)\right] \text{ (A2)}$$

According to Jensen's inequality and the evidence lower bound, the above optimization problem can be simplified as follows for ease of computation (Biarnes, 2020):

$$\max \log\left(P\left(Z|\theta\right)\right) \geq \max \sum_{n=1}^{N} \sum_{k=1}^{K} log[\lambda_k \mathcal{N}\left(\mathbf{z}_n|\ _k,\ _k\right) \text{ (A3)}$$

$$= \max \sum_{n=1}^{N} \sum_{k=1}^{K} \left[\log\left(\lambda_k\right) + \log\left(\mathcal{N}\left(\mathbf{z}_n|\ _k,\ _k\right)\right)\right] \text{ (A4)}$$

Given the above, the EM algorithm proceeds as follows (Biarnes, 2020; Bonakdarpour, 2016; Pedregosa et al., 2011):

1. Initialize the parameter vector $= [\ ,\ ,\ ]$ to a set of random values.

2. Expectation (E) step: Assign component labels to each sample using . In other words, compute the posterior probability of sample $\mathbf{z}_n$ belonging to each component. It is equal to the ratio between the component probability and the sum of all component probabilities:

$$P\left(L_n = k|\mathbf{z}_n\right) = \frac{P(\mathbf{z}_n|L_n=k)P(L_n=k)}{P(\mathbf{z}_n)} = \frac{\lambda_k \mathcal{N}(\mathbf{z}_n|\ _k,\ _k)}{\sum_{k=1}^{K} \lambda_k \mathcal{N}(\mathbf{z}_n|\ _k,\ _k)} \quad \text{(A5)}$$

where L_n denotes the component label. Moreover, compute the log-likelihood $\log\left(P(Z|\theta)\right)$ based on Equation (A1).

1. Maximization (M) step: Update the parameters θ using the just computed posterior probability $P\left(L_n = k|\mathbf{z}_n\right)$ so that the log-likelihood can be maximized. Many parameter update approaches exist in the literature, the approach used in this study is:

$$\hat{\lambda}_k = \frac{1}{N}\sum_{n=1}^{N} P\left(L_n = k|\mathbf{z}_n\right) \bullet \lambda_k \quad \text{(A6)}$$

$$\hat{}_k = \frac{1}{N}\sum_{n=1}^{N} P\left(L_n = k|\mathbf{z}_n\right) \bullet \mathbf{z}_n \quad \text{(A7)}$$

$$\hat{}_k = \sqrt{\frac{1}{N}\sum_{n=1}^{N} P\left(L_n = k|\mathbf{z}_n\right) \bullet \left(\mathbf{z}_{n_{\text{sub}}} - \hat{}_k\right)^2} \quad \text{(A8)}$$

1. Iterate steps 2 and 3 until the log-likelihood converges.

Appendix A2. GMM inference results of the Type C function

When using VISCOUS to calculate the total-order sensitivity index of x_1, ten variables $(x_2, \ldots, x_{10}, y)$ are included in the GMM inference. For the Type C function, the Gaussian mean and covariance estimates of one experiment are shown in Figure A1. In this experiment, a seven-component GMM is constructed. The component weights are = [0.13, 0.23, 0.18, 0.04, 0.09, 0.23, 0.10]. The Gaussian means and covariances are almost the same among the 2^{nd}, 3^{rd}, and 6^{th} components. Therefore, the three components are not well separated and cause the failure of the GMM inference.

