

# Human-AI-in-the-loop global sensitivity analysis framework for marine engineering systems

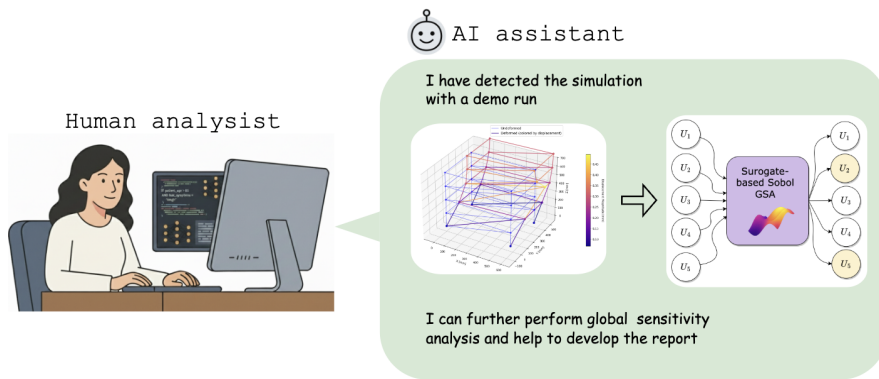
Yu Ma\*, Lin Li and Yihan Xing

Department of Mechanical and Structural Engineering and Materials Science,  
University of Stavanger, Stavanger, Norway

E-mail: yu.ma@uis.no

## Abstract.

This work introduces a feasible framework to combine the large language model (LLM) enabled AI agent with state-of-the-art surrogate-based global sensitivity analysis (GSA). GSA is a powerful tool for understanding the sensitivity of arbitrary engineering system to uncertain variables in terms of external load, system parameters and etc.. Using machine-learning-based surrogate model can ease the computational burden in brute-force Monte-Carlo tasks, and then sensitivity indices are obtained in an efficient manner. Furthermore, to automate the process of such sensitivity analysis and overcome the readiness difficulty of sensitivity results at the workspace, a large language model (LLM) engaged agentic framework is proposed in this study. The task can be launched by the user at the workspace in a human-trusted manner. It is further handled by LLM-agent to perform the surrogate GSA and interpret the sensitivity results to improve the report development. This study, thereby, sets a reference of the Human-AI-in-the-loop (H-A-I-L) framework for smart engineering solutions.



## 1 Introduction

This study targets at the integration of the Large Language Model (LLM) with the surrogate-based Global Sensitivity Analysis (GSA) analysis. GSA has become a popular approach for quantifying the sensitivity of a system's output features to changes in input parameters. It makes one feasible approach for engineers or analysts to understand the behavior of the system. Targeted at quantified sensitivity results in a statistical perspective, the Sobol's Global Sensitivity Analysis (GSA) [1] can be practiced. The parametric Monte-Carlo simulations or quasi-Monte Carlo simulations are required to be executed for the GSA [2]. The production of

GSA yields those Sobol indices which are decomposed from the total variance of the outputs [3]. If any featured output is being observed, the Sobol indices can rank the causes by their sensitivity to the output in a statistical perspective.

Furthermore, the traditional Sobol's GSA requires computationally expensive Monte-Carlo simulations. As the machine learning (ML) based surrogates are gaining traction in various fields, the GSA can be executed by the surrogates [4]. Massive Monte-Carlo tasks can be taken over by a ML-based surrogate model for efficient execution, and then, sensitivity indices are obtained in a faster way. Such surrogate-GSA approach [4–9] has gained popularity, becoming state-of-the-art (SOTA) in engineering analysis.

The above-mentioned GSA has been relying on the knowledge of a data scientist or simulation engineer to both develop adequate computer programs and interpret the results. It means that a lot of office hours are required. To leverage the productivity, it is of importance to enable automated execution and interpretation of the GSA results at workspace, and a novel solution with LLM is proposed in this study.

The LLM is witnessed as the latest success of AI technology in natural language processing, established on top of the famous “Transformer” architecture [10]. After the commercial success of OpenAI since its GPT-3 model [11], the LLMs have been extended to infrastructure services for program building [12], owing to its capability to output computer code token by token [13, 14]. The LLM's human-like processing of context and knowledge enables flexible and reliable interaction with the users [15].

Benefiting from such technology maturity in this era, it is of interest to set up an LLM-based agent to get engaged in processing the challenging GSA tasks and developing diagnostic reports. The human users may interact with the agent in a human-AI-in-the-loop (H-A-I-L) manner, to set up the new analysis tasks and leave the agent to accomplish them. This approach, then, is regarded as a viable solution for improving the quality and efficiency in the surrogate-GSA solution for engineering analysis.

## 2 Surrogate modeling for GSA

The surrogate construction and subsequent Sobol GSA analysis are the SOTA method in the latest research of engineering analysis [4–9]. The workflow is depicted in Figure 1.

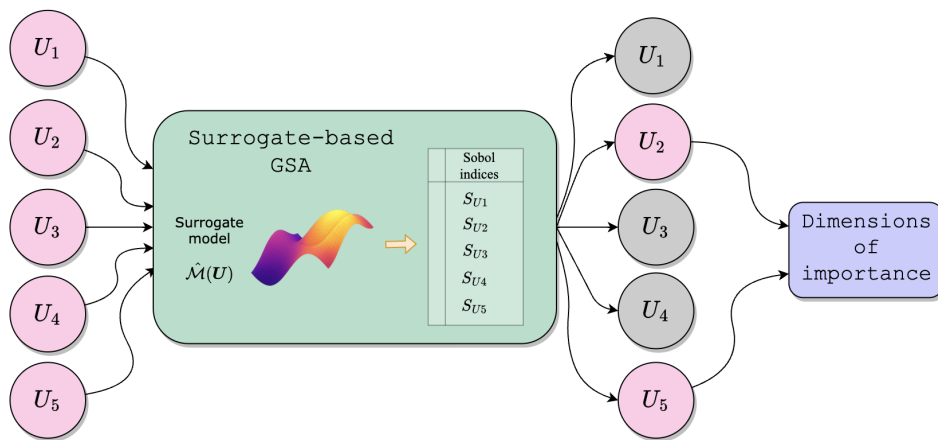


**Figure 1: Flow chart of the surrogate construction and the Sobol GSA analysis.**

The final output of the workflow is the Sobol indices matrix, which includes the sensitivity

results of the system. It can be used for verifying the modeling of the system, instructing the system design or operation, or model reduction.

### 2.1 Surrogate construction with Gaussian process regression

For any multi-variate question, if  $U = (U_1, U_2, \dots, U_n)$  is the input vector, there is a corresponding output vector  $y = \mathcal{M}(U)$  in the output space. The  $\mathcal{M}(U)$  is usually developed from the 1st principles, with the aid of mathematical inference and experimental data. If part of the data is available, neglecting the stress on the 1st principles, a surrogate  $\mathcal{M}'(U) \approx y$  can be developed for the sake of efficiency.

The Gaussian process model is a powerful tool for surrogate modeling, which has already been shown to be effective in many disciplines [16], involving bridges [17], aircrafts [18], electric towers [19], vehicles [20], vessels [21, 22], wind turbine systems [23–27] and more. In the geostatistics, such idea was implemented by Krige, yielding the famous Kriging method. In the recent work [9] with mathematical proof in detail, the Gaussian process regression (GPR) approach and the Kriging method are further explained and connected. It is concluded that the GPR and the Kriging method work in the same mathematical principles, described in a Bayesian perspective.

