FAST DIFFERENTIABLE MODAL SIMULATION OF NON-LINEAR STRINGS, MEMBRANES, AND PLATES

Rodrigo Diaz and Mark Sandler *

Centre for Digital Music
Queen Mary University of London, UK
`r.diazfernandez mark.sandler`

ABSTRACT

Modal methods for simulating vibrations of strings, membranes, and plates are widely used in acoustics and physically informed audio synthesis. However, traditional implementations, particularly for non-linear models like the von Kármán plate, are computationally demanding and lack differentiability, limiting inverse modelling and real-time applications. We introduce a fast, differentiable, GPU-accelerated modal framework built with the JAX library, providing efficient simulations and enabling gradient-based inverse modelling. Benchmarks show that our approach significantly outperforms CPU and GPU-based implementations, particularly for simulations with many modes. Inverse modelling experiments demonstrate that our approach can recover physical parameters, including tension, stiffness, and geometry, from both synthetic and experimental data. Although fitting physical parameters is more sensitive to initialisation compared to methods that fit abstract spectral parameters, it provides greater interpretability and more compact parameterisation. The code is released as open source to support future research and applications in differentiable physical modelling and sound synthesis.

1. INTRODUCTION

The accurate modelling of vibrating structures, such as strings, membranes, and plates, is crucial to many areas, including acoustics, musical instrument design, and physically based audio synthesis. These systems are governed by partial differential equations (PDEs) derived from first principles. However, solving these PDEs analytically is possible only in a few specific cases. In practical settings, we therefore rely heavily on numerical approaches, each with distinct strengths and weaknesses.

Finite-difference time-domain (FDTD) methods are often used due to their explicit numerical formulation, discretising both time and space with finite differences. While intuitive and versatile, FDTD simulations often face significant limitations, such as numerical dispersion, restrictive stability conditions, and computational inefficiency, particularly for high-resolution applications [1].

Modal methods provide a different perspective by projecting the problem onto a carefully chosen set of orthogonal basis functions, transforming the PDE into a collection of simpler ordinary differential equations (ODEs). This approach offers advantages by naturally capturing the vibration modes of the system and reducing

computational complexity. However, a significant challenge with modal methods is identifying appropriate basis functions, as these are dependent on the domain geometry and boundary conditions. Since analytical bases are available only for a limited number of simple cases [2], extending modal approaches to more complex scenarios remains difficult.

Additionally, traditional numerical methods, including most modal approaches, are generally implemented without automatic differentiation, complicating the computation of gradients with respect to physical parameters. This limits their practical use for modern inverse modelling and gradient-based parameter estimation tasks, which are increasingly common in physically informed machine learning and audio synthesis. Furthermore, existing implementations typically run on CPUs, making them computationally inefficient for larger or non-linear problems, such as simulations involving the von Kármán plate model.

To address these challenges, this paper introduces a fast and differentiable GPU-accelerated framework for modal simulations of strings, membranes, and plates. Our implementation leverages the JAX library to enable efficient computation and automatic differentiation, making gradient-based optimisation directly accessible for parameter estimation, inverse modelling, and sound synthesis.

- **Scalable computational efficiency:** Our GPU-based implementation significantly outperforms existing CPU and GPU implementations, especially for simulations involving many modes, making large-scale dataset generation and real-time applications feasible.

- **Unified and accessible implementation of multiple models:** Our framework provides a single, user-friendly Python interface to simulate strings, membranes, and plates, including non-linear models like the von Kármán, Berger and Kirchhoff-Carrier models.

- **Physically interpretable inverse modelling:** We demonstrate inverse modelling by optimizing directly for physical parameters (e.g., tension, stiffness, geometry), rather than traditional spectral parameters (poles and zeros). Although this adds complexity due to increased constraints and sensitivity to initialization, it significantly improves interpretability and compactness of representation.

The open-source implementation provided with this paper connects classical physics-based modelling with modern automatic differentiation and GPU acceleration techniques. It offers audio researchers and practitioners an accessible, efficient, and interpretable tool for real-time synthesis, instrument design, and physically informed machine learning.

<   **134**   >

2. RELATED WORK

The simulation and modelling of strings, membranes, and plates has been extensively studied in acoustics and musical instrument research. Notably, the work of Bilbao [3] provides a comprehensive numerical treatment of such systems, primarily focusing on finite difference time-domain (FDTD) approaches. These methods discretise space and time, approximating derivatives using finite differences, and have been widely applied to the simulation of musical instruments and other acoustical systems. While FDTD methods are flexible and conceptually straightforward, they suffer from numerical dispersion and have restrictive stability conditions (e.g., CFL) that demand very small time steps. These constraints make high-frequency simulations or simulations involving large spatial domains computationally expensive. Recently, techniques like the Scalar Auxiliary Variable (SAV) method have also been introduced, primarily within FDTD frameworks, to improve the efficiency and stability of handling non-linear terms by avoiding implicit equations [4, 5].

Alternatively, modal approaches have been explored in different ways. Pseudospectral methods, which exploit Fourier decompositions, have been used for the simulation of the tension modulated string [6]. Another significant modal approach is the Functional Transformation Method (FTM), introduced by Trautmann and Rabenstein [2]. This technique relies on analytical spatial modes and the Laplace transform. By applying both transformations, the PDE becomes a set of transfer functions, which are then discretised and used in a numerical time integration scheme. However, analytical modal expansions exist only for specific geometries and boundary conditions. Subsequent work has focused on the treatment of more complex boundary conditions [7]. In parallel, efficient tension-modulated modal models for strings and membranes have also been developed in the work of Avanzini and Marogna [8, 9].

