Sensitivity-guided iterative parameter identification and data generation with BayesFlow and PELS-VAE for model calibration

Yi Zhang, Lars Mikelsons

Angaben zur Veröffentlichung / Publication details:

Zhang, Yi, and Lars Mikelsons. 2023. “Sensitivity-guided iterative parameter identification and data generation with BayesFlow and PELS-VAE for model calibration.” Advanced Modeling and Simulation in Engineering Sciences 10 (1): 9. https://doi.org/10.1186/s40323-023-00246-y.

Zhang and Mikelsons *Advanced Modeling and Simulation in Engineering Sciences(2023)10:9*
https://doi.org/10.1186/s40323-023-00246-y

Advanced Modeling and Simulation in Engineering Sciences

RESEARCH ARTICLE Open Access

Sensitivity-guided iterative parameter identification and data generation with BayesFlow and PELS-VAE for model calibration

Yi Zhang[1,*] and Lars Mikelsons[1,*]

*Correspondence: yi.zhang@informatik.uni-augsburg.de; lars.mikelsons@informatik.uni-augsburg.de
[1]Chair of Mechatronics, University of Augsburg, Augsburg, Germany

Abstract

Calibration of complex system models with a large number of parameters using standard optimization methods is often extremely time-consuming and not fully automated due to the reliance on all-inclusive expert knowledge. We propose a sensitivity-guided iterative parameter identification and data generation algorithm. The sensitivity analysis replaces manual intervention, the parameter identification is realized by BayesFlow allowing for uncertainty quantification, and the data generation with the physics-enhanced latent space variational autoencoder (PELS-VAE) between two iteration steps enables inference of weakly identifiable parameters. A complete calibration experiment was conducted on a thermal model of an automotive cabin. The average relative error rate over all parameter estimates of 1.62% and the mean absolute error of calibrated model outputs of 0.108 °C validate the feasibility and effectiveness of the method. Moreover, the entire procedure accelerates up to 1 day, whereas the classical calibration method takes more than 1 week.

Keywords: Sensitivity analysis, Weak identification, BayesFlow, PELS-VAE

Introduction

By standard techniques, a model is calibrated by adjusting selected parameters to obtain a best fit between the model response and the measurement [1]. For non-linear models with possibly algebraic loops and weakly identifiable parameters, the adjustment can only be achieved through an iterative and repeated approach. Therefore, the approach is time-consuming when applied to a complex model. Mostly, only a local optimum can be reached. Besides, due to the difficulty in uncertainty quantification in this procedure, it is difficult to determine whether parameter estimates are reliable. In addition, for the selection of parameters of interest, diverse expert knowledge and experience are required, including how each parameter could possibly affect dynamic or static characteristics of the model and the dependency between the parameters.

To overcome the above-mentioned shortcomings of the classical calibration method in terms of time-consumption, demand of manual intervention and reliability, Bayesian

inference methods with deep learning can be applied to estimate the density of parameters from observations and prior knowledge. By means of this, the density estimate is provided with a confidence interval by its nature, which indicates the trustworthiness of parameter identification results. An ensemble learning method integrating stochastic components into neural networks, the Bayesian neural network (BNN) [2] has been proved to fulfill high accuracy and more robustness in out-of-distribution samples [3]. Furthermore, BNNs enable better identification of stochastic dynamical systems [4]. Another expressive probabilistic modeling and inference method, normalizing flow (NF) [5] has been deeply explored and widely applied, as summarized in [6,7]. NFs model a complex target density from a base density (e.g., Gaussian), which is realized by the invertible neural networks (INNs), equipped with structural invertibility and tractable Jacobian computation [8]. In particular, the conditional invertible neural networks (cINNs) were developed for mapping observations of a physical system to unknown input distribution parameters.

Even with deep learning methods, identification of numerous parameters from the observed data of a complex model could be challenging, especially in the presence of weakly identifiable parameters. Primarily, a large dataset is required to train a deep learning model to map the relationship between the observed data and the large number of parameters. Furthermore, the weakly identifiable parameters usually have inadequate sensitivities thus negligible contribution to the gradient, resulting in an ill-posed problem. Theoretically, this can be solved by training the deep learning model on a large enough dataset. However, in practice, both data generation and training are burdensome due to the complexity of the physical model and the resulting deep learning model.

Getting inspiration from the iterative calibration approach, it is also advantageous to sort the parameters in advance under the guidance of the parameter identifiability. Moreover, investigation of parameter identifiability supports a reasonable selection of parameters of interest without the preliminary comprehension of the model. This is commonly accomplished through local or global sensitivity analysis. To specify, parameter identifiability refers to the ability to accurately and uniquely estimate the parameters in a physical model based on the available data, whereas parameter sensitivity measures the influence of changes in parameters on outputs of a physical model. For a problem with large variation in parameter ranges, it's more important to pay attention to global sensitivity [9,10]; while in order to focus on how the variation in a parameter affects the response of a dynamic model, local sensitivity is supposed to be analyzed [11,12]. Sometimes, both local and global sensitivity analyses are necessary to clarify the importance of parameters thoroughly [13,14].

In this paper, we put forward a method combining sensitivity analysis and deep learning based Bayesian inference for parameter identification, namely sensitivity-guided iterative parameter identification and data generation. This method is built up with three fundamental pillars. Firstly, the selection of identifiable parameters is under the guidance of sensitivity analysis in the use of first-order Sobol indices [15]. Secondly, BayesFlow [16] is employed to identify parameters. Thirdly, for the purpose of fast generation of transformed time series from the still unidentified variables, the physics-enhanced latent space variational autoencoder (PELS-VAE) [17] is applied and modified into a Teacher–Student Architecture [18]. Especially, the iterative approach avoids time-consuming preparation of a large training set and further facilitates accurate and reliable identification of param-

eters from the dynamic observations of highly complex physical models step by step, even those which are weakly identifiable.

Preliminaries

In the following, the three parts of the proposed method are summarized. Parameter identifiability in "Parameter identifiability" section assesses the ability of accurately and uniquely estimating parameters from the observations. The identifiability is distinguished by sensitivity analysis in "Sensitivity analysis" section. Based on this, the strongly identifiable parameters can be directly learned by a (NFs)-based Bayesian inference method, namely BayesFlow [16] in "BayesFlow" section; while the weakly identifiable parameters can only be obtained once the strongly identifiable parameters have been determined. Between the inference steps, a data update is required, in which the already identified parameters are set to estimated values, and the observations are transformed as well. This transformation is not calculated through the original physical model simulation, but rather facilitated by a deep generative model, namely Physics-Enhanced Latent Space Variational Autoencoder [17] in "Physics-enhanced latent space variational autoencoder" section.

Parameter identifiability

From the perspective of frequentist statistics, parameter identifiability means that a distinct parameter θ_i yields a distinct probability distribution $\mathbb{P}_{\theta_i}$ of the observed data $\boldsymbol{x}$ [19]. For a partially observable physical system, the observed data is $\boldsymbol{x}(\boldsymbol{\theta}, t)$ with given $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_D]^T$. In general, parameter identifiability is divided into *structural identifiability*, which is related to the model structure but not the observed data, and *practical identifiability*, which is dependent on the amount and quality of the measured data as well [20, 21].

Structural identifiability refers to the unique parameterisation for any given model output, whereas practical identifiability can be defined as the ability to estimate a given set of parameters with an adequate accuracy [22]. A parameter θ_i is structurally non-identifiable, when the observation $\boldsymbol{x}$ does not change if the parameter value alters. This implies the existence of a manifold in parameter space upon which the observation $\boldsymbol{x}$ is unchanged. Re-parameterization of such models has been proved to be efficient and effective to solve the problem by Joubert et al. [23], through substitution of all unidentifiable parameters with new parameters.

A structurally identifiable parameter may still be practically non-identifiable, when the estimate has large confidence intervals with current available measurements. On the one hand, this can result from the insufficient amount of data or low signal-noise-ratio of the measurements, which could be resolved by increasing the amount of data and improving signal processing methods. On the other hand, practical non-identifiability occurs when a parameter is so insensitive in comparison to others that its impact is negligible. Therefore, it's necessary to analyze sensitivity in order to extract information about which parameters have the largest impact on the output.

Sensitivity analysis

In terms of range of variation, sensitivity is categorized as *local* and *global*. Local sensitivity is analyzed to determine the effect of small parameter variations on model output, while global sensitivity is applied to evaluate the influence of parameters within their possible value ranges on the model output. The first-order local sensitivity [24] of parameter θ_i from the observed data $\boldsymbol{\theta}$ is calculated by

$$S_{local,i}(t) = \frac{\partial \boldsymbol{x}(\boldsymbol{\theta}, t)}{\partial \theta_i}. \tag{1}$$

Different from the derivative-based local sensitivity analysis, the global sensitivity analysis is usually based on variance. A commonly used method is Sobol indices [15]. For a parameter θ_i, the first-order Sobol indices are calculated by

$$S_{global,i}(\boldsymbol{x}(t)) = \frac{\mathbb{V}_i}{\mathbb{V}[\boldsymbol{x}(\boldsymbol{\theta}, t)]}, \tag{2a}$$

$$\mathbb{V}_i = \int \mathbb{E}_i^2[\boldsymbol{x}(\theta_i, t)] d\theta_i. \tag{2b}$$

$S_{global,i}$ indicates the contribution of variance due to the parameter θ_i towards the total variance in the observations.

Typically, the statistical metrics including the partial variance $\mathbb{V}_i$ and variance $\mathbb{V}$ is obtained by Quasi Monte Carlo Sampling [25,26], which is faster than the standard Monte Carlo method through variance reduction techniques.

BayesFlow

BayesFlow [16] is a Bayesian inference method based on cINNs [8,27], aiming to learn a global estimator for the probabilistic mapping from observed data $\boldsymbol{x}$ to underlying model parameters $\boldsymbol{\theta}$. The invertibility of (cINNs) is facilitated by (NFs) [5], whereby a simple probability density is implicitly transformed into a complex one by applying a sequence of invertible structures. In this way, the network is used in both directions with the help of a latent variable $\boldsymbol{z_\theta}$ with a simple density. In the forward process, it is trained to map the parameter $\boldsymbol{\theta}$ to $\boldsymbol{z_\theta}$-space under the condition of $\boldsymbol{x}$, whereas in the inverse process, under the same condition $\boldsymbol{x}$, $\boldsymbol{\theta}$ can be inferred from the $\boldsymbol{z_\theta}$-space.

Bayesian inference [28] decides which estimate of posterior $p(\boldsymbol{\theta}|\boldsymbol{x})$ best reproduces the observed data $\boldsymbol{x}$ from the family of likelihoods $p(\boldsymbol{x}|\boldsymbol{\theta})$ with given prior $p(\boldsymbol{\theta})$. With Bayes's rule, the posterior is expressed by

$$p(\boldsymbol{\theta}|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}. \tag{3}$$

The posterior distribution provides the estimates of parameters $\widehat{\boldsymbol{\theta}}$, taken as the mean value, as well as confidence intervals indicating the reliability of the estimate.

In order to avoid information loss through restrictive hand-crafted summary statistics of $\boldsymbol{x}$, BayesFlow applies a separate summary network to extract information from $\boldsymbol{x}$, and a cINN to learn the transformation between the distributions of $\boldsymbol{\theta}$ and the latent variable $\boldsymbol{z_\theta}$. The architecture of BayesFlow is presented in Fig. 1.

The summary network and cINN are optimized jointly via back-propagation by minimizing the Kullback–Leibler divergence [29] between the true and the model induced posterior of $\boldsymbol{\theta}$. Then the network parameters are optimized by

$$\widehat{\boldsymbol{\phi}}, \widehat{\boldsymbol{\psi}} = \underset{\boldsymbol{\phi},\boldsymbol{\psi}}{\operatorname{argmin}} \, \mathbb{E}_{p(\boldsymbol{x})}[\mathbb{KL}(p(\boldsymbol{\theta}|\boldsymbol{x}) \parallel p_{\boldsymbol{\phi},\boldsymbol{\psi}}(\boldsymbol{\theta}|\boldsymbol{x}))]. \tag{4}$$

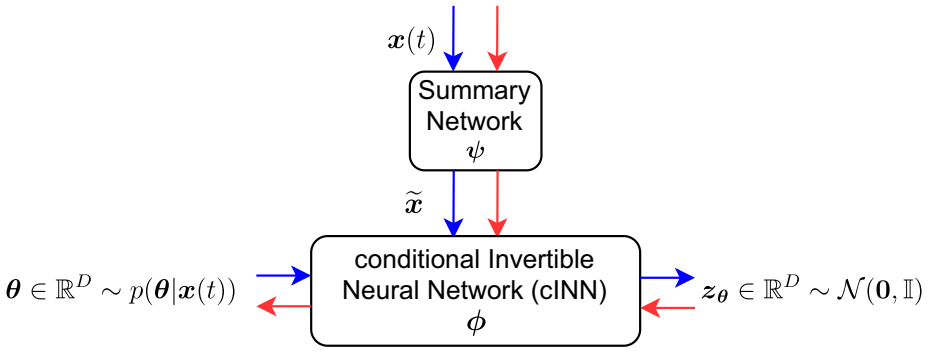


Fig. 1 BayesFlow architecture. $\boldsymbol{\psi}$, $\boldsymbol{\phi}$ denote the network parameters of the summary network and the cINN, respectively. The blue arrows stand for the forward training process; the red arrows stand for the inverse inference process. In the inverse process, only the cINN part is inverted, and the summary network part remains the same as that in the forward process

The summary network $h_{\boldsymbol{\psi}}$ is supposed to be adjusted to the observed data $\boldsymbol{x}$. For example, a long short-term memory (LSTM) layer [30] is a typical architecture for time-series data. In this way, the compressed data $\widetilde{\boldsymbol{x}}$ with informative statistics is passed through the cINN, and taken as the condition while inducing the posterior of physical parameters $\boldsymbol{\theta}$, namely, $p_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\widetilde{\boldsymbol{x}} = h_{\boldsymbol{\psi}}(\boldsymbol{x}))$.

