Learning to Predict Structural Vibrations

Jan van Delden[1,*], Julius Schultz[2], Christopher Blech[2], Sabine C. Langer[2], and Timo Lüddecke[1]

[1]Institute of Computer Science, University of Göttingen
[2]Institute for Acoustics and Dynamics, Technische Universität Braunschweig
[*]Correspondence: jan.vandelden@uni-goettingen.de

Abstract

In mechanical structures like airplanes, cars and houses, noise is generated and transmitted through vibrations. To take measures to reduce this noise, vibrations need to be simulated with expensive numerical computations. Deep learning surrogate models present a promising alternative to classical numerical simulations as they can be evaluated magnitudes faster, while trading-off accuracy. To quantify such trade-offs systematically and foster the development of methods, we present a benchmark on the task of predicting the vibration of harmonically excited plates. The benchmark features a total of 12,000 plate geometries with varying forms of beadings, material, boundary conditions, load position and sizes with associated numerical solutions. To address the benchmark task, we propose a new network architecture, named Frequency-Query Operator, which predicts vibration patterns of plate geometries given a specific excitation frequency. Applying principles from operator learning and implicit models for shape encoding, our approach effectively addresses the prediction of highly variable frequency response functions occurring in dynamic systems. To quantify the prediction quality, we introduce a set of evaluation metrics and evaluate the method on our vibrating-plates benchmark. Our method outperforms DeepONets, Fourier Neural Operators and more traditional neural network architectures and can be used for design optimization. Code, dataset and visualizations: `https://github.com/ecker-lab/Learning_Vibrating_Plates`

1 Introduction

Humans are exposed to noise in everyday life, which is unpleasant and unhealthy in the long term [1]. Therefore, designers and engineers work on reducing noise that occurs, for example, in cars, airplanes, and houses. In this work, we specifically consider vibrations in mechanical structures as a source of sound. Vibrating structures radiate sound into the surrounding air. For example in a car, the engine causes the chassis to vibrate, which then radiates sound into the interior of the car. By reducing the vibration energy of the chassis, the noise can be reduced.

Vibrations of mechanical structures depend on the frequency of the excitation force (e.g. by the engine). A special case occurs when the excitation frequency matches an eigenfrequency of a given structure. In this case, the external force adds energy in phase with the structure's natural vibration and amplifies the motion with each cycle. This continues until the energy added equals the energy lost due to damping, resulting in large vibration amplitudes. This effect is called resonance and leads to characteristic resonance peaks in the dynamic response of the system. At resonance frequencies, due to the higher vibration amplitudes, more noise is emitted. A second distinctive feature of structural vibrations is the vibration pattern, i.e. the spatial field of vibration velocity amplitudes. With increasing frequency, these vibration patterns become more complex and exhibit more local maxima and minima (Figure 1, left) [2].

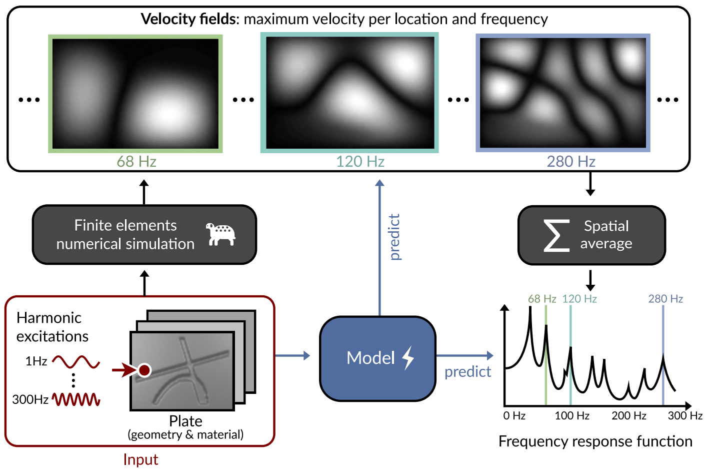
Velocity fields: maximum velocity per location and frequency
68 Hz
120 Hz
280 Hz
Finite elements numerical simulation
predict
Spatial average
Harmonic excitations
1Hz
300Hz
Plate
(geometry & material)
Input
Model
predict
68 Hz 120 Hz 280 Hz
0 Hz 100 Hz 200 Hz 300 Hz
Frequency response function

Figure 1: Left: We introduce the Vibrating Plates dataset of 12,000 samples for predicting vibration patterns based on plate geometries. A harmonic force excites the plates, causing them to vibrate. The vibration patterns of the plates are obtained through numerical simulation. Diverse architectures are evaluated on the dataset. Right: Beadings are indentations and used in many vibrating technical systems. Here, on an oil filter, a washing machine and a disk drive. They increase the structural stiffness and alter the vibration.

To reduce noise, the vibration patterns of a mechanical structure can be influenced through modifications to its design. One method is the placement of damping elements, that absorb vibrational energy and thereby reduce sound emission, but this adds weight and requires space. Another approach is introducing beadings, which are indentations in plate-like structures (Figure 1, right). Beadings increase the local stiffness of a structure, resulting in a shift in the structure's eigenfrequencies and subsequent resonance peaks. When they are well-placed, beadings can reduce the vibration energy for a range of excitation frequencies by shifting the resonance peaks out of the range. Reducing the vibration energy for a specific range of frequencies is a goal in many applications, e.g. in automotive design, where a motor excites vibrations in a range of frequencies [3].

In this work, we focus on a crucial prerequisite for targeted modifications to a design: Computing its vibrational behavior. The finite element method (FEM) is an established approach for numerically solving partial differential equations. The geometry of a design is discretized into small elements and the solution of the PDE is approximated by simple functions, e.g. polynomial functions, defined on these elements. [4, 5]. This method enables the numerical simulation of vibration patterns, but is computationally expensive. With increasing frequency and decreasing wavelength, finer meshes are required to accurately resolve the vibrations. This leads to a high increase in computational load and limits the number of designs and value of the frequencies that can be evaluated. Deep learning surrogate models could accelerate the evaluation of design candidates by several magnitudes.

Related work on predicting the solution of partial differential equations with deep learning has mostly focused on time-domain problems [e.g. 6, 7, 8]. In contrast, for our problem the change over time is not of interest. Instead, we predict steady-state vibration patterns in the frequency domain. Steady-state refers to the fact that the system vibrates harmonically and the amplitude and frequency remain constant over time since the system is in a dynamic equilibrium. Despite being practically relevant in acoustics and structural dynamics in general this problem is so far under-explored by machine learning research.

Contributions. To explore the potential of vibration prediction with deep learning methods, we (1) introduce a benchmark and define evaluation metrics on it, (2) evaluate a range of machine learning methods on the benchmark and (3) introduce our own method.

Our novel benchmark dataset consists of 12,000 instances of an exemplary structural mechanical system, a plate excited by a harmonic force, and their numerically computed vibrations given a range of excitation frequencies. Given a plate instance, the task is to predict the vibration patterns and frequency response. We vary material properties and the boundary conditions of the plate as well as the geometry by adding beadings. Plates with beadings are abundant in technical systems

(Figure 1, right). Plates are also often a component of more complex mechanical systems and their vibrational behavior on their own is similar to more complex systems [9, 10], making them a well-posed and scalable initial benchmark problem for deep learning methods.

To address the benchmark task, we propose a novel network architecture named Frequency-Query Operator (FQO). This model is trained to predict the resulting vibration pattern from plate geometries together with an excitation frequency query. This approach is inspired by work on operator learning for predicting the solution to partial differential equations [11] and implicit models for shape representation [e.g. 12, 13, 14], both techniques enable evaluating any point in the domain instead of a fixed grid. In our case, this enables predictions for any excitation frequency, including those not seen during training. On our vibrating-plates benchmark, the proposed FQO can accurately predict the highly variable resonances occurring in vibration patterns and outperforms DeepONet [11], Fourier Neural Operators [15] and other baselines.

2 Dataset and Benchmark Construction

2.1 Vibrating Plates Dataset

We introduce a dataset consisting of instances of aluminum plate geometries and their vibration patterns. The plates are simply supported, i.e. the edges cannot move up and down. Depending on the dataset setting, the rotational stiffness at the boundary is varied, which corresponds to free rotation or clamped edges. The plate is excited by a harmonic point force at varying positions with the excitation frequency varied between 1 and 300 Hz. While the specific setting in other mechanical engineering design tasks may differ, this setup functions as an exemplary engineering design problem. Analogous problems are the design of an air-conditioning enclosure [16], a washing machine [17] or parts of a car chassis [3]. Compared to these problems, our plate setup has two differences that allow for a comparatively easy experimental real world validation of the computed vibration patterns and do not change typical vibrational characteristics: First, exciting the plate with a point force is a common experimental setup, where a plate is excited via a shaker. Second, the condition of no rotational stiffness at the edges in comparison to clamped edges does not introduce additional uncertainty and parameters into the measurement and mirrors e.g. a bonnet of a car that rests on the chassis. Other typical types of fixation include screws or welding. In the following, we describe the specific quantity of interest of the vibration patterns, how the vibration patterns of the plate are obtained via numerical simulation and how the plate geometry and parameters are varied.

Vibration patterns and frequency response function. Our benchmark is designed to address a vibroacoustic engineering design problem. Therefore, the goal is to predict a quantity that best reflects the noise emitted by a mechanical structure. For a plate, a natural choice is the maximum velocity field $v_z(x, y|f)$ for a specific frequency f. Here, $v_z(x, y|f)$ represents the component of the velocity field orthogonal to the plate surface (in the following $\mathbf{V}(f)$ denotes the velocity field on the discrete grid). This component closely relates to how much sound is radiated, but specific details about where the velocity on the plate is highest are superfluous. Therefore, we use the mean of the squared velocity as a more compact representation and express it in a frequency response function $\mathcal{F}$, which is a function of the excitation frequency:

$$\mathcal{F}(f) = 10 \log_{10} \left(\frac{r}{A} \int_A v_z(x, y|f)^2 \, dA \right) \tag{1}$$

The square velocity is proportional to the kinetic energy and is therefore closely related to how strongly the vibration couples into a surrounding fluid and can then be perceived as airborne sound. In the above expression, A is the plate area over which the velocity is averaged. The result is scaled by a reference value r and converted to a decibel scale.