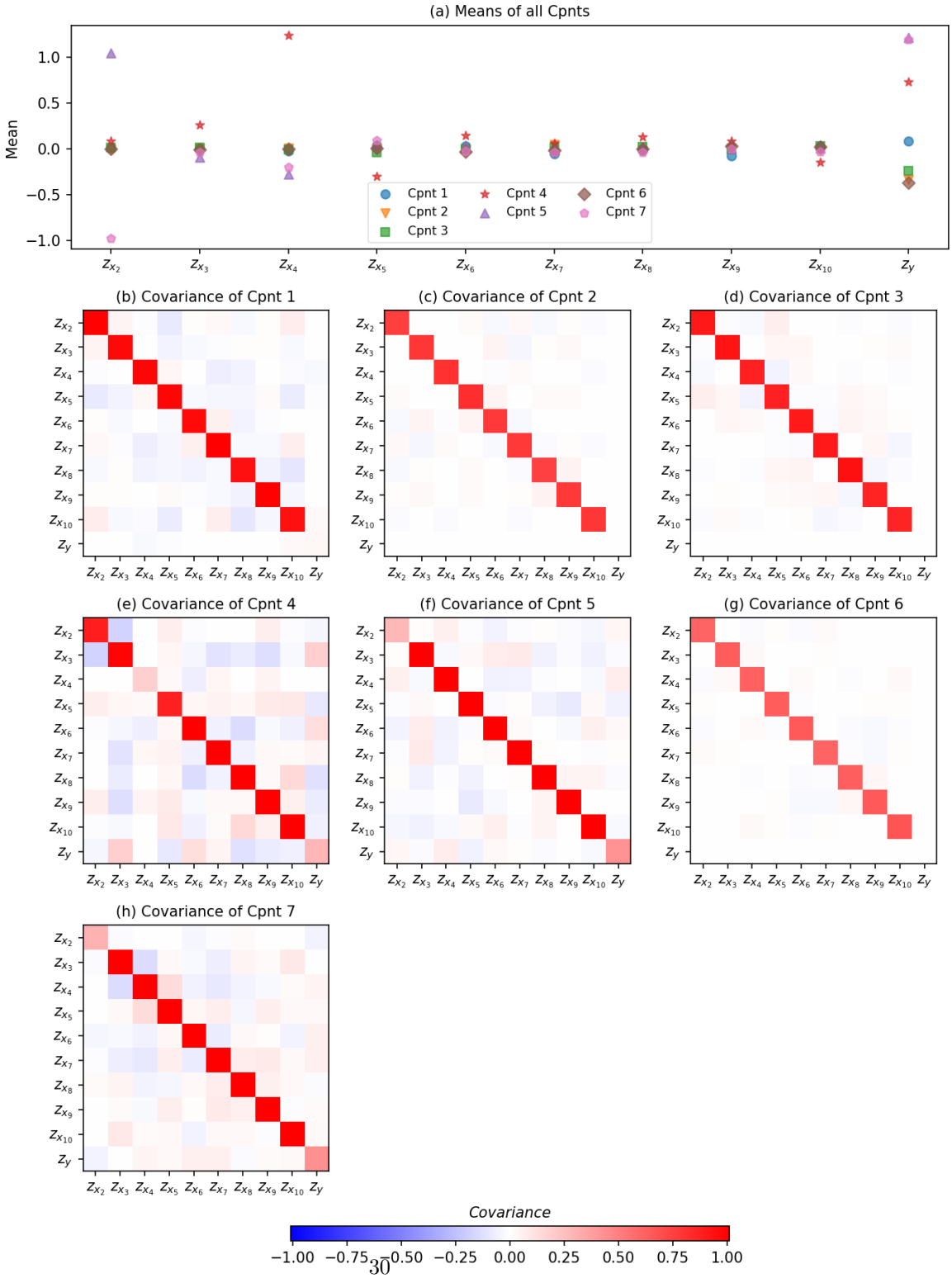
(a) Means of all Cpnts
Mean
1.0
0.5
0.0
−0.5
−1.0
Cpnt 1
Cpnt 2
Cpnt 3
Cpnt 4
Cpnt 5
Cpnt 6
Cpnt 7
z_{x_2} z_{x_3} z_{x_4} z_{x_5} z_{x_6} z_{x_7} z_{x_8} z_{x_9} $z_{x_{10}}$ z_y
(b) Covariance of Cpnt 1
(c) Covariance of Cpnt 2
(d) Covariance of Cpnt 3
(e) Covariance of Cpnt 4
(f) Covariance of Cpnt 5
(g) Covariance of Cpnt 6
(h) Covariance of Cpnt 7
Covariance
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00

Figure A1. Posterior mean and covariance estimates of seven Gaussian components for the Type C function.

