Differentiable Modal Synthesis for Physical Modeling of Planar String Sound and Motion Simulation

Jin Woo Lee [*]
Seoul National University
jinwlee@snu.ac.kr

Jaehyun Park [*]
Seoul National University
lotussoh@snu.ac.kr

Min Jun Choi [*]
Seoul National University
choimj21@snu.ac.kr

Kyogu Lee [*†‡]
Seoul National University
kglee@snu.ac.kr

Abstract

While significant advancements have been made in music generation and differentiable sound synthesis within machine learning and computer audition, the simulation of instrument vibration guided by physical laws has been underexplored. To address this gap, we introduce a novel model for simulating the spatio-temporal motion of nonlinear strings, integrating modal synthesis and spectral modeling within a neural network framework. Our model leverages physical properties and fundamental frequencies as inputs, outputting string states across time and space that solve the partial differential equation characterizing the nonlinear string. Empirical evaluations demonstrate that the proposed architecture achieves superior accuracy in string motion simulation compared to existing baseline architectures. The code and demo are available online. [4]

1 Introduction

The investigation of wave propagation along strings, encompassing both theoretical and experimental dimensions, has persisted for well over a century [1, 2]. In the relentless pursuit of verisimilitude and expressive fidelity in simulating wave phenomena, numerous studies have been investigated to bridge the gap between theoretical underpinnings and empirical sound measurements [3]. Advancements leveraging computational power to mimic the intricate physical processes in musical instruments have given rise to numerical sound synthesis, now a cornerstone in the field of virtual sound synthesis [4, 5]. Schwarz [6] presents a systematic study of parametric and physical models for music audio synthesis. Parametric models include signal models, such as spectral modeling synthesis [7]. Physical models encompass techniques such as modal synthesis, digital waveguides [8, 9], or finite-difference time-domain (FDTD) methods [4, 10].

Over recent years, the advancement of hardware acceleration for artificial intelligence has enabled the emergence of numerous techniques for neural audio synthesis [11], including autoregressive generation [12], adversarial training [13] with phase coherence [14], and approximated physical models [15]. The concept of differentiable digital signal processing (DDSP) was first introduced by Engel et al. [16], aiming to incorporate a known signal model into neural networks to achieve a domain-appropriate inductive bias. While the DDSP model can be considered a differentiable

---

[*]Music and Audio Research Group (MARG), Department of Intelligence and Information
[†]Artificial Intelligence Institute of Seoul National University (AIIS)
[‡]Interdisciplinary Program in Artificial Intelligence (IPAI)
[4]https://huggingface.co/spaces/szin94/dmsp

38th Conference on Neural Information Processing Systems (NeurIPS 2024).
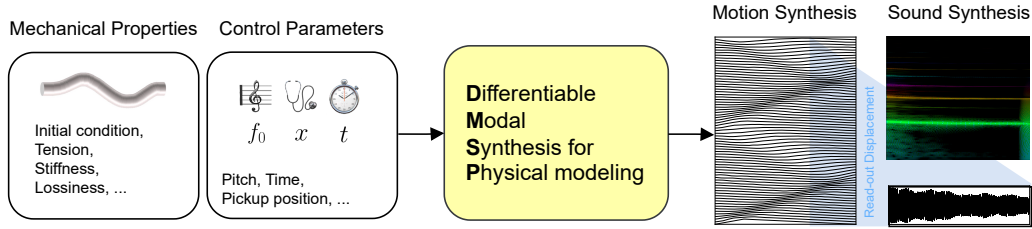
Figure 1: **System overview**. The DMSP model encodes the physical properties of a string (e.g., tension, stiffness, damping, and initial conditions) to estimate the displacement of the string plucked at pitch f_0 at a given time $t \in [0, \infty)$ and position $x \in \Omega$. By concatenating the DMSP outputs over the domain $(x, t) \in \Omega \times [0, \infty)$, the simulated motion of the string can be visualized. Reading the outputs at a particular position x allows hearing the synthesized string sound, akin to listening with a stethoscope at the pickup position.

version of spectral modeling synthesis, a wide variety of works have explored the differentiable implementation of other audio signal processing methods, such as the subtractive method [17], waveshaping [18], and frequency modulation [19, 20]. Subsequent research has demonstrated various applications of DDSP, including music performance synthesis [21], speech synthesis and voice conversion [22, 23], and sound effect generation [24]. Renault et al. [25] extended DDSP to create a polyphonic synthesizer, explicitly modeling properties specific to piano strings, such as inharmonicity and detuning induced by string stiffness, based on a parametric model of these phenomena [26]. This model efficiently synthesizes piano sound from MIDI input, achieving a high mean opinion score on naturalness. However, it still shows room for improvement compared to sampling-based methods [27] and physical modeling methods [28].

Despite the growing recognition of DDSP as a promising sound synthesis methodology, its extension to physical modeling remains underexplored. Schlecht et al. [29] have presented physical modeling using Fourier Neural Operator (FNO). They train recurrent-type FNOs to learn state transitions from data spanning a few initial timesteps in the simulation and then test generalization to the subsequent long-range data. Although they have demonstrated encouraging outcomes, their approach has room for improvement in that it is unconditional, meaning that it is challenging to generalize over dynamic scenarios (*e.g.*, glissando, vibrato). In the context of rigid-body contact sound synthesis [30–32], a few studies investigated the efficacy of training neural networks supervised by finite-element method (FEM) solvers. Jin et al. [33, 34] propose a neural network that predicts contact sounds from voxelized objects, inspired by the modal technique that synthesizes sound using eigenvalues and eigenvectors. Diaz et al. [35] leverage a differentiable infinite impulse response filter to synthesize contact sounds from rasterized occupancy grids in an end-to-end manner. These methods can interactively synthesize sounds for various contact conditions and materials with notable efficiency, circumventing the need for an offline optimization process typical of modal techniques. However, these methods, which resort to the FEM solver, are vulnerable in modeling the dynamic behavior or in simulating the motion of the object.

In this regard, we propose a novel model for simulating the spatio-temporal motion of nonlinear strings, integrating modal synthesis and spectral modeling within a neural network framework. The proposed model leverages physical properties and fundamental frequencies as inputs, outputting string states across time and space that solve the partial differential equation (PDE) characterizing the nonlinear string. Empirical evaluations demonstrate that the proposed architecture achieves superior accuracy in string motion simulation compared to the baseline architectures. The main contributions are as follows:

- We present differentiable modal synthesis for physical modeling (DMSP) that simulates dynamic nonlinear string motion by synthesizing sound using the physical properties of the string.

- To the best of our knowledge, this is the first differentiable approach that can synthesize the motion and the sound of musical strings with a dynamic control over the pitch and the material properties.

- We provide an extensive empirical evaluation demonstrating the importance of modal decomposition and the proper choice of loss function.

2

2 Background

2.1 Physical Modeling of Musical String Instrument

Linear Damped Stiff String. The string model discussed in this paper is a damped nonlinear stiff string. To introduce the nonlinear string, we first formulate the governing equations for the damped linear stiff string system and derive the corresponding modal solution.

$$\partial_{tt}u = \gamma^2 \partial_{xx}u - \kappa^2 \partial_{xxxx}u - 2\sigma_0 \partial_t u \tag{1}$$

The linear string of its length L, vibrating with wave speed γ, stiffness κ, and frequency-independent damping factor σ_0, is described by Equation 1. Given the initial conditions (IC) for $x \in \Omega = [-L/2, +L/2]$ as $u(x,0) = u_0(x)$ and $\partial_t u(x,0) = 0$, and appropriate boundary conditions (BC), the corresponding solution $u(x,t)$ represents the motion of a damped linear stiff string. Particularly, for a clamped boundary condition, *i.e.*, $u(\pm L/2, t) = \partial_x u(\pm L/2, t) = 0$ for all $t \in [0, \infty)$, a modal solution can be obtained as follows.

$$u(x,t) = \sum_{n=1}^{\infty} X_n(x)T_n(t) \tag{2a}$$

$$X_n(x) = c_1 \left(\sin \mu_n x - \frac{\sin \mu_n L/2}{\sinh \nu_n L/2} \sinh \nu_n x \right) + c_2 \left(\cos \mu_n x - \frac{\cos \mu_n L/2}{\cosh \nu_n L/2} \cosh \nu_n x \right) \tag{2b}$$

$$T_n(t) = e^{-\sigma_0 t} \cos \underbrace{\sqrt{\mu_n^4 \kappa^2 + \mu_n^2 \gamma^2 - \sigma_0^2}}_{\omega_n} t \tag{2c}$$

The derivation of Equation 2 can be found in Appendix A. The allowed values of μ_n and ν_n are determined by the boundary conditions, while the coefficients c_1 and c_2 are determined using the initial condition $\sum_{n=1}^{\infty} X_n(x) = u_0(x)$. Determining these values typically requires an offline numerical solving process, where the obtained values can be stored in memory for real-time computation of $u(x,t)$. The stiffness modeled by the 4th order derivative induces a hyperbolic solution, resulting in non-integer multiples of the mode frequencies ω_n, which leads to physical inharmonicity. The damping factor σ_0 causes an exponential decay in the temporal amplitude.

