DOI: 10.61782/fa.2023.0131

forum acusticum 2023

A DATA-BASED METHOD ENHANCING A PARAMETRICALLY MODEL ORDER REDUCED FINITE ELEMENT MODEL OF A CLASSICAL GUITAR

Pierfrancesco Cillo[1*] **Alexander Brauchler**[1] **Sebastian Gonzalez**[2] **Pascal Ziegler**[1]
Fabio Antonacci[2] **Augusto Sarti**[2] **Peter Eberhard**[1]
[1] Institute of Engineering and Computational Mechanics, University of Stuttgart, Germany
[2] Department of Electronics, Information and Bioengineering, Politecnico di Milano, Italy

ABSTRACT

Recently developed high-fidelity finite element (FE) models prove to be a state-of-the-art method for a better understanding of the vibrational behavior of musical instruments. However, some kinds of analyses, like optimization or parameter identification, require numerous model evaluations and lead to long computational times when using the FE model. Projection-based parametric model order reduction is a powerful tool to improve the computational time of FE models while preserving the parameter dependence. Despite this, projection-based methods require the complete system matrices that are often only accessible with limitations. Consequently, the reduced-order model produces a systematical discrepancy compared to the original model. We present a discrepancy modeling method to approximate the parameter-dependent effect of a radiating boundary condition in an FE model of a classical guitar that cannot be exported from the commercial FE software Abaqus. For this purpose, a projection-based reduced-order model is enhanced by a data-driven model of the error in the approximation of the eigenfrequencies and eigenmodes. Artificial neural networks account for the data-driven discrepancy models. The application of the discrepancy models obliterates 98% of the average eigenfrequency error existing in the initial reduced-order model without error estimation.

*__Corresponding author__:
pierfrancesco.cillo@itm.uni-stuttgart.de.

Keywords: guitar, data-driven model, virtual prototype, discrepancy modeling, model order reduction

1. INTRODUCTION

In recent years, the finite element method (FEM) has become one of the most widely used numerical models for studying the vibrational behavior of guitars. Recent studies have demonstrated its effectiveness [1]. Nonetheless, certain types of analyses, such as parameter identification or optimization, see [2], need numerous model evaluations and result in long computational time when using FEM.

Projection-based parametric model order reduction (PMOR) is a technique used to reduce the computational time of FEM simulations while preserving the parameter dependence of the model. This technique involves projecting the original high-dimensional model onto a lower-dimensional subspace that captures the dominant features of the model, see [3].

Projection-based methods require access to the complete system matrices. When using some commercial finite element software, this can constitute a limitation. In particular, when using the commercial finite element software Abaqus [4] it is not possible e.g. to export the full system matrices if the finite element model includes certain features, such as acoustic structural coupling or radiating boundary conditions (BCs).

To overcome this issue, the finite element model needs to undergo some minor adaptations before model order reduction can be applied. The implementation of these adaptations combined with the PMOR procedure inevitably leads to a systematical discrepancy between the reduced-order model and the original one.

This contribution, which builds on [2], aims to develop a method that can obliterate the error between a full-order model of a classical guitar and a respective reduced-order model where some features have not been exported. This is done by modeling the discrepancy between reduced-order and full-order models.

A data-driven model able to approximate the parameter-dependent error term between the eigenfrequencies and the eigenmodes of the reduced-order and the full-order models is developed. A recently developed method called discrepancy modeling, or closure modeling, is used, which employs a deep neural network as a closure learning framework for reduced-order systems [5].

This work deals with a combined approach between model order reduction and artificial neural networks (NNs). The latter have shown to accurately predict the vibrational behavior of musical instruments [6]. By combining both methods one can obtain a much more accurate prediction than using either alone.

2. MODEL DESCRIPTION

In this section, a classical guitar finite element model is presented, and its details are described. We will refer to it as full-order model. Projection-based PMOR is applied to the full-order model, producing a reduced-order, or surrogate model.

2.1 Full-Order Model

The geometry modeling and the finite element discretization are realized using the commercial software Abaqus, which allows to define the material characteristics of the model, to impose acoustic and structural BCs, and to automatically create the mesh.

Since we do not want to study a particular instrument, but show the feasibility of the method, we use a simplified model of the guitar. Details about the model can be found in [7].

The model is composed of three parts: a plane top plate or soundboard with a sound hole; a plane back plate with the same shape as the top plate; and the air inside the cavity, of the shape and size of a classical guitar. The top and back plates' materials were chosen to be, respectively, cedar and mahogany. Figure 1 shows the geometry of the guitar model after the three parts have been meshed.

The fluid-structure interaction between the plates and the air cavity is taken into account by applying a tie constraint between the surface of each plate with the corresponding underlying surface of the air cavity. This tie constraint allows to couple the structural degrees of freedom (DOFs) of the plates to the pressure DOFs of the fluid.