If we have random variables  $U_1, U_2, \dots, U_d$  as the input variables in variable set  $\mathbf{U}$ , we can have  $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_d] \in \mathbb{R}^{n \times d}$  as the input samples for these variables. Each column of  $\mathbf{U}$  is  $\mathbf{u}_i \in \mathbb{R}^{n \times 1}$  ( $i = 1, 2, \dots, d$ ) is a sample of the input variables  $U_i$ . We can also write  $\mathbf{U} = [\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \dots, \mathbf{u}^{(n)}]^T$ , with each row  $\mathbf{u}^{(i)} \in \mathbb{R}^d$  being a sample of the input variables  $U_1, U_2, \dots, U_d$ . And we have corresponding outputs  $\mathbf{y} = [y^{(1)}, y^{(2)}, \dots, y^{(n)}]^T \in \mathbb{R}^{n \times 1}$  as the output samples which are one-to-one mapped to the input samples. We assume that a function  $f(\mathbf{u})$  exists. And we note the  $\mathbf{y}$  as the noisy outputs of the function  $f(\mathbf{u})$  at the input samples  $\mathbf{U}$ .

$$\mathbf{y} = \mathbf{f} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma_n^2 I) \quad (1)$$

In the equation,  $\mathbf{f} = [f(\mathbf{u}^{(1)}), f(\mathbf{u}^{(2)}), \dots, f(\mathbf{u}^{(n)})]^T \in \mathbb{R}^{n \times 1}$  includes the outputs of the function  $f(\mathbf{u})$  evaluated at those input samples  $\mathbf{U} = [\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \dots, \mathbf{u}^{(n)}]^T \in \mathbb{R}^{n \times d}$ . We further decompose the function  $f(\mathbf{u})$  into the mean function  $m(\mathbf{u})$  and the covariance kernel function  $k(\mathbf{u}, \mathbf{u}')$ .

$$\begin{cases} E[f(\mathbf{u})] = m(\mathbf{u}) \\ Cov[f(\mathbf{u}), f(\mathbf{u}')] = k(\mathbf{u}, \mathbf{u}') \end{cases} \quad (2)$$

We have a mean function  $m(\mathbf{u})$  to describe the global trend of the output samples. And we let  $\boldsymbol{\mu} = [m(\mathbf{u}_1), m(\mathbf{u}_2), \dots, m(\mathbf{u}_n)]^T \in \mathbb{R}^{n \times 1}$  to be the mean of the output samples. If no specific  $m(\cdot)$  is defined, it can be assumed to be constant for all  $\mathbf{u} \in \mathbf{U}$ .

If we have a test input sample  $\mathbf{u}$  in  $\mathbf{U}$ , and  $\mathbf{u}'$  as an arbitrary sample in  $\mathbf{U}$ , we can have one covariance kernel function  $k(\mathbf{u}, \mathbf{u}')$ . And we have a covariance matrix  $\mathbf{K} \in \mathbb{R}^{n \times n}$ , with  $K_{i,j} = k(\mathbf{u}^{(i)}, \mathbf{u}^{(j)})$  for  $i, j = 1, 2, \dots, n$ .

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{u}^{(1)}, \mathbf{u}^{(1)}) & k(\mathbf{u}^{(1)}, \mathbf{u}^{(2)}) & \dots & k(\mathbf{u}^{(1)}, \mathbf{u}^{(n)}) \\ k(\mathbf{u}^{(2)}, \mathbf{u}^{(1)}) & k(\mathbf{u}^{(2)}, \mathbf{u}^{(2)}) & \dots & k(\mathbf{u}^{(2)}, \mathbf{u}^{(n)}) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{u}^{(n)}, \mathbf{u}^{(1)}) & k(\mathbf{u}^{(n)}, \mathbf{u}^{(2)}) & \dots & k(\mathbf{u}^{(n)}, \mathbf{u}^{(n)}) \end{bmatrix} \quad (3)$$

Then,  $\mathbf{f}$  is assumed to follow a multivariate Gaussian distribution as

$$\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}) \quad (4)$$

According to Equation 1, with observed outputs  $\mathbf{y}$ , we can have the joint distribution of  $\mathbf{y}$  as

$$\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K} + \sigma_n^2 \mathbf{I}) \quad (5)$$

In the equation,  $\mathbf{I} \in 1^{n \times n}$  is the identity matrix of size  $n \times n$ .

The unknown hyperparameters, apart from the variance of the noise  $\sigma_n^2$ , also include parameters in the kernel function  $k(\mathbf{u}, \mathbf{u}')$ . For different kernel functions, such as the squared exponential kernel, Matérn kernel and etc., the hyperparameters are different. We note the parameters in the kernel function as  $\boldsymbol{\phi} \in \mathbb{R}^p$ . We write all the parameters as  $\boldsymbol{\theta} = \{\boldsymbol{\phi}, \sigma_n^2\}$ .

To learn the hyperparameters  $\boldsymbol{\theta}$ , we can use maximum likelihood estimation (MLE). We maximize the likelihood of the observed outputs  $\mathbf{y}$  given the input samples  $\mathbf{U}$  and the hyperparameters  $\boldsymbol{\theta}$ .

$$L(\boldsymbol{\theta}) = p(\mathbf{y} \mid \mathbf{U}, \boldsymbol{\theta}) \quad (6)$$

We can further log-transform the likelihood function to maximize the log likelihood function:

$$\log L(\boldsymbol{\theta}) = -\frac{1}{2} \mathbf{y}^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{y} - \frac{1}{2} \log |\mathbf{K} + \sigma_n^2 \mathbf{I}| - \frac{n}{2} \log(2\pi) \quad (7)$$

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{u}_1, \mathbf{u}_1; \boldsymbol{\phi}) & k(\mathbf{u}_1, \mathbf{u}_2; \boldsymbol{\phi}) & \dots & k(\mathbf{u}_1, \mathbf{u}_n; \boldsymbol{\phi}) \\ k(\mathbf{u}_2, \mathbf{u}_1; \boldsymbol{\phi}) & k(\mathbf{u}_2, \mathbf{u}_2; \boldsymbol{\phi}) & \dots & k(\mathbf{u}_2, \mathbf{u}_n; \boldsymbol{\phi}) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{u}_n, \mathbf{u}_1; \boldsymbol{\phi}) & k(\mathbf{u}_n, \mathbf{u}_2; \boldsymbol{\phi}) & \dots & k(\mathbf{u}_n, \mathbf{u}_n; \boldsymbol{\phi}) \end{bmatrix} \quad (8)$$

The log likelihood function in Equation 7 can be maximized with state-of-the-art gradient-based optimization methods on top of quasi-Newton methods, including popular methods like L-BFGS [28], as well as state-of-the-art Adam (short for Adaptive Moment Estimation) optimizer [29] to find optimal kernel and noise parameters. It can be further inferred that, when the prior of  $\mathbf{U}$  and  $\mathbf{y}$  are changed, the hyperparameters  $\boldsymbol{\theta}$  need to be found again.