Nonlinear Damped Stiff String. The generalization of the linear wave Equation 1 to nonlinear string vibrations is first introduced by Kirchhoff [36] and Carrier [37]. The Kirchhoff–Carrier system models elastic strings in two dimensions, and when extended so that transverse and longitudinal motions are coupled, phantom partials can be exhibited, resulting in a richer timbre [38]. A model of such planar string vibration is as follows [39, 40].
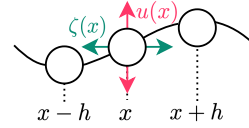
Figure 2: The planar string system.

$$\partial_{tt}u = \gamma^2 \partial_{xx}u - \gamma^2 \frac{\alpha^2 - 1}{2} \partial_x \left(q^3 + 2pq \right) - \kappa^2 \partial_{xxxx}u - 2\sigma_0 \partial_t u + 2\sigma_1 \partial_t \partial_{xx}u \tag{3a}$$

$$\partial_{tt}\zeta = \gamma^2 \alpha^2 \partial_{xx}\zeta - \gamma^2 \frac{\alpha^2 - 1}{2} \partial_x \left(q^2 \right) - 2\sigma_0 \partial_t \zeta + 2\sigma_1 \partial_t \partial_{xx}\zeta \tag{3b}$$

Here, $u(x,t)$ and $\zeta(x,t)$ represent the transverse and longitudinal displacements of a string, respectively, for all $(x,t) \in \Omega \times [0, \infty)$. $q = \partial_x u$ and $p = \partial_x \zeta$ serves as the auxiliary coupling system, as $\partial_t q = \partial_x \partial_t u$ and $\partial_t p = \partial_x \partial_t \zeta$ [41]. A more detailed background on the derivation of Equation 3 can be found in Appendix B. The boundary condition, which may vary depending on the string being modeled, is chosen to be that of the clamped boundary condition:

$$u(x,t) = \partial_x u(x,t) = 0, \quad \forall (x,t) \in \partial\Omega \times [0, \infty). \tag{4}$$

Given an initial condition $u_0 := u(x,0)$ defined on $x \in \Omega$, the solution $u(x,t)$ associated with the condition of Equation 4 simulates the motion of the string for physical modeling and sound synthesis of string instruments. Due to the coupling between Equation 3a and Equation 3b, the obtained solution exhibits features found in elastic strings, such as pitch glide and phantom partials. These features become more pronounced for larger displacements and are difficult to approach separately as in the linear case. The solution can be approximated through various physical modeling techniques such as finite-difference time-domain [42], digital waveguides [43], or functional transformation method [44].

3

Table 1: Comparison between methods. Computational complexity refers to the inference scenario.

Method	System taxonomies			Computational complexity	
	Physical	Nonlinear	Differentiable	Pre-computation	Synthesis
Modal	✓	✗	✗	$\mathcal{O}(N_m)$	$\mathcal{O}(1)$
FDTD	✓	✓	✗	N/A	$\mathcal{O}(N_x N_t)$
DDSP	✗	✓	✓	N/A	$\mathcal{O}(N_r)$
DMSP-Hybrid	✓	✓	✓	$\mathcal{O}(N_m)$	$\mathcal{O}(1)$
DMSP	✓	✓	✓	N/A	$\mathcal{O}(1)$

Finite-difference Time-domain. One straightforward approach to tackling nonlinear PDEs such as Equation 3 would be employing finite difference approximation. This method, commonly known as finite-difference time-domain (FDTD), has a long and distinguished history and is widely accepted in fields such as fluid dynamics [45] and electromagnetics [46]. FDTD is particularly effective in solving nonlinear, multidimensional, and dynamic systems. The extensive literature on its applications encompasses a diverse range of domains, including musical acoustics [10]. A recent contribution by Lee et al. [47] introduces StringFDTD-Torch, an FDTD simulator tailored for modeling planar damped stiff strings akin to Equation 3. Leveraging PyTorch C++ extension, StringFDTD-Torch facilitates FDTD computations on both CPUs and GPUs. However, its current iteration lacks support for gradient backpropagation through the FDTD module, leaving room for enhancement, particularly in optimizing the gradient computation process, which is hindered by the substantial number of temporal recursions involved (evident by the large N_t in $\mathcal{O}(N_x N_t)$ of Table 1).

Modal Synthesis. As a more efficient approach to solve the nonlinear wave equations, the modal synthesis [48–50] decomposes the complex dynamics into contributions from a set of modes, whose spatial bases are eigenfunctions of the pertinent problem. Each mode exhibits distinct oscillations at complex frequencies, contingent upon the boundary conditions. For problems with real-valued parameters, these complex frequencies occur in conjugate pairs, and the "mode" is thus defined as the pair of such eigenfunctions and frequencies [10]. Modal synthesis involves two primary steps. Initially, in an offline phase (also labeled as 'Pre-computation' in Table 1), modal shapes and frequencies are discerned from the PDE system, considering both boundary and initial conditions. This information is encapsulated in what is known as a shape matrix. Subsequently, the solution is derived by combining the modal functions, each progressing at its natural frequency. Pre-computation requires $\mathcal{O}(N_m)$ of recursions because N_m shape matrices need to be computed, but a typical N_m is typically hundreds to thousands of times less for N_t. Modal synthesis after this off-line process is very efficient as it allows us to obtain a solution for a given x and t without any recursion, but it is clear that the range of solutions that can be covered is bounded in that it relies on the ansatz $u(x,t) = X(x)T(t)$ for the separation of variables.

2.2 Differentiable Digital Signal Processing

In the field of neural networks, numerous audio researchers have been engaged in the development of techniques that leverage the ease of automatic gradient backpropagation in neural networks for the purpose of audio parameter estimation. To address the challenge of estimating the latent parameters of a sound, some approaches implement the synthesis part as-is using automatic differentiation package [20, 47] so that the gradient can back-propagate through it to update the parameters directly, while the majority of approaches train neural networks to estimate the parameters in an auto-encoder framework [16, 22, 51]. As one of the most seminal studies of the latter approach, DDSP is widely used to efficiently synthesize nonlinear and dynamic sounds. Based on the spectral modeling synthesis framework, the time-domain signal is modeled via short-time Fourier transforms (STFTs) divided into deterministic (harmonic) and stochastic (noisy) parts to synthesize the sound. As the causality of STFT frames is modeled through gated recurrent units (GRUs), DDSP requires as many recursions as N_r, the number of frames. The harmonics of a DDSP are given by an integer multiple of the fundamental frequency (f_0), and the noise is synthesized from filtered noise. These DDSPs, while still a remarkable advancement, have strong structural constraints on the deterministic part to capture enough perceptually rich tones such as inharmonicity due to stiffness or phantom partials due to nonlinearity. A study by Renault et al. [25] also points this out, and uses the parametric model [26] for inharmonicity and detune, but there is room for improvement as it is an approximation model that relies on instrument-specific modifiers rather than reflecting stiffness physics.

4

|(a) Network architecture of DMSP | (b) AM/FM Block|
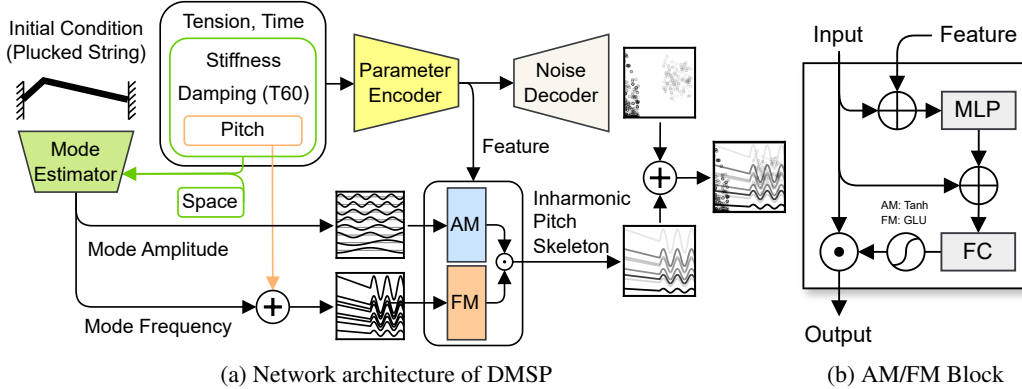
Figure 3: **Network architecture**. DMSP synthesizes a pitch skeleton with an inharmonic structure, drawing upon overtones derived from the modes of the string. The modes can either be derived directly using the modal decomposition (DMSP-Hybrid, the hybrid of DMSP and Modal), or using the neural network trained to estimate the modes (DMSP, the fully-neural-network method). Yet, relying solely on modal frequencies and corresponding shape functions delineates a linear solution, which falls short of capturing the nuances of nonlinear string motion. To address this, DMSP introduces FM and AM blocks to modulate the modes of the linear solution. This modulation process enables DMSP to estimate the pitch skeleton of the nonlinear solution. Consequently, the output waveform is synthesized through the spectral modeling pipeline, incorporating both (in)harmonic components and the filtered noise.