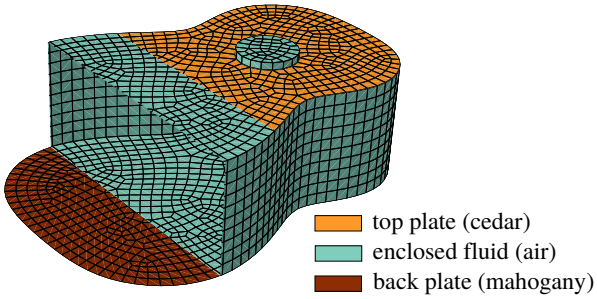

Figure 1: Assembled and meshed finite element model of a classical guitar with different sections for the different parts.

Instead of modeling the actual sides of the guitar, a homogeneous Dirichlet BC is imposed on the edges of both the top and the back plates. According to [8], the air cavity has been modified with a length correction in correspondence to the sound hole, as can be observed in Figure 1. This allows to take into account the effects due to the external air. On the surface of the sound hole, acoustic infinite elements are applied to simulate the sound radiation in that specific location as if the surrounding environment were infinitely large. The model results in $N = 19908$ total DOFs.

The woods of the guitar plates can be considered as orthotropic materials [9], so their behavior can be characterized by ten material parameters for each plate, namely, the density ρ, three Young's moduli E_{L}, E_{T} and E_{R}, three Poisson ratios ν_{LT}, ν_{LR} and ν_{TR}, and three shear moduli G_{LT}, G_{LR} and G_{TR}. Thus, we parameterize the finite element model with 20 material parameters i.e. 10 for each plate. The subscripts L, T, and R denote the longitudinal, tangential, and radial directions with respect to the wood growth rings. The material parameters are considered homogeneous throughout the plates, as done in [10].

The nominal parameter values, taken from [11], are listed in Table 1. The air cavity is characterized by a density $\rho_{\mathrm{F}} = 1.2$ kg/m^3 and a bulk modulus $K_{\mathrm{F}} = 142$ kPa.

The final system of equations of motion for the full-order model reads

$$\boldsymbol{M}_{\mathrm{FE}}(\hat{\boldsymbol{p}})\ddot{\boldsymbol{q}} + \boldsymbol{D}_{\mathrm{FE}}(\hat{\boldsymbol{p}})\dot{\boldsymbol{q}} + \boldsymbol{K}_{\mathrm{FE}}(\hat{\boldsymbol{p}})\boldsymbol{q} = \boldsymbol{f}, \qquad (1)$$

where $\boldsymbol{M}_{\mathrm{FE}} \in \mathbb{R}^{N\times N}$, $\boldsymbol{D}_{\mathrm{FE}} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{K}_{\mathrm{FE}} \in \mathbb{R}^{N\times N}$ are the mass, damping, and stiffness matrices of

Table 1: Nominal material parameter values for cedar and mahogany, from [11].

	ρ [kg/m^3]	E_L [GPa]	E_R [GPa]	E_T [GPa]	ν_LR
cedar	320	8.47	0.686	0.466	0.378
mahogany	420	10.7	1.18	0.534	0.297
	ν_LT	ν_RT	G_LR [GPa]	G_LT [GPa]	G_RT [GPa]
cedar	0.296	0.403	0.737	0.728	0.042
mahogany	0.641	0.264	0.939	0.630	0.224

the full-order model. The vector $\boldsymbol{f} \in \mathbb{R}^N$ represents the external forces acting on the system, while $\boldsymbol{q} \in \mathbb{R}^N$ contains the displacements of the N nodal DOFs. The models' parameter dependency is highlighted by the array $\hat{\boldsymbol{p}} \in \mathbb{R}^{20}$ containing the material parameters of both plates.

It should be kept in mind that Abaqus can only evaluate discrete values of $\hat{\boldsymbol{p}}$, hence an analytical parametric version of the system matrices does not exist. Also, when coupling the structure and the fluid, the system matrices are not accessible for export in Abaqus.

The eigenfrequencies and eigenmodes of the system, as a function of the material parameters, are computed solving the eigenvalue problem

$$(\boldsymbol{K}_\mathrm{FE}(\hat{\boldsymbol{p}}) - \omega_m^2 \boldsymbol{M}_\mathrm{FE}(\hat{\boldsymbol{p}}))\boldsymbol{\phi}_m = \boldsymbol{0}, \tag{2}$$

with ω_m and $\boldsymbol{\phi}_m$ being the m-th eigenfrequency and eigenmode of the system, respectively.

The employment of acoustic infinite elements allows to account for stiffness and mass contribution in the extraction of the eigenfrequencies and eigenmodes, while the damping effects are neglected [4]. However, a previous study [10] demonstrates how the employment of acoustic infinite elements yields results consistent with experimental data.

