Schlecht, Sebastian; Parker, Julian; Schäfer, Maximilian; Rabenstein, Rudolf

Physical Modeling Using Recurrent Neural Networks with Fast Convolutional Layers

Published in:
Proceedings of the 25th International Conference on Digital Audio Effects (DAFx20in22)

Published: 01/01/2022

Document Version
Publisher's PDF, also known as Version of record

PHYSICAL MODELING USING RECURRENT NEURAL NETWORKS WITH FAST CONVOLUTIONAL LAYERS

*Julian D. Parker**

Native Instruments GmbH
Berlin, Germany
firstname.lastname@native-instruments.de

*Sebastian J. Schlecht**

Aalto Acoustics Lab and Media Lab
Aalto University
Espoo, Finland
sebastian.schlecht@aalto.fi

Rudolf Rabenstein† **

Multimedia Communications and Signal Processing
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)
Erlangen, Germany
Rudolf.Rabenstein@fau.de

Maximilian Schäfer† **

Digital Communications
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)
Erlangen, Germany
max.schaefer@fau.de

ABSTRACT

Discrete-time modeling of acoustic, mechanical and electrical systems is a prominent topic in the musical signal processing literature. Such models are mostly derived by discretizing a mathematical model, given in terms of ordinary or partial differential equations, using established techniques. Recent work has applied the techniques of machine-learning to construct such models automatically from data for the case of systems which have lumped states described by scalar values, such as electrical circuits. In this work, we examine how similar techniques are able to construct models of systems which have spatially distributed rather than lumped states. We describe several novel recurrent neural network structures, and show how they can be thought of as an extension of modal techniques. As a proof of concept, we generate synthetic data for three physical systems and show that the proposed network structures can be trained with this data to reproduce the behavior of these systems.

1. INTRODUCTION

Discrete-time modeling of systems (both acoustic, mechanical & electrical) which are relevant to musical uses has a long history in the literature. This discipline is known as *physical modeling*, or as *virtual-analog* when referring to the subset of electrical systems. Popular approaches to such problems in the acoustic or mechanical domain include direct numerical solution of *ordinary differential equations* (ODEs) or *partial differential equations* (PDEs) via finite differences [1], digital waveguides [2] and modal approaches like the Functional Transformation Method (FTM) [3, 4]. Popular approaches in the electrical domain include finite differences or state-space models [5], wave digital filters [6] and the Port-Hamiltonian formalism [7]. As is well-known, in the last 10 years Machine Learning (ML) and specifically Neural Networks (NN) have seen an unprecedented explosion in research, in both theoretical topics and in their application to many diverse fields of study.Applications of such techniques to modeling musically-relevant electrical circuits has seen much research in the last several years [8–10]. Such circuits can generally be thought to be governed by ODEs. The application of NNs to model PDE-governed systems has also seen some research, but strongly in the domain of scientific computing [11, 12]. However, with some exceptions [13], there is little work on the use of NNs to model musically relevant PDE-governed systems at audio-rates. In this work we attempt to make some first steps into this domain. In particular, we extend on network types previously proposed in the scientific-computing domain for the modeling of PDE-governed systems [11] and propose novel network structures with improved capability to be applied in the audio domain. The proposed network structure is perfectly suited for the modeling of oscillating acoustical systems which we show by revealing the similarities to established techniques for physical modeling based on modal synthesis. Finally, we demonstrate the viability of the proposed structures by the modeling of three example systems. All required code for reproducing the results are provided online [14].

In Sec. 2 we review current research on the application of NNs to solve PDEs, and introduce the proposed network structures. Moreover, we review the relevant formalism of the FTM and discuss its connection to the proposed NN structure. In Sec. 3, we introduce three acoustical systems which are used for the evaluation of the proposed NN structures. In Sec. 4 we first explain the NN training procedure applied and discuss the performance of our proposed NN structure compared to analytical solutions of the example systems. Sec. 5 concludes the paper.

2. MODELING PDES USING NN STRUCTURES

The modeling of ODE-governed systems via the application of machine learning is relatively trivial to construct, given that we can write such a system in the general case as follows

$$\dot{\mathbf{u}}(t) = g(\mathbf{u}(t)), \tag{1}$$

where $\mathbf{u}$ is an n-dimensional vector of scalar-valued states, and g is an arbitrary pointwise mapping $\mathbb{R}^n \rightarrow \mathbb{R}^n$. This formulation even encompasses implicitly defined ODE systems, given that we

* Equal contribution

† This work was supported by the German Research Foundation (DFG) under grant number RA 807/7-1

do not need to know g in analytical form, and hence it can be assumed to resolve the implicit relationship.

From this formulation, it is clear that we can learn the dynamic of any ODE system by approximating the function g with a neural network. This learned function can then be used directly within a standard ODE solver [15]. Alternatively, a *recurrent neural network* (RNN)-like structure can be trained to reproduce the progression between sampled values of $\mathbf{u}$ [8].

Modeling PDEs is conceptually a little more difficult, as they deal with the evolution of higher-dimensional distributed states rather than 0-dimensional scalar states, and hence cannot be characterized by a pointwise mapping. Initial work in this area, known as *physically informed neural networks* (PINN), approached this problem by using a neural network to directly learn the solution manifold of the system as a function of coordinates. Such an approach has two major downsides in the context of audio-focused physical modeling. Firstly it requires the governing PDE to be already known. Secondly, it learns only the solution to a particular set of initial conditions, and therefore cannot be used flexibly to reproduce the response to arbitrary excitations.

Recent work has framed the problem differently, by formulating time-dependent PDE-systems analogously to (1) as follows

$$\dot{\mathbf{u}}(\mathbf{x}, t) = \mathcal{G}(\mathbf{u}(\mathbf{x}, t)), \tag{2}$$

where $\mathbf{u} = (u_i)_{i=0}^{N}$ is a vector of space and time-dependent states, e.g., physical quantities of the underlying system, on a bounded n-dimensional region $\mathbf{x} \in V \subset \mathbb{R}^n$, and $\mathcal{G}$ is an operator mapping between function spaces.