3 Differentiable Modal Synthesis for Physical Modeling (DMSP)

This section introduces a novel differentiable nonlinear string sound synthesizer. Table 1 provides a summary of the methods discussed. While modal synthesis stands out for its efficiency, it is solely applicable to linear models and necessitates pre-computation to determine the number of modes denoted as N_m. On the other hand, FDTD computes highly nonlinear and dynamic solutions but demands a substantial computational load due to iterative updates across both temporal (N_t) and spatial (N_x) samples. In contrast, differentiable audio processing methods offer efficient nonlinear sound synthesis, typically leveraging a smaller number of frames (N_r) compared to the total time samples (N_t). However, they often lack physical controllability. In this regard, we propose DMSP, which approximates the solution of Equation 3 efficiently by leveraging Equation 2 in the form of neural networks. Figure 1 provides a visual depiction of the DMSP.

3.1 Problem Statement

The objective of this study is to establish a mapping from the parameter space to the solution space, utilizing a finite set of observations comprising parameter-solution pairs from this mapping. We delineate this objective as follows: Consider the partial differential equation depicted in Equation 3, applicable for (x, t) within $\Omega \times [0, \infty)$, with clamped boundary conditions as specified in Equation 4, where Ω represents a bounded domain in $\mathbb{R}^{N_x}$. We assume that the solution $u : \Omega \times [0, \infty) \to \mathbb{R}$ resides within the Banach space $\mathcal{U}$. For a given PDE parameter $\rho \in \mathcal{P}$ and initial condition $u_0 \in \mathcal{U}$, let $\mathcal{S} : \mathcal{P} \to \mathcal{U}$ denote a nonlinear map, specifically, the FDTD numerical solver tailored to the context of this study. Assume that we are provided with observations $\{\rho^{(i)}, u^{(i)}\}_{i=1}^N$, where $\rho^{(i)}$ comprises independent and identically distributed (i.i.d.) samples drawn from a probability measure supported on $\mathcal{P}$, and $u^{(i)} = \mathcal{S}(\rho^{(i)})$ potentially contains noise. Our goal is to construct an approximation of $\mathcal{S}$ denoted as $\mathcal{S}_\theta : \mathcal{P} \to \mathcal{U}$, and select parameters $\theta^* \in \mathbb{R}^{N_\theta}$ such that $\mathcal{S}_{\theta^*} \approx \mathcal{S}$. Leveraging $\mathcal{S}_\theta$, one can compute the solution $\hat{u} = \mathcal{S}_\theta(\rho)$ corresponding to a new parameter $\rho \in \mathcal{P}$. By specifying values for x and t, one can then either synthesize the sound of the string picked-up (also referred to as read-out) at a specific location x_0 as $\hat{u}(x_0, t)$, or simulate the motion of the string by concatenating $\hat{u}(x, t)$ across all $x \in \Omega$. In practice, Ω and $[0, \infty)$ are bounded and discretized to form an evenly distributed spatio-temporal grid as $\mathbb{R}^{N_x} \times \mathbb{R}^{N_t} \subset \Omega \times [0, \infty)$, where the spatial grid samples are uniformly distributed in space according to the length of the equally spaced intervals L/N_x and the temporal grid samples uniformly distributed according to the frequency of a fixed audio sampling rate.

5

3.2 Network Architecture

Parameter Encoder. To effectively capture material features inherent in the PDE parameter values, the parameter encoder leverages a random Fourier feature (RFF) layer [52, 53]. Given T60 frequencies $f_{\mathrm{T60}}^{(i)}$ and their corresponding times $t_{\mathrm{T60}}^{(i)}$ for $i = 1, 2$, the frequency-dependent damping coefficients σ_0 (and σ_1, if applicable) are derived using Equation 24. The frequency-independent damping factor $\exp(-\sigma_0 t)$ is computed explicitly multiplied by the mode amplitudes. All PDE parameters $\rho = \{\kappa, \alpha, \sigma_0, \sigma_1\} \in \mathbb{R}^4 \subset \mathcal{P}$ are encoded into a feature vector $h \in \mathbb{R}^{4 \times d}$ with a Fourier embedding size of $d = 256$.

AM and FM Blocks. As illustrated in Figure 3, DMSP employs amplitude modulation (AM) and frequency modulation (FM) modules [5] to modulate the mode frequencies and amplitudes of the linear solution, as depicted in Equation 2, to synthesize the solution of the nonlinear wave described in Equation 3. We utilize two multilayer perceptrons (MLPs) for the modulation layers. Although a simpler and perhaps more conventional choice of architecture would be a GRU, to the decoder architecture of DDSP [16], we choose MLPs for their slightly better empirical results. It's noteworthy that DDSP decodes the sinusoidal frequency envelope with fixed frequency values. In contrast, DMSP decodes both the envelope and the frequency values independently, employing two distinct MLP blocks, namely AM and FM.

Mode Estimator. As detailed in subsection 2.1, determining allowed values for mode frequencies and amplitudes, corresponding to specific initial and boundary conditions of the string, typically involves a root-finding process conducted offline. While these numerical solvers offer high accuracy up to a specified iterative threshold, they necessitate pre-computation, as illustrated in Table 1. The mode estimator module within DMSP estimates the modes from the initial condition using an MLP. The initial condition is parameterized by a pluck position p_x and its peak amplitude p_a. Subsequently, the physical properties κ, γ, σ_0, and σ_1 are encoded using a random Fourier feature (RFF) layer. The mode frequencies and amplitudes are then estimated by the MLP, followed by the application of suitable scaling activations. It's pertinent to note that the mode estimator operates independently and is trained separately from the other modules. During training, the ground truth modes (computed using the modal decomposition method) are fed into the AM and FM blocks, ensuring accurate synthesis while training the synthesis part of the model.

3.3 Loss Function

We employ a combination of four loss terms: (1) waveform ℓ_1 loss ($\mathcal{L}_1$) that measures the L_1 discrepancy between the synthesized waveform and the ground truth waveform, (2) Multi-scale spectral (MSS) loss that captures spectral differences across multiple scales, ensuring fidelity in spectral representation, (3) Pitch loss ($\mathcal{L}_{f_0}$) that penalizes deviations from the ground truth fundamental frequency (f_0), and (4) Mode loss ($\mathcal{L}_m$), which measures the L_1 distance of the mode frequency and mode amplitude from the output of the mode estimator (if applicable) and the mode frequency and mode amplitude obtained via modal decomposition. MSS loss has been adopted as a metric for reconstruction in most neural net-based synthesis techniques [11, 16, 21]. Concerning that measuring MSS with magnitudes is not phase-sensitive by definition, we employ the $\mathcal{L}_1$ loss in the waveform to train the causality in spatio-temporal wave propagations. Challenges in optimizing the frequency parameters of sinusoidal oscillators via gradient descent over the spectral loss functions, due to the non-convex nature of the optimization problem, have been highlighted in various studies [54, 55]. Damped sinusoids offer a workaround for the issue of non-convexity concerning frequency parameters [55], or alternative metrics are proposed to mitigate the risk of falling into bad local minima [56, 57]. We adopt a parameter regression joint training strategy, akin to the pre-training phase of the work by Engel et al. [58]. Among the output mode frequencies, we train the model to match one mode frequency component (denoted by $\hat{f}_0$) to match the fundamental frequency of the target FDTD-simulated audio (f_0) annotated using CREPE [59] as $\mathcal{L}_{f_0} = \|\hat{f}_0 - f_0\|_1$.

---

[5]It is worth noting that there are some subtle differences from the AM/FM techniques commonly used in sound synthesis and the usage of the terminology in this paper. Relying on terminology commonly used in AM/FM technique, AM/FM in this paper can be described analogically as a carrier sinusoid whose center frequency and amplitude are determined by modal decomposition and varied by a non-sinusoidal modulator. While typical AM/FM synthesis also uses non-sinusoidal modulators, the intent of this paper is different in that it borrows these for the purpose of limiting the undesired periodicity in those modulators.

6

Table 2: Comparison between the baselines and the proposed models.

Method	Center frequency	Network architecture		Training configuration			
		AM block	FM block	$\mathcal{L}_1$	MSS	$\mathcal{L}_{f_0}$	$\mathcal{L}_m$
Modal	Mode frequency	✗	✗	N/A	N/A	N/A	N/A
DDSPish-XFM	Integer multiples	✓	✗	✓	✓	✗	✗
DDSPish	Integer multiples	✓	✓	✓	✓	✗	✗
DMSP-Hybrid	Mode frequency	✓	✓	✓	✓	✓	✗
DMSP	Learnt estimates	✓	✓	✓	✓	✓	✓

4 Experiments

4.1 Experimental Setup

Dataset. We use the StringFDTD-Torch [47], the open-source nonlinear string simulator, to compute the solution of the Equation 3 in a temporal sampling rate of 48 kHz and a spatial sampling rate. The solution is upsampled to a spatial resolution of 256 using a bivariate spline approximation over a rectangular spatio-temporal mesh upto the 5th order degree. We simulate 10263 different strings by randomly augmenting the material properties, *e.g.*, κ, α, σ_0, and σ_1, with various plucking profiles u_0. The simulation results in a total amount of 729.8 hours of wave files that corresponds to the 1-sec string sounds picked up at 256 different positions for each string. For the test data, 715 strings are newly synthesized with the parameters sampled in i.i.d. The test data consists of 336 linear ($\alpha = 1$) and 379 nonlinear ($\alpha > 1$) strings, each of which has a 256 spatial grid size, resulting in approximately 50 hours of wave files. Table 5 specifies the range of the sampled PDE parameters.