In the specific context of plates, the non-linear von Kármán model has been extensively studied through modal formulations. The work of Duccheschi and Touzé [10, 11] addresses this non-linear model, particularly for the simply supported boundary condition. While the transverse (out-of-plane) vibration modes can be derived analytically for some cases, accurately capturing the non-linear coupling with in-plane modes typically requires a Galerkin type approach. Such methods select appropriate basis functions for projection onto a system of ODEs with coupling coefficients, enabling efficient numerical integration despite increased complexity due to non-linear mode coupling.

In recent years, differentiable and neural-based approaches have emerged, primarily targeting rigid-body simulations [12, 13, 14]. Although many of these methods are primarily data-driven (rather than physics-based), they often leverage principles similar to classical modal expansions. In [15], non-linear modes are computed using gradient descent, and these modes are then evolved linearly, with a second neural network refining the dynamics to account for non-linear interactions. Similarly, Lee [16] proposes using analytical (or alternatively learnt) modes as a basis, modulating them with neural networks to model pitch glide and non-linear coupling effects. Alternatively, operator learning methods, such as the Fourier Neural Operator (FNO) [17], have also been applied within recurrent neural networks [18] for the simulations of strings and membranes. Collectively, these hybrid approaches highlight the potential benefits of integrating physics-based modal techniques with neural methods.

Our framework remains entirely physics-based and does not depend on neural networks for modelling. Instead, we implement traditional modal formulations in a unified, differentiable manner, leveraging automatic differentiation and GPU acceleration to improve computational efficiency. Due to its differentiability and modular design, the framework can readily be combined with neural networks if desired, enabling hybrid physics-based and data-driven approaches. By offering accessible implementations of tension-modulated string, membrane, and von Kármán plate models, our method simplifies gradient-based inverse modelling and efficient differentiable simulations, effectively bridging classical modal techniques with modern automatic differentiation methods.

3. BACKGROUND

Assuming uniform material properties, the governing equations for the tension-modulated string, membrane, and von Kármán plate with tension can be written as:

$$\rho\ddot{w} + (d_1 + d_3\Delta)\,\dot{w} + (D\Delta\Delta - T_0\Delta)w = f_{\text{ext}} - f_{\text{nl}}, \quad (1)$$

where w is the transverse displacement, ρ is the mass per unit length or area, d_1 and d_3 are the damping coefficients, D is the bending stiffness, T_0 is the initial tension, f_{ext} is the external force and f_{nl} is the non-linear term. Δ and $\Delta\Delta$ stand for the Laplacian and biharmonic operators, respectively.

For the modal approach, a suitable set of eigenfunctions must be found by solving the spatial eigenvalue problem:

$$\Delta\Phi_\mu = -\lambda_\mu\Phi_\mu, \quad (2)$$

where Φ_μ is the μ-th eigenfunction and λ_μ is the corresponding eigenvalue.

In general, analytical solutions for these eigenpairs are known only for simple geometries and boundary conditions. For more complex scenarios, numerical methods such as the Finite Element Method (FEM) or other Galerkin-type approaches are typically employed.

In the linear case, the non-linear term f_{nl} in eq. (1) vanishes and the equation can be solved as a system of uncoupled ODEs. In the tension-modulated case, that is the Kirchhoff-Carrier model in the case of a string and Berger model in the case of a membrane, the non-linear term arises due to the tension caused by the deformation of the string or membrane [8, 19, 3]:

$$f_{\text{nl}} = T_{\text{nl}}\Delta w, \quad (3)$$

$$T_{\text{nl}} = \tau \int_{\mathcal{D}} ||\nabla w||^2 d\mathbf{x} \quad (4)$$

where $\mathcal{D}$ is domain length or area and τ is a scalar dependent on the physical parameters of the string or membrane[1]:

$$\tau_{\text{string}} = \frac{EA}{2L}, \quad \tau_{\text{membrane}} = \frac{Eh}{2L_x L_y(1 - \nu^2)} \quad (5)$$

where E is the Young's modulus, A is the cross-sectional area, L is the length of the string, h is the thickness, L_x and L_y are the length and width, and ν is the Poisson's ratio.

---

[1]For brevity we focus on the rectangular membrane however the same principle applies to the circular membrane.

< **135** >

For the von Kármán plate, nonlinearities emerge from the coupling between transverse and in-plane modes [10, 11]:

$$f_{nl} = -\mathcal{L}(w, \varphi), \tag{6}$$

$$\Delta\Delta\varphi = -\frac{Eh}{2}\mathcal{L}(w, w), \tag{7}$$

where $\mathcal{L}$ is the von Kármán operator defined as:

$$\mathcal{L}(f, g) = \Delta f \Delta g - \nabla\nabla f : \nabla\nabla g, \tag{8}$$

and φ is the Airy stress function.

3.1. Modal expansion of the non-linear term

For tension-modulated strings and membranes, the non-linear term expanded in modal coordinates is given by:

$$\bar{f}_{\mu,nl} = \lambda_\mu q_\mu \tau \left(\sum_\mu \frac{\lambda_\mu q_\mu^2}{||\Phi_\mu||^2} \right), \tag{9}$$

where q_μ is the amplitude of the μ-th modal coordinate.