The eigenfrequencies and eigenvectors were computed with the Abaqus embedded eigenvalue solver. To compute the first 50 eigenfrequencies and eigenmodes with the full-order model, the computational time turns out to be about 14 s on a workstation with an AMD Ryzen 9 5950X 16-Core Processor and 128 GB of RAM.

2.2 Surrogate Model

The full-order model undergoes a projection-based model order reduction, obtaining as a result a surrogate model. However, the projection-based model order reduction relies on the analytical parametric version of the system matrices, not available from Abaqus. Moreover, also the system matrices at discrete parameter values are not available. Hence, a few preliminary steps are necessary.

One way to obtain accessible system matrices is to substitute the acoustic infinite elements at the sound hole with regular acoustic elements, to which we apply a Dirichlet BC, i.e. a pressure of $p = 0$ on the entire sound hole surface. Furthermore, the system matrices including the acoustic structural coupling are unavailable. Thus, the matrices of the structural and acoustical parts are exported separately, and the fluid-structure interaction is computed a posteriori, as described in [7]. Starting from the system thus obtained, affinely parameter-dependent (APD) system matrices are retrieved [2].

The resulting equations of motion, written as a second-order input-output system, read

$$\begin{aligned} \boldsymbol{M}_\mathrm{apd}(\boldsymbol{p})\ddot{\boldsymbol{q}} + \boldsymbol{D}_\mathrm{apd}(\boldsymbol{p})\dot{\boldsymbol{q}} + \boldsymbol{K}_\mathrm{apd}(\boldsymbol{p})\boldsymbol{q} &= \boldsymbol{B}\boldsymbol{u}, \\ \boldsymbol{y} &= \boldsymbol{C}\boldsymbol{q}, \end{aligned} \tag{3}$$

where $\boldsymbol{M}_\mathrm{apd}, \boldsymbol{D}_\mathrm{apd}, \boldsymbol{K}_\mathrm{apd} \in \mathbb{R}^{N\times N}$ are the parametric system matrices, obtained after the third step. The system inputs $\boldsymbol{u} \in \mathbb{R}^k$ are distributed on the nodal DOFs via the input matrix $\boldsymbol{B} \in \mathbb{R}^{N\times k}$. The desired system output points contained in the vector $\boldsymbol{y} \in \mathbb{R}^j$ are retrieved via the output matrix $\boldsymbol{C} \in \mathbb{R}^{j\times N}$. The parameter dependency is represented by the variable vector $\boldsymbol{p} \in \mathbb{R}^{20}$.

The core idea of model order reduction is to obtain a reduced vector of DOFs $\boldsymbol{q}_\mathrm{r} \in \mathbb{R}^n$ from which it is possible to retrieve a very good approximation of the full-order solution $\boldsymbol{q}$ by back-projecting it using a projection matrix $\boldsymbol{V} \in \mathbb{R}^{N\times n}$. This means that

$$\boldsymbol{q} \approx \boldsymbol{V}\boldsymbol{q}_\mathrm{r} \tag{4}$$

must hold for $n \ll N$. By substituting Equation (4) into Equation (3), we obtain

$$\begin{aligned} \boldsymbol{M}_\mathrm{apd}(\boldsymbol{p})\boldsymbol{V}\ddot{\boldsymbol{q}}_\mathrm{r} + \boldsymbol{D}_\mathrm{apd}(\boldsymbol{p})\boldsymbol{V}\dot{\boldsymbol{q}}_\mathrm{r} \\ + \boldsymbol{K}_\mathrm{apd}(\boldsymbol{p})\boldsymbol{V}\boldsymbol{q}_\mathrm{r} = \boldsymbol{B}\boldsymbol{u} + \boldsymbol{\epsilon}, \end{aligned} \tag{5}$$

where the term $\boldsymbol{\epsilon} \in \mathbb{R}^N$ represents the residual from the approximation. It is possible to get rid of this term by left-multiplying the system of equations with the transpose of another projection matrix $\boldsymbol{W} \in \mathbb{R}^{N\times n}$, where the rows of $\boldsymbol{W}^\mathrm{T}$ are orthogonal to the residual term $\boldsymbol{\epsilon}$, such that