Baselines. Table 2 compares the major differences between the baselines and the proposed models. As the first attempt to tackle the task of neural audio synthesis for dynamic physical properties, we compare our proposed model to three other models. We consider three baselines, namely: Modal and DDSPish, and DDSPish-XFM. Modal synthesis is the linear wave solution as in Equation 1, where the modal frequencies and the shape functions are pre-computed. DDSPish is a neural network based on the harmonic plus noise model, similar to DDSP. Yet, this -ish suffix emphasizes that this model is different from the DDSP model [16]. DDSPish does not have a reverb module but instead adds frequency modulation, and most notably it has a parameter encoder that allows generating sounds from physical parameters. Please see Appendix D for more details of the baselines.

Evaluation Metrics. We report results on three metrics, signal-distortion-ratio (SDR), scale-invariant signal-distortion-ratio (SI-SDR) [60], multi-scale spectral (MSS) distance, and the pitch difference in Hz. As with the problem statement, we consider the FDTD-simulated results as the ground truth (GT). Both SDR and SI-SDR estimate the distortions on a spatiotemporal grid and are very strict about scoring out-of-phase cases, by directly comparing the the estimated displacements to the FDTD results without any specific transformations or interpolations. Given that the magnitudes of the estimated displacements typically distribute within the small range (approximately between ± 0.02; depends on the pluck amplitude p_a, see Table 5), we compare both with scale normalization (SI-SDR) and without (SDR). The MSS metric for the evaluation is computed using the short-time Fourier transformation (STFT) with three scales of Fast Fourier Transform (FFT) points: 1024, 512, and 256, with a window length equal to the FFT point for each, and a hop length equal to a quarter of that. For each scale, the STFT magnitude is compared on both linear and log scales, weighted by 2.0 and 0.5, respectively. This choice of weighting is to help scale the loss computation with MSS and should not have a significant impact on performance, but is important to note for absolute comparisons of MSS scores. The Pitch metric is computed as the ℓ^1 norm of the difference between $\hat{f}_0$ and f_0.

4.2 Results

Differentiable Sound Synthesis. The efficacy of DMSP is studied as shown in Table 3. First, the modal synthesis method calculates a linear solution for a damped stiff string, and the mode for the linear test set is calculated offline. The discrepancy between the outcomes of modal synthesis and FDTD in the linear case can be attributed to the absence of frequency-dependent damping in Equation 1. In other words, the frequency information serves as an oracle for the Modal, as evidenced by its lowest pitch score, but the decay of amplitude over time is not sufficiently modeled, which is where the remaining models demonstrate superior performance. Considering the DDSPish models,

7

Table 3: Synthesis Results

Model	Linear ($\alpha = 1$)				Nonlinear ($\alpha > 1$)			
	SI-SDR (dB, ↑)	**SDR** (dB, ↑)	**MSS** (dB, ↓)	**Pitch** (Hz, ↓)	**SI-SDR** (dB, ↑)	**SDR** (dB, ↑)	**MSS** (dB, ↓)	**Pitch** (Hz, ↓)
Modal	–3.191	0.681	18.449	**0.420**	–16.611	–1.900	17.254	2.316
DDSPish	–39.478	–2.598	11.047	5.518	–25.951	–2.102	9.745	3.306
DDSPish-XFM	–46.609	–2.257	**10.911**	11.304	–46.858	–2.272	10.299	14.013
DMSP-Hybrid	**–2.844**	**1.496**	12.525	0.792	**15.670**	**16.455**	**4.772**	1.027
DMSP	–22.298	–2.000	12.504	1.717	–10.315	0.221	5.656	**1.437**

(a) $p_x = 0.1$ (b) $p_x = 0.2$ (c) $p_x = 0.3$ (d) $p_x = 0.4$
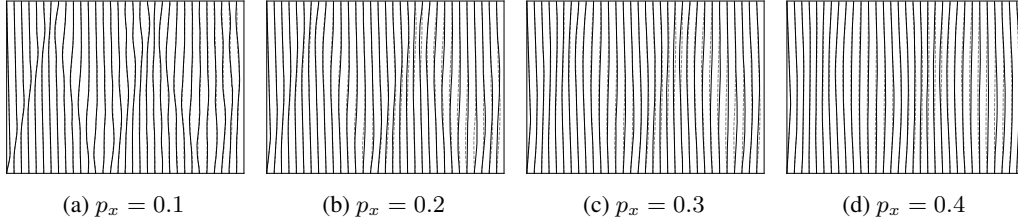
Figure 4: Visualization of the string displacement over time (horizontal) and space (vertical). For different initial conditions, the results synthesized by DMSP are shown as solid black lines and those simulated by FDTD as dashed gray lines.

the MSS is approximately 1.6 dB ahead of the DMSPs, but the difference can reach up to 44 dB in the case of the SI-SDR. It is worth noting that α is uniformly sampled within the range $(1, 25)$ for the training data, falling into the category of the nonlinear strings. Information about the ranges for sampling PDE parameters is in Table 5. In the case of the nonlinear test set, the difference between the modal solution and the FDTD solution becomes larger. On the other hand, DDSPish models, which are based on spectral modeling synthesis but learned without using any modal information, show the lowest performance. The superiority of DMSP is even more pronounced in the nonlinear case. While Modal, which can only cover the solution to the linear case, shows attenuated performance, the DMSPs demonstrate the best performance in all metrics. In particular, DMSP-Hybrid, which precomputes the mode frequency like Modal, performs FM and AM for the nonlinear solution, showing an SI-SDR improvement of nearly 31 dB and an MSS improvement of nearly 13 dB over Modal. DMSP, which estimates the mode as a neural network without the pre-computation step, also outperforms Modal on all metrics for the nonlinear case. The primary reason for the performance discrepancy between the nonlinear case and the linear case for DMSP is that the training data is rarely precisely equal to 1 in the α distribution when it is sampled.

Controllable Physical Simulation. The quantitative scores for various physical condition parameters are visualized in Figure 5. Trends show how the results of Modal synthesis and DMSP vary for different pickup positions (x), stiffness (κ), tension (α), pluck amplitude (p_a), and pluck position (p_x). Of these, α and p_a in particular are known to increase the nonlinearity of the string as they increase in magnitude, which can be seen by the lower Modal synthesis scores. For DMSP, we see an overall improvement in the score, with a lower propensity for nonlinearity. Figure 4 depicts the simulated state of the string as the pluck position in the initial condition is varied. The results synthesized by DMSP can reconstruct a very accurate initial condition, similar to the results simulated by FDTD. The

Figure 5: Objective scores over the change of physical parameters.

vibration propagating through time along the string exhibits a distinct behavior contingent upon the initial condition. FDTD employs a recursive calculation of displacement, necessitating some iterations equal to the number of samples at the audio sampling rate. In contrast, DMSP is capable of obtaining the desired displacement in both time and space simultaneously.

8

| (a) Modal | (b) DDSPish-xFM | (c) DDSPish | (d) DMSP | (e) DMSP-Hybrid | (f) FDTD (GT) |

Figure 6: Spectrogram of the synthesized samples on the test set.

| (a) Modal | (b) DDSPish-xFM | (c) DDSPish | (d) DMSP | (e) DMSP-Hybrid | (f) FDTD (GT) |
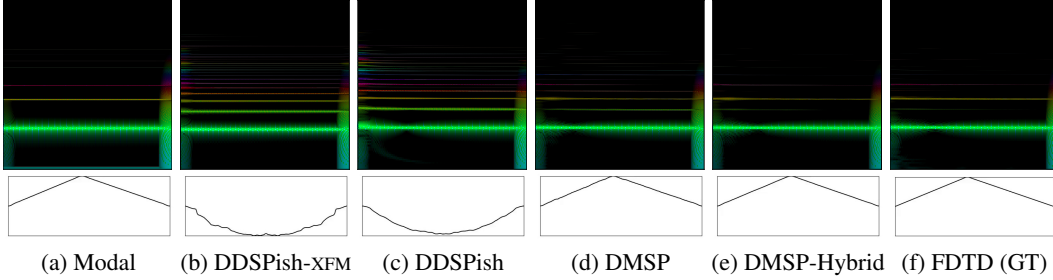
Figure 7: Spectrograms and state samples of the synthesized samples on the test set. For the spectrograms shown in the first column, the intensity of the frequency (vertical axis) component for time (horizontal axis) is expressed as brightness, and for the states shown in the second column, the displacement (vertical axis) for space (horizontal axis).