For the von Kármán plate, both the transverse modes Φ_μ and in-plane modes Ψ_μ must be considered. The non-linear modal expansion then becomes:

$$\bar{f}_{\mu,nl} = \frac{E}{2\rho_{\mathrm{m}}||\Phi_\mu||^2} \sum_{p,q,r}^n \frac{H_{q,r}^n C_{p,n}^s}{\zeta_n^4} q_p q_q q_r, \tag{10}$$

where $H_{q,r}^n$ and $C_{p,n}^s$ are third-order tensors with the coupling coefficients obtained from the projections between the modes:

$$H_{i,j}^k = \frac{\int_S \Psi_k \mathcal{L}(\Phi_i, \Phi_j)\, \mathrm{d}S}{||\Psi_k||\, ||\Phi_i||\, ||\Phi_j||}, \tag{11}$$

$$C_{i,j}^s = \frac{\int_S \Phi_s \mathcal{L}(\Phi_i, \Psi_j)\, \mathrm{d}S}{||\Phi_s||\, ||\Phi_i||\, ||\Psi_j||}, \tag{12}$$

where Ψ and ζ^4 are the eigenfunctions and eigenvalues obtained from the separate eigendecomposition of the Airy stress function φ. Note that the ρ_m denominator in eq. (10) is the mass density and not the material density $\rho = \rho_m h$. For a more detailed derivation of the modal expansion of the non-linear term, we refer the reader to [10, 11].

3.2. Integration in time

Expressing Equation (1) in modal coordinates yields a system of ODEs describing individual modes as damped harmonic oscillators:

$$\ddot{q}_\mu + 2\gamma_\mu \dot{q}_\mu + \omega_\mu^2 q_\mu = \bar{f}_{\mu,\mathrm{ext}} - \bar{f}_{\mu,\mathrm{nl}}, \tag{13}$$

where the coefficients are given by:

$$\omega_\mu^2 = \frac{D\lambda_\mu^2 + T_0\lambda_\mu}{\rho}, \tag{14}$$

$$\gamma_\mu = \frac{d_1 + d_3\lambda_\mu}{2\rho}. \tag{15}$$

One possible solution to the ODEs in Equation (13), following the FTM approach, is to use the Laplace transform and the impulse invariance discretisation to yield discrete transfer functions:

$$H_\mu(z) = \frac{b_{\mu,1}z}{z^2 + a_{\mu,1}z + a_{\mu,2}}, \tag{16}$$

$$a_{\mu,1} = -2e^{-\gamma_\mu T}\cos(\tilde{\omega}_\mu T), \tag{17}$$

$$a_{\mu,2} = e^{-2\gamma_\mu T}, \tag{18}$$

$$b_{\mu,1} = \frac{\rho^{-1}\sin(\tilde{\omega}_\mu T)}{\tilde{\omega}_\mu}e^{-\gamma_\mu T}, \tag{19}$$

with $\tilde{\omega}_\mu = \sqrt{\omega_\mu^2 - \gamma_\mu^2}$, which yields the following update scheme for all the modes $\mathbf{q} = [q_\mu]_{\mu=1}^M \in \mathbb{R}^M$:

$$\mathbf{q}^{n+1} = \mathbf{a_1} \odot \mathbf{q}^n + \mathbf{a_2} \odot \mathbf{q}^{n-1} + \mathbf{b_1}(\bar{\mathbf{f}}_{ext}^n - \bar{\mathbf{f}}_{nl}^n), \tag{20}$$

A similar second-order approach is to use Störmer-Verlet [20] directly on eq. (13). Discretising the ODE in time using centered finite differences we obtain the following update rule:

$$\mathbf{q}^{n+1} = \mathbf{g} \odot \mathbf{q}^n + \mathbf{p} \odot \mathbf{q}^{n-1} + \mathbf{r} \odot (\bar{\mathbf{f}}_{ext}^n - \bar{\mathbf{f}}_{nl}^n), \tag{21}$$

where $\mathbf{g}, \mathbf{p}, \mathbf{r}$ are vectors of coefficients obtained from the time discretisation for each μ-th mode:

$$r_\mu = \frac{2T^2}{2 + 2\gamma_\mu T} \tag{22}$$

$$g_\mu = r_\mu \left(\frac{2}{T^2} - \omega_\mu^2 \right) \tag{23}$$

$$p_\mu = r_\mu \left(-\frac{1}{T^2} + \frac{2\gamma_\mu}{2T} \right), \tag{24}$$

In both cases, $\bar{\mathbf{f}}_{ext}^n$ and $\bar{\mathbf{f}}_{nl}^n$ denote the modal coordinate vectors of external forces and non-linear terms, respectively, and T denotes the sampling period. While higher-order integration schemes could be employed for improved accuracy, we choose second-order methods due to their computational efficiency. Figure 1 compares the numerical errors of these second-order schemes against a reference solution computed with an eighth-order Runge-Kutta integrator (DOP853).
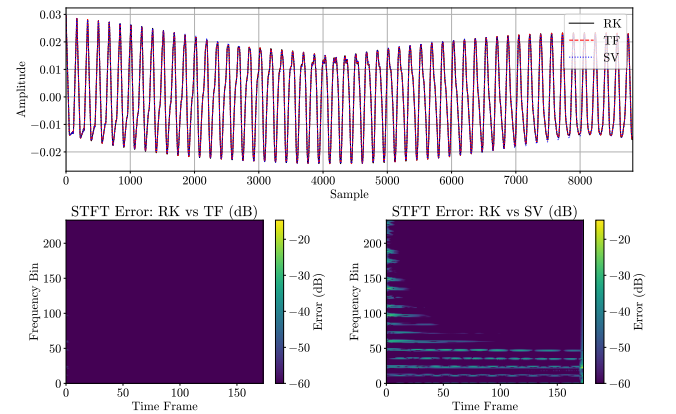

Figure 1: *Error comparison between numerical integration methods. **Top**: Simulation of the tension-modulated string from initial conditions using DOP853 (RK), the FTM-based method (TF), and the Störmer-Verlet method (SV). **Bottom**: Error computed as the magnitude STFT difference between each method and the reference DOP853 integrator. The simulation was performed for a tension-modulated string with 40 modes.*

4. METHOD

4.1. Implementation

We implemented the framework in Python using JAX [21] which enables efficient, GPU-accelerated simulations with automatic differentiation. The full source code for all experiments, benchmarks, and figures presented in this paper is available at: `https://github.com/rodrigodzf/jaxdiffmodal` as well as an accompanying website for documentation and usage of the framework.

