SIMULATION-BASED INFERENCE FOR PLATE REVERB SYSTEM IDENTIFICATION

Dylan Sechet Marc Evrard Matthieu Kowalski

Laboratoire Interdisciplinaire des Sciences du Numérique
Université Paris-Saclay, Inria, CNRS, CentraleSupélec
firstname.lastname@lisn.fr

ABSTRACT

We address Task A of the 1st DAFx Parameter Estimation Challenge, which aims to retrieve the physical parameters of a plate model from an impulse response. To do so, we use the Simulation-Based Inference (SBI) framework, in which we train a neural network to estimate a density over plate parameters given an impulse response, using a dataset generated by the simulator. Inference for a new impulse response then requires only a forward pass through the network, without involving the simulator.

For each test observation, we fine-tune a specific network: additional simulation rounds are performed by sampling parameters from the current estimated distribution, simulating the corresponding impulse responses, and fine-tuning to produce the specialized network.

1. INTRODUCTION

Plate reverberation has a long history as an audio effect, from the electromechanical units of the 1960s such as the EMT 140 to their present-day digital emulations. Its physical behaviour is by now well understood: a plate reverberator can be described as a thin metal plate undergoing small transverse vibrations, governed by the damped Kirchhoff-Love equation, and a range of accurate digital models have been built on this basis, whether by direct numerical simulation of the plate equation [1] or by modal decomposition [2].

While the forward problem is well understood, the inverse problem is considerably harder, and recovering either material or modal parameters of the response from audio impulse responses alone remains a challenging task. These difficulties motivated the 1st DAFx Parameter Estimation Challenge [3], a data challenge for plate-reverb system identification using a common simulator and evaluation protocol. In this work, we tackle Task A of the challenge: given the impulse response of a simulated plate, estimate its identifiable physical parameters.

Parameter estimation for audio effects has been approached from several directions, commonly characterized as white, grey or black-box modelling depending on how much knowledge of the target system is assumed. Black-box neural networks can emulate plate and spring reverberation with high fidelity [4], but expose no physical parameters. Grey-box methods keep a known processing structure and fit its parameters by gradient descent, as in differentiable digital signal processing [5]. Such gradient-based fitting relies on a differentiable forward model, which the challenge simulator is not.

Simulation-Based Inference (SBI) removes that requirement: it treats the simulator as a black box queried only through samples, learning the inverse map and returning a full posterior over the physical parameters.

2. PROBLEM SETUP

The challenge's reference model [3] is a thin rectangular plate of side lengths $L_x \times L_y$ undergoing small transverse vibrations, governed by the damped Kirchhoff-Love equation with simply-supported boundaries and simulated by modal synthesis. The observed data is the impulse response (IR): the transverse displacement recorded at a readout point (x_o, y_o) following a Dirac impulse applied at a fixed input point $(0.335\,L_x,\ 0.467\,L_y)$. Each IR is a 5 s displacement signal sampled at 44.1 kHz, giving us 220 500 samples.

The raw material parameters (density ρ, thickness h, Young's modulus E, tension per unit length T_0) affect the IR only through the three invariants $\mu := \rho h$, D/μ and T_0/μ, with D the flexural rigidity. Task A therefore asks to estimate, from a single IR, the six identifiable parameters

$$\boldsymbol{\alpha} := \{\mu,\ D/\mu,\ T_0/\mu,\ L_y,\ x_o,\ y_o\}, \tag{1}$$

all remaining parameters being fixed across simulations. In the following, $\boldsymbol{\alpha} \in [0,1]^6$ is normalized to the unit hypercube by the ranges of Table 1. The IRs are provided unnormalized, as the absolute amplitude scale is inversely proportional to μ and would be discarded by peak normalization.

Table 1: Ranges of the six estimated parameters, used to normalize the NMSE (2). The invariant ranges (μ, D/μ, T_0/μ) are those induced by the raw-parameter ranges [3]; $L_x = 1$ m is fixed.

