Shapley Additive Explanations for Knowledge Discovery via Surrogate Models

Pramudita S. Palar, Lavi R. Zuhal, Yohanes Bimo Dwianto, Koji Shimoyama, Joseph Morlier

AIAA SciTech Forum
23-27 January 2023, National Harbor, MD & Online
AIAA SCITECH 2023 Forum

10.2514/6.2023-0332

Shapley Additive Explanations for Knowledge Discovery via Surrogate Models

Pramudita Satria Palar*, Lavi Rizki Zuhal†, Yohanes Bimo Dwianto‡
Institut Teknologi Bandung, Jl. Ganesha No. 10, West Java, Indonesia

Koji Shimoyama§
Institute of Fluid Science, Tohoku University, Japan

Joseph Morlier¶
ICA, Université de Toulouse, ISAE–SUPAERO, INSA, CNRS, MINES ALBI, UPS, Toulouse, France

It is sometimes desirable to delve further into how the inputs affect the output in design optimization and uncertainty analysis. Surrogate models such as Gaussian Process Regression and support vector regression are useful for such tasks and can be further enhanced by introducing advanced post-processing methods. This paper investigates Shapley Additive Explanation (SHAP) as a tool to aid surrogate-assisted data-driven analysis. In particular, surrogate-enabled SHAP analysis allows visualization of the input-output relationship in a meaningful way using summary SHAP plots and SHAP dependence plots. Some important information that can be extracted and visualized from SHAP includes the importance of input variables (i.e., global sensitivity analysis), nonlinearity level, and level of interactions. The benefits of Shapley values for engineering analysis using surrogate models are demonstrated in three engineering test problems.

I. Introduction

Global sensitivity analysis (GSA) is an important field in which the objective is to quantify the relative impact of input variables and their interactions on the output [1, 2]. GSA is widely used in uncertainty analysis, although other domains, such as design optimization and exploration, also take advantage of GSA. For example, designers can obtain better information regarding which variables should be prioritized when optimizing a system with GSA. Similarly, in uncertainty analysis, GSA helps engineers determine which variables should be controlled to reduce uncertainties in the output. Variance-based sensitivity analysis techniques (primarily Sobol indices) are arguably the most widely used methods due to their intuitiveness [3, 4]. In modern practice, GSA is frequently performed using surrogate models to reduce the computation cost for estimating sensitivity indices. Polynomial chaos expansion (PCE) [5] is one of the most

*Assistant Professor, Faculty of Mechanical and Aerospace Engineering, Institut Teknologi Bandung. Email: pramsp@ftmd.itb.ac.id
†Associate Professor, Faculty of Mechanical and Aerospace Engineering, Institut Teknologi Bandung.
‡Lecturer, Faculty of Mechanical and Aerospace Engineering, Institut Teknologi Bandung.
§Associate Professor.
¶Professor.

widely used surrogate models for handling a GSA task [6]. Besides PCE, other popular models for GSA include Kriging and support vector regression [7–9]. Beyond GSA, sometimes there is a need to take a peek inside the constructed surrogate models, aiming to understand the inside of the input-output relationship better. For example, one might want to know the nonlinearity level or the nature of the interaction between variables in such a relationship.

Analyzing the importance of variables is also central to one of the current subjects of interest in machine learning: explainability [10, 11]. In this regard, explainability is particularly important for black-box machine learning models, which expressions are too complex to decipher. Such complexity is in contrast to simple but interpretable models such as linear regression or logistic regression. Interpretable models, as the name suggests, provide a form that helps gain insight and is also easily understood by a human. The main drawback of simple models is their limited predictive power, which is why complex models are desirable in many applications. It is then common to apply post hoc explanations to interpret complex black-box models.

Several tools to help interpret black-box models have been developed, such as partial dependence plot [12], individual conditional expectation (ICE) [13], and local interpretable model-agnostic explanations (LIME) [14]. The techniques mentioned above work by dissecting the black-box model into a post hoc explanation that works at either a global or local level (i.e., to a single prediction level). For example, PDP depicts the marginal effect of an input on the prediction of a model to show the relationship between the input and the output. Of interest in this paper is the SHapley Additive exPlanation (SHAP) [15], which is based on the Shapley values from game theory [16]. SHAP is convenient since it reveals several essential pieces of information at once. Moreover, some fast methods exist for calculating SHAP for several machine learning models, such as random forest and gradient boosting. However, despite the potential of SHAP for engineering design or uncertainty analysis, to the best of our knowledge, there are only a few papers that deploy SHAP for engineering problems (e.g., see [17, 18]). In this regard, SHAP can be utilized with commonly used surrogate/machine learning models in engineering, e.g., Kriging and PCE.

Often, there are situations when it is desirable to visualize the input-output relationship in either design optimization or uncertainty analysis. For example, in design exploration, one wants to investigate how specifically changing an input variable would impact the performance of the design. Unfortunately, direct visualization is impossible for high-dimensional input spaces, which is why specialized techniques are required for visual analysis. Examples of statistical machine learning and data mining techniques that have been applied to handle such tasks include self-organizing map [19, 20], active subspace [21, 22], and proper orthogonal decomposition [23]. Typical questions include: how strong is the relationship between inputs and output? How nonlinear is the input-output relationship? How does a single variable interact with other variables? GSA metrics such as Sobol indices can answer some of these questions. However, users can be better informed if these questions are answered visually. For example, if a function is nonlinear, we want to know how specifically the inputs affect the output. Tools from the interpretable ML field can then be of great advantage to aid such an endeavor.

In this paper, we investigate the use of SHAP to aid data-driven analysis using surrogate models, with applications toward design optimization and uncertainty analysis. Due to its model-agnostic nature, SHAP can work with any supervised machine learning model. A surrogate model is essentially a supervised machine learning, either as an interpolation or regression technique, in which SHAP can be conveniently applied to reveal important insight. This paper aims to introduce SHAP as an additional instrument for helping design exploration and uncertainty analysis.

II. Surrogate modeling

A surrogate model is essentially a supervised machine learning model. That is, given a vector of input variables $\boldsymbol{x} = \{x_1, x_2, \ldots, x_m\}^T$, where m is the input dimensionality, and a label $y = f(\boldsymbol{x})$, a surrogate model works by approximating the relationship between the input and the output. The first step in building a surrogate model is to provide the experimental design $\mathcal{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}\}^T$, where n is the sample size, and the responses $\boldsymbol{y} = \{y^{(1)}, y^{(2)}, \ldots, y^{(n)}\} = \{f(\boldsymbol{x}^{(1)}), f(\boldsymbol{x}^{(1)}), \ldots, \boldsymbol{x}^{(n)}\}$. The black-box function $f(\boldsymbol{x})$ is evaluated through either a numerical model or physical experiment. In this paper, we focus on data from numerical models to create a surrogate model. The input domain is defined as $\boldsymbol{\Omega}$, where $\boldsymbol{\Omega} = \prod_{i=1}^{m} \Omega_i$ and Ω_i is the marginal input for the i-th variable.

This paper only briefly explains GPR and SVR since we focus on the interpretability of a surrogate model. Readers are referred to more specific literature on GPR (e.g., [24, 25]) and SVR (e.g., [26–28]) for more details. We use KADAL, an in-house surrogate modeling code, to build the GPR and SVR model.

A. Gaussian Process Regression

The main ingredient of GPR is the assumption that the black-box function is a realization of the following stochastic process:

$$Y(\boldsymbol{x}) = \mu_{GP}(\boldsymbol{x}) + Z(\boldsymbol{x}), \tag{1}$$

where $\mu_{GP}(x)$ is the mean function and $Z(\boldsymbol{x})$ is a zero-mean Gaussian process. For simplicity, we assume that $\mu_{GP}(x)$ is constant in the in the entire input space, i.e., $\mu_{GP}(x) = \mu_{GP}$.

GPR assumes that the responses are correlated with each other. Let us denote two points in the input space, $\boldsymbol{x}$ and $\boldsymbol{x'}$, which responses are correlated to a degree. The correlation is modeled by the kernel function $k(\boldsymbol{x}, \boldsymbol{x'}; \boldsymbol{\theta}) = \mathrm{corr}(y, y'; \boldsymbol{\theta})$, where $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_m\}$ is the length scales. In this regard, the length scale indicates the degree of correlation between the responses. This paper uses the squared-exponential kernel function to model the correlation between input points. Our implementation uses the squared-exponential kernel to model the correlation between responses at input points.

