Interval Field for Spatially and Temporally Dependent Uncertainty—Machine Learning Approach

David Betancourt,[1*,2] Rafi L. Muhanna.,[2] and Robert L. Mullen[3]

[1] *School of Computational Science and Engineering, Georgia Institute of Technology, U.S.A.*
[2] *School of Civil and Environmental Engineering, Georgia Institute of Technology, U.S.A.*
[3] *School of Civil and Environmental Engineering, University of South Carolina, U.S.A.*
[*] Corresponding author: david.betancourt[at]gatech.edu

Abstract

A major shortcoming to interval uncertainty approaches in computational mechanics is the lack of an interval field to represent uncertainty. This paper introduces the supervised interval field (SIF), a model to quantify spatially and temporally dependent uncertainty using machine learning. We introduce deep learning model architectures that are used to develop the SIF for domains of any dimension using deep recurrent neural networks. We demonstrate how to unify the SIF with the Interval Finite Element Method (IFEM) and show an experiment using soil layer data.

Keywords: Interval field, Spatially dependent uncertainty, Machine learning, Finite elements, Computational mechanics, Materials

1 Introduction

The analysis and design of complex engineering systems is exposed to a great number of uncertain parameters in the system inputs. Insufficient modeling of these parameters can lead to inaccurate and/or disastrous expectations of the system's performance [1–3]. In the context of computational mechanics, random fields [4, 5] have been widely used within a probabilistic framework in order to model the spatially and temporally dependent uncertainty in the system. Despite its widespread adoption, random field theory is inadequate for cases where there is significant epistemic uncertainty (i.e., missing, imprecise data) [6–8]. Under such cases, non-probabilistic approaches such as interval models have been employed to model both aleatory and epistemic uncertainty.

One of the main drawbacks within the non-probabilistic interval framework is, unlike random fields, not being able to model spatially- or temporally-varying uncertainty in the system. Given this limitation, the spatially varying uncertainty is modeled by assigning the same interval variable over the entire domain. Therefore, the Interval Finite Element Method (IFEM) solution can be overestimated due to the lack of mutual interval dependency definition between interval parameters [9]. As a result, interval field models have been recently proposed [10–12] in order to model spatially-varying dependency within the interval model framework. Nonetheless, the interval field methods proposed are limited in their applicability and scalability.

Consequently, this paper introduces the *supervised interval field*, a data-centric model to quantify spatially and temporally dependent uncertainty using machine learning integrated with the IFEM. In computational mechanics, the prime applications are for spatially-varying material uncertain properties and uncertain time-varying loading. In order to account for the uncertain properties, machine learning model models are presented that are able to represent the spatial and temporal dependencies in the data. Once the predictions of the machine learning model are obtained, the supervised interval field is formed, discretized, and integrated in one framework with the IFEM. The proposed method is capable of extending to domains of any Eucledian dimension.

Numerical experiments are conducted using soil layer data in order to model spatially-varying material uncertainty with the proposed interval field. We demonstrate that our method achieves high accuracy when compared to ground truth values of the soil material properties. In addition, the presented method can be used in multiple engineering applications, scales, and data types.

Our contribution is as follows: (1) our method reduces the interval dependency conservatism by using the proposed interval field (SIF); (2) extension of our method to any domain dimensionality; (3) introduction of machine learning methods that work well with computational mechanics applications; (4) provide an interval enclosure for machine learning model predictions; and (5) bridge the gap between engineering and machine learning.

2 Related Work

The first to incorporate the concept of spatially varying uncertainty within IFEM was Moens et al [13–15] through different methods. In [14] an interval field concept was developed which allowed to use dependency between elements, which is otherwise not possible using interval parameters. To that end, two methods were introduced: The Local Interval Field Decomposition (LIFD), and the Inverse Distance Weighting Interpolation (IDW). For LIFD, the dependency parameters are determined from a gradient method developed by Imholz et al, which defines the dependency between parameters as the maximum gradient that can occur between them. In this definition, perfect dependency corresponds to a zero gradient and perfect independency corresponds to the maximum gradient between parameters. The dependency parameters are discretely defined at each element of the mesh—thus the dimensionality of the interval field equals the numbers of elements in the mesh. The discrete values for the dependency parameters are determined by a smooth second-order polynomial function presented in [14]. For the IDW method, the base functions needed for the interval field are determined and spatially selected based on prior engineering knowledge.