Parameter	Symbol	Range	Unit
Areal mass density	μ	$[2.43,\ 106.15]$	$\mathrm{kg\,m^{-2}}$
Bending-to-mass ratio	D/μ	$[0.28,\ 201.2]$	$\mathrm{m^4\,s^{-2}}$
Tension-to-mass ratio	T_0/μ	$[9.4{\times}10^{-5},\ 411.5]$	$\mathrm{m^2\,s^{-2}}$
Plate length (y)	L_y	$[1.1,\ 4.0]$	m
Readout position (x)	x_o	$[0.51,\ 1.0]$	frac. L_x
Readout position (y)	y_o	$[0.51,\ 1.0]$	frac. L_y

Estimates are scored by the challenge's range-normalized mean squared error, averaged over the six parameters:

$$\mathrm{NMSE} = \frac{1}{6}\sum_{i=1}^{6} \frac{\left(\alpha_i^{\mathrm{est}} - \alpha_i^{\mathrm{ref}}\right)^2}{\left(\alpha_i^{\max} - \alpha_i^{\min}\right)^2}. \tag{2}$$

A constant predictor taking the midpoint of each range achieves a NMSE of roughly 83×10^{-3}, which serves as a random-guess reference [3].

3. SIMULATION-BASED INFERENCE

Let $\mathbf{x} \in \mathbb{R}^T$ be an observed impulse response of length T. The plate simulator defines a forward model $\mathbf{x} = f(\boldsymbol{\alpha})$.

We wish to evaluate the likelihood $p(\mathbf{x} \mid \boldsymbol{\alpha})$, but doing so in closed form would be intractable. The goal of Simulation-Based Inference is to learn the inverse map: given $\mathbf{x}$, recover a distribution over the parameters $\boldsymbol{\alpha}$ that could have produced it. Concretely, we seek a neural density estimator $q_\theta(\boldsymbol{\alpha} \mid \mathbf{x})$ that approximates the true posterior

$$p(\boldsymbol{\alpha} \mid \mathbf{x}) \propto p(\mathbf{x} \mid \boldsymbol{\alpha})\, p(\boldsymbol{\alpha}) \tag{3}$$

where $p(\boldsymbol{\alpha})$ is the challenge's generating distribution: the raw parameters are sampled uniformly within their ranges, which induces a non-uniform distribution over the invariants μ, D/μ and T_0/μ.

Rather than evaluating $p(\mathbf{x} \mid \boldsymbol{\alpha})$ directly, q_θ is trained by maximum likelihood on a dataset $\{(\boldsymbol{\alpha}_n, \mathbf{x}_n{:=}f(\boldsymbol{\alpha}_n))\}_{n=1}^N$ of simulator-generated pairs, minimizing:

$$\mathcal{L}(\theta) := -\frac{1}{N} \sum_{n=1}^{N} \log q_\theta(\boldsymbol{\alpha}_n \mid \mathbf{x}_n). \tag{4}$$

Once trained, inference on a new observation $\mathbf{x}_0$ requires only a forward pass: samples are drawn directly from $q_\theta(\boldsymbol{\alpha} \mid \mathbf{x}_0)$ without any further simulator calls.

4. ARCHITECTURE & TRAINING

The neural network q_θ is built from a neural spline flow described in Section 4.1 and a summary network described in Section 4.2, which interact as described in Fig. 1.

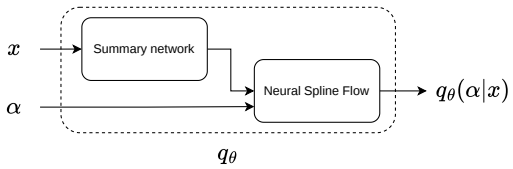

Figure 1: Summary of the neural architecture.