Construction of a GPR model requires several information. First, an $n \times n$ correlation matrix $\boldsymbol{R}$ is constructed, with its (i, j)-th component is as follows: $R_{ij} = k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}; \boldsymbol{\theta})$, where $\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)} \in \mathcal{X}$. It is also necessary to build a correlation vector $\boldsymbol{r}(\boldsymbol{x}) = \{k(\boldsymbol{x}, \boldsymbol{x}^{(1)}; \boldsymbol{\theta}), \ldots, k(\boldsymbol{x}, \boldsymbol{x}^{(n)}; \boldsymbol{\theta})\}^T$, especially for making a prediction. GPR then predicts the

response at an arbitrary location using the following expression:

$$\hat{y}(\boldsymbol{x}) = \mu_{GP} + \boldsymbol{r}^T \boldsymbol{R}^{-1}(\boldsymbol{y} - \mathbf{1}\mu_{GP}), \tag{2}$$

Hyperparameter optimization aims to find the best set of hyperparameters (including length scales) that optimizes a certain objective function. The most common approach for hyperparameter optimization is to maximize the log-likelihood function, written as:

$$\ln \mathcal{L}(\boldsymbol{\theta}) = -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\sigma_{GP}^2) - \frac{1}{2}\ln(|\boldsymbol{R}|) - \frac{(\boldsymbol{y} - \mathbf{1}\mu_{GP})^T \boldsymbol{R}^{-1}(\boldsymbol{y} - \mathbf{1}\mu_{GP})}{2\sigma_{GP}^2}, \tag{3}$$

where σ_{GP}^2 is the process variance. Let us define the vector of hyperparameters $\boldsymbol{\gamma} = \{\boldsymbol{\theta}, \mu_{GP}, \sigma_{GP}^2\}$. First, μ_{GP} is estimated as

$$\hat{\mu}_{GP} = (\mathbf{1}^T \boldsymbol{R}^{-1}\mathbf{1})^{-1}\mathbf{1}^T \boldsymbol{R}^{-1}\boldsymbol{y}. \tag{4}$$

Notice that an analytical formulation exists for estimating σ_{GP}^2 if the GPR exactly interpolates the model. However, no such formulation exists for regressing GP, which is why the process variance should be tuned together with the lengthscales using a numerical optimization procedure. Our implementation uses the covariance matrix adaptation evolution strategy [29] and local hill climbing to optimize the hyperparameters.

B. Support Vector Regression

In contrast to GPR, in which a kernel defines the correlation, SVR uses a kernel function to perform mapping into high-dimensional space. To grasp the concept of SVR, let us begin with the linear SVR that makes a prediction using the vector of coefficients $\boldsymbol{w}_{SV}$ and the bias term b, reads as

$$\hat{f}(\boldsymbol{x}) = \boldsymbol{w}_{SV}^T \boldsymbol{x} + b. \tag{5}$$

SVR trains the coefficients and the bias by minimizing the so-called ε-insensitive loss function. The definition ε is such that the loss function penalizes points outside the ε boundary.

There are various versions of SVR depending on the penalization type. In this paper, we use the L_2-SVR which reads as

$$\mathcal{L}_2^{\varepsilon} = \begin{cases} 0 & \text{if } \left|\hat{f}^{SVR}(\boldsymbol{x}) - y)\right| < \varepsilon \\ \left(\left|\hat{f}^{SVR}(\boldsymbol{x}) - y)\right| - \varepsilon\right)^2 & \text{otherwise} \end{cases} \tag{6}$$

As one can see from Eq. (6), the L_2-SVR uses a quadratic penalty in the formulation. After defining the penalization

type, an SVR model is constructed by solving the following minimization problem:

$$\min \ \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{2}\sum_{i=1}^{n}(\xi_i + \xi_i^*)^2, \tag{7}$$

$$\text{subject to } y - \boldsymbol{w}^T\boldsymbol{x}_i - b \leq \varepsilon + \xi_i, \tag{8}$$

$$\boldsymbol{w}^T\boldsymbol{x}_i + b - y_i \leq \varepsilon + \xi_i^*, \tag{9}$$

$$\xi_i, \xi_i^* \leq 0, \tag{10}$$

where ξ_i and ξ_i^* are slack-variables and C is the regularization parameter.

Central to nonlinear SVR is the "kernel trick," which allows an SVR to construct a nonlinear prediction using the mapping function $\Phi(.)$. The prediction then reads as

$$\hat{f}(\boldsymbol{x}) = \boldsymbol{w}_{SV}^T\Phi(\boldsymbol{x}) + b = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\Phi(\boldsymbol{x}_i)^T\Phi(\boldsymbol{x}_i) + b. \tag{11}$$

The high-dimensional product $\Phi(\boldsymbol{x}_i)^T\Phi(\boldsymbol{x}_i)$ is accomplished by the kernel function $k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$. We can then write the prediction of a nonlinear SVR as

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)k(\boldsymbol{x}^{(i)}, \boldsymbol{x}) + b. \tag{12}$$

The training process in SVR is accomplished by solving the following dual form:

$$\min \frac{1}{2}\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{bmatrix}^T\begin{bmatrix}\tilde{\boldsymbol{K}} & -\tilde{\boldsymbol{K}}\\ -\tilde{\boldsymbol{K}} & \tilde{\boldsymbol{K}}\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{bmatrix} + \begin{bmatrix}\boldsymbol{\varepsilon} - \boldsymbol{y}\\ \boldsymbol{\varepsilon} + \boldsymbol{y}\end{bmatrix}^T\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{bmatrix} \tag{13}$$

$$\text{subject to: } \begin{bmatrix}\mathbf{1}\\ -\mathbf{1}^*\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{bmatrix} = \begin{bmatrix}\mathbf{0}\\ \mathbf{0}\end{bmatrix}, \alpha, \alpha^* \leq 0. \tag{14}$$

where α and α^* are Lagrange multipliers, $\tilde{\boldsymbol{K}} = \boldsymbol{K} + (1/C)\boldsymbol{I}$ is a modified Gram matrix, $\boldsymbol{K}$ is the original Gram matrix, with its (i, j)-th components is $K_{ij} = k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}; \boldsymbol{\theta})$, and $\boldsymbol{I}$ is an $n \times n$ identity matrix. The set of hyperparameters in SVR includes C, $\boldsymbol{\theta}$, and ε, which need to be optimized to maximize the predictive power of SVR. This paper uses the squared-exponential function as the kernel for SVR.

C. Using a surrogate model in practice

The surrogate model can then be deployed to serve engineering purposes, such as to aid optimization, uncertainty analysis, or reliability analysis. For example, the calculation of statistical moments using Monte Carlo simulation in uncertainty analysis now only needs to call $\hat{f}(\boldsymbol{x})$ in lieu of the actual function. The surrogate model, particularly

GPR, is also helpful in Bayesian optimization that performs simultaneous exploration and exploitation of the design space [30, 31]. Finally, a surrogate model can uncover useful patterns or insights as a cheap approximation of the true relationship. Unfortunately, such information is often hidden by the highly complex expression of the model. Thus, an additional tool is needed to create a curtain to take a peek inside the model.

Most of the explanation tools in the interpretable machine learning literature are model-agnostic. Therefore, any surrogate model is explicable through the application of such tools. Although kernel-based methods are versatile and capable of approximating highly non-linear functions, they are not directly interpretable to humans. The next section explains SHAP, which acts as a post hoc explanation technique to dissect the surrogate model.

III. Shapley Additive Values for Surrogate Modeling

A. Preliminaries

Define $[1:m] := \{1, 2, \ldots, m\}$ and the corresponding subset u as $u \subseteq [1:m]$. Let us also define $\{-u\} = [1:m] \setminus u$ as the complement of u. Further, for an index u, denote Ω_u as the subset of the $\boldsymbol{\Omega}$ where $\Omega_u = \prod_{i \in u} \Omega_i$. Such a definition is important because we need to calculate the model's prediction when only the variables indexed in u are involved.

