Data-Centric Engineering (2021), 2: e11
doi:10.1017/dce.2021.13

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

On generative models as the basis for digital twins

George Tsialiamanis*, David J. Wagg, Nikolaos Dervilis and Keith Worden

Dynamics Research Group, Department of Mechanical Engineering, University of Sheffield, Sheffield, United Kingdom
*Corresponding author. E-mail: g.tsialiamanis@sheffield.ac.uk

Received: 25 May 2021; **Revised:** 29 June 2021; **Accepted:** 02 July 2021

Keywords: cGAN; digital twins; generative adversarial network; generative models; mirror models; stochastic finite elements

Abstract

A framework is proposed for generative models as a basis for *digital twins* or *mirrors* of structures. The proposal is based on the premise that deterministic models cannot account for the uncertainty present in most structural modeling applications. Two different types of generative models are considered here. The first is a physics-based model based on the stochastic finite element (SFE) method, which is widely used when modeling structures that have material and loading uncertainties imposed. Such models can be calibrated according to data from the structure and would be expected to outperform any other model if the modeling accurately captures the true underlying physics of the structure. The potential use of SFE models as digital mirrors is illustrated via application to a linear structure with stochastic material properties. For situations where the physical formulation of such models does not suffice, a data-driven framework is proposed, using machine learning and conditional generative adversarial networks (cGANs). The latter algorithm is used to learn the distribution of the quantity of interest in a structure with material nonlinearities and uncertainties. For the examples considered in this work, the data-driven cGANs model outperforms the physics-based approach. Finally, an example is shown where the two methods are coupled such that a hybrid model approach is demonstrated.

Impact Statement

In the current work, a generative model framework for digital twins is proposed. The framework is fitted to a mathematical formulation of digital twins, also referred to as mirrors, from the previous works. Different types of generative models are presented and tested on simulated data. The physics-based model is a stochastic finite element model, widely used to model structures under uncertainty. The data-driven method is a conditional adversarial network, whose functionality fits the framework of mirrors. The hybrid-model is the combination of the two aforementioned methods. All models are tested according to their ability to model uncertainty. Finally, the models are considered as two different types of mirrors, ε-mirrors and α-mirrors, which have different functionality.

1. Introduction

A recent innovation in the field of system simulation is the creation of *digital twins* for specific systems (called physical twins). For example, attempts have been made to do this in the fields of manufacturing (Rosen et al., 2015; Uhlemann et al., 2017), control systems and the internet of things (Schluse et al., 2018), smart cities (Dembski et al., 2020), social networks, and management (Macchi et al., 2018)—more

detailed literature reviews and descriptions of state-of-the-art research relating to digital twins can be found in the recent review papers (Fuller et al., 2020; Jones et al., 2020; Wagg et al., 2020). Structural dynamics is also a field in which digital twins have been a desired achievement for a number of years—see Wagg et al. (2020) and references therein. One of the motivations for creating a digital twin of a structure (the physical twin) is to enable more accurate prediction of the structure's behavior under a wider range of different situations. For example, predictions could be used to avoid scenarios under which the structure might be more likely to suffer damage or degradation. Equivalently, in the extreme case, the model might be used to limit the use of the structure in operating conditions where one might be concerned that some form of structural failure might occur (e.g., using a wind turbine in higher wind speeds than usual). In this context, the overall goal of a digital twin can be viewed as maximizing the effective operational life of the structure, and as such, is directly linked to the business objective of minimizing cost (or maximizing profit) associated with the physical twin.

For complex engineering applications, it is not possible to have complete knowledge of all the physics of the structure, including all its possible environmental and operational conditions. Therefore, one of the underlying concepts of a digital twin is that a combination of models is used to capture the overall behavior of the physical twin. In particular, a commonly proposed scenario is that physics-based model(s), such as finite-elements, are combined with data-based techniques, such as machine learning (Bishop, 2006; Murphy, 2012). In addition to this, models can be defined for different parts (or substructures) of a physical twin and then assembled into a larger model. This type of assembled modeling approach was discussed in Worden et al. (2020), where the concept of a *digital mirror* was also introduced to give a more precise mathematical set of definitions, and these definitions will be used as the framework for the results presented in this paper.

A very important part of building a digital twin is to consider the associated uncertainty of the process. This aspect includes both aleatory uncertainty, which refers to events or quantities that are inherently random and cannot be modeled using deterministic physics-based models (e.g., measurement uncertainty; Smith, 2013), and epistemic uncertainty which relates to a lack of knowledge about the properties of the physical twin. A common example of epistemic uncertainty is when the effects of nonlinearity are not captured in a physics-based model, leading to errors between the data acquired from the real structure and the model. Another type of epistemic uncertainty is not knowing all the variables that affect the result of an event. In machine learning, these variables are sometimes referred to as "lurking" variables (Bishop, 2006). Similarly, in probability-based models for uncertainty, such variables are called latent variables (Booyse et al., 2020).

Despite the separate definitions of aleatory and epistemic uncertainty, it will typically be a very challenging problem to quantify these separately within a digital twin. Therefore, the motivation for this work is to use generative models as the basis for a digital twin that can provide estimations of aleatory and epistemic uncertainty. The probabilistic framework of generative models fits naturally with models for aleatory uncertainty, and epistemic uncertainties can be inferred based on variations between the digital twin outputs and recorded data from the physical twin. A related approach has been developed in Booyse et al. (2020) to build a black-box digital twin for a structural health monitoring (SHM) application.

In the current work, two different types of generative models are studied; the first using the stochastic finite element (SFE) method (Sudret and Der Kiureghian, 2000; Ghanem and Spanos, 2003). SFE models are used to propagate uncertainty from material and loading to quantities of interest via finite element models. They are *white-box* models, directly exploiting knowledge of the physics of the structure. The second type of generative model considered is the *conditional generative adversarial network* (cGAN) (Mirza and Osindero, 2014). Using this algorithm, one can generate samples of learnt distributions, conditioned on a set of variables. These distributions are of the structural quantities that the model is built to predict. Because cGAN is a machine learning algorithm, it should be able to perform for a wide range of structural or environmental conditions for which there are data; this is in contrast to SFE models, which are able to perform only under predefined conditions. However, the machine learning model is limited to a set of system conditions for the data gathered, and cannot extrapolate beyond this, which is in contrast to the SFE model that can be used in a wider predictive role. A hybrid approach, using both generative models, is

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

an attempt to get the best aspects of both models. Specifically, what is usually expected from hybrid approaches (gray-box models) is: (a) to use the cGAN algorithm to correct the discrepancy of the SFE model in cases where the physical formulation of the finite elements do not suffice and (b) to allow the hybrid model to have some predictive capability based on the SFE model away from the operational conditions where data are available; that is, extrapolation capability.

The main thesis of this paper is that one should adopt generative models to properly accommodate uncertainty in potential digital twins. To present this idea, specific modeling technologies are used to illustrate the various shades: stochastic FE for a generative white box and the cGAN for a generative black box; combined together, these present a fully generative gray box. The presentation is not intended to suggest that these model types are the *only* possibilities; in fact, a range of paradigms could prove equally powerful. In terms of white generative models, a fairly basic implementation of the stochastic FE method has been presented here, based on the early polynomial-chaos formulation of Ghanem and Spanos (2003); however, more recent variants, like the stochastic Galerkin approach (Augustin and Rentrop, 2012), have advantages like more general expansion bases for the stochastic space. A very recent methodology *StatFEM* (Duffin et al., 2021; Girolami et al., 2021), provides an elegant Bayesian framework for both building and updating generative FE models. In terms of generative black-box models, GANs are by no means the only option; in fact, one alternative—the *variational auto-encoder* (Kingma and Welling, 2014)—has already proved to be generally useful in engineering problems; particularly in condition monitoring problems (e.g., Mylonas et al., 2020). Another versatile generative framework is provided by *Gaussian processes* (GPs) (Rasmussen and Williams, 2005). Although GPs are most often used as nonparametric black-box learners, they are also the basis for the StatFEM models mentioned earlier. Furthermore, GPs offer a direct method of building gray-box models by training them from data, but building a priori physics into their mean and kernel functions (Rogers et al., 2020; Pitchforth et al., 2021; Cross et al., 2022).

The layout of the paper is as follows. In Section 2, mirrors are defined based on the work in Worden et al. (2020). In Section 3, details of the GANs and cGANs are given. In Section 4, a white-box mirror based on a SFEM model is presented. In Section 5, a black-box mirror based on a cGAN is described. In Section 6, the combination of the two models into a hybrid mirror is presented and also the extrapolation potential of both the hybrid and the black-box mirrors is studied. Finally, the results are summarized and conclusions are drawn. Further details about the SFE method can be found in the Supplementary Appendix.

2. Digital Mirrors

Although the term digital twin has been widely used in many disciplines and in industry sectors, an alternative terminology is used here to make the subsequent analysis more precisely defined. The physical twin (also called the structure and denoted S) has N_S different states $\underline{s} = \{s_1, s_2, \ldots s_{N_S}\}$ (in contrast to Worden et al. [2020] where each state refers to a time instant t, the notation here is simplified for convenience and since the problems to be presented here are not dynamic but static). Together with the structural states, the environment E of the structure has N_E corresponding states, $\underline{e} = \{e_1, e_2, \ldots e_{N_E}\}$. In general, the set $\underline{s}$ may contain specific displacements (or accelerations when dynamic problems are considered) that are of interest from a structure, or stresses, strains, and response spectra that one might monitor in a SHM scheme. The set $\underline{e}$ may include environmental conditions affecting the structure such as temperature, humidity, wind speed, and so forth.

The approach taken in the current work is to build mirrors that predict the behavior of different parts of the structure S, rather than a global model for the whole S. This subset of states/quantities that are mirrored defines the *context* $C = \left\{ e_i^C \in E, s_j^C \in \underline{s}; i, j \right\}$, where s_j^C is the *response* or *predictive context* and e_i^C is the environmental context. This formulation is used to define the exact quantities that the model is able to predict and the exact environmental conditions under which the model is able to perform.

Following Worden et al. (2020), definitions are constructed according to the mirror's ability to predict the states s_j^C. This ability is measured using metrics defined below, and based on this the mirror models can

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

be considered to be either *ε-mirrors* or *α-mirrors*. A model is considered to be an ε-mirror if a distance metric, d^C, is less than (or equal to) a predefined tolerance, ε, such that,

$$d^C\left(\underline{p}^C, \underline{r}^C\right) \leq \varepsilon, \tag{1}$$

where $\underline{p}^C$ is the prediction of the digital mirror within some context C and $\underline{r}^C$ is the observation of the response of the structure. The definition simply implies that the response of the physical structure should be within some distance of the prediction of the mirror. If the mirror is based on a deterministic model, then the distance defines some interval or area in the prediction space similar to confidence intervals. Given that the models that are studied here are probabilistic, d^C should be some probability distribution distance metric defining the maximum distance between the predicted and real probability distributions of interest.

The second type of mirror is the α-mirror. In order for stochastic model to be considered an α-mirror, the quantity of interest of the real structure should always be within an interval defined by the output of the model with a given probability p, that is,

$$P\left(r_i^C \in \left[\overline{m}_M^C - \alpha\sigma_M^C, \overline{m}_M^C + \alpha\sigma_M^C\right]\right) = P(\alpha), \tag{2}$$

where r_i^C is an observation, P is the probability that the observation is within the defined interval, M is the model used as a mirror, m_M^C is the prediction of the model or the mean value of the outputs in the case of a generative model M, σ_M is the corresponding standard deviation, $P(\alpha)$ the probability as a function of the predefined parameter α, which controls how wide the interval defined in Equation (2) is. The probability P is a function of the parameter α and for every mirror, such a function can be defined, according to available data. The curve $P(\alpha)$ can be used to explain the potential of the generative mirror in describing the behavior of the structure S exploiting only the mean value and the standard deviation of the generated-by-the-model samples. Such an approach provides a way to define an interval, regardless the shape of the distribution, within which all observations r_i^C would fall into with some probability $P(\alpha)$.

A distinction is introduced within the environmental parameters. The first category is the *controlled variables* ($\underline{e}_c^C$), which are the variables that are used as deterministic inputs into the mirror M. They are quantities that are measured from the environment of the structure and whose effect on the behavior is modeled. The second category is the *uncontrolled variables* ($\underline{e}_u^C$), which include parameters that affect the structure but are either unknown or stochastic. A generative model M_u^{EC} that makes the best estimate of $\underline{e}_u^C$ is needed (in the case of SFEs, as will be explained later, M_u^{EC} is the stochastic process used for the random quantities of the problem). Given that the model M used as a mirror is a generative model, the model's output under the context C is a probability density function P of the prediction $\underline{p}^C$ of the quantities of interest given by,

$$P_{\underline{p}^C} = M\left(\underline{e}_c^C, \underline{\hat{e}}_u^C = M_u^{EC}\right), \tag{3}$$

where $P_{\underline{p}^C}$ is the probability density function of the quantity of interest.

Furthermore, following Worden et al. (2020), the definition of a virtualization is provided. Given some context C, a virtualization is defined as the pair,

$$V^C = \left(M_{\varepsilon_1}^C, M_{u|\varepsilon_2}^{EC}\right), \tag{4}$$

where $M_{\varepsilon 1}^C$ is a model calibrated according to data from the physical structure and an established ε-mirror for some tolerance ε_1 within the context C and $M_{u|\varepsilon_2}^{EC}$ is modeling the stochastic uncontrolled variables in the context, C, which is also an ε-mirror with tolerance ε_2, that provides the best estimate for the unknown parameters. A generative model can be considered a virtualization with clearly separated stochastic and deterministic inputs. The stochastic inputs are modeled by $M_{u|\varepsilon_2}^{EC}$ and the model $M_{\varepsilon 1}^C$ is informed from $M_{u|\varepsilon_2}^{EC}$ as well as from deterministic parameters to generate probability distributions of the quantities of interest of the outputs; for example, displacements, natural frequencies, and accelerations.

