Reliability Engineering and System Safety 199 (2020) 106876

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

Unsupervised local cluster-weighted bootstrap aggregating the output from multiple stochastic simulators

Imad Abdallah*, Konstantinos Tatsis, Eleni Chatzi

Department of Civil, Environmental and Geomatic Engineering, ETH Zürich Stefano-Franscini-Platz 5, Zürich 8093, Switzerland

ARTICLE INFO

Keywords:
Ensemble learning
Clustering
Variational Bayesian Gaussian mixture
Boostrap Aggregation (Bagging)
Stochastic simulators
Model Uncertainty

ABSTRACT

In the present work, we consider the problem of combining the output from multiple stochastic computer simulators to make inference on a quantity of interest, as a means of reducing the inherent model-form uncertainty in the absence of any measurements. In most real-world situations, judging an individual stochastic simulator to be the "best" for any given point in the input space is highly doubtful. Thus, making inference by relying on the so-deemed best simulator may not be adequate, especially when the sampled data is limited. To this end, we propose an ensemble learning method based on local *Clustering* and bootstrap aggregation (*Bagging*), which rather than treating the stochastic predictions of the simulators as competing individual information sources, treats those as part of an ensemble, thus diversifying the hypothesis space. We call the proposed method: unsupervised local cluster-weighted bootstrap aggregation. Variational Bayesian Gaussian mixture clustering is the first step in this ensemble learning approach for discriminating the outputs, and deriving the probability map (weights) of the clustered simulators output. Clustering is performed on the stochastic output corresponding to the binned input space. Performing the clustering independently and deriving the probability map for each local region of the binned input space is a novelty that guarantees an adaptive solution, whereby certain simulators are potentially more fitting than others in corresponding regions of the input space. The second step consists in a local cluster-weighted Bootstrap Aggregation, which serves the purpose of weighted combination of the clustered ensemble of outputs from the individual simulators. Based on simulations, we demonstrate how the input bin size, sample size, output dispersion and level of agreement amongst the simulators affect the performance of the proposed method. We compare the unsupervised local cluster-weighted bootstrap aggregation method to classical Bagging, Bayesian Model Averaging and Stacking of predictive distributions. Finally, we demonstrate the method by evaluating the fatigue damage equivalent load on a wind turbine blade, using 10 finite element based simulators. The results point to the need for practitioners to consider this as a useful method, when model-form uncertainty is of concern and when output from multiple stochastic simulators are available.

1. Introduction

1.1. Problem statement

Complex systems are studied by mathematical models and implemented as computer simulators. In many instances, relating to critical engineering analysis tasks, predictions from multiple stochastic computer simulators are available. These simulators reflect the state-of-the-art, have been tested, verified, validated and calibrated, and have been adopted by decision makers to solve real-world problems. Attention, however, is generally directed towards comparing and benchmarking simulators against each other in order to establish a single "true" or "best" simulator (e.g. [1]). In most real-world situations, however, the existence of a "true" simulator is highly doubtful at best, especially when no simulator may emerge as clearly superior in terms of accuracy and precision when considering all possible combinations of the input parameters, boundary conditions, operating conditions and loading patterns. This is rendered especially complicated when detailed measurements are unavailable. Consequently, basing inference regarding a quantity of interest on a single "best" simulator may underestimate the model-form uncertainty. In this paper, we thus address the problem of combining the output from multiple stochastic simulators to make inference on a quantity of interest, as a means of reducing the model-form uncertainty in the absence of any measurements.

* Corresponding author.
E-mail address: abdallah@ibk.baug.ethz.ch (I. Abdallah).

https://doi.org/10.1016/j.ress.2020.106876
Received 26 February 2019; Received in revised form 10 December 2019; Accepted 14 February 2020
Available online 02 March 2020

1.2. Contributions and novelties of this research

There is mounting evidence to suggest that aggregating the output from a diverse set of simulators may provide better predictive ability and lower model-form uncertainty than adoption of a any single simulator [2,3]. This is very much in the spirit of model combination and model averaging. The majority of model averaging or model aggregation methods in the literature assign weights that do not distinguish between different regions of the covariate input space, and make an inherent assumption that one of the models in the ensemble is the actual data-generating model. Because of these reasons we turn to alternatives. We propose an unsupervised ensemble learning approach that rather than treating the predictions of the simulators as competing individual information sources, considers these as part of an ensemble. Based on the stochastic output from a diverse set of simulators, the approach consists in first deriving the simulators local weights using variational Bayesian Gaussian mixture clustering (VBGM), and subsequently performing a weighted ensemble aggregation on the stochastic output via Bootstrap Aggregation, also known as Bagging. Abdallah et al. [4], [5] demonstrated the potential of this method in early works, and here we present a complete treatise of the framework. With very few assumptions, our method captures both model form uncertainty (often an overlooked topic in uncertainty quantification), as well as epistemic uncertainty induced by the limited output from the stochastic simulators. We call our method: "Unsupervised Local Cluster-Weighted Bootstrap Aggregation". Compared to state-of-the-art, the primary contributions of the paper are:

(i) **Local assignment of weights:** classical model averaging or aggregation methods assign a single weight to individual simulators according to their performance over the whole input space, which can be affected by sample bias or outliers in the data. Our method avoids this issue by assigning weights independently for each local region of the binned input space. This guarantees a solution whereby certain simulators are more fitting than others in certain regions of the input space.
(ii) **Local cluster-weighted combinations of the output:** Following the assignment of weights, classical model averaging or aggregation methods mix the individual simulators via convex linear combinations over the whole input space. Our method does not assign weights to individual simulators per se, rather it assigns weights to clusters of output, i.e., to the collection of "similar" output data originating from the various simulators in each local region of the binned input space. The clustered output are then combined via the cluster-weighted bootstrap aggregation step for each local region of the binned input space.
(iii) **Assimilation of measurements replications:** Our method seamlessly assimilates any number of measurements replications without any further modifications to the method. Each measurement replication is considered as a "simulator" and added to the bucket of available simulators and are not labelled as the ground truth as is the norm in classical model averaging or aggregation methods when updating the weights.

1.3. Related work

Methods that have been used for combining the output from multiple simulators include the Multivariate Normal Aggregation method [6,7], which aims at combining the normally distributed output from various simulators, while introducing dependence via a covariance matrix, resulting in effect in a multivariate Normal distribution of the output of individual simulators. A natural extension of this method was originally proposed by Jouini and Clemen [8] by using the Copula formalism instead to capture the dependence structure amongst the simulators. The Adjustment Factor Approach is another method that makes use of an adjustment factor that is added (or multiplied) to the best model amongst all models considered [9]. The weight of the simulators is assigned by expert opinion based on the merit and accuracy of each individual simulator. The method however assumes that the outputs from individual simulators are Normally, Lognormally or Beta distributed and the adjustment factor is normal or lognormal.

Bayesian Model Averaging (BMA) is a popular method for combining predictions from numerical models through a model averaging procedure [10]. BMA accounts for model uncertainty, as distinct from parameter uncertainty, by integrating over the model space and weighting each individual model by the estimated probability of being the correct model. The posterior weights are derived and updated quantitatively using measurements. In a Bayesian framework the weighting factors become Posterior Model Probabilities (PMP), which involves the calculations of a marginal likelihood and must be approximated using one of many approaches, see for example [11]. The final predictions of BMA result is a weighted average of the set of model predictions, which Hoeting et al. [12] and more recently Körner et al. [13] (amongst many) demonstrated to provide better average predictive ability than using any single model for certain applications. Rings et al. [14] introduced a variant of BMA with a joint particle filtering and Gaussian mixture modeling framework to derive the evolving forecast density of each constituent ensemble member. These distributions are subsequently combined with BMA and used to derive one overall predictive distribution. In a recent work, Liu et al. [15] proposed a Bayesian model averaging based reliability analysis method for monotonic degradation dataset based on inverse Gaussian process and Gamma process. Using BMA, they successfully introduced both the parametric and model uncertainties in the reliability analysis. Yu et al. [16] advanced the BMA by using the concept of local Bayes factors by restricting the models to regions of the covariate space. An algorithm is then deployed for model combination, where local Bayes factors are used to guide the weighting of the Bayesian models. However, the criticism of BMA is that it tries to assign weights presuming that one of the models in the ensemble is the actual data-generating model. Essentially, implicit to BMA is a model selection problem, which would work well if the ensemble were big enough to sample the entire model-space, but such is impossible for practical engineering applications. Alternatively, Bayesian Model Combination (BMC; [17]) provides an algorithmic correction to BMA in order to allow for the selection from an ensemble of combinations of the individual simulators, whose weights are obtained by sampling from a Dirichlet distribution with uniform parameters. BMC marginalizes over the uncertainty in the correct model combination, where-as BMA marginalizes over the uncertainty in identifying the correct model from the entire ensemble. BMC has proven to be superior to BMA and Bagging for certain applications.

In the domain of reliability engineering, an new sampling based approach has been recently proposed by Peherstorfer et al. [18], which combines multiple surrogate models to accelerate failure probability estimation. The method is based on mixed multifidelity importance sampling that leverages computationally cheap but erroneous models for the construction of the biasing distribution and that uses the original high-fidelity model to guarantee unbiased estimates of the probability of failure using Importance sampling. An appealing feature of this approach is that it avoids the problem of model selection, while guaranteeing a small mean squared error. Along the same logic, Cheng and Lu [19] adopted an adaptive approach for reliability analysis by ensemble learning of multiple competitive surrogate models, including Kriging, polynomial chaos expansion and support vector regression. Similarly, Strömberg [20] proposed a convex ensemble of metamodels families to better represent the limit state surfaces in reliability based design optimizations. Bayesian multimodel and deep neural network approaches were deployed to perform reliability analysis and assess model-form uncertainties in [21–24].

Finally, numerous methods fall in the so-called class of ensemble learning techniques that include, but are not limited to, Stacking [25],

Bagging [26], Boosting, modified adaptive boosting or Adaboost [27], and XGBoost [28,29]. The general principle of ensemble learning methods is to construct a combination of some model fitting method from a concrete finite set of alternative models, instead of adopting a single fit of the method [30]. For instance, Li et al. [31] successfully demonstrated a dynamic weight allocation method based on the Adaboost ensemble learning algorithm to train several multi-layer perceptron neural networks for remaining useful life prediction. These techniques are effective at reducing the error rate of unstable learners, in the sense that small variations in the training set can lead them to produce very different models.

1.4. Paper organization

The remainder of this article is organized as follows. In Section 2 we provide an overview of the proposed method. We revisit the theory of Variational Bayesian Gaussian Mixture Clustering, and then derive the probability maps in Section 3. In Section 4 we describe how the local cluster-weights are introduced to the Bootstrap Aggregation. We demonstrate the novelty and principles of the proposed framework with a set of computational experiments in Section 5. In Section 5.4 we compare how local cluster-weighted Bagging and selected existing methods compare. Finally, in Section 6 we illustrate the framework on a practical engineering application.