The cINN, assumed as an invertible function $f_{\boldsymbol{\phi}}$, is built up with a chain of conditional affine coupling blocks [27]. This structure ensures the neural network to be invertible, bijective and to have easily calculable Jacobian determinant $|\det J_{f_{\boldsymbol{\phi}}}|$ [8]. In the forward direction, the input is the physical parameters $\boldsymbol{\theta} \in \mathbb{R}^D$, while the output is the latent variable $\boldsymbol{z}_{\boldsymbol{\theta}}$, which by default follows a standard normal distribution $p(\boldsymbol{z}_{\boldsymbol{\theta}}) = \mathcal{N}_D(\boldsymbol{z}_{\boldsymbol{\theta}}|\mathbf{0}, \mathbb{I})$. Via the change-of-variables formula of probability, the posterior is reformulated as

$$p_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\widetilde{\boldsymbol{x}}) = p(\boldsymbol{z}_{\boldsymbol{\theta}})|\det J_{f_{\boldsymbol{\phi}}}|, \tag{5a}$$

$$\boldsymbol{z}_{\boldsymbol{\theta}} = f_{\boldsymbol{\phi}}(\boldsymbol{\theta}; \widetilde{\boldsymbol{x}}). \tag{5b}$$

Approximating the expectations by the Monte Carlo estimation for a batch of M samples $(\boldsymbol{x}^{(m)}, \boldsymbol{\theta}^{(m)})$, $m = 1, \ldots, M$ from the dataset, Eq. (4) becomes

$$\widehat{\boldsymbol{\phi}}, \widehat{\boldsymbol{\psi}} = \underset{\boldsymbol{\phi}, \boldsymbol{\psi}}{\operatorname{argmin}} \frac{1}{M} \sum_{m=1}^{M} \left(\frac{\left\| f_{\boldsymbol{\phi}}(\boldsymbol{\theta}^{(m)}; h_{\boldsymbol{\psi}}(\boldsymbol{x}^{(m)})) \right\|^2}{2} - log|\det J_{f_{\boldsymbol{\phi}}}| \right). \tag{6}$$

After the BayesFlow network is well trained, in the inverse direction, a single parameter estimate for the test observation $\boldsymbol{x}^{(k)}(t)$ can be derived by sampling the latent variable $\boldsymbol{z}_{\boldsymbol{\theta}}$ for one time with the optimized network parameters $\widehat{\boldsymbol{\phi}}, \widehat{\boldsymbol{\psi}}$. When $\boldsymbol{z}_{\boldsymbol{\theta}}$ is sampled for B times, the posterior distribution $p_{\widehat{\boldsymbol{\phi}}, \widehat{\boldsymbol{\psi}}}(\boldsymbol{\theta}^{(k)}|\boldsymbol{x}^{(k)}(t))$ can be obtained. The estimates of parameters are taken as the mean of the implicit posteriors:

$$\widehat{\boldsymbol{\theta}} = \frac{1}{B} \sum_{b=1}^{B} f_{\widehat{\boldsymbol{\phi}}}^{-1}(\boldsymbol{z}_{\boldsymbol{\theta}}{}^{(b)}; h_{\widehat{\boldsymbol{\psi}}}(\boldsymbol{x})). \tag{7}$$

Physics-enhanced latent space variational autoencoder

A well-known generative model, the variational autoencoder (VAE) [31] constructs a probabilistic representation of the observations $\boldsymbol{x}$, by fitting a recognition model as probabilistic encoder $q_{\boldsymbol{\zeta}}(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})$ to the intractable posterior as decoder $p_{\boldsymbol{\xi}}(\boldsymbol{x}|\boldsymbol{z}_{\boldsymbol{x}})$. The latent variable

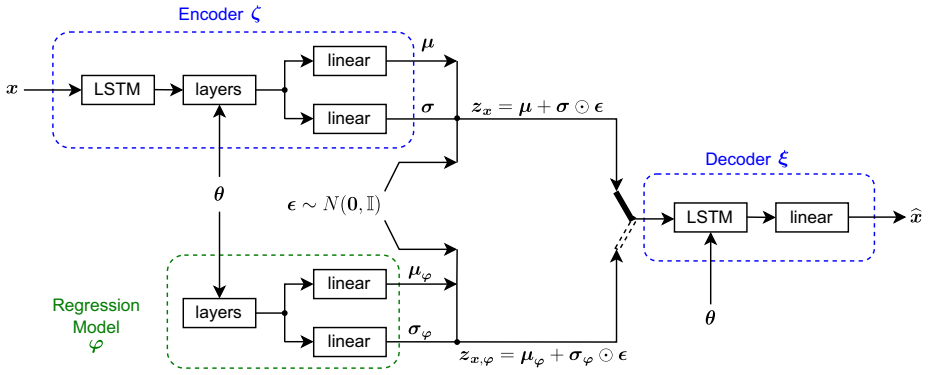


Fig. 2 PELS-VAE with regression model in Teacher–Student Architecture. $\boldsymbol{\zeta}$, $\boldsymbol{\xi}$, $\boldsymbol{\varphi}$ denote the network parameters of the encoder, decoder and the regression model respectively. The linear layer stands for: $\boldsymbol{y} = \boldsymbol{w}^T\boldsymbol{x} + \boldsymbol{b}$. The components "layers" consist of a series of linear layers with each linear layer following a ReLU activation function. In the training process, the switch connects the encoder and the decoder. And the outputs of regression model, i.e., $\boldsymbol{\mu}_{\boldsymbol{\varphi}}$, $\boldsymbol{\sigma}_{\boldsymbol{\varphi}}$ are compared with the outputs of the encoder $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$, respectively. After the three parts are well trained, the switch turns to the output of the regression model

$\boldsymbol{z_x}$ is a lower-dimensional compression of $\boldsymbol{x}$, which is re-parameterized by the mean $\boldsymbol{\mu}$, variance $\boldsymbol{\sigma}^2$ returning from the encoder, and an auxiliary noise variable $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbb{I})$,

$$\boldsymbol{z_x} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}. \tag{8}$$

The network parameters $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$ are optimized by maximizing the variational lower bound $\mathcal{L}$,

$$\widehat{\boldsymbol{\xi}}, \widehat{\boldsymbol{\zeta}} = \underset{\boldsymbol{\xi},\boldsymbol{\zeta}}{\operatorname{argmax}}\ \mathcal{L}(\boldsymbol{\zeta}, \boldsymbol{\xi}; \boldsymbol{x}) = \underset{\boldsymbol{\xi},\boldsymbol{\zeta}}{\operatorname{argmax}}\ \mathbb{E}_{q_{\boldsymbol{\zeta}}(\boldsymbol{z_x}|\boldsymbol{x})}[log\ p_{\boldsymbol{\xi}}(\boldsymbol{x}|\boldsymbol{z_x})] - \beta \mathbb{KL}(q_{\boldsymbol{\zeta}}(\boldsymbol{z_x}|\boldsymbol{x}) \parallel p_{\boldsymbol{\xi}}(\boldsymbol{z_x})), \tag{9}$$

where the hyperparameter β weights the importance of the KL-divergence term, thus playing a role in disentangling the latent representation [32].

In order to model the observations $\boldsymbol{x}$ and the latent variable $\boldsymbol{z_x}$ conditioned on the physical parameters $\boldsymbol{\theta}$, $\boldsymbol{\theta}$ is introduced into both the encoder and decoder by Martínez-Palomera et al. [17] as Physics-Enhanced Latent Space VAE (PELS-VAE), illustrated in Fig. 2. In addition, a separate regression model is required to project the physical parameters $\boldsymbol{\theta}$ to the latent variable $\boldsymbol{z_x}$ bypassing the observations $\boldsymbol{x}$. In the original paper [17], the regression model is individually designed and trained after the VAE is well trained. To save the costs of training and network design, the regression model in our method is designed the same as the encoder without the LSTM layer(s), and trained with the encoder simultaneously. In this way, the encoder (as a "teacher") and the regression model (as a "student") in parallel build up a Teacher–Student Architecture, which is widely used in semi-supervised learning [18, 33]. The difference between the outputs of these two parts is supposed to be minimized. As a result, the overall network is optimized by

$$\widehat{\boldsymbol{\xi}}, \widehat{\boldsymbol{\zeta}}, \widehat{\boldsymbol{\varphi}} = \underset{\boldsymbol{\xi},\boldsymbol{\zeta},\boldsymbol{\varphi}}{\operatorname{argmin}} - \mathcal{L}(\boldsymbol{\zeta}, \boldsymbol{\xi}; \boldsymbol{x}, \boldsymbol{\theta}) + MSE(\boldsymbol{\mu}, \boldsymbol{\mu}_{\boldsymbol{\varphi}}) + MSE(\boldsymbol{\sigma}, \boldsymbol{\sigma}_{\boldsymbol{\varphi}}), \tag{10}$$

where $\boldsymbol{\mu}_{\boldsymbol{\varphi}}$ and $\boldsymbol{\sigma}_{\boldsymbol{\varphi}}$ are obtained from the regression model, while $\mathcal{L}(\boldsymbol{\zeta}, \boldsymbol{\xi}; \boldsymbol{x}, \boldsymbol{\theta})$ is the lower bound from Eq. (9) with $\boldsymbol{\theta}$ as an additional variable.

After the PELS-VAE model is well trained, the regression-decoder part $f_{\widehat{\boldsymbol{\varphi}},\widehat{\boldsymbol{\xi}}}$ is used to generate new time series with physical parameters that are comparable to those used in the training set. In other words, the regression model $f_{\widehat{\boldsymbol{\varphi}}}$ is able to represent and thus replace the encoder $f_{\widehat{\boldsymbol{\zeta}}}$, as expressed in the equation,

$$f_{\widehat{\boldsymbol{\varphi}}}(\boldsymbol{\theta}) = \boldsymbol{z_x} = f_{\widehat{\boldsymbol{\zeta}}}(\boldsymbol{x}, \boldsymbol{\theta}). \tag{11}$$

Methods

Due to the different identifiability of each parameter, we propose the sensitivity-guided iterative parameter identification and data generation method. In each step, the strongly identifiable parameters are preliminarily selected through a sensitivity analysis. Further, these parameters are trained and validated by a BayesFlow model. If the model does not show a tendency to learn all parameters simultaneously during the initial training epochs, only the potentially well-estimated parameters should be selected. We call this process trial inference. Then, the confirmed parameters are learned and estimated by another BayesFlow model. After the strongly identifiable parameters are inferred, the weakly identifiable ones can only become strongly identifiable when the variance of the determined parameters is eliminated from the model observations. For this reason, we apply a generative model, PELS-VAE, to generate a new training set for the inference model BayesFlow. In both trial and formal inference, the corresponding training data is generated by a previously well-trained PELS-VAE model from the remaining unidentified parameters. The advantages of this method are threefold: first, the number of parameters to identify in each step is reduced, and thus the size of the required training set and complexity of the corresponding networks also scale down; second, with the sensitivity analysis hypothesizing the identifiability of parameters, the inference model becomes well-conditioned by ignoring the weakly identifiable parameters; third, as the strongly identifiable parameters in the previous step(s) are fixed, the initially weakly identifiable ones can make distinguishable contributions to the variance of the model response, and thus their identifiability is improved in the present step.

In the following, the overall algorithm is at first introduced in "Sensitivity-guided iterative parameter identification and data generation" section, then the parameter selection algorithm and the iterative workflow with fixed parameter selection are separately explained in "Selection of practically identifiable parameters by global sensitivity analysis" and "Iterative parameter identification and data generation" sections. It is worth mentioning that, in the following, $\boldsymbol{\theta}$ stands for the variable or the data set of parameters, like $\boldsymbol{\theta}^{(o)}$ for the test sample, and $\widehat{\boldsymbol{\theta}}^{(o)}$ for the estimate . In addition, for the convenience of set algebra, Θ serves as an abstract set of parameters.