The machine-learning problem can be formulated by discretizing $\mathbf{u}$ in space and time. In particular, we apply spatial sampling on an arbitrary grid of discrete sampling points $\mathbf{x_n}$ with $\mathbf{n} \in \mathbb{I}$, where the index set $\mathbb{I}$ is chosen such that $\mathbf{x_n} \in V$, and temporal sampling at regular intervals in time, i.e., $t = kT$ with discrete-time index k and sampling interval T. The temporal and spatial discretization of $\mathbf{u}$ in (2) yields the tensor

$$\mathbf{U}[\mathbf{x_n}, k] = (u_i[\mathbf{x_n}, k])_{i=0}^{N}\,. \tag{3}$$

The discretization of $\mathbf{u}$ allows us to define a discretized version of the PDE (2) in terms of the tensor (3) as follows

$$\mathbf{U}[\mathbf{x_n}, k+1] = \hat{\mathcal{G}}\left(\mathbf{U}[\mathbf{x_n}, k]\right), \tag{4}$$

where $\hat{\mathcal{G}}$ is a discretized version of the original operator $\mathcal{G}$. Approximating the operator $\hat{\mathcal{G}}$ by a NN is known as the *neural operator* approach [11]. In the following, we use the shorthand $\boldsymbol{U}^k$ to denote the tensor at time k, and $u^k_{ij\ldots}$ denotes its elements, where i is the index of the physical quantity, $j\ldots$ denote the n spatial variables.

There are many options for what type of network to use for this approximation. The first impulse might be to flatten the tensor $\mathbf{U}$ and use a standard dense network. This approach would make no assumptions about the relationship between elements of the tensor. This could work theoretically, but discards a number of structural priors that can be used to inform the network design and to ease training. The dimensions of the tensor $\mathbf{U}$ in (2) representing the discretized spatial domains are made up of progressively sampled values from continuous functions. The values along these dimensions are strongly related as they embed some concept of locality and ordering. Instead of using a fully dense transformation to operate along these dimensions, it is therefore more sensible to use a structure that contains these assumptions and transforms the data with some type of kernel. Previous work has investigated a number of approaches to this, including *graph neural networks* [16], *convolutional neural networks* with small kernels [16], and transforming into the spatial Fourier domain [11]. This last technique has seen the greatest success, and is known as the *fourier neural operator* (FNO) approach.

These kernel-type methods of approximating the operator can be considered to be closely related to general kernel methods for the solution of PDEs such as Green's Functions or the eigenfunctions of the spatial differentiation operator concealed in (2) [3, 4]. These relations are discussed in further detail in Sec. 2.3.

FNO-based techniques are significantly better suited than the PINN approach for audio-focused physical modeling, but have so far only been applied in the context of scientific computing. Compared to many typical scientific computing problems, modeling audio-range behaviour requires a system to reproduce dynamics over a wide frequency range, and across long time-scales relative to the temporal sampling frequency. Initial experiments showed that taking the existing FNO-based structures and applying them directly to audio-focused problems did not produce ideal results - in particular the modeled behaviour would degenerate after a relatively small number of timesteps. The goal of this work was therefore to improve the existing FNO-based structures in the aspects important to audio-range modeling.

2.1. Fast convolution layer

Starting from the n-dimensional spectral convolution layer proposed in the context of the FNO [11, 16], we make some modifications for more generalized usage. Firstly, we discard the concept of zeroing some of the bins of the FFT of the data. This zeroing is effectively just a crude lowpass filter implemented in the Fourier domain, and is not beneficial in the general use-case. We also add correct padding of the spatial dimensions to ensure that the layer is performing non-cyclic convolution.

The operation of the layer on the internal state tensor $\boldsymbol{H}$ with elements $h_{\nu j\ldots}$ can be written as follows

$$\mathcal{S}(h_{\nu j\ldots}) = \tilde{\mathcal{F}}^{-1}\left[\sum_{\kappa} A_{\nu\kappa j\ldots}\tilde{\mathcal{F}}(h_{\kappa j\ldots}) + b_{\nu j\ldots}\right], \tag{5}$$

where $\tilde{\mathcal{F}}$ denotes an operator encompassing padding, applying the FFT and truncating the spectrum to remove negative frequency components. Conversely $\tilde{\mathcal{F}}^{-1}$ encompasses reconstructing the negative freq-uency components from the transformed positive frequency components, applying the inverse FFT, and then truncating to remove padding. Note that this linear transformation with $\boldsymbol{A}$ and $\boldsymbol{b}$ is fully dense with respect to input and output channels but diagonal in terms of frequency bins. The elements $A_{\nu\kappa j\ldots}$ and $b_{\nu j\ldots}$ of this transformation are complex, and are the trainable parameters of the layer. With these modifications, the layer can be thought of as a standard convolutional layer with kernels fixed to the width of the domain, but implemented using the well-established method of fast-convolution via multiplication in the Fourier domain. This has an advantage in computational complexity, and also re-contextualizes the training problem by moving parameters from the spatial domain into the Fourier domain. It should be noted that whilst the general discretization shown in (3) allows arbitrary spatial sampling grids, the use of the FFT to implement convolution could be argued to enforce a regular rectilinear spatial grid. This is certainly true in the linear case, but in

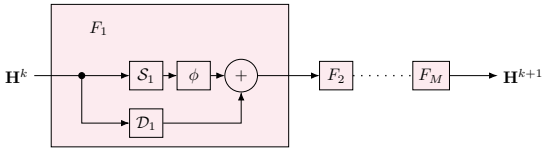

Figure 1: Block diagram showing the recurrent cell of the FRNN structure (dependencies of $\mathbf{H}$ on discrete time index k are denoted in the superscript for brevity).

the context of stacked layers with non-linearites, this may not be true. Nonetheless, it is reasonable to assume that compensating for a non-regular grid may use significant capacity of the network. A proper investigation of this topic is left to future work, with only data sampled on rectilinear spatial grids considered here.

2.2. FNO-derived RNN structures

Previously presented FNO structures have dealt with time-evolution by learning mappings from spatially sampled states at multiple input time-steps [11], or single input time-steps [17] to the next time-step. These structures can be thought of as a type of teacher-forced RNN, but were not previously discussed as such. In this work we introduce several related structures that are designed explicitly as RNNs, and trained using standard RNN training techniques such as *back propagation through time* (BPTT).

2.2.1. Fourier Recurrent Neural Network (FRNN)

This structure is shown in Fig. 1. It consists of a recurrent structure that maps sequentially between time-steps of the spatially-sampled internal states of the network, through a variable number of mapping blocks

$$\mathbf{H}^{k+1} = F_M \circ F_{M-1} \circ \dots F_1(\mathbf{H}^k), \tag{6}$$

where $\mathbf{H}$ is a tensor containing the internal states of the system at all spatial-sampling points, and the mapping block is given by

$$F_m(\mathbf{H}) = \phi(\mathcal{S}_m(\mathbf{H})) + \mathcal{D}_m(\mathbf{H}), \tag{7}$$

where $\mathcal{S}$ is the described fast convolution layer (5), ϕ is an element-wise activation function such as tanh or ReLU and $\mathcal{D}$ represents a weighted skip connection.