4.1. Density Estimator

The conditional density q_θ is parameterized as a Neural Spline Flow [6] (8 transforms, 256 hidden features) conditioned on a summary embedding of $\mathbf{x}$; both are trained jointly end-to-end by minimizing Eq. (4), using the `sbi` toolkit [7].

4.2. Summary Network

Because the impulse response is a very large time series, the flow cannot easily condition on it directly: a summary network is first used to compress it into a lower-dimensional embedding.

The summary network is a CNN-based architecture that generates an embedding of the impulse response, which is then used to condition the normalizing flow.

The impulse response is first used to generate 3 log-magnitude spectrograms at different resolutions (window sizes of 512, 2048, and 8192). Each of them is processed by a three-layer 2D CNN using 3×3 convolutions, followed by adaptive average pooling to a fixed 4×4×64 spatial map, yielding a 1024-dimensional vector per branch, as shown in Fig. 2.

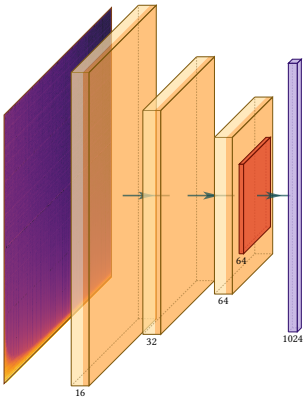

Figure 2: Architecture of one CNN branch.

The outputs of all branches are concatenated, producing a 3072-dimensional embedding. We then append the log-RMS amplitude of the raw IR to improve the reconstruction of μ, as the IR's amplitude scale is inversely proportional to it.

The resulting 3073-dimensional vector is finally projected via a two-layer MLP into a 256-dimensional embedding, which is then passed on to the neural spline flow.

4.3. Training

The model is trained on 60 000 simulator-generated pairs drawn from the prior by uniform sampling of the raw plate parameters, for 145 epochs with batch size 128, using the Adam optimizer with learning rate 10^{-3}. Training takes approximately 26 hours on an NVIDIA A100. Early stopping on a held-out validation split is used to prevent overfitting.

5. SEQUENTIAL REFINEMENT

From the previously described architecture and training, we have a neural network q_θ that provides a good estimate for any plate but has to cover the entire parameter space at once. We would prefer a network that can be worse globally but has very good performance around the test observations we are specifically interested in.

To achieve this, for each test IR, we run a few extra rounds of simulation targeted at that specific observation and fine-tune the network on the results. This creates estimators that perform worse globally but better around the specific target point. This sequential refinement strategy is a standard technique in SBI [8, 7], and we follow the SNPE-C objective [9], which ensures the refined posterior remains correctly calibrated even though the new simulations are no longer drawn from the prior. Unlike offline training, this correction requires evaluating the prior density $p(\boldsymbol{\alpha})$, which we compute in closed form, the main difficulty being the non-uniform distribution induced on the derived invariants.

For test point $\mathbf{x}^*$, for each round r:

- **Propose.** Draw a batch of proposals $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_B$ from the current estimate $q_\theta^{(r-1)}(\boldsymbol{\alpha} \mid \mathbf{x}^*)$, with $q_\theta^{(0)}$ being the initially trained density estimator from Section 4.

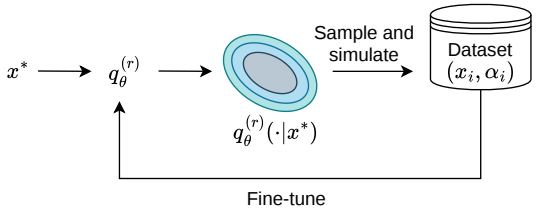

Figure 3: Sequential SBI refinement.

- **Simulate.** Run the forward simulator at each proposal to obtain the corresponding impulse responses $\mathbf{x}_1, \ldots, \mathbf{x}_B$, producing a dataset concentrated near the current estimate's mode. Each proposal is mapped to raw plate parameters by drawing ρ uniformly at random and recovering h, E and T_0 from the invariants.
- **Fine-tune.** Fine-tune q_θ using the atomic SNPE-C objective [9], which corrects for the non-prior proposal distribution to ensure the refined model still targets the true posterior $p(\boldsymbol{\alpha} \mid \mathbf{x}^*)$.