The open source repository includes utilities for computing analytical modes and eigenvalues of fixed strings, membranes and plates. For plates, we provide a Python implementation of the approach described in [11] to compute non-linear coupling coefficients[2]. To support arbitrary boundary conditions, we use the `magpie-python` library[3] together with a reimplementation for processing numerical modes based on[4].

The framework includes differentiable implementations of the time integration schemes discussed in Section 3, along with tools for modal to physical domain transformations, transfer function computation, and visualisation. All components are modular and designed to support easy experimentation and extension.

4.2. Benchmarks

A key motivation for this work is to develop a high-performance simulation framework for non-linear models, particularly aimed at efficient large-scale dataset generation. To evaluate computational performance, we benchmark our Störmer-Verlet time integration approach (computationally equivalent to the discretised FTM method) for the von Kármán plate model implemented in JAX against three alternative implementations: a MATLAB version, an optimised C++ implementation using the Eigen library with BLAS support, and a JIT-compiled PyTorch implementation. This comparison is particularly challenging due to the tensor contraction (in Equation (10)) required at each time step. As shown in Figure 2, while our JAX-based approach is moderately slower than the MATLAB and C++ implementations for smaller mode counts ($\lesssim 50$), it demonstrates better efficiency as the number of modes increases. It also substantially outperforms the optimised PyTorch implementation running on the GPU. Notably, for the von Kármán plate model with approximately 100 modes, our implementation is the only one capable of achieving roughly twice real-time performance for a one-second simulation at a 44,100 Hz sampling rate.

All benchmarks were performed on a system with an AMD Ryzen 9 5900X CPU, an NVIDIA GeForce RTX 3090 GPU, and 64GB of RAM, using single-precision floating point.

We also developed a custom CUDA kernel implementation, but it is not included in the benchmark results as it proved unexpectedly slower than the other approaches.

4.3. Experiments

We present three experiment groups designed to evaluate the differentiability and effectiveness of our framework for inverse modelling. These experiments assess the ability of our method to identify and optimise physical parameters for strings and plates using

---

[2]`https://vkgong.ensta-paris.fr`
[3]`https://github.com/Nemus-Project/magpie-python`
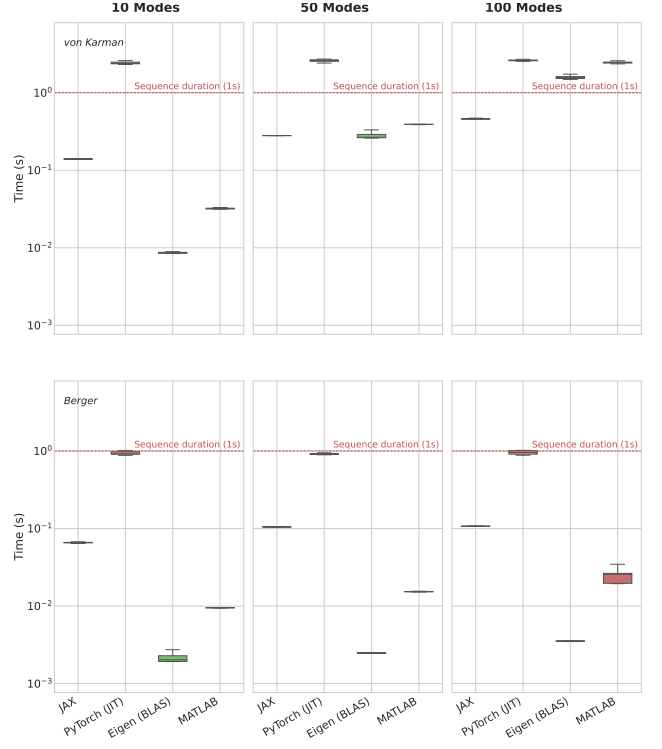[4]`https://github.com/Nemus-Project/VKPlate`

Figure 2: Benchmark comparison of different implementations. **Top:** von Kármán plate model, **Bottom:** tension modulated plate model (Berger). Performance is measured for simulations of 10, 50, and 100 modes, each running for 1 second at 44,100 Hz. A dotted line is shown to indicate real-time performance, above the line is slower than real-time. Results are displayed as box plots obtained from 50 repeated runs per configuration.

both synthetic simulations and real-world measurements (mainly for linear cases).