After the hyperparameters  $\boldsymbol{\theta}$  are learned, we can use the GP model to predict the output at a new test sample. Let  $\mathbf{x}_* \in \mathbb{R}^{1 \times d}$  be a test sample, with which we can have a predicted value  $f_* \in \mathbb{R}^1$ . We define the joint prior of  $\mathbf{y}$  and  $f_*$  as

$$\begin{bmatrix} \mathbf{y} \\ f_* \end{bmatrix} \sim \mathcal{N} \left( \begin{bmatrix} \boldsymbol{\mu} \\ m(\mathbf{u}_*) \end{bmatrix}, \begin{bmatrix} \mathbf{K} + \sigma_n^2 \mathbf{I} & \mathbf{k}_* \\ \mathbf{k}_*^\top & k_{**} \end{bmatrix} \right) \quad (9)$$

Where:

- $\mathbf{k}_* = [k(\mathbf{u}_*, \mathbf{u}_1), \dots, k(\mathbf{u}_*, \mathbf{u}_n)]^\top \in \mathbb{R}^{n \times 1}$
- $k_{**} = k(\mathbf{u}_*, \mathbf{u}_*) \in \mathbb{R}^1$
- $\boldsymbol{\mu} \in \mathbb{R}^{n \times 1}$  is the mean of the output samples  $\mathbf{y}$
- $m(\mathbf{u}_*) \in \mathbb{R}^1$  is the mean of the test sample  $\mathbf{u}_*$

If a constant mean function  $m(\mathbf{u}) = \mu$  is used, the mean of the test sample  $\mathbf{u}_*$  is  $\mu$ . From properties of conditional Gaussian distributions, the posterior distribution, based on samples  $\mathbf{U}$  and  $\mathbf{y}$ , for the given test sample  $\mathbf{x}_*$  is denoted as

$$f_* \mid \mathbf{U}, \mathbf{y}, \mathbf{u}_* \sim \mathcal{N}(E[f_*], \text{Cov}[f_*]) \quad (10)$$

Based on Equation 9, we can have the mean and covariance of the posterior predictive distribution as:

$$E[f_*] = \mu + \mathbf{k}_*^\top (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{y} \in \mathbb{R}^1 \quad (11)$$

$$Cov[f_*] = k_{**} - \mathbf{k}_*^\top (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{k}_* \in \mathbb{R}^1 \quad (12)$$

## 2.2 Global sensitivity analysis with Sobol indices

The Sobol indices are a measure of decomposed variance over the overall variance. Consider a model  $Y = f(X_1, X_2, \dots, X_k)$ , where  $X_i$  are the input variables. The total variance of the output  $Y$  can be decomposed into components attributable to each input and their interactions. This is expressed as:

$$V(Y) = \sum_{i=1}^k V_i + \sum_{1 \leq i < j \leq k} V_{ij} + \sum_{1 \leq i < j < l \leq k} V_{ijl} + \dots + V_{12\dots k} \quad (13)$$

Where,

- -  $V(Y)$  is the total variance of the output.
- -  $V_i$  is the variance contribution of the  $i$ th input alone (first order effect).
- -  $V_{ij}$  is the variance contribution of the interaction between the  $i$ th and  $j$ th inputs (second order effect).
- - Higher order terms represent interactions between three or more inputs.

The first-order Sobol index  $S_i$  for the  $i$ th input is defined as the ratio of the variance contribution of the  $i$ th input alone to the total variance in Eq. 14. The second-order Sobol index  $S_{ij}$  for the interaction between the  $i$ th and  $j$ th inputs is defined as the ratio of the variance contribution of the interaction between the  $i$ th and  $j$ th inputs to the total variance in Eq. 15. If only the first and second order Sobol indices are considered, the total Sobol index  $S_{Ti}$  for the  $i$ th input variable is defined as in Eq. 16.

$$S_i = \frac{V_i}{V(Y)} \quad (14)$$

$$S_{ij} = \frac{V_{ij}}{V(Y)} \quad (15)$$

$$S_{Ti} = S_i + \sum_{1 \leq j \leq i} S_{ij} \quad (16)$$

For  $N$  variables and  $M$  features, the total Sobol index matrix  $S_{Ti}$  is a matrix of dimension  $N \times M$ . An example of the total Sobol index matrix is shown in Table 1.

**Table 1: Total Sobol index matrix  $S_{Ti}$  for N variables and M features**

|          | $X_1$       | $X_2$       | $\cdots$ | $X_N$       |
|----------|-------------|-------------|----------|-------------|
| $Y_1$    | $S_{T,1,1}$ | $S_{T,1,2}$ | $\cdots$ | $S_{T,1,N}$ |
| $Y_2$    | $S_{T,2,1}$ | $S_{T,2,2}$ | $\cdots$ | $S_{T,2,N}$ |
| $\vdots$ | $\vdots$    | $\vdots$    | $\cdots$ | $\vdots$    |
| $Y_M$    | $S_{T,M,1}$ | $S_{T,M,2}$ | $\cdots$ | $S_{T,M,N}$ |