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

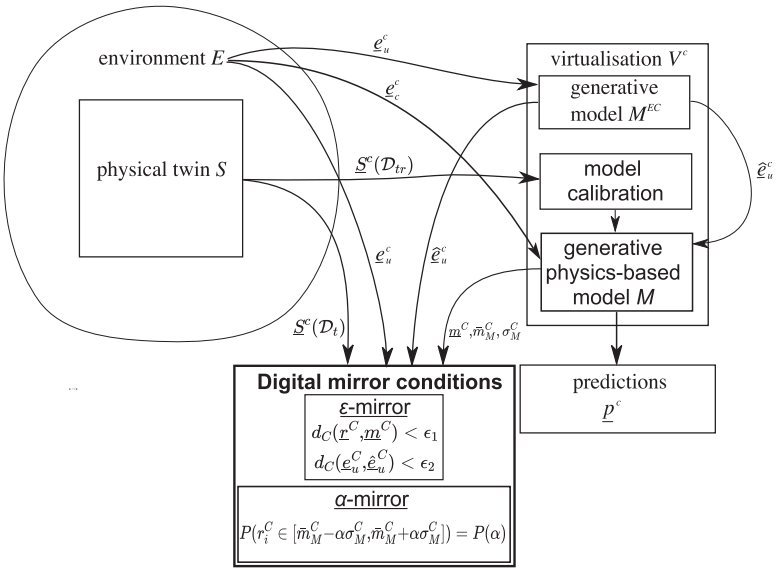

Figure 1. *Schematic representation of the proposed framework for a digital mirror.*

The framework is schematically shown in Figure 1. As mentioned, the uncontrolled variables $\underline{e}_u^C$ from the environment are modeled by a generative model M_u^{EC}. Of all the data acquired from the physical twin ($\underline{S}^C$), a subset is considered to be the training data, $\mathcal{D}_{tr}$, used to calibrate the digital mirror model, that is, $\underline{S}^C(\mathcal{D}_{tr})$. The calibrated model is used to yield predictions. Since it is a generative model, some stochastic input is used, which in this case is the best estimate of the uncontrolled environment variables $\hat{\underline{e}}_u^C$. Some controlled variables of the environment are also used as inputs $\underline{e}_c^C$. As far as the evaluation of the model M as a digital mirror is concerned, using some testing data instances $\underline{r}_C \in \underline{S}^c(\mathcal{D}_t)$ (where $\underline{S}^c(\mathcal{D}_t)$ are acquired data from the physical twin and considered the testing data) and Equations (1) and (2), the parameter ε_1 and the curve $\alpha \rightarrow P(\alpha)$ are defined; the latter two describe the ability of the model to perform as a digital mirror. It is worth noting that for a generative model, the distance d^C is computed between the probability density function of the predictions $P_{\underline{p}^C}$ and the probability density function of the recorded data of the quantity of interest $P_{\underline{r}^C}$. Finally, the model is used to get a probability density function of predictions $P_{\underline{p}^C}$ corresponding to new values of the controlled variables.

3. Generative Adversarial Networks

The SFE method is a white-box physics-based generative modeling method that can be used as a mirror of a structure. The method's performance is largely based on the knowledge one has about the physics of the problem and the finite element formulation. As an alternative, and trying to avoid unnecessary epistemic uncertainty problems, a machine learning black-box solution to the problem is proposed here.

For the purposes of using generative models as mirrors of structures, a very recently developed neural network architecture is used here. The core algorithm is the *generative adversarial network* (GAN) (Goodfellow et al., 2014) and a variation of it, the cGAN (Mirza and Osindero, 2014). The latter algorithm is used exactly in the same way as an SFE model is used. A deterministic input to the model is defined and the model generates distributions (or samples) of the output quantities. In the current section, the two algorithms are presented and their functionality is explained.

3.1. Vanilla GANs

The traditional scheme followed in machine learning is the training of a model to perform classification (Bishop, 1995) or regression (Specht, 1991). To extend this to images, convolutional neural networks

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

were developed (Krizhevsky et al., 2012), yielding superior performance in the two mentioned tasks. Recently, a new type of neural network has emerged, the GAN (Goodfellow et al., 2014). The goal of this new scheme was initially to generate images that resemble reality. This task is achieved via the use of *two* neural networks. The first one is termed the *generator* and produces "fake" images given a latent noise vector. The second network is the *discriminator*, which tries to identify whether an image, fed to it as an input, is fake (generated by the generator) or real (coming from the available dataset). By training, both of these networks improve toward their objectives and finally, the generator, provided with some latent vector, can generate images that appear to be real. More intuitively, this means that the generator maps a latent vector distribution into a distribution or a manifold of the real data. The layout of the basic (vanilla) GAN can be seen in Figure 2.

The generator is commonly a multilayer perceptron (MLP) (Bishop, 1995) that takes as input a latent noise vector $\mathbf{z}$ coming from a probability distribution $p_z(\mathbf{z})$ and maps it into a vector (or an image) $G(\mathbf{z})$ of dimension equal to the dimension of the training samples. The discriminator is another MLP that takes as inputs, vectors (or images) $\mathbf{x}$, and outputs the probability of the sample being real, $P(\mathbf{x} = real) = D(\mathbf{x})$. The training of the discriminator is carried out by maximizing the probability that it assigns the correct label ("real" or "fake") to the samples. At the same time, the training of the generator, G, is accomplished by trying to minimize the probability that the discriminator classifies the generated samples as fake, that is, minimization of $\log(1 - D(G(z)))$. Following from Goodfellow et al. (2014), the objective function $\mathcal{L}$ can be interpreted as a two-player game explained by,

$$\min_G \max_D \mathcal{L}(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[\log 1 - D(G(\mathbf{z})))]. \tag{5}$$

Training of such a network is performed in two steps per epoch. During the first step, random samples are created by the generator and concatenated with a batch of real samples from the dataset. The resulting training batch is used to train the discriminator for one epoch by back-propagating the error of the output. The target label for the real samples is 1 and for the generated ones is 0. The first term of the right-hand side of Equation (5) is set in this step as the objective function and its *maximization* is attempted. Consequently, the two networks are clipped together as in Figure 2, and random samples of the latent vector are generated to create random-generated samples. These samples are fed into the whole GAN assembly and the target outputs are labels of 1. The weights of the discriminator's connections are considered as constants during the second training phase and the error is back-propagated to train only the generator. This time, the objective function is composed exclusively of the second term of the right-hand side of Equation (5) and its *minimization* is sought. Following this training scheme, during the first step, the discriminator learns to distinguish between real and generated images and the generator to generate images that the discriminator classifies as real and (as shown in Goodfellow et al., 2014) to have probability distribution similar to the real data.

The most straightforward application of GANs is to generate artificial data to augment a dataset. Training neural networks is highly dependent on the size of the available dataset. The rule-of-thumb for training neural networks that generalize well (Tarassenko, 1998) specifies that for each trainable weight of the neural network, 10 training samples are needed. This statement probably does not stand for GANs, as they are also trained using random noise and generated samples that do not come from the available dataset. Acquiring engineering data is difficult and sometimes even expensive. Labeled images are hard to obtain and their manual labeling costs both time and money. In cases of image datasets, augmentation can

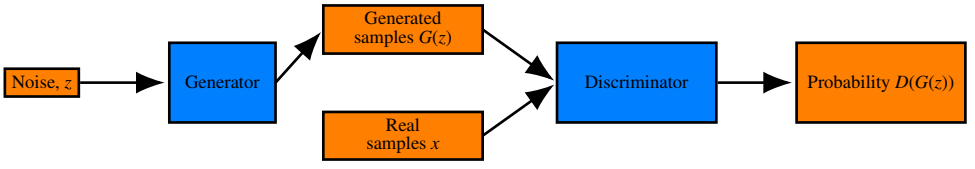

Figure 2. *Vanilla GAN layout.*

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

also be achieved by rotation of the pictures or color change, and so forth. In SHM, for example, where acquiring sufficient data is vital to efficiently monitor the health state of structures, the securing of data from structures in different damage cases or under different environmental conditions can be very expensive or even impossible; the samples are usually limited and augmentation is not trivial. Especially for deep networks, and even more for deep convolutional neural networks, where the number of trainable parameters is huge, augmentation of available dataset size could yield an efficient way to increase the generalization performance of models (Frid-Adar et al., 2018).

3.2. Conditional generative adversarial networks

cGANs are an attempt to control the output of the generator by conditioning on some variables. In contrast to the traditional GAN layout (Figure 2), where the product of the generator is completely controlled by random noise **z**, the output of the generator is here partially controlled by some vector c; thus providing a way of learning distribution and manifold transformations parameterized on the code. This code may be a continuous variable or a discrete one. Since the output of the generator depends on the code, the output of the discriminator should also depend on it and the discriminator should also have it as an input. Therefore, the layout of the cGAN is as shown in Figure 3.

Training such an assembly of networks, the discriminator learns that, for each different value of c, a different acceptance or rejection boundary is defined in the sample space. Since the decision boundary of the discriminator varies according to the code, the generator also learns to vary its outputs according to c to "fool" the discriminator. In cases of discrete or categorical variables, the result is that the generator learns to generate samples belonging to different categories. In Mirza and Osindero (2014), an illustration of this result is presented for the MNIST dataset; a collection of hand-drawn digits of numbers. By defining 10 binary categorical variables as the code, the generator is able to create sample images in predefined classes, controlled by the code.

The use of continuous variables yields more convenient results for physics modeling. A continuous variable would force the mold (the boundary around the manifold of the data) created by the discriminator, to be gradually transformed as a function of the values of the code. The decision boundaries of the discriminator then force the generator to create samples within the region they define. Consequently, the geometry of the generated manifolds is conditioned on the code vector c. Using this training scheme, the generator has learnt the transformation of the manifold and the distribution of the generated points, as a function of the code.

The algorithm may be exploited to generate artificial data as a function of some code vector but, in the current work, it shall also be exploited to learn the transformations of the aforementioned manifolds and distributions, as functions of the code. The output of interest of a generative model is the distribution of some quantity of interest, making the cGAN algorithm a suitable candidate to serve as such a model. The code plays the role of the variable that affects the output distribution and the cGAN is called to learn from the data how the distribution transforms according to the values of the code. The code in a structure modeling context can represent the loading and the environmental conditions of some structure, while the output distributions are the probability distributions of the quantity of interest, that is, displacement,

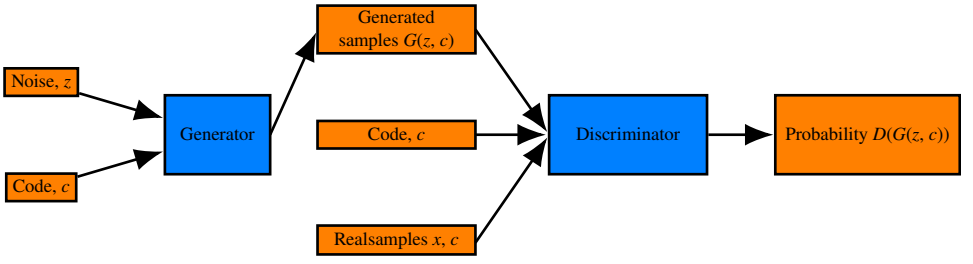

Figure 3. *Layout of a cGAN.*

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

acceleration, natural frequency, and so forth. A major advantage of using such a model as a digital mirror is that there is no need for modeling the uncontrolled variables from the environment ($\underline{e}_u^C$), that is, the generative model M^{EC} step in Figure 1 is bypassed. The effect of these variables is taken into consideration via the noise variables of the cGAN model.

4. SFE Models as Mirrors

An SFE model updated according to data acquired from a structure could be considered as a mirror under the criteria discussed earlier. It could take into account experimental noise and aleatory uncertainty that might exist in a structural problem. The model can also be continuously updated according to newly acquired data, to take into account random events and environmental conditions. In every case, model parameters have to be chosen in order for the model to fit the acquired data.

The parameters of the model that will most probably need calibration are the parameters describing the stochastic fields of the problem. Some assumptions can be made about the fields; the first might be that the field is stationary and *Gaussian* or *lognormal.* A subsequent assumption might then be about the form of the autocorrelation function. A quite common type of autocorrelation function in SFEM problems is the squared-exponential function, such as $\rho(x,x') = \exp -\left(\frac{x-x'}{l}\right)^2$, where x and x' are the points in space and l is a parameter called the *correlation length.*

Having decided on the type of the field and the autocorrelation function, the hyperparameters remaining to be defined are the mean and variance values of the random field (or the mean and variance functions if the field is not stationary). Furthermore, for the aforementioned autocorrelation equation, a third hyperparameter is the correlation length l. Fitting can be done in many ways; the most straightforward is an exhaustive search over some set of candidate parameters for the values that yield the best results.

However, a way to evaluate the performance of such generative models is needed. Since it is a generative model and its output is a distribution, a distance metric between the generated and the real distributions should be used as a performance criterion. The *Kullback–Leibler divergence* (KL divergence) (Kullback, 1997) is a quantity that measures the "distance" between two distributions; it can therefore be used as such a criterion. Regarding mirror terminology, this is the distance ε used to define an ε-mirror. The KL divergence between two distributions P and Q is given by,

$$D_{KL}(P\|Q) = \sum_{n=1}^{n_{val}} P(x) \log\left(\frac{P(x)}{Q(x)}\right), \tag{6}$$

where n_{val} is the number of available datasets to compute the KL divergence between the predicted and the real distributions. (Note that this is the discretized version of the metric.)

Stochastic FEM models take into account uncertainties in the parameters of the structure, but can have a deterministic input. Thus, the output distribution is a function of the input. A model, which might be considered as a mirror of a structure, should be able to perform under different inputs. A simple case to consider is that of a deterministic load input to the model. In this case, the model shall be evaluated for different values of the load and the best one shall be the one with the best average performance among all the cases of deterministic loads. Other deterministic inputs might also be the temperature of the environment, seismic accelerations, humidity, and so forth.

Under the framework of mirrors, the load or any other deterministic input shall be the controlled variable $\underline{e}_c^C$. Any uncertainties, such as Young's modulus, Poisson ratio, and so forth, and unknown environmental parameters are included in the uncontrolled variables $\underline{e}_u^C$. The generative model that estimates the uncontrolled parameters (M_u^{EC}) shall be the stochastic process described by the Karhunen–Loeve expansion. The SFE model will be the generative model that will provide the probability density functions of the quantities of interest.