The pitfalls for the Moens et methods are as follows: The LIFD is computationally expensive as the dimensionality of the uncertainty equals the number of elements in the mesh. Additionally, the theoretical arguments for the determination of the dependence are not fully justified. The IDP is computationally cheaper, and could produce good results where enough a-priori domain engineering knowledge is available. As a result, expert hand-engineering would be required to produce good results thus rendering it non-scalable. Also, both methods were only presented for 1D cases.

Wu et al [12] developed a hybrid method for static FE analysis (X-UISS method) that combines random fields and intervals fields. It has an algorithm that easily extends to 2D and 3D domains. For the interval field implementation, the X-UISS method defines the upper-bound UB and lower-bound LB functions from the extrema of a set of measurements at a spatial coordinate $\mathbf{x}$. Then, the UB and LB of measured points are linearly interpolated between the $\mathbf{x}$ points where data is unavailable. Then, the conventional IFEM [16] procedure is implemented using the linearly interpolated values for the domain. The pitfall of this method is that it is computationally expensive by having to solve both SFEM and IFEM methods but more importantly because it uses linear interpolation between data points, which is an oversimplification of the spatial uncertainty variability in a domain.

More recently, Sofi et al [10, 11] developed a 1D interval field approach for IFEM which uses the extra unitary interval (EUI) in order to reduce the overestimation of the IFEM due to mutual dependency. In contrast to the Hybrid method by Wu et al. and the LIFD method by Moens et al., the dimension of uncertainty does not depend of the FE mesh size of the model. However, their underlying interval field method is based on a closed-form solution of a determinate beam or an approximate solution for statically indeterminate beams. A general EUI method for 2D and 3D domains under general boundary conditions has not been developed to the authors' knowledge.

In the area of machine learning, work that seeks to include data dependencies in the prediction includes speech and image recognition, among others that make predictions on sequential data. We borrow some of these ideas for the present work and present our own contributions in order to quantify the uncertainty. To our knowledge, our work presents the first application of machine learning to model spatial and temporal dependency in computational mechanics.

3 Background

A brief introduction to the different theories and techniques used in this paper follows.

Supervised Learning (SL) Supervised learning [17, 18] seeks to learn a predictive model $f : X \to Y$ in some n-dimensional Eucledian space $\mathbb{R}^n$, given a set of training examples. In order to achieve this goal, the SL model is trained using the X inputs (*features*) with known Y outputs (*target values*). This composes the training set $\mathcal{T} = (X, Y)$ where $X \in \mathbb{R}^n$ is the feature space and $Y \in \mathbb{R}^d$ is the output space. More specifically, the set of feature vectors X is used as input to the SL algorithm, which produces at each instance an output $\hat{f}(x_i)$. During training the algorithm reduces the difference between the true target y_i and its prediction $\hat{f}(x_i)$ by minimizing a loss function $\mathcal{L}(\Theta)$, such as squared loss, for parameters Θ. Following the training phase, the testing phase uses feature instances not present during training and it predicts $\hat{f}(x_i)$ using the learned model without access to its targets y_i. After successfully testing the algorithm, the ultimate goal of SL is to generalize to predictions of unseen data where target values are not available.

Deep Recurrent Neural Networks Recurrent neural networks (RNN) are extensions to neural networks used for sequential data [19]. For a sequence of inputs $(x_1, \cdots, x_N)$, a recurrent neural network (RNN) computes a sequence of outputs $(y_1, \cdots, y_N)$ using a recurrence equation at every time step

$$h_t = f_W(h_{t-1}, x_t) \tag{1}$$

In Eq. 1, $h_t \in \mathbb{R}^n$ is the new hidden state of the RNN, f_W is a function with weight parameter matrix W, h_{t-1} is the previous hidden state, and x_t is the input vector at time t. RNNs can be represented as a deep multilayer model and are generally optimized using gradient-based methods. Vanilla RNNs have the pitfall of exploiting gradients due to long-term dependencies in the data when processing long sequences [19]. Long short-term memory [20] (LSTM) units are used to build the deep RNN network (or deep LSTM network) to bypass this problem.