Sensitivity-guided iterative parameter identification and data generation

Taking the advantages of sensitivity analysis, which provides guidance on parameter selection, and thus obviously reduces the dimension of an inference model, the decision of identification order can be integrated into the iterative workflow of parameter identification and data generation. The overall algorithm is encapsulated in Algorithm 3 in Appendix A.1 and visualized in Fig. 3.

Sensitivity analysis and parameter identification are achieved synchronously in a divide-and-conquer paradigm. Nevertheless, the selected parameters by sensitivity analysis are not guaranteed to be well-identified. To compensate for the inconsistency, a trial inference procedure is inserted after selection by sensitivity according to Algorithm 1 in "Selection of practically identifiable parameters by global sensitivity analysis" section before the formal inference. During the trial inference, the same training set is applied as the formal inference, which is generated by a well-trained PELS-VAE model according to Algorithm 2 in "Iterative parameter identification and data generation" section. Through the trial procedure, the well-converged parameters are taken as actually identifiable ones Θ_n.

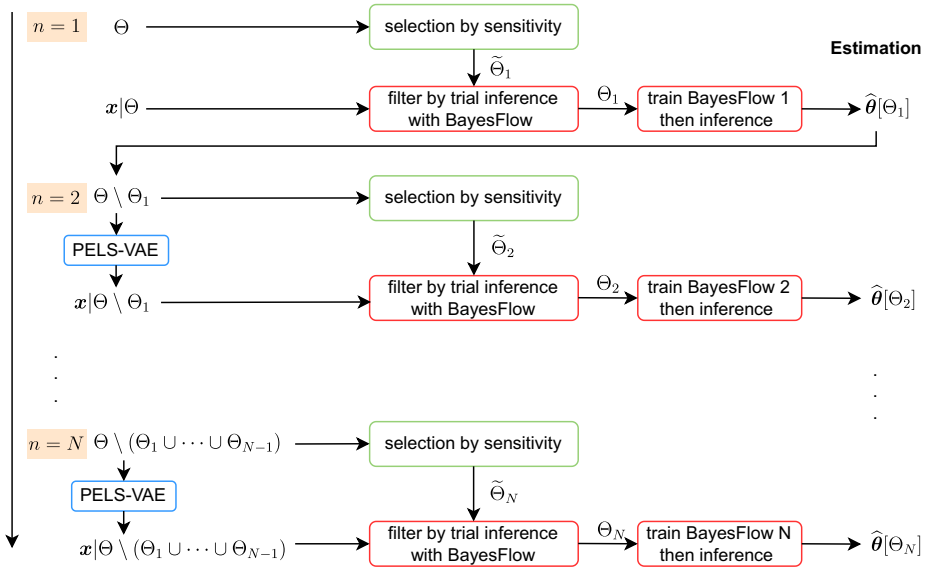


Fig. 3 Sensitivity-guided iterative parameter identification and data generation. In the loop step n, $\widetilde{\theta}_n$ stands for the set of possibly identifiable parameters, Θ_n for the the set of filtered truly identifiable parameters, and $\widehat{\theta}[\Theta_n]$ for the estimates of the filtered parameter with the subset Θ_n. PELS-VAE stands for a well-trained model with network parameters $\widehat{\boldsymbol{\varphi}}, \widehat{\boldsymbol{\xi}}$. In each step n, first, the potentially identifiable parameters $\widetilde{\Theta}_n$ are selected with sensitivity analysis according to Algorithm 1 in "Selection of practically identifiable parameters by global sensitivity analysis" section. Second, except step 1, the training and test samples, which are denoted as $\boldsymbol{x}|\Theta \setminus (\Theta_1 \cup \cdots \cup \Theta_{n-1})$, are transformed by the well-trained PELS-VAE model. Third, $\widetilde{\Theta}_n$ is filtered by trial inference. At last, the parameter set Θ_n can be identified by training a BayesFlow model. Further details are shown in Algorithm 2 in "Iterative parameter identification and data generation" section

Subsequently, Θ_n is supposed to be learned and estimated by another BayesFlow model. Assuming all parameters are practically identifiable, the procedure stops when there is no remaining unidentified parameters.

Selection of practically identifiable parameters by global sensitivity analysis

An effective method to calculate global Sobol indices is Saltelli Sampling [34], a quasi-random sampling method with low-discrepancy sequences. Compared with random sampling, points are chosen from a high-dimensional space in such a way that they fill the space more evenly. For each parameter, the quasi-random sampling is performed for S times. Then, with the output calculated by a simulation model, the first-order Sobol indices at each time step $S_{global,i}(\boldsymbol{x}(t))$ by Eq. (2) need to be analyzed. For example, when $S_{global,i}(\boldsymbol{x}(t))$ keeps unchanged at zero for all time steps, this parameter could be structurally non-identifiable and thus should be excluded from further considerations from the perspective of practical identification. Moreover, in order to representatively compare the sensitivity among all parameters, for each parameter θ_i, $S_{global,i}(\boldsymbol{x}(t))$ is expected to be averaged over all time steps and all output channels as follows,

$$\overline{S_{global,i}} = \mathbb{E}_{\boldsymbol{x}(t)}[S_{global,i}(\boldsymbol{x}(t))]. \tag{12}$$

The boundary of determination whether a parameter is strongly or weakly identifiable depends on the model and its observations. Furthermore, with varying amounts of unidentified parameters, the sensitivity behavior of the parameters alters consequently in an unknown manner. Therefore, we develop Algorithm 1 for selecting possibly practically identifiable parameters with first-order Sobol indices without any insight into the

physical model. Instead, the selection is based on the comparison of sensitivity among the parameters. At first, the list of average first-order Sobol indices should be sorted in descending order, denoted as $[\overline{S}]_{\downarrow}$. The threshold K is considered to be the maximally tolerable drop ratio from the larger Sobol index $[\overline{S}]_{\downarrow}[i-1]$ to the next smaller Sobol index $[\overline{S}]_{\downarrow}[i]$. Assuming that at least two parameters are chosen, then (1) the ratio between 1 (the theoretical maximum of sensitivity) and the practically maximal sensitivity of these parameters; (2) the ratio between the maximal and the second maximal sensitivity are tolerable. The ratio is defined in Eq. (13). The lager value between $r(0,1)$ and $r(1,2)$ is then chosen as the threshold K for the list $[\overline{S}]_{\downarrow}$, i.e., $K = max(r(0,1), r(1,2))$.

$$r(i-1,i) = log_{10}\left(\frac{[\overline{S}]_{\downarrow}[i-1]}{[\overline{S}]_{\downarrow}[i]}\right), \forall i > 1; r(0,1) = log_{10}\left(\frac{1}{[\overline{S}]_{\downarrow}[1]}\right). \tag{13}$$

The list $[\overline{S}]_{\downarrow}$ is looped through, until the drop from the former one to the latter one exceeds the threshold, namely, $r(i-1,i) > K$.

Algorithm 1: Selection of practically identifiable parameters by global sensitivity analysis with first-order Sobol indices

Input:
(1) value range of each parameter $[\theta_{i,L}, \theta_{i,U}]$ and the corresponding default value $\theta_{i,0}, \forall i \in 1, \ldots, D$;
(2) identified parameters set Θ_{done}, unidentified parameters set $\Theta_{todo} = \Theta \setminus \Theta_{done}$;
(3) the physical model for generating observations from parameters;
(4) S as the sampling size of each parameter in Saltelli sampling procedure;
(5) minimal average first-order Sobol indices δ.

Perform Saltelli's sampling for S times $\forall \theta_i \in \Theta_{todo}, |\Theta_{todo}|, \theta_i \sim U(\theta_{i,L}, \theta_{i,U})$, with $\forall \theta_j \in \Theta_{done}, \theta_j = \theta_{j,0}$.
Calculate each simulation output with each parameter set.
Calculate the average first-order Sobol indices $\overline{S_{global,i}}, \forall \theta_i \in \Theta_{todo}$ with Eq. (2) and Eq. (12).
Sort the average first-order Sobol indices in descending order, obtain the list of sorted average first-order Sobol indices $[\overline{S}]_{\downarrow}$, and the corresponding parameter indices $arg([\overline{S}]_{\downarrow})$.
Select the parameters according to average first-order Sobol indices:
if $length([\overline{S}]_{\downarrow}) \leq 2$ **then**
    $\widetilde{\Theta} = \{\Theta_{todo}[arg([\overline{S}]_{\downarrow})]\}$
else
    $K = max(r(0,1), r(1,2)) = max(log_{10}\frac{1}{[\overline{S}]_{\downarrow}[1]}, log_{10}\frac{[\overline{S}]_{\downarrow}[1]}{[\overline{S}]_{\downarrow}[2]})$
    $\widetilde{\Theta} = \{\Theta_{todo}[arg([\overline{S}]_{\downarrow})[1]], \Theta_{todo}[arg([\overline{S}]_{\downarrow})[2]]\}$
    for $i = 3, \ldots, length([\overline{S}]_{\downarrow})$ **do**
        if $[\overline{S}]_{\downarrow}[i] > \delta$ **then**
            if $r(i-1,i) > K$ **then**
                break
            else
                add $\Theta_{todo}[arg([\overline{S}]_{\downarrow})[i]] \rightarrow \widetilde{\Theta}$

Output: the possibly identifiable parameters, $\widetilde{\Theta}$.

Afterwards, the sorted parameters before the unacceptable drop in sensitivity are collected as strongly identifiable parameters. However, the sensitivity analysis cannot ensure that the selected parameters are able to be well-estimated in practice. On the one hand, well-posed inverse problems have to satisfy the three conditions: existence, uniqueness, and stability of solutions. Satisfaction of all the three conditions is defined as *estimability* [35]. But identifiability corresponds mainly to the question of uniqueness. On the other hand, the global sensitivity, which only considers the contribution of each parameter to the total variance, cannot characterize the dynamics of the model thoroughly. Hence, an inference model is necessary to further examine the identification of these parameters.

Iterative parameter identification and data generation

Decoupling the sensitivity analysis from Algorithm 3, the workflow of iterative parameter identification with BayesFlow and data generation with pre-trained PELS-VAE is described in Algorithm 2. The selected parameters of interest in all steps are already categorized by decreasing sensitivity and denoted as $\Theta_1, \Theta_2, \ldots, \Theta_N$. This algorithm can be executed independent of Algorithm 1, when the order of parameter identification is fixed. The PELS-VAE should be well-trained with the original training set previously.

Algorithm 2: Iterative parameter identification and data generation

Input:
(1) beforehand selected parameter sets $\Theta_1, \Theta_2, \ldots, \Theta_N$, with $\Theta = \Theta_1 \cup \Theta_2 \cup \cdots \cup \Theta_N$;
(2) trained PELS-VAE model $\widehat{\boldsymbol{\varphi}}, \widehat{\boldsymbol{\xi}}$ with training set $(\boldsymbol{\theta}^{(m)}, \boldsymbol{x}^{(m)})$, $m = 1, \ldots, M$;
(3) B as the number of samples in BayesFlow for inference of the estimates of parameters.

Required data:
training set $(\boldsymbol{\theta}^{(m)}, \boldsymbol{x}^{(m)})$, $m = 1, \ldots, M$, test sample $(\boldsymbol{\theta}^{(o)}, \boldsymbol{x}^{(o)})$.

Initialization: identified parameters set $\Theta_{done} = \varnothing$, unidentified parameters set $\Theta_{todo} = \Theta$.
for $n = 1, \ldots, N$ **do**
  if $\Theta_{done} \neq \varnothing$ **then**
    Transform training set and test sample applying PELS-VAE model:
    Compute $\boldsymbol{x}_n^{(m)}$, $m = 1, \ldots, M$ and $\boldsymbol{x}_n^{(o)}$ according to Eq. 14.
  Update BayesFlow network parameters $\boldsymbol{\phi}_n, \boldsymbol{\psi}_n$ with training set $(\boldsymbol{\theta}^{(m)}[\Theta_n], \boldsymbol{x}_n^{(m)})$, $m = 1, \ldots, M$ by Eq. 6,
  until convergence to well-trained network parameters $\widehat{\boldsymbol{\phi}}_n, \widehat{\boldsymbol{\psi}}_n$.
  Perform inference for $\widehat{\boldsymbol{\theta}}^{(o)}[\Theta_n] = \frac{1}{B}\sum_{b=1}^{B} f_{\widehat{\boldsymbol{\phi}}_n}^{-1}(\boldsymbol{z}_{\boldsymbol{\theta}}^{(b)}; h_{\widehat{\boldsymbol{\psi}}_n}(\boldsymbol{x}_n^{(o)}))$.
  Add $\Theta_n \rightarrow \Theta_{done}$, remove Θ_n from Θ_{todo}.

Output: estimate of parameters $\widehat{\boldsymbol{\theta}}^{(o)}$.