2. Overview of the algorithm

Assume that stochastic predictions of a quantity of interest $\mathcal{Y}$ are available as a function of $\boldsymbol{x}$ from several simulators. We let $\boldsymbol{x} = \{x_1, ..., x_d\}^T$ be a d-dimensional vector of input variables. This input vector is sampled at N distinct locations in the input space $\mathcal{D}_{\boldsymbol{x}}$ and the corresponding scalar stochastic output realizations are $\{\mathcal{Y}^{(i)}, i = 1, ..., N\}$. The N distinct input samples are collected in matrix $\mathcal{X} = (\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(N)})$, and the output in vector $\mathcal{Y} = \{\mathcal{Y}^{(1)}, ..., \mathcal{Y}^{(N)}\}^T$. For a given simulator $\{S_l, l = 1, ..., s\}$ the vector of stochastic output is $\mathcal{Y}_l$, s being the total number of available simulators. Given this scenario, the two central questions we strive to answer are: how to take advantage and make use of the output from the various stochastic simulators? How can predictions from multiple simulators, demonstrably, result in reduced model-form uncertainties? To answer these questions we propose Algorithm 1, which we call "Unsupervised Local Cluster-Weighted Bootstrap Aggregation". The algorithm starts by discretising the input space into a finite number of bins. The local learning set is all the data falling within a specific bin. Unsupervised Variational Bayesian Gaussian Mixture clustering (VBGM) is then applied to data in each bin, and serves the purpose of distinguishing the output from individual simulators, and deriving the probability map (weights). VBGM is chosen because the number of clusters is an optimization variable and is not given as input in advance. Performing the clustering independently for each local region of the binned input space guarantees an adaptive solution, whereby certain simulators are more fitting than others in corresponding regions of the input space. An intermediate step examines the cluster stability [32] on perturbed versions of the data in a given bin. The reason for performing cluster stability analysis is that, as opposed to supervised classification, there is no ground truth against which we could test our clustering results. The final step of the algorithm is the local cluster-weighted Bootstrap Aggregation (Bagging), which serves the purpose of weighted combination of the clustered ensemble of outputs from the individual simulators.

3. Variational Bayesian Gaussian mixture clustering

The clustering step serves the purpose of (1) distinguishing the output from individual simulators and (2) deriving an apriori probability map (weights) of the clustered simulators output. A number of clustering algorithms has been proposed in the literature, such as k-means, Gaussian Mixtures, Hierarchical clustering, self organizing maps, and deep unsupervised clustering with Gaussian Mixture Variational Autoencoders. Here we chose a Bayesian treatment using Variational Bayesian Gaussian Mixture Clustering (VBGM).

3.1. Overview of VBGM

Finite Gaussian mixtures are a flexible probabilistic modeling tool for irregularly shaped densities and data samples from heterogeneous stochastic populations generated from a mixture of a finite number K of Gaussian distributions with unknown parameters:

$$p(\mathcal{Y}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathcal{Y}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{1}$$

where π_k is the mixing coefficient, and each Gaussian density $\mathcal{N}(\mathcal{Y}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is a component of the mixture comprising its own parameters: mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$ [33]. The traditional approach to estimating the parameters is by maximizing the likelihood function of the Gaussian mixture:

$$p(\mathcal{Y}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{N} \left[\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathcal{Y}^{(i)}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right] \tag{2}$$

which is not always a well posed problem because of the singularities that will always occur whenever one of the Gaussian components "collapses" onto a specific data point in the dataset (see [33] for proof). Training of Gaussian mixture models suffers the problem that most unsupervised learning algorithms face; Knowing which component a sample belongs to, we can use the Maximum Likelihood Estimation (MLE) estimates to update the component. Conversely, knowing the parameters of the components we can predict where each sample lies within each component. This is solved using the expectation-maximization (*EM*) algorithm [34], which iterates between the two until convergence. *EM* is a well-founded statistical algorithm for getting around these problems by means of an iterative process in order to determine maximum likelihood solutions (for details see [33] or [35]). A limitation of *EM* is that it requires a proper/multiple initialization(s) in order to consistently find good maximum likelihood solutions. Another limitation is that there is no clear guidance on the choice of the number of Gaussian mixtures (components) K to be used. The Bayesian variational treatment of the Gaussian mixture model circumvents this by automatically providing the number of clusters determined by model selection. The variational framework can be viewed as a complementary approach to that of Markov Chain Monte Carlo (MCMC), which delivers considerable computational advantages at the cost of not being asymptotically exact [36]. In practical terms, the ultimate objective is to cluster the data into K components, each of which comprises a mixing coefficient $\{\pi_k, k = 1, ..., K\}$. The end goal is to evaluate the posterior distribution:

$$P(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{Y}) = \frac{P(\boldsymbol{Y}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})}{P(\boldsymbol{Y})} = P(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \frac{P(\boldsymbol{Y}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})}{P(\boldsymbol{Y})} \tag{3}$$

which is generally intractable. Variational methods are rather used in order to define a tractable lower bound on $P(\boldsymbol{Y})$, by minimizing the Kullback–Leibler divergence $KL(q(\boldsymbol{\Theta})||P(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{Y}))$. Replacing the posterior distribution, it turns out that the following approximation holds true [37,38]:

$$\begin{aligned} ln\, P(\boldsymbol{Y}) &= ln \frac{P(\boldsymbol{Y}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})}{P(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\,|\,\boldsymbol{Y})} \\ &= \underbrace{\mathcal{F}(q(\boldsymbol{\Theta}))}_{\text{Neg. free energy}} + \underbrace{KL(q(\boldsymbol{\Theta})||P(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{Y}))}_{\text{KL divergence}} \end{aligned} \tag{4}$$

where $\mathcal{F}(q(\boldsymbol{\Theta}))$ is the so-called negative free energy, which is a lower bound approximation of $P(\boldsymbol{Y})$. The variational method involves the introduction of a distribution $q(\boldsymbol{\Theta})$, which provides an approximation to

Input : $\{\mathcal{Y}_l\}_{l=1}^{s}$: vector of stochastic observations from each simulator $\{\mathcal{S}_l\}_{l=1}^{s}$, $nMCs >> 1$: Number of repeats, $\boldsymbol{\mathcal{X}} = \left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{N}$: d-dimensional explanatory input variables i.i.d. draws from the joint distribution, Δx: input bin size, Q: a large pre-specified number, M: a large pre-specified number

Result: $\hat{g}_{w,bagg}$: unsupervised local cluster-weighted Bagged estimator

Set $\boldsymbol{\mathcal{Z}} = \left[\boldsymbol{\mathcal{X}}^T \left\{\mathcal{Y}_l^T\right\}_{l=1}^{s}\right]$;

for $iter = 1, \ldots, nMCs$ **do**

  Randomly permute the cases in $\boldsymbol{\mathcal{Z}}$, and then select n training samples such that $n = 0.7N$;

  Assign every $\left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{n}$ to be the center of bins of size Δx;

  foreach Δx **do**

    // Local Clustering

    for $q = 1, ..., Q$ **do**

      Randomly draw 90% of local responses of simulators $\left\{\mathcal{Y}_l^T\right\}_{l=1}^{s}$ without replacement;

      Identify local clusters $C_q\left(\left\{\mathcal{Y}_l^T\right\}_{l=1}^{s}, K_q \mid \Delta \boldsymbol{x}\right)$, using Variational Bayesian Gaussian Mixture;

      Compute local weights $P_q\left(\mathcal{S}, C \mid \Delta \boldsymbol{x}\right)$;

    end

    // Expected local cluster weights

    Cluster stability: Use majority vote $out - of - Q$ to establish the expected local number of clusters $K_{\Delta \boldsymbol{x}}$;

    Compute expectation of local weights: $P\left(\mathcal{S}, C \mid \Delta \boldsymbol{x}\right) = \mathbb{E}\left[P_q\left(\mathcal{S}, C \mid \Delta \boldsymbol{x}\right)\right]_Q$;

    // Local weighted Bagging

    for $k = 1, ..., M$ **do**

      Construct a weighted bootstrap sample $\left(\boldsymbol{x}_*^{(1)}, \mathcal{Y}_*^{(1)}\right), \ldots, \left(\boldsymbol{x}_*^{(n_{\Delta x})}, \mathcal{Y}_*^{(n_{\Delta x})}\right)$ by randomly drawing $n_{\Delta x}$ times with replacement from the local clustered data, the local weights being $P\left(\mathcal{S}, C \mid \Delta \boldsymbol{x}\right)$;

      Compute the bootstrapped estimator $h_n(\cdot)$:

$$\hat{g}_*^k = h_n\left(\left(\boldsymbol{x}_*^{(1)}, \mathcal{Y}_*^{(1)}\right), \ldots, \left(\boldsymbol{x}_*^{(n_{\Delta x})}, \mathcal{Y}_*^{(n_{\Delta x})}\right)\right)$$

    end

    Compute the local cluster-weighted Bagged estimator is:

$$\hat{g}_{w,bagg} = \frac{\sum_{k=1}^{M} \hat{g}_*^k}{M}$$

  end

end

Algorithm 1. Local cluster-weighted bootstrap aggregating output from stochastic simulators.

the true posterior distribution, where $\boldsymbol{\Theta}$ is a vector collecting all parameters. $q(\boldsymbol{\Theta})$ generally corresponds to a simple parametric family of the posterior probability density. In this regard, the Kullback–Leibler divergence is often chosen as a relative measure of the dissimilarity of the two probability densities $p(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{Y})$ and $q(\boldsymbol{\Theta})$[39]. Minimization of $KL(q(\boldsymbol{\Theta})||P(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{Y}))$ reduces the divergence between the true posterior $p(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{Y})$ and its approximation $q(\boldsymbol{\Theta})$. The problem then reduces to finding the set of probability densities $q(\boldsymbol{\Theta})$ that maximize the lower bound $\mathcal{F}(q(\boldsymbol{\Theta}))$, which is equivalent to minimizing the $KL(q(\boldsymbol{\Theta})||P(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{Y}))$ or tightening $\mathcal{F}$ as a lower bound to the log model evidence $\ln P(\boldsymbol{Y})$. In this work, the algorithm in [40] is adopted to solve the Variational Bayesian Gaussian Mixture inference in order to identify the posterior distribution of the clusters in the binned input space. The choice of approximate posterior distribution is one of the core problems in variational inference. Instead, Normalizing Flows is a potential alternative for specifying flexible, arbitrarily complex and scalable posterior distributions compared to the known limitations of variational inference [41,42]. It must be noted that we introduce an intermediate step to examine the cluster stability on perturbed versions of the data in a given bin. A simple approach is adopted, where we repeatedly draw sub-samples from the population and apply the Variational Gaussian Mixture clustering algorithm, and subsequently perform a majority vote

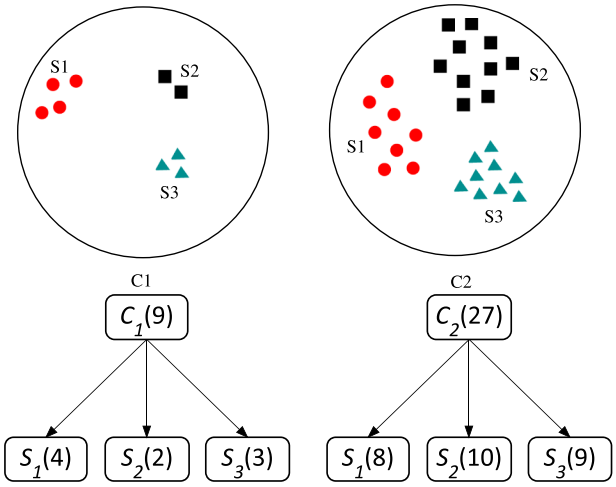


Fig. 1. Illustration of 2 clusters (C_1, C_2) of the stochastic output from 3 simulators (S_1, S_2, S_3), each consisting of 12 data points.

to select the most commonly occurring clusters mixture structure.