### 3 LLM-based agentic solution

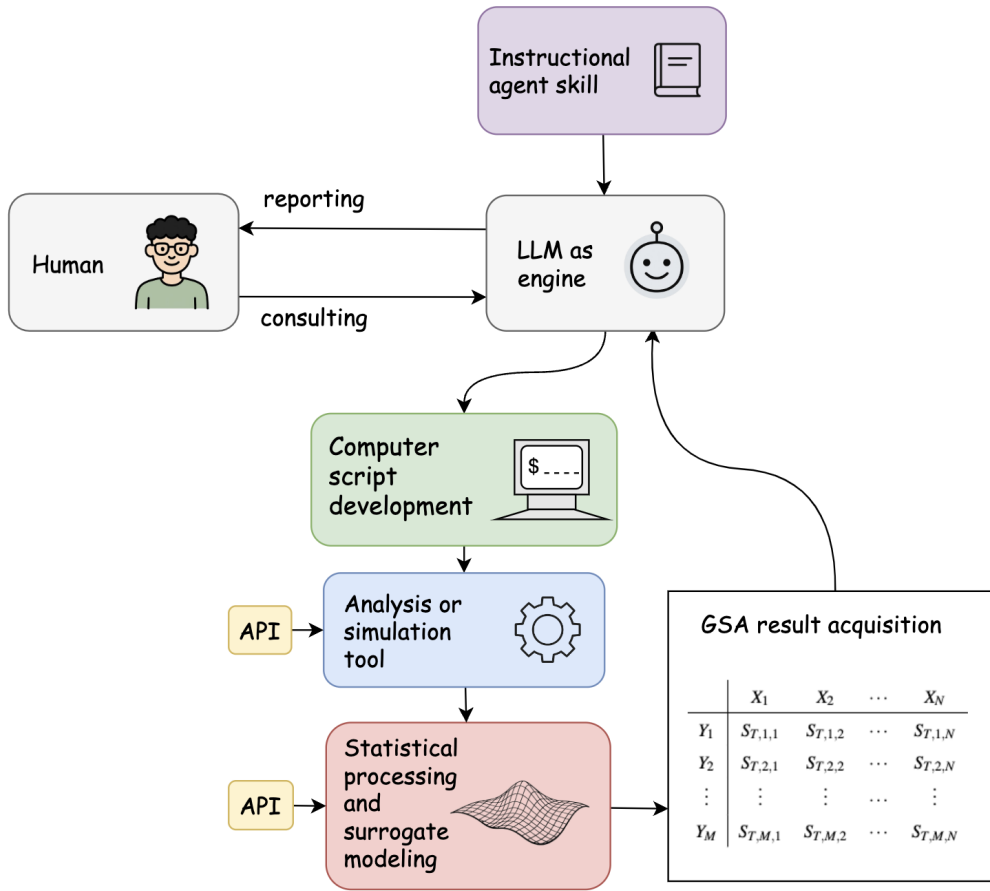


**Figure 2: LLM agent integration framework**

The overall framework of LLM agent integration is shown in Figure 2. The human analyst are given with the right to trigger the overall analysis process. The AI agent, with the instructional agent skill and human analyst's prompts, may target at the specific analysis task, developing the computer script, interacting provided simulation tools and statistical processing tools. For equipping AI agent with the capability of performing the surrogate-based GSA analysis, several prerequisite conditions are needed to be met, as shown in Table 2.

**Table 2: Prerequisite conditions for the surrogate-based GSA analysis**

| Prerequisite condition      | Example                                             |
|-----------------------------|-----------------------------------------------------|
| LLM foundation model        | GPT-5.6-luna [30]                                   |
| LLM agentic framework       | Codex-server [31]                                   |
| Programming language        | Python [32]                                         |
| Simulation tool             | In-house or licensed FEM program, ODE solvers, etc. |
| Statistical processing tool | Scipy [33]                                          |
| GSA tool                    | SALib [34, 35]                                      |
| Surrogate modeling          | Scikit-learn [36]                                   |

The agent skill instruction may tell the AI agent how to call the simulation tools and the procedure of the surrogate-based GSA analysis. In this study, the simulation tools are self-developed and set to be detectable by the working environment of the AI agent. In addition, the simulation tool and the post processing can also be obtained through APIs that the private or industry can adapt to their solutions. For example, the licensed numerical tool such as finite element software can be called through the API. And the preprocessing related to any parameters can be done through the API.

#### 4 Design of case studies

Case studies are designed in this study to test if the AI agent can provide reliable solutions for the surrogate-based global sensitivity analysis (GSA). The studies at this initial stage should be simple but mathematically representative for the problems in the marine engineering domain.

Therefore, three case studies are arranged according to increasing model complexity. Their respective roles within the validation framework can be summarized as

$$\underbrace{\text{Ishigami}}_{\text{Analytical pseudo limite state function}} \longrightarrow \underbrace{\text{10-bar truss}}_{\text{Deformable structural finite element model}} \longrightarrow \underbrace{\text{mass-spring-damper}}_{\text{Floating vessel's dynamic ODE model}}. \quad (17)$$

For each case, the same general computational procedure is followed. A reference dataset is first generated from the original model. A subset of these samples is subsequently used to construct the surrogate model, after which global sensitivity indices are evaluated using the surrogate rather than the original simulator. The surrogate-based results are finally compared against reference sensitivity indices obtained through a sufficiently large direct Monte Carlo analysis. The comparison therefore evaluates two related quantities: the approximation of the model response,

$$\mathbf{X} \rightarrow Y \approx \hat{Y}(\mathbf{X}), \quad (18)$$

and the approximation of the corresponding sensitivity measures,

$$\{S_i, S_{T_i}\} \approx \{\hat{S}_i, \hat{S}_{T_i}\}, \quad (19)$$

where  $S_i$  and  $S_{T_i}$  denote the reference first-order and total-effect Sobol indices, respectively. By progressing from a benchmark with known analytical sensitivity indices to structural and dynamic numerical models, the three examples provide a systematic basis for investigating the accuracy, computational efficiency, and generalizability of the proposed surrogate-GSA methodology.

#### 4.1 Case I: Ishigami Function

The Ishigami function is adopted as the first benchmark because it is a well-established analytical test function for global sensitivity analysis. It is defined as

$$y = \sin x_1 + a \sin^2 x_2 + bx_3^4 \sin x_1, \quad (20)$$

where

$$x_i \sim \mathcal{U}(-\pi, \pi), \quad i = 1, 2, 3, \quad (21)$$

and the commonly used parameter values are

$$a = 7, \quad b = 0.1. \quad (22)$$

Although the model contains only three uncertain variables, it exhibits pronounced nonlinearity and a strong interaction between  $x_1$  and  $x_3$ . Moreover, its Sobol sensitivity indices are available analytically, making it particularly suitable for verifying the complete surrogate-GSA workflow.

This case therefore provides a controlled benchmark in which errors arising from surrogate construction can be distinguished from errors introduced by the sensitivity-analysis procedure itself. A successful surrogate-based method should reproduce both the first-order and total-effect sensitivity indices with a relatively small number of evaluations of the original function.

#### 4.2 Case II: Ten-Bar Truss Structure

The second case considers the classical ten-bar planar truss problem as shown in Figure 3. Compared with the Ishigami function, this example introduces a physically meaningful structural model in which uncertain design or material parameters are propagated through a finite-dimensional structural equilibrium problem.

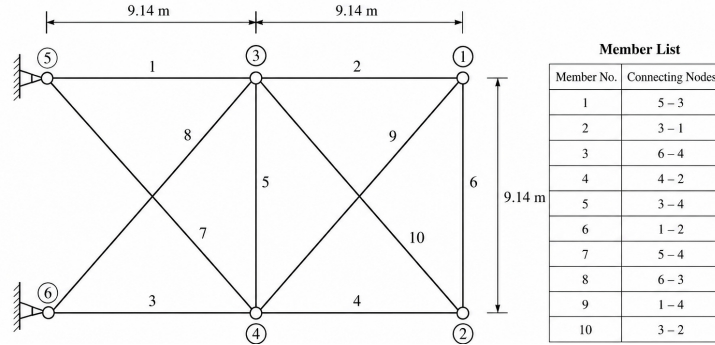


**Figure 3: Illustration of the ten-bar truss structure.**

For a linear elastic truss structure, the static equilibrium equation can be written as

$$\mathbf{K}(\mathbf{u})\mathbf{x} = \mathbf{F}(\mathbf{u}), \quad (23)$$

where  $\mathbf{K}$  is the global stiffness matrix,  $\mathbf{x}$  is the nodal displacement vector, and  $\mathbf{F}$  denotes the externally applied load vector. The uncertain parameter vector  $\mathbf{x}$  may include quantities such as the cross-sectional areas of selected members, Young's modulus, and applied loads.

A representative quantity of interest may be defined as a nodal displacement,

$$y(\mathbf{u}) = x_j(\mathbf{u}), \quad (24)$$

or alternatively as the maximum member stress,

$$y(\mathbf{u}) = \max_i |\sigma_i(\mathbf{u})|. \quad (25)$$

The ten-bar truss provides an intermediate benchmark between an analytical function and a dynamic simulation. Its input–output relationship is generated through a structural solver, while each individual model evaluation remains sufficiently inexpensive that a large Monte Carlo reference solution can be produced. Consequently, high-fidelity estimates of the Sobol indices can be obtained and used as reference values for assessing the surrogate-assisted analysis.