String Sound Synthesis. Spectrograms of the test samples are visualized in Figure 6 and Figure 7. For the spectrograms, the instantaneous frequencies are identified in a rainbow color map, where the color intensities represent the logarithmic magnitude of the power spectra. Observing from Figure 6, the FDTD-simulated spectrogram clearly shows pitch glide and phantom partials at the beginning of the pluck. In contrast, modal synthesis methods that model linear solutions do not show these nonlinear characteristics. The DDSPish-xFM model employs a harmonic pitch skeleton comprising integer multiples of f_0, thereby precluding the inharmoicities resulting from stiffness. The DDSPish model demonstrates enhanced mode estimation capabilities through learned FM, which modulates the harmonic pitch skeleton to be inharmonic. However, there is scope for further improvement in frequency estimation, particularly in instances where the FM learning process is unstable for high frequencies. On the other hand, the DMSP model, which estimates the mode frequency and amplitude from u_0, shows an improved pitch skeleton and stable frequency estimation. The DMSP-Hybrid model, which learns to AM and FM the sinusoidal oscillators of the modal solution that requires mode precomputation, shows the most similar results to FDTD.

Figure 7 shows a similar trend for the spectrograms. In this example, however, we reveal a key difference that is not apparent in the spectrograms, but is very important for physical modeling: It is the measure of displacement to space, namely the state plot. The timestep for the state plot can be arbitrary, but we show the initial state for ease of comparison. For methods that do not have a separate method for accurately predicting the mode information, unlike DMSP, difficulties can be found in accurately predicting the state. This partially explains the SI-SDR scores for DDSPish and DDSPish-xFM in Table 3. Considering that the major difference between DMSP and DDSPish is whether the input frequency of the FM block is the mode frequency or an integer multiple, it can be inferred that accurately estimating the mode frequency is critical from a physical modeling perspective to fit the displacement of the strings over time.

Ablation Study. The ablation study on the choice of the training loss functions is presented in Table 4. Overall, for linear strings, the performance does not vary much depending on which loss is used, especially for pitch. This is due to a gating applied to the FM block, which is designed to prevent FM from occurring when α is 1. More specifically, we apply a gating, *e.g.*, $\tanh(\alpha - 1)$, in such a way that frequency modulation only occurs when the value of α deviates from 1, which can directly affect pitch, and otherwise forces the mode estimation result to be used as is. This is the main factor that determines the nonlinearity of the string. For AM, there is no such masking depending on α, so the remaining metrics except pitch do vary. For nonlinear data, the training results vary depending on the

9

Table 4: Ablation Study

Model	Linear				Nonlinear			
	SI-SDR (dB, ↑)	SDR (dB, ↑)	MSS (dB, ↓)	Pitch (Hz, ↓)	SI-SDR (dB, ↑)	SDR (dB, ↑)	MSS (dB, ↓)	Pitch (Hz, ↓)
DMSP-Hybrid	−2.844	1.496	12.525	0.792	15.670	16.455	4.772	1.027
w.o. $\mathcal{L}_{f_0}$	−2.919	0.774	13.487	0.792	−5.418	1.509	8.983	2.653
DMSP	−22.298	−2.000	12.504	1.717	−10.315	0.221	5.656	1.437
w.o. $\mathcal{L}_{f_0}$	−21.351	−2.699	13.482	1.717	−16.435	−1.074	9.060	2.922
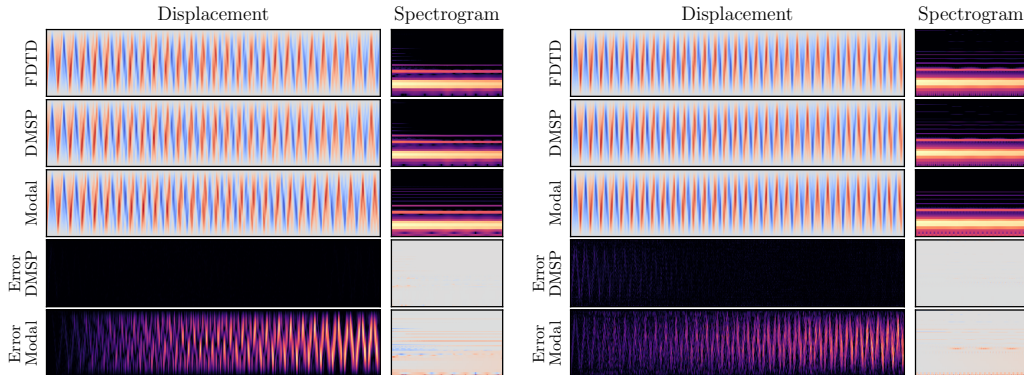
Figure 8: Simulated string state visualization.

loss design. In particular, for all metrics, using $\mathcal{L}_{f_0}$ loss significantly improved performance over not using it. This difference is especially noticeable in the MSS scores between DMSP-Hybrid w.o. $\mathcal{L}_{f_0}$ and DMSP, where DMSP even gives better results over the model that uses precomputed mode frequencies for nonlinear MSS scores if $\mathcal{L}_{f_0}$ is applied. This result reaffirms the validity of the FM block for nonlinear strings in terms of predicting dynamically varying frequencies such as pitch glide and shows that $\mathcal{L}_{f_0}$ loss is the loss function to train it effectively.

Motion Synthesis. The main advantage of DMSP, compared to the existing sound synthesis models, is that it can synthesize not only sound but also motion, which is one of the main characteristics of physical modeling techniques. In particular, the DMSP can visualize the corresponding string motion as a video, although the receptive field and computational complexity required to obtain a solution $u(x, t)$ for a single spatio-temporal point is of order 1, as shown in Table 1, when these solutions are pooled for a given $x \in \Omega$ and $t \in [0, 1)$, the corresponding string motion can be visualized as a video. Figure 8 visualizes the resulting transverse displacement of the string over time (horizontal axis) and space (vertical axis). The transverse displacement of the FDTD is coupled to the longitudinal motion, which is why it differs from the Modal synthesis output, which synthesizes the motion of a linear damped stiff string. The results output by DMSP show improved accuracy.

5 Conclusion

We present a novel neural network-based method that efficiently simulates plucked string motions. Our differentiable modal synthesis for physical modeling (DMSP) can simulate a dynamic nonlinear string motion by synthesizing the sound using the physical properties of the string. It is an efficient approximation of existing physical modeling methods. We demonstrate the efficacy of training the neural network using mode frequency information by extending the DDSP with a modal synthesis pipeline. This opens the door to a new field of differentiable audio signal processing, extending it to the field of physical modeling for musical sound synthesis. While the proposed method offers control over several physical parameters of a musical instrument, it still faces limitations in terms of generalizing to physical parameters and sounds in real-world measurements. This study paves the way for future research in this area. To the best of our knowledge, this is the first study to simultaneously synthesize sound and motion from the properties of a stringed instrument.

10

Acknowledgments

This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) [NO. RS-2023-00219429] and partly by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [NO.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)].

References

[1] John William Strutt Baron Rayleigh. *The theory of sound*, volume 2. Macmillan, 1896.

[2] William Fishburn Donkin. *Acoustics: Theoretical. Part 1*. Clarendon Press, 1884.

[3] Harvey Fletcher. Normal vibration frequencies of a stiff piano string. *The Journal of the Acoustical Society of America*, 36(1):203–209, 1964.

[4] Pierre Michel Ruiz. *A technique for simulating the vibration of strings with a digital computer*. PhD thesis, University of Illinois at Urbana-Champaign, 1970.

[5] Lejaren Hiller and Pierre Ruiz. Synthesizing musical sounds by solving the wave equation for vibrating objects: Part 1. *Journal of the Audio Engineering Society*, 19(6), 1971.

[6] Diemo Schwarz. Corpus-based concatenative synthesis. *IEEE signal processing magazine*, 24 (2):92–104, 2007.

[7] Xavier Serra and Julius Smith. Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition. *Computer Music Journal*, 14(4):12–24, 1990.

[8] Julius O Smith III. *Physical audio signal processing: For virtual musical instruments and audio effects*. 2010.

[9] Alfred Fettweis. Wave digital filters: Theory and practice. *Proceedings of the IEEE*, 74(2): 270–327, 1986.

[10] Stefan Bilbao. *Numerical sound synthesis: finite difference schemes and simulation in musical acoustics*. John Wiley & Sons, 2009.

[11] Ben Hayes, Jordie Shier, György Fazekas, Andrew McPherson, and Charalampos Saitis. A review of differentiable digital signal processing for music and speech synthesis. *Frontiers in Signal Processing*, 3:1284100, 2024.

[12] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alexander Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *Arxiv*, 2016. URL `https://arxiv.org/abs/1609.03499`.

[13] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. In *International Conference on Learning Representations*, 2018.

[14] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. Gansynth: Adversarial neural audio synthesis. In *International Conference on Learning Representations*, 2018.

[15] Jean-Marc Valin and Jan Skoglund. Lpcnet: Improving neural speech synthesis through linear prediction. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5891–5895. IEEE, 2019.

[16] Jesse Engel, Chenjie Gu, Adam Roberts, et al. Ddsp: Differentiable digital signal processing. In *International Conference on Learning Representations*, 2019.

[17] Xin Wang, Shinji Takaki, and Junichi Yamagishi. Neural source-filter waveform models for statistical parametric speech synthesis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:402–415, 2019.