3.2. Probability map (weights)

We elaborate on how we derive the probability map (weights) of the clustered simulators output, using a simple illustrative example, where the stochastic output from 3 simulators (each individual simulator consists of twelve data points) are assumed to be clustered as shown in Fig. 1.

The weights should not be tied to the total probability of a simulator as this is not very relevant. Why? Considering we have 3 simulators, then the $P(S_1) = 1/3$, $P(S_2) = 1/3$ and $P(S_3) = 1/3$, which does not offer any additional information. On the other hand, the weights should not be assigned by the conditional probabilities $P(S_i|C_j, \Delta\boldsymbol{x})$, since this ignores the probability of a cluster itself. The idea would be to push the aggregate toward larger density clusters (i.e., areas of stronger agreement), allowing the simulators to reinforce each other when consensus exists, or, conversely, negate each other when there is no consensus. Based on the clusters illustrated in Fig. 1, we can calculate the weights, which are the joint probability $P(S_i, C_j|\Delta\boldsymbol{x})$ of a clusters $\{C_j; j = 1, 2\}$ and a simulators $\{S_i; i = 1, 3\}$:

$$P(S_i, C_j|\Delta\boldsymbol{x}) = P(S_i|C_j, \Delta\boldsymbol{x})\cdot P(C_j|\Delta\boldsymbol{x}) \tag{5}$$

where,

$$\begin{aligned} P(C_j|\Delta\boldsymbol{x}) &= \frac{N_{C_j|\Delta\boldsymbol{x}}}{N} \\ P(S_i|C_j, \Delta\boldsymbol{x}) &= \frac{N_{S_i|C_j,\Delta\boldsymbol{x}}}{N_{C_j|\Delta\boldsymbol{x}}} \end{aligned} \tag{6}$$

$N_{C_j|\Delta\boldsymbol{x}}$ is the number of data points in cluster C_j and N is the total number of all available output samples from all simulators in $\Delta\boldsymbol{x}$, and $N_{S_i|C_j,\Delta\boldsymbol{x}}$ is the number of data points in cluster C_j corresponding to simulator S_i. The outcome is shown in Fig. 1. In cluster 1, 4 data points correspond to simulator 1, 2 data points correspond to simulator 2 and 3 data points correspond to simulator 3. In cluster 2, 8 data points correspond to simulator 1, 10 data points correspond to simulator 2 and 9 data points correspond to simulator 3. Finally, the joint probabilities of a individual clusters and simulators are shown in Table 1.

Table 1
Weights of the clustered simulators output.

	S_1	S_2	S_3
C_1	0.111	0.056	0.083
C_2	0.222	0.278	0.250

4. Local cluster-weighted Bootstrap Aggregating

The clustering step is followed by the local cluster-weighted Bootstrap Aggregating (Bagging [26]) step, which serves the purpose of weighted combination of the ensemble of outputs from the individual simulators. Bagging predictors comprise a method for generating multiple versions of a predictor, each on random subsets of the original dataset, and fusing these into a unique final aggregated predictor. This aggregated predictor can typically be used for reducing the variance of black-box estimators, by introducing randomization into the construction procedure and forming an ensemble. The bagging algorithm consists in:

(i) constructing a weighted bootstrap sample $(\boldsymbol{x}_*^{(1)}, \mathcal{Y}_*^{(1)}), \ldots, (\boldsymbol{x}_*^{(n_{\Delta\boldsymbol{x}})}, \mathcal{Y}_*^{(n_{\Delta\boldsymbol{x}})})$ by randomly drawing $n_{\Delta\boldsymbol{x}}$ times with replacement from the local clustered data $(\boldsymbol{x}^{(1)}, \mathcal{Y}^{(1)}), \ldots, (\boldsymbol{x}^{(n_{\Delta\boldsymbol{x}})}, \mathcal{Y}^{(n_{\Delta\boldsymbol{x}})})$, the local weights being $P(S_i, C_j|\Delta\boldsymbol{x})$,
(ii) computing the bootstrapped estimator: $\hat{g}_* = h_n((\boldsymbol{x}_*^{(1)}, \mathcal{Y}_*^{(1)}), \ldots, (\boldsymbol{x}_*^{(n_{\Delta\boldsymbol{x}})}, \mathcal{Y}_*^{(n_{\Delta\boldsymbol{x}})}))$, where $h_n(\cdot)$ defines the estimator as a function of the data,
(iii) repeating steps 1 and 2 M times, where M is often chosen between 50 and 100, yielding $\{\hat{g}_*^k, k = 1, \ldots, M\}$ and the weighted bagged estimator is defined as the expectation of all the bootstrap estimators:

$$\hat{g}_{w,bagg} = \frac{\sum_{k=1}^{M} \hat{g}_*^k}{M} \tag{7}$$

The estimator $h_n(\cdot)$ may represent a function that computes the expected value (or any other quantile), it may be a function that fits a probability distribution, or may be a function that fits surrogate, etc. Therefore, $h_n(\cdot)$ can be any learning algorithm $\Psi: \gamma \rightarrow \boldsymbol{\phi}$ that given any input data $L \in \gamma$, it produces a predictor $\Phi = \Psi(L) \in \boldsymbol{\phi}$. $h_n(\cdot)$ is designated as the base learner when it is trained with the whole original dataset. Bagging has been well acknowledged to achieve better performance than the base learner, especially when the base learner is unstable with respect to the random training data [30].

5. Computational experiments

We present simulated analytical examples to motivate the approach, and demonstrate the principles and improvements that are possible with the local cluster-weighted bootstrap aggregation algorithm. Suppose that a target (true) expected response is given by the following analytical function:

$$y_{target}(x) = (6x - 2)^2 \sin(12x - 4) + 12 \tag{8}$$

where the input parameter x varies over [0, 1]. Furthermore, we assume that the response is stochastic and Normally distributed:

$$P(\mathcal{Y}_{target}|X) \sim \mathcal{N}(y_{target}(x), COV = 0.05) \tag{9}$$

where the coefficient of variation COV is assumed to be homoscedastic, and random variable X is uniform, $X \sim U(0, 1)$. We propose five synthetic analytical stochastic simulators aiming at predicting $\mathcal{Y}_{target}$. In this demonstration, the local cluster-weighted bagged estimator is the expected value of the stochastic response conditional on binned input space Δx and the output from the five stochastic simulators: $\mathbb{E}[\mathcal{Y}|\Delta x, \mathcal{Y}_1, \mathcal{Y}_2, \mathcal{Y}_3, \mathcal{Y}_4, \mathcal{Y}_5]$. We compare our results to classical bagging. The comparison is based on the generalization error (GE) with respect to the known target response and change in the 95% empirical bootstrap confidence interval (CI). The GE is a global metric of the accuracy of the the predictions. It is computed as:

$$GE = \frac{\sum_{i=1}^{n_v} [y^{(i)} - \hat{g}_{w,bagg}(\boldsymbol{x}^{(i)})]^2}{\sum_{i=1}^{n_v} [y^{(i)} - \mu_y]^2}, \quad \mu_y = \frac{1}{n}\sum_{i=1}^{n_v} y^{(i)} \tag{10}$$

where n_v is the size of the validation set, and μ_y is the mean of the computer simulator response for the validation set. On the other hand, the 95% CI reflects the local predictive precision. In the following case studies, we analyse the performance when varying the number of simulators in agreement with the target response, bias of simulators with respect to target response, COV, number of stochastic samples, and bin sizes Δx. $M = 100$ Bootstraps are used in the weighted Bagging step. The results of the evaluations are averaged across 50 repetitions, so that they are not susceptible to a specific split of the sampled data.

5.1. Case study 1: simulators in varying agreement and bias levels with target response

Scenario 1: In this scenario 4 out of 5 simulators are in perfect agreement amongst each other and with the target response, i.e., $\{P(\mathcal{Y}_i|X) \sim \mathcal{N}(y_{target}(x), COV = 0.05), i = 1, ...,4\}$. Simulator 5 is in disagreement with the target, and its expected response is described by the transformation function:

$$y_5(x) = Ay_{target}(x) + B(x - 0.5) - C;\ P(\mathcal{Y}_5|X) \sim \mathcal{N}(y_5(x), COV = 0.05) \tag{11}$$

where the constants $A = 0.5$, $B = 10$ and $C = 20$. We choose the constants such that $y_5(x)$ largely displays the same trend as y_{target} but exhibits a large bias. The performance of the local cluster-weighted Bagged estimator is evaluated when 2000 observational samples are available from each simulator, and $\Delta x = 0.02$. The expected values of the target and the five simulators are shown in Fig. 2a. The formation of clusters conditional on the binned input space is the first step in the proposed ensemble learning framework. An example of clustering and their corresponding log-evidence for $x \in [0.6, 0.62]$ are illustrated in Fig. 4a and b, respectively. As per intuition, for $x \in [0.6, 0.62]$ two clusters, comprising the highest log-evidence, are the most likely grouping of the output from the 5 simulators, yielding weights $P(S_i, C_j|X)$ as shown in Table 2.

The weights $\{P(S_i, C_j|X)i = 1, ...,5, j = 1, 2\}$ for each Δx in the input space are then used in the cluster-weighted Bagged estimation step. The cluster-weighted Bagged estimator, shown in Fig. 2b, yields a notably improved predictor of the target (true) response compared to classical Bagging. We report that the generalization error is reduced by 90%, while the 95% confidence interval is reduced by 40% averaged across the binned input space x. A reduction in the GE translates into a better accuracy of the estimator, while a reduction in the CI, for a 95% confidence level, translates into a better precision of the estimator.

Table 2
Weights $\{P(S_i, C_j|X)i = 1, ...,5, j = 1, ...,2\}$ for $x \in [0.6, 0.62]$.