Using the language of function decomposition, the prediction of a surrogate model can be seen as the sum of the contributions of individual variables and their interactions (or, using game theory jargon, "coalitions"):

$$\hat{f}(\boldsymbol{x}) = \sum_{u \subseteq [1:m]} \hat{f}_u(\boldsymbol{x}_u), \tag{15}$$

where $u \subseteq [1:m]$ includes both the empty set $\emptyset$ and all non-empty subsets, and $\hat{f}_u(\boldsymbol{x}_u)$ is the prediction of the model using the subset of variables as indexed in u.

B. Shapley values

Define a "game" with m players and a value function val(.) that returns a real value to the individual players or their combinations thereof. All possible coalitions between players are defined in the subset of $[1:m]$. Let us define $u \subseteq [1:m]$ as the subset of input variables and the corresponding value function as val(u). The Shapley value for a player j is written as follows:

$$\phi_j = \frac{1}{m} \sum_{u \subseteq \{-j\}} \binom{m-1}{|u|}^{-1} (\text{val}(u \cup \{j\}) - \text{val}(u)). \tag{16}$$

Consider a coalition u that excludes j. The definition of $(\text{val}(u \cup \{j\})$ is the gain obtained when u and j are involved in the coalition. It is then easy to see that the term inside the bracket on the right-hand side of the equation is the marginal

contribution of player j to the specific coalition u. Notice that the marginal contribution is calculated for all possible coalitions $u \subseteq \{-j\}$, for the marginal contributions to be summed out after being multiplied by the weight. Eq.(16) can be interpreted as follows. The Shapley value for a player j is defined as the sum of the marginal contribution of player j to all possible coalitions divided by the number of coalitions excluding j of this size. The sum is then divided again by the number of players to obtain the Shapley value of player j.

One important and desirable property of Shapley value is efficiency, which states that the sum of Shapley values for all players equals the value of the grand coalition (i.e., all variables are involved) val($[1 : m]$). The Shapley value can then be interpreted as the individual contributions of the players to the grand coalition. Notice that we have not defined the "game" and the "player." Let us now put it into a context; that is, the "game" is the prediction of a surrogate model, and the "players" are the input variables. The value of the grand condition is then the prediction of the surrogate model, that is, val($[1 : m]$) = $\hat{f}(\boldsymbol{x})$. The following section explains SHAP, which uses the principle of Shapley value to explain a supervised machine learning model.

C. SHAP

There are several methods that use the principle of Shapley values (e.g., Shapley effects in GSA [32, 33]).Of interest in this paper is the SHAP that decomposes the prediction of an interpolation/prediction at any instance, in contrast to Shapley effects that work globally. Therefore, SHAP can explain why a machine learning model makes a certain prediction for any point in the input space. In addition, SHAP can also explain the model on a more global scale by using the information from multiple instances.

Let us formally define SHAP directly within the context of a predictive model. Consider a single vector of input $\boldsymbol{x}$ and the corresponding prediction of the model, $\hat{f}(\boldsymbol{x})$. The following decomposition is the fundamental building block of SHAP:

$$\hat{f}(\boldsymbol{x}) = \phi_0 + \sum_{j=1}^{m} \phi_j(\boldsymbol{x}), \tag{17}$$

where ϕ_0 is the prediction of the model without involving any variables (i.e., empty set), and ϕ_j is the contribution of the j-th input to the prediction at $\boldsymbol{x}$. It is common to define $\phi_0 = \mathbb{E}[\hat{f}(\boldsymbol{x})]$, or as the mean of the responses of the training set/experimental design. From a pragmatical viewpoint, ϕ_0 is the prediction made without any information regarding the input variables. The sum of the Shapley values (i.e., SHAP) equals the difference between $\hat{f}(\boldsymbol{x})$ and ϕ_0. In this sense, ϕ_j then represents the contribution of the input j to the main prediction. A variable that contributes positively to the mean will have a positive SHAP value and vice versa.

Using Eq. (16), the calculation of SHAP for an arbitrary point $\boldsymbol{x}$ in the input space is written as

$$\phi_j(\boldsymbol{x}) = \frac{1}{m} \sum_{u \subseteq \{-j\}} \binom{m-1}{|u|}^{-1} \left(\hat{f}_{u \cup \{j\}}(\boldsymbol{x}_{u \cup \{j\}}) - \hat{f}_u(\boldsymbol{x}_u) \right), \tag{18}$$

where for a coalition u, $\hat{f}_{u\cup\{j\}}(\boldsymbol{x}_{u\cup\{j\}})$ is the prediction of the model involving all variables in the index $u \cup \{j\}$, while $\hat{f}_u(\boldsymbol{x}_u)$ involves only the variables indexed in u. In this regard, the term inside the bracket on the right-hand side of Eq.(18) is the marginal contribution of variable j to the model's prediction on the coalition u (which excludes j).

Although the formulation looks simple, the calculation of SHAP is time-consuming since it is necessary to construct 2^m models. Such a requirement comes from the necessity to calculate the value function for all possible permutations. Therefore, the SHAP is usually estimated using approximation techniques, most notably KernelSHAP, which fits a weighted local linear model at the query point [15].

D. Aggregated SHAP values

A metric for global sensitivity is obtained by averaging the absolute SHAP values, written as

$$\text{Avg }(\phi) = \frac{1}{n_{mcs}} \sum_{i=1}^{n_{mcs}} \phi(\boldsymbol{x}^{(i)}) \tag{19}$$

where n_{mcs} is the number of random samples for calculating Avg (ϕ). It is a common practice to compute the averaged SHAP values using only the training data set. For design optimization and uncertainty analysis purposes, it is preferable to use as many samples as possible to compute Avg (ϕ). To that end, we generate an external set $\mathcal{X}_{mcs} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n_{mcs})}\}$, where $n_{mcs} \ggg n$ to estimate Avg.(ϕ). The distribution of $\mathcal{X}_{mcs}$ is the same as that of the training set (i.e., $\boldsymbol{\Omega}$).

In machine learning literature, SHAP is commonly used together with tree-based models, e.g., random forest or gradient boosting. However, such models are not suitable for simulation-based engineering problems. The reason why techniques such as GPR and SVR are preferred is that they provide smooth predictions. It is correct that SHAP is unsuitable if there is a strong dependency between the input variables [34]. Fortunately, most engineering design problems feature a set of independent variables.

E. Presenting SHAP values

There are several ways to present and visualize SHAP values to users. The information presented depends on how the SHAP values are visualized. In this regard, SHAP can be presented for a single prediction or at a more global level by showing the values for multiple predictions. However, in this paper, there is little interest in the SHAP values for a single prediction. Instead, we have more interest in displaying the information at a single predictor (i.e., input variable) or a more global level. First, one can compute and show the averaged SHAP values to determine the most important variables. Although it is a common practice to calculate the averaged SHAP values using only the training samples, we recommend using an external sampling set consisting of a large amount of samples for such a purpose. The reason is that the limit and the upper bounds of the input variables, or the distributions, are typically known in the first place.

The SHAP summary plot depicts multiple useful information in a single plot. Notice that variables in the summary plot are usually sorted according to their importance as measured by the averaged SHAP values, in which the uppermost variable in the plot is the most important one. The correlation between input variables and the corresponding SHAP values is visible in the summary plot. However, it is better to visualize such information in a single dependent plot. The SHAP summary plot is primarily useful to visualize the impact of all variables on the output simultaneously. Nevertheless, observing the interaction and degree of nonlinearity from the SHAP summary plot is relatively difficult, which is where independent dependence plots are more useful.

The SHAP values for one particular variable can be shown with respect to the input value of the variable in the form of a scatter plot. Such a dependency plot is useful since it reveals multiple information regarding that particular variable in a single plot, including the level of nonlinearity and possible interactions with other variables. Interactions appear as scattered dots for a particular value of the input variable. The dots in SHAP dependence plot are usually colored by the input values of the other variable. The coloring will show a noticeable pattern if there is a strong interaction between the two variables.