Long Short-Term Memory Units Long short-term memory (LSTM) [20] units are designed to process and predict sequential data while addressing the numerical shortcomings of the exploiting gradients in vanilla RNNs [21]. In order to do so, LSTMs mantain a memory cell state c_t along with the hidden state h_t at every time t. At each time t, the LSTM can choose to *remember* or *forget* information by using the following gating mechanisms, and shown on Fig 1. When combined in a deep architecture, deep LSTM networks are a formidable machine learning predictive model [19, 21].

Random Fields Random fields [4] $H(\mathbf{x}, \theta)$ are generalizations of random processes where the random variable $\mathbf{x} \in \mathbb{R}$ and θ is the outcome of the random phenomenon. Random fields are used to represent the spatial variation of a system property by a random variable defined over the region on which the variation occurs.

Stochastic Finite Elements In the context of computational mechanics, random field theory has been adopted with the Finite Element Method (FEM) to form the Stochastic FEM (SFEM) [5] in order to find the response of a stochastic system. In particular for most SFEM problems, the Random Field is Gaussian and discretized through a Kauhunen-Loève expansion, while the response is found with a polynomial chaos expansion. Together this procedure is known as the spectral approach to SFEM [5].

Interval Finite Elements The Interval Finite Element Method (IFEM) is a numerical analysis method that describes the uncertainty by using interval valued parameters. IFEM is used in situations where a probabilistic characterization of the system is not possible. The IFEM finds guaranteed upper and lower bound enclosures of the response [16].

4 Methods

4.1 Supervised Interval Field

We now present the first data-centric approach to generate interval fields using supervised machine learning (SL) models. Up to this point, spatially- and temporally-varying uncertainty in computational mechanics is modeled using the probabilistic framework of the SFEM and, to a lesser extend, with non-probabilistic frameworks using IFEM. Both approaches require direct prior knowledge about the uncertain properties as a global value for the domain in question. To do so, SFEM uses a random field to distribute the variability of the uncertain property (with prior mean μ and standard deviation σ^2), while IFEM uses interval bounds on a constant uncertain property obtained with prior knowledge of some kind. The recently proposed interval field techniques [10–12, 14] also require some direct information about the uncertain properties for the domain and the spatial variability is determined in a number of ways, some of which are probabilistic in nature.

On the other hand, the SIF only requires data in the form of features in the subspace of the domain where we must find the uncertain properties (the target values)—given that the model has been trained on enogh data. In this fashion, we are able to predict the uncertain properties without making assumptions on its probability distribution, such as would be the case with a Random field.

In order to develop the SIF, the trained models presented in Sections **??** are used to obtain a prediction at each spatial coordinate $\mathbf{x}$ in the domain. These point estimates are in turn used as the interval midpoint values. Interval uncertainty bounds can be obtained with expert knowledge or using method such as Interval Monte Carlo [22]. In this work, interval bounds are set at 10% for the experiments.

4.1.1 Deep LSTM Network

Deep LSTMs are the driving force behind recent advancements in time-dependent applications such as speech recognition [23], time series prediction [24], and image captioning [25]. Using deep LSTMs, a model can learn to extract complex abstractions as sequential data representations. Additionally, using deep LSTMs can allow for direct inference on images, which can be very helpful for structural health monitoring and other applications where image data are routinely obtained.

In order to obtain the hidden state vector representation at time t, each of the gates in the LSTM units performs the following computations:

$$\begin{aligned} i_t &= \text{sig}(W_{hi}h_{t-1} + W_{xi}x_t) \\ f_t &= \text{sig}(W_{hf}h_{t-1} + W_{hf}x_t) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{hc}h_{t-1} + W_{xc}x_t) \\ o_t &= \text{sig}(W_{ho}h_{t-1} + W_{hx}x_t + W_{co}c_t) \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{2}$$

Figure 1: LSTM cell architecture

where the i_t, f_t, o_t are the input, forget, and output gates, respectively. The $sig(\cdot)$ and $tanh(\cdot)$ are the logistic sigmoid activation function and hyperbolic tangent function, respectively. The weight matrices are represented by the recurrent weight matrices W_h and the input-to-hidden weight matrices W_x. Fig. 1 shows a diagram of an LSTM unit.