For each test point, we fine-tune the model for 3 rounds, each round generating an extra 10 000 IRs near the current estimate. For each test point, the 3 rounds take a total of $\sim$2.5 h to run on an A100 GPU.

The final point estimate is obtained by averaging over samples drawn from the refined density. As we have found that refinement can cause the model to degenerate on some observations, we additionally guard against such failures with a selection step, described and evaluated in Section 6.

6. RESULTS AND DISCUSSION

To evaluate our model's performance, we generated a custom test dataset of 50 random IRs following Task A's format. Table 2 reports the per-parameter NMSE for different estimators, while Fig. 4 shows the corresponding error distributions. The table reports averages, which on this task are dominated by a few catastrophic failures of the refined estimator; the figure shows what happens on typical IRs. Both views are needed to understand the results.

We compare three estimators. The *offline* model is the base density estimator $q_\theta^{(0)}$ of Section 4, trained entirely offline and requiring no fine-tuning at inference time. The *refined* estimator is the per-observation model obtained directly by the sequential procedure of Section 5. Finally, the *selected* estimator is the one we submitted to the challenge[1]: it tries to pick the best between the offline and refined estimators for each test IR, by keeping whichever yields the lower ℓ_2 waveform reconstruction error.[2][2][2]

The *oracle* represents the upper bound obtained by always picking the lower-NMSE estimate between the offline and refined methods for each IR.

All our estimators improve substantially over the challenge's PSO baseline, whose global NMSE of 50.56×10^{-3} remains close to the constant-predictor reference of 83×10^{-3} (Section 2). The baseline particularly struggles to localize the readout position (x_o, y_o), which our models recover well.

In Table 2, refinement appears to degrade performance: the refined average NMSE is worse than the offline one. However, looking at Fig. 4, we can see this is an artifact of averaging. Refinement improves 40 of the 50 test IRs and cuts the median global NMSE from 0.75×10^{-3} to 0.13×10^{-3}. But on the few IRs where it degenerates, it increases the error by several orders of magnitude. These failures, appearing as the long upper tails in Fig. 4, are rare yet large enough to dominate the mean.

The selection step exists to catch exactly these failures and reduce the variance of refining: in Fig. 4, the refined and selected medians are close for every parameter, so on most IRs, selection changes nothing. But discarding the few catastrophic cases is enough to cut the mean global NMSE by more than a factor of 4 relative to the offline estimator, from 2.17×10^{-3} to 0.47×10^{-3}, improving every individual parameter (Table 2). The remaining gap to the oracle (0.34×10^{-3}) measures how often the ℓ_2 criterion picks the wrong estimate: finding a better criterion would hopefully close this gap.

Finally, this accuracy has a computational cost: refinement takes $\sim$2.5 h per observation at inference time, whereas the offline estimator runs in less than a second.

Table 2: Average per-parameter NMSE ($\times 10^{-3}$, lower is better) on the 50-IR test set. *PSO* is the challenge's particle-swarm baseline.

		Per parameter					
	Global	μ	D/μ	T_0/μ	L_y	x_o	y_o
PSO baseline	50.56	26.16	9.72	8.46	80.43	95.18	83.39
Offline	2.17	2.18	0.10	0.23	5.50	3.23	1.75
Refined	3.80	2.49	0.03	0.14	11.45	4.93	3.74
Selected	0.47	0.68	0.02	0.08	1.66	0.15	0.25
Oracle	*0.34*	*0.44*	*0.03*	*0.08*	*1.25*	*0.10*	*0.15*