In the next section, the functionality of SHAP visualization is demonstrated in multiple forms on various engineering problems. The SHAP is compared with the Morris elementary effect (EE) method [35, 36]. Morris' method yields the mean and standard deviation of the elementary effect, which gives information on the global sensitivity and nonlinearity to the user. In this paper, we use UQLab to calculate the Morris EE [37].

IV. Computational Results and Discussions

A. Wing weight function

The wing weight problem is an analytical function that models a light aircraft wing. The output of interest is the wing's weight which is affected by ten input variables listed in Table 1. The expression for the wing weight function reads as

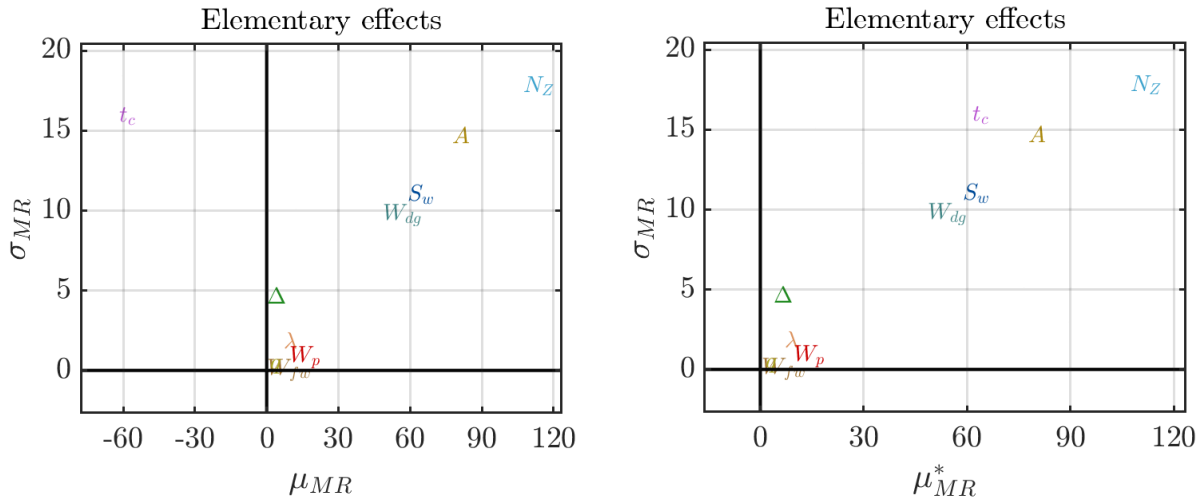
$$f(\boldsymbol{x}) = 0.036 S_w^{0.758} W_{fw}^{0.0035} \left(\frac{A}{cos^2(\Delta)}\right)^{0.6} q^{0.006} \lambda^{0.04} \left(\frac{100 t_c}{cos(\Delta)}\right)^{-0.3} (N_z W_{dg})^{0.49} + S_w W_p. \tag{20}$$

It is difficult to guess the true complexity of the problem by looking at Eq.(20). However, there is an impression that Eq.(20) is highly nonlinear, given the presence of power and trigonometric terms. An SVR model with $n = 500$ is built to fit the function. The model yields $R^2 = 1$ and $RMSE = 7.3 \times 10^{-3}$, which accuracy is more than sufficient for knowledge discovery.

The result from EE analysis is shown in Fig. 1. Shown in these figures are the mean and standard deviation of the distributions, denoted as μ_{MR} and σ_{MR}, respectively. Also shown is the mean of the distribution of the absolute values μ^*_{MR}, which is useful for analyzing the strength of the input variables. Notice that the EE information is extracted from the SVR model and not directly from the wing-weight function. As expected, the EE method does not fully reveal

Table 1 Variables used in the wing weight function.

Variables	Variable Name (unit)	[lower, upper bound]
S_w	Wing area (ft^2)	[150, 200]
W_{fw}	Weight of fuel in the wing (lb)	[220, 300]
A	Aspect ratio	[6, 10]
Δ	Quarter-chord sweep (degrees)	[-10, 10]
q	Dynamic pressure at cruise (lb/ft^2)	[16, 45]
λ	Taper ratio	[0.5, 1]
t_c	Airfoil thickness to chord ratio	[0.08, 0.18]
N_z	Ultimate load factor	[2.5, 6]
W_{dg}	Flight design gross weight (lb)	[1700, 2500]
W_p	Paint weight (lb/ft^2)	[0.025, 0.08]



(a) μ_{MR} **vs** σ_{MR} **(b)** μ^*_{MR} **vs** σ_{MR}

Fig. 1 Summary plots of Morris' elementary effects for the wing weight function.

the nonlinearity of the wing-weight function. According to sample means, EE shows that N_z, A, t_c, S_w, and W_{dg} are the five variables that contribute most to the aircraft's weight. EE also indicates that N_Z, A, and t_c have the highest sample standard deviation among the ten variables, indicating strong nonlinearity and/or interactions. EE also correctly identified that Δ, λ, W_p, W_{fw} and q contribute only a little to the weight function. The airfoil thickness-to-chord ratio has negative μ_{MR}, indicating that increasing t/c leads to decreased wing weight (due to heavier internal structure). Regardless, it is difficult to distinguish which one is most dominant from EE: nonlinearity or interactions. Therefore, it can be seen that EE only shows partial information to the user. EE is useful on its own, but its capability to reveal information is relatively limited.

The averaged SHAP values calculated from $\mathcal{X}_{mcs}$ (10000 samples) are shown in Fig. 2. The averaged SHAP values

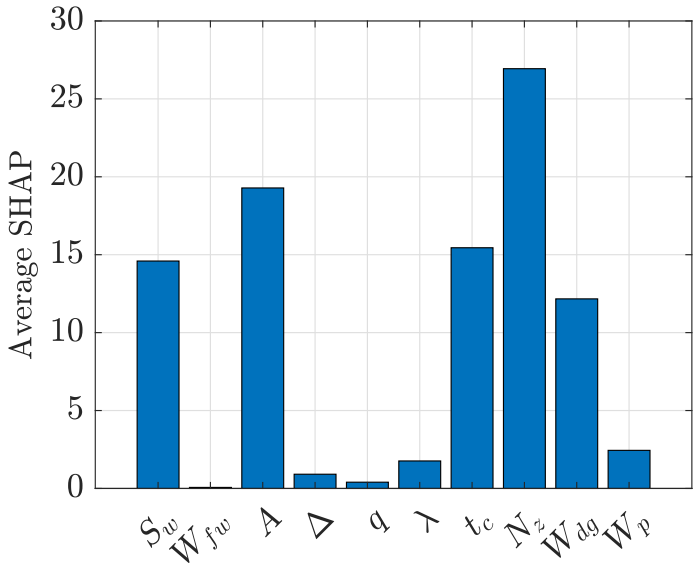
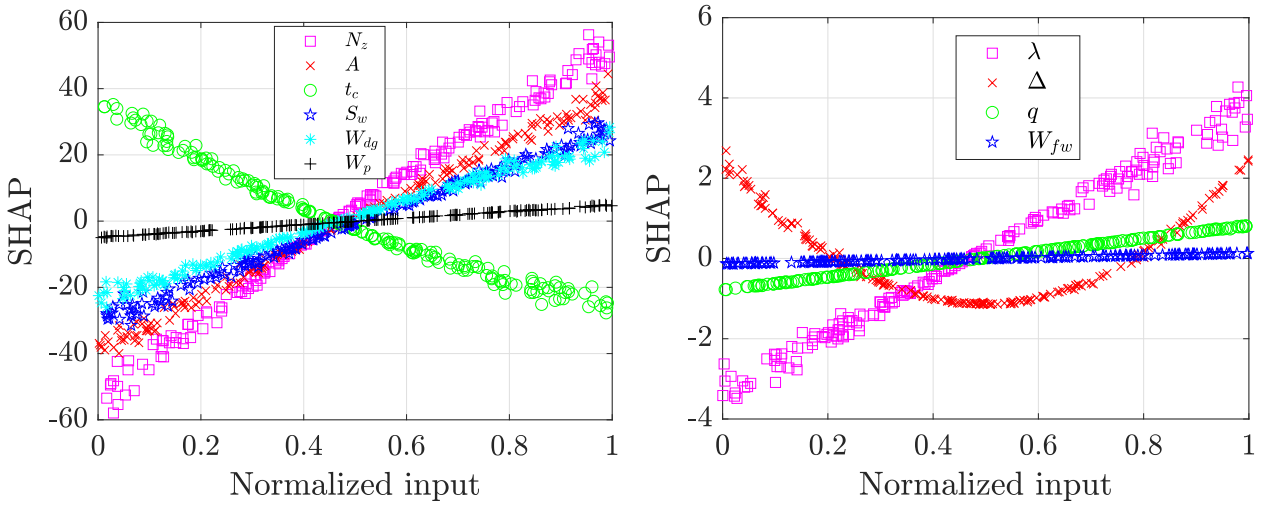


Fig. 2 Averaged SHAP values estimated from the SVR model for the wing weight function.