To train the model, we use the Adam optimizer [26] and add dropout regularization [27, 28] to prevent overfitting.

4.1.2 XGBoost

This section presents Extreme Gradient Boosting (XGBoost), a state-of-the-art ensemble algorithm that is broadly easier to implement than deep LSTMs while being a robust predictor. Nonetheless, unlike deep LSTMs, XGBoost as well as many SL algorithms do not explicitly handle spatial or temporal dependencies. Hence, we present it to use as comparison with the deep LSTM model and because it can be valuable for some engineering SIF applications.

Ensemble algorithms work by combining different models in order to reduce the overall variance and bias of the prediction. The most popular ensemble methods are bagging [29] and boosting [30]. In Bagging, different models or different algorithms are combined in a way that reduces the variance of the prediction, while in Boosting, different *weak* models are combined to reduce the overall bias and variance of the prediction. For regression problems, generally an average of the learners is taken as the final prediction for each instance.

XGBoost is a tree ensemble algorithm that uses CART regression trees [31] as its base learner. XGBoost [32] was introduced in 2014 and has become a dominant algorithm for structured data that has won numerous data science competitions such as *the HiggsML challenge* [33], and *ACM RecSys Challenge* [34]. XGBoost is an updated variant of Gradient Tree Boosting [35] and provides higher speed, more regularization options, and greater scalability.

In order to train the model, the objective function minimizes a regularized loss composed of the training loss $\mathcal{L}(\Theta)$ and a regularization term $\Omega(\Theta)$ to prevent overfitting. XGBoost optimizes the objective function additively. For the data instance i in iteration t, the prediction of the model is the sum of the individual predictions of the k trees in the ensemble, as follows:

$$\hat{f}^{(t)}(x_i) = \sum_{k=1}^{t} \hat{f}_k(x_i) = \hat{f}^{(t-1)}(x_i) + \hat{f}^{(t)}(x_i) \tag{3}$$

The objective function to be minimized at every iteration t is

$$\mathcal{L}^{(t)} = \sum_{i=1}^{N} \mathcal{L}(\Theta) + \Omega(\Theta) = \sum_{i=1}^{N} \mathcal{L}(y_i, \hat{f}_t(x_i)) + \Omega(f_t) = \sum_{i=1}^{N} \mathcal{L}(y_i, \hat{f}^{(t-1)}(x_i) + \hat{f}^{(t)}(x_i)) + \Omega(f_t) \tag{4}$$

This optimization problem is solved using first and second-order gradient statistics as shown on [32].

In our experiments we show that the Deep LSTM network is superior to XGBoost. Nonetheless, engineers might find that XGBoost performs well for their particular application and gain time savings from its implementation and training time.

4.1.3 Model Training and Architectures

Algorithm 1 is used for the SIF.

We show results for XGBoost and deep LSTM network on the experiment in Section 5. Figure 2 shows the different learning models f used for the SIF.

Algorithm 1 SIF Algorithm

1: Obtain training set $\mathcal{T}$
2: Obtain testing set $\mathcal{D}$
3: Choose learning model f
4: Choose loss function $\mathcal{L}$
5: Obtain interval uncertainty bounds $\mathcal{U}$
6: **for** $i = 1 : M$ instances **do**
7: Train f to minimize a training loss $\mathcal{L}(\Theta)$
8: Obtain prediction $\hat{f}(x_i)$
9: **end for**

Figure 2: Model Architectures
Left: Simple XGBoost model
Right: Deep LSTM model stacked with feed-forward layer

4.1.4 A Unified Framework: SIF Discretization for IFEM

The SIF predictions are independent of the IFEM mesh. Thus a post-processing discretization of the SIF to match the elements in the mesh of the IFEM is required. The procedure is as follows: the spatially- and/or temporally- varying uncertain properties of the domain obtained with the SIF are midpoint input values to the IFEM in order to obtain the interval responses. In particular, the discretization of the SIF for the purpose of IFEM analysis is performed by taking the expected value of the predictions of the SIF corresponding to each the size l of each finite element in the mesh, which gives the interval midpoint value I_n^{mid} at element n, as follows:

$$I_n^{mid} = \mathbb{E}_l[\hat{f}(x_i)], \quad \text{where} \quad \hat{f}(x_i) \text{ is the model prediction for sample } i. \tag{5}$$

After the discretization, a conventional IFEM is performed with the postulated uncertainty bounds.