In each step n, first, the identified parameters in the training set are fixed with the estimated values. Correspondingly, the time series from fixed identified parameters is supposed to be generated by a previously well-trained PELS-VAE model with network parameters $\widehat{\boldsymbol{\varphi}}, \widehat{\boldsymbol{\xi}}$. In this way, the new training set $(\boldsymbol{\theta}^{(m)}[\Theta_n], \boldsymbol{x}_n^{(m)})$, $m = 1, \ldots, M$ are prepared to identify parameters in Θ_n, where $\boldsymbol{x}_n$ is

$$\begin{aligned} &let\ \boldsymbol{\theta}[\Theta_1 \cup \cdots \cup \Theta_{n-1}] := \widehat{\boldsymbol{\theta}}[\Theta_1 \cup \cdots \cup \Theta_{n-1}] \Rightarrow \boldsymbol{\theta}_n, \\ &\boldsymbol{x}_n = f_{\widehat{\varphi},\widehat{\xi}}(\boldsymbol{\theta}_n). \end{aligned} \tag{14}$$

Here, $\boldsymbol{\theta}_n$ denotes the transformed parameter set with fixed identified parameters. It is different from $\boldsymbol{\theta}[\Theta_n]$, which means the filtered parameter set with the to be identified subset in the step n. The test sample $\boldsymbol{x}^{(o)}$ is also transformed in the same way. Second, a BayesFlow model is trained with the transformed training set. Third, the parameters of the test sample $\widehat{\boldsymbol{\theta}}^{(o)}[\Theta_n]$ are estimated with the well-trained BayesFlow by sampling B parameter vectors from the approximate posterior.

In such a manner, all parameters in Θ of the test sample can be identified. For other test samples within the same physical model family, the PELS-VAE model and the first BayesFlow model, which are both trained with the original training set, can also be used. On the contrary, the intermediate BayesFlow models are all specialized with the identified parameters of an individual test sample, and thus should be trained afresh.

Experiments

In this section, the thermal model of an automotive cabin and the dataset generated from it are explained in "Simulation model and dataset" section. Then, the results of model calibration applying the sensitivity-guided iterative algorithm described in "Methods" section are demonstrated separately in accordance with the three parts of the methods: sensitivity analysis in "Sensitivity analysis" section, signal reconstruction performance with PELS-VAE in "Signal reconstruction with PELS-VAE" section and the total parameter inference results with BayesFlow in "Parameter inference with BayesFlow" section.

The sensitivity analysis is implemented with the aid of Sensitivity Analysis Library (SALib) [36]. The model PELS-VAE is modified on the basis of the published scripts[1] into a Teacher–Student Architecture.

The model BayesFlow is developed using the package FrEIA[2] with the framework PyTorch [37], in reference to the published scripts,[3] which is developed with the framework TensorFlow 1.0 [38].

Training of the PELS-VAE and BayesFlow model is supported by a single GPU equipped with NVIDIA® A30. For the purpose of better convergence of the training process, the optimizer Adam is employed with a multi-step learning rate scheduler for PELS-VAE and an exponentially decayed one for BayesFlow.

Simulation model and dataset

The thermal dynamics of an automotive cabin on an industrial scale is modeled by one-dimensional heat transfer, pressure loss and air exchange in the use of the Human-Comfort Modelica-library [39]. More information about the model is available at [40]. Our applied simulation model is based on the single-air-volume approach with the same component implementations. The model includes stiffness, discontinuities, 2 nonlinear algebraic loops, 250 states and approximately 14,300 equations. Assuming ideal mixed air volumes, the temperature development is observed with 8 sensors installed in the front/rear, left/right, top/bottom of the cabin. With the aim of calibrating the model, 17

[1] https://github.com/jorgemarpa/PELS-VAE.
[2] https://github.com/VLL-HD/FrEIA.
[3] https://github.com/stefanradev93/BayesFlow.

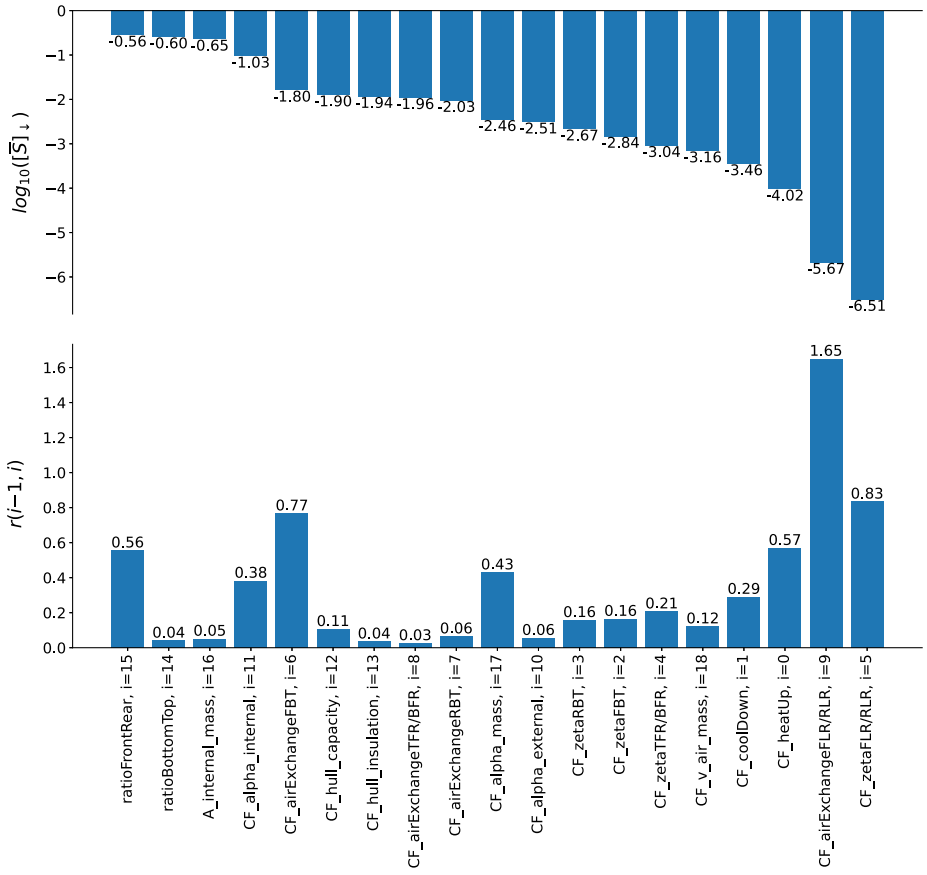


Fig. 4 Comparison of average first-order Sobol indices within 19 parameters. The first diagram displays the sorted parameters by decreasing $log_{10}([\bar{S}]_{\downarrow})$. In order to detect the sharp drop in the $\overline{S_{global,i}}$, the logarithms of relative ratios between the former one and the latter one $r(i-1, i)$ are shown in the second diagram

adjustable parameters are utilized including the internal/external convective heat transfer coefficients ("CF_alpha_internal", "CF_alpha_external"), insulation and capacity of the surrounding partitions ("CF_hull_insulation", "CF_hull_capacity"), air exchange rates ("CF_airExchangeFBT", "CF_airExchange*"), the internal mass of seats and center console ("A_internal_mass", "CF_alpha_mass"), the pressure loss coefficients ("CF_zeta*"), the air velocity at the surface of the internal mass ("CF_v_air_mass") and the weighting factors of the thermal load position ("ratioBottomUp", "ratioFrontRear"). These properties are investigated from the dynamics and steady state of the model response while the model is (0) passively heating up, then (1) cooling down and (2) heating up again. Each step is simulated from 0 to 1800 s. For calibration of a model, the latter two processes are usually inspected. Thus, cool-down and heat-up factors ("CF_coolDown", "CF_heatUp") are designed respectively. Totally, 19 (= 17 + 2) parameters are supposed to be adapted to fit the simulation model to the measurement. The parameters are listed in Fig. 4.

In practice, the experts calibrate the simulation model from the measurement using these 19 parameters by observing corresponding cool-down and heat-up processes. Without specification, the average air temperatures of the steady states and relevant dynamic values are selected as points of interest to fit the simulation model to the measurement. The following procedure is performed repeatedly. First, "CF_alpha_internal",

"CF_alpha_external", "CF_hull_insulation" are optimized by comparing the steady state points. Then, "A_internal_mass", "CF_alpha_mass" are optimized by comparing the dynamic values. Afterwards, when it is necessary to analyze the internal comfort, then the temperature difference can be minimized by pressure loss coefficients ("CF_zeta*"). The other parameters are used for fine-tuning.

Because this procedure is highly empirical, and it usually takes more than one week to calibrate a model, we evaluate our experiment results with respect to the error between the estimate and the ground truth of parameters, as well as the error between the model response from the estimate and that from the ground truth.

The simulation dataset with 2000 samples is split into 1600 samples for training, one sample for testing and the other 399 for validation. To specify, the validation samples are not used for the stopping criterion of the training processes. Instead, they support the evaluation of the trial inference and inference results in each iterative step. Because, for only one test sample, the parameter estimates are hard to evaluate. Each sample contains 19 uniformly sampled parameters, which are all practically identifiable and can be considered independent of each other according to expert knowledge. The corresponding observation includes 8 ($= 2 \times 4$) time series, which are collected first in a cool-down process then in a heat-up process from 4 temperature sensors on the left side of the cabin. Only 4 sensors instead of all 8 sensors are considered because of the symmetry of the left and the right side of the thermal model. In order to reduce the data dimension, the time series are sampled with a step size of 18 s (0:18:1800). To sum up, the parameter $\boldsymbol{\theta}$ is in the shape of $D = 19$, while the observation $\boldsymbol{x}$ is in the shape of 100×8.

Sensitivity analysis

Applying the Saltelli sampling with the size of $S = 64$, the sensitivity of 19 parameters with first-order Sobol indices is analyzed in a decremental way, through which the relative strongly identifiable parameters are selected, then the remaining ones together with the ones ruled out by trial inference are further examined. The parameters are arranged into 5 groups through the iterative procedure. (Figures for the sensitivity analyses of these groups are shown in Figs. 7, 8, 9, 10, 11, and 12 in Appendix A.2.)

According to Algorithm 1, a parameter can be chosen when satisfying two conditions, (1) the $\overline{S_{global,i}}$ reaches the threshold $\delta = 0.001$, (2) the ratio $r(i-1, i)$ in Eq. (13) should be restricted by the maximal ratio $r(i-1, i) \leqslant K$, where K is determined by the first two indices, namely $K = max(r(0, 1), r(1, 2))$.

For example, as shown in Fig. 4, among the sorted parameters with index from $i = 15$ to $i = 2$, where $\overline{S_{global,i}} > \delta = 0.001$, the maximal acceptable ratio drop is given by $K = 0.56 = r(0, 1)$. Then, the parameters "ratioFrontRear", "ratioBottomUp", "A_internal_mass" and "CF_alpha_internal" with their drop ratios not greater than K, are chosen as the possibly identifiable parameters. Since the drop ratio at the parameter "airExchangeFBT" exceeds K, the others including this one are screened out in this step. Similarly, for further examination, the following parameters can be pre-selected in each step: the first 9 ones in the group of 15 parameters in Fig. 9 in Appendix A.2, the first 5 ones in the group of 12 parameters in Fig. 10b in Appendix A.2, all the 7 in the group of 7 parameters in Fig. 11b in Appendix A.2 and naturally the last left 2 parameters.

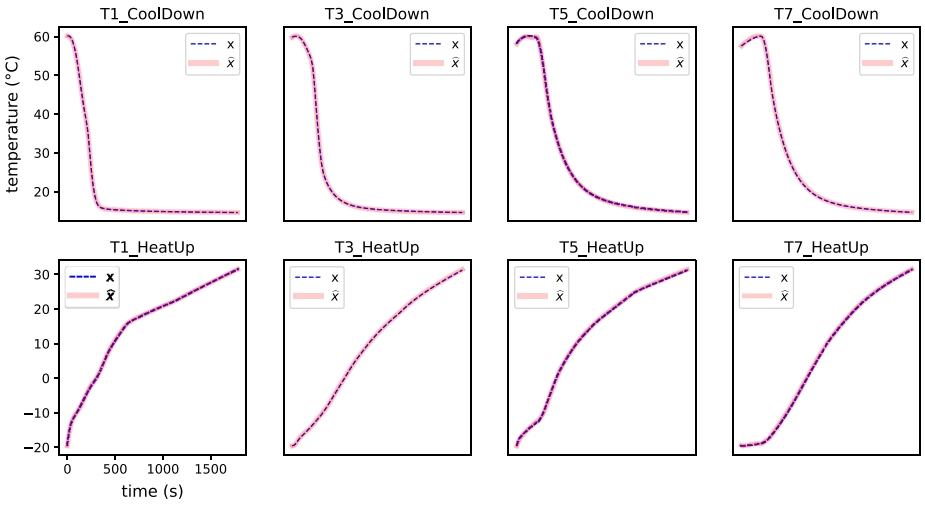


Fig. 5 Signal reconstruction with PELS-VAE. T1, T3, T5, T7 denote the temperature sensors on the left side. The blue dashed lines stand for the ground truth generated by simulation model, while the red solid lines are predicted by a well-trained PELS-VAE model. They are taken as the means of 20 passes. The blue ones overlaps the red ones because of negligible difference with mean absolute error 0.085 °C

Signal reconstruction with PELS-VAE

As shown in Fig. 5, the PELS-VAE model trained with dataset in Eq. (14) performs well in prediction of time series from the parameters. Thanks to good prediction performance, the surrogate model can be regarded as a simplification of the complicated physical model, and thus enables the fast generation of new data between iteration steps. Generating a dataset of 2000 samples with PELS-VAE takes on average 16 s, while running the simulation model with at first heat-up then cool-down for 2000 times lasts longer than 2.5 h.

