Archive of Applied Mechanics
https://doi.org/10.1007/s00419-024-02601-w

ORIGINAL

Markus Franke · Marcus Wagner

Transient surrogate modeling of modally reduced structures with discontinuous loads and damping

Received: 17 July 2023 / Accepted: 12 April 2024

Abstract In this work, a surrogate model for structural, transient and discontinuously excited finite element method simulations is developed. This allows to reduce the computational effort of repeated calculations of identical models under different load cases. The architecture of the surrogate combines fully connected neural network layers with long short-term memory layers. For the reproduction of different damping ratios, a categorical variable is added to the continuous input data. Based on a recursive flow of the predicted data back to the input layer, long-term dependencies do not vanish due to short-input sequences. The system dimension is reduced by applying the model-order reduction technique for modal decomposition. The high accuracy of the surrogate and the reduction of computational costs are shown on an academic example of a cantilever beam and a real-world example of a robot. The advantages of our approach are illustrated in comparison with state-of-the-art surrogates for transient finite element analysis. By using the surrogate proposed in this study, oscillations due to discontinuous excitation of mechanical structures can be reproduced. For this purpose, only short-input sequences are necessary since the excitation of the oscillations does not have to be part of the input sequence during the whole duration of the oscillations. Due to the categorical variable for the damping ratio, the surrogate can account for the influence of different damping in parameter studies.

Keywords Finite element method · Long short-term memory · Surrogate modeling · Modal decomposition · Discontinuous loads

1 Introduction

The finite element method (FEM) is a powerful numerical technique used to analyze and optimize mechanical structures under transient and discontinuous loading for maximum performance and efficiency. However, transient FEM calculations can be computationally expensive and time-consuming, particularly for complex structures with a large number of degrees of freedom. This may limit the optimization of various designs in a reasonable amount of time.

To address this challenge, a wide spectrum of model-order reduction techniques were developed, see Lu et al. [1]. With the developments in the field of artificial intelligence (AI), researchers now have additional possibilities to speed up calculations. To address this issue from a hardware point of view, Tong and Schiavazzi [2] improve parallelization of the solving process and intelligent communication between the processors with neural networks. Thus, computational costs can be saved. This approach is especially useful for problems,

M. Franke (✉) · M. Wagner
Laboratory Finite-Element-Method, Faculty of Mechanical Engineering, OTH Regensburg, Galgenbergstraße 30, 93053 Regensburg, Bavaria, Germany
E-mail: markus.franke@oth-regensburg.de

M. Wagner
E-mail: marcus.wagner@oth-regensburg.de

Published online: 10 May 2024

which have to be calculated only a few times. If calculations have to be carried out often, building a surrogate model helps to cut computation times by a fraction. In recent years, the use of surrogate models has become increasingly common in engineering design, as they provide a valuable tool for accelerating the optimization process. By reducing the computational burden of FEM calculations, surrogate models allow engineers to explore more design options and to improve the performance and efficiency of mechanical structures.

Surrogates of static structural analysis have been developed for replacing FE simulations and are already established for a wide range of applications. Kneifl et al. [3] build surrogate models of musculoskeletal systems based on PCA and autoencoders for model reduction purposes. Another approach of surrogate modeling is presented by Funk et al [4], where mechanical systems are modeled based on support vector regression. Additional examples of such models include [5–10]. These models are capable of modeling both linear and nonlinear problems with great accuracy. To gain a more detailed overview of these models, we refer readers to Phellan et al. [11].

Further research was conducted in the area of FE surrogates for dynamic problems. For reliability analysis, Barbosa and Rade [12] investigate rotor-bearing systems with a surrogate, among others. Therefore, the vibration amplitudes of the bearings are computed with FEM and used to construct the surrogate model based on kriging. Compared to FEM calculations, the calculation time of the surrogate is lower in the inference phase. However, this method of surrogate modeling is not used to represent the dynamics of a system as a function of time.

If a dynamic problem is not only dependent on time, but should also be solved in the time domain, recurrent neural networks that can process data over time must be used. Thereof, long short-term memory (LSTM) neural networks are the most prominent architecture.

In the area of short-term dynamics, Kohar et al. [13] use LSTMs to model the deformations of crash tubes. Various approaches have been employed to reconstruct the deformation of either a single node or the entire crash tube. In both cases, LSTMs are utilized to manage the time space. If the surrogate should represent the complete model with all degrees of freedom, convolutional neural network (CNN) autoencoders are used to handle the spatial space. These approaches enable reductions in computational costs depending on how many degrees of freedom of the original FE model are to be reproduced.

The focus in the work of Zhang et al. [14] is on seismic, thus oscillating and time-dependent excitation on buildings. Using a physics-informed LSTM metamodel, they calculate the displacements of buildings with scarce data. However, damping issues are not explicitly addressed in the work. Vibrations occur due to excitation with oscillating signals and not because of discontinuous loading.

Ma et al. [15] relate deep learning to sub-modeling methods. For coupling linear sub-models, nonlinear connections are used. These are modeled by artificial neural networks (ANN) to reduce the computation time.

In the work of Baiges et al. [16], low-fidelity models are used as a baseline and corrections to their results are added with the help of ANN to reduce calculation time. Under static load, only a slight error to the reference solution can be detected. Since a phase shift accumulates over time, only the general behavior can be predicted under dynamic excitation.

For applications in the field of soft robotics, Tariverdi et al. [17] use LSTMs to predict the displacement of soft continuum manipulators. As input, magnetic forces and torques which drive the manipulators are available. The architecture of the neural network concatenates this information with the location of the magnets, which are optically measured. Due to the low dimensionality, this surrogate is useful for real-time applications.

Kharazmi et al. [18] investigate marine risers for the prediction of fatigue damage. A LSTM network reconstructs the dynamics of the system by learning measured data from modal decomposition.

For FE models with only few degrees of freedom, it is possible to build a surrogate which handles all degree of freedom directly without any additional methods for the latent space. Koeppe et al. [19] investigate a frame structure with 45 degrees of freedom and use a time-dependent one-dimensional force as single input for the surrogate.

Using decomposition and replacing the numerically expensive time integration schemes through LSTMs is part of recent research work, whereby reduced-order models for nonlinear systems are built. Simpson et al. [20] propose an approach that uses autoencoder to reduce dimensionality. This representation of the problem is learned over time by LSTMs. As an example, a mass spring damper system is evaluated. Damping is introduced with the proportional approach on a fixed basis. Later, this approach is applied to steel monopile foundations, see Simpson et al. [21]. The work of Dutta et al. [22] follows a similar approach for fluid dynamics. With the use of advection-aware autoencoders, data are mapped to shifted versions of the input data. The experiments are subdivided using different Reynold numbers. Hence, no categorical variable is used; a single model has to

be trained for every value. Furthermore, different model architectures are used within the range of Reynolds numbers.

So far, no investigations on surrogates of transient FE simulations with discontinuous loads and damping are reported in the literature, but this is important for modern engineering tasks. Our overarching objective is to methodically demonstrate the enhancement and implementation of surrogate modeling for the aforementioned use case of mechanical systems. Especially for parameter studies, motion planning or control applications, where a large number of simulations have to be computed, our approach is beneficial. Therefore, we present a surrogate for mechanical FE simulations which is capable of reproducing oscillations excited by discontinuous loads. Our surrogate also can handle different damping characteristics by adding a categorical variable to the input data. Through the recursive architecture of the surrogate only short-input sequences are necessary to process long-time dependencies in the output values. This is especially important for the decay process of oscillations, if the damping values allow multiple oscillation periods. Furthermore, our surrogate is able to model the complete domain of a FE simulation. For this purpose, the latent space is modeled with the contribution vector based on modal decomposition.

In this paper, we first show the fundamentals for modal reduction in Sect. 2.1 and for LSTMs in Sect. 2.2. Thereafter, a detailed insight into the architecture of the surrogate is given in Sect. 2.3, followed by the presentation of the data generation scheme in Sect. 2.4 and the training process in Sect. 2.5. Afterward, results are presented for a cantilever beam. In Sect. 3.1, we focus on the recursive architecture of the proposed surrogate, and in Sect. 3.2 different damping values are evaluated. Finally, a conclusion is drawn in Sect. 4.

2 Surrogate modeling

The surrogate is based on a specific combination of neural networks (NN), which are several fully connected dense layers combined with recurrent neural networks (RNNs). With this setup, it is possible to predict the structural dynamics of a mechanical system. Instead of modeling all degrees of freedom from the underlying FE model, we only use a limited number of the systems eigenvectors for modal reduction. Since the eigenvectors are dependent on the boundary conditions, they have to be known in advance. On the basis of this model-order reduction (MOR) technique, the surrogate is trained on systematic provided training data. The main relevant steps of the surrogate are explained in the following.

2.1 Modal reduction

In general, for the FEM, the problem can be described in the discretized linear form of the partial differential equation (PDE), the equations of motion

$$\boldsymbol{M}\ddot{\boldsymbol{u}}(t) + \boldsymbol{C}\dot{\boldsymbol{u}}(t) + \boldsymbol{K}\boldsymbol{u}(t) = \boldsymbol{f}(t). \tag{1}$$

Here, vector $\boldsymbol{u}(t) \in \mathbb{R}^{N\times 1}$ describes displacements, where N is the number of degrees of freedom of the full order model (FOM). Derivatives of $\boldsymbol{u}(t)$ denote velocities and accelerations. $\boldsymbol{M} \in \mathbb{R}^{N\times N}$, $\boldsymbol{C} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{K} \in \mathbb{R}^{N\times N}$ are the mass, damping and stiffness matrix, respectively. Solving a time-dependent mechanical problem, described by Eq. 1, usually requires a valid time integration scheme. For static problems, the damping matrix is often neglected. We focus on dynamic problems and therefore have to consider damping. By assuming a constant $\boldsymbol{C}$, a simplification called modal damping or Rayleigh damping can be made, see Wriggers [23]. In this case, $\boldsymbol{C}$ is assumed to be:

$$\boldsymbol{C} = \alpha_{\text{damp}}\boldsymbol{M} + \beta_{\text{damp}}\boldsymbol{K}, \tag{2}$$

with $\alpha_{\text{damp}} \in \mathbb{R}_0^+$ and $\beta_{\text{damp}} \in \mathbb{R}_0^+$ as parameters for adjusting the damping ratio $D \in \mathbb{R}_0^+$. The damping ratio depends on the damping frequency $f_{\text{damp}} \in \mathbb{R}^+$:

$$D = \frac{1}{2}\left(\frac{\alpha_{\text{damp}}}{2\pi f_{\text{damp}}} + \beta_{\text{damp}} 2\pi f_{\text{damp}}\right). \tag{3}$$

According to Bathe [24], it has to be considered that modes with eigenfrequencies higher than f_{damp} are damped more than lower modes due to the stiffness proportional damping.