11

[18] Ben Hayes, Charalampos Saitis, and György Fazekas. Neural waveshaping synthesis. *arXiv preprint arXiv:2107.05050*, 2021.

[19] Franco Caspe, Andrew McPherson, and Mark Sandler. DDX7: Differentiable FM Synthesis of Musical Instrument Sounds. *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, 2022.

[20] David Braun. DX7-JAX, November 2023. URL https://github.com/DBraun/DX7-JAX.

[21] Yusong Wu, Ethan Manilow, Yi Deng, Rigel Swavely, Kyle Kastner, Tim Cooijmans, Aaron Courville, Cheng-Zhi Anna Huang, and Jesse Engel. Midi-ddsp: Detailed control of musical performance via hierarchical modeling. In *International Conference on Learning Representations*, 2021.

[22] Hyeong-Seok Choi, Juheon Lee, Wansoo Kim, Jie Lee, Hoon Heo, and Kyogu Lee. Neural analysis and synthesis: Reconstructing speech from self-supervised representations. *Advances in Neural Information Processing Systems*, 34:16251–16265, 2021.

[23] Hyeong-Seok Choi, Jinhyeok Yang, Juheon Lee, and Hyeongju Kim. Nansy++: Unified voice synthesis with neural analysis and synthesis. In *The Eleventh International Conference on Learning Representations*, 2022.

[24] Adrián Barahona-Ríos and Tom Collins. Noisebandnet: controllable time-varying neural synthesis of sound effects using filterbanks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:1573–1585, 2024.

[25] Lenny Renault, Rémi Mignot, and Axel Roebel. Differentiable piano model for midi-to-audio performance synthesis. In *25th International Conference on Digital Audio Effects (DAFx20in22)*, 2022.

[26] François Rigaud, Bertrand David, and Laurent Daudet. A parametric model of piano tuning. In *Proc. of the 14th Int. Conf. on Digital Audio Effects (DAFx-11)*, pages 393–399, 2011.

[27] David Henningsson and FD Team. Fluidsynth real-time and thread safety challenges. In *Proceedings of the 9th International Linux Audio Conference, Maynooth University, Ireland*, pages 123–128, 2011.

[28] Balazs Bank and Juliette Chabassier. Model-based digital pianos: from physics to sound synthesis. *IEEE Signal Processing Magazine*, 36(1):103–114, 2018.

[29] Sebastian Schlecht, Julian Parker, Maximilian Schäfer, and Rudolf Rabenstein. Physical modeling using recurrent neural networks with fast convolutional layers. In *International Conference on Digital Audio Effects*, pages 138–145. DAFx, 2022.

[30] Doug L James, Jernej Barbič, and Dinesh K Pai. Precomputed acoustic transfer: output-sensitive, accurate sound generation for geometrically complex vibration sources. *ACM Transactions on Graphics (TOG)*, 25(3):987–995, 2006.

[31] Jui-Hsien Wang and Doug L James. Kleinpat: optimal mode conflation for time-domain precomputation of acoustic transfer. *ACM Trans. Graph.*, 38(4):122–1, 2019.

[32] James F O'Brien, Chen Shen, and Christine M Gatchalian. Synthesizing sounds from rigid-body simulations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 175–181, 2002.

[33] Xutong Jin, Sheng Li, Tianshu Qu, Dinesh Manocha, and Guoping Wang. Deep-modal: real-time impact sound synthesis for arbitrary shapes. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1171–1179, 2020.

[34] Xutong Jin, Sheng Li, Guoping Wang, and Dinesh Manocha. Neuralsound: learning-based modal sound synthesis with acoustic transfer. *ACM Transactions on Graphics (TOG)*, 41(4):1–15, 2022.

12

[35] Rodrigo Diaz, Ben Hayes, Charalampos Saitis, György Fazekas, and Mark Sandler. Rigid-body sound synthesis with differentiable modal resonators. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[36] Gustav Kirchhoff. *Vorlesungen über mechanik*, volume 1. BG Teubner, 1897.

[37] GF Carrier. On the non-linear vibration problem of the elastic string. *Quarterly of applied mathematics*, 3(2), 1945.

[38] Stefan Bilbao and Michele Ducceschi. Models of musical string vibration. *Acoustical Science and Technology*, 2023.

[39] Philip McCord Morse and K Uno Ingard. *Theoretical acoustics*. Princeton university press, 1986.

[40] GV Anand. Large-amplitude damped free vibration of a stretched string. *The Journal of the Acoustical Society of America*, 45(5):1089–1096, 1969.

[41] Stefan Bilbao. Conservative numerical methods for nonlinear strings. *The Journal of the Acoustical Society of America*, 118(5):3316–3327, 2005.

[42] Stefan Bilbao. Energy-conserving finite difference schemes for tension-modulated strings. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages iv–iv. IEEE, 2004.

[43] Esteban Maestre, Gary P Scavone, and Julius O Smith. Joint modeling of bridge admittance and body radiativity for efficient synthesis of string instrument sound by digital waveguides. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(5):1128–1139, 2017.

[44] Lutz Trautmann and Rudolf Rabenstein. Multirate simulations of string vibrations including nonlinear fret-string interactions using the functional transformation method. *EURASIP Journal on Advances in Signal Processing*, 2004:1–15, 2004.

[45] Parviz Moin and Krishnan Mahesh. Direct numerical simulation: a tool in turbulence research. *Annual review of fluid mechanics*, 30(1):539–578, 1998.

[46] Allen Taflove, Ardavan Oskooi, and Steven G Johnson. *Advances in FDTD computational electrodynamics: photonics and nanotechnology*. Artech house, 2013.

[47] Jin Woo Lee, Min Jun Choi, and Kyogu Lee. String sound synthesizer on gpu-accelerated finite difference scheme. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1491–1495. IEEE, 2024.

[48] Jean-Marie Adrien. The missing link: Modal synthesis. In *Representations of musical signals*, pages 269–298. 1991.

[49] Jean-Marie Adrien and Xavier Rodet. Physical models of instruments: A modular approach, application to strings. In *ICMC*, 1985.

[50] Joseph Derek Morrison and Jean-Marie Adrien. Mosaic: A framework for modal synthesis. *Computer Music Journal*, 17(1):45–56, 1993.

[51] Sungho Lee, Hyeong-Seok Choi, and Kyogu Lee. Differentiable artificial reverberation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:2541–2556, 2022.

[52] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

[53] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

[54] Joseph Turian and Max Henry. I'm sorry for your loss: Spectrally-based audio distances are bad at pitch. In *"I Can't Believe It's Not Better!"NeurIPS 2020 workshop*, 2020.

13

[55] Ben Hayes, Charalampos Saitis, and György Fazekas. Sinusoidal frequency estimation by gradient descent. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[56] Bernardo Torres, Geoffroy Peeters, and Gaël Richard. Unsupervised harmonic parameter estimation using differentiable dsp and spectral optimal transport. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1176–1180. IEEE, 2024.

[57] Simon Schwär and Meinard Müller. Multi-scale spectral loss revisited. *IEEE Signal Processing Letters*, 30:1712–1716, 2023.

[58] Jesse Engel, Rigel Swavely, Lamtharn Hanoi Hantrakul, Adam Roberts, and Curtis Hawthorne. Self-supervised pitch detection by inverse audio synthesis. In *ICML 2020 Workshop on Self-supervision in Audio and Speech*, 2020.

[59] Jong Wook Kim, Justin Salamon, Peter Li, and Juan Pablo Bello. Crepe: A convolutional representation for pitch estimation. In *ICASSP*, pages 161–165. IEEE, 2018.

[60] Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R Hershey. Sdr–half-baked or well done? In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 626–630. IEEE, 2019.

[61] Balázs Bank and László Sujbert. Physics-based sound synthesis of string instruments including geometric nonlinearities, 2006.

[62] Balaz Bank and Laszlo Sujbert. A piano model including longitudinal string vibrations. In *Proceedings of the Digital Audio Effects Conference*, pages 89–94, 2004.

[63] Lutz Trautmann and Rudolf Rabenstein. *Digital sound synthesis by physical modeling using the functional transformation method*. Springer Science & Business Media, 2012.

[64] Maximilian Schäfer, Petr Frenstátskỳ, and Rudolf Rabenstein. A physical string model with adjustable boundary conditions. In *19th International Conference on Digital Audio Effects (DAFx-16), Brno, Czech Republic*, pages 159–166, 2016.

[65] Rudolf Rabenstein, Tilman Koch, and Christian Popp. Tubular bells: A physical and algorithmic model. *IEEE transactions on audio, speech, and language processing*, 18(4):881–890, 2009.

[66] Maximilian Schäfer, Sebastian Schlecht, and Rudolf Rabenstein. A string in a room: Mixed-dimensional transfer function models for sound synthesis. In *International Conference on Digital Audio Effects*, pages 187–194. DAFx, 2020.