Numerical simulation. Historically, plate structures have been the subject of intense research regarding their vibrational behavior [e.g. 18, 19]. A common approach in plate modeling is to reduce the model to a two-dimensional problem with the goal to accurately describe the vibrational behavior while being computationally efficient [20]. To model the vibrational behavior of plates in this work, we use a shell formulation based on Mindlin's plate theory [18]. This theory is applicable for moderately thin plates and represents the plate using a mid-plane with constant thickness. Mindlins

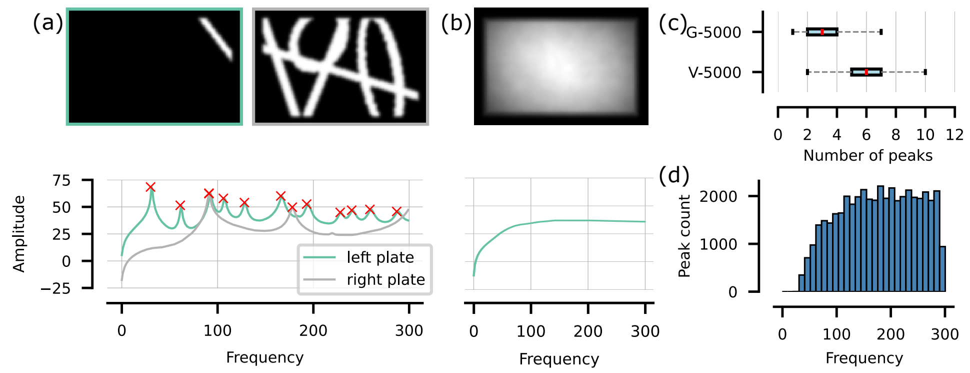

Figure 3: Dataset analysis. (a) shows two discretized plate geometries with their corresponding frequency response, the red crosses mark the detected peaks. (b) shows the mean plate design and frequency response. (c) shows number of peaks in different dataset settings. (d) shows the distribution of the peaks over the frequencies.

plate theory is a standard choice in many engineering applications and has been experimentally validated [21, 22].

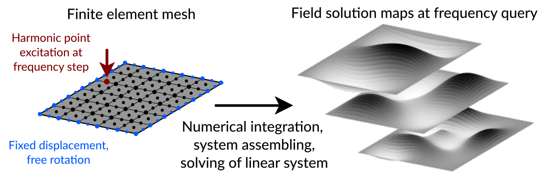
Finite element mesh
Field solution maps at frequency query
Harmonic point excitation at frequency step
Fixed displacement, free rotation
Numerical integration, system assembling, solving of linear system

Figure 2: Process of the finite element solution in frequency domain in order to compute the velocity field at each frequency query.

We apply the finite element method to solve the shell formulation and simulate the vibrational behavior of the plate [4] (Figure 2). This involves partitioning the plate geometry into discrete elements and approximating the solution on these elements by simple ansatzfunctions. By choosing a sufficiently large number of elements, the solution converges to the exact solution of the model [23]. We discretize the plate with a regular grid and use triangular elements in the domain to allow a flexible representation of beadings. The discretization is sufficient to resolve wave lengths in the plate structure, but limits the detail that can be represented with the beading patterns. After discretizing the plate, the PDE is integrated over the elements and a linear system of equations is derived. This linear system describes the dynamics of the discretized structure and is solved with a direct solver. We perform the computations with a specialized FEM software for acoustics [24]. Further details on the setup and mechanical model are given in Appendix A.1.

Dataset variations. The plate instances are varied in two settings: For the V-5000 setting, we generate random beading patterns consisting of 1 - 3 lines and 0 - 2 ellipses. Also, the width of the beading-elements is randomly varied. The size of the plates as well as material, boundary and loading parameters are fixed. For the G-5000 setting, we apply the same beading pattern variation and additionally vary the plate geometry (length, width and thickness) as well as the damping loss factor, rotational stiffness at the boundary and forcing position. For each setting, 5000 instances for training and validation are generated. 1000 further instances are generated as a test set and are not used during training or to select a model. Further details are given in Appendix A.2.

Dataset analysis. The mean plate design shows a close to uniform distribution, with a margin at the plate's edge (see Figure 3b). With a greater proportion of beaded area in a given plate, the number of peaks tends to decrease (see Figure 3a). This is due to additional beadings stiffening the plates, and it represents an interesting trait specific to our problem. The density of peaks is related to the frequency. As the frequency increases, so does the peak density. Starting from around 120 Hz the peak density plateaus (see Figure 3d). The average number of peaks in the G-5000 setting is smaller than in the V-5000 setting. This is influenced by the on average smaller plates being stiffer and therefore having less peaks in the frequency range (see Figure 3c).

2.2 Evaluation

Before computing our metrics, we perform the following preprocessing steps to address numerical issues as well as facilitate an easier interpretation of the evaluation metrics. We normalize the frequency response and the velocity fields. To do this, we first take the log of the velocity fields, to align it with the dB-scale of the frequency response. Then, we subtract the mean per frequency over all samples (depicted in Figure 3b for frequency response) and then divide by the overall standard deviation across all frequencies and samples. Small changes in the beading pattern can cause frequency shifts, potentially pushing peaks out of the considered frequency band. To reduce the effect of such edge cases, we predict frequency responses between 1 and 300 Hz but evaluate on the frequency band between 1 and 250 Hz.

We propose three complementary metrics to measure the quality of the frequency response predictions.

Mean squared error. The *mean squared error (MSE)* is a well-known regression error measure: For the global deviation we compare the predicted $\hat{\mathcal{F}}(f)$ and numerically computed frequency response $\mathcal{F}(f)$ by the MSE error $\mathcal{E}_{\text{MSE}} = \sum_i (\hat{\mathcal{F}}(f_i) - \mathcal{F}(f_i))^2$.

Earth mover distance. The *earth mover distance* [25, 26] expresses the work needed to transmute a distribution P into another distribution Q. As a first step, the optimal flow $\hat{\gamma}$ is identified. Based on $\hat{\gamma}$ the earth mover distance is expressed as follows:

$$\mathcal{E}_{\text{EMD}}(P,Q) = \frac{\sum_{i,j} \hat{\gamma}_{ij} \cdot d_{ij}}{\sum_{i,j} \hat{\gamma}_{ij}} \qquad \text{with } \hat{\gamma} = \min_{\gamma} \sum_{i,j} \gamma_{ij} \cdot d_{ij}$$

where d_{ij} is the distance between bins i and j in P and Q. Correspondingly, γ_{ij} is the flow between bins i and j. We calculate the $\mathcal{E}_{\text{EMD}}$ based on the original amplitudes in m/s that have not been transformed to the log-scale (dB) and normalize these amplitudes with the sum over all frequencies. As a consequence and unlike the MSE, $\mathcal{E}_{\text{EMD}}$ is invariant to the mean amplitude and only considers the shape of the frequency response. In this form, our metric is equivalent to the W_1 Wasserstein metric [27, 28].

Peak frequency error. To specifically address the prediction of resonance peaks, which are particularly relevant for noise emission, we introduce a third metric called *peak frequency error*. The metric answers two questions: (1) Does the predicted frequency response contain the same number of resonance peaks as the true response? (2) How far are corresponding ground truth and prediction peaks shifted against each other? To this end, we set up an algorithm that starts by detecting a set of peaks K in the ground truth and a set of peaks $\hat{K}$ in the prediction using the `find_peaks` function in scipy [29] (examples in Appendix B). Then, we match these peaks pairwise using the Hungarian algorithm [30] based on the distance between the frequencies of the peaks $\mathcal{E}_{\text{F}}$. This allows us to determine the ratio between predicted and actual peaks $\frac{|\hat{K}|}{|K|}$ and $\frac{|K|}{|\hat{K}|}$. To equally penalize predicting too many and too few peaks we consider the minimum of both ratios: $\mathcal{E}_{\text{PEAKS}} = 1 - \min\{\frac{|\hat{K}|}{|K|}, \frac{|K|}{|\hat{K}|}\}$.

3 Predicting Vibrations with Neural Networks

We propose a method to predict the frequency response, $\mathcal{F}_{\mathbf{g},\mathbf{m}}(f)$, for plates characterized by their geometry $\mathbf{g}$ (influenced by beading patterns) and scalar parameters $\mathbf{m}$ (height, width, thickness, damping loss factor, rotational stiffness at boundary, loading position). This process involves two steps: (1) First, the input $\mathbf{g}$ and $\mathbf{m}$ are encoded by an encoder Φ. Because $\mathbf{g}$ is defined on a regular grid, standard image processing architectures are suitable. (2) Frequency response predictions are generated for specific excitation frequencies f by a decoder Ψ (Figure 4). The computation can then be expressed as:

$$\Psi(\Phi(\mathbf{g}, \mathbf{m}), f) = \hat{\mathcal{F}}_{\mathbf{g},\mathbf{m}}(f) \tag{2}$$

This problem formulation, training a neural network to predict a function and evaluating this function, given some input values, is a common paradigm in operator learning [11]. It allows for the evaluation of any frequency query f, even if it has not been part of the training data. In contrast, predicting frequencies on a fixed grid only allows for the evaluation of those frequencies. This formulation shares similarities with implicit models, for instance by [13] in the context of 3d shape prediction. Based on this, we investigate the following central aspects of our architecture:

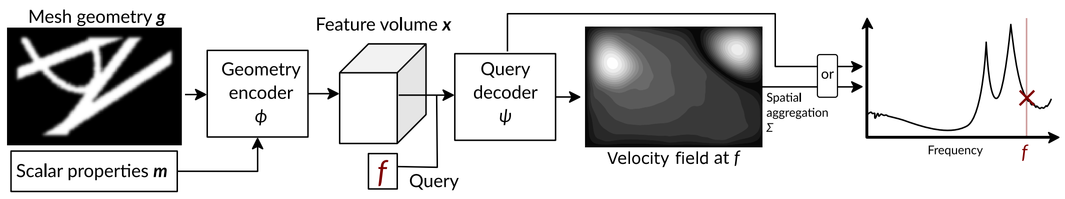

Figure 4: Frequency-Query Operator method. The geometry encoder takes the mesh geometry and the scalar properties as input. The resulting feature volume along with a frequency query is passed to the query decoder, that either predicts a velocity field or directly a frequency response. The velocity field is aggregated to arrive at the frequency response at the query frequency f.