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

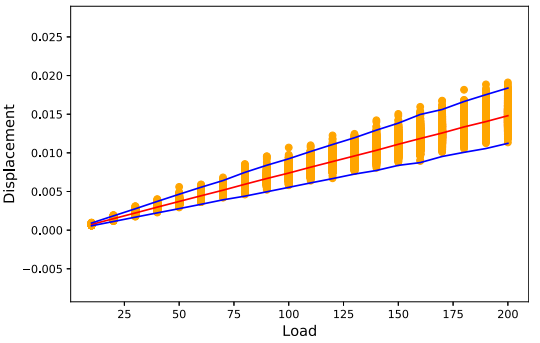

Figure 4. *Samples of tip displacements (orange points), their mean values (red line), and ±3 standard deviations (blue line).*

4.1. Definition of simulation dataset

To test the algorithm, data should be available from some structure of interest. Such data may refer to different deterministic inputs such as load, temperature, and so forth. For every available value of the deterministic input, a set of samples should be available, from which the distribution of the quantity of interest is extracted. The model should perform well in generating distributions close to the ones in the dataset for different values of the inputs. Since SFE models are *white-box* physics-based models, fitting them to data for a set of input values increases the belief that the model will generalize. This assumption is only true if the physical formulation of the model corresponds to the real physical mechanism, that is, if epistemic uncertainty is not present.

To define the required dataset, a simulated structure is considered here to generate data. The structure is a simple cantilever with Young's modulus defined as a stochastic field, similar to the one in Figure 21 in the Supplementary Appendix. In real structures, such cases may be observed in a bridge, for example, when many heat sources affect the temperature of the structure. This situation would result in fluctuations of the stiffness of the structure within its volume. The final field is of course a stochastic field with some correlation function.

The model cantilever here has length equal to 5 (length units), a rectangular cross section with height equal to 0.4, and width equal to 0.1. The stochastic field was chosen to be a stationary Gaussian stochastic field with mean value equal to 2×10^9 and standard deviation equal to $0.2 \times 2 \times 10^9$ (pressure units). The correlation length was 3.0. The procedure described in the Supplementary Appendix was performed, and Equation (16) with order of the expansion $m = 2$ was used to generate realizations of the stiffness matrix of the structure. As an input, a deterministic distributed load along the cantilever with varying values was considered. The values of the load f were $10, 20, 30, \ldots, 200$ force units/length units. The dataset was split into three datasets, one for training, one for validation, and one for testing. Samples of the corresponding tip displacements of the cantilever are shown in Figure 4. For each load value, 1,000 samples were generated. The results comply with the linearity of the problem, since the mean value (red line) is almost linearly increasing as the load increases.

4.2. Model calibration (updating) for a simulated structure

The calibration procedure followed is simply an exhaustive search in a subset of the three-dimensional parameter space. The search is performed over some logical range of values for the parameters. The range could be defined using engineering insight of the problem. The three parameters are the mean Young's modulus (μ_E), the Young's modulus standard deviation (σ_E), and the correlation length (l_{corr}). The model was calibrated using a subset (loads $\{10, 40, 70, 100, 130, 160, 200\}$) of the total set of load cases of the dataset. It is, however, tested in all the cases and some distribution comparisons are shown in Figure 5 for selected load cases. The overall average KL divergence is 0.0028. This value means that the model almost

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

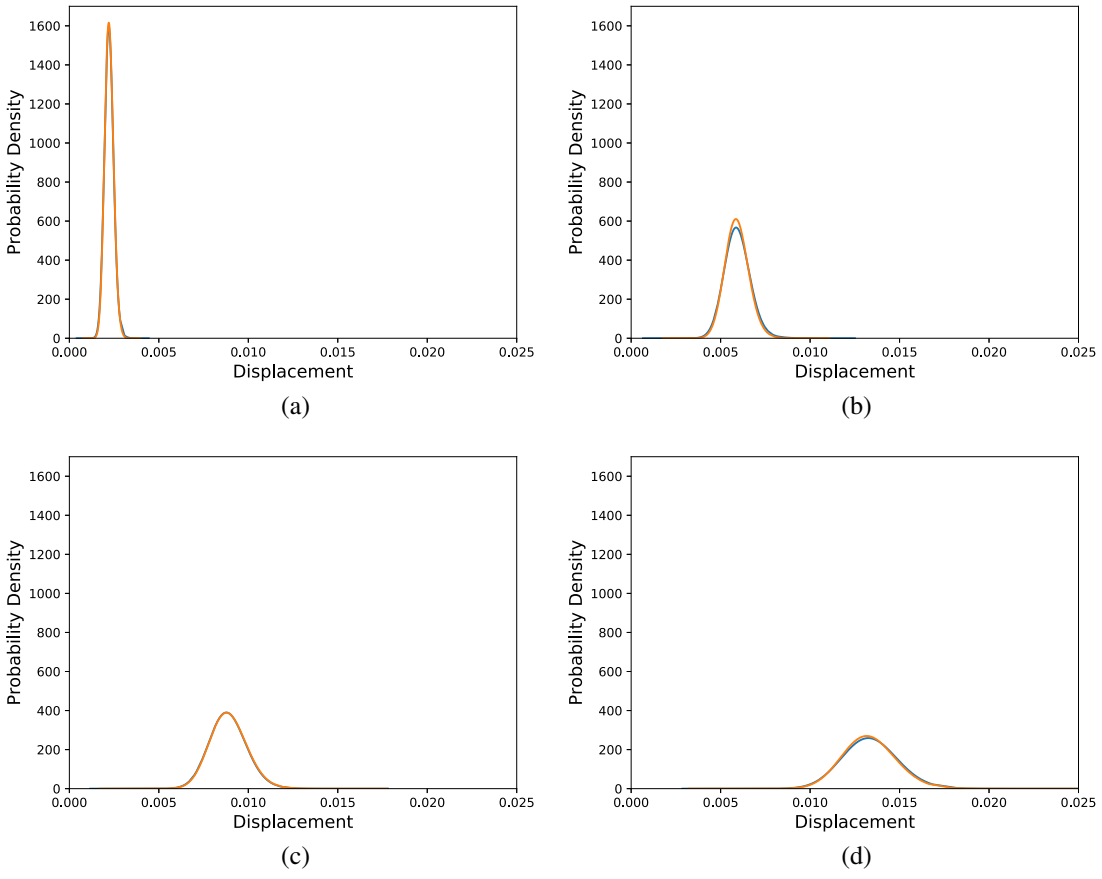

Figure 5. *Distributions of tip displacements corresponding to Monte Carlo samples (orange) and SFEM generated samples (blue) and different load cases, (a) 30 load units, (b) 80 load units, (c) 120 load units, and (d) 180 load units.*

perfectly explains the behavior of the structure and is able to make accurate predictions regarding the distribution of the tip displacements, as evidence, the visual comparisons in Figure 5 are presented.

Of course, the model performs so well because the formulation of the finite elements in the SFE model was exactly the same as the one used to generate the dataset. In real structure situations, epistemic uncertainty may be present. In addition, more uncertain parameters may affect the structure, such as humidity. In the latter case, fitting a model with uncertainty imposed in the Young's modulus, could perform well enough incorporating the uncertainty of the unknown parameters, as uncertainty existing in the stochastic field of the model. In such cases, however, the values of the model parameters from the fitting will not resemble the real values of the random field of the stochastic quantity.

Taking into account the definitions of ε-mirror and α-mirror, the SFE model may serve as both types. For the case of an ε-mirror, the distance metric ε to be used is the KL divergence of the real data from the SFE model outputs. Considering ε equal to the maximum KL divergence, of the available datasets, between the simulated and the predicted by the SFE model distributions. The maximum KL divergence was 0.0029 and so, the SFE model can be an ε-mirror with $\varepsilon \geq 0.0029$ considering engineering judgment or some safety factor.

Regarding the use of the model as an α-mirror, the curve $\alpha - p(\alpha)$ from Equation (2) should be defined. Using only the available data, and various values for the α parameter, Figure 6 shows the probability p as a function of α. For $\alpha = 2$, 90% of the observations fall into the interval defined by Equation (2) while for

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

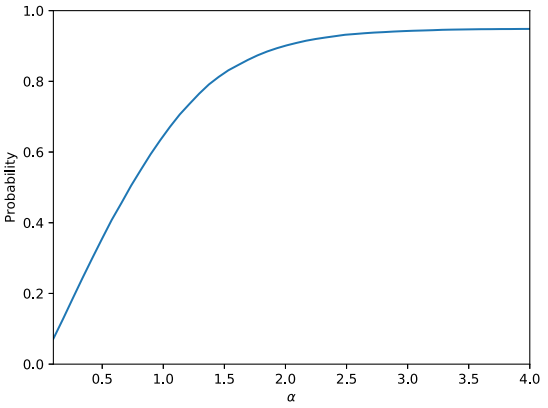

Figure 6. *Probability defined in Equation (2) as a function of the parameter α for the SFE model applied on the linear cantilever case study.*

$\alpha = 3$, 95% of the observations are in the aforementioned interval. A straightforward evaluation of the quality of the model according to this curve is not available. Certainly, as close as this curve is to the corresponding curve of the real data, the better it is. However, this is equivalent (and maybe a more loose evaluation criterion) to the requirement that the distribution of the real and the generated data are similar. The curve represents the potential of the model to explain the structural behavior and can definitely be used as a tool to perform a probabilistic cost–benefit analysis of the performance of the structure, according to the predictions. Moreover, if one is interested only in defining an interval of the potential quantity of interest, under some environmental conditions, the model should, for some value of α, include a sufficient percentage of the observed outcomes.

5. cGAN as Mirrors of a Structure

Since a cGAN is able to generate manifolds of data as a function of a code vector, it fits the desired functionality of a mirror, as described in the current work. Moreover, as stated in Goodfellow et al. (2014), the generator learns, apart from generating data that look real, to generate data close to the distribution of the original data. The same results are expected herein by the use of a cGAN. Two examples are presented below; one refers to the same problem that an SFEM model was used for, in the previous section, and the second is the same problem but with material nonlinearities. Although SFE approaches to nonlinear problems exist (Stefanou, 2009), knowledge about the physics and the source of the nonlinearity is required for their application. The current work is focused on illustrating the convenience of the machine learning approach and its applicability without any knowledge about the nature of the problem (linear or nonlinear) or the source of the uncertainty. Therefore, a nonlinear SFE approach is not considered. Nevertheless, as illustrated in the application of the SFE model on the linear structure, if all the underlying physics of the problem matched the nonlinear finite element formulation (this should include the nature of any noise processes), it would outperform any black-box approach, since there would be no epistemic uncertainty. The cGAN method is a completely data-driven method and it is expected to be able to perform regardless of the linearity or otherwise of the underlying problem.

The cGAN also fits the defined context about controlled and uncontrolled environmental parameters, $\underline{e}_c^C$ and $\underline{e}_u^C$. There is a direct relationship between the cGAN code and the controlled variables and the noise vector and the uncontrolled variables. Defining such a separation could be crucial when one wants to quantify uncertainty. According to the way the cGAN works, the separation of known and unknown parameters and their effect on the predicted distribution is clearly given by the separation of the input vector into the noise and the code. Continuing, two applications of the cGAN are presented showing the versatility of the algorithm and its ability to perform both in linear and nonlinear structural problems.

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

5.1. Application of the cGAN in a linear problem

Using the same dataset as before, a cGAN was trained. The training procedure followed was a standard neural network cross-validation training procedure. The dataset was split into three subsets: training, validation, and testing. The split is performed to train according to the first dataset, select as the best model the one that performs best in the validation dataset and confirm that it is able to perform well on data that it has never "seen" before, that is, the testing dataset. The split was made according to the codes/loads. Each load belonged to only one of the three datasets. More specifically, the samples having loads $\{10,40,70,100,130,160,200\}$ were the training dataset and the ones with loads $\{20,50,80,110,140,170,190\}$ and $\{30,60,90,120,150,180\}$ were the validation and testing sets, respectively.

Both the generator and discriminator are three-layered neural networks here; each has an input layer, a hidden layer, and an output layer. The activation function was chosen to be a hyperbolic tangent function, except for the activation of the output layer of the discriminator, which is a sigmoid function to map to a probability in the interval $[0,1]$. The noise vector was 10-dimensional. Different sizes were tested incrementally regarding the noise vector. It was noted that as the size increased, the performance was increased, as was the convergence speed toward the Nash equilibrium. Therefore, the size chosen here was a 10-dimensional noise vector, since increasing it further did not yield notably better results. The code vector was one-dimensional, since the control variable is only the load. Finally, using different hidden layer sizes in the set $[10,20,30\ldots3,000]$ and selecting the model with the lowest KL divergence in the validation set, the sizes of the hidden layers that yielded good results were 200 nodes for both networks. Using the same size of hidden layer in both the generator and the discriminator might conceal a physical meaning, since the generator decodes the noise into the real feature space and the discriminator maps the feature space into some latent code (in its hidden layer) to distinguish real and fake samples.

The quantity to be minimized is the KL divergence between the generated and the acquired dataset distributions. However, the value of the loss function during training does not directly represent this quantity. The KL divergence is used as a model selection criterion and it is calculated between the generated and acquired distributions for the validation dataset every 100 training epochs. At the end of training, the cGAN instance that had the lowest average KL divergence among the codes of the validation dataset is selected as the most accurate model and is tested on the testing dataset. To define distributions on both the database samples and the cGAN generated ones, kernel density estimates (Epanechnikov, 1969) are fitted in both cases. The kernel used in the current work is a Gaussian kernel. Throughout the paper, every KL divergence is calculated for distributions of neural network outputs. These outputs are scaled to the interval $[-1,1]$ and the bandwidth parameter used to the fitted kernel distributions was in all cases considered equal to 0.1. That is considered an appropriate value, since the range of the outputs is equal to 2; therefore, a bandwidth value equal to $\frac{1}{20}$ of the range yields meaningful distributions about the quantities of interest. This could be another training hyperparameter whose optimization might be the objective of the cross-validation procedure (Silverman, 1981), but was not in the current work.