4.3.1. Loss functions and setup

For inverse modelling of physical parameters, we employ a composite loss function based on spectral differences between predicted and target spectral magnitudes. This loss function, common across all experiments, combines three complementary spectral metrics:

$$\mathcal{L}_{\log} = || \log(Y + \epsilon) - \log(\hat{Y} + \epsilon)||_1, \tag{25}$$

$$\mathcal{L}_{\text{sc}} = \frac{||Y - \hat{Y}||_F}{||Y||_F}, \tag{26}$$

$$\mathcal{L}_{\text{sot}} = \frac{1}{N} \sum_{i=0}^{N} \mathcal{W}_1 \left(Y_i, \hat{Y}_i \right), \tag{27}$$

$$\mathcal{L}_{\text{total}} = \alpha \mathcal{L}_{\log} + \beta \mathcal{L}_{\text{sc}} + \eta \mathcal{L}_{\text{sot}}, \tag{28}$$

where x and $\hat{x}$ represent the time-domain target and predicted signals, respectively, with $Y = |\text{STFT}(x)|$ and $\hat{Y} = |\text{STFT}(\hat{x})|$ denoting their corresponding STFT magnitude spectrograms. The terms Y_i and $\hat{Y}_i$ refer to individual (normalised) time frames of the spectrograms used in the spectral optimal transport loss $\mathcal{L}_{\text{sot}}$ [22]

< **137** >

via the Wasserstein-1 distance $\mathcal{W}_1$. The constant ϵ ensures numerical stability in the logarithmic term. The weights α, β, and η are calibrated for each experiment to balance the contribution of each loss component. For linear cases, we also compute a slight variation of these losses where the STFT is replaced by the sampled magnitude frequency response, which provides a more direct measure of the systems transfer function (available in this specific case). Although $\mathcal{L}_{\text{sot}}$ helps avoid poor local minima, the overall spectral loss landscape remains non-convex, as illustrated in Figure 3, even for a single parameter.

We perform optimisation using the Adam optimiser with a one-cycle cosine learning rate schedule. For linear cases, we run 15,000 optimisation steps, while for non-linear cases, we conduct 1,000 steps with 100 parallel random initializations to improve convergence.

Some of the parameters are optimised in a *density-normalised* form: $\hat{D} = D/\rho$ and $\hat{T}_0 = T_0/\rho$. For damping parameters, we also directly optimise the γ_μ coefficients in some of the experiments to enhance model flexibility when fitting to real-world data.
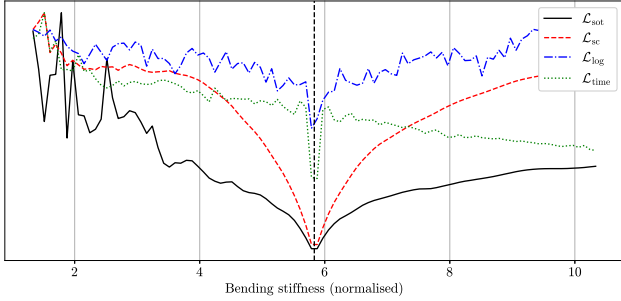

Figure 3: Loss landscapes with respect to the normalized bending stiffness $\hat{D}$. Three losses are considered: log-magnitude ($\mathcal{L}_{\text{log}}$), spectral convergence ($\mathcal{L}_{\text{sc}}$), and spectral optimal transport ($\mathcal{L}_{\text{sot}}$). The time-domain MSE ($\mathcal{L}_{\text{time}}$) is included for comparison. The vertical dotted line indicates the optimal value of $\hat{D}$ for the target response. The losses are scaled for better visualization.

By optimising the parameters $\hat{D}$ and $\hat{T}_0$, we implicitly optimise the modal frequencies while preserving the dispersion relation and eigenvalues determined by the geometry and boundary conditions (solutions to eq. (2)). This approach differs fundamentally from methods that optimise modes independently of physical constraints [14].

4.3.2. String

We first consider the linear string model, focusing on parameter fitting for both synthetic and real string responses. Two optimisation approaches are used for this task.

In the first approach, and in particular for the real-world scenarios, we extract the spectral envelope from the target response using high-order Linear Predictive Coding (LPC), though alternative methods could be employed. We then sample the LPC coefficients in frequency using the Bark scale. On the model side, we compute the transfer function of the linear system using eq. (16), based on a set of parameters to be optimised. The predicted response is sampled in the same way as the LPC envelope. We then compute the loss using the functions defined in eq. (25), replacing the STFT magnitude with the Bark-sampled magnitude response.
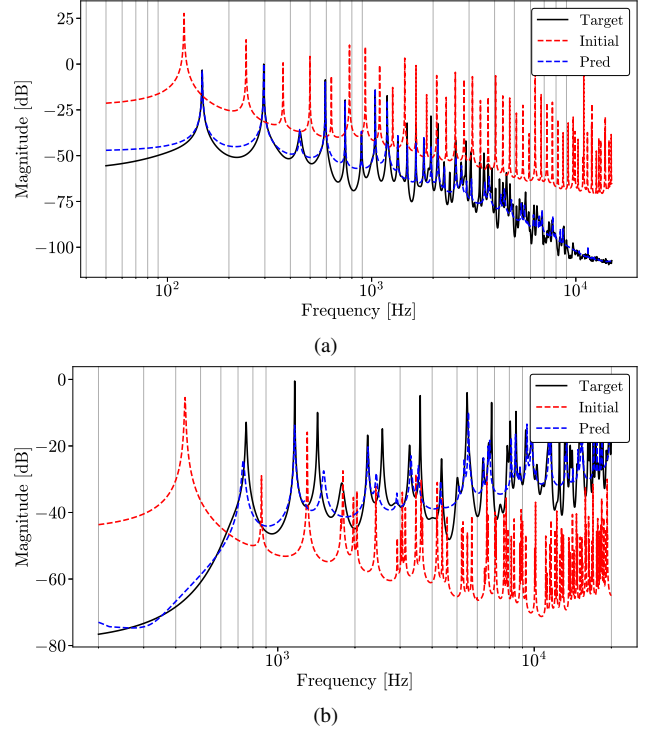

Figure 4: Inverse modelling experiments. (a) Semi-log plot of the fit to a real plucked string. (b) Semi-log plot of the fit to a real struck thick plate.