This structure is related to that presented in previous FNO literature [17] but with the skip connection positioned to prevent vanishing gradients when using BPTT on longer sequences, and without the restriction that internal states must correspond with physical states. Compared to this structure, we also do not condition the input with the coordinates of the spatial grid. This structure can also be considered to be closely related to a generic RNN, and to the STN structure proposed for modeling ODE-governed systems [8].

2.2.2. Fourier Gated Recurrent Unit (FGRU)

The *gated recurrent unit* (GRU) [18] is an RNN structure that has seen great success in a variety of tasks from language modeling to black-box modeling of electrical circuits [9]. We propose a generalization of this structure by replacing dense layers with fast convolution layers. The structure is shown in Fig. 2, and can be written

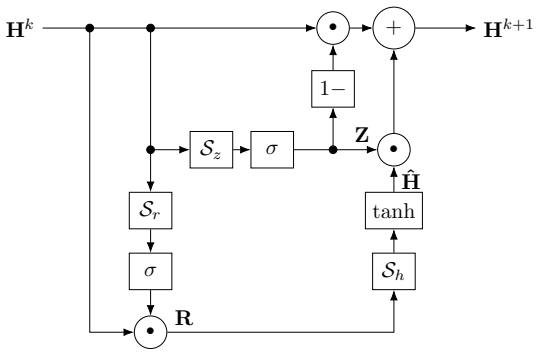

Figure 2: Block diagram showing the recurrent cell of the FGRU structure (dependencies of $\mathbf{H}$ on discrete time index n are denoted in the superscript for brevity).

as follows

$$\mathbf{Z} = \sigma(\mathcal{S}_z(\mathbf{H}^k)), \tag{8}$$

$$\mathbf{R} = \sigma(\mathcal{S}_r(\mathbf{H}^k)), \tag{9}$$

$$\hat{\mathbf{H}} = \tanh(\mathcal{S}_h(\mathbf{R} \odot \mathbf{H}^k)), \tag{10}$$

$$\mathbf{H}^{k+1} = (1 - \mathbf{Z}) \odot \mathbf{H}^k + \mathbf{Z} \odot \hat{\mathbf{H}}, \tag{11}$$

where $\mathbf{H}$ is a tensor as in (6), (7), σ is the element-wise sigmoid function, and $\mathcal{S}_z$, $\mathcal{S}_r$, $\mathcal{S}_h$ are fast convolution layers.

This structure inherits the well-known beneficial qualities of the GRU, including a linear path to avoid vanishing gradients during BPTT, the ability to effectively multiply inputs, and safety against explosive instability due to the bounded nature of the states.

2.2.3. Training procedure

We formulate the training problem as taking a tensor of the initial conditions of the states of the physical system, sampled on an n-dimensional regular grid, and producing a tensor of one dimension higher, representing the evolution of these spatially sampled states over a number of time-steps. This is a form of BPTT, in contrast to the single time-step to single time-step mappings previously described [17], and hence has the advantage that the network can learn to deal with its own error as propagated through the recursion.

For the presented RNN structures, the number of internal states can be freely specified. This is especially important in the case of the FGRU, as it is the only mechanism by which to scale the capacity of the structure. Restricting the internal states $\boldsymbol{H}$ to correspond exactly to the physical states $\boldsymbol{U}$ of the system we are modeling would therefore be overly restrictive. Instead, we apply a process called *soft state-matching* [19]. This consists of defining two trainable linear maps $\boldsymbol{A}$, $\boldsymbol{b}$ and $\tilde{\boldsymbol{A}}$, $\tilde{\boldsymbol{b}}$, respectively,

$$h^0_{\nu j\dots} = \mathcal{M}_{\text{in}}(u^0_{ij\dots}) = \sum_\kappa A_{\nu\kappa} u^0_{\kappa j\dots} + b_\nu, \tag{12}$$

$$u^k_{ij\dots} = \mathcal{M}_{\text{out}}(h^k_{\nu j\dots}) = \sum_\kappa \tilde{A}_{i\kappa} h^k_{\kappa j\dots} + \tilde{b}_i, \tag{13}$$

where $h_{\nu j\dots}$ are the elements of the network's internal state tensor $\mathbf{H}$ and $u_{ij\dots}$ are the elements of the tensor $\mathbf{U}$ containing the states of the system being modeled. The superscript k denotes time-step.

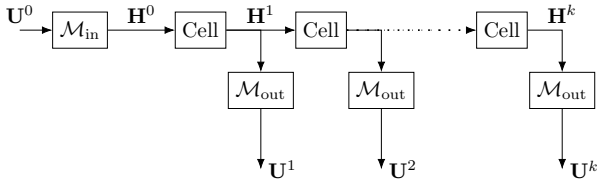

Figure 3: Block diagram showing the repeated application of an RNN cell from Fig. 1 along with the action of the state-matching layers.

These maps translate from the dimensionality of the states of the system being modeled to that of the network, and vice-versa. The first mapping is applied to translate the initial conditions to the initial internal states of the network, and the second mapping is used to translate the states of the network to the states of original system. With the addition of these mappings, the network can be thought of as operating in a higher-dimensional latent space which contains a linear embedding of the original state-space. The combination of recursive training via BPTT with these mapping layers is shown in Fig. 3.

2.3. Relation between FNO structures and Kernel Methods

As previously mentioned, there is a strong relation between the FNO approach and general kernel methods for the solution of PDEs in the form of (2). In the following we investigate the FTM, as a kernel method for finding analytical solutions to (2), the relation to FNOs and to the proposed FNO-derived RNN Structures.

2.3.1. Functional Transformation Method (FTM)

The FTM is a powerful method for modeling the oscillation of acoustic systems in terms of transfer functions and a detailed description of its derivation and application can be found, e.g., in [3,4,20]. For example, the method has been applied for the modeling of strings (see, e.g., [3,21]), membranes (see, e.g., [22,23], and room acoustics (see, e.g., [24, 25]). The basic idea of the FTM is to transform the mathematical description of an acoustical system in terms of PDEs, e.g., (2), into a discrete-time simulation model by the application of functional transformations for time and space dependencies. Consider a vector valued PDE, which is a specific realization of the general PDE in (2)

$$\mathbf{C}\dot{\mathbf{u}}(\mathbf{x},t) = \mathbf{L}\mathbf{u}(\mathbf{x},t), \qquad \mathbf{x} \in V,\, t > 0, \tag{14}$$

defined on spatial domain V and its boundary ∂V. In PDE (14), matrix $\mathbf{C} \in \mathbb{R}^{N\times N}$ is a capacitance matrix and $\mathbf{L}$ is a $N \times N$ sized spatial differentiation operator. The state vector $\mathbf{u} \in \mathbb{R}^{N\times 1}$ contains N physical quantities of the underlying acoustical system such as deflection, sound pressure or particle velocities (see, e.g., [25, Eqs. (26), (34)]). To complete PDE (14), we assume a set of homogeneous boundary conditions defined on ∂V, cf. [20], and an initial condition for the vector $\mathbf{u}$ at $t = 0$, i.e., $\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_\mathrm{i}(\mathbf{x})$.