The best average KL divergence, which was achieved by a model whose generator had 3,000 neurons in its hidden layer, was in the validation dataset 0.081, and in the testing dataset 0.083. Some of the distributions from the testing dataset are presented in Figure 7. It can be seen that the performance is not as good as the performance of the SFEM model, but given that the algorithm is a machine learning algorithm, it is acceptable.

The cGAN model is also tested according to its ability to serve as an ε- and an α-mirror. As far as its potential use as an ε-mirror is concerned, the maximum KL divergence observed on the available testing datasets was 0.168 and therefore, based only on the data, the model can be considered an ε-mirror for $\varepsilon = 0.168$ regarding the distribution of the tip displacement of the cantilever. As far as the ability of the model to serve as an α-mirror is concerned, similarly to Figure 6, the equivalent plot for the cGAN for the linear problem is shown in Figure 8.

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

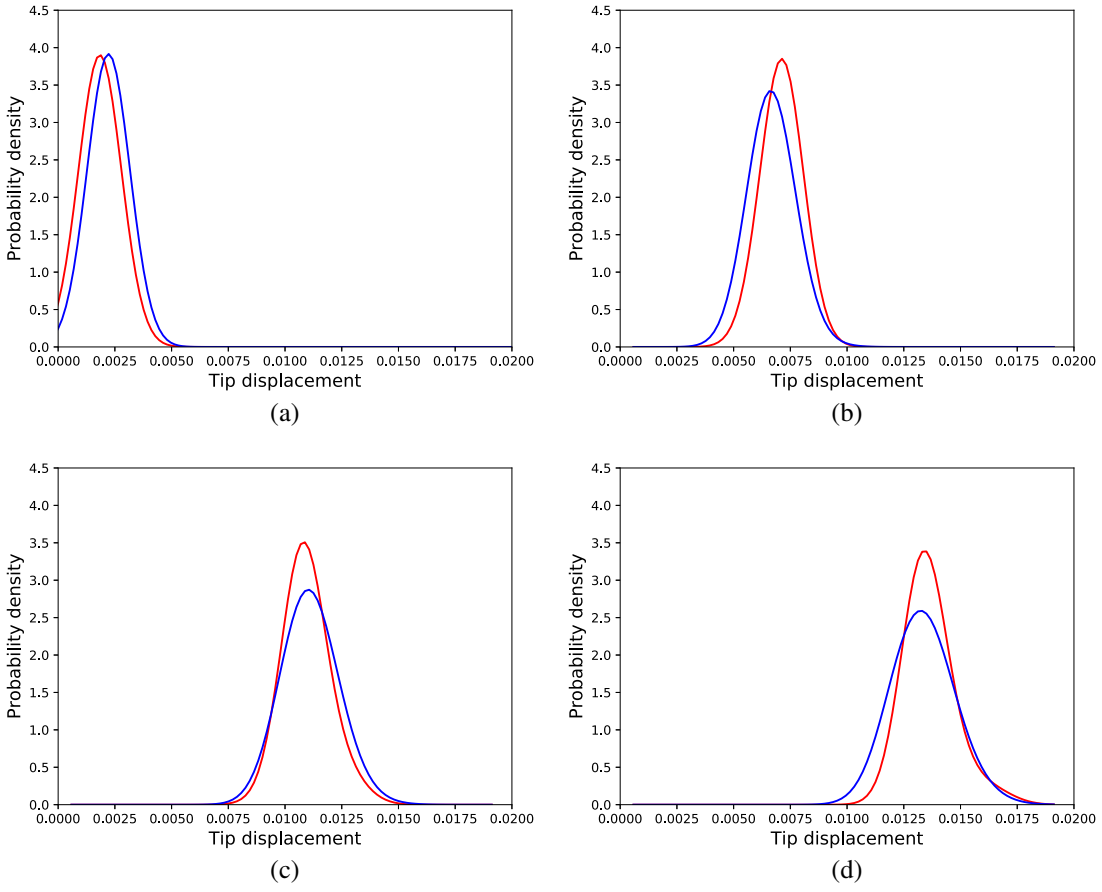

Figure 7. *Distributions of tip displacements corresponding to Monte Carlo samples (blue) and cGAN generated samples (red) regarding the linear problem, different load cases, (a) 30 load units, (b) 90 load units, (c) 150 load units, and (d) 180 load units.*

5.2. Application of cGAN in a nonlinear problem

The next application for testing the potential of the cGAN in approximating the distribution conditioned on the load, is a similar cantilever but with material nonlinearity. More specifically, a softening material is considered; again, it is a simulated structure. The dimensions of the cantilever are the same. As a comparison, in Figure 9, the load curves for the linear and the nonlinear structures are shown.

In this case, random Young's moduli were sampled from a normal distribution with mean value equal to 2×10^9 and standard deviation $0.1 \times 2 \times 10^9$. Each nonlinear simulation was performed using a total load of 400 load units and 40 loadsteps. After finding the solution of the *Newton–Raphson* iterations, the displacements were stored for every iteration; in this way, every nonlinear simulation provided 40 tip displacements, one for every load in the set $\{10, 20, 30 \ldots, 400\}$. For every load case, 500 samples were generated. In Figure 10, samples are shown for different loads in the dataset $\{10, 40, 70 \ldots, 400\}$ which is also considered the training dataset for the cross-validation procedure followed (load units $\{20, 50, 80 \ldots, 380\}$ comprised the validation dataset and load units $\{30, 60, 90 \ldots, 390\}$ the testing dataset).

An identical approach to the linear case was followed except for the sizes considered for the hidden layer of the networks. The sizes tested belonged to the set $\{10, 20, 30, \ldots 3000\}$. Again, following the same cross-validation procedure, the networks that yielded the best results had 110 neurons in their hidden layers. The lowest average KL divergence in the validation dataset was found to be 0.045 and that network

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

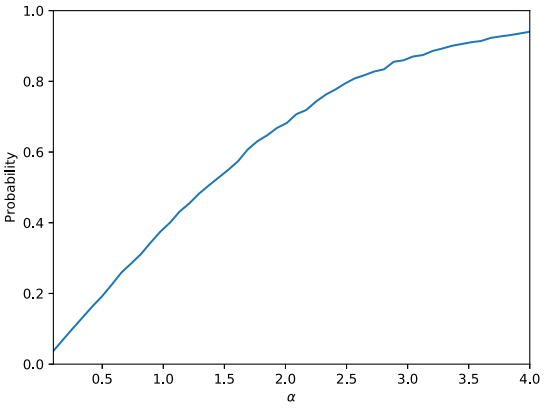

Figure 8. *Probability as defined in Equation (2) as a function of the parameter α for the cGAN model applied on the linear cantilever case study.*

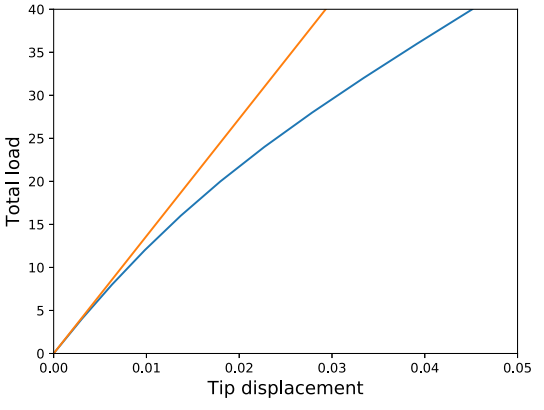

Figure 9. *Load curves for the linear (orange) and the nonlinear (blue) material cantilevers.*

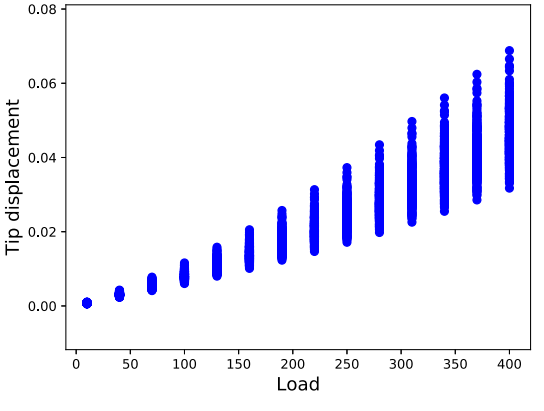

Figure 10. *Tip displacement samples generated by the nonlinear cantilever.*

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

yielded KL divergence equal to 0.050 on the testing dataset. Generated distributions corresponding to codes of the testing dataset, in comparison to the real ones (acquired from the simulated structure) are shown in Figure 11. It is observed that as the spread of the distribution along with the load increases, the algorithm has efficiently learnt to generate samples with greater spread.

To compare the cGAN approach with the SFE method, a SFE model was calibrated (performing an exhaustive search for the optimal parameters) according to the data of the Monte Carlo simulations of the nonlinear cantilever. Although the linear physics formulation does not suffice to accurately explain the behavior of the nonlinear cantilever, the SFE model could yield accurate enough predictions to be used as a mirror. The results for the same loads as in Figure 11 are shown in Figure 12. The results reveal that the SFE model is unable to capture the effect of the nonlinearity on the distributions of the tip displacements; such a result is expected, since nonlinearity is not included in the formulation of the SFE model used. This result is also confirmed by the average KL divergence between the real and predicted distributions in the testing dataset, which was 3.43.

Similarly, the cGAN is tested as an ε-mirror regarding its ability to predict the distribution of the tip displacement of the cantilever. The maximum value of the KL divergence on the available testing datasets now is 0.25. Regarding its use as an α-mirror, the plot showing the probability defined in Equation (2) is shown in Figure 13. Although the KL divergence in this later case study was lower than in the application of the cGAN in the linear case, the α-curve this time is not as good as the one in Figure 8, in the sense that eventually, for larger values of α (e.g., $\alpha = 4.0$), the cGAN applied in the linear case, is able to include in the desired interval a larger portion of the observed data.

6. Hybrid Model Approach

6.1. Definition of the hybrid model

The approaches presented so far were either completely physics-based (SFE model) or data-driven, informed partially by the physics of the problem (a cGAN informed by the value of the load). It became clear that a SFE model without the appropriate physics included (such as nonlinear effects) cannot describe efficiently a phenomenon like the tip displacement of the nonlinear cantilever. In real-life applications, insufficient physics in SFE formulation might mean that the users do not know where the uncertainty is coming from (epistemic uncertainty about the aleatory uncertainty). On the other hand, the cGAN was able to capture the effect of the load on the distribution of the tip displacement both in the linear and the nonlinear case without any knowledge about the source of uncertainty. The efficiency of the algorithm lies in its insensitivity to the linearity of the underlying problem.

Naturally, one would prefer a model that includes further understanding of the underlying physics rather than just using the value of the input in the physical system; that is, the load in the case studies. To define such a coupling, a hybrid approach is followed. The SFE model is used as a first estimator of the target distributions and afterward the cGAN algorithm is applied to correct them. A similar approach for nonlinear modeling has been presented in Rogers et al. (2017) and another about learning such model discrepancies has been developed in Gardner et al. (2020).

In Rogers et al. (2017), two types of such models are defined. The first is the *A*-type models in which the black-box model is exploited to infer the error between the white-box model predictions and the observations. Inference is performed on the error between the white-box model and the real observation, that is,

$$\delta(X) = y(X) - f(X), \tag{7}$$

where y is the observation, f is the white-box model, X is the input variable to the model, and δ is the error sought to be modeled using a black-box/machine learning model. This type of model is not suitable for the current work, since it contradicts the desired definition of a generative model as a mirror. This is because the generative modeling framework requires unknown parameters affecting the structure, while A-type models requires definition of exact errors for specific inputs on the model. To define a training dataset to

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

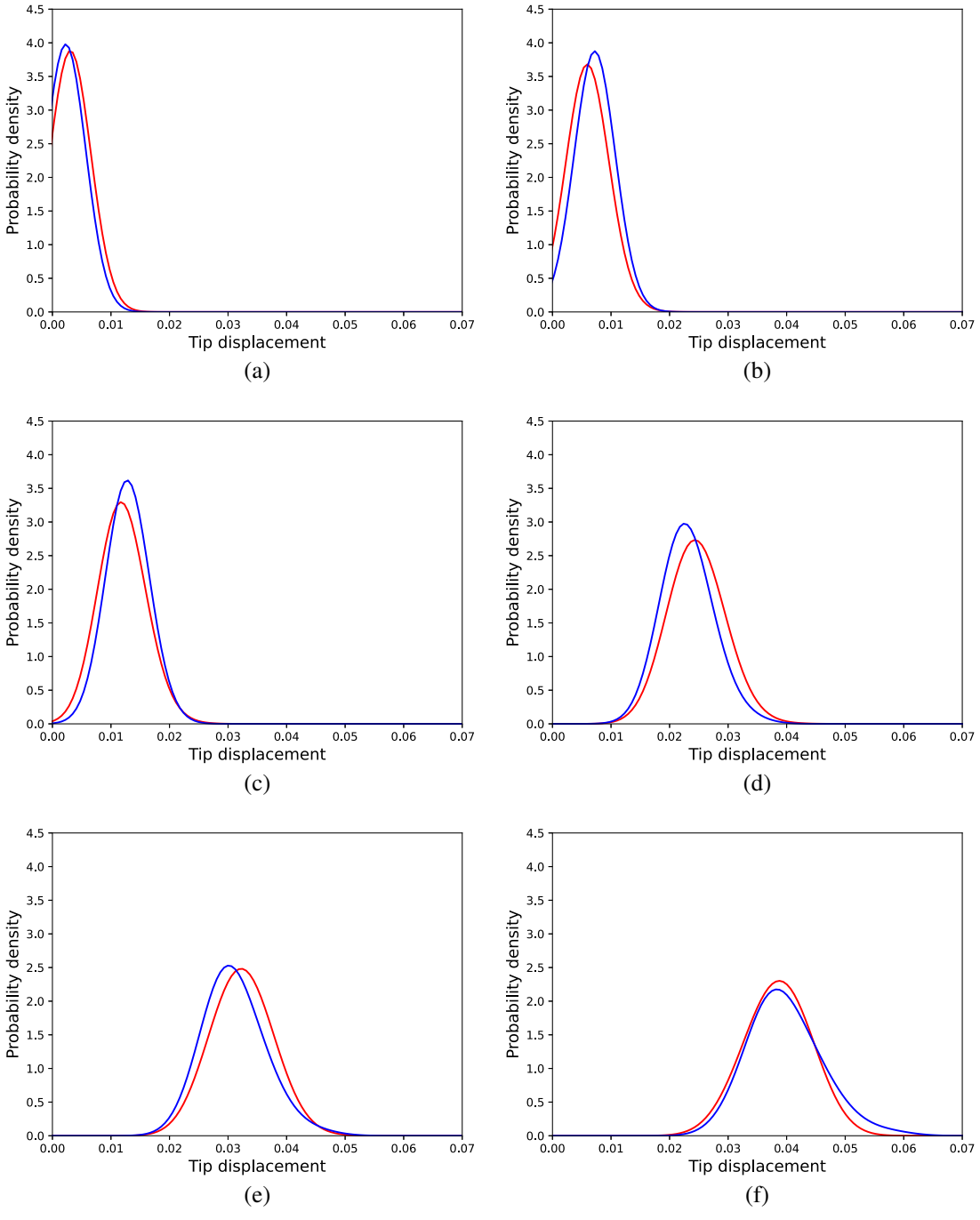

Figure 11. *Distributions of tip displacements corresponding to Monte Carlo samples (blue) and cGAN generated samples (red) regarding the nonlinear problem, for different load cases; (a) 30 load units, (b) 90 load units, (c) 150 load units, (d) 240 load units, (e) 300 load units, and (f) 360 load units.*