This provides a simple and efficient route for inverse modelling in the linear case, avoiding time-domain simulation. Figure 4a shows the semi-log plot of the fit to a real plucked string.

In the second approach, we simulate the string response in time by solving the linear ODE directly. This is done through recursive multiplication of the discrete complex eigenvalues derived from the dispersion relation $s_{\mu\pm} = \gamma_\mu \pm i\tilde{\omega}_\mu$. This can be efficiently implemented using the prefix-sum algorithm [23] allowing fast evaluation across modes. It is also possible to use the solving schemes (eqs. (20) and (21)), however for the linear case this is not necessary.

We fit the parameters of a single recording of a real plucked string from the IDMT-SMT-Guitar dataset [24]. For this experiment, we assume the string is fixed at both ends and that nonlinear effects (e.g. pitch glide) are minimal.

While optimisation converges more quickly in the synthetic case, the real-world scenario presents greater challenges. This increased complexity arises from the need to account for not only the PDE parameters but also the external force excitation, the acoustic transfer function, measurement noise, and other effects.

We extend the model by optimising an additional parameter $b_{\mu,2}$ (in Equation (16)) for each mode, to address the unknown initial conditions. Furthermore, instead of projecting the solution back to the physical domain using analytical mode shapes, we simply optimise a set of abstract weights for each mode. These weights account for effects such as the acoustic transfer and microphone placement, even if they are not physically interpretable.
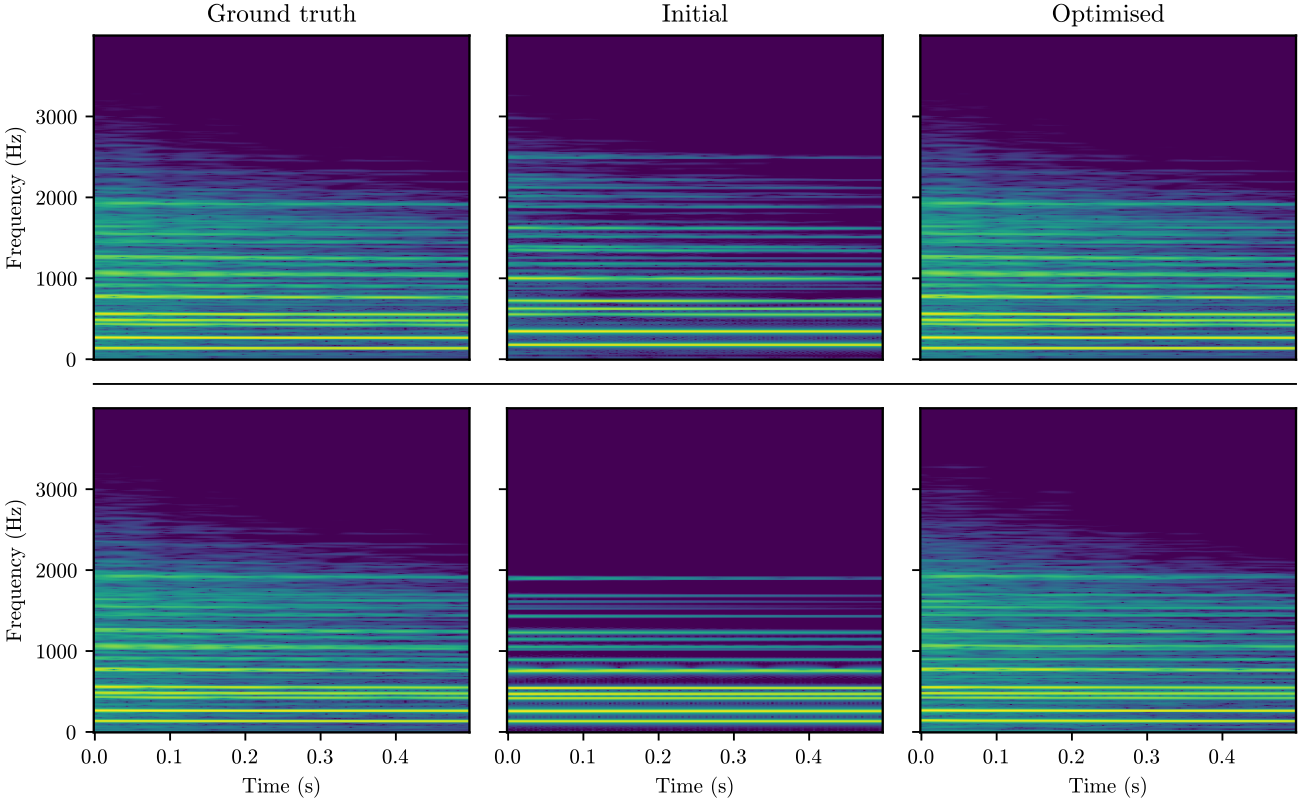
< **138** >

Figure 5: Spectrograms of the target and optimised forced responses from simulations of the von Kármán plate model. **Top**: Matching the response at a single point on the plate, by optimising only the normalised bending stiffness ($\hat{D}_{\text{target}} = 5.8328$, $\hat{D}_{\text{initial}} = 10$, $\hat{D}_{\text{optimised}} = 5.8329$). **Bottom:** Matching the response by optimising only the coupling coefficients H, initialised with random values from a normal distribution.

4.3.3. Plate

We apply a similar procedure to the linear plate model simulations, examining both synthetic and real scenarios. The real-world case uses a single hammer strike recording from a thick rigid plate taken from the RealImpact dataset [25].