5 Experiments

The experiments performed consist of using the SIF method to predict the material properties of soil-column layers at various locations in a 3D domain to produce the interval field, which later is discretized as material input to the IFEM. The soil layer properties were obtained at the University of Massachusetts-Amherst by Prof. Dr. Paul W. Mayne of the Georgia Institute of Technology. They were obtained via CPT tests [36], which are performed with two separate measurement devices inserted from the surface into each soil column obtaining soil properties with respect to depth for about 15 m. In particular, Test 1 records three different measurements $\{q_c, fs, u_2\}$ [36], while Test 2 records the shear wave velocity $\{V_s\}$.

Then, we use a subset of the data and denominate it as the training set $\mathcal{T}$, with the feature space X consisting of the measurements of Test 1, and the target space Y consisting of the measurements of Test 2 (shear wave velocity V_s for each data point). Once the model is trained, we predict the target values for shear wave velocity for a subset of unseen data (testing set $\mathcal{D}$) by querying the trained model with only the features X of subset $\mathcal{D}$. Thus, by using the SIF method, we can produce the interval field based on feature data alone without making any assumptions on the distribution of the target values. A hypothetical scenario for this application would be one where a subset of the data in Test 2 are corrupted or missing, or that only performing the first test on a portion of the domain results in significant cost and time savings. Further, for other applications, such as aerospace batch manufacturing and structural health monitoring, the SIF approach is superior to the random fields assumptions of the SFEM, since SIF does not require assumptions on the probability distributions of the unknown properties.

5.1 Datasets

The dataset consist of text files corresponding to the CPT results for 15 different 'soil column' locations in the same site. Fig. 3 shows examples for features and target values at two distinct locations.

Figure 3: Dataset Samples
Top and bottom: examples of data features (left) and corresponding targets (right).

5.2 Model Evaluation

Randomized Search [37] was used for the hyperparameter optimization of the models. In order to account for the effect of the uncertainty bounds, the inputs X for the testing set $\mathcal{D}$ were queried under four different cases using the trained deep LSTMs, as follows: (1) Raw inputs X without any postulated uncertainty added (i.e. midpoint input); (2) Lower bound inputs $\underline{X}$ with -5% uncertainty bounds added; (3) Upper bound inputs $\overline{X}$ with +5% uncertainty bounds added; (4) Raw inputs X with noise multiplier $\epsilon \sim \mathcal{N}(0, 0.05)$. Table 1 shows that the deep LSTM outperforms XGBoost in these experiments. Fig 4 shows the predictions for the testing column using the different LSTM model inputs. Given that the uncertainty bounds are small (10% total) the predictions are similar. In the SIF-IFEM example show in Sec. 5.3, the raw input X is used to obtained the material predictions.

Table 1: R^2 Coefficients for Testing Set

Model	R^2 **Coefficient**
LSTM (X)	0.915
LSTM ($\underline{X}$)	0.909
LSTM ($\overline{X}$)	0.875
LSTM (X+noise)	0.896
XGBoost*	0.228

*XGBoost is only tested with raw features.

Figure 4: Predictions on the testing set for the different LSTM models used.

5.3 IFEM Example using SIF

The trained model for the SIF was used to predict the shear wave velocities in the testing set $\mathcal{D}$ using raw inputs X. A comparison is done with the SFEM, where the parametrization consists of the mean and standard deviation of the prior distribution. To obtain the parameters of the prior for the SFEM, we use the shear wave velocities stored in the training set $\mathcal{T}$ and compute the mean and standard deviation. In order to proceed to the FE analysis, the shear wave velocities $\{V_{s,i}\}$ are converted to moduli of elasticity E_i, for every sample i, as follows:

$$\begin{aligned} E &= 2G_{max}(1+\nu) \\ G_{max} &= V_s^2 \rho_t \\ \text{where} \quad \rho_t &= \gamma_t / g \\ \gamma_t &= 26 - \frac{14}{1 + (0.5 \log(f_s + 1))^2} \\ f_s &= \text{CPT measurement (kPa)}. \\ \nu &= 0.15 \end{aligned} \tag{6}$$