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

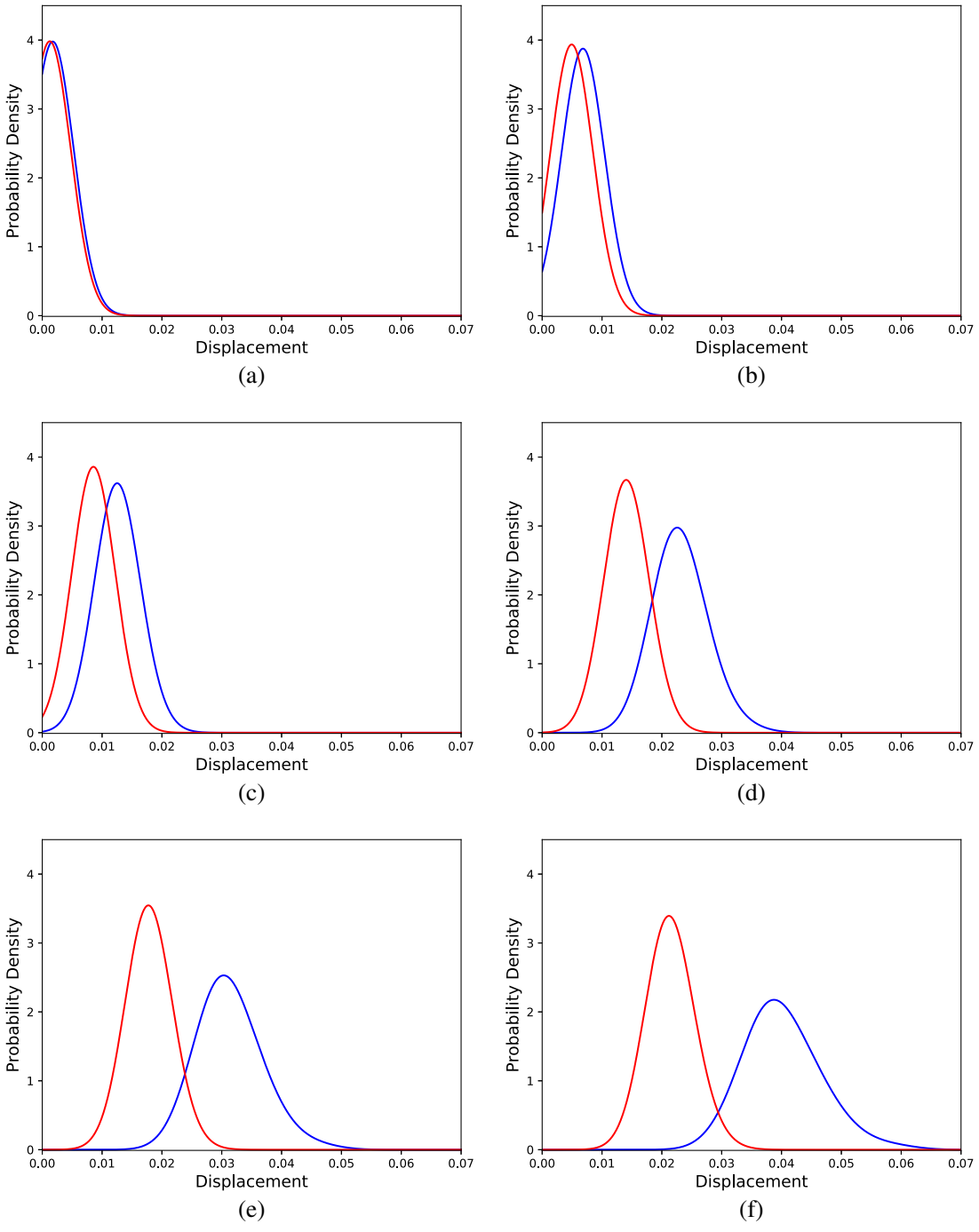

Figure 12. *Distributions of tip displacements corresponding to Monte Carlo samples (blue) and SFEM generated samples (red) regarding the nonlinear problem, for different load cases; (a) 30 load units, (b) 90 load units, (c) 150 load units, (d) 240 load units, (e) 300 load units, and (f) 360 load units.*

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

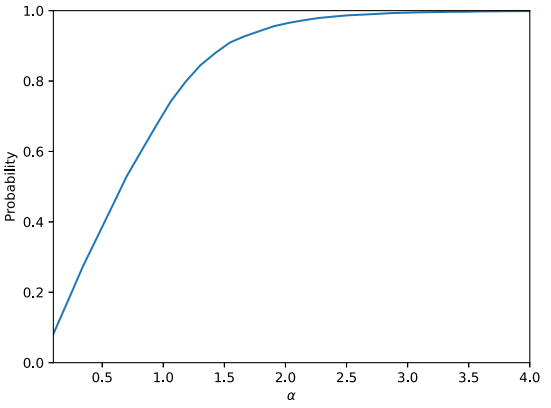

Figure 13. *Probability defined in Equation (2) as a function of the parameter α for the cGAN model applied on the nonlinear cantilever case study.*

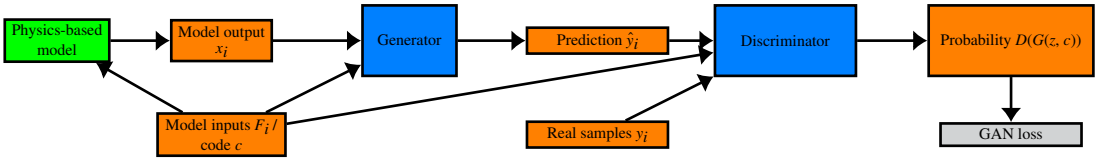

Figure 14. *Layout of the SFE-cGAN hybrid model.*

follow the *A*-type model approach, one would need to specify errors between specific predictions of the white box model and observation. Although, when uncertainty is admitted, that cannot happen, as the output of the white-box model is a function of both a set of controlled and uncontrolled variables; that is, $f(X) = f(\underline{e}_c, \underline{e}_u)$. Since some variables $\underline{e}_u$ are not observable in nature (e.g., the Young's modulus exact field within the volume of a structure), their exact values cannot be used to define such a dataset, which should comprise pairs between model outputs $f(\underline{e}_c, \underline{e}_u)$ and observations $y(\underline{e}_c, \underline{e}_u)$.

As a result, the approach to be followed here is to define a *B*-type model. This latter type of hybrid model is based on the idea that the black-box model performs inference using the output of the white-box model and the controlled variables, that is,

$$y(X) = g(X, f(X)), \tag{8}$$

where g is the black-box model. Specifically for the problem addressed here, the cGAN is called to transform the predicted distributions of the SFE model into the correct ones observed in the data (red and blue distributions, respectively, in Figure 11) being also informed by the value of the load/code. Type B models are applicable in the case of generative models, since to define the real and the predicted distributions, one needs to integrate both sides of Equation (8), which can be done without knowledge of the uncontrolled variables because their effect is explained via the generative model g. On the contrary, integration of both sides of Equation (7) is not feasible without knowledge of the values of the uncontrolled variables for every observation.

The output distributions of the SFE model are used as the latent sampling space of the cGAN. The physics-based model is yielding results based on the "linear part" of the physics of the system and the cGAN is called to learn only the "nonlinear part." This approach is expected to assist the cGAN training by providing latent variables whose distribution is closer to the real ones. In the completely data-driven case presented earlier, the cGAN simply learnt transformations of the same latent distribution as a function of the code. In the hybrid approach, the latent distribution is also a function of the code, aiming in assisting the training procedure. The layout of the combined model is shown in Figure 14.

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

6.2. Application of the hybrid model

The same training procedure was followed. The hidden layer sizes considered were in the set $\{10, 20, 30, \ldots, 1500\}$. The model was trained given codes and data of the training dataset. The model that yielded the lowest KL divergence on the validation dataset had 1,000 neurons in its hidden layer and was selected as the best model. The aforementioned model was tested on the testing dataset and the results for a selection of the input loads are presented in Figure 15. The average KL divergence on the validation dataset and on the testing dataset was 0.049 and 0.044, respectively, meaning that the predicted and the real distributions were quite close; the maximum value of KL divergence was 0.22, which will be the value of ε, if the hybrid model is to be considered an ε-mirror. Regarding its ability to serve as an α-mirror, Figure 16 shows the probability of the observation falling into the interval defined by Equation (2) as a function of α.

6.3. Extrapolation capability study

As already mentioned, the greatest advantage of physics-based models is that they can provide accurate predictions even for data coming from a different domain than the one used to calibrate them. This advantage is based on the definition of the physics of the model that fit sufficiently the physical phenomena, which the model is called to explain. On the other hand, data-based models tend to have no such capabilities. If one has trained a machine learning model on a specific domain of the data, use of the model outside this domain (extrapolation) should be avoided.

When a hybrid model is built, inheritance of the extrapolation capabilities of the physics-based part of the model is desired. In the current application, the output of the hybrid model is controlled by the input code to the model and also by the distribution output of the physics-based SFE model. The SFE model's physics do not suffice to explain the nonlinear behavior of the material of the cantilever. Sometimes, even the physics of the underlying problem might not be clear making the use of a white-box model even more unfavorable. However, a part of the physics of the linear SFE model herein applies also in the nonlinear case. Specifically, that higher values of input load correspond to higher values of tip displacements and higher values of standard deviation (as shown in Figure 5). Therefore, in the current subsection, the potential extrapolation capabilities of the hybrid model are studied and compared with the extrapolation capabilities of the black-box model.

A first step, that can be made to increase the extrapolation capabilities of the black-box model (and therefore also of the hybrid model), is to redefine the normalization scheme of the input and output values of the neural networks used herein. The approach that has been followed so far is to scale the input and the outputs onto the interval $[-1, 1]$. This technique brings all the values within the effective range of the sigmoid functions used (hyperbolic tangent herein) and allows proper gradients to be calculated and the networks to be properly trained using the back-propagation process. However, normalization on the interval $[-1, 1]$ most probably saturates the sigmoids for the extreme values of the inputs and the outputs and minimizes the extrapolation capabilities of the models. To bypass such an issue, in the current section, the scaling is performed having as target the interval $[-0.8, 0.8]$. If one knows that a model might be called to perform outside its training domain, this scaling strategy is a viable option to increase the extrapolation capability of the model.

The initial dataset is split in two to compare the extrapolation potential of the hybrid and the black-box models. The first subset includes data corresponding to load units $\{10, 20 \ldots 310\}$ and the second subset to load units $\{320, 330 \ldots 400\}$. Using the first subset, a hybrid model and a black-box model were trained; that is, the dataset was split in training, validation, and testing datasets and the same procedure as before was followed. After training, the models were also tested on the second subset, which is outside the training domain of the models, to evaluate their extrapolation capabilities.