In the following, the classical procedure for a modal reduction is shown. For further information, we refer to Wagner [25]. We introduce the modal coordinates $\boldsymbol{z}(t)$ and the rectangular modal matrix $\boldsymbol{\Psi}$

$$\boldsymbol{u}(t) = \boldsymbol{\Psi}\boldsymbol{z}(t) \tag{4}$$

to create an approximation of the full model with displacements $\boldsymbol{u}(t)$. At first, the eigenvalues are calculated. The obtained eigenvectors are summarized in the modal matrix $\boldsymbol{\Phi} \in \mathbb{R}^{N\times N}$. For the reduced model, one can choose a limited number n of eigenvectors to approximate the full model. The selected subset of eigenvectors from $\boldsymbol{\Phi}$ is now denoted as $\boldsymbol{\Psi} \in \mathbb{R}^{N\times n}$. This matrix is used to reduce the matrices on the left and the vector on the right-hand side of Eq. (1):

$$\boldsymbol{\Psi}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{\Psi}\ddot{\boldsymbol{z}}(t) + \boldsymbol{\Psi}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{\Psi}\dot{\boldsymbol{z}}(t) + \boldsymbol{\Psi}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{\Psi}\boldsymbol{z}(t) = \boldsymbol{\Psi}^{\mathrm{T}}\boldsymbol{f}(t), \tag{5}$$

so that a full model approximation with less degrees of freedom is derived. Thereby, $\boldsymbol{z}(t) \in \mathbb{R}^{n\times 1}$ denotes the contribution vector or modal coordinates. Under the premise of nonnegative eigenvalues of $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{K}$, multiplications with the modal matrix result in diagonal matrices, so that

$$\begin{aligned} \boldsymbol{\Psi}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{\Psi} &= \boldsymbol{m}, \\ \boldsymbol{\Psi}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{\Psi} &= \boldsymbol{d}, \\ \boldsymbol{\Psi}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{\Psi} &= \boldsymbol{k} \end{aligned} \tag{6}$$

can be stated. The reduced mass, damping and stiffness matrices are expressed as $\boldsymbol{m} \in \mathbb{R}^{n\times n}$, $\boldsymbol{c} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{k} \in \mathbb{R}^{n\times n}$, respectively. They all are diagonal matrices. The reduced counterpart of Eq. (1) is given by:

$$\boldsymbol{m}\ddot{\boldsymbol{z}} + \boldsymbol{c}\dot{\boldsymbol{z}} + \boldsymbol{k}\boldsymbol{z} = \boldsymbol{\Psi}^{\mathrm{T}}\boldsymbol{f}. \tag{7}$$

Since the matrices are now of diagonal shape, they are decoupled. This allows an efficient time integration scheme, because each row of the system of equations can be solved independently. Time integration results in $\boldsymbol{z}(t)$.

2.2 Long short-term memory

LSTM neural networks can be assigned to the field of recurrent neural networks, which are especially suitable for depicting time series. The input is a sequence of data or time series, which we call history. The output is an approximation of the successive value. RNNs have in common that for one prediction of a time series, the cells have to be forward propagated more than once. The network has to be iterated for each time-step in an input sequence.

The main disadvantage of early RNNs is that with long input sequences, the past history has a vanishing influence on the prediction of a new value. This is because of back-propagation trough time (BPTT), since the error for the weights adjustment in the training process is calculated for each time iteration. One can imagine that multiplying a small value with a small value over and over again leads to a vanishing error and therefore gradient. Same with multiplying values greater than one over and over in the BPTT process. The network does not learn long-term dependencies. The vanishing and exploding gradient problem is addressed in detail by Hochreiter and Schmidhuber [26]. They solved this problems by introducing LSTM networks. The main improvement are the hidden state $\boldsymbol{h}_t$ and the cell state $\boldsymbol{c}_t$, which act as memory over the input sequence. These two streams of information are manipulated in each iteration through time. Since a LSTM cell is a gated network, it can decide whether old information is still relevant or new information should be added and which information is relevant for the output. In summary, a LSTM cell is a small neural network on its own with three parts, the forget, input, and output layer, see Fig. 1.

Data are processed with a sigmoid activation function $\sigma = \frac{1}{1+e^{-x}}$ and a hyperbolic tangent activation function $\tanh = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$. Weights are stored in the variables $\boldsymbol{W}$ with two indices. The first index denotes the affiliation to the data of $\boldsymbol{h}_t$ or $\boldsymbol{x}_t$, the second index indicates forget (d), output (o) or input (i) layer. In the input layer, a new candidate for the cell state, $\tilde{\boldsymbol{c}}$, is introduced. Every multiplication of a weight with data is followed by an addition of a bias. There can be a bias for processing the input data and the hidden state each.

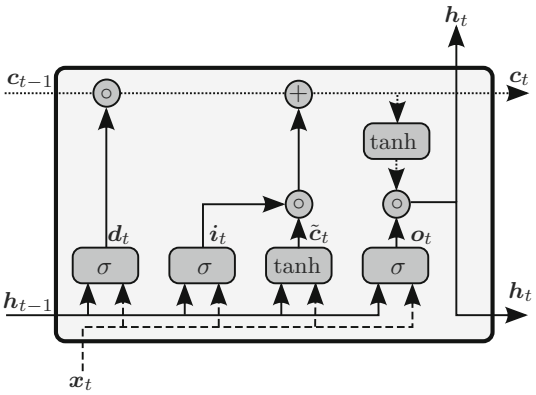

Fig. 1 Architecture of a LSTM cell at time t. With the cell input $\boldsymbol{x}_t$ (dashed line), the previous hidden state $\boldsymbol{h}_{t-1}$ (solid line) and the previous cell state $\boldsymbol{c}_{t-1}$ (dotted line), a new hidden state $\boldsymbol{h}_t$ and cell state $\boldsymbol{c}_t$ can be calculated. In optional additional LSTM layers, $\boldsymbol{h}_t$ will be processed as input, equivalent to $\boldsymbol{x}_t$ in this figure. After the cell has been looped sequence length times, $\boldsymbol{c}_t$ represents a predicted value in the last step

At the beginning of every BPTT, $\boldsymbol{h}_{t-1}$ and $\boldsymbol{c}_{t-1}$ are set to a tensor of zeros. First, the input layer has to be passed. Therefore, information from the input sequence $\boldsymbol{x}_t$ of length l_{seq} and the previous, here initial, hidden state $\boldsymbol{h}_{t-1}$ are processed:

$$\boldsymbol{d}_t = \sigma(\boldsymbol{W}_{\text{xd}}\boldsymbol{x}_t + \boldsymbol{b}_{\text{xd}} + \boldsymbol{W}_{\text{hd}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\text{hd}}). \tag{8}$$

Elementwise multiplication of the tensor $\boldsymbol{d}_t$ with $\boldsymbol{c}_t$ leads to the first update of $\boldsymbol{c}_{t-1}$. In neural networks, data are processed batchwise, which means all samples in a batch are processed at once due to parallelization.

Thereby, the weights apply to $\boldsymbol{W}_{\text{xd}} \in \mathbb{R}^{n_{\text{f}} \times n_{\text{h}}}$ and $\boldsymbol{W}_{\text{hd}} \in \mathbb{R}^{n_{\text{h}} \times n_{\text{h}}}$ with n_{f} as the number of features and n_{h} as the number of hidden units. For clarity, the states for one timestep are defined as $\boldsymbol{h}_t \in \mathbb{R}^{n_{\text{b}} \times n_{\text{h}}}$ and $\boldsymbol{c}_t \in \mathbb{R}^{n_{\text{b}} \times n_{\text{h}}}$, where n_{b} is the batch size. For one timestep of the input sequence, $\boldsymbol{x}_t \in \mathbb{R}^{n_{\text{b}} \times n_{\text{f}}}$ can be stated with $\boldsymbol{x}_t$ from the input sequences $\boldsymbol{x} \in \mathbb{R}^{l_{\text{seq}} \times n_{\text{b}} \times n_{\text{f}}}$.

For the next update of $\boldsymbol{c}_{t-1}$, input data and hidden state have to be processed in the input layer:

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{\text{xi}}\boldsymbol{x}_t + \boldsymbol{b}_{\text{xi}} + \boldsymbol{W}_{\text{hi}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\text{hi}}), \tag{9}$$

$$\tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_{\text{xg}}\boldsymbol{x}_t + \boldsymbol{b}_{\text{xg}} + \boldsymbol{W}_{\text{hg}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\text{hg}}). \tag{10}$$

Thereby, $\boldsymbol{i}_t$ decides which data should be part of the update of $\boldsymbol{c}_{t-1}$ and $\tilde{\boldsymbol{c}}_t$ decides how strongly these new information should update $\boldsymbol{c}_{t-1}$. For the summarized update of $\boldsymbol{c}_{t-1}$ to $\boldsymbol{c}_t$, the expression $\boldsymbol{c}_t = \boldsymbol{d}_t \circ \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \circ \tilde{\boldsymbol{c}}_t$ has to be evaluated, where $\circ$ denotes element-wise multiplication.

The last layer in the LSTM which will be processed is the output layer. There, the hidden state $\boldsymbol{h}_t$ will be updated by

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{\text{xo}}\boldsymbol{x}_t + \boldsymbol{b}_{\text{xo}} + \boldsymbol{W}_{\text{ho}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\text{ho}}) \tag{11}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \circ \tanh(\boldsymbol{c}_t). \tag{12}$$

Equations (8)–(12) will be evaluated l_{seq} times for each value in $\boldsymbol{x}$ with the same weights $\boldsymbol{W}$. The predicted value $\hat{\boldsymbol{x}}_1$ is set as the latest cell state $\boldsymbol{c}_{l_{\text{seq}}+1}$.