$\boldsymbol{W}^{\mathrm{T}}\boldsymbol{\epsilon} = \boldsymbol{0}$. The resulting system reads

$$\begin{aligned} &\underbrace{\boldsymbol{W}^{\mathrm{T}}\boldsymbol{M}_{\mathrm{apd}}(\boldsymbol{p})\boldsymbol{V}}_{\boldsymbol{M}_{\mathrm{r}}(\boldsymbol{p})}\ddot{\boldsymbol{q}}_{\mathrm{r}} + \underbrace{\boldsymbol{W}^{\mathrm{T}}\boldsymbol{D}_{\mathrm{apd}}(\boldsymbol{p})\boldsymbol{V}}_{\boldsymbol{D}_{\mathrm{r}}(\boldsymbol{p})}\dot{\boldsymbol{q}}_{\mathrm{r}} \\ &+ \underbrace{\boldsymbol{W}^{\mathrm{T}}\boldsymbol{K}_{\mathrm{apd}}(\boldsymbol{p})\boldsymbol{V}}_{\boldsymbol{K}_{\mathrm{r}}(\boldsymbol{p})}\boldsymbol{q}_{\mathrm{r}} = \underbrace{\boldsymbol{W}^{\mathrm{T}}\boldsymbol{B}}_{\boldsymbol{B}_{\mathrm{r}}}\boldsymbol{u} + \underbrace{\boldsymbol{W}^{\mathrm{T}}\boldsymbol{\epsilon}}_{\boldsymbol{0}}, \\ &\boldsymbol{y}_{\mathrm{r}} = \underbrace{\boldsymbol{C}\boldsymbol{V}}_{\boldsymbol{C}_{\mathrm{r}}}\boldsymbol{q}_{\mathrm{r}}. \end{aligned} \tag{6}$$

The matrices $\boldsymbol{M}_{\mathrm{r}} \in \mathbb{R}^{n\times n}$, $\boldsymbol{D}_{\mathrm{r}} \in \mathbb{R}^{n\times n}$, $\boldsymbol{K}_{\mathrm{r}} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B}_{\mathrm{r}} \in \mathbb{R}^{n\times k}$ and $\boldsymbol{C}_{\mathrm{r}} \in \mathbb{R}^{j\times n}$ represent the reduced-order mass, damping, stiffness, input, and output matrices.

The identification of appropriate bases $\boldsymbol{V}$ and $\boldsymbol{W}^{\mathrm{T}}$ with $n \ll N$ is needed such that the original system is well approximated. A worthwhile choice is to find appropriate bases with the so-called moment-matching methods [12]. For a detailed description of the method used to compute the projection matrices and the order n of the reduced system please refer to [2].

All of the previously described steps imply a further approximation from the full-order model we start with, meaning that the discrepancy between the final surrogate model and the full-order model will be caused by the piling-up of the errors of each approximation step.

A surrogate model is computed, which only keeps six material parameter variables, namely three for each plate. We chose the density ρ and the longitudinal Young's modulus E_{L} as they have a high influence on the eigenmodes [2], and one with lower influence, i.e. longitudinal-tangential shear modulus G_{LT}. The other parameters are fixed to their nominal values. The final reduced-order system has $n = 2471$ DOFs, which is a number significantly smaller than the $N = 19908$ DOFs of the full-order model.

The eigenfrequencies $\omega_{\mathrm{r},m}$ and eigenmodes $\boldsymbol{\phi}_{\mathrm{r},m}$ of the reduced-order model are found solving the eigenvalue problem

$$(\boldsymbol{K}_{\mathrm{r}}(\boldsymbol{p}) - \omega_{\mathrm{r},m}^2\boldsymbol{M}_{\mathrm{r}}(\boldsymbol{p}))\boldsymbol{\phi}_{\mathrm{r},m} = \boldsymbol{0}. \tag{7}$$

On the same workstation mentioned in the previous section, the computational time for the first 50 eigenfrequencies and eigenmodes turns out to be about 1.7 s, which is more than eight times faster than the computation for the full-order model.

3. DISCREPANCY MODELING

In this section, a data-based approach to learn the discrepancy between the full-order and the surrogate models in the eigenfrequencies and eigenmodes is proposed. In what follows, two distinct discrepancy models are developed based on NNs: one for the eigenfrequencies and one for the eigenmodes.

To train the NNs, we generate a dataset by solving the eigenvalue problems (2) and (7) for 1000 different parameter configurations, computing the first 50 eigenfrequencies and eigenmodes. The configurations have been computed using a quasi-random Sobol sequence bounded between $\pm 30\%$ of the nominal values. The hyperparameters of the two networks have been tuned independently, based on a random hyperparameter search, using the mean squared error on the dataset as performance criteria.

3.1 Eigenfrequencies Correction

The parameter-dependent difference between the eigenfrequencies of the two models is written as

$$g_m(\boldsymbol{p}) = \omega_m(\boldsymbol{p}) - \omega_{\mathrm{r},m}(\boldsymbol{p}), \tag{8}$$

and a function $\tilde{g}_m(\boldsymbol{p})$ is searched, for each mode m, that approximates the parameter dependent eigenfrequency error $g_m(\boldsymbol{p})$. This allows to compute an approximation of the eigenfrequencies of the reduced-order model as

$$\tilde{\omega}_m(\boldsymbol{p}) = \omega_{\mathrm{r},m}(\boldsymbol{p}) + \tilde{g}_m(\boldsymbol{p}) \approx \omega_m(\boldsymbol{p}). \tag{9}$$