The best black-box model had in terms of KL divergence on the validation dataset had 1,000 neurons in its hidden layer. It yielded average KL divergence equal to 0.048 on the validation dataset and 0.054 on the testing dataset. A distribution comparison between the predicted and the real distributions for loads of the testing dataset is shown in Figure 17. The results are quite close to the ones observed in the application on

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

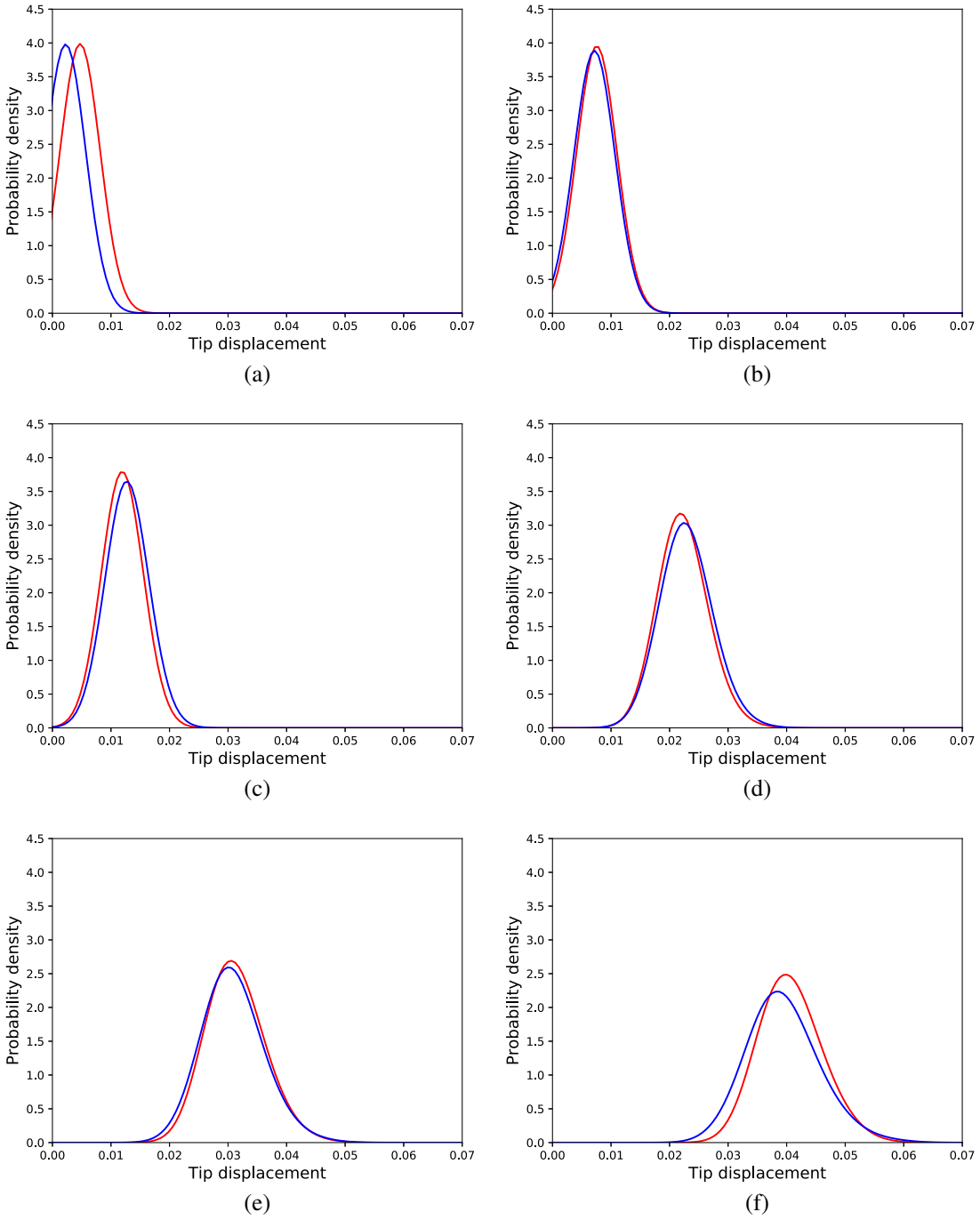

Figure 15. *Distributions of tip displacements corresponding to Monte Carlo samples (blue) and generated samples by the cGAN-SFEM hybrid approach (red) regarding the nonlinear problem, for different load cases; (a) 30 load units, (b) 90 load units, (c) 150 load units, (d) 240 load units, (e) 300 load units, and (f) 360 load units.*

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

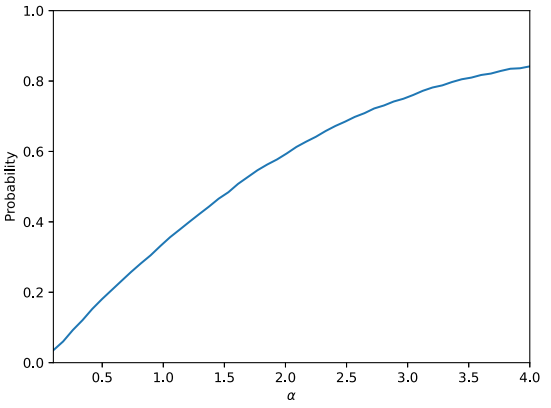

Figure 16. *Probability defined in Equation (2) as a function of the parameter α for the cGAN-SFEM hybrid model applied on the nonlinear cantilever case study.*

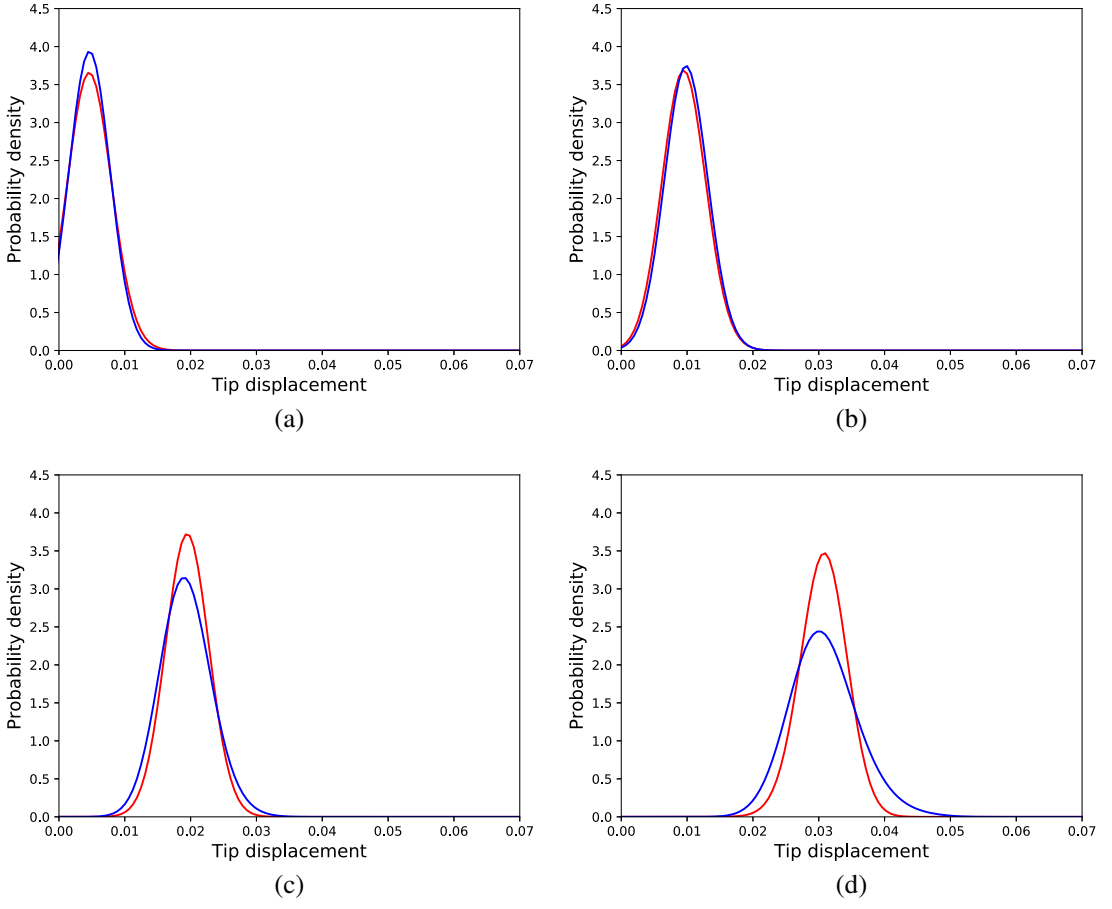

Figure 17. *Distributions of tip displacements corresponding to Monte Carlo samples (blue) and generated samples by the cGAN model (red) regarding the nonlinear problem, trained according to the reduced dataset for different load cases; (a) 60 load units, (b) 120 load units, (c) 210 load units, and (d) 300 load units.*

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

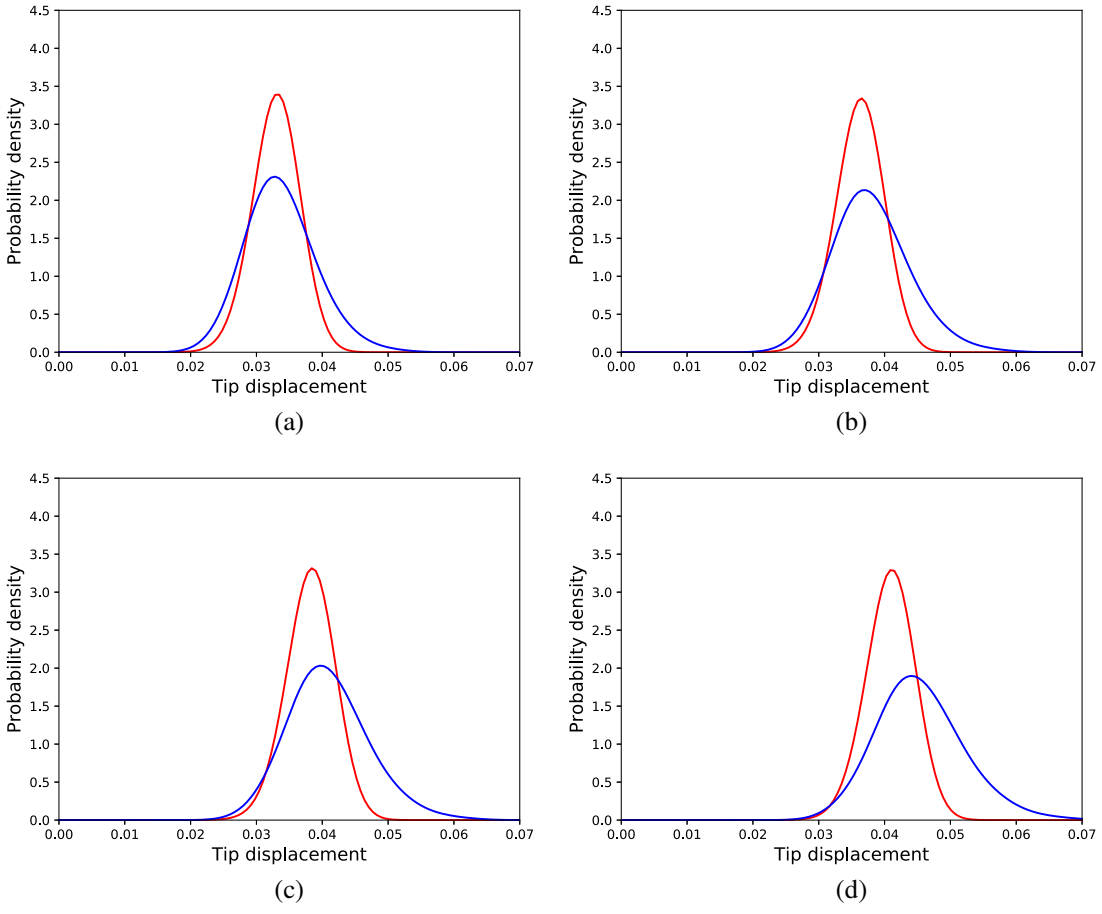

Figure 18. *Distributions of tip displacements corresponding to Monte Carlo samples (blue) and generated samples by the cGAN model (red) regarding the "extrapolation dataset" for the nonlinear problem for different load cases; (a) 320 load units, (b) 350 load units, (c) 370 load units, and (d) 400 load units.*

the complete dataset. To see how the model extrapolates, the same model was tested on the second subset of loads (load units 320, 330…400), which is outside the training domain. Some distributions are shown in Figure 18 and the average KL divergence for the "extrapolation dataset" is equal to 0.77. It is clear that outside the training domain, the model does not perform well and as the load increases, the distribution moves only slightly toward higher values of displacements.

The hybrid model is tested in exactly the same way. The model with the minimum KL divergence on the validation dataset had 1,500 neurons in its hidden layer. This model yielded KL divergence equal to 0.034 on the validation dataset and equal to 0.038 on the testing dataset. Samples of the predicted distributions in comparison with the real ones are shown in Figure 19. Consequently, the same model was tested on the "extrapolation dataset." Similar distribution comparison figures are shown in Figure 20. It is clear that the performance this time is better than in the case of the black-box model and this is confirmed by the average KL divergence observed on the second dataset, which is equal to 0.288. One can see that the part of the physics which is correctly incorporated in the physics-based model (higher load values correspond to higher displacements) has affected positively the performance of the hybrid model on data outside its training domain.

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

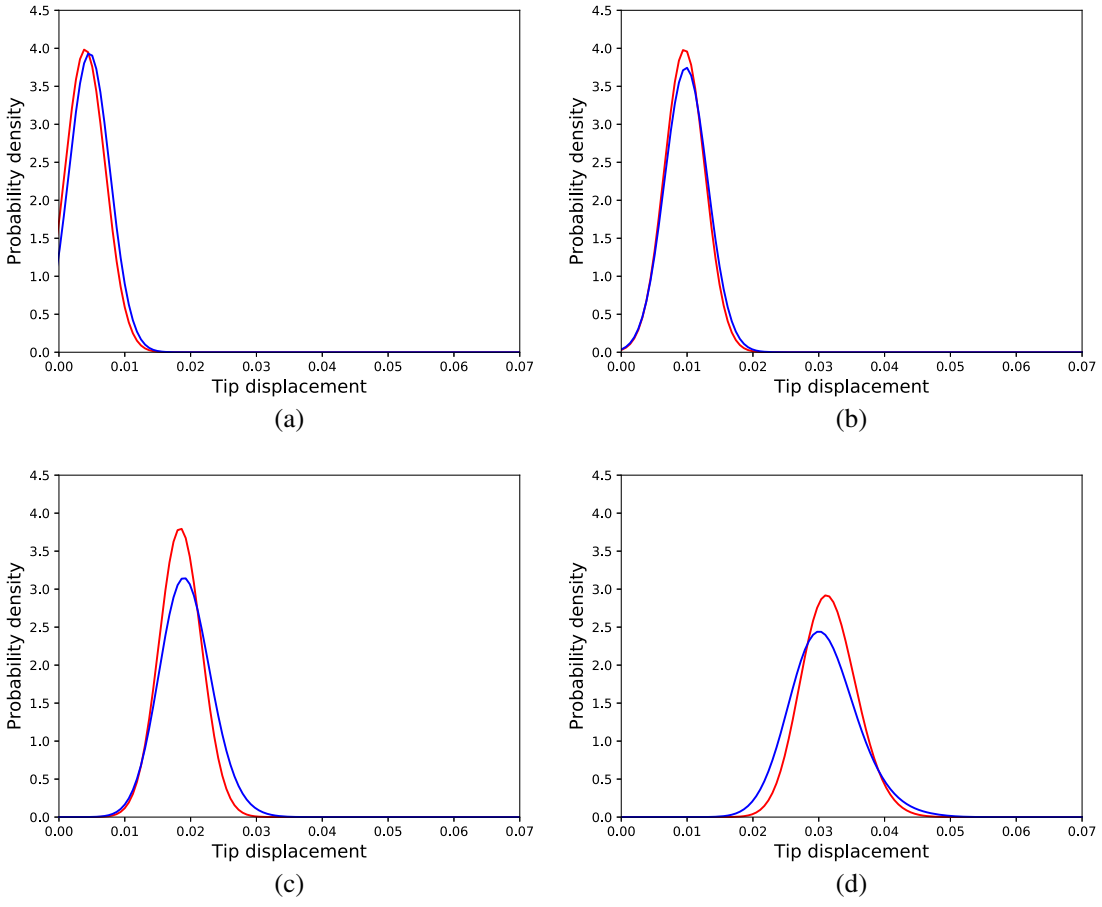

Figure 19. *Distributions of tip displacements corresponding to Monte Carlo samples (blue) and generated samples by the hybrid model (red) regarding the nonlinear problem, trained according to the reduced dataset for different load cases; (a) 60 load units, (b) 120 load units, (c) 210 load units, and (d) 300 load units.*

7. Discussion and Conclusions

In the current work, models were developed that serve as mirrors of a specific structure. The models were chosen to be generative models in an attempt to take into account various uncertainties that might be present during the modeling procedure. A physics-based method, a data-driven method, and a hybrid approach were presented along with their results and performance according to the chosen metric (KL-divergence).