	S_1	S_2	S_3	S_4	S_5
Cluster 1	0.0	0.0	0.0	0.0	0.2
Cluster 2	0.2	0.2	0.2	0.2	0.0

Furthermore, we investigate the cases when only 3 out of 5, 2 out of 5 and 1 out of 5 simulators are in perfect agreement with the target response. The results shown in Fig. 3b indicate that the GE of the cluster-weighted Bagged estimator drops by $40 - 90\%$ compared to the classical Bagged estimator when the majority of simulators agree with the target response. Otherwise, the classical Bagged estimator exhibit a better generalization error. Fig. 3a shows that the 95% CI decreases by $35 - 40\%$ when the majority of simulators are in mutual agreement and not necessarily in agreement with the target response. When 2 out of 5 or 3 out 5 simulators are in mutual agreement and further in agreement with the target response (i.e., no clear established majority), the reduction in confidence interval is rather marginal and of the order of 5%. Those results can be interpreted by the fact that the weights $P(S_i, C_j|X)$ are high when the majority of simulators are in mutual agreement, and not necessarily with the target response.

Scenario 2: simulators 4 and 5 are in mutual disagreement and at the same time in disagreement (in unison) with the target response, while simulators 1-3 are in perfect agreement with the target response as shown in Fig. 5a:

$$\begin{aligned} y_4(x) &= A_4 y_{target}(x) + B_4(x - 0.5) \\ &\quad - C_4;\ P(\mathcal{Y}_4|X) \sim \mathcal{N}(y_4(x), COV = 0.05) \\ y_5(x) &= A_5 y_{target}(x) + B_5(x - 0.5) \\ &\quad - C_5;\ P(\mathcal{Y}_5|X) \sim \mathcal{N}(y_5(x), COV = 0.05) \end{aligned} \tag{12}$$

The constants are set to $\{A_4 = 0.5, B_4 = 10, C_4 = 20\}$, and $\{A_5 = 0.25, B_4 = 0.5, C_5 = 9\}$. We choose the constants such that $y_4(x)$ largely displays the same trend as y_{target} but exhibits a bias, and $y_5(x)$ is non-linear for $x \in [0.1, 0.6]$. The cluster-weighted Bagged estimator is shown in Fig. 5b yielding a much improved predictor of the target (true) response compared to classical Bagging. The performance of the cluster-weighted Bagged estimator is evaluated when 2000 samples are available from each simulator, and $\Delta x = 0.02$. In scenario 1, we idealized a situation when 2 out of 5 simulators are in perfect mutual agreement, and at the same time in disagreement (in unison) with the target response, resulting in a rather marginal reduction in the confidence interval of the order of 5%. Here, the 2 simulators are rather in mutual disagreement, and at the same time in disagreement (in unison)

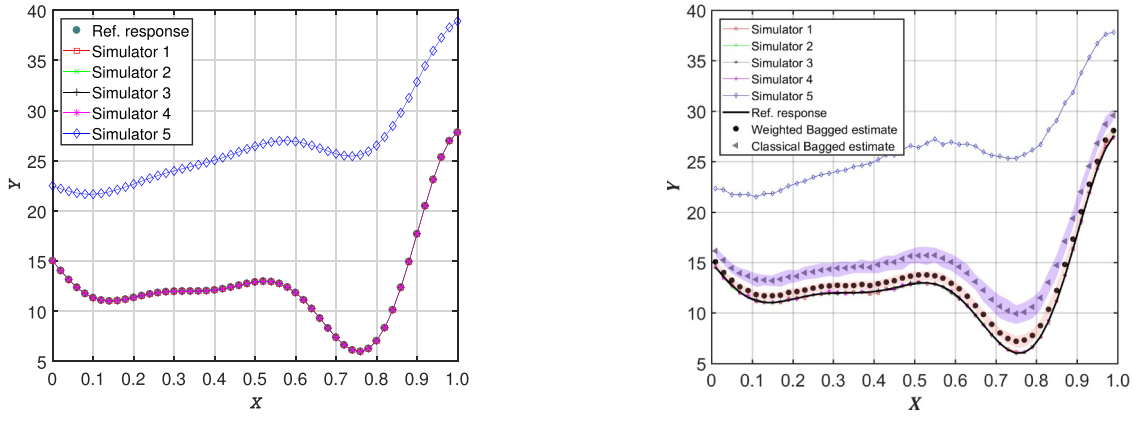
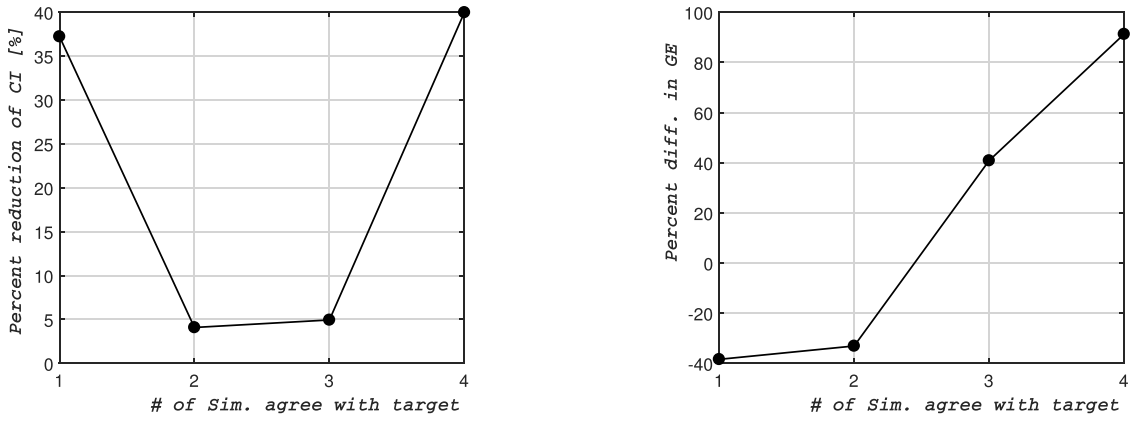
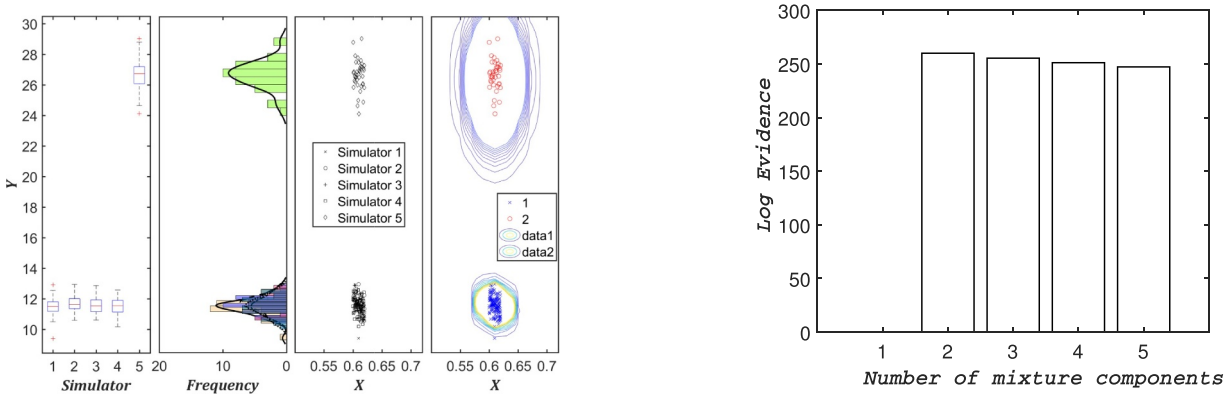


Fig. 2. Scenario 1: (a) The expected values of the target and the five simulators' responses. (b) The cluster-weighted Bagged estimator.



(a) (b)

Fig. 3. Scenario 1: comparing the local cluster-weighted Bagging method to classical Bagging for 2000 samples from each simulator for varying number of simulators in perfect agreement with the target response (50 repeats, $M = 100$). Reduction in the (a) 95% confidence interval, and (b) the generalization error.



(a) (b)

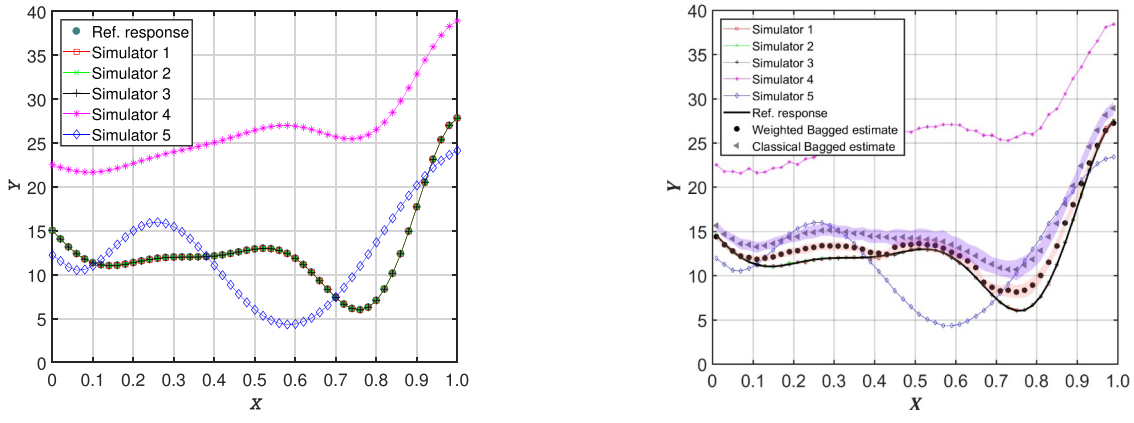
Fig. 4. Scenario 1: (a) Clustering of response for the binned input $x \in [0.6, 0.62]$. (b) Log-evidence of clusters.

with the target response, resulting instead in the reduction of the CI of the cluster-weighted Bagged estimator by 32% averaged across the binned input space x (the GE is reduced by 83%). In a sense, this is a result of diversifying the hypothesis space.

Scenario 3: simulators 1–4 are slightly biased, and simulator 5 is in significant disagreement with respect to the expected value of the target response:

$$
\begin{aligned}
P(\mathcal{Y}_1|X) &\sim \mathcal{N}(y_{target}(x) - 2,\ COV = 0.05)\\
P(\mathcal{Y}_2|X) &\sim \mathcal{N}(y_{target}(x) - 1,\ COV = 0.05)\\
P(\mathcal{Y}_3|X) &\sim \mathcal{N}(y_{target}(x) + 2,\ COV = 0.05)\\
P(\mathcal{Y}_4|X) &\sim \mathcal{N}(y_{target}(x) + 3,\ COV = 0.05)\\
P(\mathcal{Y}_5|X) &\sim \mathcal{N}(0.5y_{target}(x) + 10(x - 0.5) - 20,\ COV = 0.05)
\end{aligned}
\tag{13}
$$

We report that the GE is reduced by 77% while the 95% CI is reduced by 28% averaged across the binned input space. According to



(a) (b)

Fig. 5. Scenario 2: (a) The expected values of the target and the five simulator responses. (b) The cluster-weighted Bagged estimator.