In order to derive an FTM model for (14), $\mathbf{u}$ is expanded into an infinite set of bi-orthogonal basis functions $\boldsymbol{K}_\mu \in \mathbb{C}^{N\times 1}$ and $\tilde{\boldsymbol{K}}_\mu \in \mathbb{C}^{N\times 1}$ for the operator $\mathbf{L}$, where the dedicated eigenvalues s_μ, $\mu \in \mathbb{N}_0$ constitute the discrete spectrum of $\mathbf{L}$ [26]. With basis function $\boldsymbol{K}_\mu$ and $\tilde{\boldsymbol{K}}_\mu$ a forward and inverse Sturm-Liouville transform (SLT) are defined as follows

$$\bar{u}_\mu(t) = \mathcal{T}\{\mathbf{u}(\mathbf{x},t)\} = \int_V \tilde{\boldsymbol{K}}_\mu^\mathrm{H}(\mathbf{x})\mathbf{C}\mathbf{u}(\mathbf{x},t)\,\mathrm{d}\mathbf{x}, \tag{15}$$

$$\mathbf{u}(\mathbf{x},t) = \mathcal{T}^{-1}\{\bar{u}_\mu(t)\} = \sum_{\mu=0}^{\infty} \frac{1}{N_\mu}\bar{u}_\mu(t)\boldsymbol{K}_\mu(\mathbf{x}). \tag{16}$$

Forward SLT (15) expands $\mathbf{u}$ into the basis functions $\tilde{\boldsymbol{K}}_\mu$ yielding the expansion coefficients $\bar{u}_\mu$. In the context of acoustics, (15) expands a system into its modes and the time-dependent expansion coefficients $\bar{u}_\mu$ describe their temporal evolution. The inverse SLT in (16) represents $\mathbf{u}$ as series expansion with basis functions $\boldsymbol{K}_\mu$ and scaling factors $N_\mu = \int_V \tilde{\boldsymbol{K}}_\mu^\mathrm{H}\mathbf{C}\boldsymbol{K}_\mu\,\mathrm{d}\mathbf{x}$. Further properties of (15) and (16) such as the existence of a differentiation theorem and the bi-orthogonality of $\boldsymbol{K}$ and $\tilde{\boldsymbol{K}}$ are discussed in detail in [4, Sec. 4.7.3]. We note that the shape of $\boldsymbol{K}$ and $\tilde{\boldsymbol{K}}$ depends on the spatial shape of the underlying system, e.g., for geometrically simple systems basis functions are often trigonometric functions, Bessel functions or combinations thereof [4, Sec. 4.7.4].

Application of the forward SLT (15) to PDE (14) and exploiting the differentiation theorem [25, Eq. (18)] leads to expansion coefficients defined in terms of Laplace transfer functions

$$\bar{u}_\mu(t) = \mathrm{e}^{s_\mu t}\bar{u}_{\mathrm{i},\mu} \quad \circ\!\!-\!\!\bullet \quad \bar{U}_\mu(s) = \frac{1}{s - s_\mu}\bar{u}_{\mathrm{i},\mu}, \tag{17}$$

where the coefficients $\bar{u}_{\mathrm{i},\mu}$ follow from the expansion of the initial values $\mathbf{u}_\mathrm{i}$, i.e., $\bar{u}_{\mathrm{i},\mu} = \mathcal{T}\{\mathbf{u}_\mathrm{i}(\mathbf{x})\}$.

Truncating the number of eigenvalues s_μ to be finite, i.e., $\mu = 0, \ldots, Q-1$, and transforming (16) and (17) into the discrete-time domain allows the formulation of the FTM model in terms of a vector valued state space description (SSD) [4, 25]

$$\bar{\mathbf{u}}[k+1] = \mathrm{e}^{\boldsymbol{\mathcal{A}}T}\bar{\mathbf{u}}[k] + T\bar{\mathbf{u}}_\mathrm{i}\delta[k], \tag{18}$$

$$\mathbf{u}[\mathbf{x},k] = \boldsymbol{\mathcal{C}}(\mathbf{x})\bar{\mathbf{u}}[k], \tag{19}$$

with the discrete time index k, sampling interval T, i.e., $t = kT$, and the discrete-time Dirac delta function $\delta[k]$. *State equation* (18) is a vector valued discrete-time version of (17), where the vector $\bar{\mathbf{u}} = (\bar{u}_\mu)_{\mu=0}^{Q-1}$ contains the expansion coefficients, the diagonal matrix $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{Q\times Q}$ contains the eigenvalues s_μ on its main diagonal and $\bar{\mathbf{u}}_\mathrm{i} = (\bar{u}_{\mathrm{i},\mu})_{\mu=0}^{Q-1}$. *Output equation* (19) is a clever reformulation of the inverse SLT in (16) by representing the truncated sum by a matrix-vector multiplication with the transformation matrix

$$\boldsymbol{\mathcal{C}}(\mathbf{x}) = \left[\frac{1}{N_0}\boldsymbol{K}_0(\mathbf{x}),\, \ldots,\, \frac{1}{N_{Q-1}}\boldsymbol{K}_{Q-1}(\mathbf{x})\right]. \tag{20}$$

2.3.2. Relation between FTM and FNO-derived RNN Structures

The FTM formulation allows some interpretation of the fast convolution layer proposed in the FNO-derived RNN structure. The state equation (18) describes the time-evolution of the modes phase and amplitude $\bar{\mathbf{u}}$. The state transition matrix $\mathrm{e}^{\boldsymbol{\mathcal{A}}T}$ is diagonal, such that time-evolution of each mode state $\bar{\mathbf{u}}$ is decoupled.