SFEM is the physics-based method, which, calibrated according to acquired data from a structure, was tested as a potential mirror model of a beam structure. As for every physics-based method, if the model's physics formulation fits exactly the physics of the problem, the model is able to outperform any other method and have almost perfect accuracy in predicting the behavior of the structure it describes. This situation is also the case for the SFE model in the concept described in this work. The SFE model was calibrated using an exhaustive search (within some range of values for each parameter) in the parameter space according to acquired data and the generated distributions almost perfectly fit the ones corresponding to the measured distributions.

Even though such a model outperforms every other method, if there is epistemic uncertainty, that is, the model does not describe fully the physics or one does not know which parameter exactly is stochastic or

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

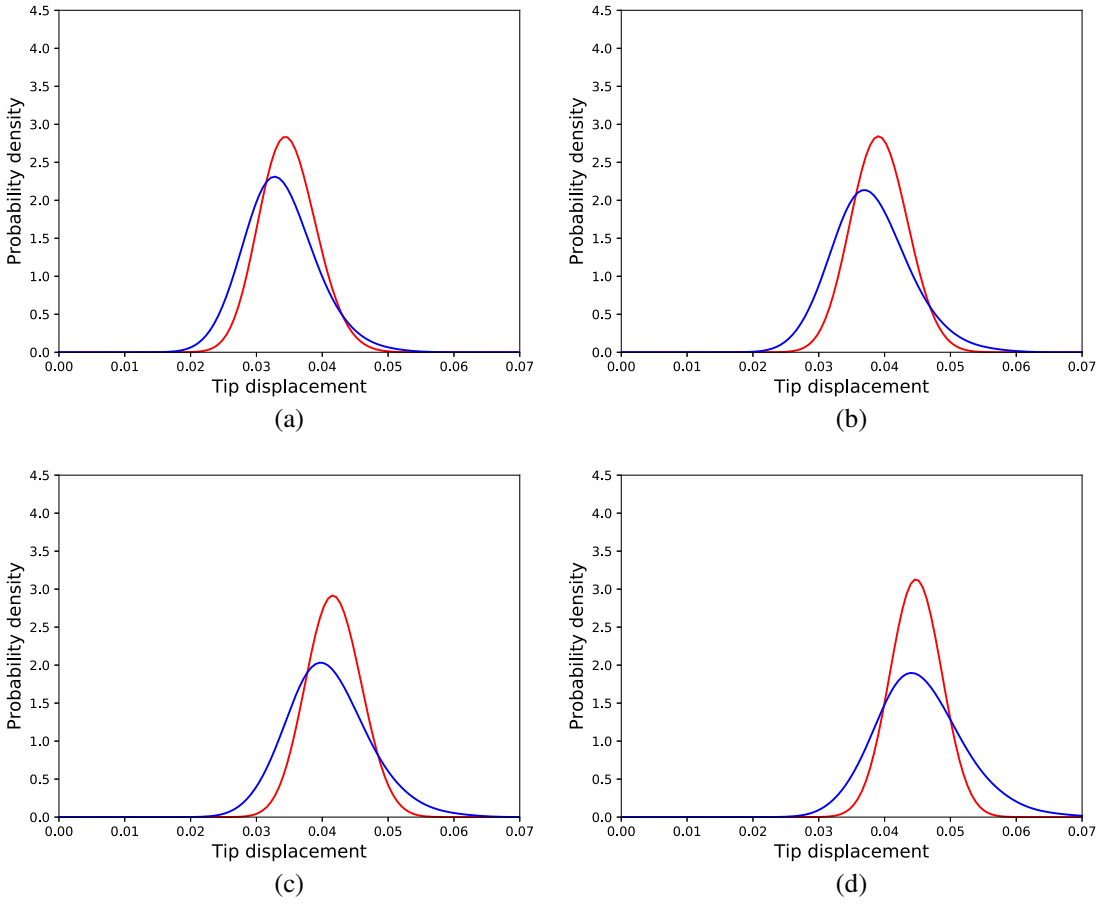

Figure 20. *Distributions of tip displacements corresponding to Monte Carlo samples (blue) and generated samples by the hybrid model (red) regarding the "extrapolation dataset" for the nonlinear problem for different load cases; (a) 320 load units, (b) 350 load units, (c) 370 load units, and (d) 400 load units.*

the form of the stochasticity, it would not perform as well as in the previous case. In such cases, a data-driven method such as the one described herein should be employed. The nonlinear problem described is one such case.

Machine learning approaches could prove even more useful in cases where a physical phenomenon cannot be modeled by finite elements or any other such physics-based method. Situations like this are when one has no knowledge about how an environmental parameter affects the structure. For example, temperature and stiffness reduction (or increase) are usually assumed to have a linear relationship. If that does not stand, a finite element model, in which stiffness is reduced or increased linearly according to temperature, would not suffice regarding generation of accurate predictions.

Within the current framework, a cGAN was considered as a mirror of a simulated structure with material nonlinearities. The algorithm was able to perform well enough to be an ε-mirror of the structure for values of ε calculated by the data. The algorithm fits the framework of controlled and uncontrolled environmental variables and is able to incorporate within its formulation any uncertainties that might exist. It can be also considered an α-mirror, given different probabilities, for the observation to fall into the interval, as a function of α. The latter type of mirror models may serve as a conservative aspect of the digital mirror model, since it provides an interval, within which any observation should fall into with

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

probability $P(\alpha)$. The α-mirror approach is irrelevant to the shape of the distribution and is only informed by the mean value and the standard deviation of the samples generated by the model.

Finally, a hybrid approach of the SFEM and cGAN is presented. The approach is based on using the cGAN to correct the predicted-by-the-SFE-model probability density functions. The hybrid model is able to perform slightly better than the cGAN in the nonlinear case. The advantages of the method are considered the information imposed into the model by the physics-based SFE method and the versatility of the cGAN algorithm in being able to perform regardless the nonlinearity of the problem. It appears to be a viable strategy for such applications, since it is further based on the physics of the problem, and also provides a framework of correcting the predictions of a physics-based generative model when unknown and even immeasurable parameters affect the result. Moreover, the presented results reveal that such models have greater extrapolation capabilities. Even if the physics-based model used does not completely incorporate the underlying physics of the problem, the model can still be informed by the valid parts of the captured physics and, that way, outperform black-box models in terms of accuracy in situations outside the training domain of the model.

Acknowledgments. K.W. would like to thank the UK Engineering and Physical Sciences Research Council (EPSRC) for an Established Career Fellowship (EP/R003645/1). D.J.W. would like to acknowledge the support of EPSRC grant EP/R006768/1.

Funding Statement. This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 764547.

Competing Interests. The authors declare no competing interests exist.

Data Availability Statement. The data used in the applications can be found in https://drive.google.com/drive/folders/1Ykv_lC1SkRbWtWFoSE_PWK2zd1CgyZqx?usp=sharing and the code to recreate the results of the methods described in the current work can be found in the GitHub repository: https://github.com/GiorgTsial/cGANS_DT.

Author Contributions. Conceptualization, G.T., D.J.W., N.D., K.W.; Methodology, G.T., D.J.W., N.D., K.W.; Investigation, G.T.; Validation, G.T.; Visualization, G.T.; Software, G.T.; Writing—original draft, G.T.; Writing—review and editing, D.J.W., N.D., K.W.; Funding acquisition, D.J.W., N.D., K.W.; Project administration, D.J.W., N.D., K.W.; Supervision, D.J.W., K.W.

Supplementary Materials. To view supplementary material for this article, please visit http://dx.doi.org/10.1017/dce.2021.13.

References

Augustin F and Rentrop P (2012) Stochastic Galerkin techniques for random ordinary differential equations. *Numerische Mathematik 122*, 399–419.

Bathe K-J (2006) *Finite Element Procedures*. Hoboken, NJ: Prentice Hall.

Bishop C (1995) *Neural Networks for Pattern Recognition*. New York: Oxford University Press.

Bishop C (2006) *Pattern Recognition and Machine Learning*. Berlin: Springer-Verlag.

Booyse W, **Wilke DN and Heyns S** (2020) Deep digital twins for detection, diagnostics and prognostics. *Mechanical Systems and Signal Processing 140*, 106612.

Cross E, **Gibson S**, **Jones M**, **Zhang DPS and Rogers T** (2022) *Structural Health Monitoring Based on Data Science Techniques, Chapter Physics-Informed Machine Learning for Structural Health Monitoring*. Cham: Springer (in press).

Cross E and Rogers T (2021) *Physics-derived covariance functions for machine learning in structural dynamics. 19th IFAC Symposium on System Identification (SYSID): Learning Models for Decision and Control (in press).*

Dembski F, **Wössner U**, **Letzgus M**, **Ruddat M and Yamu C** (2020) Urban digital twins for smart cities and citizens: The case study of Herrenberg, Germany. *Sustainability 12*(6), 2307.

Duffin C, **Cripps E**, **Stemler T and Girolami M** (2021) Statistical finite elements for misspecified models. *Proceedings of the National Academy of Sciences of the USA 118*, e2015006118.

Epanechnikov VA (1969) Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications 14*(1), 153–158.

Frid-Adar M, **Diamant I**, **Klang E**, **Amitai M**, **Goldberger J and Greenspan H** (2018) GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification. *Neurocomputing 321*, 321–331.

Fuller A, **Fan Z**, **Day C and Barlow C** (2020) Digital twin: Enabling technologies, challenges and open research. *IEEE Access 8*, 108952–108971.

Gardner P, **Rogers T**, **Lord C and Barthorpe R** (2020) Learning model discrepancy: A Gaussian process and sampling-based approach. *Mechanical Systems and Signal Processing 152*, 107381.

Ghanem R and Spanos P (2003) *Stochastic Finite Elements: A Spectral Approach*. North Chelmsford, MA: Courier Corporation.

Girolami M, **Febrianto E**, **Yin G and Cirak F** (2021) The statistical finite element method (statFEM) for coherent synthesis of observation data and model predictions. *Computer Methods in Applied Mechanics and Engineering 375*, 113533.

Goodfellow I, **Pouget-Abadie J**, **Mirza M**, **Xu B**, **Warde-Farley D**, **Ozair S**, **Courville A and Bengio Y** (2014) Generative adversarial nets. *Advances in Neural Information Processing Systems 27*, 2672–2680.

Jones D, **Snider C**, **Nassehi A**, **Yon J and Hicks B** (2020) Characterising the digital twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology 29*, 36–52.

Kingma D and Welling M (2014) Auto-encoding variational Bayes. *Proceedings of the 2nd International Conference on Learning Representations (ICLR).*

Krizhevsky A, **Sutskever I and Hinton G** (2012) Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems 25*, 1097–1105.

Kullback S (1997) *Information Theory and Statistics*. North Chelmsford, MA: Courier Corporation.

Loeve M (1977) Elementary probability theory. In *Probability Theory I*. Berlin: Springer, pp. 1–52.

Macchi M, **Roda I**, **Negri E and Fumagalli L** (2018) Exploring the role of digital twin for asset lifecycle management. *IFAC-PapersOnLine 51*(11), 790–795.

Mirza M and Osindero S (2014) *Conditional generative adversarial nets. arXiv Preprint arXiv:1411.1784.*

Murphy KP (2012) *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: The MIT Press.

Mylonas C, **Abdallah I and Chatzis E** (2020) Deep unsupervised learning for condition monitoring and prediction of high-dimensional data with application on windfarm SCADA data. *Model Validation and Uncertainty Quantification 3*, 189–196.

Papoulis A and Pillai S (2002) *Probability, Random Variables, and Stochastic Processes*. New York: Tata McGraw-Hill Education.

Pitchforth D, **Rogers T**, **Tygesen U and Cross E** (2021) Grey-box models for wave loading prediction. *Mechanical Systems and Signal Processing 159*, 107741.

Rasmussen C and Williams C (2005) *Gaussian Processes for Machine Learning*. Cambridge, MA: The MIT Press.

Rogers T, **Gardner P**, **Dervilis N**, **Worden K**, **Maguire A**, **Papatheou E and Cross E** (2020) Probabilistic modelling of wind turbine power curves with application of heteroscedastic Gaussian process regression. *Renewable Energy 148*, 1124–1136.

Rogers T, **Holmes G**, **Cross E and Worden K** (2017) On a grey box modelling framework for nonlinear system identification. In *Special Topics in Structural Dynamics*, Vol. *6*. Cham: Springer, pp. 167–178.

Rosen R, **von Wichert G**, **Lo G and Bettenhausen KD** (2015) About the importance of autonomy and digital twins for the future of manufacturing. *IFAC-PapersOnLine 48*(3), 567–572.