This case is particularly useful for examining whether the surrogate model can correctly identify influential structural parameters and reproduce the relative importance of variables affecting displacement or stress responses.

#### 4.3 Case III: Mass–Spring–Damper Dynamic System

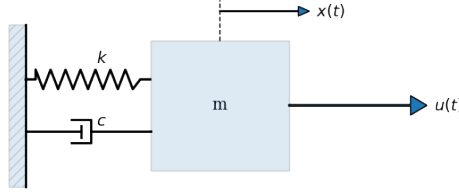


**Figure 4: Illustration of the linear mass–spring–damper system.**

The third case introduces a time-dependent mechanical system represented by a single-degree-of-freedom mass–spring–damper oscillator, as shown in Figure 4. The governing equation is

$$m\ddot{x}(t) + c\dot{x}(t) + kx(t) = u(t; \boldsymbol{\theta}), \quad (26)$$

where  $m$ ,  $c$ , and  $k$  denote the mass, damping coefficient, and stiffness, respectively, while  $f(t; \boldsymbol{\theta})$  represents an external excitation characterized by a set of uncertain parameters  $\boldsymbol{\theta}$ .

The uncertain input vector can therefore be expressed generally as

$$U = [m, c, k, \theta_1, \dots, \theta_p]^T. \quad (27)$$

Unlike the previous two cases, the direct output of the numerical model is a time series. A scalar quantity of interest must consequently be extracted from the dynamic response before constructing the surrogate model. Possible response quantities include the maximum absolute displacement,

$$Y_{\max} = \max_{0 \leq t \leq T} |x(t)|, \quad (28)$$

the root-mean-square response,

$$Y_{\text{RMS}} = \sqrt{\frac{1}{T} \int_0^T x^2(t) dt}, \quad (29)$$

or other response statistics relevant to the considered application. This example represents a more realistic surrogate-modeling problem because each input sample requires the solution of a differential equation followed by response post-processing. It therefore allows the computational advantage of surrogate-assisted GSA to be demonstrated more clearly. In addition, the case provides a natural transition toward more computationally demanding engineering simulations in which repeated direct evaluations for Monte Carlo-based sensitivity analysis may become prohibitively expensive.

#### 4.4 Variables in case studies

To evaluate the effectiveness and robustness of the surrogate-assisted global sensitivity analysis (GSA) framework, three benchmark problems with increasing levels of physical and computational complexity are considered. The selected cases are: (i) the Ishigami function, (ii) a ten-bar truss structural model, and (iii) a mass–spring–damper dynamic system. Together, these examples provide a progressive test hierarchy ranging from a purely analytical nonlinear function to structural and time-dependent engineering models. The uncertain inputs of each problem, together with the corresponding sampling distributions, are listed in Table 3.

**Table 3: Input variables and sampling distributions. All inputs within each problem are sampled independently.**

| Problem            | Variable(s)     | Distribution or sampling range                                                                                                                                                    |
|--------------------|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Ishigami           | $x_1, x_2, x_3$ | $\mathcal{U}[-\pi, \pi]$ for each input; $a = 7$ and $b = 0.1$ are fixed.                                                                                                         |
| 10-bar truss       | $E$             | Young’s modulus, Lognormal with arithmetic mean 68.9 GPa and coefficient of variation 0.05.                                                                                       |
|                    | $P$             | Applied load, Lognormal with arithmetic mean 444.8 kN and coefficient of variation 0.10; the same downward load magnitude is applied at both right-hand nodes.                    |
|                    | $A_h, A_v, A_d$ | Cross-sectional area of the horizontal, vertical, and diagonal members, each $\mathcal{U}[0.01548384, 0.02322576]$ m <sup>2</sup> , corresponding to $30 \text{ in}^2 \pm 20\%$ . |
| Mass–spring–damper | $m$             | Mass, Lognormal with arithmetic mean 1.00 kg and coefficient of variation 0.05.                                                                                                   |
|                    | $k$             | Stiffness, Lognormal with arithmetic mean $100.0 \text{ N m}^{-1}$ and coefficient of variation 0.10.                                                                             |
|                    | $c$             | Damping coefficient, $\mathcal{U}[0.8, 1.6]$ N s m <sup>-1</sup> .                                                                                                                |
|                    | $F_0$           | Amplitude of the harmonic excitation, Lognormal with arithmetic mean 10.0 N and coefficient of variation 0.10.                                                                    |
|                    | $\omega$        | Frequency of the harmonic excitation, $\mathcal{U}[8.0, 12.0]$ rad s <sup>-1</sup> ; this interval crosses the nominal natural frequency.                                         |

The purpose of using multiple case studies is (i) to assess the predictive accuracy of the surrogate models, (ii) to examine whether the resulting sensitivity measures can reliably reproduce those obtained from the corresponding reference models, and (iii) to represent different characteristics that commonly appear in engineering uncertainty quantification, including strong nonlinearity, interaction effects between uncertain variables, structural response calculations, and dynamic response under external excitation.

## 5 Results and discussions

### 5.1 Example agent solutions

The sequential designs are determined by the AI-agent to construct the three Kriging surrogates are summarized in Table 4. All models share the same kernel family and input scaling, while the acquisition function, output transformation, and sampling budget are adapted to each case.

**Table 4: Surrogate and sequential-design settings. Case I: Ishigami function; Case II: ten-bar truss; Case III: mass–spring–damper. Held-out test evaluations are excluded from the sampling budget  $N_{\max}$ .**

| Setting                             | Case I                 | Case II                  | Case III                              |
|-------------------------------------|------------------------|--------------------------|---------------------------------------|
| Model                               | GPR (Kriging)          | GPR (Kriging)            | GPR on $\log y$                       |
| Kernel, Eq. (31)                    | Matérn 5/2, ARD, $d=3$ | Matérn 5/2, ARD, $d=6$   | Matérn 5/2, ARD, $d=5$                |
| Input scaling                       |                        | unit hypercube, Eq. (30) |                                       |
| Output                              | $y$ (direct)           | $y$ (direct)             | $\log y$ , $\hat{y} = e^{\mu_{\log}}$ |
| $y$ -normalization                  | yes                    | yes                      | yes (log scale)                       |
| Acquisition                         | $\alpha_I$ , Eq. (32)  | $\alpha_{II}$ , Eq. (33) | $\alpha_{III}$ , Eq. (34)             |
| Initial samples $n_{\text{init}}$   | 10                     | 16                       | 14                                    |
| Budget $N_{\max}$                   | 150                    | 75                       | 150                                   |
| Candidate pool $N_{\text{pool}}$    | 6000                   | 7000                     | 9000                                  |
| Sobol checkpoint                    | every 5 samples        | every 6 samples          | every 5 samples                       |
| Held-out test set $N_{\text{test}}$ | 2000                   | 1000                     | 2000                                  |

Inputs are mapped onto the unit hypercube before training,

$$\tilde{x}_i = \frac{x_i - x_{i,L}}{x_{i,U} - x_{i,L}}, \quad i = 1, \dots, d, \quad (30)$$

where  $x_{i,L}$  and  $x_{i,U}$  are the lower and upper bounds of the  $i$ th coordinate. The covariance is a constant-scaled Matérn kernel of smoothness  $\nu = 5/2$  with automatic relevance determination (ARD), plus a fixed white-noise nugget,

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \left( 1 + \sqrt{5} r + \frac{5}{3} r^2 \right) e^{-\sqrt{5} r} + \sigma_n^2 \delta_{\mathbf{x}\mathbf{x}'}, \quad r = \sqrt{\sum_{i=1}^d \frac{(\tilde{x}_i - \tilde{x}'_i)^2}{\ell_i^2}}. \quad (31)$$

Here  $\sigma_f^2$  is the signal variance,  $\boldsymbol{\ell} = (\ell_1, \dots, \ell_d)$  are the ARD length-scales,  $\sigma_n^2$  is a fixed nugget, and  $\delta_{\mathbf{x}\mathbf{x}'}$  is the Kronecker delta. The training responses are standardized to zero mean and unit variance.