In deep learning, there are several concatenated neural networks. For the specific use cases of LSTM networks, the input sequence of the second layer will not be $\boldsymbol{x}$, it will be the hidden state of the previous layer instead, see $\boldsymbol{h}_t$ in the right upper corner of Fig. 1. Consequently, the number of features in the second LSTM cell layer will be the number of hidden units in the first LSTM cell layer. The number of features in the cell state of the last LSTM cell is not necessarily the number of output units for further data processing. This issue is solved by adding a fully connected neural network layer after the last LSTM cell.

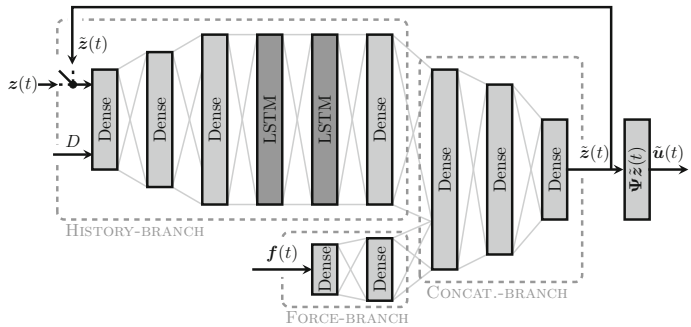

Fig. 2 Architecture of the proposed surrogate with additional categorical variable D. Depending on whether the model is in the inference or in the training phase, and depending on the training method itself, $\tilde{z}(t)$ or $z(t)$ is an input to the first dense layer. In the figure this is depicted with a switch before the first dense layer. Data from the HISTORY- BRANCH and the FORCE- BRANCH are concatenated for further processing. Whether the architecture has to reduce or increase the feature space depends on the problem statement. Here, the feature space is increased

2.3 Architecture of the surrogate

The surrogate approximates linear FE models for transient structural analysis, which are reduced with modal decomposition. Therefore, we calculate eigenvectors of the mechanical system with which a modal reduction can be carried out, as explained in 2.1. To distinguish variables of the surrogate model from their physical counterpart, the tilde symbol is added above them.

We strive to build a surrogate which is as far as possible independent of the sequence length l_{seq}, as it is the case in FE calculations. There, the transient solution is dependent on just the latest timestep for a one-step procedure or only a few timesteps for a multi-step procedure, respectively. To achieve that, the output of the model, the surrogate to $\boldsymbol{u}(t)$, which we denote as $\tilde{\boldsymbol{u}}(t)$, should not only depend on the excitation, and moreover, the history of $\tilde{z}(t)$ should be considered for further prediction steps. To fulfill this requirement, we choose a recursive model architecture, see Fig. 2. Due to the recursive architecture, the input of the model differs during training and inference phase. When training in an open loop, $z(t)$ is an input variable to the model, when training in a closed loop, $\tilde{z}(t)$ is an input variable to the model. The training routine is explained in detail in Sect. 2.5. In the inference phase, always $\tilde{z}(t)$ is an input to the model. The output of the model in the inference phase is $\tilde{\boldsymbol{u}}(t)$, but one has to note at this point, that the model is trained on $\tilde{z}(t)$, see Sect. 2.5. No derivatives of $\tilde{z}(t)$ with respect to time are part of the surrogate model. Hence, it is necessary to model $\tilde{z}(t)$ based on a time series or sequence in order to predict new values according to the current status of the system. Therefore, LSTMs are a proper choice from the field of neural networks. The term recursive has to be distinguished from the term recurrent. Recursive describes the architecture of the surrogate, recurrent is associated with LSTMs. In the surrogate the contribution vector $\tilde{z}(t)$ is fed back to the input layer for every new prediction. On its way to the output layer, new information regarding the excitation or even the damping ratio in form of a categorical variable join the data stream and have a significant influence on it.

First, data of the main input sequence are processed with several fully connected layers. Depending on the complexity of the FE model, the feature space is successively expanded or reduced to a higher or lower order representation. This specific step is recommend by Goodfellow et al. [27]. Therefore, as activation function tanh is used. As an alternative, the rectified linear unit (ReLU) function can be chosen, as proposed by Koeppe [28]. Processing data with ReLU functions adds also some nonlinearities to the model, since ReLU is defined as $f(x) = \max(0, x)$. However, experiments showed an inferior capability to model the system dynamics. A possible explanation could be that ReLU cannot produce negative values and therefore does not match the original data range. Second, data will be processed with LSTMs. Here, also several layers are lined up. To prevent overfitting in the training process, dropout layers, see Goodfellow et al. [27], with a weight clearing probability of 0.2 are introduced. The hidden size of the LSTMs can vary too, in order to have even more instances for predicting new values. Up to this point, only information of previous contribution vectors and the optional damping ratio are considered. Hence, only information about oscillations are handled. We call that the HISTORY- BRANCH. The architecture is physics inspired, which means that the flow of information into the surrogate must be given at its intended layer. Since damping information has to be evaluated in the HISTORY- BRANCH to achieve a dissipation effect, they cannot be attached after the LSTM layer. Therefore, the values of this categorical variable are repeated, so that a constant sequence is created which can now be

concatenated to the main input sequence. If damping information joins the data stream after the LSTMs, the damping ratios will not influence the surrogate. Moreover, the results will always be near to a solution with the mean value of the presented damping ratios in the training set.

In a third step, information about the external excitation is added to the data stream. We call that the FORCE-BRANCH. For this purpose, we studied multiple concepts. The best performing concept is depicted in Fig. 2. Information of the HISTORY- BRANCH and the FORCE- BRANCH are concatenated in the CONCATENATION-BRANCH. As an alternative, tensors from the different branches can be brought to the same dimension and can be added. We find that approach less suitable for the surrogate, due to the fact that it has a superposition character. Of course, this is no downside for linear problems, but if the surrogate should be extended for nonlinear problems, this could lead to false interpretations.

The introduction of the damping information at the CONCATENATION- BRANCH is also investigated. However, this approach was not capable of building a sufficient surrogate. Dissipation effects are not correctly reproduced, and therefore, this approach is inferior compared to processing damping information as categorical variable through the HISTORY- BRANCH.

In the fourth and last step, the previous concatenated data are processed with one or more fully connected layers. Again, the ReLU and tanh activation functions are investigated. In this step, it proved to be even more important to use tanh to produce output values between $[-1, 1]$. This part of the neural network reduces or increases the dimension of the output tensor in order to match the dimension of the contribution vector again. Otherwise, the recursive architecture of the model could not be established.

Finally, the output of the neural network, $\tilde{z}(t)$, has to be expanded with equation 4 to the full model dimension.

One can explain the behavior of the surrogate as follows. The surrogate contribution vector follows the excitation under the FORCE- BRANCH. Hence, without external manipulation of $\tilde{z}(t)$ and no changes of the force, $\tilde{z}(t)$ will stay constant over time. If forces change in an discontinuous manner, the next prediction of $\tilde{z}(t)$ will add some oscillations to $\tilde{z}(t)$ due to the recursive architecture of the surrogate. Depending on the damping ratio as categorical variable, the oscillation will decay. Moreover, the oscillations are now preserved in the input sequence. Processing the sequence in the HISTORY- BRANCH will preserve the oscillations, even if the excitation is held constant again. In other surrogates, the output purely depends on the information in the input sequence [13,18], which is in most cases a sequence containing forces over time [19,20]. Comparatively, our recursive architecture has one main benefit. An adequate sequence length of the input has no influence on reproducing the history of the excitation, since history is decoupled from excitation. To keep an oscillation ongoing, its cause must be within the input sequence for non-recursive architectures. More precisely, if an oscillation takes t_{stat} to fully dissipate, an input sequence length of $\frac{t_{\text{stat}}}{\Delta t}$ is necessary to model the system. Modeling a complete oscillation in the dimension of $t_{\text{stat}} = 1s$ and $\Delta t = 1 \cdot 10^{-4}$s with a nonrecursive architecture would need 10^5 entries in the input sequence. Following Neil et al. [29], training sequences of this cardinality is not practicable. They first have to be reduced, for example by a decoder–encoder network. This is an unwanted additional computational effort. Of course, the recursive structure needs also additional computations, but they are negligible compared to longer input sequences. In the examples for this work, we use a timestep of $\Delta t = 1 \cdot 10^{-4}$s. Prediction time with the surrogate correlates linearly with Δt. Hence, choosing a timestep of $1 \cdot 10^{-3}$s would lead to a 10 times higher reduction of computational costs. Also l_{seq} for a nonrecursive architecture would be reduced by a factor of 10. On the other side, a sufficient resolution of the displacements over time has to be guaranteed. This trade-off is dependent on the mechanical system which is evaluated.

Due to the use of modal reduction, as a downside of our surrogate, the location of the boundary conditions has to be known for the generation of the surrogate. This is because the eigenvectors with these specific boundary conditions have to be computed first. However, this disadvantage is common practice for a wide range of modal reduction applications as in the approach of Craig-Bampton and others, see Sonneville et al. [30]. Noteworthy, the locations of the exciting forces also have to be known for the data generation. After training, additional forces cannot be applied to other nodes. In order to start predictions with the surrogate, one has to define an initial input sequence for $\tilde{z}(t)$ due to the recursive architecture. This limitation can be circumvented by setting all entries in $\tilde{z}(t)$ for the whole initial sequence to zero or to constant values based on the load level. As a result, a nonoscillating starting point for the following predictions is found. As l_{seq} is short compared to the length of the training data, the additional zeros do not affect the robustness of the solutions obtained with the surrogate.

2.4 Generation of training data

As depicted in Sect. 2.3, the surrogate uses force and contribution vectors as input data. Optionally, the damping coefficient as categorical variable is also part of the datasets. These three main types of features are time-dependent, except of the categorical variable, and since their values may have different orders of magnitude, they need to be normalized.