In order to model $\tilde{g}_m$, we use a fully-connected multilayer perceptron with two hidden layers of dimension $\mathbb{R}^{10}$, the input layer containing the parameter vector $\boldsymbol{p} \in \mathbb{R}^6$ and the output layer containing the approximated eigenfrequency error $\tilde{g}_m \in \mathbb{R}$. The activation function of the inner layers of the proposed network is a Rectified Linear Unit function, while the output layer has a linear activation function. The back propagation during the training is implemented through a Broyden-Fletcher-Goldfarb-Shanno quasi-Newton algorithm. One NN is trained for each mode m over the training set. The training finishes after 1000 iterations to avoid overfitting. Eighty percent of the data is used for training the networks, while the remaining twenty percent is provisioned as test set.

Finally, we want to compare the performance of the proposed method with the sole employment of NN to predict the parameter-dependent eigenfrequencies. To do so, we train a NN with the same structure as the one previously described, where the output layer contains ω_m.

3.2 Eigenmodes Correction

We consider the eigenmodes error term as

$$\boldsymbol{\Delta}_\mu = \bar{\boldsymbol{\phi}}_\mu(\boldsymbol{p}) - \bar{\boldsymbol{\phi}}_{\mathrm{r},\mu}(\boldsymbol{p}), \tag{10}$$

where $\bar{\boldsymbol{\phi}}_\mu$ and $\bar{\boldsymbol{\phi}}_{\mathrm{r},\mu}$ represent the μ-th normalized eigenmodes. Our aim, now, is to find a parameter-dependent function $\tilde{\boldsymbol{\Delta}}_\mu(\boldsymbol{p})$ that approximates the discrepancy between the eigenmodes, in order to compute a corrected version of the reduced-order eigenmodes, as

$$\tilde{\boldsymbol{\phi}}_\mu(\boldsymbol{p}) = \bar{\boldsymbol{\phi}}_{\mathrm{r},\mu}(\boldsymbol{p}) + \tilde{\boldsymbol{\Delta}}_\mu(\boldsymbol{p}) \approx \bar{\boldsymbol{\phi}}_\mu(\boldsymbol{p}). \tag{11}$$

3.2.1 Particle Swarm Optimization

Long computational time can incur when training a NN if we consider all the structural nodal displacements of the modeshapes. Thus, a criterion to choose a subset of nodal displacements upon which to compare the modeshapes of the models is needed. To do this, we need to determine a number D of nodal displacements which adequately approximate the relationship between the modeshapes of the full-order and the reduced-order model. We will consider the surface node displacements only (displacement in the direction normal to the plates of the guitar) since these are more accessible for possible experimental measurements. The modeshapes are compared using the Modal Assurance Criterion (MAC) [13].

We search for the minimum error between the diagonals of two different MAC matrices: the first is the MAC between the modeshapes of the two models computed on all the structural nodal displacements; while for the second we only consider the surface displacement of D nodes. To find an optimal set of nodes, we use an optimization algorithm based on Particle Swarm Optimization (PSO). The basic idea of PSO is to represent candidate designs to an optimization problem as particles that move through a search space. For further details about the algorithm, please refer to [14].

In our specific case, the search space corresponds to the array containing all the surface nodes. The positions of the particles correspond to D indices of the aforementioned array. The position and the velocity of the particles are updated following the goal of minimizing a given objective function $J(\boldsymbol{x})$, which will be computed for each time step, where $\boldsymbol{x}$ is the array containing the position of all the particles. The objective function

$$J(\boldsymbol{x}) = \sqrt{\sum_{\mu=1}^{50} \left| \frac{\mathrm{MAC}_{\mu\mu} - \mathrm{MAC}(\boldsymbol{x})_{\mu\mu}}{\mathrm{MAC}_{\mu\mu}} \right|^2} \tag{12}$$

is proposed, where $\mathrm{MAC}_{\mu\mu}$ represents the diagonal elements of the MAC matrix considering all the structural DOFs, and $\mathrm{MAC}(\boldsymbol{x})_{\mu\mu}$ represents the MAC matrix considering the D surface node displacements at position $\boldsymbol{x}$.

We performed the computation considering the modeshapes of the full-order and surrogate model with the material parameter fixed to their nominal values. We perform the computation on different numbers of points, in a range between 100 and 200, and we choose the solution with the lower value of the objective function, i.e. $D = 160$. The position of the nodes, found from the PSO on the top and back plate, is shown in Figure 2.