Let  $\mu(\mathbf{x})$  and  $\sigma(\mathbf{x})$  denote the GP posterior mean and standard deviation. At each iteration a new sample is chosen from a discrete candidate pool  $\mathcal{P}$  of size  $N_{\text{pool}}$  by maximizing an acquisition score  $\alpha(\mathbf{x})$ . The three scores are

$$\alpha_I(\mathbf{x}) = \sigma(\mathbf{x}), \quad (32)$$

$$\alpha_{II}(\mathbf{x}) = [\sigma(\mathbf{x})^2 + 2|\mu(\mathbf{x}) - \bar{\mu}| \sigma(\mathbf{x})] d(\mathbf{x}, \mathcal{X}), \quad (33)$$

$$\alpha_{III}(\mathbf{x}) = \sigma_{\log}(\mathbf{x}) \left[ 1 + 0.75 [z(\mu_{\log}(\mathbf{x}))]_+ \right]. \quad (34)$$

Equation (32) is maximum posterior uncertainty. In (33),  $\bar{\mu}$  is the mean of  $\mu$  over  $\mathcal{P}$  and  $d(\mathbf{x}, \mathcal{X}) = \min_{\mathbf{x}' \in \mathcal{X}} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}'\|_2$  is the distance to the current training set  $\mathcal{X}$ , so that both variance

impact and space-filling are rewarded. For Case III the surrogate is trained on  $\log y$ ;  $\mu_{\log}$  and  $\sigma_{\log}$  are the corresponding posterior statistics, predictions are recovered by  $\hat{y}(\mathbf{x}) = \exp(\mu_{\log}(\mathbf{x}))$ , and the positive  $z$ -score

$$[z(\mu_{\log}(\mathbf{x}))]_+ = \max\left(0, \frac{\mu_{\log}(\mathbf{x}) - \bar{\mu}_{\log}}{s_{\log}}\right) \quad (35)$$

emphasizes high-amplitude (near-resonance) regions. Here  $\bar{\mu}_{\log}$  and  $s_{\log}$  are the mean and standard deviation of  $\mu_{\log}$  over  $\mathcal{P}$ .

### 5.2 Performance comparison

According to the surrogate-based GSA strategies and the executable code, the objective sobol indices are obtained. The convergence of the surrogate's predictive performance for the three case studies are shown in the following Figures 5, 6, and 7. The surrogate GSA solutions suggested by the AI agent achieve good convergence for the three case studies.

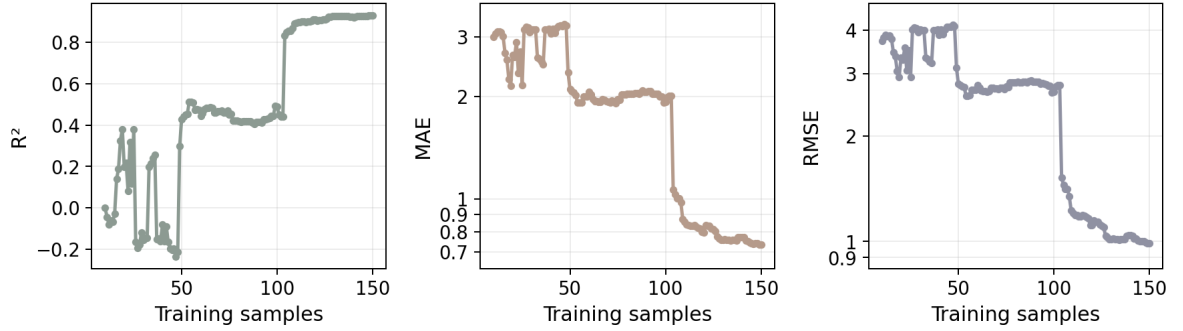


**Figure 5: Surrogate accuracy convergence of the Ishigami function (Final results:  $R^2 = 0.9299$ ,  $\text{MAE} = 0.73618$ ,  $\text{RMSE} = 0.98650$ ).**

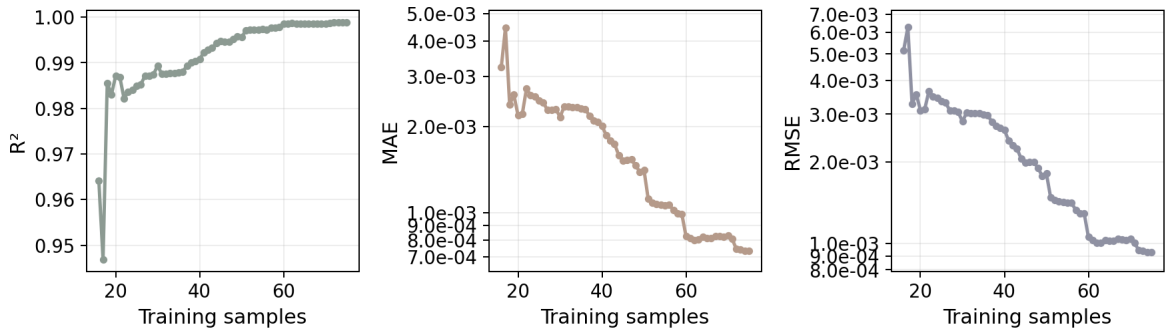


**Figure 6: Surrogate accuracy convergence of the ten-bar truss structure (Final results:  $R^2 = 0.9988$ ,  $\text{MAE} = 7.32938 \times 10^{-4}$ ,  $\text{RMSE} = 9.30918 \times 10^{-4}$ ).**

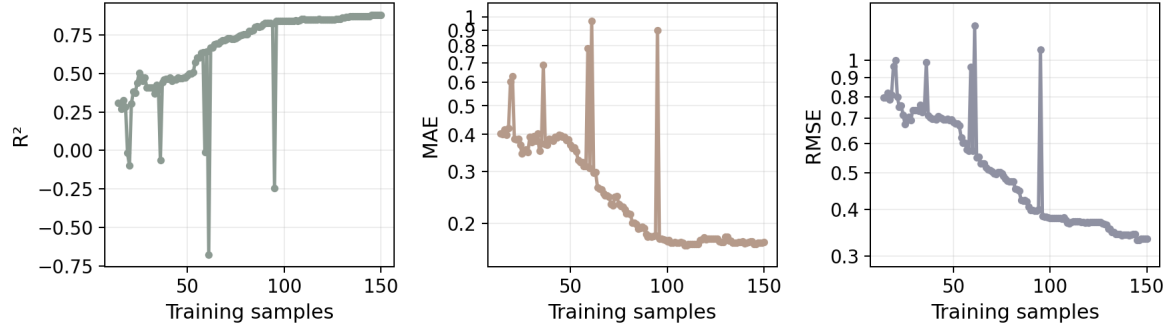


**Figure 7: Surrogate accuracy convergence of the mass-spring-damper dynamic system. (Final results:  $R^2 = 0.8780$ , MAE= 0.17294, RMSE= 0.33368).**

In the surrogate learning setup developed by the AI agent, the final performance of the surrogate in the case study of the 10-bar truss structure becomes the best. The surrogate training remains room of improvement for the other two case studies.