Schluse M, **Priggemeyer M**, **Atorf L and Rossmann J** (2018) Experimentable digital twins—Streamlining simulation-based systems engineering for industry 4.0. *IEEE Transactions on Industrial Informatics 14*(4), 1722–1731.

Silverman B (1981) Using kernel density estimates to investigate multimodality. *Journal of the Royal Statistical Society: Series B (Methodological) 43*(1), 97–99.

Smith RC (2013) *Uncertainty Quantification: Theory, Implementation, and Applications*, Vol. *12*. Philadelphia, PA: SIAM.

Specht D (1991) A general regression neural network. *IEEE Transactions on Neural Networks 2*(6), 568–576.

Spiridonakos M and Chatzi E (2015) Metamodeling of dynamic nonlinear structural systems through polynomial chaos NARX models. *Computers & Structures 157*, 99–113.

Spiridonakos M, **Chatzi E and Sudret B** (2016) Polynomial chaos expansion models for the monitoring of structures under operational variability. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering 2*(3), B4016003.

Stefanou G (2009) The stochastic finite element method: Past, present and future. *Computer Methods in Applied Mechanics and Engineering 198*(9–12), 1031–1051.

Sudret B and Der Kiureghian A (2000) *Stochastic Finite Element Methods and Reliability: A State-of-the-Art Report*. Berkeley, CA: Department of Civil and Environmental Engineering, University of California.

Tarassenko L (1998) *Guide to Neural Computing Applications*. Amsterdam: Elsevier.

Uhlemann THJ, **Schock C**, **Lehmann C**, **Freiberger S and Steinhilper R** (2017) The digital twin: Demonstrating the potential of real time data acquisition in production systems. *Procedia Manufacturing 9*, 113–120.

Wagg D, **Worden K**, **Barthorpe R and Gardner P** (2020) Digital twins: State-of-the-art and future directions for modeling and simulation in engineering dynamics applications. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems Part B Mechanical Engineering 6*(3), 030901.

Worden K, **Cross E**, **Barthorpe R**, **Wagg D and Gardner P** (2020) On digital twins, mirrors, and virtualizations: Frameworks for model verification and validation. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems Part B Mechanical Engineering 6*(3), 030902.

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

A. Appendix

A.1. Stochastic finite elements

Finite element method (FEM) models (Bathe, 2006) have been a very powerful and useful tool to numerically solve differential equations which describe mechanical systems. FEM transforms a continuous problem and a continuous differential equation into a discrete system of equations. Solutions are calculated only for a discrete number of points. Solutions for intermediate points are calculated using interpolation functions called shape functions. The continuous static differential equation most commonly used in FE models is given by:

$$\int_V \sigma \varepsilon dV = \int_V f^V dV + \int_{S_e} f^S dS, \tag{A.1}$$

where the LHS is the internal potential energy of a body, where σ is stress, ε is strain, and V is the volume of the body of interest, the RHS is the potential work of the forces applied on the body, which is the integral over the volume of all the volume forces (f^V) plus the integral over the surface of the surface forces (f^S). Using a finite element formulation and minimizing the total potential energy or the difference between the two sides of equation, one gets,

$$[K]\{U\} = \{F\}, \tag{A.2}$$

where $[K]$ is the stiffness matrix of the structure for a specific meshing scheme applied, $\{U\}$ is the displacement vector of the nodal displacements, and $\{F\}$ is the equivalent nodal force vector to the total applied forces on the body.

For dynamic problems, inertia forces are introduced into the RHS of Equation (A.1) resulting in,

$$\int_V \sigma \varepsilon dV = \int_V f^V dV + \int_{S_e} f^S dS - \int_V \rho \ddot{u} dV - \int_V c \dot{u} dV, \tag{A.3}$$

where ρ refers to the mass density function of the body, $\ddot{u}$ is the acceleration, c is the damping parameter, and $\dot{u}$ is the velocity at every point. This equation holds for every time instant t of the simulation. Once again, following the FEM formulation and defining a discretization of the body, the system of equations are,

$$[M]\{\ddot{U}\}(t) + [C]\{\dot{U}\}(t) + [K]\{U\}(t) = \{F\}(t), \tag{A.4}$$

where $[M]$, $[C]$, and $[K]$ are the mass, damping, and stiffness matrices and $\{\ddot{U}\}$, $\{\dot{U}\}$, and $\{U\}$ are the nodal accelerations, velocities, and displacement vectors, respectively.

The matrices in Equations (A.2) and (A.4) often are calculated assuming deterministic structural parameters, for example, Young's modulus (E), Poisson's ratio (ν), and mass density (ρ). However, these parameters are quite often *not* deterministic. Especially in composites, such parameters are almost certainly random. Young's modulus might vary within the volume of the body one tries to analyse using FEM. These variations are not just discrete variables, almost certainly they are *stochastic processes* (Papoulis and Pillai, 2002). A stochastic process has a correlation function that defines how values over some distance (spatial or temporal) are correlated. Furthermore, every point has a mean value and a variance defined by functions $\mu(x)$ and $\sigma^2(x)$, respectively. If the two functions are constant everywhere over the space where the process is defined, then it is called a *stationary* process.

An example of a deterministic consideration of Young's modulus and a stochastic Young's modulus for a cantilever beam problem would look like the functions shown in Figure A1. The black line represents how conventional FEM is applied assuming a constant and deterministic value for structural parameters, while the red and blue lines show samples drawn from a stochastic process. A way to address problems like this, in general, would be to sample from the stochastic process and follow a Monte Carlo scheme. This approach would require solving a large number of deterministic FEM problems, performing a sensitivity analysis, and

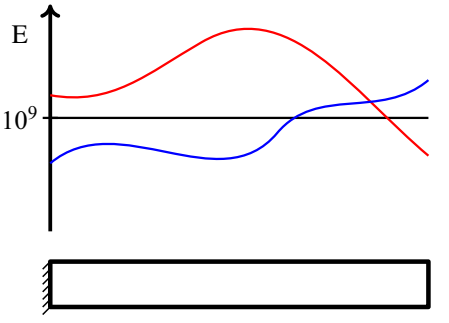

Figure A1. *Cantilever beam with constant Young's modulus (black line) or spatially varying (red and blue lines).*

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

post-processing the results in order to infer the statistics of the quantities of interest. In this case, the quantity of interest could be the displacement of the tip and a probability distribution would be defined over the potential values of this displacement.

The SFEM already mentioned, is a quite popular means of propagating uncertainty from material properties and randomness in the excitation forces into the response characteristics of a structure. In contrast to a Monte Carlo approach, SFEM infers the distribution of interest as a function of a set of discrete normally distributed variables. In order to explain the SFEM formulation of a problem, first a method to decompose the random field is needed. The expansion method used herein is the Karhunen–Loève expansion (Loeve, 1977; Sudret and Der Kiureghian, 2000).

A.2. Karhunen–Loève (KL) expansion

The KL expansion is based on the spectral (i.e., eigenvalue) decomposition of the autocovariance function of the given random field. Given a random field $H(x)$ and its autocovariance function $C_{HH}(x,x')$, any realization of the field $H(x)$ is expanded over a basis of deterministic functions, defined by the eigenvalue problem (Sudret and Der Kiureghian, 2000),

$$\int_{\Omega} C_{HH}(x,x')\phi(x')d\Omega_{x'} = \lambda_i\phi_i(x), \tag{A.5}$$

where the kernel $C_{HH}(x,x')$ is a kernel autocovariance function, bounded, symmetric, and positive definite, and $\Omega_{x'}$ is the total space of x'. The set of eigenvalues λ_i and eigenfunctions/eigenvectors $\{\phi_i\}$ form a complete basis to express every realization of the field as,

$$H(x,\theta) = \mu(x) + \sum_{i=1}^{\infty}\sqrt{\lambda_i}\xi_i(\theta)\phi_i(x), \tag{A.6}$$

where $\xi_i(\theta)$ are the coordinates of the realization which are independent random variables and θ is the random event.

In practice, if one wishes to define an expansion of a random field and either generate random realizations or use it, as it will subsequently be used for the purposes of stochastic FEM, a truncation is performed at mth order yielding,

$$H(x,\theta) = \mu(x) + \sum_{i=1}^{m}\sqrt{\lambda_i}\xi_i(\theta)\phi_i(x), \tag{A.7}$$

where it is assumed that the eigenvalues λ_i are sorted in ascending order.

A.3. Solution of static SFEM problems

After defining the stochastic field in the finite element formulation of a problem, this field is expressed via the KL expansion. This approach leads to the stiffness matrix of Equation (A.2) appearing as a summation over stiffness matrices constructed according to the eigenfunctions of Equation (A.7); more specifically,

$$K(\theta) = K_0 + \sum_{i=1}^{m}\xi_i(\theta)K_i, \tag{A.8}$$

where the $K_i s$ are deterministic matrices that are calculated using the eigenfunctions ϕ_i from Equation (A.7). Realizations from the set ξ_i can be used in order to generate realizations for the stiffness matrix $K(\theta)$, in case a Monte Carlo simulation is to be followed.

Now, substituting Equation (A.8) into Equation (A.2) yields (assuming a deterministic load),

$$\left[K_0 + \sum_{i=1}^{m}K_i\xi_i(\theta)\right]U(\theta) = \sum_{i=0}^{m}[K_i\xi_i(\theta)]U(\theta) = F. \tag{A.9}$$

In order to move further from this point, a *polynomial chaos expansion* (PCE) is used (Ghanem and Spanos, 2003). PCE has been a very efficient and widely used method in engineering applications, for meta-modeling, structural health monitoring, and so forth (Spiridonakos and Chatzi, 2015; Spiridonakos et al., 2016). Here, it is used to decompose the field of displacements $U(\theta)$ yielding,

$$U(\theta) = \sum_{j=0}^{\infty}U_j\Psi_j(\theta), \tag{A.10}$$

where the $\Psi_j(\theta)$, for $j = 0,1,2\ldots$ are polynomials defined in $\xi_i(\theta)$ which satisfy,

$$\Psi_0 \equiv 1, \tag{A.11a}$$

$$\mathbb{E}_{\xi(\theta)}\left[\Psi_j\right] = \mathbb{E}_{\theta}\left[\Psi_j\right] = 0 \quad j > 0, \tag{A.11b}$$

https://doi.org/10.1017/dce.2021.13 Published online by Cambridge University Press

$$\mathbb{E}_{\xi(\theta)}\left[\Psi_j(\theta)\Psi_k(\theta)\right] = \mathbb{E}_{\theta}\left[\Psi_j(\theta)\Psi_k(\theta)\right] = 0 \quad j \neq k, \tag{A.11c}$$

that is, the terms $\Psi_j(\theta)$ are orthogonal in expectation ($\mathbb{E}[]$).

By truncating at P terms and substituting into Equation (A.9), one obtains,

$$\sum_{i=0}^{m} K_i \xi_i(\theta) \sum_{j=0}^{P-1} U_j \Psi_j(\theta) = F. \tag{A.12}$$

The solution to this equation can be found by minimizing the error,

$$\varepsilon_{m,P} = \sum_{i=0}^{m} K_i \xi_i(\theta) \sum_{j=0}^{P-1} U_j \Psi_j(\theta) - F. \tag{A.13}$$

The best approximation of the exact solution $U(\theta)$ in the space H_P spanned by the $\{\Psi_k\}_{k=0}^{P-1}$ is obtained by minimizing this residual in a mean-square sense. In a Hilbert space this is equivalent to requiring that the residual be orthogonal to H_P, yielding,

$$\mathbb{E}_{\theta}[\varepsilon_{m,P}\Psi_k] = 0 \quad k = 0, \ldots P-1. \tag{A.14}$$

Substituting Equation (A.14) in Equation (A.12), one finds,

$$\mathbb{E}_{\theta}\left[\sum_{i=0}^{m}\sum_{j=0}^{P-1} K_i \xi_i \Psi_j(\theta)\Psi_k(\theta) U_j\right] = \mathbb{E}_{\theta}[\Psi_k(\theta)F]. \tag{A.15}$$

Introducing the following notation,

$$c_{ijk} = \mathbb{E}_{\theta}\left[\xi_i \Psi_j \Psi_k\right], \tag{A.16}$$

$$F_k = \mathbb{E}_{\theta}[\Psi_k F], \tag{A.17}$$

and,

$$K_{jk} = \sum_{i=0}^{M} c_{ijk} K_i, \tag{A.18}$$

the stochastic FEM equation finally becomes,

$$\sum_{j=0}^{P-1} K_{jk} U_j = F_k \quad k = 0, 1 \ldots P-1. \tag{A.19}$$

In this equation, every U_j is an N-dimensional vector, where N is the number of degrees of freedom in the system. In total, the P equations from above can be written as,

$$\begin{bmatrix} K_{00} & \ldots & K_{0,P-1} \\ K_{10} & \ldots & K_{1,P-1} \\ \vdots & & \vdots \\ K_{P-1,0} & \ldots & K_{P-1,P-1} \end{bmatrix} \begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_{P-1} \end{bmatrix} = \begin{bmatrix} F_0 \\ F_1 \\ \vdots \\ F_{P-1} \end{bmatrix}. \tag{A.20}$$

Having solved this system, for U_j, samples $U(\theta)$ can be generated by sampling $\xi_i(\theta)$ values and using Equation (A.10). Thus, samples of the distribution of all displacements are generated. Solving this system is equivalent to solving the problem for every potential value of the random parameters. The augmented matrices in Equation (A.20) are of dimension $NP \times NP$, where N are the degrees of freedom of the deterministic problem and P the order of the PCE. Solving such a system instead of a deterministic one is much more computationally intense, that is, $\mathcal{O}\left(N^3P^3\right)$. However, it might not be as computationally inefficient as a sufficient number of Monte Carlo simulations. Solving the system yields samples and therefore distributions, establishing SFE models as generative models.

Cite this article: Tsialiamanis G, Wagg DJ, Dervilis N and Worden K (2021). On generative models as the basis for digital twins. *Data-Centric Engineering*, 2: e11. doi:10.1017/dce.2021.13