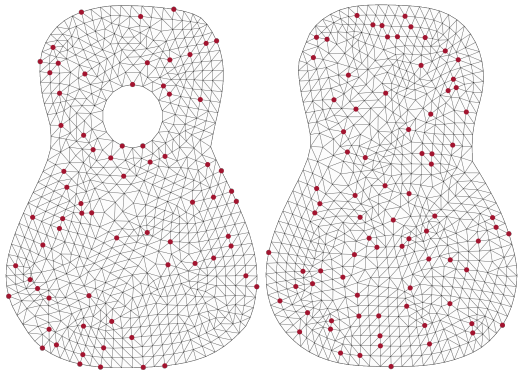

Figure 2: Position of the nodes on top and back found through the Particle Swarm Optimization algorithm. These nodes are used for comparing the modeshapes

3.2.2 Dictionary of Modeshapes

The eigenmodes change continuously with the parameters, and moreover, they appear and disappear in certain parameter ranges. Therefore, we need a method to classify the modeshapes, investigating which regions of the parameter space they appear. To achieve this, we built a so-called *dictionary* of modeshapes. All the modeshapes of the dataset are compared using the MAC, and they are clustered in groups of similar modeshapes. The dictionary items are initialized with the modeshapes sequence with parameters fixed to their nominal values. All the modeshapes of the dataset are compared, one by one, to the items of the dictionary. If MAC>0.8, the modeshape and its correspondent parameter configuration are stored. If not, a new record is created and the modeshape is added to the new item. Each record constitutes a category to which a modeshape can belong. Using this approach,

our aim now is to learn a regression model for each dictionary item, and not for each mode number as we did for the eigenfrequencies. Thus, the symbol μ in Equations (10) and (11) does not represent the mode number, as for the eigenfrequencies, but it represents the dictionary item number.

3.2.3 Network Architecture

As a learning method, a feedforward NN has been used. The input layer of the network has dimension $\mathbb{R}^6$ and contains the parameter array $\boldsymbol{p}$. Four hidden layers are present. The first two have dimension $\mathbb{R}^{20}$, while the last two have dimension $\mathbb{R}^{160}$. The output layer contains the approximated error for one dictionary item, of dimension $\mathbb{R}^{160}$. Each neuron of the first three hidden layers is characterized by a hyperbolic tangent transfer function. The neurons of the last hidden layer are characterized by a linear transfer function.

The first three hidden layers of the NN are fully connected. Between the two large hidden layers of dimension $\mathbb{R}^{160}$, the weights will not be learned. Instead, they are predefined in a neighborhood relation that can be written in matrix form with the entries $w_{ji}(\boldsymbol{x}_j, \boldsymbol{x}_i) = 0.9\left(1 - \frac{||\boldsymbol{x}_j - \boldsymbol{x}_i||_2}{\delta}\right) + 0.1$, if $||\boldsymbol{x}_j - \boldsymbol{x}_i||_2 \leq \delta$, where $||\boldsymbol{x}_j - \boldsymbol{x}_i||_2$ is the Euclidean distance between coordinates of two nodes $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ of the mesh, and $\delta = 5$ cm represents the threshold distance. Nodes within the threshold distance from each other receive weights that increase linearly from 0.1 to 1 as the Euclidean distance between them decreases. Nodes outside the threshold distance will have weight $w_{ji}(\boldsymbol{x}_j, \boldsymbol{x}_i) = 0$. The use of predefined weights grants a significant reduction in training time.

We train one NN for each dictionary item using the Levenberg-Marquardt backpropagation algorithm. In contrast to the NN employed for the eigenfrequencies, this algorithm has been chosen because of the larger NN size, due to its fast convergence feature [15].

The layer weights are initialized randomly, and the training finishes after 100 iterations. All the training runs converged and their error plateaued. Eighty percent of the data is used for training the networks, while the remaining twenty percent is used as test set.

4. RESULTS

On the test set, the discrepancy modeling correction to the eigenfrequencies is computed using Equation (9). We calculate the relative eigenfrequency error ε_m as

$$\varepsilon_m = \frac{\omega_m - \omega^*}{\omega_m}, \tag{13}$$

where $\omega^* = \omega_{\mathrm{r},m}$ in case no correction is applied, or $\omega^* = \tilde{\omega}_m$ if the eigenfrequencies of the surrogate model are corrected via discrepancy modeling, using the NNs we introduced in Section 3.1.

The results are illustrated using box plots. Each box contains the data between the 25$^{\mathrm{th}}$ and the 75$^{\mathrm{th}}$ percentile, with a central mark representing the median. The lines going out from the edge of the box, the so-called whiskers, extend to a distance that is 1.5 the interquartile range, i.e. the width of the box. Points lying at a greater distance are considered outliers, which are represented as scattered points.