In the synthetic non-linear case, we simulate the displacement of the plate at a fixed point in the domain. We then optimise the parameters by propagating gradients through the time integrator to match this response (i.e., using BPTT). This type of optimisation is more sensitive to parameter initialisation of parameters and is prone to getting trapped in local minima. Even when optimising a single parameter, such as $\hat{D}$, the loss landscape exhibits many local minima (Figure 3). This behaviour is also consistent with previously observed challenges in the optimisation of frequencies [26]. In addition, care must be taken to ensure that all parameters remain within numerically stable ranges. In particular, the damping coefficients γ_μ must be constrained to ensure all poles lie in the left half of the Laplace domain to avoid numerical instability. To address the challenge of optimising the bending stiffness and avoiding poor local minima, we adjust the weighting of $\mathcal{L}_{\text{total}}$ to emphasise the spectral optimal transport and spectral convergence losses. Since the optimisation using automatic differentiation is fast (for approximately 0.1 seconds at 44,100 Hz), we can efficiently explore multiple random initialisations in parallel [27].

We also experiment with the optimisation of the coupling coefficients C and H (in Equation (11)) of the von Kármán plate model, which are otherwise difficult to compute analytically or numerically. In the simply supported case, only the H tensor needs to be optimised, since it satisfies the identity $H_{i,j}^k = C_{i,k}^j$ [10]. We initialise the tensor with random values drawn from a normal distribution and optimise it to match the simulated plate displacement at a fixed point in space. This optimisation is carried out over a sequence of 17,640 time steps (0.4 seconds at 44,100 Hz).

Figure 5 shows the spectrograms of the target, initial, and optimised forced responses from a synthetic von Kármán plate simulation. The results show that the optimisation successfully converges to the correct behaviour. Although the fit can be performed on a short segment (0.4 seconds), the optimised parameters remain valid and consistent for longer simulations.

5. CONCLUSION

In this paper, we have presented a fast, differentiable modal simulation framework for strings, membranes, and plates. By leveraging GPU acceleration through the JAX library, our method significantly outperforms traditional CPU-based implementations, enabling efficient simulation, dataset generation, and inverse modelling, even for non-linear models such as the von Kármán plate.

<   **139**   >

The benchmarks demonstrate that our implementation scales efficiently with an increasing number of modes, providing substantial performance gains compared to existing MATLAB, optimised C++, and GPU-based PyTorch (JIT) implementations. Furthermore, it supports effective gradient-based optimisation of physical parameters from both synthetic simulations and real-world measurements. We successfully recovered parameters such as tension, bending stiffness, damping, and geometry, including the coupling tensors of the non-linear von Kármán plate model.

However, directly optimising physical parameters rather than fitting more abstract representations such as damped exponentials or poles introduces significant challenges. The physical parameterisation adds complexity and non-linear constraints, making optimisation sensitive to initialisation and prone to convergence to poor local minima. Despite these challenges, the physical approach offers considerable benefits. First, the results are directly interpretable since the parameters correspond to physically meaningful quantities. Second, the representation is highly compact, capturing behaviour that would otherwise require many free parameters with just a few well-defined physical ones.

Despite the advantages, several limitations remain. The ill-posed nature of inverse modelling in this case needs careful initialisation and robust optimisation strategies are essential. While our experiments demonstrate successful parameter recovery, a quantitative benchmark against alternative approaches such as gradient-free optimisers or fitting to non-physical modal representations would be required to fully assess the efficiency and robustness of our physically constrained gradient-based method. Although using multiple random initialisations helps mitigate these issues, a more structured approach to parameter initialisation and optimisation scheduling is needed to improve robustness and convergence. For example, a staged optimisation approach similar to that of [13], in which damping and frequencies are optimised at different stages, may improve convergence stability.

Looking ahead, we plan to explore several improvements and extensions. First, we aim to generalise the solver to support three-dimensional structures using a differentiable finite element method (FEM) solver. Second, we will reimplement a modal SAV method and the energy-conserving integration scheme in Python, both of which have been previously implemented in MATLAB. Third, the current coupling tensor computations could benefit from sparsity-aware techniques. Exploiting the inherent structure of these tensors through sparse matrix representations could lead to significant speedups for high mode count simulations.

In summary, our work bridges classical modal techniques with modern automatic differentiation methods, offering a highly efficient, differentiable, and physically interpretable framework. It enables new applications in audio synthesis, inverse modelling, and physically informed machine learning, while opening avenues for future research into differentiable physics-based modelling.

6. ACKNOWLEDGMENTS

7. REFERENCES

[1] Lauri Savioja, "Real-time 3D finite-difference time-domain simulation of low- and mid-frequency room acoustics," in *Proc. Int. Conf. on Digital Audio Effects (DAFx 10)*, Graz, Austria, 2010.

[2] Lutz Trautmann and Rudolf Rabenstein, "Functional Transformation Method," in *Digital Sound Synthesis by Physical Modeling Using the Functional Transformation Method*, Lutz Trautmann and Rudolf Rabenstein, Eds., pp. 95–187. Springer US, Boston, MA, 2003.

[3] Stefan Bilbao, *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*, Wiley, first edition, Oct. 2009.

[4] Stefan Bilbao, Craig Webb, Zehao Wang, and Michele Ducceschi, "Real-time gong synthesis," in *Proc. Int. Conf. on Digital Audio Effects (DAFx 23)*, Copenhagen, Denmark, Sept. 2023, pp. 1–8.