Then, without loss of generality, a cantilever structure meshed with 10 beam equal-length elements is employed to show the difference between the SIF-IFEM and the SFEM, as shown on Figure 2. The length of the beam is $L = 10\ m$, the length of each element n is $l = 1\ m$, the distributed load is $q = 1\ kN$, and the mid-point interval material elasticity for each element E_n is determined using Eq. 5, and shown on Table 2 (obtained from raw input predictions). 10% uncertainties are assigned to all the elements E_n.

The comparison shows that the SFEM has a 22.7% error from the deterministic solution at the tip of the cantilever, while the IFEM closely encloses the solution with a postulated 10% level of uncertainty. As shown on Fig. 4, the SIF predictions are very near ground truth point values. Moreover, using the SIF discretization procedure for IFEM (SIF-IFEM) of Sec. 4.1.4, smoothens the mesh element material properties and brings it to close agreement with ground truth values. The large error of the SFEM from deterministic values could even be larger if the soil layers had a more drastic stiffness change (e.g., from soft soil to hard rock).

Table 2: Material Properties used for IFEM at each beam element

Element Number	E (Ground Truth) (MPa)	E (Prediction) (MPa)	% Error
1	139.4	142.7	2.4
2	134.2	132.4	1.3
3	101.3	97.8	3.5
4	74.1	72.9	1.6
5	78.1	71.4	8.6
6	76.3	72.5	5.0
7	68.3	73.9	8.2
8	75.5	73.9	2.1
9	77.5	72.7	6.2
10	73.6	70.1	4.8

Table 3: Meshed Cantilever Structure

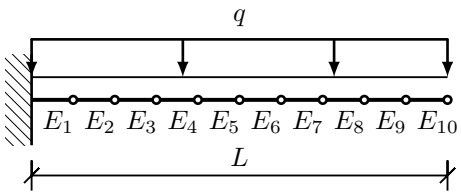

Figure 5: Deflection Comparison of SIF-IFEM and SSEM with Cantilever Beam
IFEM with 10% uncertainties
For SFEM: $\mu = 117MPa$ and $\sigma = 46MPa$

6 Conclusions and Discussion

Engineering systems must contend with great deal of uncertainty. Whenever the spatial- or temporally-varying uncertainty is epistemic and cannot be characterized by a random field, the non-probabilistic interval analysis framework presents an alternative. The corresponding concept of an interval field was not well-developed yet until the introduction of the SIF and the unified framework of the SIF-IFEM, both presented in this paper. The power of the SIF lies in not having to make a-priori assumptions on the probability distribution of the uncertainties. Moreover, the SIF-IFEM framework presents a method to bound the machine learning model predictions provided by the SIF by any given level of uncertainty.

Acknowledgments

We would like to thank Dr. Paul W. Mayne for graciously providing the soil layer data.

References

[1] M. Aoki and G. Rothwell, A comparative institutional analysis of the fukushima nuclear disaster: Lessons and policy implications, *Energy Policy* **53**, 240 (2013).

[2] E. Zio and T. Aven, Industrial disasters: Extreme events, extremely rare. some reflections on the treatment of uncertainties in the assessment of the associated risks, *Process Safety and Environmental Protection* **91**, 31 (2013).

[3] Y. Wen, B. Ellingwood, D. Veneziano, and J. Bracci, Uncertainty modeling in earthquake engineering, *MAE center project FD-2 report* (2003).

[4] E. Vanmarcke, *Random fields: analysis and synthesis* (World Scientific, 1983).

[5] R. G. Ghanem and P. D. Spanos, *Stochastic finite elements: a spectral approach* (Courier Corporation, 2003).

[6] H. Zhang, Ph.D. thesis, Georgia Institute of Technology (2005).

[7] C. Jiang, B. Ni, N. Liu, X. Han, and J. Liu, Interval process model and non-random vibration analysis, *Journal of Sound and Vibration* **373**, 104 (2016).