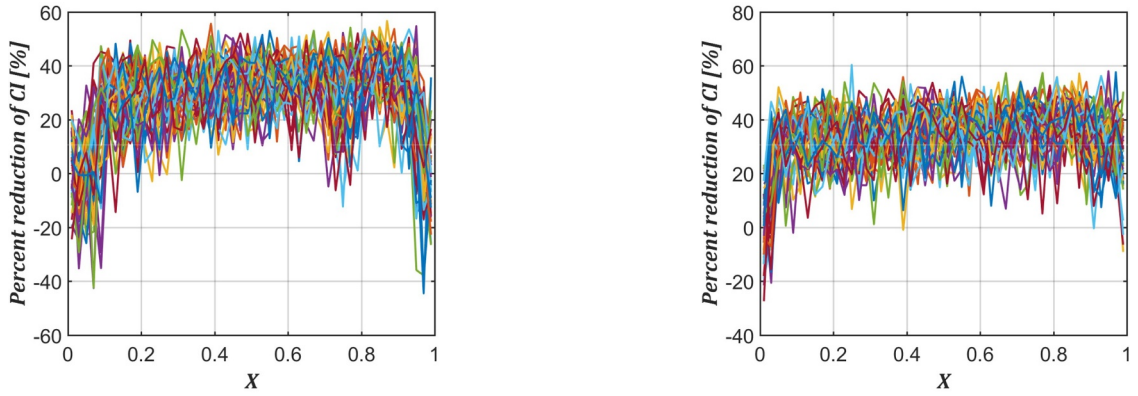


Fig. 6. Scenario 3: percent reduction of the 95% confidence interval as a function of x over 50 repetitions (a) for the original bias of simulators 3 and 4, and (b) when the bias of simulators 3 and 4 is reduced by 0.75 points.

Fig. 6a, the percent reduction of the predictor 95% CI deteriorates for $x \in [0, 0.1]$ and $x \in [0.9, 1]$. It is useful to notice that simulator 5 is nearest to the target response and to the biased simulators 3 and 4 in this region of the input space. As a consequence, clustering via Variational Bayesian Gaussian Mixture is challenged to discriminate clear clusters of majority simulators, and hence the weights default to those of classical Bagging, i.e., equal weights for all simulators conditional on the binned input space. When stochastic outputs are not well separated, the notion of a cluster is not anymore well defined. Reducing the bias of simulators 3 and 4 by 0.75 points closer to the target response, results in the confidence interval being reduced by 34% instead of 28%. This improvement is especially apparent for $x \in [0.9, 1]$ as depicted in Fig. 6b as compared to Fig. 6a. This reflects the importance of discriminating, locally, amongst the simulators in the clustering step. This case study clearly showcases how local weighting stands in contrast to other methods, which assign global weights to each model. Other effects contributing to the performance of the clustering step such as the COV, number of stochastic samples and the bin size are analysed next.

5.2. Case study 2: majority of simulators are imperfect

Here, 4 out 5 simulators (simulators 2-5) are in mutual disagreement and at the same time in disagreement (in unison) with the target response (see Fig. 7a):

$$
\begin{aligned}
&y_1(x) = y_{target}(x);\ P(\mathcal{Y}_1|X) \sim \mathcal{N}(y_1(x),\ COV = 0.05)\\
&y_2(x) = y_{target}(x) + 1;\ P(\mathcal{Y}_2|X) \sim \mathcal{N}(y_2(x),\ COV = 0.05)\\
&y_3(x) = A_3 y_{target}(x^2) + B_3(x^3 - 0.5)\\
&\qquad + C_3;\ P(\mathcal{Y}_3|X) \sim \mathcal{N}(y_3(x),\ COV = 0.05)\\
&y_4(x) = A_4 y_{target}(x) + B_4(x - 0.5)\\
&\qquad + C_4;\ P(\mathcal{Y}_4|X) \sim \mathcal{N}(y_4(x),\ COV = 0.05)\\
&y_5(x) = A_5 y_{target}(\sqrt{x} + B_5) + C_5;\ P(\mathcal{Y}_5|X) \sim \mathcal{N}(y_5(x),\ COV = 0.05)
\end{aligned}
\tag{14}
$$

The constants are set to $\{A_3 = 0.5, B_3 = 8, C_3 = 2\}$, $\{A_4 = 0.5, B_4 = 10, C_4 = 5\}$, and $\{A_5 = 0.25, B_5 = 0.5, C_5 = 12\}$. The choice of the constants is motivated by the introduction of a more diverse set of simulators exhibiting various (mixed) types of disagreements, namely, on the nature of the physical response (linear versus non-linear), on the expected value of the response (systematic versus random bias), and finally on the value and location of specific maximum/minimum peak responses. The local cluster-weighted Bagged estimator is shown in Fig. 7b yielding an improved predictor compared to classical Bagging, based on 2000 samples from each simulator, and bin size $\Delta x = 0.02$. Compared to the classical Bagged estimator, we report that the GE of the local cluster-weighted Bagged estimator is reduced by 39% while the 95% CI is reduced by 7% averaged across the binned input space.

Next, we study the performance of the local cluster-weighted Bagged estimator when varying the simulators output COV for a fixed

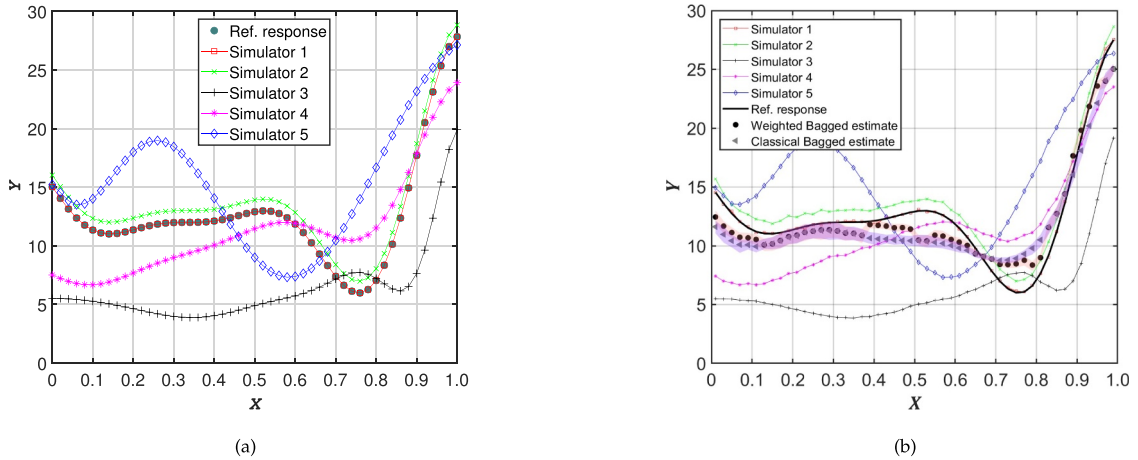
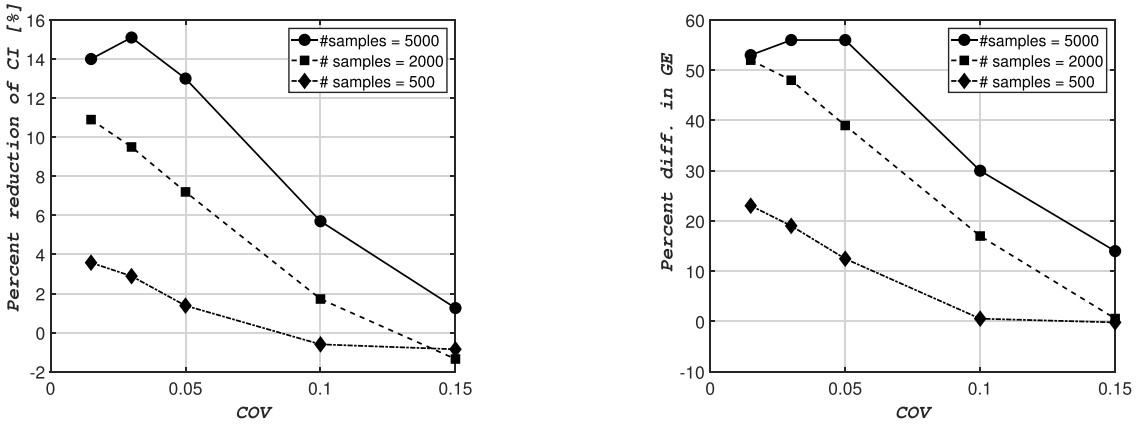


Fig. 7. Case study 2: (a) The expected values of the target and the five simulators' responses. (b) The cluster-weighted Bagged estimator.



(a) (b)

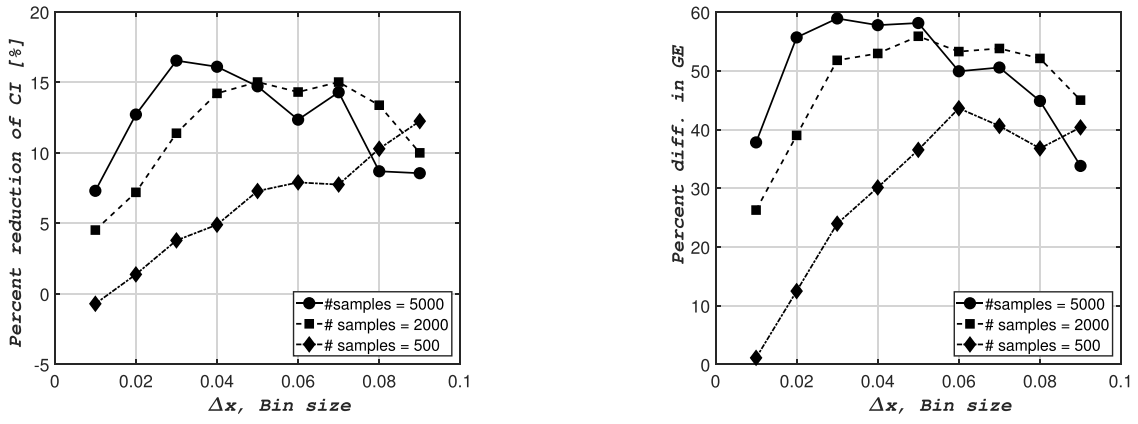
Fig. 8. Case study 2: comparing the local cluster-weighted Bagging method to classical Bagging for 500, 2000 and 5000 samples from each simulator for varying COV (50 repeats, $M = 100$). Reduction in the (a) 95% confidence interval, and (b) the generalization error.

input bin size $\Delta x = 0.02$. The hypothesis here is that large output variability would challenge the VGBM clustering algorithm to discriminate unique clusters. The results shown in Fig. 8 point to a deterioration in the average reduction in the CI and GE of the local cluster-weighted Bagged estimator compared to classical Bagging with increasing COV; the reduction in the CI and GE tends to zero for $COV \geq 10\%$ when 500-2000 observations are sampled for each simulator. However, it turns out that increasing the number of samples to 5000 for each simulator helps improve the performance because when more points are sampled they tend to fall nearer to the Gaussian distribution center, increasing the likelihood to identify unique components of the mixture. It is not clear however if this behaviour of the Gaussian mixture clustering holds true when dealing with skewed and extreme value distributions of the simulators outputs, something that needs to be further investigated in the future.