14

A Proof of the Linear Damped Stiff String Solution

Proposition 1. *A solution to the damped linear stiff string model Equation 1 with a clamped boundary condition $u(\pm L/2, t) = u_x(\pm L/2, t) = 0$ and the initial condition given as $u(x, 0) = u_0(x)$ and $\partial_t u(x, 0) = 0$ can be expressed as*

$$u(x, t) = \sum_{n=1}^{\infty} X_n(x) T_n(t)$$

$$X_n(x) = c_1 \left(\sin \mu_n x - \frac{\sin \mu_n L/2}{\sinh \nu_n L/2} \sinh \nu_n x \right) + c_2 \left(\cos \mu_n x - \frac{\cos \mu_n L/2}{\cosh \nu_n L/2} \cosh \nu_n x \right)$$

$$T_n(t) = e^{-\sigma_0 t} \cos \left(\sqrt{\mu_n^4 \kappa^2 + \mu_n^2 \gamma^2 - \sigma_0^2} \right) t$$

Proof. As it is hinted in Equation 2a, the procedure to derive Equation 2b and Equation 2c uses the method of separation of variables. The derivation consists of three main steps that start by trying the ansatz $u(x, t) = X(x)T(t)$. Substituting this ansatz into the Equation 1 gives

$$\gamma^2 \frac{X''}{X} - \kappa^2 \frac{X^{(4)}}{X} = \frac{T''}{T} + 2\sigma_0 \frac{T'}{T} = \begin{cases} \varsigma \\ 0 \\ -\varsigma \end{cases} \tag{5}$$

with $\varsigma \in \mathbb{R}$ as the separation constant.

1. **Solving for T.**
$$T'' + 2\sigma_0 T \pm \varsigma^2 T = 0 \tag{6}$$
Roots of the characteristic polynomial of this equation are $\beta_\pm = -\sigma_0 \pm \sqrt{\sigma_0^2 \mp \varsigma^2}$. Three solutions are available:

   - Overdamping ($\sigma_0^2 > \varsigma^2$): $T = A_1 e^{\beta_+ t} + A_2 e^{\beta_- t}$
   - Critical damping ($\sigma_0^2 = \varsigma^2$): $T = (A_1 + A_2 t) e^{-\sigma_0 t}$
   - Underdamping ($\sigma_0^2 < \varsigma^2$): Rewrite the roots as $\beta_+ = -\sigma_0 + i\omega$ and $\beta_- = -\sigma_0 - \hat{\omega}$ with $\omega = \sqrt{\varsigma^2 - \sigma_0^2}, \hat{\omega} = \sqrt{\varsigma^2 + \sigma_0^2}$, then
   $$T = e^{-\sigma_0 t} (A_1 e^{i\omega t} + A_2 e^{-\hat{\omega} t}) \tag{7}$$
   The initial condition $\partial_t u(x, 0) = 0$ implies $A_2 = 0$ yielding the real solution of the form $T = A_1 e^{-\sigma_0 t} \cos \omega t$. This implies that the only valid root is $\beta_+ = -\sigma_0 + i\omega$, hence it is enough to consider the $-\varsigma$ case only, in the Equation 5.

   This work only considers the underdamped case where the σ_0 is sufficiently small, as it should be in a reasonable string model for any musical purposes.

2. **Solving for X.**
$$X^{(4)} - \frac{\gamma^2}{\kappa^2} X'' - \frac{\varsigma^2}{\kappa^2} X = 0 \tag{8}$$
Substitute $l = \gamma^2/2\kappa^2$ and $m = \varsigma^2/\kappa^2$. The roots of the characteristic polynomial for this equation are $\pm\sqrt{l \pm \sqrt{l^2 + m}}$ where $l - \sqrt{l^2 + m} \leq 0$. Therefore the solution is rewritten as
$$X(x) = \underbrace{B_1 \sin \mu x + B_2 \sinh \nu x}_{\text{odd function}} + \underbrace{B_3 \cos \mu x + B_4 \cosh \nu x}_{\text{even function}} \tag{9}$$
where $\mu = \sqrt{\sqrt{l^2 + m} - l}$ and $\nu = \sqrt{\sqrt{l^2 + m} + l}$. By applying the boundary condition $u(L/2, t) = 0$ to the even and odd functions each, then obtain
$$B_2 = -\frac{\sin \mu \frac{L}{2}}{\sinh \nu \frac{L}{2}} B_1 \quad \text{and} \quad B_4 = -\frac{\cos \mu \frac{L}{2}}{\cosh \nu \frac{L}{2}} B_3 \tag{10}$$
reducing the solution as
$$X(x) = B_1 \left(\sin \mu x - \frac{\sin \mu \frac{L}{2}}{\sinh \nu \frac{L}{2}} \sinh \nu x \right) + B_3 \left(\cos \mu x - \frac{\cos \mu \frac{L}{2}}{\cosh \nu \frac{L}{2}} \cosh \nu x \right). \tag{11}$$

15

3. **Finding allowed values that satisfy the conditions.**
   Substituting the ansatz by Equation 7 and Equation 11 summarizes the solution as

$$u(x,t) = e^{-\sigma_0 t} \cos\left(\sqrt{\mu^4 \kappa^2 + \mu^2 \gamma^2 - \sigma_0^2} \cdot t\right)$$

$$\times \left[c_1 \left(\sin \mu x - \frac{\sin \mu \frac{L}{2}}{\sinh \nu \frac{L}{2}} \sinh \nu x \right) + c_2 \left(\cos \mu x - \frac{\cos \mu \frac{L}{2}}{\cosh \nu \frac{L}{2}} \cosh \nu x \right) \right] \quad (12)$$

As mentioned earlier, the coefficients $c_1 = A_1 B_1$ and $c_2 = A_1 B_3$ are determined by the allowed values that satisfy the initial condition $\sum_{n=1}^{\infty} X_n(x) = u_0(x)$. Yet, this is usually preceded by obtaining the μ and ν values. To determine the allowed values of μ, apply the boundary condition $u(L/2, t) = u_x(L/2, t) = 0$ to the even function that gives

$$B_3 \cos \mu \frac{L}{2} = -B_4 \cosh \nu \frac{L}{2} \quad \text{and} \quad -\mu B_3 \sin \mu \frac{L}{2} = -\nu B_4 \sinh \nu \frac{L}{2} \quad (13)$$

or alternatively,

$$\mu \tan \mu \frac{L}{2} = -\nu \tanh \nu \frac{L}{2}. \quad (14)$$

Similarly, applying the same boundary condition to the odd function gives

$$B_3 \sin \mu \frac{L}{2} = -B_4 \sinh \nu \frac{L}{2} \quad \text{and} \quad \mu B_3 \cos \mu \frac{L}{2} = -\nu B_4 \cosh \nu \frac{L}{2} \quad (15)$$

or equivalently,

$$\nu \tan \mu \frac{L}{2} = \mu \tanh \nu \frac{L}{2}. \quad (16)$$

By finding the values of μ and $\nu = \sqrt{\mu^2 + 2l}$ that satisfy the Equation 14 and Equation 16 as μ_n and ν_n, one can determine the allowed values of $m_n = (\mu_n^2 + l)^2 - l^2$ and $\varsigma_n = \sqrt{m_n \kappa^2}$.

Following these three steps gives the n-th mode oscillation of $u(x,t)$ as $X_n(x)T_n(t)$ where

$$X_n(x) = c_1 \left(\sin \mu_n x - \frac{\sin \mu_n L/2}{\sinh \nu_n L/2} \sinh \nu_n x \right) + c_2 \left(\cos \mu_n x - \frac{\cos \mu_n L/2}{\cosh \nu_n L/2} \cosh \nu_n x \right) \quad (17)$$

and

$$T_n(t) = e^{-\sigma_0 t} \cos\left(\sqrt{\mu_n^4 \kappa^2 + \mu_n^2 \gamma^2 - \sigma_0^2} \right) t. \quad (18)$$

Hence, the modal solution is expressed as a superposition of modes, where the infinite summation as in Equation 2a finalizes the modal expression of the modal solution. ∎

B Nonlinear Damped Stiff String Vibration

The Kirchhoff–Carrier model stands out as one of the simplest representations of nonlinear distributed strings, as it effectively captures the pitch glide effect and can be easily simulated using relatively straightforward finite difference schemes. However, it is also known that only the transverse motion is explicitly accounted for, so the longitudinal motion is averaged across the length of the string. Obviously, there are reports that this approximation may suffice under specific conditions [61]. On the other hand, it is also reported that the interaction between longitudinal and transverse motion can result in generating rich timbres with perceptually crucial effects such as phantom partials [38]. A comprehensive model of string vibration, which incorporates both longitudinal and transverse motion within a single plane (in dimensional form), is outlined as follows:

$$\rho A \partial_{tt} u = EA \partial_{xx} u - (EA - T_0) \partial_x \left(\frac{\partial \Phi}{\partial(\partial_x u)} \right) \quad (19a)$$

$$\rho A \partial_{tt} \zeta = EA \partial_{xx} \zeta - (EA - T_0) \partial_x \left(\frac{\partial \Phi}{\partial(\partial_x \zeta)} \right) \quad (19b)$$

The constants ρ and T_0 are the material density and tension, respectively. In order to model the stiffness, Young's modulus E and the cross-sectional area A are introduced.