For a set of discrete spatial sampling points $\mathbf{x}_\mathbf{n}$ with $n \in \mathbb{I}$ and discrete time k, the transformation (15) and (16) are

$$\bar{\mathbf{u}}[k] = \sum_{\mathbf{n}\in\mathbb{I}} \tilde{\boldsymbol{\mathcal{C}}}^\mathrm{H}(\mathbf{x}_\mathbf{n})\mathbf{C}V_\mathbf{n}\,\mathbf{u}[\mathbf{x}_\mathbf{n},k], \tag{21}$$

$$\mathbf{u}[\mathbf{x}_\mathbf{n},k] = \boldsymbol{\mathcal{C}}(\mathbf{x}_\mathbf{n})\,\bar{\mathbf{u}}[k], \tag{22}$$

where $V_\mathbf{n}$ denotes the finite volume element replacing the infinitesimal volume element $\mathrm{d}\mathbf{x}$ and $\tilde{\boldsymbol{C}}$ is defined by the eigenfunctions $\tilde{\boldsymbol{K}}$ similar to (20). For geometrically simple examples, such as strings and rectangles, the underlying basis functions $\boldsymbol{K}_\mu$ are trigonometric. In a linear time-invariant system, the internal modes can be recovered readily from the time evolution of the spatial variables. A relation between two consecutive time steps in the space domain follows by inserting (18) and (21) for $k > 0$ into (19)

$$\mathbf{u}[\mathbf{x}_\mathbf{n}, k+1] = \sum_{\nu \in \mathbb{I}} \mathbf{G}(\mathbf{x}_\mathbf{n}|\mathbf{x}_\nu)\mathbf{u}[\mathbf{x}_\nu, k] \tag{23}$$

with the matrix in the form of an eigenvalue decomposition

$$\mathbf{G}(\mathbf{x}_\mathbf{n}|\mathbf{x}_\nu) = \boldsymbol{C}(\mathbf{x}_\mathbf{n})\mathrm{e}^{\boldsymbol{\mathcal{A}}T}\tilde{\boldsymbol{C}}^\mathrm{H}(\mathbf{x}_\nu)\mathbf{C}V_\mathbf{n}. \tag{24}$$

Note that (23) is the linear version of the general discretized PDE (4) and $\mathbf{G}(\mathbf{x}_\mathbf{n}|\mathbf{x}_\nu)$ resembles a discrete version of the Green's function of (14). Transition matrix $\mathbf{G}$ can be numerically recovered from the spatial variables by solving a least squares problem in (23) on a sufficiently long time frame. The eigenvalue decomposition of $\mathbf{G}$ yields then eigenvalues $\mathrm{e}^{\boldsymbol{\mathcal{A}}T}$.

Similar operations are performed by the fast convolution layer in (5). An FFT transforms the spatial variables into a transform domain with complex exponential basis functions, thus also trigonometric functions such as in $\boldsymbol{C}$. The processing step in (5) applies a diagonal transition matrix $A_{\nu\kappa j\ldots}$ to the transform-domain variables. The result is transformed back to the spatial variables via the inverse FFT. There are also notable differences between the given approaches. The fast convolution layer applies a bias term, i.e., $b_{\nu j\ldots}$ in (5). The FNO-derived RNN structure typically concatenates multiple layers (see (6)), uses non-linear activation functions and skip connections in-between (see (7)). Because of the mapping performed by the first and last network layers as in (13), the NN recombines and expands the spatial variable inputs and outputs.

3. ACOUSTIC SYSTEMS

In this section we introduce a number of acoustic systems which are used to test the viability of the proposed NN structures. First, we investigate two linear systems, – a *lossy dispersive string* and a *2D wave equation*. For both systems we employ well investigated models obtained by the FTM as introduced in Sec. 2.3.1. Second, we investigate a *tension modulated string* as a non-linear system which we solve, after a few modifications, by a non-linear ODE-solver in Python.

The models presented have been well investigated previously therefore we just present the parts necessary for a comprehensible presentation. We refer the reader to the relevant literature where appropriate.

3.1. Linear Lossy Dispersive String

The oscillation of a vibrating string of length ℓ, i.e., $0 \le x \le \ell$ at times $t > 0$ can be described by a PDE in terms of the string deflection $u_0(x,t)$ as follows [2]

$$\begin{aligned}\rho_\mathrm{s} A u_1(x,t) + EI u_0''''(x,t) - T_{\mathrm{s}0} u_0''(x,t) \\ + d_1 u_1(x,t) - d_3 u_1''(x,t) = 0,\end{aligned} \tag{25}$$

where partial derivatives for space are denoted by a prime, i.e., $\frac{\partial u}{\partial x} = u'$, and velocity by u_1. The oscillation of the string in (25) is influenced by several physical parameters. In particular, ρ_s denotes the string density, A is the cross section area, I is the moment of inertia and $T_{\mathrm{s}0}$ is the constant tension of the string. Constant E denotes Young's modulus. The parameters d_1 and d_3 introduce frequency-independent and frequency-dependent damping into the oscillation of the string. As boundary conditions we assume that the string is simply supported, which requires that the deflection u_0 and its second derivative u_0'' are zero at both ends, see [21]. At $t = 0$, deflection is defined by an initial value $u_0(x,0) = u_\mathrm{i}(x)$.

The derivation of an FTM model for the lossy dispersive string has been discussed, e.g., in [3, 20, 21, 25] for different applications in sound synthesis. To this end, we don't show the exact formulas for all components of the FTM model, instead we refer the reader to the most recent FTM references:

- To derive the state transition matrix $\mathrm{e}^{\boldsymbol{\mathcal{A}}T}$ in (18), we used the eigenvalues s_μ from [25, Eq. (37)].
- The eigenfunctions $\boldsymbol{K}_\mu$ in $\boldsymbol{C}$ in (20) and $\tilde{\boldsymbol{K}}_\mu$ for the expansion in (15) can be found in [25, Eq. (39)].

The vector of states $\mathbf{u}$, c.f. (14), (2), comprises string deflection u_0 and velocity u_1, i.e., $\mathbf{u} = [u_0,\, u_1]^\mathrm{T}$. For training and evaluation we use different initial values u_i, i.e., a delta impulse, a smooth excitation by a raised cosine, and a random initial condition.