Finally, we evaluate the effect of the input bin size Δx on the performance of the local cluster-weighted Bagged estimator when 500, 2000 and 5000 observations are sampled for each simulator. When the target response is highly non-linear, evaluating the local cluster-weighted Bagged estimator over a large input bin size will tend to miss important aspects of the response. The results shown in Fig. 9 point to the presence of an input bin size Δx that maximizes the performance of the local cluster-weighted Bagged estimator in terms of reduction of the estimator CI as well as the GE, which in this very case study seems to lie in the range $\Delta x \in [0.03, 0.06]$. Interestingly though, increasing the observational sample size per simulator from 2000 to 5000 deteriorates the performance of the local cluster-weighted Bagged estimator for $\Delta x > 0.05$. This is attributed to the increased likelihood of additional mixture components being elucidated "horizontally" over the width of Δx, and can be seen as the trade-off between the bin size and the number of samples per simulator.

5.3. Case study 4: measurements assimilation

In our method, the local weights are derived based on an unsupervised learning approach, where a-priori information about the performance of any individual simulator in a given region of the input space is unknown or highly uncertain. In other words, the derived local weights may be inadequate, since the method fails to recognize certain simulators are more fitting than others in certain regions of the input space. The conventional approach is to update the weights via assimilation of measurements of the quantity of interest but rarely from multiple measurements replications. Even though measurements assimilation is not the focus of this paper, we advance our method, exploiting measurements to update and condition the local weights. Our insight is that the measurements are just another set of physical (real, non-numerical) "simulator" output, which are added to the bucket of available output from the numerical simulators. The unsupervised ensemble local cluster-weighted Bagged framework assimilates any number of measurements replications without explicitly labelling the



(a) (b)

Fig. 9. Case study 2: comparing the local cluster-weighted Bagging method to classical Bagging for 500, 2000 and 5000 samples from each simulator for varying bin size Δx (50 repeats, $M = 100$). Reduction in the (a) 95% confidence interval, and (b) generalization error.

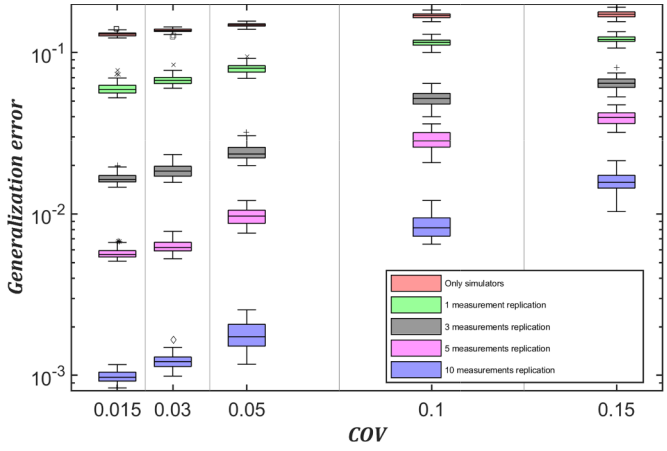


Fig. 10. Case study 3: generalization error for the measurements assimilations as a function of the output coefficient of variation.

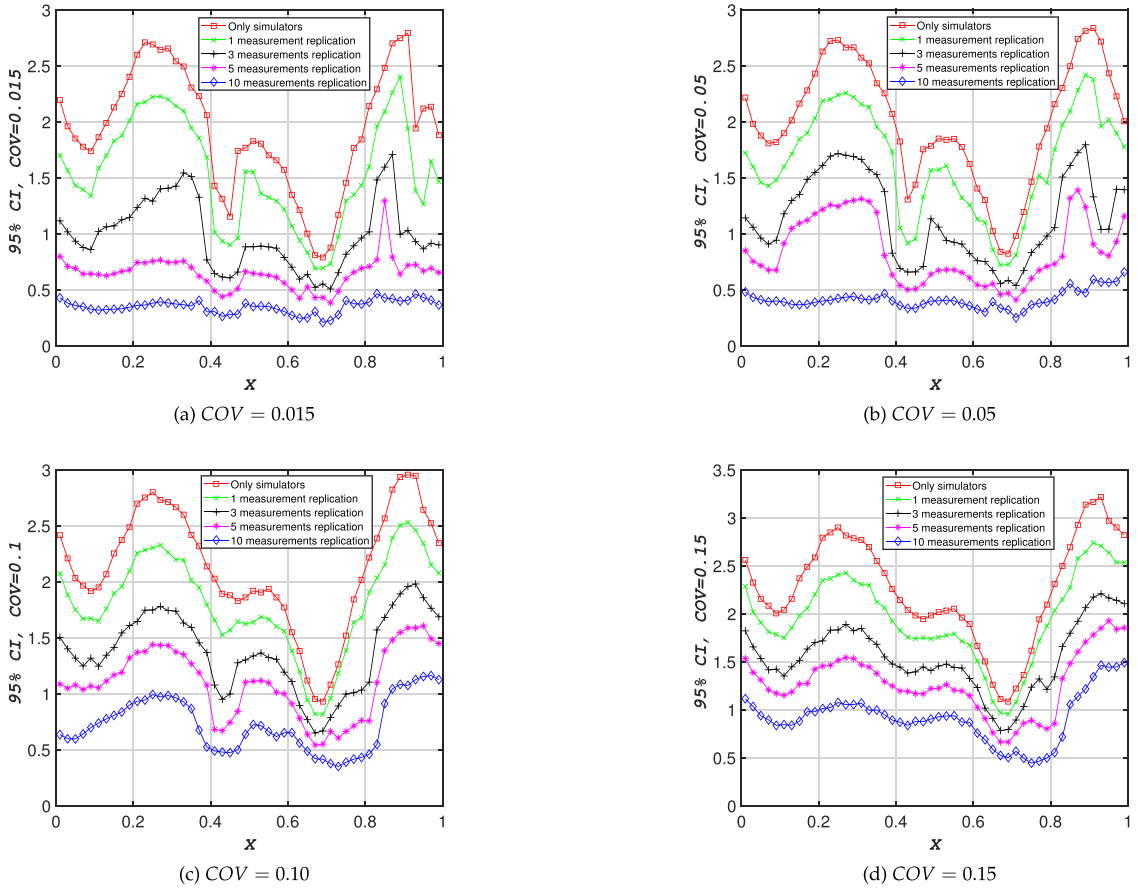
measurements as ground truth, and as such are not assigned individual weights for their likelihood being the true system process predictors, but are assigned weights for being part of local clusters. Therefore, the measurements are added to the mix with other numerical simulators, and Algorithm 1 is simply re-run without any additional mathematical manipulations or assumptions. In this case study, we re-use the same analytical simulators as in (5.2) (Eq. (15)), and we demonstrate how we assimilate 1, 3, 5 and 10 measurements replications. Each measurements replication are samples from the target (true) response, but we also include an error term ϵ: $y_{meas}(x) = y_{target}(x) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 0.25)$. 2000 samples are available from each of the 5 numerical simulators and measurements replications, and $\Delta x = 0.02$. We calculate the GE and the 95% CI of the estimator as a function of the COV. According to Fig. 10, lower COV (i.e., $COV \leq 0.05$) achieve a 2-order of magnitude reduction in the GE, when upto 10 measurement replications are assimilated via the ensemble local cluster-weighted Bagged framework. For $COV \geq 0.10$ the reduction in the GE is only 1-order of magnitude. Similarly, Fig. 11 shows that the 95% CI drops as more and more measurement replications are included to update the local weights and subsequently the cluster-weighted Bagged estimator, which translates into a better precision of the estimator for a given confidence level. Interestingly, when 10 measurements replications are assimilated in the algorithm for lower $COV \leq 0.05$), the 95% CI exhibits little fluctuations over the input range x.

5.4. Comparison to other methods

We consider an $\mathcal{M}$-open setting in which the true data-generating process is not one of the candidate models in the model list. In this comparison the stochastic simulators list is as follows:

$$
\begin{aligned}
y_1(x) &= A_1 y_{target}(x) + B_1(x - 0.5) \\
&\quad + C_1;\ P(\mathcal{Y}_1|X) \sim \mathcal{N}(y_1(x),\ COV = 0.05) \\
y_2(x) &= A_2 y_{target}(x^2) + B_2(x^3 - 0.5) \\
&\quad + C_2;\ P(\mathcal{Y}_2|X) \sim \mathcal{N}(y_2(x),\ COV = 0.05) \\
y_3(x) &= A_3 y_{target}(x) + B_3(x - 0.5) \\
&\quad + C_3;\ P(\mathcal{Y}_3|X) \sim \mathcal{N}(y_3(x),\ COV = 0.05) \\
y_4(x) &= A_4 y_{target}(\sqrt{x} + B_4) + C_4;\ P(\mathcal{Y}_4|X) \sim \mathcal{N}(y_4(x),\ COV = 0.05)
\end{aligned}
\tag{15}
$$

The constants are set to $\{A_1 = 1, B_1 = 10, C_1 = 2\}$, $\{A_2 = 0.5, B_2 = 8, C_2 = 2\}$, $\{A_3 = 0.5, B_3 = 10, C_3 = 5\}$, and



(a) $COV = 0.015$ (b) $COV = 0.05$

(c) $COV = 0.10$ (d) $COV = 0.15$

Fig. 11. Case study 3: 95% Confidence Interval for the measurements replications assimilations as a function of input x.

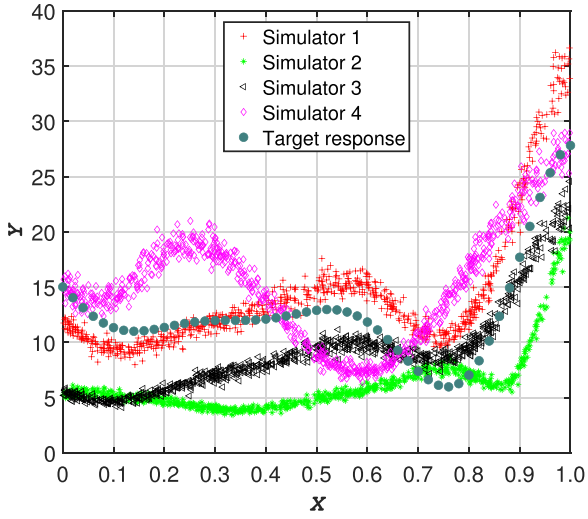


Fig. 12. 4 Stochastic responses used to compare the unsupervised local cluster-weighted Bagging to other methods. None of the individual simulators are the data generating process for the target response

Table 3
Generalization error of the various methods (50 repeats, $M = 100$).