(a) Six most important variables **(b) Four least important variables**

Fig. 3 SHAP dependence plots for the wing weight function. The left and right figures show the six most important and the four less influential variables, respectively. Notice the different scales on the y-axis.

indicate that N_z, A, t_c, S_w, and W_{dg} significantly dominate the change in the wing-weight function. On the other hand, the impact of the other variables is much smaller. In terms of the input importance, the observation from SHAP agrees well with Morris' method. However, EE only informs whether the input non-linearly affects the output or interacts with other variables without any information regarding the level of nonlinearity and interaction.

Most importantly, SHAP can simultaneously display the non-linearity and interaction in single or multiple plots. Fig. 3 depicts the SHAP dependence plots for the wing weight function. The inputs are shown in their normalized values (i.e., $[0, 1]$) to allow simultaneous visualization of multiple variables within a single plot. Fig. 3a shows the

six most influential variables, while Fig. 3b is for the rest of the variables, which are significantly less influential than the others. Only 200 samples are shown to avoid too cluttered plots. It can be seen from Fig. 3 that the impact of all variables on the weight is mostly linear or slightly nonlinear at best. The only exception is the quarter-chord sweep (Δ) which nonlinearly affects the wing weight in a quadratic fashion, but its contribution is relatively insignificant.

The interactions between input variables are small and unimportant since the SHAP values are slightly scattered for only a few variables. For example, the scattered values in the SHAP for N_z indicates the slight nonlinear effect or interactions. Interestingly, SHAP uncovers that the wing-weight function is relatively simple, despite the complex expression in Eq.(20). The expression in Eq.(20) is also not easily decipherable by humans and might give a wrong impression of the complexity, while visualization from SHAP is easier to comprehend. SHAP also shows the correlation between input variables and the wing's weight. Variables N_z, A, S_w, and W_{dg} positively correlate with the wing's weight, which makes sense. In addition, the SHAP plot also reveals that all variables but Δ monotonously affect the output of interest. The knowledge obtained from SHAP analysis matches the actual knowledge regarding the wing weight function. For example, keeping the wing area small is important to avoid an excessively heavy wing. To take another example, it also makes sense the paint weight W_p do not significantly affect the weight, compared to the flight design gross weight W_{dg}.

B. Piston simulation function

The next problem is a simple analytical piston simulation function with seven input variables [38]. The problem is widely used in UQ and GSA literature to benchmark various methods. The piston simulation function describes the circular motion of a piston within a cylinder, with the output of interest being cycle time C (the time for completing one cycle) in seconds. There are seven input variables for this problem as listed in Table 2. The equation for the piston simulation function is written as

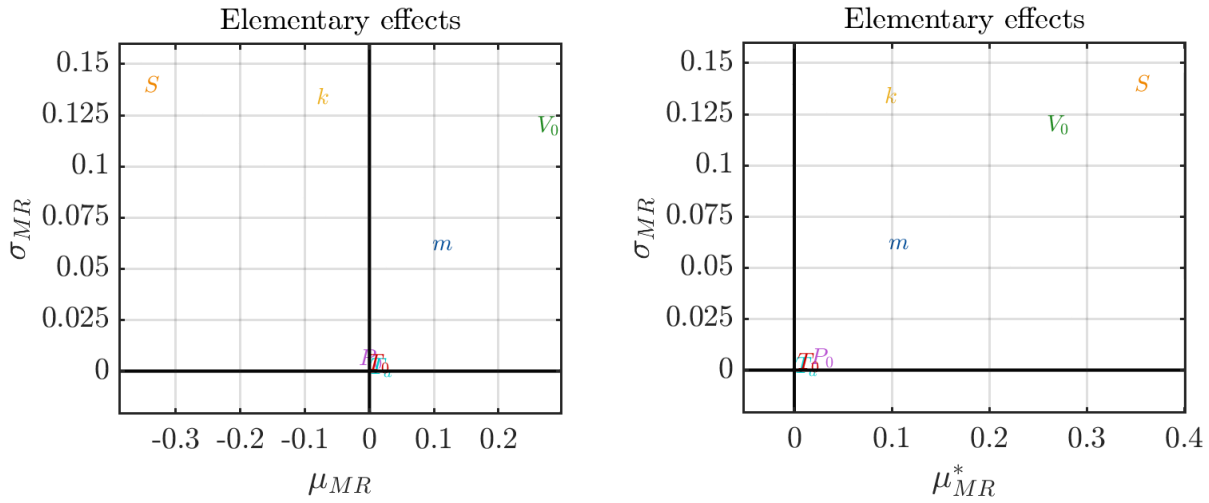
$$\begin{aligned} C(\boldsymbol{x}) &= 2\pi\sqrt{\frac{M}{k + S^2\frac{P_0V_0}{T_0}\frac{T_a}{V^2}}}, \text{ where} \\ V &= \frac{S}{2k}\left(\sqrt{A^2 + 4k\frac{P_0V_0}{T_0}T_a} - A\right) \\ A &= P_0S + 19.62M - \frac{kV_0}{S} \end{aligned} \tag{21}$$

As one can see from Eq.(21), the problem involves a chain of nonlinear functions. The first impression when seeing Eq.(21) is that the function is highly nonlinear. For this problem, a GPR model with Gaussian kernel and $n = 500$ samples is fitted. The accuracy of the built SVR is $R^2 = 0.999$ and $NRMSE = 0.0066$, which is already highly accurate for the model to be reliable in terms of the extracted knowledge.

The summary plots of the Morris' EE for the piston simulation function are shown in Fig. 4. The plots identified that the strongest variable is S, followed by V_0, m, and k, with the contribution of the rest of the variables are not significant.

Table 2 Variables used in the piston simulation function

Variables	[lower, upper bound]
Piston weight (m)	[30,60] kg
Piston surface area (S)	[0.005,0.020] m^2
Initial gas volume (V_0)	[0.002, 0.010] m^3
Spring coefficient (k)	[1000, 5000] N/m
Atmospheric pressure (P_0)	[90000, 110000] $\frac{N}{m^2}$
Ambient temperature (T_a)	[290, 296] K
Filling gas temperature (T_0)	[340, 360] K



(a) μ_{MR} **vs** σ_{MR} **(b)** μ^*_{MR} **vs** σ_{MR}

Fig. 4 Summary plots of Morris' elementary effects for the piston simulation function.

Increasing S and k tends to decrease the cycle time, while it is the opposite for the other variables. We can also see that k and m have similar values of μ^*_{MR}, but the former has a higher standard deviation. This indicates that the effect of k is more nonlinear than m, or k interacts stronger with other variables. Again, EE does not reveal how the inputs specifically affect the behavior of the cycle time.