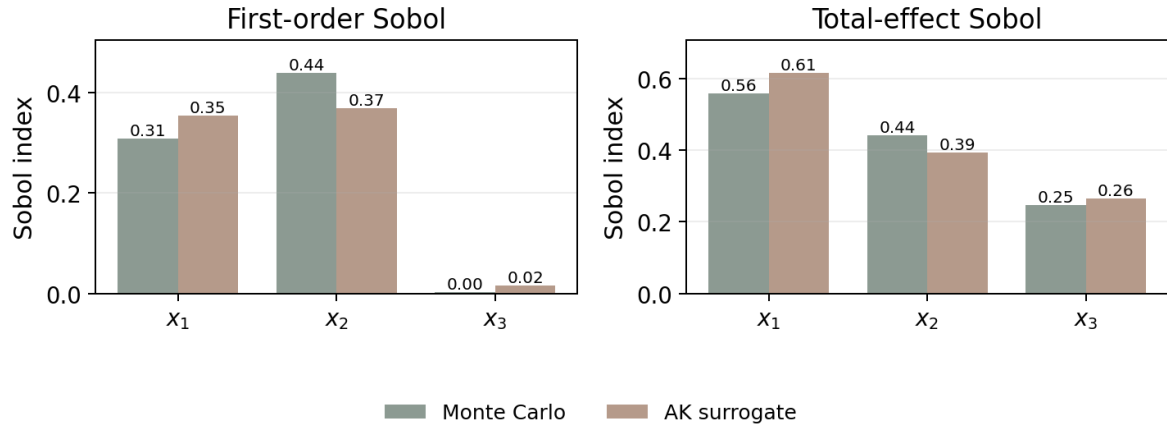


**Figure 8: Sobol indices of the Ishigami function.**

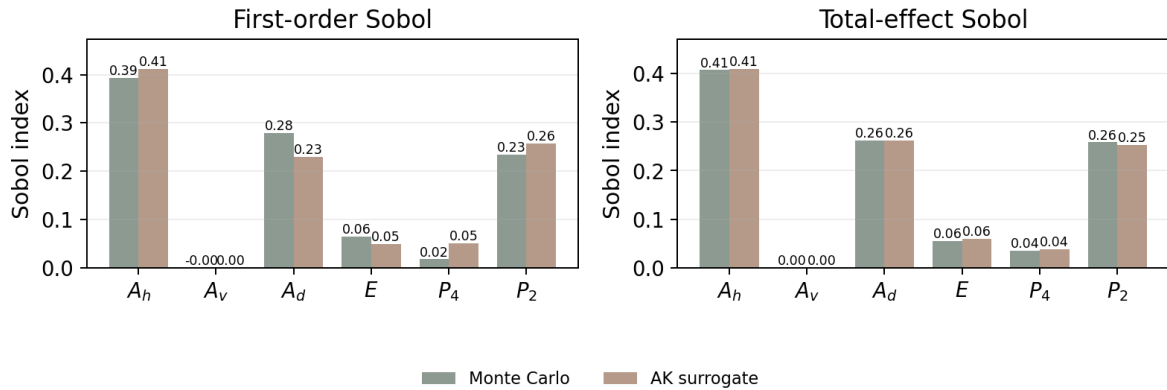


**Figure 9: Sobol indices of the ten-bar truss structure.**

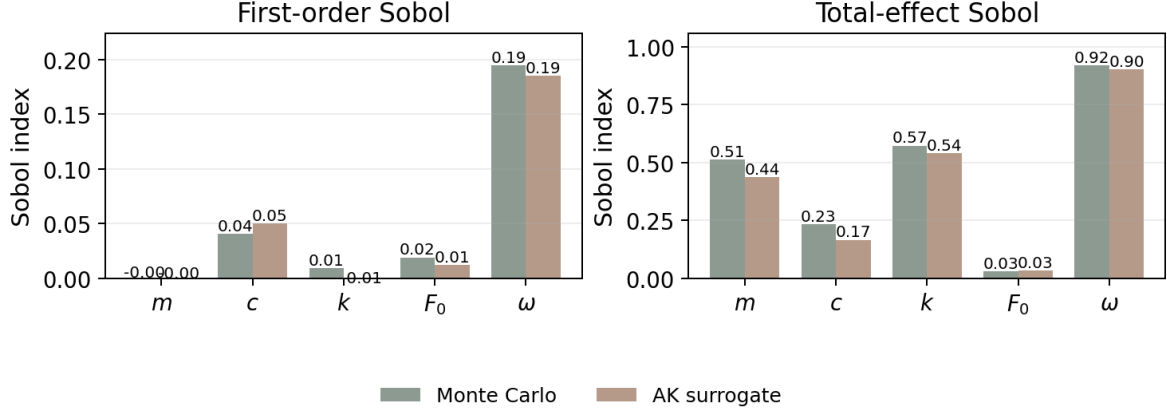


**Figure 10: Sobol indices of the mass-spring-damper dynamic system.**

In the comparison of the Sobol indices in Figures 8, 9, and 10, the agent designed surrogate GSA solutions are generally consistent with the reference Sobol indices obtained from direct Monte Carlo simulation. Even though there exists imprecision in the surrogate’s predictive performance, the ranking of the sensitivity indices is correct for the three case studies. This further confirms that the agentic solution has the potential to automate such surrogate-based GSA analysis for more complex engineering problems.

### 5.3 Discussions

In the designed case studies, the agents are instructed to perform the surrogate-GSA analysis and interpret the results. The current solutions of the agent can meet the fundamental requirements of the surrogate-based GSA analysis. The advantage of LLM’s integration lies at its efficient generation of computer scripts. To further evaluate this, the token usage of the agent’s solutions is counted, by the “tiktoken” [37] package. The token usage of the agent’s solutions is shown in Table 5. A token may include a word, a symbol, or a punctuation. Approximately 5000 tokens are needed to develop a computer script for the surrogate-based GSA analysis. If the LLM foundation model provider can output 100 tokens per second, the total time for developing the computer scripts by the agent can be approximately 1-2 minutes. The workload was estimated to be 2-3 days according to the authors, with respect to script development and troubleshooting in a manual manner. The introduction of the AI agentic solution is hereby confirmed to leverage the efficiency of the complex engineering analysis.

**Table 5: Token usage of the agent’s solutions in average.**

| Tokenizer     | Tokens |
|---------------|--------|
| “cl100k_base” | 4,295  |
| “o200k_base”  | 4,326  |
| “p50k_base”   | 5,517  |

## 6 Conclusions and recommendations

This study sets a foundation of the Human-AI-In-the-Loop (H-A-I-L) framework for smart engineering solutions. The framework is designed to be a standalone service, with the aid of the Large Language Model (LLM)-enabled agent.