References

Bartolucci, F., 2005. Clustering univariate observations via mixtures of unimodal normal mixtures. J. Classif. 22, 203–219. https://doi.org/10.1007/s00357-005-0014-7Biarnes, A., 2020. Gaussian Mixture Models and Expectation-Maximization (A full explanation) [WWW Document]. Towar. Data Sci. URL https://towardsdatascience.com/gaussian-mixture-models-and-expectation-maximization-a-full-explanation-50fa94111ddd (accessed 5.13.22).Bonakdarpour, M., 2016. Introduction to EM: Gaussian Mixture Models [WWW Document]. URL https://stephens999.github.io/fiveMinuteStats/intro_to_em.html (accessed 5.13.22).Box, G.E.P., Meyer, R.D., 1986. An analysis for unreplicated fractional factorials. Technometrics 28, 11–18. https://doi.org/10.1080/00401706.1986.10488093Campolongo, F., Cariboni, J., Saltelli, A., 2007. An effective screening design for sensitivity analysis of large models. Environ. Model. Softw. 22, 1509–1518. https://doi.org/10.1016/J.ENVSOFT.2006.10.004Demaria, E., … B.N.-J. of G., 2007, undefined, 2007. Monte Carlo sensitivity analysis of land surface parameters using the Variable Infiltration Capacity model. Wiley Online Libr. 112, 11113. https://doi.org/10.1029/2006JD007534Homma, T., Saltelli, A., 1996. Importance measures in global sensitivity analysis of nonlinear models. Reliab. Eng. Syst. Saf. 52, 1–17. https://doi.org/10.1016/0951-8320(96)00002-6Hu, Z., Mahadevan, S., 2019. Probability models for data-Driven global sensitivity analysis. Reliab. Eng. Syst. Saf. 187, 40–57. https://doi.org/10.1016/j.ress.2018.12.003Kucherenko, S., Feil, B., Shah, N., Mauntz, W., 2011. The identification of model effective dimensions using global sensitivity analysis. Reliab. Eng. Syst. Saf. 96, 440–449. https://doi.org/10.1016/j.ress.2010.11.003Li, J., 2005. Clustering based on a multilayer mixture model. J. Comput. Graph. Stat. 14, 547–568. https://doi.org/10.1198/106186005X59586Malsiner-Walli, G., Frühwirth-Schnatter, S., Grün, B., 2017. Identifying Mixtures of Mixtures Using Bayesian Estimation. J. Comput. Graph. Stat. 26, 285. https://doi.org/10.1080/10618600.2016.1200472Markstrom, S.L., Hay, L.E., Clark, M.P., 2016. Towards simplification of hydrologic modeling: identification of dominant processes. Hydrol. Earth Syst. Sci 20, 4655–4671. https://doi.org/10.5194/hess-20-4655-2016Morris, M.D., 1991. Factorial sampling plans for preliminary computational experiments. Technometrics 33, 161–174. https://doi.org/10.1080/00401706.1991.10484804Nossent, J., Elsen, P., Bauwens, W., 2011. Sobol' sensitivity analysis of a complex environmental model. Environ. Model. Softw. 26, 1515–1525. https://doi.org/10.1016/J.ENVSOFT.2011.08.010Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine Learning