The averaged SHAP values for the piston simulation function are shown in Fig. 5. It can be seen that the most important variable is piston surface area, followed by initial gas volume, piston weight, and spring coefficient (the ranking is similar to that of the EE). The impact of the other three variables are significantly less than the first four variables. Fig. 6 depicts the SHAP dependence plot for all the seven variables, normalized to $[0, 1]^7$, within a single plot. Starting from the less influential variables, we observe that the effects of the ambient temperature (T_a) and filling gas temperature T_0 are significantly smaller than the other, as evidenced by their extremely small SHAP values. However, it is difficult to observe the non-significance of T_a and T_0 from the piston simulation equation as shown in Eq. (21). It can be seen that the piston weight m and initial gas volume V_0 positively correlate with the cycle time, with the increment in

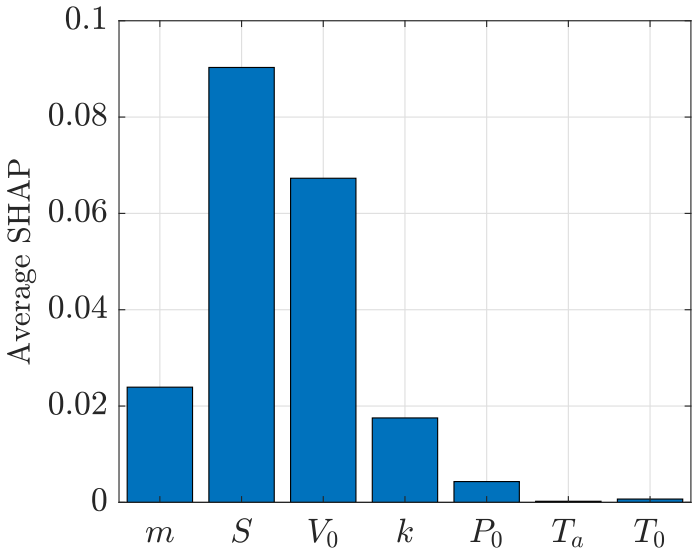


Fig. 5 Average SHAP values estimated from the GPR model for the piston simulation function.

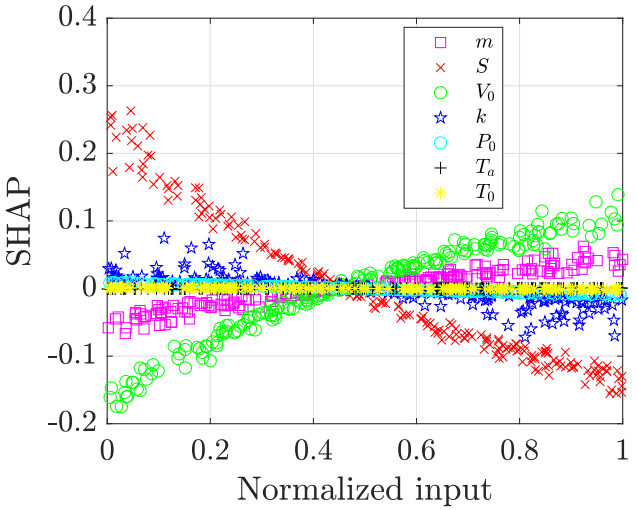


Fig. 6 SHAP dependence plots for the piston simulation extracted from the GPR model.

piston surface S significantly decreasing the cycle time. Another interesting observation is how the spring coefficient k affects the cycle time. The increase in spring coefficient generally decreases the cycle time, but it strongly interacts with the other variables, as seen from the highly dispersed SHAP values. Therefore, the high standard deviation of k is mainly due to the interaction with the other variables rather than nonlinearity. Similarly, the SHAP values of the piston surface area also show evidence of weak interaction with other variables.

C. Re-entry trajectory problem

The last problem is the stochastic re-entry trajectory analysis of an Apollo type with seven input parameters. The problem is an uncertainty quantification problem in which the input parameters are assumed to be normally distributed

as shown in Table 3. We use the falling range, which is defined as the flight distance in the horizontal direction (see Fig. for the schematic), as the output of interest to be approximated. The aerodynamic quantities are calculated using a modified Newtonian impact theory, with the capsule modeled with a mesh of triangle facets. More details on this problem can be found in Tokunaga et al. [39].

Table 3 Stochastic input variables used in the re-entry trajectory analysis problem

Variables	Variable names (unit)	[Mean value, Standard deviation]
h	Altitude (m)	[80000,7000]
V	Velocity (m/s)	[7140,200]
α	Attitude angle/angle of attack (deg)	[180, 5]
m	Mass (kg)	[5762, 576]
$\dot{q}$	Pitch rate (deg/)	[0, 20]
I	Moment of inertia (kg.m^2)	[7150, 715]
γ	Path angle (deg)	[15, 5]

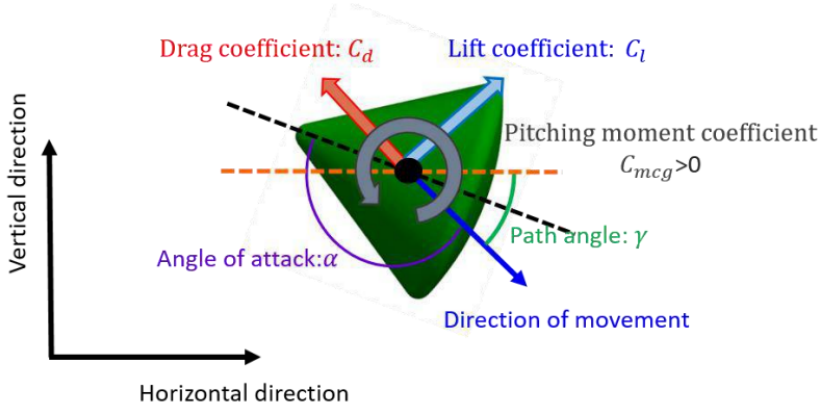


Fig. 7 A schematic depiction of the Apollo-type capsule re-entry trajectory analysis problem [39].

A GPR model with 1000 samples was constructed to extract the SHAP and EE values. Despite the large sample size, we found the coefficient of determination between the prediction and the actual falling range is only 0.952 (we expected around 0.99). However, the obtained accuracy is deemed sufficient for SHAP and EE analysis to uncover the general trend. To further verify this, the histograms of the falling range from the actual simulation and the GPR model closely resemble each other, giving us confidence in the extracted SHAP values.

First, the Morris' EE summary plots are shown in Fig. 9. The significantly strong contribution of the attitude angle α is obvious from the summary plot. Moreover, the EE summary plots also reveal that the overall impact of attitude angle is decreasing the falling range. Notice that the EE summary plots give the impression that increasing all the other variables lead to an increased falling range. However, as shown in the SHAP analysis soon, this turns out to be incorrect for the path angle γ.

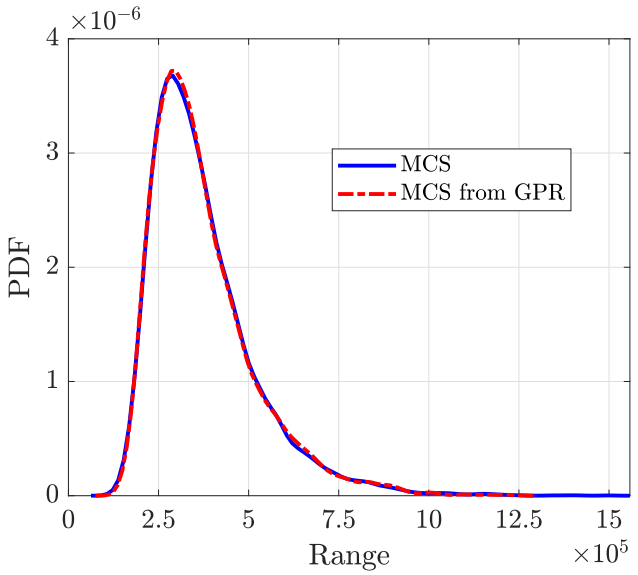
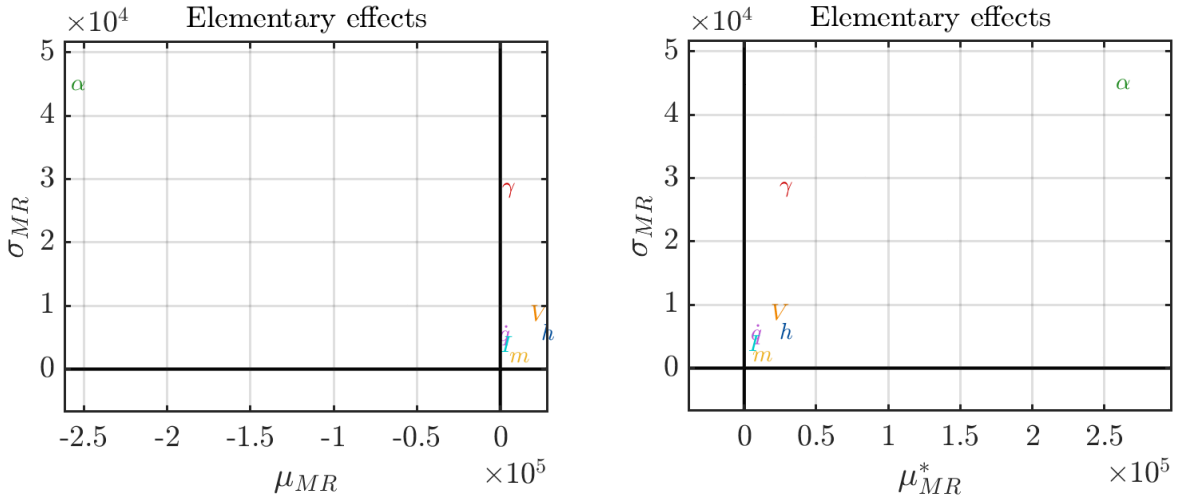