A popular approach is scaling to a range of $z_{\text{scaled}} \in [-1, 1] = \{z_{\text{scaled}} \in \mathbb{R} | -1 \leq z_{\text{scaled}} \leq 1\}$ by:

$$z_{\text{scaled}} = \frac{2\big(z - \min(z)\big)}{\max(z) - \min(z)} - 1. \tag{13}$$

Scaling is done for each feature individually, as well as for the force vector $\boldsymbol{f}$. For this reason, each feature is equally important for the neural network and the focus is not on the features with the highest absolute values. Furthermore, it is important to generate a balanced dataset, where all features have to be centered around zero. That means in our case that displacements in positive and negative direction should be predicted with equal accuracy. Since removing the mean is not conducted by Eq. (13), training data has to be chosen carefully on both sides of the abscissa.

Due to scaling, the maximum absolute value of displacements and forces needed in the inference phase has to be known at this point. Predictions with the model are limited to values within training range. But this disadvantage is not relevant for the proposed surrogate, since linear models are approximated. Hence, one can first scale the exciting force to the trained range, and after prediction rescale displacements. Therefore, the range of the modal coordinates is not exceeded.

To fulfill the aforementioned requirements, the training data routine creates at first three-dimensional ramped force curves where the plateaus and slopes are set in a random manner. The plateaus are chosen from a uniform distribution in the range [-1,1]. These values are then scaled with a force, which is constant for all datasets and therefore marks the maximum force, which can be applied to the system in the inference phase. The times associated with forces are sampled by taking values from an uniform distribution [0,1]. After that, they are sorted and scaled with the simulation time of the dataset. Thus, pairs of forces and times are generated. To obtain suitable force curves, an interpolation is carried out between these grid points. The load curves of the ramped configuration differ on the following points, that means, generalization of the model has to be interpreted according to them:

- force plateaus are on different load levels,
- force plateaus are held for a different time, hence, oscillation decays differently,
- the slope is different, and hence, the mechanical system is excited with oscillations of different amplitudes.

Due to the modal reduction process, the force curves are transformed according to Eq. (5). Therefore, generality is limited to the spectral characteristics of the training data. The length of the curves is set to be $t_{\text{td}} = 0.3s$. All curves begin with $\boldsymbol{f}_{\text{x, sel.}}(0) = \boldsymbol{f}_{\text{y, sel.}}(0) = \boldsymbol{f}_{\text{z, sel.}}(0) = 0$ for at least $0.01s$. The term sel. in the index denotes selected nodes where the force is applied. Each curve is duplicated, and the additional curves are then inverted. This leads to a symmetric dataset with a mean value of zero. Since discontinuities in the force curves lead to extraordinary strong and therefore unrealistic oscillations, depending on the damping associated with the simulation, tangential transitions are implemented between slopes and plateaus.

Next, the force curves are used as input sequences for explicit FEM calculations. For that, an appropriate timestep Δt_{FEM} is chosen for integration over time. The surrogate does not need an adequate timestep for a stable solution, rather, a fixed timestep is mandatory. Hence, results are interpolated with $\Delta t_{\text{Surr.}} = 1 \cdot 10^{-4}$s, independent of Δt_{FEM}. As described in Sect. 2.3, the initial FE degrees of freedom are not used to approximate the system's spatial space. Indeed, only the modal coordinates are used to train the model. Basically, these are several oscillating curves, which can be seen in Fig. 3a.

The surrogate should not learn fixed oscillations on different load levels, rather it should be able to generalize. Hence, oscillations on not trained load levels should be predicted with high accuracy. Therefore, the surrogate has to predict oscillations because of changes in the force curves. On the other side, oscillation can occur because of oscillating force curves. A finding here is that it is necessary to add input sequences of that type to the training data set, see Fig. 3b. In the data generation scheme, these force curves are also manipulated in a random manner. The basis is a sine, so for one degree of freedom $f(t) = \sin(bt + d)a + c$ can be stated, which undergoes random adjustments which are scaling time with b as well as shifting and scaling force f with c and a.

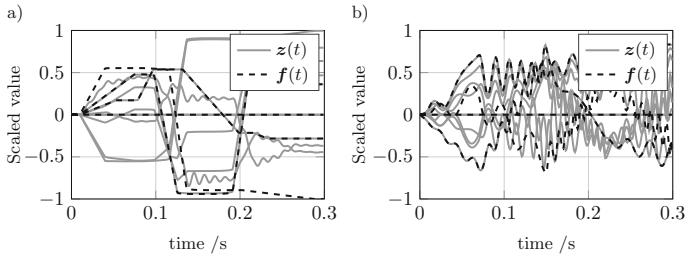

Fig. 3 Modal coordinates (solid lines) based on an excitation of the example shown in Sect. 3.1 with force curves (dashed lines) in the ramped configuration (**a**) and oscillating configuration (**b**)

Consequently, these curves are inverted too by multiplying c with -1 and setting $d = \pi$. Contribution vectors, compare Fig. 3b, are computed with FEM. They show oscillations due to the excitation with an oscillation force as well as the dynamics of the system itself.

In order to obtain identical training data for each component of the three-dimensional excitation, each force profile is used for every component. This prevents the surrogate from predicting displacements caused by a specific force component with lower accuracy.

In a last step, data are generated for a one-dimensional excitation. Therefore, the same force curve is only used for one component, the other two components are set to zero. This setup is shuffled until every combination is represented in the dataset.

2.5 Training of the model

Each evaluated dataset consists of a time series with sequence length plus one modal coordinates z_i, $i = 1, ..., l_{\text{seq}} + 1$. All of them, except the last one, are used as input sequence for the forward propagation. The last modal coordinate vector of this series, $z_{l_{\text{seq}}+1}$, is used for the mean-squared error (MSE) calculation with its predicted counterpart, $\tilde{z}_{l_{\text{seq}}+1}$,

$$\epsilon_{\text{MSE}} = \frac{(\tilde{z}_{l_{\text{seq}}+1} - z_{l_{\text{seq}}+1})^2}{n}. \tag{14}$$

After calculation of the error, which is the squared L2 norm between the prediction and the target, BPTT is performed. For training of the complete network Adam optimizer, see Kingma and Ba [31], is used. Since adjustment of weights W is done in batches, the errors ϵ_{MSE} of each sample in a batch are averaged or summed before BPTT is conducted.

For LSTM neural networks, training data has to undergo preprocessing. Sequences with the desired sequence length l_{seq} have to be extracted out of the time series generated in the previous section. Here, this is done with a stride of one.

As usual in all machine learning or AI tasks the dataset has to be split into a training, test and validation set:

- Training dataset: 70% of the training/test dataset,
- Test dataset: 30% of the training/test dataset,
- Validation dataset: Additional dataset generated with identical data generation routine as for training/test dataset, but with different random state, see Sect. 2.4.

In order to prevent data leakage a strict separation of these datasets is preserved. This is done by storing each dataset as a separate file and loading this file when necessary.

A bottleneck in the training process is the memory requirement, since a lot of repeated data, approximately l_{seq} times the data generated, has to be stored. This can amount several gigabyte, which is not to be stored in one matrix. Thus, only a subset of sequences is processed at once. The affiliation of the sequences to such a partition is shuffled over the epochs.

Before training can start, the weights have to be initialized. This is done with a Xavier normal distribution based on the work of Glorot and Bengio [32]. In a first step, training follows the TEACHER- FORCING (TF) concept. Thereby, an input sequence with true data from the training set is forward propagated. After that, an

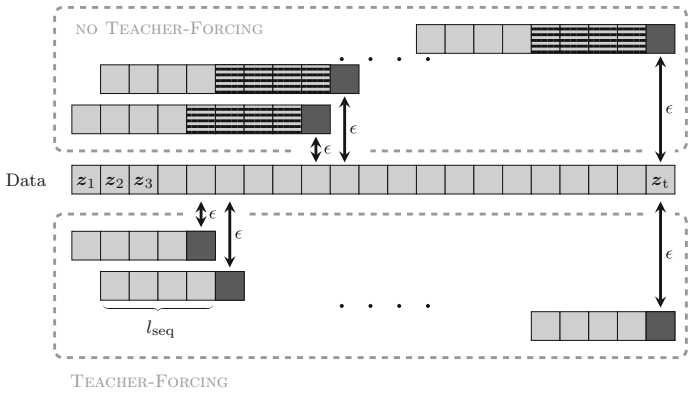

Fig. 4 The concept of TEACHER- FORCING and NO TEACHER- FORCING with a sequence length of four. Fields with stripes are predicted values, which are used as input sequence for the training process. Dark gray fields represent predicted values for the error calculation. The index of z denotes the timestep

error between predicted vector $\tilde{z}_{l_{\text{seq}}+1}$ and true vector $z_{l_{\text{seq}}+1}$ is calculated by use of Eq. (14). This concept promises fast convergence of the weights. But one has to note that in the inference phase of the surrogate, we use an open-loop strategy. Prediction starts based on an initial input sequence. The new predicted value is set in front of the initial input sequence. In order to keep the sequence length, the first value of the input sequence is dropped, see Fig. 4. In summary, we do predictions on predicted values. This is a completely new use case compared to the one seen in the training process. Since predictions vary from true values, predictions on top of that can become less precise over the number of predictions, see Goyal et al. [33]. In our usecase, strong, not expected oscillations occur. This problem is known as exposure bias and addressed in Schmidt et al. [34].

To counter exposure biasing, training is conducted with use of the NO TEACHER- FORCING (NTF) concept investigated by Bengio et al. [35]. Given an input sequence from the dataset in a first step, l_{seq} predictions are made. On the basis of this sequence of predictions, an additional forward propagation with saved gradients is conducted for the weight adjustment process. This is basically the process in the inference mode, where predictions on the basis of predictions are made. This forward propagation results in a new vector, $\tilde{z}_{l_{\text{seq}}+1}$. Finally, an error can be calculated again. With BPTT and the Adam optimizer, the surrogate is updated. Thus, the length of a training sequence now has to be $2l_{\text{seq}}$. Figure 4 illustrates the two concepts for the training process.