Q1 - Frequency-query approach: Vibrations are dominated by resonance peaks at specific frequencies. The resonance frequencies vary strongly across instances. An implicit or operator learning approach has been shown to be able to deal with high variation better in other contexts. In the context of vibration prediction, a frequency-query approach could be employed to generate predictions for one specific frequency.

Q2 - ViT encoder: Image processing architectures based on convolutions encode local features. In contrast, vision transformers have a global receptive field size from early layers. As vibrations are determined by the full geometry, we expect vision transformers to perform better.

Q3 - Velocity field prediction: We can train networks to either directly predict the aggregate frequency response $\mathcal{F}$ or to predict the velocity field $\mathbf{V}$ and compute $\mathcal{F}$ from $\mathbf{V}$ via Equation 1. For predicting the velocity field, much richer training data is available, since it describes a field over the plate instead of the scalar frequency response. Most of this information is not represented in the frequency response.
In the following, we describe architectural variations explored for these aspects.

3.1 Geometry Encoder Φ

To parse the plate geometry into a feature vector, we employ three variants: ResNet18 [31, RN18], a vision transformer [32, 33, ViT] and the encoder part of a UNet [34]. For the RN18, we replace batch normalization with layer normalization [35], as we found this to work substantially better. Compared to the CNN-based RN18, the ViT architecture supports interactions across different image regions in early layers. For both, the RN18 and the ViT encoder, we obtain a feature vector $\mathbf{x}$ by average pooling the last feature map. Since the UNet generates velocity fields, no pooling is applied.

FiLM conditioning. For including the scalar parameters $\mathbf{m}$, we introduce a film layer [36]. The film layer first encodes the scalar parameters with a linear layer. The resulting encoding is then multiplied element-wise with the feature of the encoder and a bias is added. This operation is applied before the last layer of the geometry encoder (UNet) or after it (RN18, ViT).

3.2 Decoder Ψ

FQO-RN18 and FQO-ViT: Predicting $\mathcal{F}(f)$ directly. Having obtained an encoding of the plate geometry and properties $\mathbf{x}$, a decoder now takes this as well as a frequency query as input and maps them towards a prediction. For the RN18 and ViT geometry encoders, the decoder is implemented by an MLP taking both $\mathbf{x}$ and a scalar frequency value f as input to predict the response for that specific query frequency, i.e. $\Psi(\mathbf{x}, f) \in \mathbb{R}$. The frequency query is merged to $\mathbf{x}$ by a film layer [36]. By querying the decoder with all frequencies individually, we obtain results for the frequency band between 1 and 300 Hz. The MLP has six hidden layers with 512 dimensions each and ReLU activations.

FQO-UNet: Predicting $\mathcal{F}(f)$ through the velocity field $\mathbf{V}(f)$. To incorporate physics-based contraints and take advantage of the larger amount of available data, we employ a UNet to predict the velocity fields, $\mathbf{V}(f)$. From $\mathbf{V}(f)$, we derive the frequency response $\mathcal{F}(f)$ (analogous to Equation 1). A frequency query, introduced via a FiLM layer after the encoder, enables frequency-specific predictions. To reduce the memory and computation demands per geometry during training, we select a random subset of k frequency queries per geometry in a batch, with $k < 300$. If not otherwise specified, k is set to 50.

Table 1: Test results for frequency response prediction. Column $\mathbf{VF}$ indicates if $\mathcal{F}$ is indirectly predicted through the velocity field (Q3), column $\mathbf{FQ}$ indicates if frequency queries (Q1) are used. Q1 to Q3 refer to the model components described in Section 3.

			V-5000				**G-5000**			
	FQ	**VF**	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$
			Baselines							
k-NN	-	-	0.63	21.50	0.45	8.7	0.88	32.48	0.68	21.0
RN18 + FNO	-	-	0.42	10.76	0.34	5.6	0.28	14.12	0.21	6.1
DeepONet	✓	-	0.49	16.91	0.48	5.4	0.44	23.05	0.57	9.9
FNO (velocity field)	-	✓	0.47	13.10	0.36	6.3	0.49	21.16	0.39	10.7
Grid-RN18	-	-	0.44	13.29	0.36	5.4	0.30	14.95	0.26	6.5
FQO-RN18 (Q1)	✓	-	0.32	10.70	0.17	5.3	0.24	13.51	0.13	5.1
FQO-ViT (Q2)	✓	-	0.68	20.96	0.54	7.1	0.52	24.34	0.49	11.5
Grid-UNet	-	✓	0.19	7.57	0.24	2.7	0.17	9.41	0.14	4.6
FQO-UNet	✓	✓	0.08	4.24	0.07	1.7	0.11	7.47	0.08	3.1

Grid-Unet and Grid-RN18: Predicting $\mathcal{F}$ for a fixed grid of frequencies. To ablate the frequency-query approach, we employ two variations of the FQO-RN18 and FQO-UNet architectures, that do not employ frequency queries. They instead generate predictions for 1-300 Hz at once. This is done by setting the output size of the respective last layer to 300.

3.3 Baseline Methods

We further report baseline results on the following alternative methods: A k-Nearest Neighbors regressor, that finds the nearest neighbors in the latent space of an autoencoder. DeepONet [11], with a RN18 as backbone and a MLP to encode the query frequencies as a branch net. Two architectures based on Fourier Neural Operators [15]. One employing an FNO as a replacement for the query-based decoder based on RN18 features. The second directly takes the input geometry and is trained to map it to the velocity fields.

3.4 Training

All methods are trained in a data-driven fashion for 500 epochs on the training dataset of 5000 samples. 500 samples from the training dataset are excluded and employed for validation. We report evaluation results on the previously unseen test set consisting of 1000 additional samples.

For methods that predict $\mathbf{V}(f)$, i.e. UNet based methods and the FNO variation, the training loss is set to $L_{\mathbf{V}}$ where $L_{\mathbf{V}}$ represents the MSE on the log-transformed, normalized squared velocity field (Ablation on loss function in Appendix D). For methods that directly predict $\mathcal{F}$, the loss is set to L_F, the MSE on the normalized frequency response. Choosing the log-transformed quantities enables the loss to be sensitive to errors outside of resonance frequencies. Otherwise, such errors would have little influence on the total loss, as their magnitude is much lower. See Appendix C for further details on the architectures and training procedure.

4 Experiments

We train the architecture variations and baseline methods on the Vibrating Plates dataset (see Table 1). To assess which architecture aspects described in Section 3 are beneficial, we perform the following comparisons. Regarding Q1 (frequency-query approach), the Frequency-Query Operator variations consistently yield better predictions than equivalent grid-based methods, where responses for all frequencies are predicted at once: The $\mathcal{E}_{\text{MSE}}$ and the $\mathcal{E}_{\text{EMD}}$ are lower, more peaks are reproduced, and the peak positions are more precise. Regarding Q3 (velocity field prediction), predicting the velocity fields and then transforming them to the frequency response leads to better results than directly predicting the frequency response. Specifically, the UNet based architectures strongly outperform all alternatives, which we attribute to the richer training data of velocity fields. Regarding Q2, the ViT encoder leads to worse results than the CNN-based encoders.

All evaluated baseline methods achieve comparatively worse results than our proposed methods. Despite using the same RN18 geometry encoder as FQO-RN18, DeepONet [11] performs worse. We assume that this is due to incorporating frequency information through a single weighted summation, which limits the model's expressivity [37]. In contrast, FQO-RN18 introduces the queried frequency

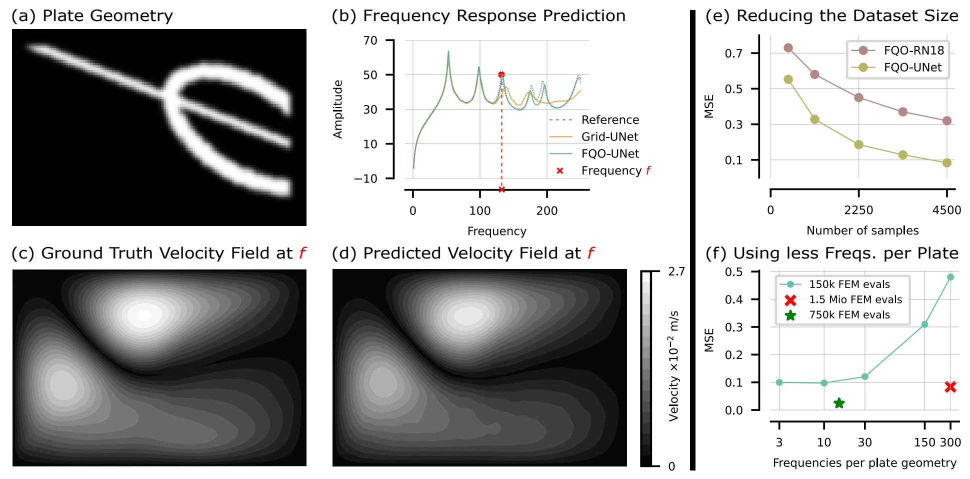

Figure 5: Results. (b) to (d) show the velocity field at one frequency and prediction for the plate geometry in (a) from FQO-UNet. (e) shows the test MSE for training two methods with reduced numbers of samples from V-5000. (f) shows effects of different data generation strategies. The blue line is an isoconture for a fixed compute budget of 150,000 data points, with varying number of frequencies per plate geometry. The green star represents using a larger dataset at 15 frequencies per plate (half of V-5000). The red cross represents a model trained on V-5000. Training with fewer frequencies per plate is more efficient.

earlier into the model. Two Fourier Neural Operator [15] baseline methods are evaluated: the first, RN18 + FNO, which substitutes the query-based decoder with an FNO decoder, underperforms compared to FQO-RN18 on both datasets. The second FNO baseline, trained directly to predict velocity fields, yields poorer results.