Parameter inference with BayesFlow

Applying Algorithm 3, 19 parameters are identified iteratively in 5 groups using BayesFlow models. In the iterative process, the first BayesFlow model is trained with the original training set, while the other 4 models are trained with adapted datasets. Each dataset is generated with the well-trained PELS-VAE model through fixing already identified parameters. Hence, the test and validation samples to examine the regression performance of each BayesFlow model are also different. The regression performance of each model regarding one test sample and 399 validation samples is visualized in Fig. 14 in Appendix A.3, with the size of sampling parameter vectors from the approximate posterior $B = 100$. In each iterative step, the density estimate results of which are demonstrated in Fig. 13 in Appendix A.3. The Fig. 6 shows the results in all iteration steps including the selection and filtering of parameters, the duration of each step, as well as the evaluation of estimates.

The average relative error rate across all parameters achieves 1.62%, among which parameter 7 ("CF_airExchangeRBT") with $ER = 5.008\%$ in the third step is slightly more error-prone than the others. Besides, the density estimation allows for uncertainty quantification. Out of the 19 posterior distributions, 16 confidence intervals cover the ground truth, denoting the reliability of the estimate. On the contrary, the other three distributions of the parameters 7, 9 and 14 ("CF_airExchangeRBT", "CF_airExchangeFLR/RLR" and "ratioBottomUp") tend to overfit on account of undersized normalized standard deviation.

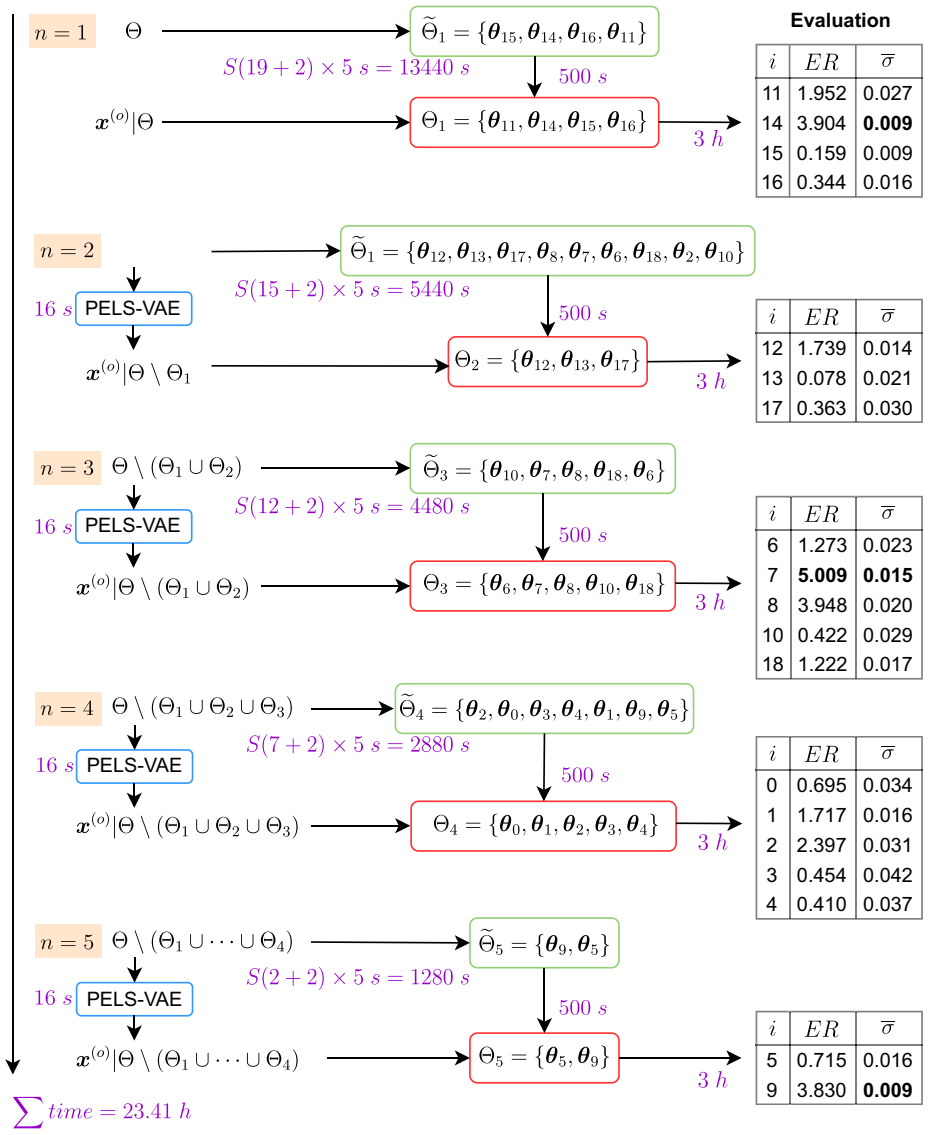


Fig. 6 Parameter selection and inference results in each iteration step. The indices are in reference to the parameter names listed in Figs. 4 and 13 in Appendix A.3. The relative error rate $ER = |estimate - ground\ truth| / ground\ truth \times 100\%$. $\overline{\sigma}$ is the standard deviation of the posterior distribution normalized by each value range. The text in magenta shows the duration of a process. S denotes the sampling size of each parameter in Saltelli sampling procedure, $S = 64$. The time consumption shown in the figure is summed up to 23.41 h

To further validate the calibration results, the model outputs calculated by the simulation model from the estimate and the ground truth are compared. For a successful calibration, the deviation of any air temperature of the steady states and relevant dynamic values should be within ±0.5 °C. The mean absolute error of the time series reaches 0.108 °C, which confirms the accuracy of the estimate. Moreover, our approach shows obvious speed improvements over the classical calibration method. The generation of dataset (2000 × 5 s), then training of the PELS-VAE model (1 h) can be performed during the first iteration step. Except for the two preparation procedures, execution of the entire algorithm takes 23.41 h, as shown in Fig. 6.

Conclusion

In this study, we propose a novel algorithm of automatic, accurate and reliable model calibration from the point of view of parameter identification. Our contributions are threefold:

- Foremost, we utilize the relationship between sensitivity, identifiability and estimability; that is, the sensitivity analysis indicates whether a parameter is strongly or weakly identifiable from the observed data, leading to potential estimability, which is then confirmed by the inference methods. Supported by sensitivity analysis, inference can be efficiently performed by focusing on strongly identifiable parameters. Thus, the learning process converges faster because of reduction in the required training data size and the complexity of the inference model.
- Moreover, we embed a data generation process as a forward problem between the parameter identification processes as inverse problems. Because for a physical model, both forward and inverse problems share the same surrogate model, the forward model assists the inverse model to exclude the influence of the determined parameters. Through this iterative process, weakly identifiable parameters become distinguishable in the variance of observations.
- In addition, we choose the BayesFlow as the inference model, because, on the one hand, it is built with a separate summary network and a conditional invertible neural network. This separation encourages the summary network to extract the most important information from the observed data, in order to learn the posterior distribution of parameters precisely. On the other hand, the confidence interval obtained from the distribution provides useful guidance for further processing, as it implies the credibility of the inference.

Nevertheless, a major limitation of this algorithm is that the deep learning based inference model and generative model have their inherent constraints on generalization ability owing to data. Another limitation is that we assume each setting of parameters corresponds to a unique model response. Due to the large amount of parameters and the complexity of model, it is possible that two settings of parameters coincidentally have the same model response, which has not been investigated. Besides, it is worth mentioning the fact that the method can be hard to implement for practitioners without knowledge of deep learning, as both neural architectures have many tunable hyperparameters which can affect performance.

In future work, firstly, the optimal size of training set with regard to the number and the value ranges of parameters should be explored. Secondly, this proposed method, although validated by the simulation data, should be examined on the real measurements. Thirdly, in our research, the overall time series is applied to identify parameters. In order to examine the steady states and the dynamic behavior closely, the time series, where the sensitivity of parameters is higher, can be particularly given to the BayesFlow model as conditions. Last but not the least, each part of this algorithm, namely the sensitivity analysis, the PELS-

VAE model and BayesFlow model can be potentially incorporated with other methods. For example, the sensitivity analysis can assist experts to arrange the order of parameters in the classical model calibration method; the PELS-VAE model can also generate data in use cases where the demand in rapidness is higher than precision; the BayesFlow model can estimate the properties of products in end-of-line testing [41].

Appendix

Algorithm of sensitivity-guided iterative parameter identification and data generation

Algorithm 3: Sensitivity-guided iterative parameter identification and data generation

Input:
for selection of possibly identifiable parameters:
(1) value range of each parameter $[\theta_{i,L}, \theta_{i,U}]$ and the corresponding default value $\theta_{i,0}, \forall i \in 1, \ldots, D$;
(2) the physical model $f(\cdot)$ and its observation function $g(\cdot)$;
(3) S as the sampling size of each parameter in Saltelli sampling procedure.
for iterative parameter identification and data generation:
(4) trained PELS-VAE model $\widehat{\boldsymbol{\varphi}}, \widehat{\boldsymbol{\xi}}$ with training set $(\boldsymbol{\theta}^{(m)}, \boldsymbol{x}^{(m)}), m = 1, \ldots, M$;
(5) B as the number of samples of BayesFlow for inference the estimates of parameters. (6) minimal average first-order Sobol indices δ.

Required data:
training set $(\boldsymbol{\theta}^{(m)}, \boldsymbol{x}^{(m)}), m = 1, \ldots, M$, test sample $(\boldsymbol{\theta}^{(o)}, \boldsymbol{x}^{(o)})$.

Initialization: step $n = 1$, identified parameters set $\Theta_{done} = \varnothing$, unidentified parameters set $\Theta_{todo} = \Theta$.
while $\Theta_{todo} \neq \varnothing$ **do**
  Select the parameters with comparable average first-order Sobol indices with threshold δ according to Algorithm 1, get $\widetilde{\Theta}_n$.
  Transform training set and test sample with PELS-VAE model by computing $\boldsymbol{x}_n^{(m)}, m = 1, \ldots, M$ and $\boldsymbol{x}_n^{(o)}$ according to Equation (14).
  Filter selected parameters $\widetilde{\Theta}_n$ **by trial inference:**
  Train BayesFlow with training set $(\boldsymbol{\theta}^{(m)}[\widetilde{\Theta}_n], \boldsymbol{x}_n^{(m)}), m = 1, \ldots, M$,
  select the parameter, the regression of which tends to converge towards the ground truth during the first 500 training epochs, $\Theta_n = \left\{\theta_i \in \widetilde{\Theta}_n : |\widehat{\theta}_i - \theta_i| < \epsilon\right\}$.
  Inference of Θ_n **with BayesFlow:**
  Update BayesFlow network parameters $\boldsymbol{\phi}_n, \boldsymbol{\psi}_n$ with training set $(\boldsymbol{\theta}^{(m)}[\Theta_n], \boldsymbol{x}_n^{(m)}), m = 1, \ldots, M$ by Equation (6),
  until convergence to well-trained network parameters $\widehat{\boldsymbol{\phi}}_n, \widehat{\boldsymbol{\psi}}_n$.
  Perform inference for $\widehat{\Theta}_n = \widehat{\boldsymbol{\theta}}^{(o)}[\Theta_n] = \frac{1}{B}\sum_{b=1}^{B} f_{\widehat{\boldsymbol{\phi}}_n}^{-1}(\boldsymbol{z}_{\boldsymbol{\theta}}^{(b)}; h_{\widehat{\boldsymbol{\psi}}_n}(\boldsymbol{x}_n^{(o)}))$.
  Add $\widehat{\Theta}_n \rightarrow \Theta_{done}$, remove $\widehat{\Theta}_n$ from Θ_{todo}.
  $n = n + 1$

Output:
the identification order of parameters $\Theta_1, \Theta_2, \ldots, \Theta_N$;
estimate of parameters $\widehat{\boldsymbol{\theta}}^{(o)}$.

Results of sensitivity analysis with first-order sobol indices

See Figs. 7, 8, 9, 10, 11, and 12.