[5] Zehao Wang, Stefan Bilbao, Tom Erbe, and Miller Puckette, "Real-time implementation of the Kirchhoff plate equation using finite-difference time-domain methods on CPU," in *Proc. of the 20th Sound and Music Computing Conference*, Stockholm, Sweden, June 2023, pp. 354–361.

[6] Stefan Bilbao, "Modal-type synthesis techniques for nonlinear strings with an energy conservation property," in *Proc. Int. Conf. on Digital Audio Effects (DAFx 04)*, Naples, Italy, 2004.

[7] Maximilian Schäfer and Rudolf Rabenstein, "A continuous frequency domain description of adjustable boundary conditions for multidimensional transfer function models," in *Proc. Int. Conf. on Digital Audio Effects (DAFx 17)*, Edinburgh, UK, 2017, pp. 315–322.

[8] Federico Avanzini, Riccardo Marogna, and Balázs Bank, "Efficient synthesis of tension modulation in strings and membranes based on energy estimation," *The Journal of the Acoustical Society of America*, vol. 131, no. 1, pp. 897–906, Jan. 2012.

[9] Federico Avanzini and Riccardo Marogna, "A Modular Physically Based Approach to the Sound Synthesis of Membrane Percussion Instruments," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 891–902, May 2010.

[10] Michele Ducceschi and Cyril Touzé, "Modal approach for nonlinear vibrations of damped impacted plates: Application to sound synthesis of gongs and cymbals," *Journal of Sound and Vibration*, vol. 344, pp. 313–331, May 2015.

[11] Michele Ducceschi and Cyril Touzé, "Simulations of non-linear plate dynamics: An accurate and efficient modal algorithm," in *Proc. Int. Conf. on Digital Audio Effects (DAFx 15)*, Trondheim, Norway, 2015.

[12] Samuel Clarke, Negin Heravi, Mark Rau, Ruohan Gao, Jiajun Wu, Doug James, and Jeannette Bohg, "Diffimpact: Differentiable rendering and identification of impact sounds," in *Conference on Robot Learning*. 2022, pp. 662–673, PMLR.

[13] Xutong Jin, Chenxi Xu, Ruohan Gao, Jiajun Wu, Guoping Wang, and Sheng Li, "DiffSound: Differentiable modal sound rendering and inverse rendering for diverse inference

tasks," in *ACM SIGGRAPH 2024 Conference Papers*, New York, NY, USA, 2024, Siggraph '24, Association for Computing Machinery.

[14] Rodrigo Diaz, Ben Hayes, Charalampos Saitis, György Fazekas, and Mark Sandler, "Rigid-Body Sound Synthesis with Differentiable Modal Resonators," in *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 23)*, June 2023, pp. 1–5.

[15] Rodrigo Diaz, Carlos De La Vega Martin, and Mark Sandler, "Towards efficient modelling of string dynamics: A comparison of state space and koopman based deep learning methods," in *Proc. Int. Conf. on Digital Audio Effects (DAFx 24)*, Guildford, UK, 2024.

[16] Jin Woo Lee, Jaehyun Park, Min Jun Choi, and Kyogu Lee, "Differentiable Modal Synthesis for Physical Modeling of Planar String Sound and Motion Simulation," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2024, vol. 37, pp. 1058–1081.

[17] Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew M. Stuart, and Anima Anandkumar, "Fourier Neural Operator for Parametric Partial Differential Equations.," in *Proc. of the 9th International Conference on Learning Representations*, Austria, 2021.

[18] Julian Parker, Sebastian Schlecht, Maximilian Schäfer, and Rudolf Rabenstein, "Physical Modeling Using Recurrent Neural Networks with Fast Convolutional Layers," in *Proc. Int. Conf. on Digital Audio Effects (DAFx 22)*, 2022, pp. 138–145.

[19] Lutz Trautmann and Rudolf Rabenstein, "Sound Synthesis with Tension Modulated Nonlinearities Based on Functional Transformations," in *Proc. of the Conference on Acoustics and Music: Theory and Applications*, Montego Bay, Jamaica, 2000, pp. 444–449, World Scientific and Engineering Academy and Society.

[20] Ernst Hairer, Christian Lubich, and Gerhard Wanner, "Geometric numerical integration illustrated by the Störmer–Verlet method," *Acta Numerica*, vol. 12, pp. 399–450, May 2003.

[21] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang, "JAX: Composable transformations of Python+NumPy programs," 2018, Available at http://github.com/jax-ml/jax.

[22] Bernardo Torres, Geoffroy Peeters, and Gaël Richard, "Unsupervised Harmonic Parameter Estimation Using Differentiable DSP and Spectral Optimal Transport," in *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 24)*, Apr. 2024, pp. 1176–1180.

[23] Guy E Blelloch, "Prefix sums and their applications," Tech. Rep., School of Computer Science, Carnegie Mellon University, 1990.

[24] Christian Kehling, Andreas Männchen, and Arndt Eppler, "IDMT-SMT-Guitar Dataset," Jan. 2023, Available at https://zenodo.org/records/7544110.

[25] Samuel Clarke, Ruohan Gao, Mason Wang, Mark Rau, Julia Xu, Jui-Hsien Wang, Doug L. James, and Jiajun Wu, "RealImpact: A dataset of impact sound fields for real objects," in *Proc. of the Conference on Computer Vision and Pattern Recognition*, Los Alamitos, CA, USA, 2023, pp. 1516–1525, IEEE Computer Society.

[26] 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*, Dec. 2020.

[27] Francesco Della Santa, "Automatic Differentiation-Based Multi-Start for Gradient-Based Optimization Methods," *Mathematics*, vol. 12, no. 8, pp. 1201, Jan. 2024.

< **141** >