3.2. 2D Wave Equation

The evolution of sound pressure u_0 and particle velocities $\mathbf{u}_\mathrm{v} = [u_1,\, u_2]^\mathrm{T}$ in a bounded 2-dimensional (2d) region of size $0 \le x \le L_x$, $0 \le y \le L_y$ is described by the 2d wave equation

$$\rho_0 \dot{\mathbf{u}}_\mathrm{v}(x,y,t) + \operatorname{grad} u_0(x,y,t) = 0, \tag{26}$$

$$\rho_0\, c_0^2 \operatorname{div} \mathbf{u}_\mathrm{v}(x,y,t) + \dot{u}_0(x,y,t) = 0, \tag{27}$$

where ρ_0 denotes the density of air and c_0 is the speed of sound. The operators grad and div denote gradient and divergence in 2D Cartesian coordinates, respectively. For the boundary conditions we assume fully reflective boundaries, i.e., particle velocities $\mathbf{u}_\mathrm{v}$ vanish at the boundaries. At $t = 0$, sound pressure u_0 is defined by an initial value, i.e., $u_0(x,y,0) = u_\mathrm{i}(x,y)$. The 2D wave equation (26), (27) and its extension to 3D are frequently used in room acoustics [27, 28]. The components required to establish a FTM model for the 2D wave equation are as follows:

- To derive the state transition matrix $\mathrm{e}^{\boldsymbol{\mathcal{A}}T}$ in (18), we used the eigenvalues s_μ from [25, Eq. (30)].
- The eigenfunctions $\boldsymbol{K}_\mu$ in $\boldsymbol{C}$ in (20) and $\tilde{\boldsymbol{K}}_\mu$ for the expansion in (15) can be found in [25, Eq. (31)].

The vector of states $\mathbf{u}$, c.f. (14), (2), comprises sound pressure u_0 and particle velocities $\mathbf{u}_\mathrm{v}$, i.e., $\mathbf{u} = [u_0,\, u_1,\, u_2]^\mathrm{T}$. Similar to the string we use different types of initial conditions u_i, i.e., a 2d delta impulse, and a random sound pressure distribution.

3.3. Non-linear Tension Modulated String

As a non-linear acoustical system we consider a tension modulated string [3, 29–32]. Tension modulation is a non-linear phenomenon which affects the pitch of the string. In particular, when the string is deflected from the rest position, the arc length measured along the string is larger than the string length ℓ at rest. This additional "stretching" leads to an increased tension and subsequently to an increased pitch.

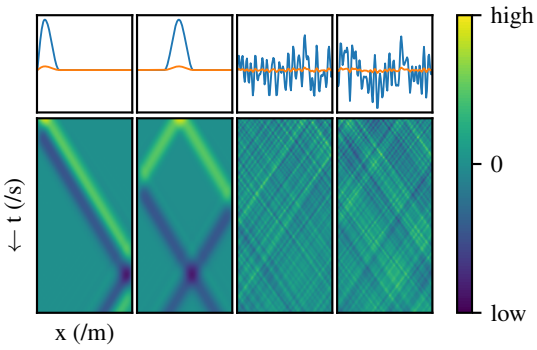

Figure 4: Examples of training pairs in the case of the linear string model. The top row shows the initial conditions, in this case of two states. The bottom row shows the evolution of the first state over a small time period. The dataset includes both states.

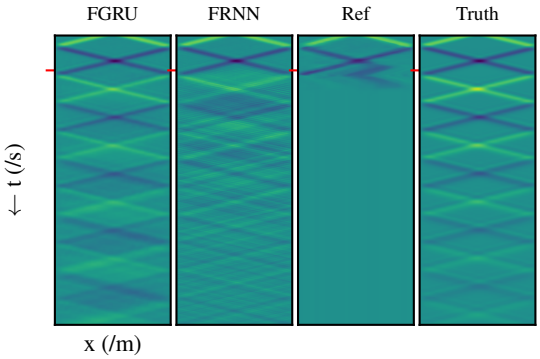

Figure 5: Deflection of the linear string over time and space as obtained by the proposed FGRU and FRNN structures, the reference FNO model and the analytical solution obtained by FTM. The color scaling is the same as given in Fig. 4. The red tick mark represents the duration of examples used during training.

The PDE describing the oscillation on a tension modulated string is similar to the linear case (25), except for the tension which depends on $u_0(x,t)$ and is defined as follows

$$T_\mathrm{s}(u_0) = T_{\mathrm{s}0} + T_{\mathrm{s}1}(u_0), \tag{28}$$

where $T_{\mathrm{s}0}$ is the tension of the string at rest as defined in (25). The additional tension $T_{\mathrm{s}1}(u_0)$ arising from the additional arc length $\Delta l_\mathrm{str}(u_0)$ of the string is derived from Hooke's law [32]

$$T_{\mathrm{s}1}(u_0) = EA\frac{\Delta l_\mathrm{str}(u_0)}{\ell} = \frac{EA}{2\ell}\int_0^\ell \left(u_0'(x,t)\right)^2 \mathrm{d}x. \tag{29}$$

In contrast to the linear example systems, we derived a numerical model for the tension modulated string. In particular, we employed a two-staged procedure to obtain numerical solutions for the deflection u_0 and velocity u_1 of the tension modulated string:

1) First we decompose PDE (25), extendend by the non-linear tension (28), into a set on non-linear ODEs by the application of a Fourier-Sine transformation [30].
2) The system of ODEs is converted into a vector formulation of size M ODEs, solved numerically in Python.

Similar to the linear string we employ different types of initial conditions for u_0, i.e., a delta impulse and a random initial condition, and the same state vector $\mathbf{u}$.

4. EVALUATION

The networks described in Sec. 2 were implemented in the PyTorch framework [33], and are available at the accompanying website [14]. A reference implementation of the previous Markov-FNO approach was also adapted from existing code, with the addition of support for BPTT. These networks were then trained on datasets generated from the models described in 3. These datasets consist of 1024 pairs of a set of initial conditions and the response over a set period of time. The initial conditions are split between two categories. Half consist of the response to randomly positioned impulses or plucks, and the other half the response to setting the initial conditions to random values across the domain. The datasets are normalized to have unit variance. One tenth of each dataset is retained for validation. Further physical parameters of the datasets are given at the accompanying website [14]. Fig 4 shows an example of the types of training pairs provided in the case of the 1d string.

4.1. Training methodology

Training was conducted using the AdamW optimizer [34] with default parameters, and the 1-cycle learning-rate scheduling scheme [35] modulating from a learning rate of 10^{-4} to 10^{-3}, with MSE between the target and predicted output sequences as the objective. Training was conducted for 5000 epochs with batch size set individually for each dataset in order to maximize GPU memory usage. The training was conducted on cloud-hosted virtual machines equipped with NVIDIA T4 GPUs. All training code is made available at the accompanying website, which also documents the used hyper-parameters [14]. Network capacities were set to be approximately equal between the different network architectures. In practice, this means using the same number of internal states, with 3 stacked layers in the FRNN and reference FNO model used to match the non-variable 3 layers in the FGRU.