Noteworthy, training with NTF is very time-consuming, since the whole training dataset has to be predicted additionally before prediction for the calculation of the error can be made. Also, convergence is achieved not as fast compared to TF, since the input sequence for the BPTT is totally different to the input sequence, which will later lead to sufficient predictions. For solving that problem, MIXED- TEACHER- FORCING (MTF) can be used. This splits the number of epochs into three sections. First, one follows purely the TF approach, where the surrogate learns to predict on true values. In the second section, TF and NTF are mixed in a randomized manner going from TF to NTF. If $\frac{epoch - epoch_{\text{start,MTF}}}{epoch_{\text{end,MTF}} - epoch_{\text{start,MTF}}}$ is greater than a random number between zero and one, NTF is chosen. The decision between TF and NTF is made within one epoch several times. Consequently, the training and test scores increase again since NTF is not known yet to the surrogate and therefore must be trained on first. Also, a high variance in both scores can be seen, compare Fig. 5. In the third section, pure NTF is performed. The surrogate learns to predict in an open-loop application. Both training and validation scores decrease again until convergence is achieved.

Decreasing the learning rate in a cyclic scheme, as proposed by Smith [36], does not lead to a faster convergence. Thus, only exponential decay of the base learning rate of $\alpha = 1 \cdot 10^{-4}$ is used. Batch size per weight adjustment is set to 1024. Due to steep flanks in the optimization domain, the optimizer may suffer from high gradients. Pascanu et al. [37] solved this problem by introducing gradient clipping, where the gradient norm is truncated at a maximum value.

Another hyperparameter, which has a big impact on the results, is the sequence length l_{seq}. This parameter defines how far in the past the data influences the predictions and has to be defined already in the first preprocessing step. A summary of all hyperparameter is given in Table 1.

When comparing the train and the test datasets, one notices that the test error is smaller than the train error. This is not because of an easy to predict test dataset or data leakage. Rather it is because of the dropout

Table 1 Best hyperparameter for the training of the surrogate

Hyperparameter	Value
Sequence length, l_{seq}	50
Number of epochs	3000
Number of weight updates	390
Batch size, n_{b}	1024
Learning rate, α	$1 \cdot 10^{-4}$
Gradient clipping	0.5

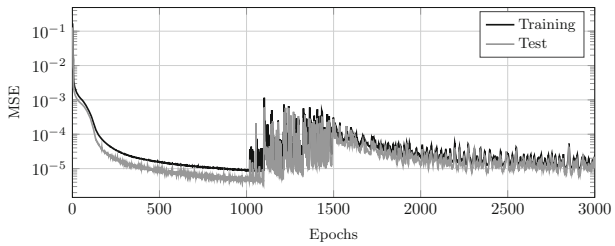

Fig. 5 Train and test score of the surrogate. Both start to increase in the region of MTF between epoch 1000 and 1500

layers, which are active while training and inactive while testing. Therefore, better results can be expected from testing. Furthermore, the training error for an epoch is calculated by taking the mean batchwise error. Hence, especially for early batches of a training epoch, the neural net is not as often updated as at the calculation of the test error. It is important to mention, that this score in fact computes how accurate the surrogate will deal with one-step predictions. How the surrogate will behave in an open-loop configuration is not estimated at this point.

In the training process of the example in Sect. 3.1, one can see in Fig. 5 a decreasing train and test score until convergence has been accomplished. Training could be stopped after 2600 epochs, since no further improvement of the surrogate can be found. Furthermore, it can be clearly seen where the training concept starts to change from TF to NTF. Noteworthy, the low error level at the end of the pure TF phase cannot be reached at the end of the NTF phase. For one-step predictions, the surrogate at epoch 1000 promises best results, but if an open-loop configuration is necessary, the surrogate with weights from the last epoch will outperform the TF trained model.

3 Results and discussion

In this section, we present results from the surrogate and compare them with solutions obtained from FE simulations. First we introduce an academic example of a cantilever beam investigated with fixed damping. Second, the cantilever beam is evaluated under variable damping due to the introduced of a categorical variable. Third, a real-world problem of a six-axis robot is investigated. For this purpose, the validation dataset is used, which is not part of the training process.

During training only one-step predictions are made. In contrast, for the validation process, all predictions, except when based on the starting sequence, are done based on predicted values. This scenario is the use case we are aiming at. Hence, the switch at the top left in Fig. 2 is connected to $\tilde{z}(t)$. Thus, the surrogate model is validated methodically on other examples as used for training, which, apart from the strict separation of all datasets, additionally ensures the absence of data leakage.

3.1 Cantilever beam with constant damping

The cantilever beam is fixed on one side and deflected by a force on the opposite side, as shown in Fig. 6. Also the node for the following comparisons is located on the deflected side. The initial FE model consists of 640 linear hexahedron elements with 1025 nodes. A fixed damping ratio of $D = 0.057$ is chosen. For the modal reduction, the first ten eigenvectors are used. The model is then solved with the explicit leap-frog time integration scheme, see Wagner [25].

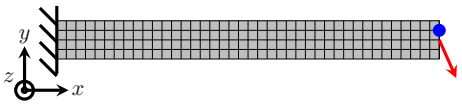

Fig. 6 Investigated cantilever beam with fixed support on the left-hand side and load on the right-hand side. Validation of the displacements is done at the node highlighted with dot

Table 2 Architecture of the surrogate for the cantilever beam example. The last number in HISTORY- BRANCH, LSTM is for the dense layer

Layer	$n_{\text{features, out}}$
HISTORY- BRANCH, dense	12, 16, 24, 32
HISTORY- BRANCH, LSTM	32, 48, 32, 32
FORCE- BRANCH	6
CONCAT.- BRANCH	32, 24, 16, 10

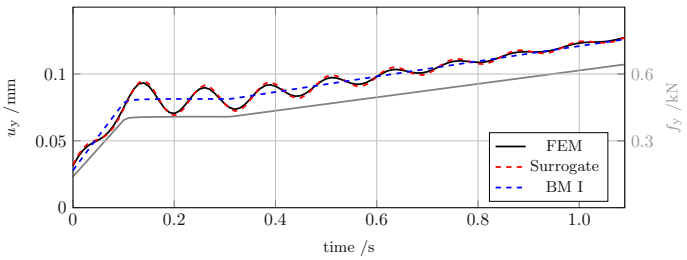

Fig. 7 Displacement of the cantilever beam in the y-direction where the uniaxial force is applied. Solutions from the FEM, our surrogate and BM I are shown

A complete training set for the cantilever beam with fixed damping consists of a total of 40 subsets, where 30 are from the ramped configuration and ten are from the oscillating configuration, see Sect. 2.4. Experiments have shown that fewer subsets have a negative impact on the generalization ability of the surrogate. In summary, there are 116,000 samples for training.

The surrogate has 37,400 trainable parameters; a detailed overview of the specific architecture is given in Table 2.

To emphasize the advantages of our recursive architecture, we compare our solution with two different baseline models. For consistency, the baseline models are trained with the same dataset as the surrogate.

As a first baseline model (BM I) for a surrogate of transient FE simulations, a neural network is consulted. This neural network consists only of fully connected dense layers. Using this naive model greatly reduces the effort for the training strategy and duration due to a more simple architecture. A comparison of BM I with FEM shows that displacements only follow the force curves, see Fig. 7. Dynamic effects in the oscillations are not reproduced. Hence, for transient simulations, the surrogate outperforms BM I.

The second baseline model (BM II) is derived from the work of [13,19,20], where only the force is used as input variable. For the LSTM layers in BM II, an identical number of neurons as in our surrogate is used. Equal dimensionality of the input features is increased by several dense layers, see Table 2. The need for a high sequence length for the reproduction of the FE results is illustrated in Fig. 8, as different sequence lengths are evaluated.

To show the influence of the sequence length, a load case of the ramped configuration is investigated. To simplify the training process of BM II, we use a training dataset for this comparison. This is valid here, since generalization is not in the focus. For BM II - 100, oscillations start at the point where the first discontinuity in the load curve can be found at $t = 0.1$s. The first kink dynamically excites the system. Compared to the displacements obtained by FE, we consider the results sufficient. At $t = 0.2$s, we note an abrupt decay of the oscillation. The same behavior can be observed by BM II with a sequence length of 250. The first and second kink of the load curve are only separated by 0.2s; hence, BM II - 250 is able to reproduce the displacements for that plateau. For displacements after $t = 0.3$s, BM II calculates under both sequence lengths again oscillating results. This is due to the repeated change in the force. After 0.1s and 0.25s, respectively, the

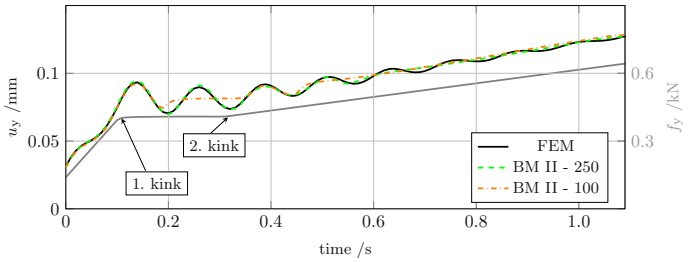

Fig. 8 Displacement of the cantilever beam in the y-direction where the uniaxial force is applied. Solutions from the FEM and BM II with two different sequence lengths are shown. For simplicity, the result of the surrogate is not shown again, see Fig. 7

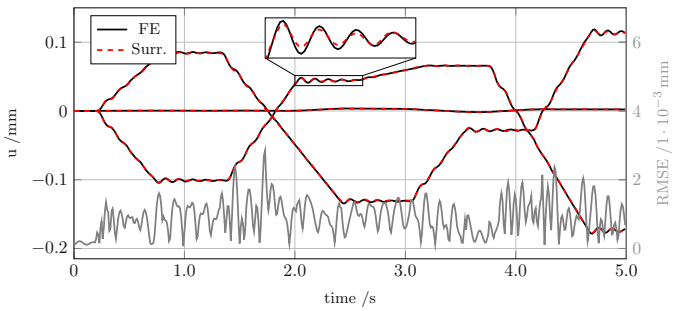

Fig. 9 Displacement of the cantilever beam's evaluation node based on FE calculations and the surrogate under a load case of the ramped configuration. Error is calculated as RMSE over the three displacements u_x, u_y and u_z. The FE and surrogate solution match accurately

oscillation decays again, although this is not the expected behavior. In contrast, the surrogate follows the FE solution exactly, although the sequence length is only 50, see Fig. 7.