Method	Generalization error
UnLoCWeB	0.511
Stacking	0.602
Pseudo-BMA	0.723
BB-Pseudo-BMA	0.723

$\{A_4 = 0.25, B_4 = 0.5, C_4 = 12\}$. 500 samples are assumed available from each of the 4 simulators, and $\Delta x = 0.02$. We aim at predicting the expected value of the response given all the available output from the simulators, see Fig. 12. We compare the predictions of our method to the following methods: ***(i)*** Pseudo Bayesian model averaging (Pseudo-BMA) in which Akaike's information criterion (AIC) weights are used, ***(ii)*** Pseudo Bayesian model averaging with Bayesian Bootstrapping (BB-Pseudo-BMA), which is a variant of the previous method, whereby the uncertainty in the future data distribution is taken into account when computing the AIC weights, and ***(iii)*** Stacking of predictive distributions (Stacking). Here, we implement the BMA and Stacking methods following the detailed descriptions in [43] and implemented in PyMC3 [44]. Using the simulators output in Fig. 12 we compute the stacking weights to be 0.88 and 0.12 corresponding to simulators 1 and 3, respectively, and both the Pseudo-BMA and BB-Pseudo-BMA yield a weight of 1 towards simulator 1. The generalization error of the predictions of the various methods are shown in Table 3. Even though simulator 1 is seemingly not adequate (but arguably better than the remaining simulators) at representing the target response, in the BMA setting it is still assigned a disproportionately higher marginal likelihood compared to the remaining candidate simulators, across the whole input space. With the Stacking of predictive distributions method, further proportion of the weight is assigned to simulator 3, which results in an improvement of the generalization error from 0.723 for BMA to 0.602. Because our method is adaptive and assigns weights locally to the clustered simulators output, it succeeds in reducing the generalization error further to 0.511, a 15% reduction compared to Stacking.

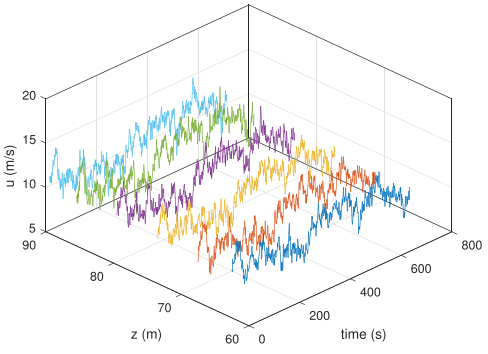
6. Engineering application

We demonstrate the method by evaluating the Fatigue Damage Equivalent Load (DEL) on a 30 m long wind turbine blade, using 10 Finite Element based (FEM) simulators. ***Wind loads:*** The turbulent wind field (Fig. 13a) is generated using spatially correlated wind inflow time series along the span of the blade, based on an exponential coherence model and the Kaimal turbulence auto-spectrum to account for the spatial correlation structure of the longitudinal velocity component (Annex B in [45]). Thereafter, the time varying normal F_N and tangential F_T aerodynamic forces along the span of the blade (Fig. 13b) are computed using a quasi-static Blade Element Momentum model. ***FEM models:*** For simplicity, the complex structure of the blade is converted into an equivalent tapered clamped-free beam. Ten finite element simulators of the beam were developed to compute the time varying blade root in-plane bending moment M_X from the applied nodal normal F_N and tangential F_T aerodynamic forces along the span of the blade. The constitutive parameters of the FEM simulators are shown in Table 4. The input are 100 times series of F_N and F_T, each 600 seconds long with a time step of 0.01 sec, applied at the cross-sectional aerodynamic center. ***Fatigue:*** The short-term fatigue damage equivalent loads (DEL) at the blade clamped root end are calculated on the basis of the M_X output times series, which, for a given mean wind speed, are determined by:

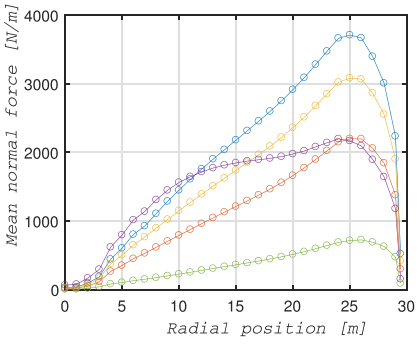
$$DEL = \left(\frac{1}{N_{eq}} \sum_i n_i (\Delta M_{X,i})^m \right)^{1/m} \tag{16}$$

where n_i is the number of load cycles with range $\Delta M_{X,i}$, i is the fatigue cycle index, and N_{eq} is the equivalent number of load cycles, typically 10^7 cycles. The Whöler exponent $m = 10$ for a composite structure.

The computed DEL for each of the 10 FEM simulators and the local cluster-weighted Bagged expected DEL are plotted and compared as a



(a)



(b)

Fig. 13. (a) An example of the free stream turbulent wind speed time series at various radial positions along the span of the blade. (b) An example of the mean value of the normal force F_N along the span of the blade for different mean wind speeds.

Table 4
The constitutive parameters of the 10 FEM simulators of the clamper-free tapered beam.

Simulator	1	2	3	4	5	6	7	8	9	10
Dimensions	1D	1D	3D	3D	2D	2D	3D	3D	3D	3D
Element type	Euler–Bernoulli Beam	Euler–Bernoulli Beam	Euler–Bernoulli Beam	Timoshenko Beam	Plane-Stress (2D Elas.)	Plane-Stress (2D Elas.)	Solid (3D Elas.)	Solid (3D Elas.)	Solid (3D Elas.)	Solid (3D Elas.)
Shape function	Linear	Linear	Linear	Quadratic	Linear	Quadratic	Linear	Quadratic	Quadratic	Quadratic
Modes	2	4	4	8	4	8	4	8	10	12
Number of elements	16	48	48	48	16	48	2x2x96	4x4x192	8x8x384	16x16x768
Load nodes	6	14	10	14	6	14	6	10	14	28
**Nodal force orientation	In-plane projection	In-plane projection	According to γ	According to γ	In-plane projection	In-plane projection	In-plane projection	According to γ	According to γ	According to γ
Torsion DOF	off	off	on	on	off	off	off	off	on	on
Stiffness Matrix	Full	Full	Full	Full	Full	Full	Full	Full	Full	Full

"In-plane projection" means that the resultant of the forces at a given cross-section is projected onto the $YZ-$ *plane*. *"According to γ" means that the forces at a given cross-section are oriented according to the angle γ, which is dependent on the relative wind speed, inflow angle, the blade pitch and twist angles.

function of wind speed in Fig. 14(a). We observe that the dispersion of the simulators predictions is not significant because we are examining damage equivalent fatigue loads, which by itself is an aggregate quantity and under certain input wind loads the simulators will exhibit similar predictions due to the linear nature of the blade response, except for the predictions from simulators 5 and 6 which start to drift away at wind speeds above 15 m/s. This reflects one of the assumptions made in our method, namely that the simulators can be hard to distinguish because they are equally fitting in certain regions of the input covariate space, irrespective of their a-priori fidelity level. The local cluster-weighted Bagged predictor displays a small variance compared to the original data set, which is a very important target because it reflects a lower model uncertainty in the prediction of fatigue loads. The outcome of clustering is a map of probabilities, which effectively is the joint probability $P(\mathcal{S}_i, C_j|\Delta\boldsymbol{x})$ of a cluster C_j and a simulator S_i. Introducing these probabilities when bootstrapping the data in the weighted-Bagging step of the algorithm has the effect of pulling the ensemble aggregated predictor towards the clusters with higher densities, which is indeed what happens at wind speeds above 15 *m/s, where the predictions from simulators 5 and 6 start to drift away form the main predictions cluster*. We further experiment with removing the highest FEM fidelity simulator 10 from the mix and re-run the algorithm to compute the expected DEL. According to Fig. 14(b), the local cluster-weighted Bagged predictor remains little changed, which is computationally attractive since the output of the highest fidelity FEM simulator 10 are indinstinguishable from the cluster of output of simulators 1-4 and 7-9, but takes an order of magnitude more time to run with no perceptible change in the ensemble aggregate predictive ability..

7. Conclusion and future perspective

In this contribution we proposed an ensemble learning framework based on unsupervised variational Bayesian Gaussian mixture clustering and local weighted bootstrap aggregating the stochastic output from multiple distinct simulators, in the absence of any measurements. We call our method: unsupervised local cluster-weighted Boostrap aggregation. Clustering served the purpose of deriving the probability map (weights). Clustering is carried out on the stochastic output corresponding to the binned input space. Our method does not assign weights to individual simulators per se, rather it assigns weights to clusters of output, i.e., to the collection of "similar" output data originating from the various simulators in each local region of the binned input space. The clustered outputs are then combined via the cluster-weighting bootstrap aggregation step for each local region of the binned input space. Furthermore, even though assimilation of measurements is not the focus of this paper, we advance the method, demonstrating how measurements replications may be exploited in order to update the weights. Our insight is that the measurements may be considered as yet another physical simulator(s) in the mix with other numerical simulators, with the proposed algorithm simply re-run without any additional mathematical manipulations. We argued that such a way of performing data assimilation is critical when the correct simulator output is clustered separately from the larger density clusters for certain regions of the input space. In other words, the derived weights may be inadequate because the method fails to recognize that certain simulators are more fitting than others in certain regions of the input space. We illustrated the framework with analytical and engineering applications and compared its effectiveness to that of classical Bagging. We demonstrated tangible performance gains in terms of reduction in the generalization error and the 95% confidence interval of the estimator. This is crucial because a reduction in the generalization error translates into a better accuracy of the estimator, while a reduction in the confidence interval translates into a better precision of the estimator for a given confidence level. Finally, we showed how the performance of our method varies depending on the input bin size, sample size per simulator, simulations

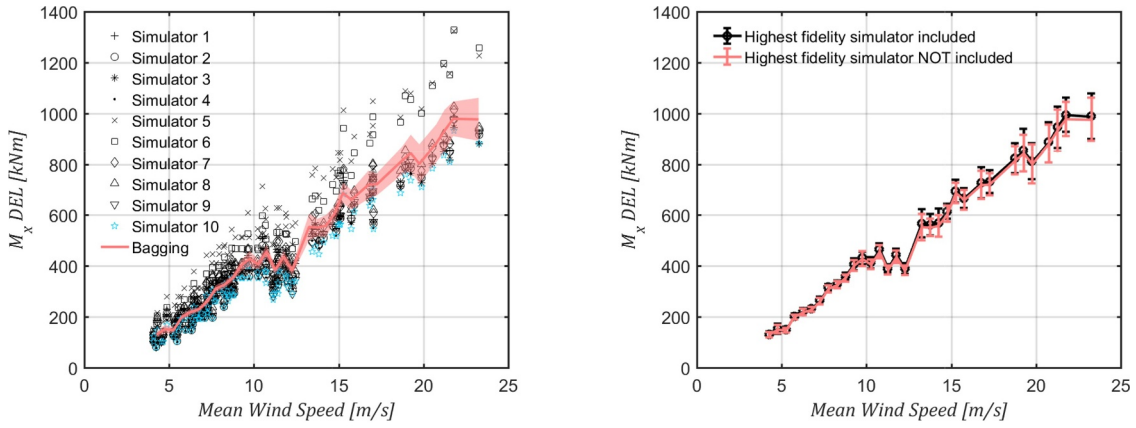


Fig. 14. (a) Comparing the DEL of the blade root bending from 10 FEM simulators, and the local cluster-weighted Bagged predictor with the 95% CI. (b) Comparing the local cluster-weighted Bagged predictor when the highest fidelity simulator is removed from the ensemble.

output dispersion and level of agreement amongst the simulators (diversity).