Fig. 8 Histograms of falling range from the actual simulation and the GPR model



(a) μ_{MR} **vs** σ_{MR} **(b)** μ^*_{MR} **vs** σ_{MR}

Fig. 9 Summary plots of Morris' elementary effects for the re-entry trajectory analysis problem.

SHAP proved to be useful for performing knowledge discovery in the context of this problem because we obtained additional useful information. The computed averaged SHAP values are shown in Fig. 10. The result shows that the attitude angle plays a prominent role in determining the falling range, followed by the path angle. The next important variables are altitude h and velocity V, with the remaining variables contributing little to the variation in the falling range. The SHAP dependence plot shows clear trends (see Fig. 11), especially how the attitude angle affects the falling range. That is, the increase in attitude angle decreases the falling range, with a slight but visible nonlinear trend. The SHAP dependence plot for the path angle shows that increasing and decreasing the path angle from its nominal value leads to a decrease in the falling range (although the impact is less significant than that of the attitude angle); It is worth

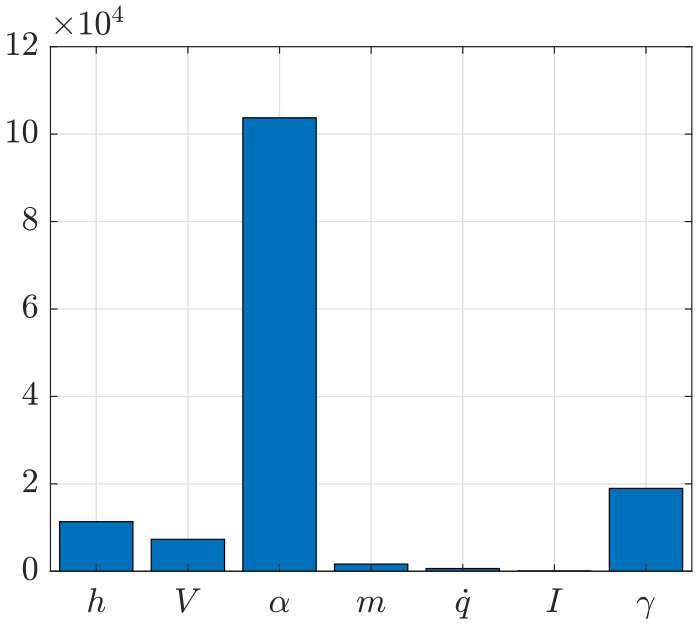
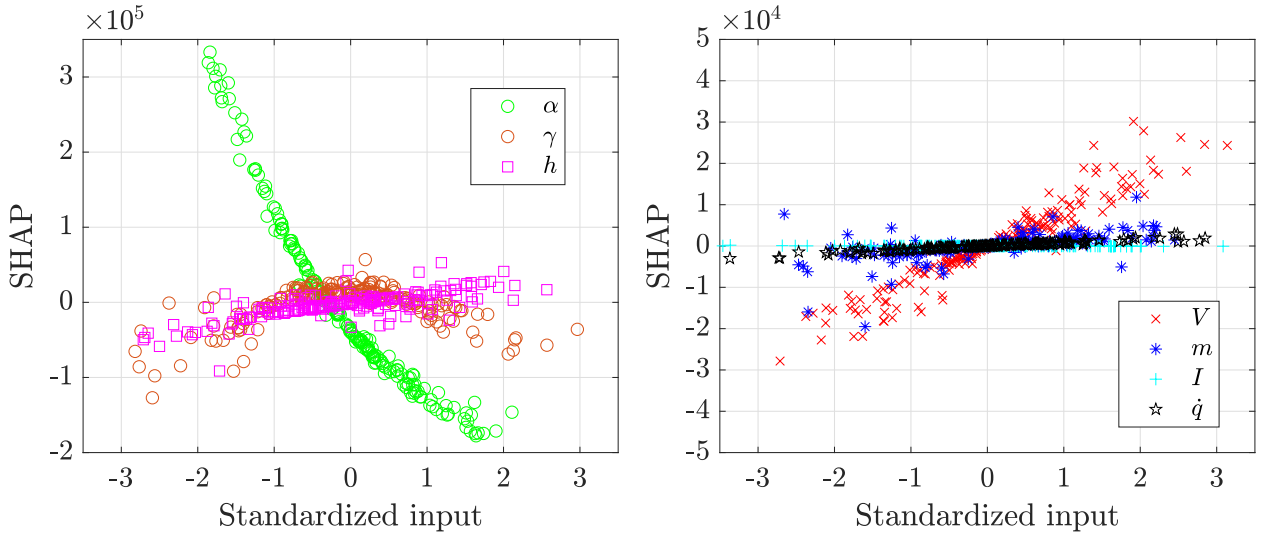


Fig. 10 Averaged SHAP values estimated from the GPR model for the re-entry trajectory analysis problem.



(a) Three most influential variables **(b) Four less influential variables**

Fig. 11 SHAP dependence plots for the re-entry trajectory analysis problem. The left and right figures show the three most influential variables and the four less influential variables, respectively. Notice the different scales on the y-axis and the standardized x-scale.

noting that such a variation is not visible from the EE summary plot. It can also be observed that the increase in altitude h and velocity V yields a higher falling range. As for the mass, increasing the mass leads to only an insignificant increase in the falling range. Interactions between variables exist, primarily those between the three most influential variables (namely, α, γ, and h). Lastly, the effect of pitch rate $\dot{q}$ and moment of inertia I on the falling range is negligible.

SHAP clearly shows its advantage over Morris' EE analysis on the re-entry trajectory problem. Although Morris' EE

analysis is still useful on its own to provide a quick interpretation of the result (especially to show the relative strength of the input variables), it does not reveal important trends, such as the non-monotonous relationship between the path angle and the falling angle.

V. Conclusions and future works

This paper demonstrates the application of SHAP with surrogate modeling for aiding visualization and analyzing input-output relationships in an informative way. Despite its wide application in a general machine learning context, the use of SHAP is still not widespread in the engineering domain. In this paper, we demonstrate the applicability of SHAP for helping data-driven engineering analysis and design. SHAP allows users to observe and see several important trends intuitively, which is useful for exploratory analysis in the context of design, optimization, or uncertainty analysis.

The usefulness of SHAP as a tool to explain surrogate models is demonstrated on a ten-variable wing-weight function, seven-variable piston simulation function, and seven-variable re-entry trajectory analysis problem. The results show that SHAP can uncover and visualize how the inputs change the output of interest, which greatly aids engineers in gaining important insight into the problem. In addition, SHAP also shows the magnitude of importance and how the input variables specifically interact with each other. We contrasted SHAP with Morris' elementary effect, in which the latter only gives information on the presence of nonlinearity and interaction without further detailed information. SHAP is also a versatile post-processing tool because it is a model-agnostic method; that is, it can be used with any surrogate model.

The KernelSHAP algorithm can still be slow, especially for a large sample size. For future works, analytical or fast calculation of SHAP is necessary to accelerate the analysis problem (especially in the context of the GPR and SVR model). Another interesting research direction is to compare SHAP with other interpretability techniques or established methods in global sensitivity analysis, e.g., Sobol indices. Finally, studying SHAP for individual points in multi-objective optimization is also one potential future research direction.