in Python. J. Mach. Learn. Res. 12, 2825–2830.Pianosi, F., Beven, K., Freer, J., Hall, J.W., Rougier, J., Stephenson, D.B., Wagener, T., 2016. Sensitivity analysis of environmental models: A systematic review with practical workflow. Environ. Model. Softw. https://doi.org/10.1016/j.envsoft.2016.02.008Rakovec, O., Hill, M.C., Clark, M.P., Weerts, A.H., Teuling, A.J., Uijlenhoet, R., 2014. Distributed evaluation of local sensitivity analysis (DELSA), with application to hydrologic models. Water Resour. Res. 50, 409–426. https://doi.org/10.1002/2013WR014063Razavi, S., Gupta, H. V., 2015. What do we mean by sensitivity analysis? the need for comprehensive characterization of "global" sensitivity in Earth and Environmental systems models. Water Resour. Res. 51, 3070–3092. https://doi.org/10.1002/2014WR016527Razavi, S., Jakeman, A., Saltelli, A., Prieur, C., Iooss, B., Borgonovo, E., Plischke, E., Lo Piano, S., Iwanaga, T., Becker, W., Tarantola, S., Guillaume, J.H.A., Jakeman, J., Gupta, H., Melillo, N., Rabitti, G., Chabridon, V., Duan, Q., Sun, X., Smith, S., Sheikholeslami, R., Hosseini, N., Asadzadeh, M., Puy, A., Kucherenko, S., Maier, H.R., 2021. The Future of Sensitivity Analysis: An essential discipline for systems modeling and policy support. Environ. Model. Softw. 137, 104954. https://doi.org/10.1016/J.ENVSOFT.2020.104954Renard, B., Kavetski, D., Kuczera, G., Thyer, M., Franks, S.W., 2010. Understanding predictive uncertainty in hydrologic modeling: The challenge of identifying input and structural errors. Water Resour. Res. 46, W05521. https://doi.org/10.1029/2009WR008328Rosenbrock, H.H., 1960. An automatic method for finding the greatest or least value of a function. Comput. J. 3, 175–184. https://doi.org/10.1093/comjnl/3.3.175Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., Tarantola, S., 2008. Global Sensitivity Analysis. The Primer, Global Sensitivity Analysis. The Primer. https://doi.org/10.1002/9780470725184Sheikholeslami, R., Gharari, S., Papalexiou, S.M., Clark, M.P., 2021. VISCOUS: A Variance-Based Sensitivity Analysis Using Copulas for Efficient Identification of Dominant Hydrological Processes. Water Resour. Res. 57. https://doi.org/10.1029/2020wr028435Sheikholeslami, R., Razavi, S., Gupta, H. V., Becker, W., Haghnegahdar, A., 2019. Global sensitivity analysis for high-dimensional problems: How to objectively group factors and measure robustness and convergence while reducing computational cost. Environ. Model. Softw. 111, 282–299. https://doi.org/10.1016/J.ENVSOFT.2018.09.002Singh, A., 2019. Gaussian Mixture Models Clustering Algorithm Python [WWW Document]. URL https://www.analyticsvidhya.com/blog/2019/10/gaussian-mixture-models-clustering/ (accessed 4.17.22).Sklar, A., 1959. Fonctions de répartition à n dimensions et leurs marges. Publ. L'Institut Stat. L'Universit{é} Paris 8, 229–231.Sobol, I.M., 2001. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Math. Comput. Simul. 55, 271–280. https://doi.org/10.1016/S0378-4754(00)00270-6Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M., 2006. Hierarchical Dirichlet processes. J. Am. Stat. Assoc. 101, 1566–1581. https://doi.org/10.1198/016214506000000302Tewari, A., Giering, M.J., Raghunathan, A., 2011. Parametric characterization of multimodal distributions with

non-Gaussian modes. Proc. - IEEE Int. Conf. Data Mining, ICDM 286–292. https://doi.org/10.1109/ICDMW.2011.135van Griensven, A., Meixner, T., Grunwald, S., Bishop, T., Diluzio, M., Srinivasan, R., 2006. A global sensitivity analysis tool for the parameters of multi-variable catchment models. J. Hydrol. 324, 10–23. https://doi.org/10.1016/J.JHYDROL.2005.09.008Wagener, T., Boyle, D.P., Lees, M.J., Wheater, H.S., Gupta, H. V., Sorooshian, S., 2001. A framework for development and application of hydrological models. Hydrol. Earth Syst. Sci. 5, 13–26. https://doi.org/10.5194/hess-5-13-2001Xu, L., Jordan, M.I., 1996. On Convergence Properties of the EM Algorithm for Gaussian Mixtures. Neural Comput. 8, 129–151. https://doi.org/10.1162/neco.1996.8.1.129Zio, M. Di, Guarnera, U., analysis, R.R.-C. statistics & data, 2007, undefined, 2007. A mixture of mixture models for a classification problem: The unity measure error. Elsevier 51, 2573–2585. https://doi.org/10.1016/j.csda.2006.01.001