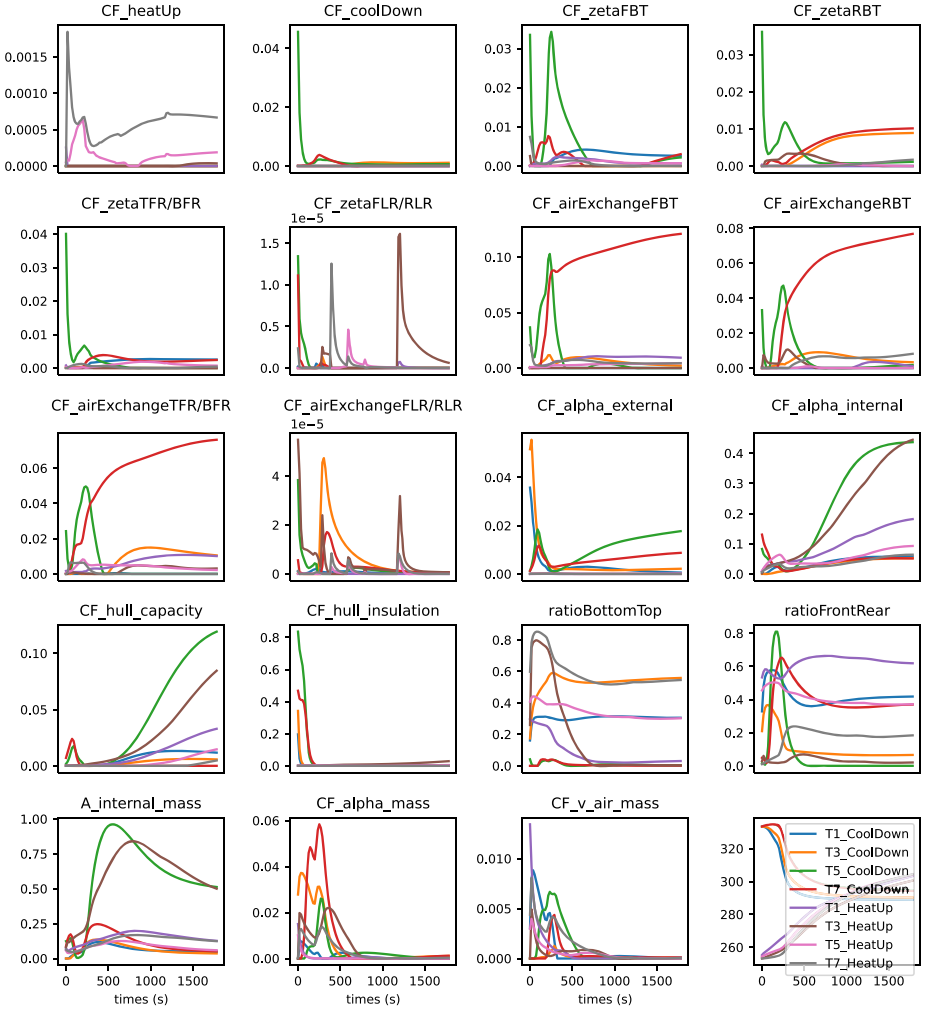


Fig. 7 Sensitivity analysis with first-order Sobol indices of 19 parameters from 8 time series with respect to time from 0 to 1800 s with step size of 18 s

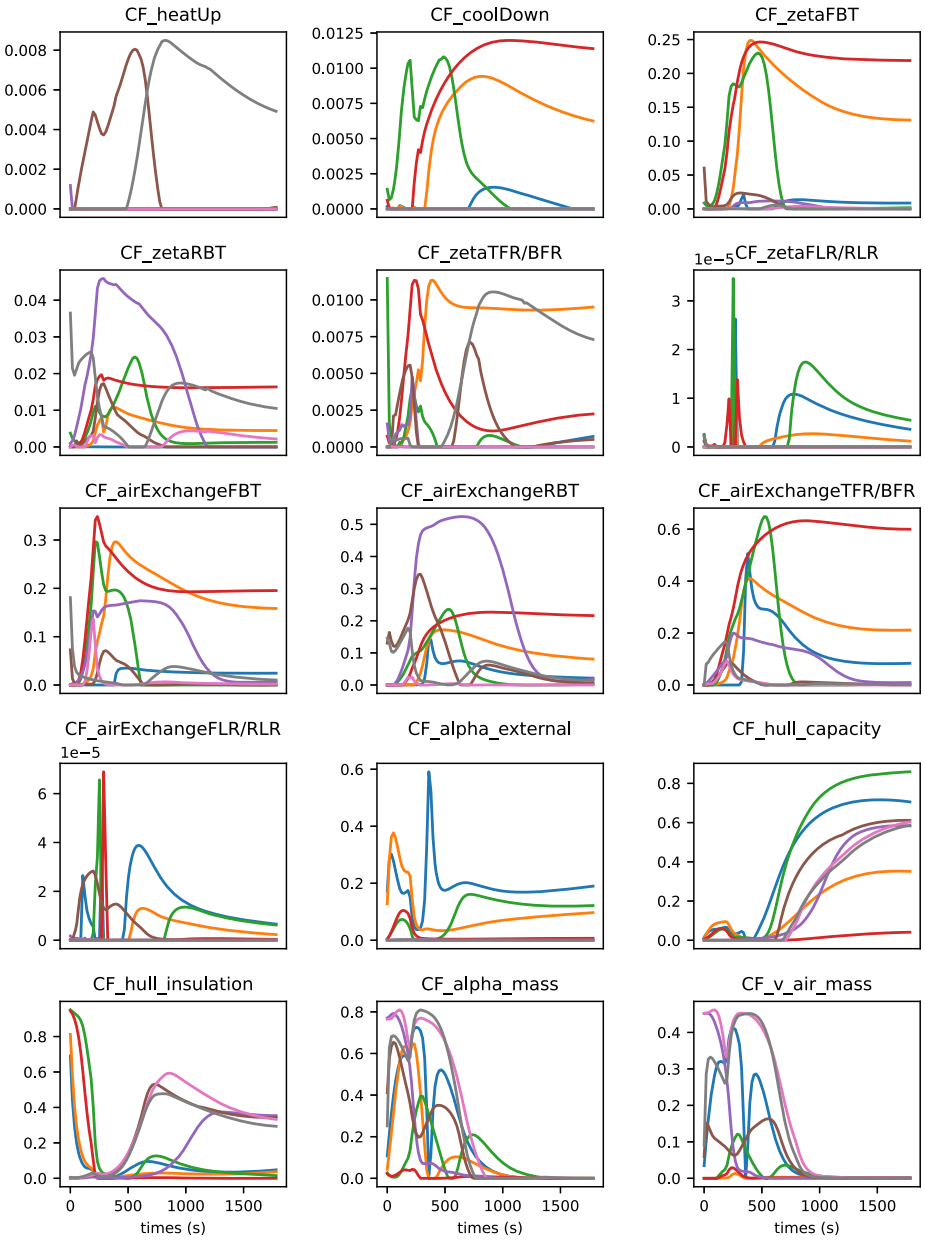


Fig. 8 Sensitivity analysis with first-order Sobol indices of 15 parameters from 8 time series with respect to time from 0 to 1800 s with step size of 18 s

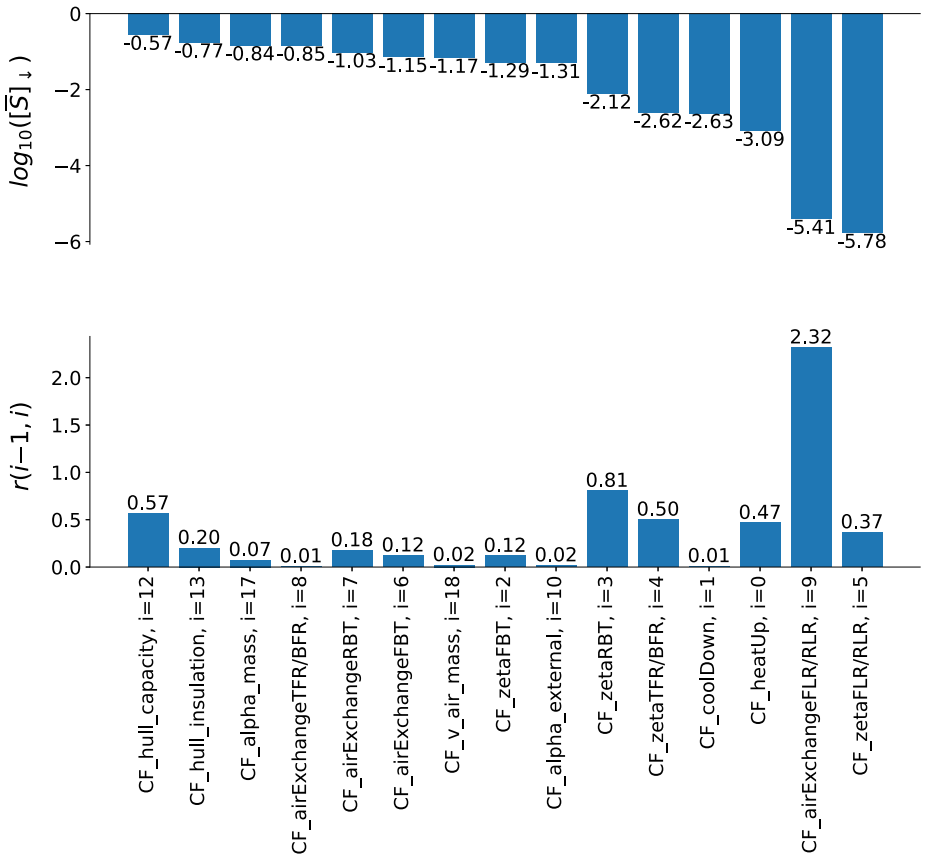


Fig. 9 Comparison of average first-order Sobol indices within 15 parameters. The first diagram displays the sorted parameters by decreasing $log_{10}([\bar{S}]_{\downarrow})$. In order to detect the sharp drop in the $\overline{S_{global,i}}$, the logarithms of relative ratios between the former one and the latter one $r(i-1, i)$ are shown in the second diagram

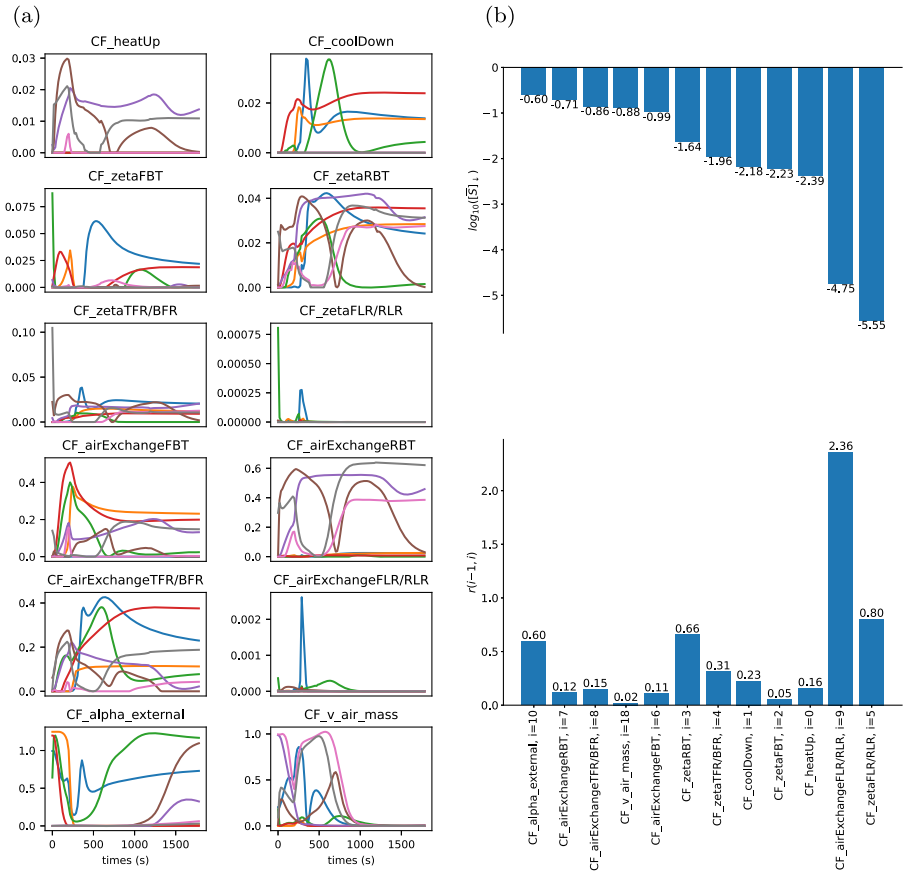


Fig. 10 Sensitivity analysis and comparison of 12 parameters. **a** Sensitivity analysis with first-order Sobol indices of 12 parameters from 8 time series with respect to time from 0 s to 1800 s with step size of 18 s. **b** Comparison of average first-order Sobol indices within 12 parameters. The first diagram displays the sorted parameters by decreasing average first-order Sobol indices. In order to detect the sharp drop in the $\overline{S_{global,i}}$, the logarithms of relative ratios between the former one and the latter one $r(i-1,i)$ are shown in the second diagram