Results for the G-5000 setting are slightly worse than for the V-5000 setting. The difference is surprising small considering the seven additional varied parameters in the G-5000 setting. One reason might be the average number of peaks in the frequency response: the plates in G-5000 are on average smaller and because of this stiffer, leading to fewer peaks (on average 3.9 in G-5000 and 5.9 in V-5000). This interpretation is supported by the fact that the average error becomes higher with increasing frequency and thus increasing peak density (Figure 3d).

Looking at a prediction example (Figure 5a-d) for our best model, FQO-UNet, the predicted velocity field has subtle differences to the ground truth. The prediction captures the two modes and their shape quite well, but the shape is slightly less regular than in the reference. Despite that, the resulting frequency response prediction at $f = 131$ is close to the FEM reference. In comparison to the grid-based prediction, where peaks tend to be blurry, the frequency response peaks generated by FQO-UNet are more pronounced. Additional visualizations are provided in Appendix E.3 and in the code repository. For the best architecture in our experiments, FQO-UNet, we report mean and standard deviation results for multiple runs in Appendix E.2 and provide an ablation of model size for the FQO-UNet and Grid-UNet architectures in Appendix D.

Transfer learning. To quantify to which degree features learned on a subset of the design space transfer to a different subset, the V-5000 setting is split into two equally-sized parts based on the number of mesh elements that are part of a beading. The ”more beadings” set contains only 5.1 peaks on average because the plates are stiffened by the beadings, compared to 6.7 peaks on average for the ”less beadings” set. The training on plates with less beadings leads to a smaller drop in prediction quality (see Table 2). This indicates that training on data with more complex frequency responses might be more efficient. In addition, we train a single model on both G-5000 and V-5000. Performance increases, indicating that training can benefit from training with data based on similar mechanical models (Table 3).

Sample efficiency. We train the FQO-UNet and the FQO-RN18 with reduced numbers of samples (see Figure 5e). It is notable, that the FQO-UNet with a quarter of the training data achieves nearly the same prediction quality as the FQO-RN18 with full training data. This highlights the benefit of including the velocity fields into the training process. Quantitative results are given in Appendix E.1 for both dataset settings.

Table 2: Transfer learning performance: We split V-5000 into two halves based on amount of beadings and evaluate transfer learning performance across these splits: training subset $\mapsto$ test subset. The gray rows denote test results on the original subset that has been used for training.

	less beadings $\mapsto$ more beadings				more beadings $\mapsto$ less beadings			
	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$
FQO-RN18	0.61	16.19	0.20	9.3	0.82	15.79	0.36	8.3
(origin)	0.33	10.48	0.18	5.0	0.42	12.00	0.29	5.8
FQO-UNet	0.39	11.17	0.21	5.6	0.54	12.02	0.25	5.5
(origin)	0.18	8.68	0.19	2.6	0.17	7.83	0.13	3.0

Table 3: A FQO-UNet is trained in parallel on batches from V-5000 and G-5000 and evaluated on the G-5000 test set. Performance increases in all metrics.

	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$
G-5000	0.111	7.47	0.079	3.1
G-5000 + V-5000	0.093	6.97	0.071	2.9

We further investigate the optimal ratio of numbers of frequencies per geometry and total number of geometries, by generating an additional dataset in the V-5000 setting consisting of 50,000 plate geometries but with only 15 frequency evaluations per geometry. These frequencies are uniformly spaced with a random starting frequency. Reducing the frequencies per geometry drastically increases the data efficiency of our method. With a tenth of data points compared to our original dataset, the MSE metric approaches the original value (Figure 5f, quantitative results in Appendix E.1).

Design optimization. We investigate the potential of our FQO-UNet to be used for optimizing a beading pattern for reduced vibrations in a specified frequency range. Following the approach described in [38], to generate plates with reduced vibrations, a diffusion model trained to generate novel beading patterns is combined with gradient information from our FQO-UNet as follows: A gradient on the pixels of the input beading pattern is obtained by passing a beading pattern through the network, computing the sum of the predicted frequency response as a loss and then performing backpropagation to the input beading pattern. This gradient is then used to guide the diffusion model to generate beading patterns with reduced vibrations. We optimize beading patterns to reduce vibrations between 100 and 200 Hz using the FQO-UNet trained on the V-5000 dataset (Figure 6). Resulting plates have a lower mean frequency response in the targeted range than any plate in the training dataset.

5 Related Work

Acoustics. While research on surrogate models for the spatio-temporal evolution of vector fields is fairly common [39, 40, 41], directly predicting frequency responses through neural networks is an understudied problem. A general CNN architecture is applied in [42] to calibrate the parameters of an analytical model for a composite column on a shake table. The data includes spectrograms representing the structural response in time-frequency domain. The frequency-domain response of acoustic metamaterials is considered in a material design task by conditional generative adversarial networks or reinforcement learning [43, 44, 45]. The frequency response of a multi-mass oscillator is

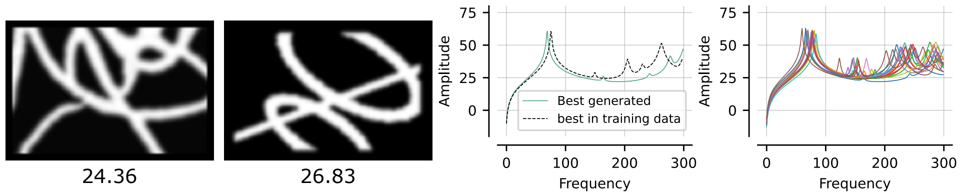

Figure 6: Design optimization. Exemplary generation result with lowest mean response between 100 Hz and 200 Hz out of 32 generations (left, mean response below). Plate with lowest response out of all 5000 training examples from V-5000 (middle left). Comparison of responses from left plates (middle right). Responses from 16 generated plates (right).

predicted with transformer-based methods [46]. Within the context of aeroacoustics, the propagation of a two-dimensional acoustic wave while considering sound-scattering obstacles is predicted in time-domain by a CNN [47, 48]. A review of machine learning in acoustics is given by [49]. Several acoustic benchmarks for numerical methods are available [50], however, these benchmarks do not systematically vary input geometries, making them not directly applicable to data-driven models.

Scientific machine learning. Data-driven machine learning techniques were successfully applied in many different disciplines within engineering and applied science; for example for alloy discovery [51], crystal structure prediction [52], climate modeling [53] and protein folding [54]. A popular use case for data-driven methods is to accelerate fluid dynamics, governed by the Navier-Stokes equations [39, 40, 55, 56, 57].
The question of how to structure and train neural networks for predicting the solution of partial differential equations (PDE) has been the topic of intense research. Many methods investigate the inclusion of physics informed loss terms [58, 59, 60, 56, 61]. Some methods directly solve PDEs with neural networks as a surrogate model [62, 63]. Graph neural networks are often employed, e.g. for interaction of rigid and deformable objects as well as fluids [64, 65].

Operator learning and implicit models. A promising avenue of research for incorporating inductive biases for physical models has been operator learning [11, 15, 66, 37, 41]. Operator learning structures neural networks such that they implement a function that can be evaluated at real values instead of a fixed discrete grid. DeepONet [11] implements operator learning by taking the value at which it is evaluated as an input and processes this value in a separate branch. Fourier Neural Operators [15] use a point-wise mapping to a latent space which is processed through a sequence of individual layers in Fourier space before being projected to the output space.
Implicit models (or coordinate-based representation) are models where location is utilized as an input to obtain a location-specific prediction, instead of predicting the entire grid at once and thus fit in the operator learning paradigm. Such models were used to represent shapes [12, 67, 68, 13], later their representations were improved [69, 70] and adapted for representing neural radiance fields (NeRFs) [71, 14]. Our method applies techniques from these implicit models to operator learning.

6 Conclusion

We introduced the problem of predicting structural vibrations and associated frequency response functions of mechanical systems. Unlike other benchmarks for deep learning surrogate models, this task necessitates predicting a steady-state solution that remains constant over time, but varies across different excitation frequencies. To this end, we created the Vibrating Plates dataset and benchmark and provide reference scores for several methods. Our Frequency-Query Operator method addresses the benchmark and achieves better results than the DeepONet and FNO baselines. We find that query-based approaches and the indirect prediction of a mean frequency response through predicted field quantities lead to better results. Surrogate models as shown in this work can greatly accelerate the prediction of physical quantities over the finite element method: Our models achieved a speed-up of around 4 to 6 orders of magnitude (see Appendix C), which makes tasks such as design optimization feasible. This efficiency, however, depends on the availability of enough pre-generated training data and requires model training. We further investigated effects of changing the composition of the training dataset and found that using less frequencies per plate and more different plates positively impacts prediction accuracy.

Limitations and future work. Our dataset and method serve as an initial step in the development of surrogate models for vibration prediction. The dataset focuses on plates, a common geometric primitive used in a great number of applications. However, many structures beyond plates exist, involving curved shells, multi-component geometries and complex material parameters. While some results from our study might transfer to these cases, more flexible architectures, able to deal with 3D data, would be needed. Different mechanical models, might also produce more complex frequency responses with e.g. more closely spaced modes, making the prediction task more challenging. As more complex geometries incur higher computational costs of FEM simulations, key questions are how to enhance sample-efficiency further, for example through transfer learning. A further limitation is the manufacturability of the considered beading patterns. The plate beadings could in principle be manufactured by deep drawing of sheet metal, but would require specifically designed stamps.

Acknowledgements. This research is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), project number 501927736, within the DFG Priority Programme

2353: Daring More Intelligence - Design Assistants in Mechanics and Dynamics'. The authors gratefully acknowledge the computing time made available to them on the high-performance computers HLRN-IV at GWDG at the NHR Centers NHR@Göttingen. These centers are jointly supported by the German Federal Ministry of Education and Research and the German state governments participating in the NHR (www.nhr-verein.de/unsere-partner).

References

[1] Mathias Basner, Wolfgang Babisch, Adrian Davis, Mark Brink, Charlotte Clark, Sabine Janssen, and Stephen Stansfeld. Auditory and non-auditory effects of noise on health. *The Lancet*, 383(9925):1325–1332, 2014.