16

The function Φ, which nonlinearly connects the two equations, is defined as follows:

$$\Phi = \sqrt{(1 + \partial_x\zeta)^2 + \partial_x u^2} - 1 - \partial_x\zeta \tag{20}$$

Note that the final term $-1 - \partial_x\zeta$ does not affect the dynamics of system Equation 19; it is included solely to adjust the zero-point energy of the entire system. Substituting $x' = x/L$, $u' = u/L$, and $\zeta' = \zeta/L$ into the above system, and then removing the primes, one obtains:

$$\partial_{tt}u = \gamma^2\partial_{xx}u - \gamma^2(\alpha^2 - 1)\partial_x\left(\frac{\partial\Phi}{\partial q}\right) \tag{21a}$$

$$\partial_{tt}\zeta = \gamma^2\alpha^2\partial_{xx}\zeta - \gamma^2(\alpha^2 - 1)\partial_x\left(\frac{\partial\Phi}{\partial p}\right) \tag{21b}$$

where the parameters γ and α are defined by

$$\gamma = \frac{1}{L}\sqrt{\frac{T_0}{\rho A}} \qquad \text{and} \qquad \alpha = \sqrt{\frac{EA}{T_0}}. \tag{22}$$

For Φ, one has $\Phi(p, q) = \sqrt{(1 + p)^2 + q^2} - 1 - p$. Series approximations have been instrumental in analyzing nonlinear systems like the string model mentioned earlier; approximations up to third or fourth order are frequently used [39]. The function $\Phi(p, q)$ can be approximated with a variety of orders in p and q:

$$\Phi_2 = \frac{1}{2}q^2 \qquad \Phi_3 = \frac{1}{2}q^2 - \frac{1}{2}pq^2 \qquad \Phi_4 = \frac{1}{2}q^2 - \frac{1}{2}pq^2 + \frac{1}{2}q^2p^2 - \frac{1}{8}q^4$$

In this paper, we consider the following approximation to $\Phi(p, q)$, similar to Φ_4, but lacking one of the fourth-order terms, following Bilbao [10].

$$\Phi_4^* = \frac{1}{2}q^2 - \frac{1}{2}pq^2 - \frac{1}{8}q^4$$

Under this choice Φ_4^*, the system Equation 21 reduces to

$$\partial_{tt}u = \gamma^2\alpha^2\partial_{xx}u - \gamma^2(\alpha^2 - 1)\partial_x\left(q^3 + 2pq\right) \tag{23a}$$

$$\partial_{tt}\zeta = \gamma^2\alpha^2\partial_{xx}\zeta - \gamma^2(\alpha^2 - 1)\partial_x\left(q^2\right) \tag{23b}$$

which can be augmented by the stiffness and the damping term to become Equation 3. The nonlinear damped stiff string system as Equation 3 is very similar to the work by Bank and Sujbert [62] but with the damping coefficients given as the same values for the longitudinal and transverse directions.

It is also noteworthy that there are many approaches to modeling the nonlinear string vibration, other than the aforementioned Kirchhoff–Carrier-style one. A good example is the Functional Transformation Method (FTM), which is a powerful method for modeling the oscillation of acoustic systems in terms of transfer functions. Interested readers are advised to read its derivation and applications from [63–66]. Those with a background in neural network frameworks may also find reading [29] particularly engaging, as Schlecht et al. [29] points out the connection between the underlying mathematical ideas in the FTMs and the FNOs.

C   Damping coefficient

The damping coefficients σ_0 and σ_1 are typically determined experimentally, especially those for nonlinear systems. However, it can be somewhat difficult to set the values without any prior knowledge of the observations. In this regard, the practical settings for the damping coefficients can be approximately derived from the decay times, by resorting to the values for linear systems. The linear string damping coefficients σ_0 and σ_1 are derived from the (frequency-dependent) T60 values: where $f_{\text{T60}}^{(1)}$ and $f_{\text{T60}}^{(2)}$ denotes the two distinct frequencies, and $t_{\text{T60}}^{(1)}$ and $t_{\text{T60}}^{(2)}$ denotes the corresponding decay times. In this case,

$$\sigma_0 = \frac{6\log(10)}{\xi_1 - \xi_2}\left(\frac{\xi_1}{t_{\text{T60}}^{(2)}} - \frac{\xi_2}{t_{\text{T60}}^{(1)}}\right), \qquad \sigma_1 = \frac{6\log(10)}{\xi_1 - \xi_2}\left(\frac{1}{t_{\text{T60}}^{(1)}} - \frac{1}{t_{\text{T60}}^{(2)}}\right), \tag{24}$$

where $\xi_i = -\gamma^2 + \sqrt{\gamma^4 + 4\kappa^2 \times (2\pi f_{\text{T60}}^{(i)})^2}$ for $i = 1, 2$. For this model, as the energy loss increases monotonically with frequency, one must choose $t_{\text{T60}}^{(1)} \geq t_{\text{T60}}^{(2)}$ when $f_{\text{T60}}^{(1)} < f_{\text{T60}}^{(2)}$.

17
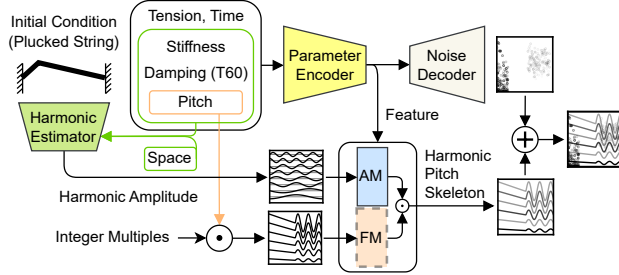
Figure 9: The DDSPish is designed in a similar way to DMSP,

D   Baselines

In this section, we present more details of the baselines. All models, including DMSP, are trained using RAdam optimizer, with Noam learning rate scheduler with a peak learning rate of 10^{-3} reaching at 1000 number of warmup steps. Figure 9 summarizes the architectures of DDSPish models. The only architectural difference between DDSPish and DDSPish-XFM is the existence of the FM module. DDSPish-XFM is *harmonic* as the mode frequencies are not modulated after its initialization by the integer multiples of f_0, while DDSPish is *inharmonic* as FM is trained to modulate the harmonic pitch skeleton to match the inharmonic mode frequencies.

For the Modal synthesis, we compute the solution Equation 1 with the allowed values of μ_n, ν_n, c_1, and c_2 obtained using the Levenberg–Marquardt algorithm. We compute the modes up to the 100th order with the double-precision floating-point arithmetics. The modes are then post-processed to be cut under the Nyquist limit. Subsequently, all models are trained using 40 numbers of modes. We use 65 number of bands used for the filtered noise.

Table 5: PDE parameter sampling range

	Min.	Max.	Unit
f_0	98.00	440.0	Hz
κ	0.01γ	0.03γ	-
α	1	25	-
$t_{\mathrm{T60}}^{(1)}$	10	25	sec
$t_{\mathrm{T60}}^{(2)}$	10	30	sec
$f_{\mathrm{T60}}^{(1)}$	1100	1200	Hz
$f_{\mathrm{T60}}^{(2)}$	100	$f_{\mathrm{T60}}^{(1)} - 1000$	Hz
p_a	0.001	0.02	-
p_x	0.1	0.5	-

E   Datasets

As described in subsection 4.1, this paper utilizes the nonlinear string simulator, StringFDTD-Torch, presented by Lee et al. [47]. For the simulation, PDE parameters are uniformly random-sampled within the moderate parameter ranges. Table 5 summarizes the infimum and the supremum values for each uniform distribution of the PDE parameter. For a given random-sampled parameter set, the simulator outputs the transverse and the longitudinal solutions (u and ζ, resp.) while this work considers u only. Yet, it is worth mentioning that the adopted u is different from that of the one-dimensional string since u and ζ are *coupled* as evident in Equation 3. The obtained u is defined over a spatio-temporal grid, where the spatial resolutions are carefully chosen to mitigate the numerical stability criteria and the numerical dispersions while keeping the temporal resolution by the prefixed audio sampling rate. As the grid spacing is fixed by these values, the raw simulation data are consisted of diverse grid sizes depending on f_0, κ, and α, making it difficult to batchify for training. For this reason, we spatially upsample the data to a fixed spatial grid size of 256, using a bivariate spline approximation over a rectangular spatio-temporal mesh up to the 5th order degree.

18

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We reflect the paper's contributions and scope to the main claims.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in the conclusion section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

19

Justification: We included the assumptions and proofs in the manuscript.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We tried our best to fully disclose all the information needed to reproduce the results, and are willing to supplement any additional information, if needed.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

20

Answer: [Yes]

Justification: We provide code and data attached in the link at the footnote.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We specify the training and test details.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report the experimental results using appropriate statistics of the measured scores.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

21

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provide the computational resources for the dataset construction and the computation.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: The research conducted in this paper conform with the NeurIPS code of ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: The research proposes neural network for the physical simulation of musical instrument for scientific computing.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

22

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper does not poses such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We tried our best to include as many proper credits for the original owners as possible by citing the original works. We will include the missing credits if any.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

23

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We documented the code and its usage properly, and will maintain the documentation.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

24