The dynamics of the system are reproduced with high accuracy, as long as the causal excitation region is less than l_{seq} steps in the past. After that, BM II has a similar behavior to BM I, as no dynamic effects are captured.

If the excitation only affects a few hundred timesteps, BM II is a feasible approach to build surrogates. Since in the BM II approach no predictions of the past are needed to predict new values, parallelization in the prediction of different timesteps allows a enormous reduction of the simulation time. But if dynamic effects need several hundred timesteps to decay, training BM II will be very difficult, due to the underestimation of the long-term dependencies. With our surrogate the sequence length will not influence the accuracy of a prediction due to the recursive architecture.

Similarly, there are applications where one is not able to predict a range of timesteps at once, because predictions are dependent on environmental data. In this case, despite the advantages with a short sequence length, BM II cannot benefit from parallelization. To summarize, the presented surrogate will also outperform BM II.

To prove the generalization ability of our surrogate, new and therefore unseen load curves are generated. On the one hand, these load curves are used as an input for the reference solution computed with FEM. On the other hand, displacements based on these load curves are calculated with the surrogate, see Fig. 9. The summed RMSE of the surrogate over all components of the displacement has a maximum value of $\epsilon_{\text{RMSE, sum}} = 2.8 \cdot 10^{-3}$mm. Noteworthy, the RMSE is a pessimistic metric for the description of the similarity between the results, since a slight phase shift creates high errors. This is one reason for the oscillating error, which has the same oscillation frequency as the displacements. Also oscillation amplitudes of the surrogate are slightly underrepresented. Apart from that, dissipation effects and frequency information are met with high accuracy, see the detail in Fig. 9. Furthermore, 100 validation load cases are evaluated. For every timestep of the expanded solution, the RMSE of the evaluated node is calculated, $\epsilon_{\text{RMSE},\,t} = \sqrt{\sum_{i=1}^{3}(\boldsymbol{u}_{\text{FEM}} - \boldsymbol{u}_{\text{Surr.}})^2}$. The mean error of all evaluated $3 \cdot 10^5$ predictions is $\epsilon_{\text{RMSE, mean}} = 1.08 \cdot 10^{-3}$mm. Values for the displacement are in the range $u = [-0.2, 0.2]$ mm.

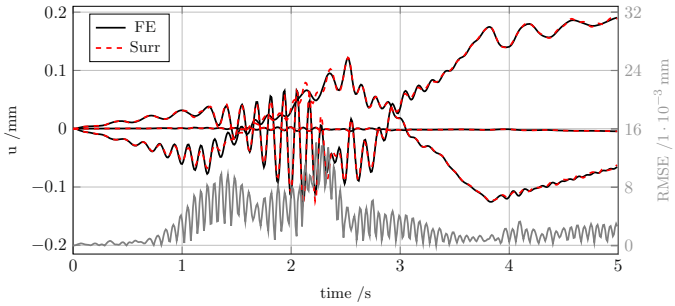

Fig. 10 Displacement of the cantilever beam's evaluation node based on FE calculations and the surrogate under a load case of the sine configuration. Error is calculated as RMSE over the three displacements u_x, u_y and u_z. The FE and surrogate solution match accurately

Table 3 Duration for a simulation time of $t_{sim} = 0.5$s for the cantilever beam example. Reference simulations are computed with the transient structural ANSYS solver. For reduced-order solutions, 10 eigenvectors are used

one-time computations		repeated computations			
$t_{comp.\ eig.}$	$t_{training\ Surrogate}$	t_{FOM}	$t_{mod.\ red.}$	$t_{Surrogate}$	$t_{mod.\ red.}/t_{Surrogate}$
2s	$315 \cdot 10^3$s	177s	21s	12s	1.8

The accuracy of the results obtained from the surrogate is identical for the whole load spectrum. This is only possible because of the symmetric training dataset and the evenly distributed load curves for each load component.

With an excitation in the oscillating configuration, similar results can be obtained, see Fig. 10. Again, 100 new load curves are used for this calculations. The mean RMSE of all evaluated $3 \cdot 10^5$ predictions is $\epsilon = 6.49 \cdot 10^{-3}$mm, which is lower than that of the ramped configuration. In the training process, data out of the oscillating configuration are underrepresented compared to the ramped configuration. Especially force curves with a high frequency in the sine have difficulties to match displacements with the high accuracy of the other examples. Nevertheless, results are sufficient, considering that we focus on discontinuous loads and this type of force curves are only necessary in the training to separate handling of data in the HISTORY BRANCH and in the FORCE BRANCH.

Computation of results with the surrogate is faster compared to full- and reduced-order FE models. Reference calculations are computed using the ANSYS solver for transient structural analysis. In Table 3, the durations for the generation of the results are presented for a problem with a simulation time of $t_{sim} = 0.5$s. All computational times shown in Table 3 are computed with an i9-10900K CPU.

For both model reduction techniques, standard modal reduction and surrogate modeling, the computational time of extracting eigenvalues is identical and only necessary once for both approaches.

Computational time for training the surrogate is 5.3s for each TF epoch and 176s for each NTF epoch on a Quadro P2200 graphic card. One has to note that for this example no measures are taken to reduce training time. To summarize, the NTF phase is crucial for the training due to l_{seq} forward propagations for every dataset, see Sect. 2.5. Nevertheless, in a feed forward controlling application the surrogate model will amortize the training effort. Therefore, the surrogate will lead to a substantial benefit.

3.2 Cantilever beam with variable damping ratio

Investigating the same problem of the cantilever beam, the capability of the model to predict deformations in dependency of different damping ratios is evaluated. Therefore, a categorical variable D is attached to the LSTM input tensor, see Fig. 2.

In contrast to the force curve, the damping ratio is out of the set of numbers $D = \{0.032, 0.057, 0.083, 0.110\}$. These values are calculated using Eq. (3), with $\alpha_{damp} = 6$ and $\beta_{damp} = \{1, 2, 3, 4\} \cdot 10^{-4}$. The damping frequency is set to be $f_{damp} = 82$Hz, since this is the first natural frequency with the relevant eigenvector for the investigated load cases. For the training, values of D are chosen to result in underdamped oscillations. A complete training set for this study consists of a total of 192 subsets, where 168 are from the ramped configu-

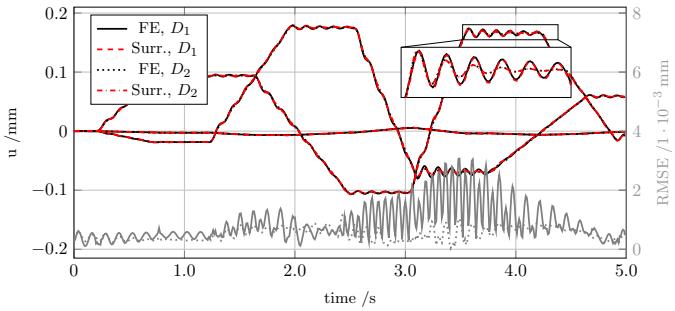

Fig. 11 Displacement of the cantilever beam's evaluation node based on FE calculations and the surrogate under a load case of the ramped configuration and variable damping ratios $D_1 = 0.032$ and $D_2 = 0.110$. Error is calculated as RMSE over the three displacements u_x, u_y and u_z. The FE and surrogate solution match accurately

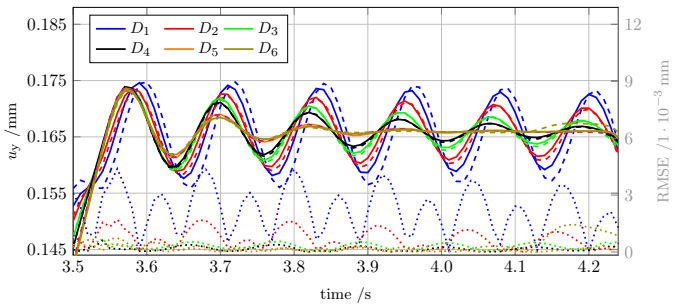

Fig. 12 Displacement of the cantilever beam's evaluation node based on FE calculations and the surrogate. The unseen damping values are $D = \{8, 19, 44, 70, 148, 173\} \cdot 10^{-3}$. Dashed lines refer to results of the surrogate, solid lines to FE results and dotted lines to RMSE

ration and 24 are from the oscillating configuration. In summary, there are 556,800 samples for training. The sequence length as well as the architecture of this model are identical to the one presented in Sect. 3.1 and described in Table 2.

For different damping coefficients, comparison of FE and surrogate results are shown in Fig. 11. All curves match with high accuracy again. The load case in this example is unknown, hence, not included in the training set. In contrast, the values of D are included in the training set.

It can be noted that the surrogate has a high generalization capability. For 100 evaluated unknown load curves, which total $3 \cdot 10^5$ predictions, a mean RMSE of $\epsilon_{\text{RMSE, mean}} = 5.6 \cdot 10^{-3}$mm is calculated. Thereby, values for the displacement are in the range $u = [-0.21, 0.21]$mm.

Apart from only using the force as an input variable for which generalization should be evaluated, also generalization of the categorical variable is investigated. This slightly contradicts the nature of a categorical variable, but following Powers and Xie [38], a categorical variable is a variable which can only be measured with a limited number of values. In most FE simulations, the damping value is constant over time. Hence, we identify the damping ratio as a categorical variable. Even though the damping ratio is a categorical quantity, it can have a wide range of values. Powers and Xie [38] call that *categorization of a continuous variable*. Consequently, interpolation between the values is valid, since D is still constant within a simulation and changes only over different simulations. Results of this investigation are presented in Fig. 12 with $D = \{8, 19, 44, 70, 148, 173\} \cdot 10^{-3}$. There, the detail in Fig. 11 is reproduced with unseen values for damping ratio D_3 and D_4. Displacements computed by the surrogate and FEM for D_3 and D_4 match with high accuracy. With a maximum RMSE of $\epsilon_{\text{RMSE, max}} = 0.8 \cdot 10^{-3}$mm, the error is similar to the error obtained by the trained damping ratios. Noteworthy, the displacement curves from the surrogate and FE cannot be distinguished in Fig. 12, since they overlay. Hence, the generalization ability of our surrogate between the trained damping values is shown.