4.2. Results

In the below table we give MSE values for the proposed models and the reference model, validated on a reserved subset of the data not seen during training:

	FGRU	FRNN	Ref.
1d linear string	**7.79e-3**	1.43e-2	3.50e-1
1d nonlinear string	**2.27e-3**	2.63e-2	1.06
2d wave equation	1.86e-2	1.54e-2	**1.12e-4**

4.2.1. Linear Lossy Dispersive String

Fig. 5 shows the result of exciting the models trained on the linear string data with an impulse halfway along its length. The timespan shown is approximately 10x that seen by the models during training. As can be clearly seen, the best performing model is the FGRU, which manages to sustain accurate behavior over the majority of the time-period considered. The FRNN also decays at

approximately the correct rate, but exhibits more and more dispersion after the time-span seen during training. The reference FNO model seems to fit the behavior well during the time-span seen during training, but breaks down completely after that. This ordering of accuracy is also confirmed by the validation MSE values.

We can further examine the behavior of the models by using the approximate eigenvalue decomposition derived in 2.3.2. The results are shown in Fig. 6. We observe that for many of the prominent low-frequency poles of the original FTM model, the NNs recover a pole with similar frequency and magnitude. As expected from the comparison in Fig. 5, the FGRU best matches the pole magnitudes, although the FRNN also matches them well at low frequencies. The reference FNO model seems to have estimated some pole frequencies well, but significantly damped. All of the models seems to struggle above 4kHz.

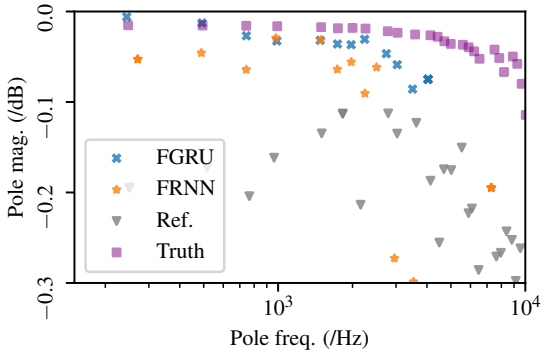

Figure 6: Estimated poles of the linear string inferred from the model outputs presented in Fig. 5.

4.2.2. *Non-linear Tension Modulated String*

Fig. 7 shows the result of exciting the models of the nonlinear tension modulated string with a pluck halfway along the string's length, with an initial pluck amplitude of 1mm. Again the time period shown is roughly 10x that used for training. Broadly the same hierarchy is seen as in the case of the linear string in Fig. 5. The result from FGRU fits the behaviour well over a quite long period, although it appears to be slightly more damped than the ground truth. Moreover, the triangular-like waveshape induced by the nonlinear behavior is reproduced well by the FGRU. The results from FRNN fit the data during the initial cycle which has been seen during training, but then spurious damping and dispersion dominates. The reference FNO model is not able to fit the data at all, with training plateauing very early and never converging on a reasonable approximation. Again, this hierarchy seems to generalize to a wider set of examples, as is reflected in the validation MSE values seen above.

4.2.3. *2d Wave Equation*

Fig. 8 shows the result of exciting the trained models with an impulse halfway along the x dimension, and slightly less than halfway along the y dimension. In this case, the time period shown is around 2x that seen during training. In this case we see an interesting reversal of the results seen on the string models. The reference FNO model performs the best, with the FRNN and especially

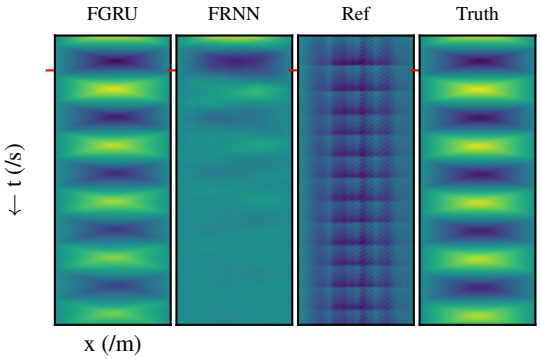

Figure 7: Deflection of the tension modulated string over time and space as obtained by the proposed FGRU and FRNN structures, the reference FNO model and the numerical solution described in Sec. 3.3. The color scaling is the same as given in Fig. 4. The red tick mark represents the duration of examples used during training.

the FGRU falling off in accuracy after 1ms. These observations are in agreement with the validation MSE calculated over a wider range of examples. It is beyond the scope of the current work to properly examine why the hierarchy of performances is reversed in this case. An initial speculation might be that given the mathematically very simple structure of this model (despite its higher dimension count), the FNO might be benefiting from some kind of regularization effect gained by the constriction of the internal states back to the physical states between each time step. Further investigation of this phenomenon is left to future work.

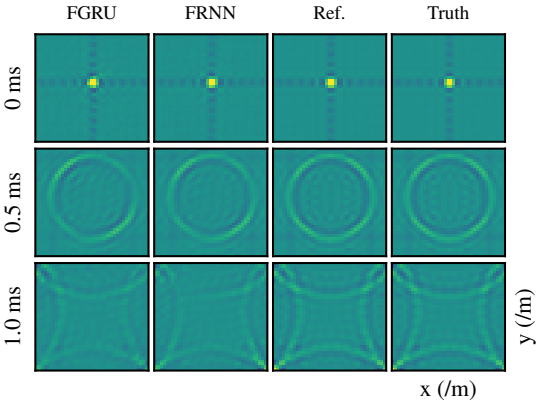

Figure 8: Spatial distribution of sound pressure at different points in time as obtained by the proposed FGRU and FRNN structures, the reference FNO model and the analytical solution obtained by the FTM model.

5. CONCLUSIONS

In this work we gave an overview of existing methods of modeling PDEs using NNs. We presented two new structures based on the FNO approach, designed to perform better in audio-oriented tasks. We compared these structures theoretically to the FTM,

and showed that they have significant mathematical parallels. This opens up an interesting new avenue in terms of the interpretability of these networks.

As an initial proof of concept, we trained the proposed network structures to reproduce datasets generated by existing physical models. The results of this training show that the proposed networks have potential for physical modeling in the audio domain. Future work could explore the applicability of such models to systems which are harder or more computationally expensive to model using existing techniques.

6. REFERENCES

[1] S. Bilbao, *Numerical Sound Synthesis*, John Wiley and Sons, Chichester, UK, 2009.