[8] D. Moens and M. Hanss, Non-probabilistic finite element analysis for parametric uncertainty treatment in applied mechanics: Recent advances, *Finite Elements in Analysis and Design* **47**, 4 (2011).

[9] N. Xiao, Ph.D. thesis, Georgia Institute of Technology (2015).
[10] A. Sofi, Structural response variability under spatially dependent uncertainty: stochastic versus interval model, *Probabilistic Engineering Mechanics* **42**, 78 (2015).
[11] A. Sofi, Euler–bernoulli interval finite element with spatially varying uncertain properties, *Acta Mechanica* 1–17 (2017).
[12] D. Wu and W. Gao, Hybrid uncertain static analysis with random and interval fields, *Computer Methods in Applied Mechanics and Engineering* **315**, 222 (2017).
[13] D. Moens, M. Munck, W. Desmet, and D. Vandepitte, in *IUTAM symposium on the vibration analysis of structures with uncertainties* (Springer, 2011), 71–83.
[14] M. Imholz, M. Faes, J. Cerneels, D. Vandepitte, and D. Moens, in *Proceedings of the 7th international workshop on reliable engineering computing* (University of Bochum, 2016), vol. 7, 367–378.
[15] W. Verhaeghe, W. Desmet, D. Vandepitte, and D. Moens, in *Fuzzy Information Processing Society (NAFIPS), 2011 Annual Meeting of the North American* (IEEE, 2011), 1–6.
[16] R. L. Muhanna and R. L. Mullen, Uncertainty in mechanics problems—interval–based approach, *Journal of Engineering Mechanics* **127**, 557 (2001).
[17] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, vol. 1 (Springer series in statistics New York, 2001).
[18] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach* (Malaysia; Pearson Education Limited,, 2016).
[19] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, vol. 1 (MIT press Cambridge, 2016).
[20] S. Hochreiter and J. Schmidhuber, Long short-term memory, *Neural computation* **9**, 1735 (1997).
[21] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, *nature* **521**, 436 (2015).
[22] H. Zhang, R. L. Mullen, and R. L. Muhanna, Interval monte carlo methods for structural reliability, *Structural Safety* **32**, 183 (2010).
[23] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al., Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, *IEEE Signal Processing Magazine* **29**, 82 (2012).
[24] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, and G. Cottrell, A dual-stage attention-based recurrent neural network for time series prediction, *arXiv preprint arXiv:1704.02971* (2017).
[25] J. Lu, C. Xiong, D. Parikh, and R. Socher, Knowing when to look: Adaptive attention via A visual sentinel for image captioning, *CoRR* **abs/1612.01887** (2016), arXiv:1612.01887.
[26] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980* (2014).
[27] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, *arXiv preprint arXiv:1207.0580* (2012).
[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, *The Journal of Machine Learning Research* **15**, 1929 (2014).
[29] L. Breiman, Bagging predictors, *Machine learning* **24**, 123 (1996).
[30] Y. Freund, R. Schapire, and N. Abe, A short introduction to boosting, *Journal-Japanese Society For Artificial Intelligence* **14**, 1612 (1999).
[31] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees*, The Wadsworth and Brooks-Cole statistics-probability series (Taylor & Francis, 1984), ISBN 9780412048418, https://books.google.com/books?id=JwQx-WOmSyQC.
[32] T. Chen and C. Guestrin, in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (ACM, 2016), 785–794.
[33] C. Adam-Bourdarios, G. Cowan, C. Germain-Renaud, I. Guyon, B. Kégl, and D. Rousseau, in *Journal of Physics: Conference Series* (IOP Publishing, 2015), vol. 664, 072015.
[34] M. Volkovs, G. W. Yu, and T. Poutanen, in *Proceedings of the Recommender Systems Challenge 2017* (ACM, 2017), 7.
[35] J. H. Friedman, Greedy function approximation: a gradient boosting machine, *Annals of statistics* 1189–1232 (2001).
[36] P. W. Mayne, *Cone penetration testing*, vol. 368 (Transportation Research Board, 2007).
[37] J. Bergstra and Y. Bengio, Random search for hyper-parameter optimization, *Journal of Machine Learning Research* **13**, 281 (2012).