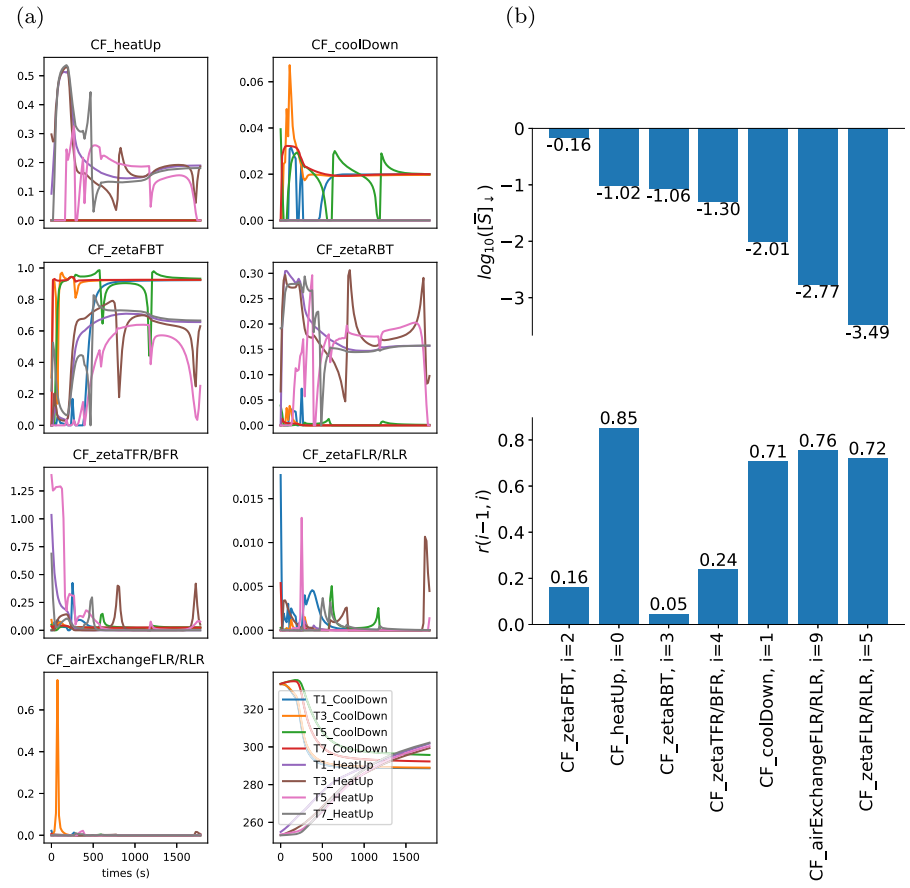


Fig. 11 Sensitivity analysis and comparison of 7 parameters. **a** Sensitivity analysis with first-order Sobol indices of 7 parameters from 8 time series with respect to time from 0 s to 1800 s with step size of 18 s. **b** Comparison of average first-order Sobol indices within 7 parameters. The first diagram displays the sorted parameters by decreasing average first-order Sobol indices. In order to detect the sharp drop in the $\overline{S_{global,i}}$, the logarithms of relative ratios between the former one and the latter one $r(i-1,i)$ are shown in the second diagram

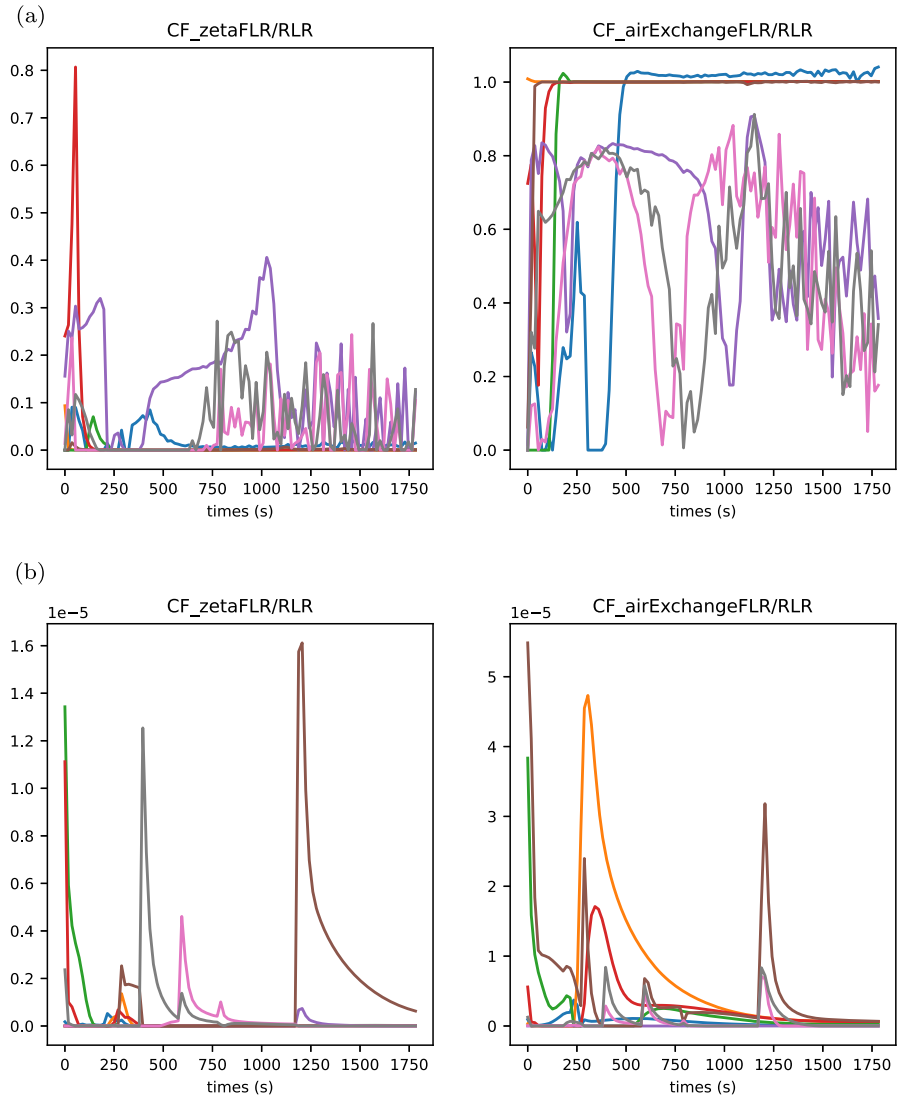


Fig. 12 Sensitivity analysis with first-order Sobol indices of 2 parameters in comparison with that in the group of 19 parameters, indicating that the sensitivity of originally weakly identifiable parameters has been obviously improved by the iterative algorithm. **a** Sensitivity analysis with first-order Sobol indices of 2 parameters from 8 time series with respect to time from 0 to 1800 s with step size of 18 s. **b** Sensitivity analysis with first-order Sobol indices of 2 parameters in the group of 19 parameters from 8 time series with respect to time from 0 to 1800 s with step size of 18 s

Results of parameter identification with BayesFlow

See Figs. 13 and 14.

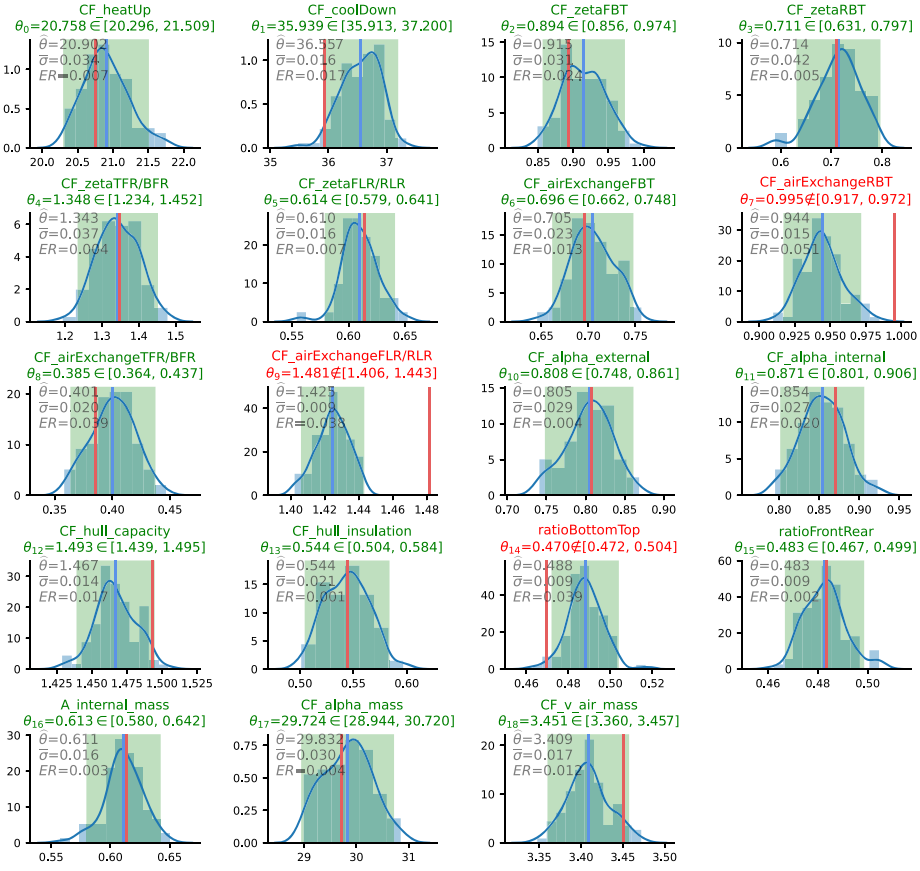
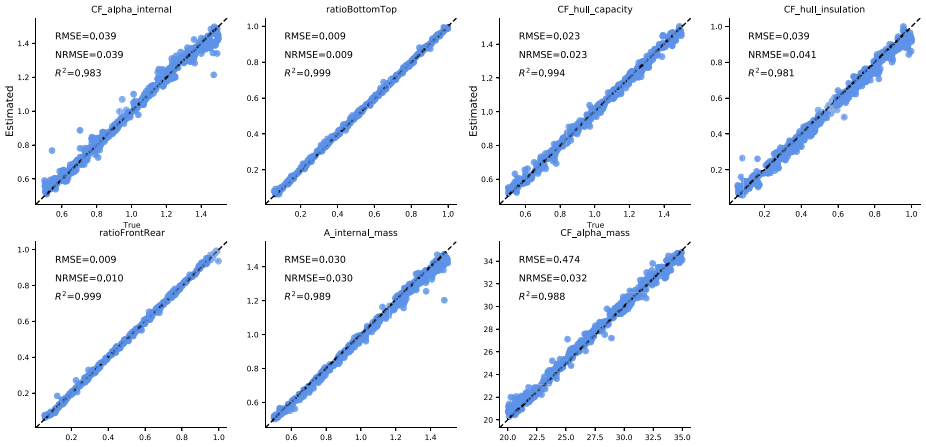
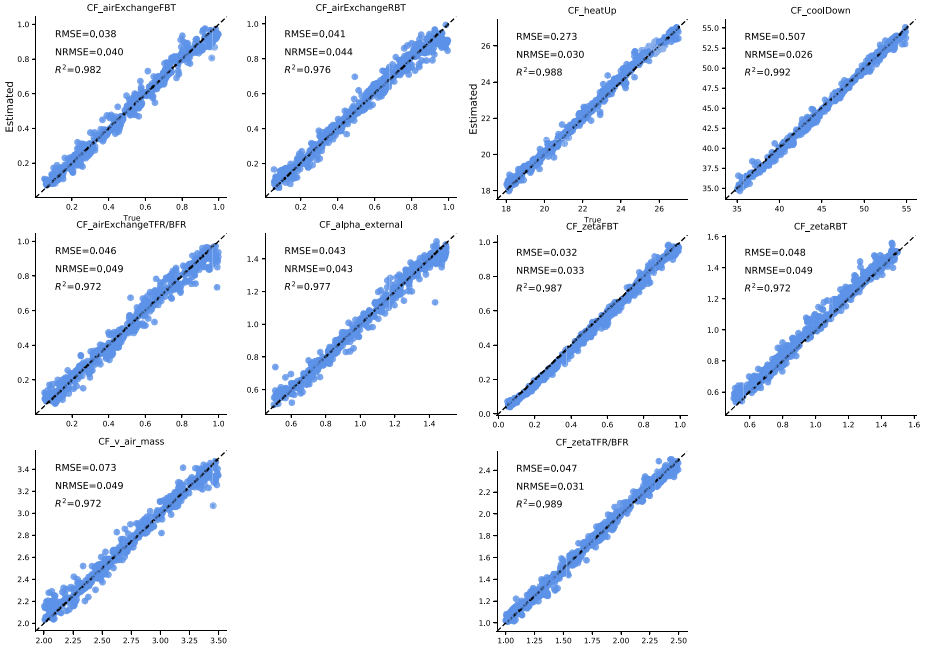


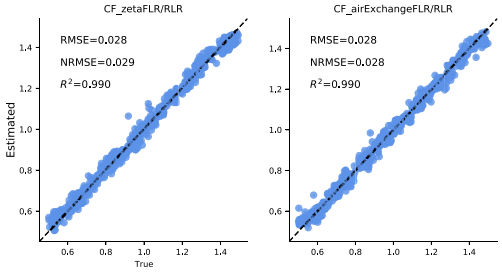
Fig. 13 Parameter identification result of a test sample. The red lines stand for the ground truth, while the blue lines are the mean values of posterior distributions induced by the well-trained BayesFlow models. The green regions are the 95% confidence intervals given by the distributions. $\widehat{\theta}_i$ denotes the estimated value of θ_i, $\overline{\sigma}$ is the standard deviation normalized by the value range of θ_i, and
$ER = |estimate - ground\ \ truth|\ /ground\ \ truth \times 100\%$ is the relative error rate. When the ground truth locates in the range of 95% confidence interval, the title including the parameter name is shown green; while the out-of-range parameters are shown in red



1. group. **2.** group.