[2] Thomas D Rossing and Neville H Fletcher. *Principles of vibration and sound*. Springer Science & Business Media, 2012.

[3] Sebastian Rothe. *Design and placement of passive acoustic measures in early design phases*, volume 2 of *Schriften des Instituts für Akustik*. Shaker, Düren, Aug 2022. Dissertation, Technische Universität Braunschweig, 2022.

[4] Olek C Zienkiewicz, Robert Leroy Taylor, and Jian Z Zhu. *The finite element method: its basis and fundamentals*. Elsevier, 2005.

[5] Klaus-Jürgen Bathe. Finite element method. *Wiley Encyclopedia of Computer Science and Engineering*, pages 1–12, 2007.

[6] Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench: An extensive benchmark for scientific machine learning. *Advances in Neural Information Processing Systems*, 35:1596–1611, 2022.

[7] Karl Otness, Arvi Gjoka, Joan Bruna, Daniele Panozzo, Benjamin Peherstorfer, Teseo Schneider, and Denis Zorin. An Extensible Benchmark Suite for Learning to Simulate Physical Systems. *arXiv preprint arXiv:2108.07799*, 2021.

[8] Florent Bonnet, Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. AirfRANS: High Fidelity Computational Fluid Dynamics Dataset for Approximating Reynolds-Averaged Navier–Stokes Solutions. *Advances in Neural Information Processing Systems*, 35:23463–23478, 2022.

[9] Ulrich Römer, Matthias Bollhöfer, Harikrishnan Sreekumar, Christopher Blech, and Sabine Christine Langer. An adaptive sparse grid rational Arnoldi method for uncertainty quantification of dynamical systems in the frequency domain. *International Journal for Numerical Methods in Engineering*, 122(20):5487–5511, 2021.

[10] Christopher Blech, Christina K Appel, Roland Ewert, Jan W Delfs, and Sabine C Langer. Wave-resolving numerical prediction of passenger cabin noise under realistic loading. In *Fundamentals of High Lift for Future Civil Aircraft: Contributions to the Final Symposium of the Collaborative Research Center 880, December 17-18, 2019, Braunschweig, Germany*, pages 231–246, 2021.

[11] Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

[12] Lars M. Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy Networks: Learning 3D Reconstruction in Function Space. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4455–4465, 2018.

[13] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. PIFu: Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2304–2314, 2019.

[14] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural Radiance Fields from One or Few Images. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4576–4585, 2020.

[15] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier Neural Operator for Parametric Partial Differential Equations. *arXiv preprint arXiv:2010.08895*, 2020.

[16] Hyun-Guk Kim, Can Nerse, and Semyung Wang. Topography optimization of an enclosure panel for low-frequency noise and vibration reduction using the equivalent radiated power approach. *Materials & Design*, 183:108125, 2019.

[17] Yong Seung Ji, Kim Kwon Hee, and Kim Young Kwan. Cabinet design for vibration reduction of a drum type washing machine. *Journal of the Korean Society for Precision Engineering*, 33(9):731–737, 2016.

[18] R. D. Mindlin. Influence of Rotatory Inertia and Shear on Flexural Motions of Isotropic, Elastic Plates. *Journal of Applied Mechanics*, 18(1):31–38, 1951.

[19] M Touratier. An efficient standard plate theory. *International journal of engineering science*, 29(8):901–916, 1991.

[20] Eduard Ventsel, Theodor Krauthammer, and EJAMR Carrera. Thin plates and shells: theory, analysis, and applications. *Appl. Mech. Rev.*, 55(4):B72–B73, 2002.

[21] Holm Altenbach, Natalia Chinchaladze, Reinhold Kienzler, and Wolfgang H Müller. *Analysis of Shells, Plates and Beams*. Springer, 2020.

[22] David L Russell and Luther W White. Formulation and validation of dynamical models for narrow plate motion. *Applied mathematics and computation*, 58(2-3):103–141, 1993.

[23] Noureddine Atalla and Franck Sgard. *Finite element and boundary methods in structural acoustics and vibration*. CRC Press, 2015.

[24] Harikrishnan K. Sreekumar and Sabine C. Langer. elPaSo Core - Elementary parallel solver core module for high performance vibroacoustic simulations, 2023.

[25] Ofir Pele and Michael Werman. Fast and robust earth mover's distances. In *2009 IEEE 12th International Conference on Computer Vision*, pages 460–467, September 2009.

[26] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99, 2000.

[27] Leonid Nisonovich Vaserstein. Markov processes over denumerable products of spaces, describing large systems of automata. *Problemy Peredachi Informatsii*, 5(3):64–72, 1969.

[28] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in Neural Information Processing Systems*, 26, 2013.

[29] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, 2020.

[30] Harold W Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[32] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[34] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015.

[35] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[36] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[37] Jacob H Seidman, Georgios Kissas, Paris Perdikaris, and George J Pappas. NOMAD: Nonlinear Manifold Decoders for Operator Learning. *arXiv preprint arXiv:2206.03551*, 2022.

[38] Jan van Delden, Julius Schultz, Christopher Blech, Sabine C Langer, and Timo Lüddecke. Minimizing structural vibrations via guided diffusion design optimization. In *ICLR 2024 Workshop on AI4DifferentialEquations In Science*, 2024.

[39] Steven L Brunton, Bernd R Noack, and Petros Koumoutsakos. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52:477–508, 2020.

[40] Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences of the United States of America*, 118, 2021.

[41] Nikola B Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew M Stuart, and Anima Anandkumar. Neural Operator: Learning Maps Between Function Spaces With Applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

[42] Angela Lanning, Arash E. Zaghi, and Tao Zhang. Applicability of Convolutional Neural Networks for Calibration of Nonlinear Dynamic Models of Structures. *Frontiers in Built Environment*, 8, 2022.

[43] Caglar Gurbuz, Felix Kronowetter, Christoph Dietz, Martin Eser, Jonas Schmid, and Steffen Marburg. Generative adversarial networks for the design of acoustic metamaterials. *The Journal of the Acoustical Society of America*, 149(2):1162–1174, 2021.

[44] Tristan Shah, Linwei Zhuo, Peter Lai, Amaris De La Rosa-Moreno, Feruza Amirkulova, and Peter Gerstoft. Reinforcement learning applied to metamaterial design. *The Journal of the Acoustical Society of America*, 150(1):321–338, 2021.

[45] Peter Lai, Feruza Amirkulova, and Peter Gerstoft. Conditional Wasserstein generative adversarial networks applied to acoustic metamaterial design. *The Journal of the Acoustical Society of America*, 150(6):4362–4374, 2021.

[46] Julius Schultz, Jan van Delden, Christopher Blech, Sabine C Langer, and Timo Lüddecke. Deep learning for frequency response prediction of a multimass oscillator. *PAMM*, page e202300091, 2023.

[47] Antonio Alguacil, Michaël Bauerheim, Marc C. Jacob, and Stéphane Moreau. Predicting the propagation of acoustic waves using deep convolutional neural networks. *Journal of Sound and Vibration*, 512:116285, 2021.

[48] Antonio Alguacil, Michael Bauerheim, Marc C Jacob, and Stéphane Moreau. Deep Learning Surrogate for the Temporal Propagation and Scattering of Acoustic Waves. *AIAA Journal*, 60(10):5890–5906, 2022.

[49] Michael J Bianco, Peter Gerstoft, James Traer, Emma Ozanich, Marie A Roch, Sharon Gannot, and Charles-Alban Deledalle. Machine learning in acoustics: Theory and applications. *The Journal of the Acoustical Society of America*, 146(5):3590–3628, 2019.

[50] Maarten Hornikx, Manfred Kaltenbacher, and Steffen Marburg. A platform for benchmark cases in computational acoustics. *Acta Acustica united with Acustica*, 101(4):811–820, 2015.

[51] Ziyuan Rao, Po-Yen Tung, Ruiwen Xie, Ye Wei, Hongbin Zhang, Alberto Ferrari, T. P. C. Klaver, Fritz Körmann, Prithiv Thoudden Sukumar, Alisson Kwiatkowski da Silva, Yao Chen, Zhiming Li, Dirk Ponge, Jörg Neugebauer, Oliver Gutfleisch, Stefan Bauer, and Dierk Raabe. Machine learning–enabled high-entropy alloy discovery. *Science*, 378:78–85, 2022.

[52] Kevin M. Ryan, Jeff Lengyel, and Michael Shatruk. Crystal Structure Prediction via Deep Learning. *Journal of the American Chemical Society*, 140 32:10158–10168, 2018.

[53] Stephan Rasp, Michael S. Pritchard, and Pierre Gentine. Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences of the United States of America*, 115:9684–9689, 2018.

[54] John M. Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Zídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A A Kohl, Andy Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David A. Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596:583–589, 2021.

[55] Octavi Obiols-Sales, Abhinav Vishnu, Nicholas Malaya, and Aparna Chandramowlishwaran. CFDNet: a deep learning-based accelerator for fluid simulations. *Proceedings of the 34th ACM International Conference on Supercomputing*, 2020.

[56] Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards Physics-informed Deep Learning for Turbulent Flow Prediction. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019.

[57] Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin. Accelerating Eulerian Fluid Simulation With Convolutional Networks. *International Conference on Machine Learning*, pages 3424–3433, 2017.

[58] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[59] Ehsan Haghighat, Maziar Raissi, Adrian Moure, Hector Gomez, and Ruben Juanes. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 379:113741, 2021.

[60] Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 26548–26560. Curran Associates, Inc., 2021.

[61] N. Heilenkötter and T. Freudenberg. torchPhysics GitHub repository. `https://github.com/boschresearch/torchphysics`, 2023.

[62] Bing Yu et al. The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 2018.

[63] Jie Bu and Anuj Karpatne. Quadratic residual networks: A new class of neural networks for solving forward and inverse problems in physics involving pdes. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 675–683, 2021.

[64] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in Neural Information Processing Systems*, 29, 2016.

[65] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. *International Conference on Machine Learning*, pages 8459–8468, 2020.