Another important aspect is extrapolation. Due to the linearity of the investigated problems, extrapolation for the force input is not necessary, since this can be done by multiplying $z(t)$ directly with a factor after the output of the surrogate. However, this is infeasible for the damping and therefore, predictions with values for D lower and higher compared to the trained range, which are D_1, D_2 and D_5, D_6 in Fig. 12, are conducted.

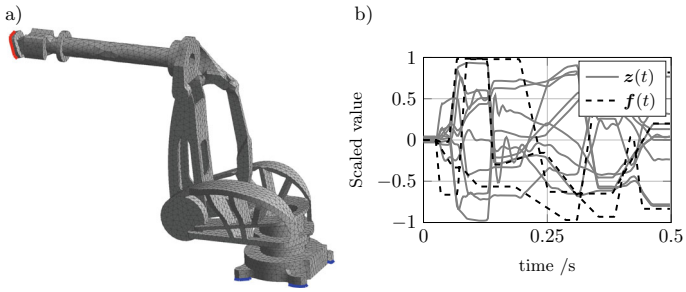

Fig. 13 Mesh of the robot with fixed support marked in blue on the robots base and position of the force, marked in red on the robots gripper mounting flange (**a**) and modal coordinates of the training dataset in ramped configuration (**b**)

For D_1 and D_2, the maximum RMSE is $\epsilon_{\text{RMSE, max}} = 4.4 \cdot 10^{-3}$mm. This comparatively high error is caused by a slight phase shift and an inexact reproduction of the amplitudes. The same observation can be made for the second lowest damping value. On the other hand, the oscillation frequency and the decaying of the curves match. Although for low-fidelity simulations these results may be sufficient, for high-fidelity simulations the deviations are too high. With a closer look to the damping values on the other end of the training data set, the extrapolation succeeds within a higher range of accuracy. For D_5, and predominantly for D_6, the maximum RMSE is $\epsilon_{\text{RMSE, max}} = 0.6 \cdot 10^{-3}$mm, hence, lower than for D_1 and D_2. It can be noted that the error is in a similar range as for the trained damping values. Starting from $t = 4.12$s results of D_6 differ with a maximum RMSE of $\epsilon_{\text{RMSE, max}} = 1.4 \cdot 10^{-3}$mm and therefore by twice the error. To summarize, extrapolation is possible without a loss of accuracy compared to the trained parameters. This applies to damping values greater than the trained ones under undamped decaying properties. This is a big advantage, since retraining of the model must not be conducted to extend flexibility of the surrogate for this specific variable. Still, we always recommend a critical verification of the model for the new range of input features.

Under the premise of sufficient accuracy for both examples, this leads to the finding that the history branch size for the surrogate presented in Sect. 3.1 could be decreased. Due to the missing variability for damping, complexity of the signals to learn for the example presented in Sect. 3.1 is less. Hence, less trainable parameters are needed for a surrogate with fixed damping. Consequently, this has a beneficial effect on the computational time in both the training and inference phases.

3.3 Robot

The results shown previously are of an academic nature. The proposed method of surrogate modeling is also suitable for more complex use cases. Therefore, a six-axis robot is evaluated, see Fig. 13a. The robot is simplified for the computation of displacements, since no gears are modeled and rigid connections of all arms are assumed. Furthermore, all parts are modeled with a linear elastic structural steel material model. The robot is discretized with 24102 linear tetrahedron elements, described by 7121 nodes. For modal reduction, the number of modes taken into account is set to be 10. The lowest eigenfrequency is 16 Hz, the highest $238\,Hz$. Damping parameters for the construction of the damping matrix $\boldsymbol{D}$ are $\alpha_{\text{damp}} = 12$ and $\beta_{\text{damp}} = 0.002$, hence, $D = 0.16$. Procedures for the computation of the datasets are identical to the cantilever beam example.

In this example, again ten eigenvectors are used for the modal reduction, since higher eigenfrequencies are not necessary to model the mounting flanges displacements. Hence, as architecture of the NN the same values as for the cantilever beam are used, see Table 2. The obligatory fixed time step of the surrogate is $\Delta t = 5 \cdot 10^{-4}$s, the sequence length is again $l_{\text{seq.}} = 50$. Nevertheless, this example has a higher complexity compared to the cantilever beam. In the cantilever beam example, displacements in y and z directions are identical for identical excitations. In the robot example, this is different. As shown in Fig. 13b, no modal coordinate does follow a force curve directly. In contrast, at the cantilever beam example, such a behavior of the modal coordinates can be observed, see Fig. 3.

The surrogate model of the robot is trained with 1000 epochs, see Fig. 14. Evaluating the test score, training can be finished after 850 epochs. In the following epochs, only the train score decreases. Hence, the model starts to overfit. The results shown in this section are generated with the model at epoch 850.

Evaluation of displacements is done on a node in the middle of the face, where the load is applied. Results from the surrogate model show high accuracy compared to results obtained with the standard modal

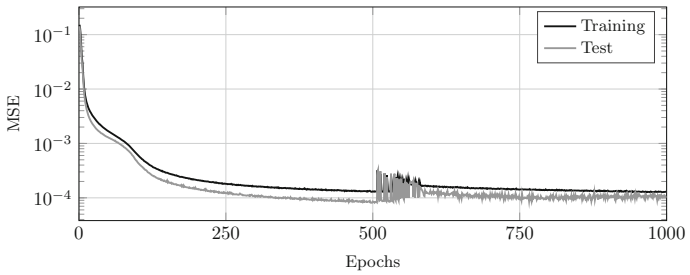

Fig. 14 Train and test score of the surrogate model in the robot example. Both scores start to increase again in the region of MTF between epoch 500 and 600. After epoch 850, the test scores does not decrease anymore

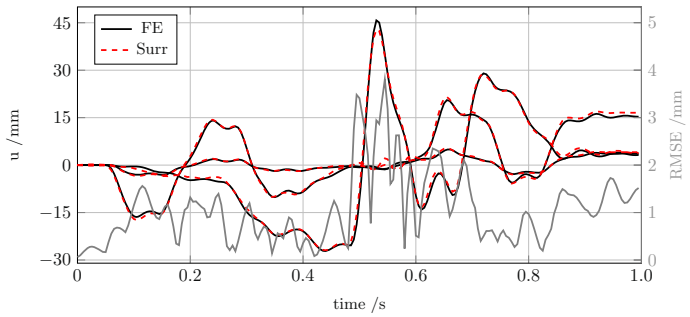

Fig. 15 Displacement of the robots mounting flange evaluation node based on FE calculations and the surrogate under a load case of the ramped configuration. Error is calculated as RMSE over the three displacements u_x, u_y and u_z. The FE and surrogate solution match accurately

Table 4 Duration for a simulation time of $t_{sim} = 1.0$s for the robot example

one-time computations		repeated computations			
$t_{\text{comp. eig.}}$	$t_{\text{training Surrogate}}$	t_{FOM}	$t_{\text{mod. red.}}$	$t_{\text{Surrogate}}$	$t_{\text{mod. red.}}/t_{\text{Surrogate}}$
5s	$3.7 \cdot 10^3$s	$2.1 \cdot 10^3$s	273s	6s	45

Reference simulations are computed with the transient structural ANSYS solver. For reduced-order solutions, 10 eigenvectors are used

reduction technique. Figure 15 shows displacement curves from the validation dataset, hence generalization within the models limitations can be stated. For 100 validation examples a RMSE of $\epsilon_{\text{RMSE, mean}} = 0.7$mm can be calculated. Values for the displacement are in the range $u = [-47, 48]$ mm. For this, the RMSE of the evaluated node is calculated and averaged over every timestep from all validation computations.

To emphasize the applicability of the proposed method for real-world problems, the training times were optimized by choosing a higher learning rate of $\alpha = 5 \cdot 10^{-4}$. Also the epochs in the NTF phase and the transition from TF to NTF while training are at a minimum for that specific problem, as MTF starts at epoch 500 for 100 epochs, see Fig. 14. Further, most important is the reduction of the sequences in the training data set. We use 82 simulations each of 0.5s, which leads to 16,300 sequences for training. Compared to the training sampled of the cantilever beam from Sect. 3.1, this is just a tenth. Due to that measures, a total training time of $t_{\text{training Surrogate}} = 3.7 \cdot 10^3$s can be achieved, see Table 4. The training and computation of the reference solutions is done with the same hardware as described in Sect. 3.1. As shown in Table 4, computations with the surrogate are faster compared to reference, which is the standard modal reduction solved by the ANSYS transient solver. A speed-up factor of 45 can be achieved. For both methods, a timestep of $\Delta t = 5 \cdot 10^{-4}$s is used. The timestep is sized on the maximal eigenfrequency taken into account.

To summarize, as for the academic examples, the proposed method for transient surrogate modeling of modally reduced structures applies also to practical examples. With this example, due to the higher number of initial degrees of freedom, one can benefit from an even higher speed-up factor. Furthermore, the streamlined training procedure of this example reduces the training effort to a minimum. The generation of the datasets takes $2.7 \cdot 10^3$s. Taking all that into account, amortization of the one-time computations for training is reached after 25 calculations.

4 Conclusion

In this study, we present an ANN surrogate for modeling discontinuously excited transient mechanical structures. As a mechanical reference, training and evaluation data are based on FE simulations. For this purpose, not the full discretized model is used. Moreover, dimensionality is reduced by modal reduction. The surrogate consists of several dense and LSTM layers, which reproduce the contribution vector from the modal reduction over time.

As in many machine learning models, the applicability of the model is restricted to mechanical systems with a specific load scenario. For the proposed method of surrogate modeling, where a complete system is considered, one is limited to geometry and material properties of the FE model being used for training. Furthermore, the characteristics of the force curves in the prediction phase have to be similar to the ones used during training. For example, step functions as force curves can not be modeled with our method, since they are not part of the training dataset.

The main contribution is the recursive architecture of the surrogate in this context. Predictions are done in a closed-loop manner, where the previous contribution vectors and only the force for the next timestep are needed. The history of the displacements of the mechanical system is stored in the input sequence, consisting of the previous contribution vectors. However, this leads to short-input sequences for the surrogate. In the state-of-the-art LSTM surrogates for transient problems, the length of the input sequence limits the dependency of the displacements to the exciting event in the input. Consequently, long input sequences are necessary to handle slowly decaying oscillations. Since the computational time for the LSTMs is in linear dependency to the input sequence length, our surrogate achieves high accuracy in transient load cases at low computational costs.