[2] J. Bensa, S. Bilbao, R. Kronland-Martinet, and J. O. Smith, "The Simulation of Piano String Vibration: From Physical Models to Finite Difference Schemes and Digital Waveguides," *J. Acoust. Soc. Am.*, vol. 114, no. 2, pp. 1095–1107, 2003.

[3] L. Trautmann and R. Rabenstein, *Digital Sound Synthesis by Physical Modeling Using the Functional Transformation Method*, Springer US, Boston, MA, 2003.

[4] M. Schäfer, *Simulation of Distributed Parameter Systems by Transfer Function Models*, Doctoral thesis, Friedrich-Alexander-Universtität Erlangen-Nürnberg (FAU), 2019, https://opus4.kobv.de/opus4-fau/frontdoor/index/index/docId/13174.

[5] M. Holters and U. Zölzer, "A generalized method for the derivation of non-linear state-space models from circuit schematics," in *23rd Eur. Signal Process. Conf. (EUSIPCO)*, 2015, pp. 1073–1077.

[6] A. Fettweis, "Wave Digital Filters: Theory and Practice," *Proc. IEEE*, vol. 74, no. 2, pp. 270–327, 1986.

[7] A. Falaize and T. Hélie, "Passive guaranteed simulation of analog audio circuits: A port-Hamiltonian approach," *Appl. Sci.*, vol. 6, pp. 273, 9 2016.

[8] J. Parker, F. Esqueda, and A. Bergner, "Modelling of nonlinear state-space systems using a deep neural network," in *Proc. 22nd Int. Conf. Digit. Audio Eff. (DAFx-19)*, Birmingham, UK, 2019.

[9] A. Wright, E.-P. Damskägg, and V. Välimäki, "Real-time black-box modelling with recurrent neural networks," in *Proc. 22nd Int. Conf. Digit. Audio Eff. (DAFx-19)*, Birmingham, UK, 2019.

[10] E.-P. Damskägg, L. Juvela, E. Thuillier, and V. Välimäki, "Deep learning for tube amplifier emulation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP 2019)*, Brighton, UK, 2019, pp. 471–475.

[11] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier Neural Operator for Parametric Partial Differential Equations," 2020, http://arxiv.org/abs/2010.08895.

[12] M. Raissi, P. Perdikaris, and G.E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comp. Phys.*, vol. 378, pp. 686–707, 2019.

[13] L. Gabrielli, S. Tomassetti, S. Squartini, and C. Zinato, "Introducing deep machine learning for parameter estimation in physical modelling," in *Proc. 20th Int. Conf. Digit. Audio Eff. (DAFx-17)*, Edinburgh, UK, 2017.

[14] J. D. Parker, S. J. Schlecht, M. Schäfer, and R. Rabenstein, "Accompanying website and code repository," 2022, https://julian-parker.github.io/DAFX22_FNO/.

[15] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Multistep neural networks for data-driven discovery of nonlinear dynamical systems," 2018, https://arxiv.org/abs/1801.01236.

[16] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Neural Operator: Learning Maps Between Function Spaces," pp. 1–89, 2021, http://arxiv.org/abs/2108.08481.

[17] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Markov neural operators for learning chaotic systems," 2021, https://arxiv.org/abs/2106.06898.

[18] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," 2014, https://arxiv.org/abs/1409.1259.

[19] A. Peussa, E.-P. Damskägg, T. Sherson, S. Mimilakis, L. Juvela, A. Gotsopoulos, and V. Välimäki, "Exposure bias and state matching in recurrent neural network virtual analog models," in *Proc. Int. Conf. Digit. Audio Eff. (DAFx-21), Vienna, Austria*, 2021.

[20] S. Petrausch and R. Rabenstein, "A Simplified Design of Multidimensional Transfer Function Models," in *Proc. Int. Workshop Spec. Meth. Multirate Sig. Process.*, Vienna, Austria, 2004, pp. 35–40.

[21] M. Schäfer, P. Frenstátský, and R. Rabenstein, "A Physical String Model with Adjustable Boundary Conditions," in *19th Int. Conf. Digit. Audio Eff. (DAFx-16)*, Brno, Czech Rep., 2016, pp. 159–166.

[22] R. Rabenstein, T. Koch, and C. Popp, "Tubular Bells: A Physical and Algorithmic Model," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 4, pp. 881–890, 2010.

[23] H. Han and V. Lostanlen, "wav2shape: Hearing the Shape of a Drum Machine," http://arxiv.org/abs/2007.10299, 2020.

[24] S. Petrausch and R. Rabenstein, "Simulation of room acoustics via block-based physical modeling with the functional transformation method," in *IEEE Workshop Appl. Sig. Process. Audio Acoust. (WAASPA)*, 2005, pp. 195–198.

[25] M. Schäfer, R. Rabenstein, and S. J. Schlecht, "A String in a Room: Mixed-Dimensional Transfer Function Models for Sound Synthesis," in *23rd Int. Conf. Digit. Audio Eff. (DAFx2020)*, 2020, pp. 24–30.

[26] R. V. Churchill, *Operational Mathematics*, Mc Graw Hill, Boston, Massachusetts, 1972.

[27] J. Blauert and N. Xiang, *Acoustics for Engineers*, Springer, Berlin, Heidelberg, 2008.

[28] H. Kuttruff, *Room Acoustics*, CRC Press, Boca Raton, 6. edition, 2016.

[29] L. Trautmann and R. Rabenstein, "Sound synthesis with tension modulated nonlinearities based on functional transformations," in *Acoust. Music: Theory Appl. (AMTA)*, Montego Bay, Jamaica, 2000.

[30] S. Bilbao, "Modal-type synthesis techniques for nonlinear strings with an energy conservation property," in *7th Int. Conf. Digit. Audio Eff. (DAFx'04)*, Naples, Italy, 2004, pp. 119–124.

[31] F. Avanzini, R. Marogna, and B. Bank, "Efficient synthesis of tension modulation in strings and membranes based on energy estimation," *J. Acoust. Soc. Am.*, vol. 131, pp. 897–906, 2012.

[32] T. Tolonen, V. Välimäki, and M. Karjalainen, "Modeling of tension modulation nonlinearity in plucked strings," *IEEE Trans. Speech Audio Process.*, vol. 8, no. 3, pp. 300–310, 2000.

[33] A. Paszkeand et al., "Pytorch: An imperative style, high-performance deep learning library," in *Adv. Neural Inf. Process. Syst. 32*, pp. 8024–8035. 2019.

[34] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, https://arxiv.org/abs/1711.05101.

[35] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," 2017, https://arxiv.org/abs/1708.07120.