Acknowledgement

Pramudita Satria Palar and Lavi Rizki Zuhal were funded partly through the Program Penelitian, Pengabdian kepada Masyarakat dan Inovasi ITB (P2MI) administered by Institut Teknologi Bandung. Part of the work was also carried out under the Collaborative Research Project 2022 of the Institute of Fluid Science, Tohoku University.

References

[1] Iooss, B., and Lemaître, P., "A review on global sensitivity analysis methods," *Uncertainty management in simulation-optimization of complex systems*, Springer, 2015, pp. 101–122.

[2] Borgonovo, E., and Plischke, E., "Sensitivity analysis: a review of recent advances," *European Journal of Operational Research*,

Vol. 248, No. 3, 2016, pp. 869–887.

[3] Sobol, I. M., "Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates," *Mathematics and computers in simulation*, Vol. 55, No. 1-3, 2001, pp. 271–280.

[4] Piano, S. L., Ferretti, F., Puy, A., Albrecht, D., and Saltelli, A., "Variance-based sensitivity analysis: The quest for better estimators and designs between explorativity and economy," *Reliability Engineering & System Safety*, Vol. 206, 2021, p. 107300.

[5] Xiu, D., and Karniadakis, G. E., "The Wiener–Askey polynomial chaos for stochastic differential equations," *SIAM journal on scientific computing*, Vol. 24, No. 2, 2002, pp. 619–644.

[6] Sudret, B., "Global sensitivity analysis using polynomial chaos expansions," *Reliability engineering & system safety*, Vol. 93, No. 7, 2008, pp. 964–979.

[7] Cheng, K., Lu, Z., Zhou, Y., Shi, Y., and Wei, Y., "Global sensitivity analysis using support vector regression," *Applied Mathematical Modelling*, Vol. 49, 2017, pp. 587–598.

[8] Lu, C., Feng, Y.-W., Liem, R. P., and Fei, C.-W., "Improved Kriging with extremum response surface method for structural dynamic reliability and sensitivity analyses," *Aerospace Science and Technology*, Vol. 76, 2018, pp. 164–175.

[9] Cheng, K., Lu, Z., Ling, C., and Zhou, S., "Surrogate-assisted global sensitivity analysis: an overview," *Structural and Multidisciplinary Optimization*, Vol. 61, No. 3, 2020, pp. 1187–1213.

[10] Roscher, R., Bohn, B., Duarte, M. F., and Garcke, J., "Explainable machine learning for scientific insights and discoveries," *Ieee Access*, Vol. 8, 2020, pp. 42200–42216.

[11] Burkart, N., and Huber, M. F., "A survey on the explainability of supervised machine learning," *Journal of Artificial Intelligence Research*, Vol. 70, 2021, pp. 245–317.

[12] Friedman, J. H., "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, 2001, pp. 1189–1232.

[13] Goldstein, A., Kapelner, A., Bleich, J., and Pitkin, E., "Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation," *journal of Computational and Graphical Statistics*, Vol. 24, No. 1, 2015, pp. 44–65.

[14] Ribeiro, M. T., Singh, S., and Guestrin, C., ""Why should i trust you?" Explaining the predictions of any classifier," *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[15] Lundberg, S. M., and Lee, S.-I., "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, Vol. 30, 2017.

[16] Shapley, L. S., *17. A value for n-person games*, Princeton University Press, 1953.

[17] Mangalathu, S., Hwang, S.-H., and Jeon, J.-S., "Failure mode and effects analysis of RC members based on machine-learning-based SHapley Additive exPlanations (SHAP) approach," *Engineering Structures*, Vol. 219, 2020, p. 110927.

[18] Feng, D.-C., Wang, W.-J., Mangalathu, S., and Taciroglu, E., "Interpretable XGBoost-SHAP machine-learning model for shear strength prediction of squat RC walls," *Journal of Structural Engineering*, Vol. 147, No. 11, 2021, p. 04021173.

[19] Obayashi, S., Jeong, S.-K., Shimoyama, K., Chiba, K., and Morino, H., "Multi-objective design exploration and its applications," *International Journal of Aeronautical and Space Sciences*, Vol. 11, No. 4, 2010, pp. 247–265.

[20] Nagar, D., Ramu, P., and Deb, K., "Visualization and analysis of Pareto-optimal fronts using interpretable self-organizing map (iSOM)," *Swarm and Evolutionary Computation*, 2022, p. 101202.

[21] Constantine, P. G., Dow, E., and Wang, Q., "Active subspace methods in theory and practice: applications to kriging surfaces," *SIAM Journal on Scientific Computing*, Vol. 36, No. 4, 2014, pp. A1500–A1524.

[22] Zhou, C., Shi, Z., Kucherenko, S., and Zhao, H., "A unified approach for global sensitivity analysis based on active subspace and Kriging," *Reliability Engineering & System Safety*, Vol. 217, 2022, p. 108080.

[23] Oyama, A., Nonomura, T., and Fujii, K., "Data mining of Pareto-optimal transonic airfoil shapes using proper orthogonal decomposition," *Journal of Aircraft*, Vol. 47, No. 5, 2010, pp. 1756–1762.

[24] Rasmussen, C. E., "Gaussian processes in machine learning," *Summer school on machine learning*, Springer, 2003, pp. 63–71.

[25] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and analysis of computer experiments," *Statistical science*, Vol. 4, No. 4, 1989, pp. 409–423.

[26] Drucker, H., Burges, C. J., Kaufman, L., Smola, A., and Vapnik, V., "Support vector regression machines," *Advances in neural information processing systems*, Vol. 9, 1996.

[27] Smola, A. J., and Schölkopf, B., "A tutorial on support vector regression," *Statistics and computing*, Vol. 14, No. 3, 2004, pp. 199–222.

[28] Schölkopf, B., Smola, A. J., Bach, F., et al., *Learning with kernels: support vector machines, regularization, optimization, and beyond*, MIT press, 2002.

[29] Hansen, N., Müller, S. D., and Koumoutsakos, P., "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evolutionary computation*, Vol. 11, No. 1, 2003, pp. 1–18.

[30] Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and De Freitas, N., "Taking the human out of the loop: A review of Bayesian optimization," *Proceedings of the IEEE*, Vol. 104, No. 1, 2015, pp. 148–175.

[31] Zhan, D., and Xing, H., "Expected improvement for expensive optimization: a review," *Journal of Global Optimization*, Vol. 78, No. 3, 2020, pp. 507–544.

[32] Song, E., Nelson, B. L., and Staum, J., "Shapley effects for global sensitivity analysis: Theory and computation," *SIAM/ASA Journal on Uncertainty Quantification*, Vol. 4, No. 1, 2016, pp. 1060–1083.

[33] Plischke, E., Rabitti, G., and Borgonovo, E., "Computing Shapley effects for sensitivity analysis," *SIAM/ASA Journal on Uncertainty Quantification*, Vol. 9, No. 4, 2021, pp. 1411–1437.

[34] Aas, K., Jullum, M., and Løland, A., "Explaining individual predictions when features are dependent: More accurate approximations to Shapley values," *Artificial Intelligence*, Vol. 298, 2021, p. 103502.

[35] Morris, M. D., "Factorial sampling plans for preliminary computational experiments," *Technometrics*, Vol. 33, No. 2, 1991, pp. 161–174.

[36] Campolongo, F., Cariboni, J., and Saltelli, A., "An effective screening design for sensitivity analysis of large models," *Environmental modelling & software*, Vol. 22, No. 10, 2007, pp. 1509–1518.

[37] Marelli, S., and Sudret, B., *UQLab: A framework for uncertainty quantification in Matlab*, American Society of Civil Engineers, 2014.

[38] Ben-Ari, E. N., and Steinberg, D. M., "Modeling data from computer experiments: an empirical comparison of kriging with MARS and projection pursuit regression," *Quality Engineering*, Vol. 19, No. 4, 2007, pp. 327–338.

[39] Tokunaga, A., Sotoguchi, A., Shimoyama, K., and Fujimoto, K., "Stochastic re-entry trajectory analysis with uncertain initial conditions for safety assessment," *AIAA Scitech 2019 Forum*, 2019, p. 2235.