Figure 3(a) shows the box plots of the relative eigenfrequency error for the first 10 modes, both before and after the eigenfrequency correction. The results after the correction from the prediction show a significant improvement. After the correction, the relative error for the considered eigenfrequencies is bounded $-0.65\% < \varepsilon < 0.65\%$, with an average value of 0.053%. Since the average value of the eigenfrequencies relative error before the correction is 2.88%, the discrepancy modeling is able to erase 98% of the average error. The eigenfrequencies of the reduced-order model have a systematically lower value than the ones of the full-order model, and the discrepancy modeling method here proposed is able to shift the median value to zero, reducing the variance as well. The relative error of the first eigenfrequency before the correction is by far the largest, with an average value of -21.35%. After the correction, it decreases to 0.0043%. The modes 7 to 10, before the correction, have an absolute relative error of less than 0.1%, which can be considered sufficiently low already. Regardless, after the correction via discrepancy modeling, it is further reduced.

The performance is compared with the results from the method employing the sole NN described in Section 3.1. Assessing the NN on the test set, we find that the average relative error over the first 10 eigenfrequencies is 1.19%. The method combining PMOR and NN can produce results that are over 20 times more accurate.

The performance assessment of the modeshape discrepancy modeling is not straightforward. Therefore, Figure 4 shows a visual example of a modeshape correction, for the 9$^{\mathrm{th}}$ dictionary eigenmode of a specific parameter configuration $\hat{\boldsymbol{p}}$. In this example, it is visible how the formerly erroneous modeshape from the reduced-

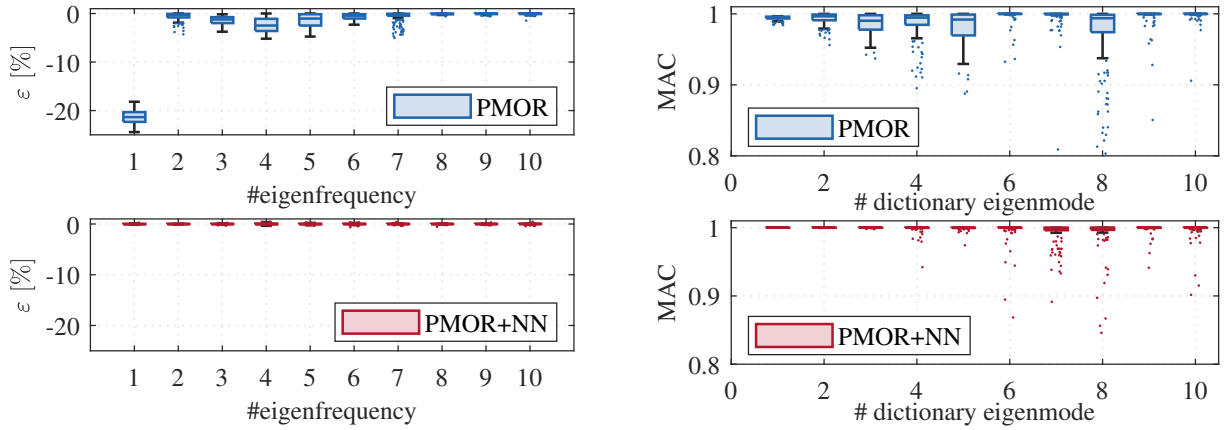

Figure 3: Box plots representing (a) the relative eigenfrequency errors for the first 10 modes, both before (up) and after (down) the discrepancy model correction, and (b) the MAC values between full-order and surrogate model, before (up) and after (down) the discrepancy model correction, for the first 10 dictionary modes.

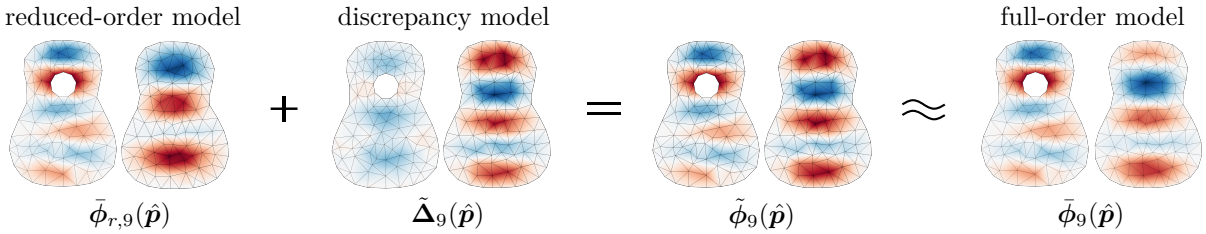

Figure 4: Visual example of modeshape correction for the 9th dictionary eigenmode of a specific parameter configuration $\hat{p}$.

order model is corrected through the discrepancy model, to form a final modeshape which is, then, much more similar to the full-order model's modeshape. The MAC value increases from 0.94 before the correction to 0.99 after the correction.

Figure 3(b) shows the distribution of the MAC values between the full-order and reduced-order model, both before and after the discrepancy model correction for the first 10 dictionary modes computed on the test set. The data are shown as box plots. The modeshapes of the dictionary items 6, 7, 9 and 10 are already well approximated, with an average MAC$>$ 0.99, so the discrepancy model correction does not bring a significant improvement. Indeed, for items 6, 7 and 10 the corrected model is even slightly worse with a higher number of outliers. For all the other dictionary items, where the distribution of the MAC value was not already concentrated around one, the correction results bring an overall improvement. After the correction, their average value is 1.00 for items 1 to 5 and 0.95 for item 8. Their variance is reduced by 17% on average.