An additional contribution is the introduction of a categorical variable to the surrogate. Thereby, it is possible to reproduce the results of FE calculations in dependency of the damping ratio, allowing more flexibility to parameter studies. Moreover, this opens the way for the creation of surrogates with other categorical variables as material properties. High efficiency and accuracy of the surrogate are exemplified on a cantilever beam. Further, a comparison with state-of-the-art surrogates shows the advantages of the proposed surrogate on dynamic excitations.

Future investigations could focus on the integration of the presented surrogate into submodeling techniques by modifying the contribution vector with peripheral data before it is fed back to the input layer as part of the recursive architecture. Replacing the eigenvectors of the presented linear problems with a reduced representation of nonlinear systems obtained by an encoder–decoder system under shock loading could also be a matter of future work.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data availability The datasets generated during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors have no conflict of interest to declare that are relevant to the content of this article.

References

1. Lu, K., Zhang, K., Zhang, H., Gu, X., Jin, Y., Zhao, S., Fu, C., Yang, Y.: A review of model order reduction methods for large-scale structure systems. Shock Vibr. **2021**, 1–19 (2021). https://doi.org/10.1155/2021/6631180
2. Tong, G.G., Schiavazzi, D.E.: Data-driven synchronization-avoiding algorithms in the explicit distributed structural analysis of soft tissue. Comput. Mech. **71**(3), 453–479 (2023). https://doi.org/10.1007/s00466-022-02248-w

3. Kneifl, J., Rosin, D., Avci, O., Röhrle, O., Fehr, J.: Low-dimensional data-based surrogate model of a continuum-mechanical musculoskeletal system based on non-intrusive model order reduction. Arch. Appl. Mech. **93**(9), 3637–3663 (2023). https://doi.org/10.1007/s00419-023-02458-5
4. Funk, S., Airoud Basmaji, A., Nackenhorst, U.: Globally supported surrogate model based on support vector regression for nonlinear structural engineering applications. Arch. Appl. Mech. **93**(2), 825–839 (2023). https://doi.org/10.1007/s00419-023-02458-5
5. De, S., Deo, D., Sankaranarayanan, G., Arikatla, V.S.: A physics-driven neural networks-based simulation system (PhyNNeSS) for multimodal interactive virtual environments involving nonlinear deformable objects. Presence **20**(4), 289–308 (2011). https://doi.org/10.1162/pres_a_00054
6. Morooka, K., Chen, X., Kurazume, R., Uchida, S., Hara, K., Iwashita, Y., Hashizume, M.: Real-Time nonlinear FEM with neural network for simulating soft organ model deformation. In: MICCAI vol. 11, pp. 742–749. Springer, Berlin (2008)
7. Ordaz-Hernandez, K., Fischer, X., Bennis, F.: Model reduction technique for mechanical behaviour modelling: efficiency criteria and validity domain assessment. Proc. Inst. Mech. Eng., Part C **222**(3), 493–505 (2008). https://doi.org/10.1243/09544062JMES683
8. Mitusch, S.K., Funke, S.W., Kuchta, M.: Hybrid FEM-NN models: combining artificial neural networks with the finite element method. J. Comput. Phys. **446**, 110651 (2021). https://doi.org/10.1016/j.jcp.2021.110651
9. Runge, G., Wiese, M., Raatz, A.: FEM-based training of artificial neural networks for modular soft robots. In: IEEE Int. Conf. Robot., pp. 385–392 (2017). https://doi.org/10.1109/ROBIO.2017.8324448
10. Ribeiro, J.P.A., Tavares, S.M.O., Parente, M.: Stress-strain evaluation of structural parts using artificial neural networks. Proc. Inst. Mech. Eng. L: J. Mater. Des. Appl. **235**(6), 1271–1286 (2021). https://doi.org/10.1177/1464420721992445
11. Phellan, R., Hachem, B., Clin, J., Mac-Thiong, J.-M., Duong, L.: Real-time biomechanics using the finite element method and machine learning: review and perspective. Med. Phys. **48**(1), 7–18 (2021). https://doi.org/10.1002/mp.14602
12. Barbosa, M.P.F., Rade, D.A.: Kriging/FORM reliability analysis of rotor-bearing systems. J. Vib. Eng. Techno. (2022). https://doi.org/10.1007/s42417-022-00511-1
13. Kohar, C.P., Greve, L., Eller, T.K., Connolly, D.S., Inal, K.: A machine learning framework for accelerating the design process using CAE simulations: an application to finite element analysis in structural crashworthiness. Comput. Methods Appl. Mech. Eng. (2021). https://doi.org/10.1016/j.cma.2021.114008
14. Zhang, R., Liu, Y., Sun, H.: Physics-informed multi-lstm networks for metamodeling of nonlinear structures. Comput. Methods Appl. Mech. Eng. (2020). https://doi.org/10.1016/j.cma.2020.113226
15. Ma, Z.-S., Ding, Q., Zhai, Y.-J.: Hybrid modeling of nonlinear-jointed structures via finite-element model reduction and deep learning techniques. J. Vib. Eng. Techno. **9**(4), 575–585 (2021). https://doi.org/10.1007/s42417-020-00249-8
16. Baiges, J., Codina, R., Castañar, I., Castillo, E.: A finite element reduced-order model based on adaptive mesh refinement and artificial neural networks. Int. J. Numer. Methods Eng. **121**(4), 588–601 (2019). https://doi.org/10.1002/nme.6235
17. Tariverdi, A., Venkiteswaran, V.K., Richter, M., Elle, O.J., Tørresen, J., Mathiassen, K., Misra, S., Martinsen, Ø.G.: A recurrent neural-network-based real-time dynamic model for soft continuum manipulators. Front. Robot. AI (2021). https://doi.org/10.3389/frobt.2021.631303
18. Kharazmi, E., Wang, Z., Fan, D., Rudy, S., Sapsis, T., Triantafyllou, M.S., Karniadakis, G.E.: From Data to Assessment Models, Demonstrated through a Digital Twin of Marine Risers. In: OTC Offshore Technology Conference (2021). https://doi.org/10.4043/30985-MS
19. Koeppe, A., Bamer, F., Markert, B.: An efficient Monte Carlo strategy for elasto-plastic structures based on recurrent neural networks. Acta Mech. **230**(9), 3279–3293 (2019). https://doi.org/10.1007/s00707-019-02436-5
20. Simpson, T., Dervilis, N., Chatzi, E.: Machine learning approach to model order reduction of nonlinear systems via autoencoder and LSTM networks. J. Eng. Mech. (2021). https://doi.org/10.1061/(ASCE)EM.1943-7889.0001971
21. Simpson, T., Dervilis, N., Couturier, P., Maljaars, N., Chatzi, E.: Reduced order modeling of non-linear monopile dynamics via an AE-LSTM scheme. Front. Energy Res. (2023). https://doi.org/10.3389/fenrg.2023.1128201
22. Dutta, S., Rivera-Casillas, P., Styles, B., Farthing, M.W.: Reduced order modeling using advection-aware autoencoders. Math. Comput. Appl. (2022). https://doi.org/10.3390/mca27030034
23. Wriggers, P.: Nonlinear Finite Element Methods. Springer, Berlin (2008)
24. Bathe, K.-J.: Finite Element Procedures. Prentice-Hall of India, New Delhi (2007)
25. Wagner, M.: Lineare und Nichtlineare FEM: Eine Einführung Mit Anwendungen in der Umformsimulation mit LS-DYNA, 3rd edn. Springer, Wiesbaden (2022)
26. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
27. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016). https://doi.org/10.1007/s10710-017-9314-z
28. Koeppe, A.: Deep learning in the finite element method. Dissertation, RWTH Aachen, Aachen (2021). https://doi.org/10.18154/RWTH-2021-04990
29. Neil, D., Pfeiffer, M., Liu, S.-C.: Phased LSTM: Accelerating Recurrent Network Training for Long or Event-Based Sequences. In: Proceedings of the 30th NeurIPS. NIPS'16, pp. 3889–3897. Curran Associates Inc., Red Hook, NY, USA (2016)
30. Sonneville, V., Scapolan, M., Shan, M., Bauchau, O.A.: Modal reduction procedures for flexible multibody dynamics. Multibody Sys.Dyn. **51**(4), 377–418 (2021). https://doi.org/10.1007/s11044-020-09770-w
31. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. International Conference on Learning Representations (2014)
32. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. JMLR Proceedings, vol. 9, pp. 249–256. PMLR, Chia Laguna Resort, Sardinia, Italy (2010)

33. Goyal, A., Lamb, A., Zhang, Y., Zhang, S., Courville, A., Bengio, Y.: Professor Forcing: A New Algorithm for Training Recurrent Networks. In: Proceedings of the 30th NeurIPS. NIPS'16, pp. 4608–4616. Curran Associates Inc., Red Hook, NY, USA (2016). abc
34. Schmidt, F.: Generalization in Generation: A closer look at Exposure Bias. In: 3rd Workshop on NGT, pp. 157–167 (2019). https://doi.org/10.18653/v1/D19-5616
35. Bengio, S., Vinyals, O., Jaitly, N., Shazeer, N.: Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks. In: Proceedings of the 28th NeurIPS. NIPS'15, pp. 1171–1179. MIT Press, Cambridge, MA, USA (2015)
36. Smith, L.N.: Cyclical Learning Rates for Training Neural Networks. In: IEEE Winter Conference on Applications, pp. 464–472 (2017). https://doi.org/10.1109/WACV.2017.58
37. Pascanu, R., Mikolov, T., Bengio, Y.: On the Difficulty of Training Recurrent Neural Networks. In: Proceedings of the 30th ICML - Volume 28. ICML'13, pp. 1310–1318. JMLR.org, Atlanta, GA, USA (2013)
38. Powers, D.A., Xie, Y.: Statistical Methods for Categorical Data Analysis, 2nd edn. Academic Press, Bingley (2008)