[66] Lu Lu, Xuhui Meng, Shengze Cai, Zhiping Mao, Somdatta Goswami, Zhongqiang Zhang, and George Em Karniadakis. A comprehensive and fair comparison of two neural operators (with practical extensions) based on FAIR data. *Computer Methods in Applied Mechanics and Engineering*, 393:114778, 2022.

[67] Zhiqin Chen and Hao Zhang. Learning Implicit Fields for Generative Shape Modeling. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5932–5941, 2018.

[68] Jeong Joon Park, Peter R. Florence, Julian Straub, Richard A. Newcombe, and S. Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 165–174, 2019.

[69] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020.

[70] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *Advances in Neural Information Processing Systems*, 2020.

[71] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[72] Patrick R Amestoy, Iain S Duff, Jean-Yves L'Excellent, and Jacko Koster. Mumps: a general purpose distributed memory sparse solver. In *International Workshop on Applied Parallel Computing*, pages 121–130. Springer, 2000.

[73] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[74] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[75] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

A Dataset Construction

A.1 The Mechanical Model

In the following, we give a technical description of the mechanical model that is applied to generate the datasets. For moderately thin plates, the plate theory by Mindlin is a valid differential equation [18]:

$$B\,\nabla^4 u_z - \omega^2 \rho_s h\, u_z + \omega^2 \left(\frac{B\rho_s}{G} + \rho_s I \right)\, \nabla^2 u_z + \omega^4 I \frac{\rho_s^2}{G}\, u_z = p_l$$

This equation is combined with a disc formulation for in-plane loads in order to receive a shell formulation for the mechanical description of arbitrarily formed moderately thin structures considering in-plane and transverse loads. The plate part is the dominating and important part for resolving bending waves. In the equation, u_z denotes the normal displacement of the plate structure as degree of freedom of interest. B represents the bending stiffness, ρ_s the density, h the thickness, G the shear modulus and I the moment of inertia. The angular frequency ω is defined as $\omega = 2\pi f$. The right hand-side excitation p_l describes an applied pressure load, which is converted to point forces through integration. As boundary conditions we apply homogeneous dirichlet boundary conditions, i.e. $u_z(x) = 0$ on the boundary and include a rotational stiffness at the boundary to model different boundary conditions, ranging from free rotating to clamped plates. The equation is transformed into a weak integral formulation by weighted residuals, discretized using finite elements and integrated numerically. In particular, we use triangular shell elements with 3 nodes and linear ansatz functions. The integration delivers the sparse linear system of equations. This linear system is solved using the direct solver MUMPS [72] with a specialized FEM implementation for acoustics [24]. The discretization is chosen, such that the bending waves are resolved with a minimum of 10 nodes. The bending wave length λ_B of a plate can be calculated by

$$\lambda_B = \sqrt{\frac{2\pi}{f}}\sqrt[4]{\frac{Et^2}{12(1-\nu^2)\rho}},$$

where E is the Young's modulus, t the thickness, ν the Poisson ratio and ρ the density of the plate. The final discretization is set to 181×121 for G-5000 and 121×81 for V-5000, which is sufficient for convergence.

A.2 Datasets

The exact physical setting and variation of our mechanical model to form the V-5000 and G-5000 datasets are given in Table 4, Table 6 and Table 5. Both dataset settings contain 6000 samples, each consisting of a plate geometry with associated physical and material parameters and the computed velocity fields $\mathbf{V}(f)$ and frequency response $\mathcal{F}(f)$ for frequencies f 1 to 300 Hz. 1000 samples from the 6000 samples are selected as a test dataset and not considered during neural network training or validation. Computing a single sample out of the 6000 samples takes 2 minutes and 19 seconds on a machine with a 2 Ghz CPU (20 physical cores).

Table 4: Dataset settings. Width is the width of lines and ellipses in mm. Properties. (prop.) involves plate size, thickness, material, boundary and loading properties.

	Sample space				Sample number	
Setting	Prop.	Lines	Ellipses	Width	Train	Test
V-5000	fix	1 - 3	0 - 2	30 - 70	5000	1000
G-5000	vary	1 - 3	0 - 2	40 - 60	5000	1000

For the G-5000 dataset, the geometry, material, boundary condition and loading parameters are varied. The effect of two of the material parameters is visualized in Figure 7. Increasing the damping reduces amplitudes at resonance peaks but does not shift the overall form of the frequency response. Increasing the thickness increases the overall stiffness and shifts resonance peaks towards higher frequencies in a less regular manner. For boundary condition variation, we include one rotational stiffness parameter, which models the rotational stiffness along the x-axis at the lower and upper edge and along the y-axis at the left and right edge. The rotational stiffness is added at the respective rotational degree of freedom at the boundaries and varied as given in Table 6.

Table 5: Geometry and material parameters for V-5000 and G-5000 datasets.

	Geometry			Material (Aluminum)			
	length	width	thickness	density	Young's mod.	Poisson ratio	loss factor
V-5000	0.9 m	0.6 m	0.003 m	2700 kg/m^3	7e10 N/m^2	0.3	0.02
G-5000	0.6 - 0.9 m	0.4 - 0.6 m	0.002 - 0.004 m	2700 kg/m^3	7e10 N/m^2	0.3	0.01 - 0.03

Table 6: Loading and boundary condition parameters for V-5000 and G-5000 datasets.

	Loading (Point force)		Boundary condition (rot. stiffness)
	x-position	y-position	c_{ry}/c_{rx}
V-5000	0.36 m	0.225 m	0.0 Nm
G-5000	0.18 - 0.72 m	0.12 - 0.48 m	0.0 - 100 Nm

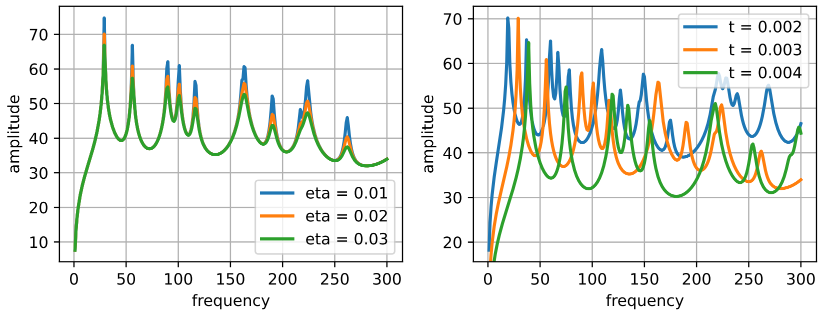

Figure 7: One-at-a-time parameter variation of the thickness parameter and the damping loss factor. Increasing the damping reduces the amplitudes at the resonance peaks. Increasing the plate thickness increases the stiffness of the plate and thus shifts the resonance peaks towards higher frequencies

B Metrics - Peak Frequency Error

We provide examples of the find_peak operation which serves as the basis for the peak frequency error on ground truth (Fig. 8) and predictions (Fig. 9, using a kNN baseline) and visualize the matched peaks for calculating the peak frequency error (Fig. 10). Note that find_peaks is run with the prominence threshold set to 0.5 meaning that the peak must be at least 0.5 units higher than their surroundings.

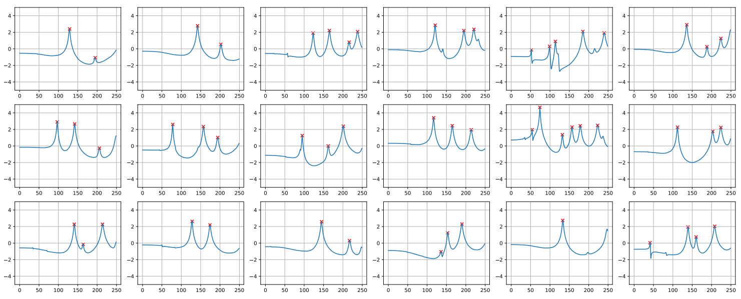

Figure 8: Find peak results on random ground truth samples.

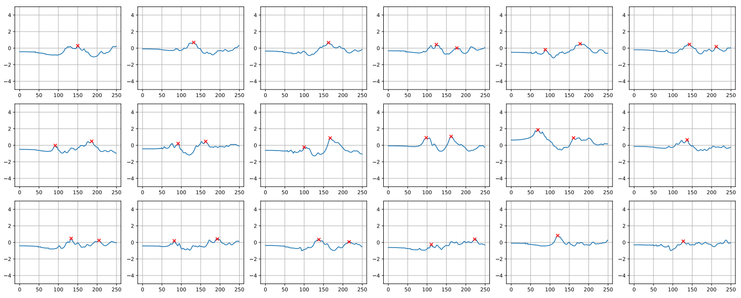

Figure 9: Find peak results on predictions.

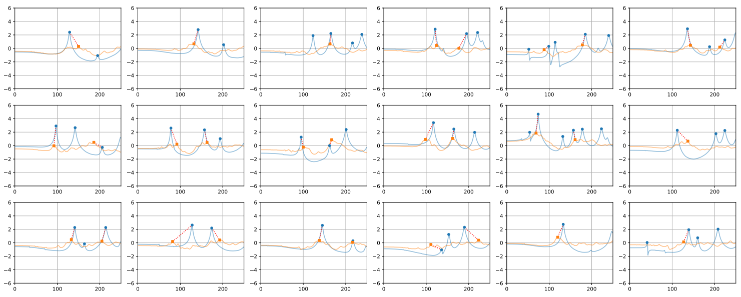

Figure 10: Visualization of the matching between ground truth (blue) and prediction (orange) peaks. Matched peaks are indicated in red.

C Architectures

In the following, our neural network architectures and the training procedure are described. The training and neural networks can be reproduced based on the publicly available code repository. To give an overall impression of the employed models, Table 7 gives an overview of the number of parameters and the speed for a forward pass prediction.

Table 7: Model size and speed comparison for a forward pass of a batch of 16 plate geometries on an A100 GPU. In comparison, solving one geometry via FEM takes 2 minutes and 19 seconds. The slowest deep learning method is then around 6000 times faster.