The major contribution of this study sits at providing reference testing framework about agentic solutions for the surrogate-based GSA analysis. The framework is designed to be scalable and adaptable to the needs of the private or industrial applications.

As proved in the designed case studies, the LLM agent in the H-A-I-L framework is able to develop efficient surrogate-based GSA solutions for the Sobol indices feedback to the human engineer or analyst. The outcomes are found to satisfy the need of ranking those sensitivity indices.

Based on the framework, several recommendations for improvements are provided as follows.

- Different LLM solutions can be compared and evaluated.
- The APIs for various commercial or open-source tools can be developed and tested.
- More complex case studies can be established to evaluate the framework.
- Multiple agent runs can be tested to evaluate the stability and reliability of the framework.

## References

- [1] Sobol I M 2001 Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Math. Comput. Simul.* **55**(1), 271–280
- [2] Saltelli A, et al. 2007 Sensitivity analysis: from theory to practice. *Global Sensitivity Analysis. The Primer* (Chichester: John Wiley & Sons) chap 6, pp 237–275
- [3] Saltelli A, et al. 2007 Variance-based methods. *Global Sensitivity Analysis. The Primer* (Chichester: John Wiley & Sons) chap 4, pp 155–182
- [4] Cheng K, Lu Z, Ling C and Zhou S 2020 Surrogate-assisted global sensitivity analysis: an overview. *Struct. Multidiscip. Optim.* **61**(3), 1187–1213
- [5] Ma Y, Jiang Z, Ong M C and Li L 2025 Surrogate-based global sensitivity analysis of a floating fish cage array with shared mooring. *Ocean Eng.* **342**, 122993
- [6] Sudret B 2008 Global sensitivity analysis using polynomial chaos expansions. *Reliab. Eng. Syst. Saf.* **93**(7), 964–979
- [7] Tsokanas N, Pastorino R and Stojadinović B 2022 A comparison of surrogate modeling techniques for global sensitivity analysis in hybrid simulation. *Mach. Learn. Knowl. Extr.* **4**(1), 1–21
- [8] Abbiati G, Marelli S, Tsokanas N, Sudret B and Stojadinović B 2021 A global sensitivity analysis framework for hybrid simulation. *Mech. Syst. Signal Process.* **146**, 106997
- [9] Marinescu M 2024 Explaining and connecting Kriging with Gaussian process regression. *Preprint* arXiv:2408.02331
- [10] Vaswani A, et al. 2017 Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30**, 5998–6008
- [11] Brown T B, et al. 2020 Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **33**, 1877–1901

- [12] Radford A, et al. 2019 Language models are unsupervised multitask learners. Tech. rep. OpenAI URL [https://cdn.openai.com/better-language-models/language\\_models\\_are\\_unsupervised\\_multitask\\_learners.pdf](https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf)
- [13] Chen M, et al. 2021 Evaluating large language models trained on code. *Preprint arXiv:2107.03374*
- [14] Hoffmann J, et al. 2022 Training compute-optimal large language models. *Adv. Neural Inf. Process. Syst.* **35**, 30016–30030
- [15] Ouyang L, et al. 2022 Training language models to follow instructions with human feedback. *Adv. Neural Inf. Process. Syst.* **35**, 27730–27744
- [16] Kaymaz I 2005 Application of kriging method to structural reliability problems. *Struct. Saf.* **27**(2), 133–151
- [17] Lu P, Xu Z, Chen Y and Zhou Y 2020 Prediction method of bridge static load test results based on Kriging model. *Eng. Struct.* **214**, 110641
- [18] Zhao X, Yang Y and Ma X 2021 Kriging aerodynamic modeling and multi-objective control allocation for flying wing UAVs with morphing trailing-edge. *IEEE Access* **9**, 62394–62404
- [19] Cai Y and Wan J 2021 Wind-resistant capacity modeling for electric transmission line towers using Kriging surrogates and its application to structural fragility. *Appl. Sci.* **11**(11), 4714
- [20] Yan W J, Zhan J Z, Yuen K V, Ren W X and Papadimitriou C 2024 PC-Kriging-powered parallelizing Bayesian updating for stochastic vehicle-track dynamical system with contact force measurements and Gaussian process discrepancy model. *Eng. Struct.* **318**, 118578
- [21] Li M, Zhang P, Liu J and Cheng Y 2018 A methodology for wave load prediction of damaged ship based on Kriging model. *Proc. 28th Int. Ocean and Polar Engineering Conf.* (Sapporo, Japan: ISOPE)
- [22] Liu X, Zhao W and Wan D 2022 Multi-fidelity Co-Kriging surrogate model for ship hull form optimization. *Ocean Eng.* **243**, 110239
- [23] Golparvar B, Papadopoulos P, Ezzat A A and Wang R Q 2021 A surrogate-model-based approach for estimating the first and second-order moments of offshore wind power. *Appl. Energy* **299**, 117286
- [24] Rajiv G, Verma M and Subbulakshmi A 2023 Gaussian process metamodels for floating offshore wind turbine platforms. *Ocean Eng.* **267**, 113206
- [25] Chen Y, et al. 2023 Surrogate models for twin-VAWT performance based on Kriging and artificial neural networks. *Ocean Eng.* **273**, 113947
- [26] Han F, Li X, Shi W, Wang B and Wang W 2024 Generic fully coupled framework for reliability assessment of offshore wind turbines under typical limit states. *Eng. Struct.* **304**, 117692
- [27] Ren C and Xing Y 2024 An efficient active learning Kriging approach for expected fatigue damage assessment applied to wind turbine structures. *Ocean Eng.* **305**, 118034

- [28] Byrd R H, Lu P, Nocedal J and Zhu C 1995 A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.* **16**(5), 1190–1208
- [29] Kingma D P and Ba J 2017 Adam: a method for stochastic optimization. *Preprint* arXiv:1412.6980
- [30] OpenAI 2026 GPT-5.6 Luna model. OpenAI API documentation URL <https://developers.openai.com/api/docs/models/gpt-5.6-luna> (accessed 31 August 2026)
- [31] OpenAI 2026 Codex App Server. Official OpenAI documentation URL <https://learn.chatgpt.com/docs/app-server> (accessed 31 August 2026)
- [32] Python Software Foundation 2026 *The Python Language Reference* URL <https://docs.python.org/3/reference/> (accessed 31 August 2026)
- [33] Virtanen P, et al. 2020 SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Methods* **17**(3), 261–272
- [34] Herman J and Usher W 2017 SALib: an open-source Python library for sensitivity analysis. *J. Open Source Softw.* **2**(9), 97
- [35] Iwanaga T, Usher W and Herman J 2022 Toward SALib 2.0: advancing the accessibility and interpretability of global sensitivity analyses. *Socio-Environ. Syst. Model.* **4**, 18155
- [36] Pedregosa F, et al. 2011 Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830
- [37] OpenAI 2026 tiktoken. OpenAI API documentation URL [https://developers.openai.com/cookbook/examples/how\\_to\\_count\\_tokens\\_with\\_tiktoken](https://developers.openai.com/cookbook/examples/how_to_count_tokens_with_tiktoken) (accessed 31 August 2026)