We further compared our method to Bayesian Model Averaging and Stacking of predictive distributions on an analytical computational example. BMA disproportionately assigned higher marginal likelihood to one out of four candidate simulators, resulting in a poor performance. With Stacking of predictive distributions method, further proportion of the weight is assigned to an additional simulator, resulting in a further drop of the generalization error. Because our method is adaptive and assigns weights locally to the clustered simulators output, we end with reducing the generalization error 15% compared to Stacking, which is very promising.

Several extensions to this research are identified. First, there is a need to investigate if the framework may efficiently operate in high dimensional problems, especially in view of the challenges of binning data in a high dimensional input space. Second, in case the stochastic outputs follow skewed or extreme value distributions then the Variational Gaussian Mixture clustering may not be adequate. It is thus of interest to deploy non-Gaussian based clustering, such as model-based clustering with non-normal mixture distributions or clustering with Normalizing Flows to specify flexible, arbitrarily complex and scalable posterior mixtures. Finally, further research is needed in order to formulate a theoretical basis as to how diversifying the hypothesis space affects the optimal number of simulators subject to a target accuracy and precision when doing inference.

Regardless, the initial results of our method point to the need for practitioners to consider this as a useful framework, when output from multiple stochastic simulators are available, particularly when it is hard to distinguish a best model out of an ensemble.

CRediT authorship contribution statement

Imad Abdallah: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing - original draft, Writing - review & editing. **Konstantinos Tatsis:** Validation, Data curation, Investigation. **Eleni Chatzi:** Supervision, Project administration, Funding acquisition, Resources.

Declaration of Competing Interest

No conflict of interest exists.

Acknowledgements

The authors acknowledge the support of the European Research Council via the ERC Starting Grant WINDMIL (ERC-2015-StG #679843) on the topic of Smart Monitoring, Inspection and Life-Cycle Assessment of Wind Turbines. The authors would further like to acknowledge the support the ERC Proof-of-Concept Grant ERC-2018-PoC WINDMIL RT-DT (funded under H2020-EU.1.1., grant agreement ID 825833). Finally, this project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 769373 (Project FORESEE).

References

[1] Popko W, Vorpahl F, Zuga A, Kohlmeier M, Jonkman J, et al. Offshore code comparison collaboration continuation (OC4), phase i - results of coupled siumulations of an offshore wind turbine with jacket support structure. J Ocean Wind Energy 2014;1:1–11.
[2] Ranjan R, Gneiting T. Combining probability forecasts. J Royal Stat Soc, Ser B 2010;72:71–91.
[3] Merrick J. Aggregation of forecasts from mutiple simulation models. Proceedings of the 2013 winter simulation conference. 2013.
[4] Abdallah I, Tatsis K, Chatzi E. Fatigue assessment of a wind turbine blade when output from multiple aero-elastic simulators are available. Proceedia engineering, X International conference on structural dynamics, EURODYN 2017, Rome, Italy. 199. 2017. p. 3170–5.
[5] Abdallah I, Chatzi E. An ensemble learning approach to aggregate the output from multiple simulators and measurements. The 19th working conference of the IFIP Working Group 7.5 on reliability and optimization of structural systems (IFIP 2018). 2018.
[6] Clemen R, Winkler R. Combining probability distributions from experts in risk analysis. Risk Anal 1999;19(2):187–203.
[7] Allaire D, Willcox K. A mathematical and computational framework for multifidelity design and analysis with computer models. Int J UncertainQuantif 2014;4(1):1–20.
[8] Jouini M, Clemen R. Copula models for aggregating expert opinions. Oper Res 1995;44(3):444–58.
[9] Riley M, Grandhi R. Quantification of model-form and predictive uncertainty for multi-physics simulation. Comput Struct 2011;89(11–12):1206–13. https://doi.org/10.1016/j.compstruc.2010.10.004.
[10] Gibbons J, Cox G, Wood A, Craigon J, Ramsden S, Tarsitano D, et al. Applying Bayesian model averaging to mechanistic models: an example and comparison of methods. Environ Modell Softw 2008;23(8):973–85.
[11] Fragoso T, Bertoli T, Louzada F. Bayesian model averaging: a systematic review and conceptual classification. Int Stat Rev 2017.
[12] Hoeting JA, Madigan DM, Raftery AE, Volinsky CT. Bayesian model averaging: a tutorial. Stat Sci 1999;14:382–401.
[13] Körner S, Holzäpfel F, Sölch I. Probabilistic multimodel ensemble wake-vortex prediction employing Bayesian model averaging. J Aircraft 2019;56(2):695–706.
[14] Rings J, Vrugt J, Schoups G, Huisman J, Vereecken H. Bayesian model averaging using particle filtering and Gaussian mixture modeling : theory, concepts, and simulation experiments. Water Resour Res 2012;48(5).
[15] Liu D, Wang S, Zhang C, Tomovic M. Bayesian model averaging based reliability analysis method for monotonic degradation dataset based on inverse gaussian process and gamma process. Reliab Eng Syst Safety 2018;180(C):25–38.
[16] Yu Q, MacEachern SN, Peruggia M. Clustered Bayesian model averaging. Bayesian Anal 2013;8(4):883–908.
[17] Monteith K, Carroll JL, Seppi K, Martinez T. Turning Bayesian model averaging into Bayesian model combination. 2011 international joint conference on neural networks, IEEE, San Jose, CA, USA. 2011. p. 2657–63.

[18] Peherstorfer B, Kramer B, Willcox K. Combining multiple surrogate models to accelerate failure probability estimation with expensive high-fidelity models. J Comput Phys 2017;341:61–75.
[19] Cheng K, Lu Z. Structural reliability analysis based on ensemble learning of surrogate models. Struct Safety 2020;83.
[20] Strömberg N. Reliability-based design optimization by using ensembles of metamodels. UNCECOMP 2019, 3rd ECCOMAS thematic conference on uncertainty quantification in computational sciences and engineering, Crete, Greece. 2019.
[21] McFarland J, Bichon B. Bayesian model averaging for reliability analysis with probability distribution model form uncertainty. In Proc. 50th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference, California, USA. 2009.
[22] Giovanis DG, Shields MD. Structural reliability analysis from sparse data. 13th international conference on applications of statistics and probability in civil engineering, ICASP13, Seoul, South Korea. 2019.
[23] Geneva N, Zabaras N. Quantifying model form uncertainty in Reynolds-averaged turbulence models with Bayesian deep neural networks. J Comput Phys 2019;383:125–47.
[24] Radaideh MI, Borowiec K, Kozlowski T. Integrated framework for model assessment and advanced uncertainty quantification of nuclear computer codes under Bayesian statistics. Reliab Eng Syst Safety 2019;189:357–77.
[25] Wolpert D. Stacked generlization. Neural Netw 1992;5(2):241–59.
[26] Breiman L. Bagging predictors. Mach Learn 1996;24:123–40.
[27] Yu Q. Weighted bagging: a modification of adaboost from the perspective of importance sampling. J Appl Stat 2011;38:451–63.
[28] Chen T, Guestrin C. XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, San Francisco, California, USA. 2016. p. 785–94.
[29] Zhang D, Qian L, Mao B, Huang C, Huang B, Si Y. A datadriven design for fault detection of wind turbines using random forests and XGBoost. IEEE Access 2018;6:21020–31.
[30] Bühlmann P. Bagging, boosting and ensemble methods. In: Gentle J, Mori Y, editors. Handbook of comp. stat. Springer; 2012. p. 985–1022.
[31] Li Z, Wub D, Hua C, Terpenny J. An ensemble learning-based prognostic approach with degradation-dependent weights for remaining useful life prediction. Reliab Eng Syst Safety 2019;184:110–22.
[32] Fu W, Perry PO. Estimating the number of clusters using cross-validation. J Comput Graph Stat 2019:1–12. https://doi.org/10.1080/10618600.2019.1647846.
[33] Bishop C. Pattern recognition and machine learning. Springer, ISBN 978-0-387-31073-2; 2006.
[34] Dempster A, Laird N, Rubin D. Maximum likelihood from incomplete data via the EM algorithm. J Royal Stat Soc 1977;39(1):1–38.
[35] McLachlan G, Krishnan T. The EM algorithm and extensions, 2nd Edition. Wiley series in prob. and stat., ISBN 978-0-471-20170-0; 2008.
[36] Brodersen K, Daunizeau J, Mathys C, Chumbley J, Buhmann J, Stephan K. Variational bayesian mixed-effects inference for classification studies. NeuroImage 2013;76:345–61.
[37] Ormerod J, Wand MP. Explaining variational approximations. Handbook of computational statistics. 64. Taylor & Francis, Ltd. on behalf of the American Statistical Association; 2010. p. 140–53.
[38] Sun S. A review of deterministic approximate inference techniques for Bayesian machine learning. Neural Comput Appl 2013;23:2039–50.
[39] Nagel J, Sudret B. Spectral likelihood expansions for Bayesian inference. J Comput Phys 2016;309:267–94.
[40] Penny W. Variational Bayes for d-dimensional gaussian mixture models. Tech. Rep.. University College London; 2001.
[41] Jimenez Rezende D, Mohamed S. Variational inference with normalizing flows. Proceedings of the 32nd international conference on machine learning, Lille, France. 37. 2015.
[42] Kingma D, Salimans T, Jozefowicz R, Chen X, Sutskever I, Welling M. Improved variational inference with inverse autoregressive flow. 30th conference on neural information processing systems, Barcelona, Spain. 2016.
[43] Yao Y, Vehtari A, Simpson D, Gelman A. Using stacking to average bayesian predictive distributions (with discussion). Bayesian Anal 2018;13:917–1007.
[44] Salvatier J, Wiecki TV, Fonnesbeck C. Probabilistic programming in python using PyMC3. PeerJ Comput Sci 2016;2.
[45] Wind Turbines, Part 1 Design Requirements. Tech. Rep.. International Electrotechnical Commission; 2005.