3. group. **4.** group.



5. group.

Fig. 14 Parameter inference results of one test and 399 validation samples in 5 groups. In terms of the regression accuracy between the estimate and the ground truth of the model parameters, root mean squared error (*RMSE*), normalized root mean squared error (*NRMSE*) and coefficient of determination (R^2) are standard metrics. $NRMSE = \frac{\sqrt{\frac{1}{M}\sum_{m=1}^{M}(\theta^{(m)}-\widehat{\theta}^{(m)})^2}}{\theta_{max}-\theta_{min}}$, $R^2 = 1 - \sum_{m=1}^{M} \frac{(\theta^{(m)}-\widehat{\theta}^{(m)})^2}{(\theta^{(m)}-\overline{\theta})^2}$

Abbreviations

BNN	Bayesian neural network
cINN	Conditional invertible neural network
INN	Invertible neural network
LSTM	Long short-term memory
NF	Normalizing flow
PELS-VAE	Physics-enhanced latent space variational autoencoder
VAE	Variational autoencoder

Acknowledgements
The authors would like to thank Boris Michaelsen and Johannes Brunnemann for their support for this research.

Author contributions
The authors confirm contribution to the paper as follows: study conception and design: YZ; data collection: YZ; analysis and interpretation of results: YZ; draft manuscript preparation: YZ; critical revision of the article: LM; supervision: LM. All authors reviewed the results and approved the final version of the manuscript.

Funding
Open Access funding enabled and organized by Projekt DEAL. This research was supported by the PHyMoS Project (Proper Hybrid Models for Smarter Vehicles) funded by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) under grant number 19I20022H. The open access publication of this article was supported by the DFG sponsored Open Access Fund of the University of Augsburg.

Availability of data and materials
The copyright of the simulation model of the automobile cabin is owned by XRG Simulation GmbH. Due to the nature of this research, participants of this study did not agree for their data to be shared publicly, so supporting data is not available. Data are however available from the authors upon reasonable request and with permission of XRG Simulation GmbH.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 26 July 2022 Accepted: 26 March 2023
Published online: 24 June 2023

References

1. Boyd DW. Chapter 2–Systems modeling principles. In: Systems analysis and modeling. San Diego: Academic Press; 2001. p. 35–73. https://doi.org/10.1016/B978-012121851-5/50002-2.
2. Jospin LV, Laga H, Boussaid F, Buntine W, Bennamoun M. Hands-on Bayesian neural networks—a tutorial for deep learning users. IEEE Comput Intell Mag. 2022;17(2):29–48. https://doi.org/10.1109/MCI.2022.3155327.
3. Mitros J, Mac Namee B. On the validity of Bayesian neural networks for uncertainty estimation. arXiv preprint. 2019. arXiv:1912.01530. https://doi.org/10.48550/ARXIV.1912.01530.
4. Merkatas C, Särkkä S. System identification using Bayesian neural networks with nonparametric noise models. arXiv preprint. 2021. arXiv:2104.12119. https://doi.org/10.48550/ARXIV.2104.12119.
5. Rezende DJ, Mohamed S. Variational inference with normalizing flows. 2015. https://doi.org/10.48550/arXiv.1505.05770.
6. Kobyzev I, Prince SJ, Brubaker MA. Normalizing flows: an introduction and review of current methods. IEEE Trans Pattern Anal Mach Intell. 2020;43(11):3964–79. https://doi.org/10.1109/TPAMI.2020.2992934.
7. Papamakarios G, Nalisnick E, Rezende DJ, Mohamed S, Lakshminarayanan B. Normalizing flows for probabilistic modeling and inference. J Mach Learn Res. 2021;22(57):1–64. https://doi.org/10.48550/ARXIV.1912.02762.
8. Ardizzone L, Kruse J, Wirkert S, Rahner D, Pellegrini EW, Klessen RS, Maier-Hein L, Rother C, Köthe U. Analyzing inverse problems with invertible neural networks. arXiv preprint. 2018. arXiv:1808.04730. https://doi.org/10.48550/ARXIV.1808.04730.
9. Garcia-Hernandez EA, Elmoukrie ME, Leveneur S, Gourich B, Vernieres-Hassimi L. Global sensitivity analysis to identify influential model input on thermal risk parameters: to cottonseed oil epoxidation. J Loss Prev Process Ind. 2022;77:104795. https://doi.org/10.1016/j.jlp.2022.104795.

10. Bouchkira I, Latifi AM, Khamar L, Benjelloun S. Global sensitivity based estimability analysis for the parameter identification of Pitzer's thermodynamic model. Reliab Eng Syst Saf. 2021;207: 107263. https://doi.org/10.1016/j.ress.2020.107263.
11. Pavithra CR, Deepak T. Parameter estimation and computation of the fisher information matrix for functions of phase type random variables. Comput Stat Data Anal. 2022;167: 107362. https://doi.org/10.1016/j.csda.2021.107362.
12. Paredes-Salazar EA, Calderón-Cárdenas A, Varela H. Sensitivity analysis in the microkinetic description of electrocatalytic reactions. J Phys Chem A. 2022;126(17):2746–9. https://doi.org/10.1021/acs.jpca.2c00624.
13. Ramancha MK, Astroza R, Madarshahian R, Conte JP. Bayesian updating and identifiability assessment of nonlinear finite element models. Mech Syst Signal Process. 2022;167: 108517. https://doi.org/10.1016/j.ymssp.2021.108517.
14. Yang H, Li J, Shao C, Qian Y, Qi Q, He J. Parameter sensitivity analysis and identification of an improved symmetrical hysteretic model for rc hollow columns. Symmetry. 2022;14(5):945. https://doi.org/10.3390/sym14050945.
15. Sobol IM. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Math Comput Simul. 2001;55(1):271–80. https://doi.org/10.1016/S0378-4754(00)00270-6.
16. Radev ST, Mertens UK, Voss A, Ardizzone L, Köthe U. Bayesflow: learning complex stochastic models with invertible neural networks. IEEE Trans Neural Netw Learn Syst. 2022;33(4):1452–66. https://doi.org/10.48550/ARXIV.2003.06281. https://doi.org/10.48550/ARXIV.2003.06281https://doi.org/10.48550/ARXIV.2003.06281.
17. Martínez-Palomera J, Bloom JS, Abrahams ES. Deep generative modeling of periodic variable stars using physical parameters. arXiv preprint. 2020. arXiv:2005.07773. https://doi.org/10.48550/ARXIV.2005.07773.
18. Sun L, Wu J, Ding X, Huang Y, Wang G, Yu Y. A teacher–student framework for semi-supervised medical image segmentation from mixed supervision. arXiv preprint. 2020. arXiv:2010.12219. https://doi.org/10.48550/ARXIV.2010.12219.
19. Florens J-P, Simoni A. Revisiting identification concepts in Bayesian analysis. Ann Econ Stat. 2021;144:1–38.
20. Raue A, Kreutz C, Maiwald T, Bachmann J, Schilling M, Klingmüller U, Timmer J. Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood. Bioinformatics. 2009;25(15):1923–9. https://doi.org/10.1093/bioinformatics/btp358.
21. Wieland F-G, Hauber AL, Rosenblatt M, Tönsing C, Timmer J. On structural and practical identifiability. Curr Opin Syst Biol. 2021;25:60–9. https://doi.org/10.1016/j.coisb.2021.03.005.
22. Kabanikhin S, Bektemesov M, Krivorotko O, Bektemessov Z. Practical identifiability of mathematical models of biomedical processes. J Phys Conf Ser. 2021;2092: 012014.
23. Joubert D, Stigter H, Molenaar J. An efficient procedure to assist in the re-parametrization of structurally unidentifiable models. Math Biosci. 2020;323: 108328. https://doi.org/10.1016/j.mbs.2020.108328.
24. Gunawan R, Cao Y, Petzold L, Doyle FJ III. Sensitivity analysis of discrete stochastic systems. Biophys J. 2005;88(4):2530–40. https://doi.org/10.1529/biophysj.104.053405.
25. Caflisch RE. Monte Carlo and quasi-Monte Carlo methods. Acta Numer. 1998;7:1–49. https://doi.org/10.1017/S0962492900002804.
26. Rubinstein RY, Kroese DP. Simulation and the Monte Carlo method. 3rd ed. Hoboken: Wiley; 2016.
27. Ardizzone L, Lüth C, Kruse J, Rother C, Köthe U. Guided image generation with conditional invertible neural networks. arXiv preprint. 2019. arXiv:1907.02392. https://doi.org/10.48550/ARXIV.1907.02392.
28. Harney HL. Bayesian inference: parameter estimation and decisions. Berlin: Springer; 2003. https://doi.org/10.1007/978-3-662-06006-3.
29. Hershey JR, Olsen PA. Approximating the Kullback Leibler divergence between Gaussian mixture models. In: 2007 IEEE international conference on acoustics, speech and signal processing—ICASSP '07, vol. 4. 2007. p. 317–20 . https://doi.org/10.1109/ICASSP.2007.366913.
30. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80. https://doi.org/10.1162/neco.1997.9.8.1735.
31. Kingma DP, Welling M. Auto-encoding variational Bayes. arXiv preprint. 2013. arXiv:1312.6114. https://doi.org/10.48550/ARXIV.1312.6114.
32. Burgess CP, Higgins I, Pal A, Matthey L, Watters N, Desjardins G, Lerchner A. Understanding disentangling in β-vae. arXiv preprint. 2018. arXiv:1804.03599. https://doi.org/10.48550/ARXIV.1804.03599.
33. Luo F, Nagesh A, Sharp R, Surdeanu M. Semi-supervised teacher-student architecture for relation extraction. In: Proceedings of the third workshop on structured prediction for NLP. Association for Computational Linguistics, Minneapolis; 2019. p. 29–37. https://doi.org/10.18653/v1/W19-1505.
34. Saltelli A, Ratto M, Andres T, Campolongo F, Cariboni J, Gatelli D, Saisana M, Tarantola S. Chap. 2.4.7. Quasi-random sampling with low-discrepancy sequences. In: Global sensitivity analysis: the primer. Chichester: Wiley; 2008. p. 82–9.
35. Maclaren OJ, Nicholson R. What can be estimated? Identifiability, estimability, causal inference and ill-posed inverse problems. arXiv preprint. 2019. arXiv:1904.02826. https://doi.org/10.48550/ARXIV.1904.02826.
36. Herman J, Usher W. SALib: an open-source python library for sensitivity analysis. J Open Source Softw. 2017;2(9):1–2. https://doi.org/10.21105/joss.00097.
37. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Köpf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S. Pytorch: an imperative style, high-performance deep learning library. In: Proceedings of the 33rd international conference on neural information processing systems. Curran Associates Inc., Red Hook; 2019. https://doi.org/10.48550/ARXIV.1912.01703.
38. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado G, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I, Harp A, Irving G, Isard M, Jia Y, Jozefowicz R, Kaiser L, Kudlur M, Levenberg J, Mané D, Monga R, Moore S, Murray D, Olah C, Schuster M, Shlens J, Steiner B, Sutskever I, Talwar K, Tucker P, Vanhoucke V, Vasudevan V, Viégas F, Vinyals O, Warden P, Wattenberg M, Wicke M, Yu Y, Zheng X. TensorFlow: large-scale machine learning on heterogeneous distributed systems. 2015. https://doi.org/10.48550/ARXIV.1603.04467.
39. Michælson B, Eiden J. Humancomfort modelica-library thermal comfort in buildings and mobile applications. In: Proceedings of the 7th international Modelica conference, Como, Italy, 20–22 September 2009; 2009. p. 403–12. https://doi.org/10.3384/ecp09430082.

40. Wischhusen S. Modelling and calibration of a thermal model for an automotive cabin using humancomfort library. In: Proceedings of the 9th international MODELICA conference, September 3–5, 2012; Munich, Germany; 2012. p. 253–63. https://doi.org/10.3384/ecp12076253.
41. Heringhaus ME, Zhang Y, Zimmermann A, Mikelsons L. Towards reliable parameter extraction in MEMS final module testing using Bayesian inference. Sensors. 2022;22(14):5408. https://doi.org/10.3390/s22145408.

Publisher's Note