	# weights in Mio	Time (s)
RN18 + FNO	11.8	0.014
DeepONet	11.3	0.005
FNO (velocity field)	134	0.008
Grid-RN18	12.9	0.005
FQO-RN18	12.7	0.005
FQO-ViT	9.28	0.013
Grid-UNet	27.9	0.010
FQO-UNet	7.1	0.338
FEM (20 CPU cores)	-	$\sim$ 2224.000

C.1 Frequency-Query-based Methods

Predicting $\mathcal{F}(f)$ directly: FQO-RN18 and FQO-ViT. To directly predict the frequency response instead of predicting the velocity fields and then transforming it to the frequency response, we use a ResNet [31] and a vision transformer (ViT) [32] as geometry encoders. For the ResNet, we opt for the ResNet18 backbone. We replace batch normalization with layer normalization [35], as we found this to work substantially better. In addition, we employ the vision transformer (ViT) architecture [32]. The ViT supports interactions across different image regions in early layers. We use a variation of the ViT-Base configuration with a reduced token size of 192, an intermediate size of 768 and three attention heads. For both the RN18 and the ViT encoder, we obtain the d-dimensional global feature $\mathbf{x}$ through average pooling from the last feature map or the encoded tokens. Scalar parameters are introduced to the encoding by a film layer. As a decoder, we employ an MLP. The MLP r takes both $\mathbf{x}$ and a scalar frequency value f as input to predict the response for that specific query frequency, i.e. $r(\mathbf{x}, f) \in \mathbb{R}$. The frequency query is introduced by a film layer to $\mathbf{x}$. By querying the decoder with all frequencies individually, we obtain results for the frequency band between 1 and 300 Hz. The MLP has six hidden layers with 512 dimensions each and ReLU activation functions.

Predicting $\mathcal{F}(f)$ through the velocity field $\mathbf{V}(f)$: FQO-UNet. To predict $\mathbf{V}(f)$ instead of directly predicting $\mathcal{F}(f)$, we employ a UNet. The plate geometry is encoded by the UNet encoder and the scalar material parameters are introduced before the last contraction block of the UNet. A frequency query is introduced after the encoder again by a film layer and the decoder then produces predictions of size 40×60, which is sufficient to capture the structure and modes of the velocity fields. Since the decoder has to be evaluated for each frequency query individually, we opt to map to predictions for five velocity fields per query.

The UNet consists of three contraction blocks, two spatial-shape-preserving blocks and two expansion blocks. Additionally, two self-attention layers are included in the encoder and one self attention layer in the decoder. To ensure global features are included in the full feature volume after the encoder, adaptive average pooling is applied to the feature volume and the result is concatenated to the feature volume.

C.2 Grid-based Methods

To provide a direct comparison to the query-based approach, two methods that predict all frequency responses at once are tested.

Grid-RN18. The same RN18 is used to generate a global feature $\mathbf{x}$ as in the FQO-RN18. Given $\mathbf{x}$, an MLP r predicts the frequency response on a fixed 1 Hz interval grid, with $r(\mathbf{x}) \in \mathbb{R}^{300}$. We employ six hidden layers with 512 dimensions each and ReLU activations.

Grid-based U-Net. For the grid-based U-Net we also employ the same architecture as for the query-variation but double the number of channels to account for the larger number of predictions that the network has to produce at once. The U-Net is trained to predict all 300 velocity fields at once.

C.3 Baseline Methods

RN18 + Fourier Neural Operator (FNO). A 1d FNO as constructed by [15] takes as input $\mathbf{x}$ processed by a linear layer to size 300, the number of frequencies to be predicted. We keep 32 modes and use eight FNO blocks with 128 hidden channels.

DeepONet. We further test a DeepONet with the RN18 as branch network and as trunk network, a four layer MLP of width 128 and 512 as output width to match the size of $\mathbf{x}$. The trunk network processes the frequency queries and is then combined with $\mathbf{x}$ to produce the prediction [11, 66]. Note, the RN18 branch network is the same as the encoder of the FQO-RN18.

FNO (velocity fields). The FNO takes as input the geometry interpolated to the resolution 40×60. The 2d FNO then consists of eight FNO blocks with 128 hidden channels and finally 300 output channels to map to the 300 velocity fields in this resolution. In the FNO blocks, 32 modes are preserved after the Fourier transform. Scalar parameters are introduced by a film layer after the first FNO block.

C.4 k-Nearest Neighbors (k-NN)

We further test a k-Nearest Neighbors algorithm as a baseline, which predicts the frequency response of a plate as the mean frequency response of the k closest plates in the training set. To determine the distance between different plate designs, we use the cosine distance on the 96-dimensional latent space of a convolutional autoencoder [73] trained on the beading pattern geometries. The normalized scalar properties are appended to the latent space to include them. To obtain a prediction, the frequency responses of the k neighbors are averaged and the optimal k in the range $[1, 25]$ is empirically determined to minimize the MSE. Determining the nearest neighbors directly in the geometry space was tried out, but yielded worse results.

C.5 Training

The networks are trained using the AdamW optimizer [74], with $\beta = [0.9, 0.99]$ and weight decay set to 0.00005. We further choose a cosine learning rate schedule with a warm-up period [75] of 50 epochs. The maximum learning rate is set to 0.001, except for the UNet and ViT architectures, for which it is set to 0.0005. In total, the networks are trained for 500 epochs. As a validation set, 500 samples from the training dataset are set and excluded from the training and the checkpoint with the lowest MSE on these samples is selected. We report evaluation results on the previously unseen test set.

Compute resources. All trainings reported in this work were computed on a cluster with single A100 GPUs. The most resource intense training run took roughly 1d and 16h on a single A100 GPU and was the ablation of the number of channels with the highest scaling factor for the FQO-UNet method detailed in Section D. All other training runs with the FQO-UNet were completed in less than 24h. The trainings for the other methods were substantially shorter with i.e. the Grid-UNet finishing training in roughly 2h - 3h and likewise the FNO (velocity field) method. The roughly 20 to 30 training runs for the FQO-UNet dominate the required compute resources. We estimate that it took in total 1 A100 for 30 days. In addition, preliminary and failed experiments required further computational resources.

D Ablations

We provide ablation results for the loss function for training methods that predict the velocity field. We consider the loss function $L_{total} = \alpha L_{\mathbf{V}} + (1 - \alpha) L_F$ and provide results in Table 8. This ablation was performed with training batches consisting of 300 frequencies per geometry, instead of a subset. We find that the loss on the velocity field prediction $L_{\mathbf{V}}$ is more important than the loss on the frequency response.

Table 8: Ablation of α value for weighing loss on the predicted velocity field vs. predicted frequency response. Higher α value indicates more weight on velocity field. 1 indicates only velocity field loss. The selected α parameter is printed in bold.

	V-5000			
α	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$
0	0.25	8.02	0.15	4.4
0.5	0.15	5.52	0.11	2.9
0.9	0.09	3.90	0.08	1.8
1	0.09	4.00	0.07	1.8

We provide ablation results on the number of channels in the FQO-UNet and Grid-UNet architectures in Table 9. For the FQO-UNet, this ablation was performed with training batches consisting of 300 frequencies per geometry, instead of a subset. We find that increasing the number of channels did not lead to a perfomance improvement for the Grid-UNet and leads to marginal further improvements for the FQO-UNet architecture.

Table 9: Ablation of number of channels of FQO-UNet and Grid-UNet. The number of channels is multiplied by a constant factor over the depth, named width. The selected width parameter is printed in bold.

	V-5000			
width	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$
FQO-UNet				
16	0.12	5.00	0.09	2.2
32	0.09	3.90	0.08	1.8
64	0.08	3.82	0.07	1.8
Grid-UNet				
16	0.31	11.44	0.31	3.9
32	0.24	8.56	0.25	3.3
64	0.19	7.57	0.24	2.7
128	0.24	8.17	0.21	3.7

E Additional Results

E.1 Sample Efficiency

To provide full baseline results for the training with a reduced amount of samples, we refer to Table 10.

Table 10: Test results for different training dataset sizes for both settings, V-5000 and G-5000.

	V-5000				**G-5000**			
	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$
FQO-UNet								
10 %	0.55	15.26	0.37	6.4	0.54	22.70	0.37	11.1
25 %	0.33	12.00	0.24	4.2	0.33	16.67	0.17	7.3
50 %	0.19	8.54	0.17	2.9	0.19	11.02	0.12	4.7
75 %	0.13	6.95	0.13	2.3	0.14	9.14	0.10	3.8
Full dataset	0.08	4.24	0.07	1.7	0.11	7.47	0.08	3.1
FQO-RN18								
10 %	0.73	19.44	0.49	7.9	0.65	27.57	0.55	13.9
25 %	0.58	14.27	0.31	7.2	0.45	21.04	0.44	8.9
50 %	0.45	12.20	0.20	6.4	0.35	17.00	0.15	7.2
75 %	0.37	10.96	0.18	5.5	0.29	15.06	0.15	5.9
Full dataset	0.32	10.70	0.17	5.3	0.24	13.51	0.13	5.1

Full results on training with different ratios of frequencies and geometries are provided in Table 11.

Table 11: Data generation experiment.

		V-5000			
# Freqs	# geometries	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$
300	500	0.48	13.16	0.32	5.7
150	1,000	0.31	11.18	0.22	4.3
30	5,000	0.12	8.74	0.14	2.1
10	15,000	0.10	11.18	0.17	1.6
3	50,000	0.10	11.47	0.20	1.4
15	50,000	0.02	4.06	0.04	0.08
original 300	5,000	0.08	4.24	0.07	1.7

E.2 Multiple Trainings with Random Splits

To provide a notion of the variability of the results for different training runs, we performed four trainings for the FQO-UNet with different initial seeds and different random splits in training and validation set. These trainings were performed with training batches consisting of all 300 frequencies per geometry, instead of a subset. In Table 12 we report evaluation results on the respective validation sets and on the unseen test set and observe only modest variation.

Table 12: 4 models were trained on random splits in training and validation sets (4500 and 500 samples respectively). The results are denoted as mean [standard deviation].