5. CONCLUSIONS

We developed a data-driven method for an enhanced reduced-order model of a classical guitar by modeling the discrepancy between the full-order and reduced-order models. We found that this discrepancy modeling significantly reduced the error for the first 50 eigenfrequencies, especially beneficial for the lower order modes that are crucial for the guitar's sound characterization [16]. It offers a substantial improvement to the surrogate model without any noticeable downside.

However, when applied to eigenmodes, the method did not yield a universal improvement but rather benefited certain modeshapes. Therefore, discretion is advised in its application, considering the specific modeshapes to apply it to. Despite its effectiveness, there's room for further refinement, potentially through an improved dictionary assembly procedure that could employ a classification learning model for assembling the clusters.

In this work, we choose to apply the discrepancy modeling on the parameter-dependent eigenfrequencies and eigenmodes of the full-order and surrogate models. Yet a different way of proceeding could be to apply the discrepancy modeling method directly to the models' mass and stiffness matrices, with the purpose of learning the missing terms in the affine parametric matrices.

We wish to extend the application of our method to a fully-detailed finite element guitar model, as the one developed in [10]. This would result in better-approximated efficient models, which might enhance the knowledge about existing instruments.

6. ACKNOWLEDGMENTS

This research was partially supported by the German Research Foundation DFG (Project No. 455440338). The authors would like to thank Ingeborg Wenger for the valuable discussions and thoughtful remarks, especially regarding the application of artificial neural networks.

7. REFERENCES

[1] E. Kaselouris, M. Bakarezos, M. Tatarakis, N. A. Papadogiannis, and V. Dimitriou, "A review of finite element studies in string musical instruments," *Acoustics*, vol. 4, no. 1, pp. 183–202, 2022.

[2] A. Brauchler, D. Hose, P. Ziegler, M. Hanss, and P. Eberhard, "Distinguishing geometrically identical instruments: Possibilistic identification of material parameters in a parametrically model order reduced finite element model of a classical guitar," *Journal of Sound and Vibration*, vol. 535, p. 117071, 2022.

[3] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems*. Philadelphia: SIAM, 2005.

[4] Abaqus, *Analysis User's Guide, Version 6.14*. Simulia, Providence, 2014.

[5] X. Xie, C. Webster, and T. Iliescu, "Closure learning for nonlinear model reduction using deep residual neural network," *Fluids*, vol. 5, no. 1, p. 39, 2020.

[6] S. Gonzalez, D. Salvi, D. Baeza, F. Antonacci, and A. Sarti, "A data-driven approach to violin making," *Scientific Reports*, vol. 11, no. 1, p. 9455, 2021.

[7] J. Rettberg, D. Wittwar, P. Buchfink, A. Brauchler, P. Ziegler, J. Fehr, and B. Haasdonk, "Port-Hamiltonian fluid-structure interaction modeling and structure-preserving model order reduction of a classical guitar," *Mathematical and Computer Modelling of Dynamical Systems*. doi: 10.1080/13873954.2023.2173238.

[8] A. Ezcurra, M. Elejabarrieta, and C. Santamaria, "Fluid-structure coupling in the guitar box: Numerical and experimental comparative study," *Applied Acoustics*, vol. 66, no. 4, pp. 411–425, 2005.

[9] D. Konopka, C. Gebhardt, and M. Kaliske, "Numerical modelling of wooden structures," *Journal of Cultural Heritage*, vol. 27, pp. S93–S102, 2017.

[10] A. Brauchler, P. Ziegler, and P. Eberhard, "An entirely reverse-engineered finite element model of a classical guitar in comparison with experimental data," *The Journal of the Acoustical Society of America*, vol. 149, no. 6, pp. 4450–4462, 2021.

[11] D. E. Kretschmann, "Mechanical properties of wood," in *Wood Handbook: Wood as an Engineering Material*, (Madison), Forest Products Laboratory, 2010.

[12] U. Baur, C. Beattie, P. Benner, and S. Gugercin, "Interpolatory projection methods for parameterized model reduction," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2489–2518, 2011.

[13] M. Pastor, M. Binda, and T. Harčarik, "Modal assurance criterion," *Procedia Engineering*, vol. 48, pp. 543–548, 2012.

[14] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. of ICNN'95-International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, 1995.

[15] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the marquardt algorithm," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 989–993, 1994.

[16] J. Meyer, "Quality aspects of the guitar tone," in *Function, Construction and Quality of the Guitar* (E. V. Jansson, ed.), pp. 51–75, Stockholm: Royal Swedish Academy of Music, 1983.