	V-5000			
Evaluation set	$\mathcal{E}_{\text{MSE}}$	$\mathcal{E}_{\text{EMD}}$	$\mathcal{E}_{\text{PEAKS}}$	$\mathcal{E}_{\text{F}}$
Validation set	0.096 [0.0023]	4.1 [0.072]	0.081 [0.0071]	2 [0.061]
Test set	0.094 [0.0031]	4.1 [0.091]	0.08 [0.003]	1.9 [0.096]

E.3 Visualizations

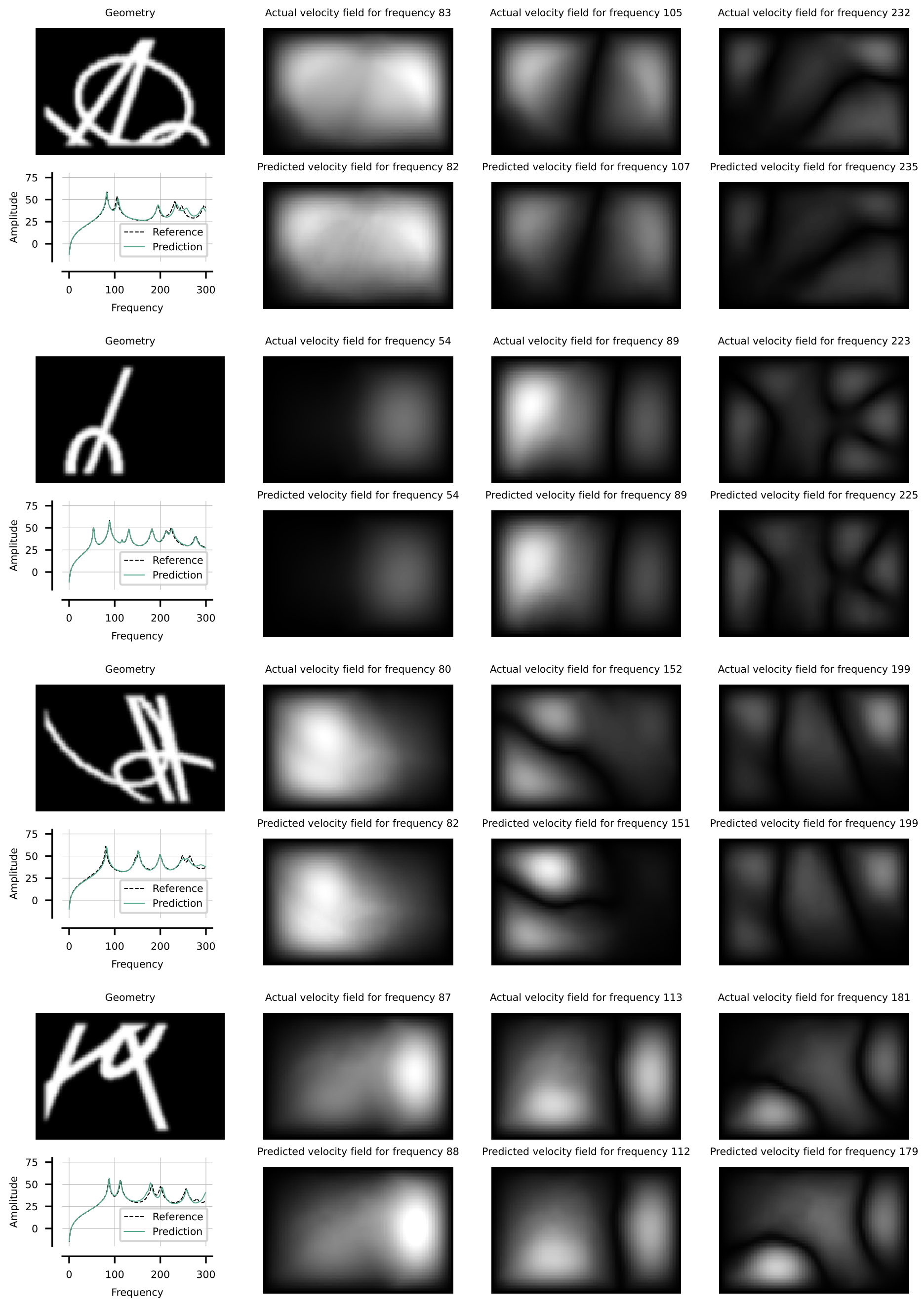

Figure 11: V-5000 example predictions. The velocity fields at the three peaks with the highest amplitude are shown. The plots are scaled with respect to the maximum velocity in the prediction and reference velocity field to make the differences in magnitude visible.

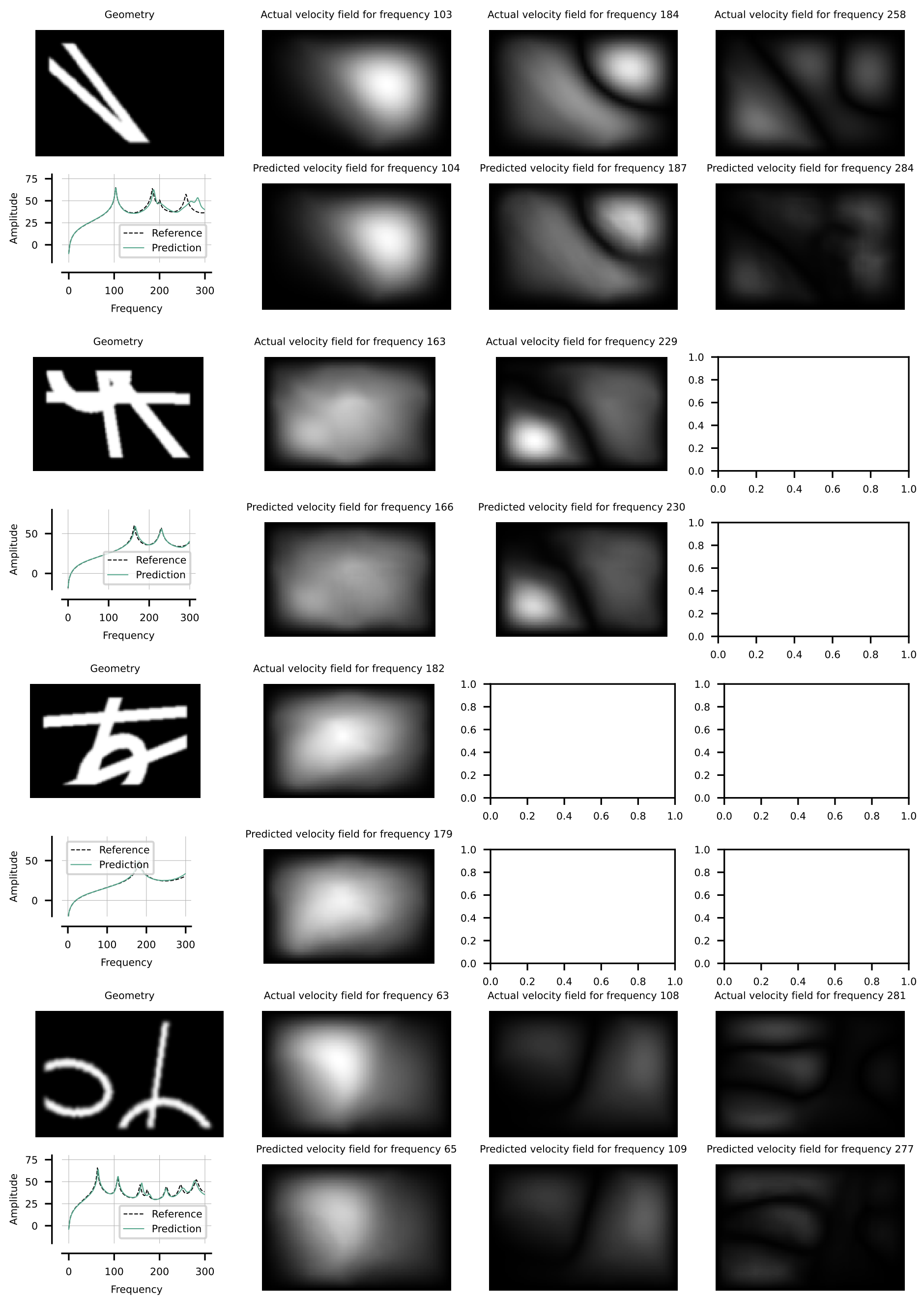

Figure 12: G-5000 example predictions. The velocity fields at the three peaks with the highest amplitude are shown. Empty axes indicates less than three peaks in the response. The plots are scaled with respect to the maximum velocity in the prediction and reference velocity field to make the differences in magnitude visible.

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: This paper introduces a dataset and method for predicting vibrations of plates. This is communicated in the abstract and introduction.

   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 included a limitations paragraph, detailing the limitation in scope of our data as well as our method. Computational efficiency is also discussed and numerical values are given in the Appendix.

   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: [NA]

Justification: This paper does not include theoretical results, as it mainly deals with applicational issues.

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: Experimental reproducibility is guaranteed by providing code and data. Further, we included several sections in the appendix, giving concrete architectural and training details and describing the simulation setup to obtain our dataset. Evaluating design optimization results requires proprietary Finite-Element simulation software.

   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?

   Answer: [Yes]

   Justification: The link to code and data is included directly in the abstract. It is `https://github.com/ecker-lab/Learning_Vibrating_Plates`

   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, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: In addition to our code, Section C.5 in the Appendix details specific training settings. The training and test datasets are provided in separate files.

   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: In Appendix E.2 mean and standard deviation results for four training runs with different seeds and training and validation splits are reported, giving a notion of variability.

   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).
- 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: Information on compute resources for deep learning is provided in Section C.5. Information on compute resources for the dataset generation is provided in Section A.2.

   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: Our research conforms with the 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: Including deep learning in the engineering pipeline for actual applications is at a very preliminary stage.

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.
- 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: Our data and models do not have a high risk for misuse.

    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: [NA]

    Justification: This work does not use existing assets, save for publically available codebases like pytorch.

    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.
- 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: The dataset included in our work is published in a curated dataset repository hosted by the University of Göttingen and accompanied by structured metadata. It is available from: `https://doi.org/10.25625/UWF7RB`

    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: Human subjects were not involved.

    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: Human subjects were not involved.

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.