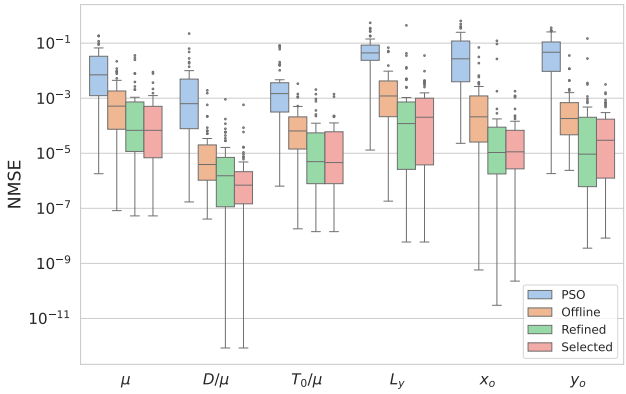

Figure 4: Distribution of the per-parameter NMSE across the 50-IR test set.

6.1. Posterior uncertainty and calibration

While the challenge forces us to produce a point estimate, one of the advantages of SBI is that we can access a full predicted distribution, which can notably help assess uncertainty.

NMSE says nothing about whether the posterior's uncertainty

[1]The version actually scored by the challenge used a uniform approximation of $p(\boldsymbol{\alpha})$ in the SNPE-C correction (Section 5) rather than the exact density, reaching a slightly worse global NMSE of 0.94×10^{-3} for the selected model (oracle: 0.45×10^{-3}) on the same test set.

is trustworthy: an overconfident model can be accurate on average yet report completely wrong distributions. To verify the model's calibration, we use TARP [10]. TARP estimates the posterior's expected coverage: across many held-out plates, it measures how often the true parameters fall within the posterior's α-probability region, at each level α. A well-calibrated posterior makes that fraction equal α at each level.

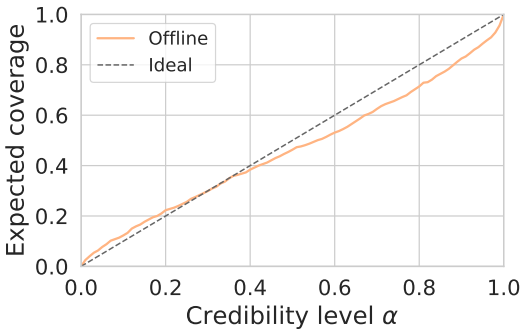

Figure 5: TARP expected coverage of the offline posterior over 500 held-out plates.

As shown in Fig. 5, the offline posterior's coverage stays relatively close to the ideal diagonal, sitting slightly below it: our predicted distributions are a bit too narrow and mildly underestimate the long tails of the true posterior.

7. CONCLUSION

We cast plate-reverb system identification as simulation-based inference: a neural spline flow, conditioned on a multi-resolution CNN summary of the impulse response, yields a posterior over the six identifiable plate parameters. Finally, a per-observation refinement stage specializes it around each test IR, with a waveform-reconstruction heuristic falling back to the offline estimator when refinement degrades the fit. The submitted estimator reduces the global NMSE from 50.6×10^{-3} for the challenge's PSO baseline to 0.47×10^{-3}.

The remaining gap to the per-IR oracle (0.34×10^{-3}) makes a better criterion for accepting refined estimates the clearest lever for future work, alongside reducing the $\sim$2.5 h per-observation cost of refinement, which currently stands against sub-second offline inference.

8. ACKNOWLEDGEMENTS

Many thanks to Sébastien Velut for his guidance on SBI methods.

This project was provided with computing and storage resources by GENCI at IDRIS under grant 2025-AD011017041 on the supercomputer Jean Zay's A100 partition.

9. REFERENCES

[1] S. Bilbao, "A Digital Plate Reverberation Algorithm," *Journal of The Audio Engineering Society*, Mar. 2007.

[2] M. Ducceschi and C. J. Webb, "Plate reverberation: Towards the development of a real-time physical model for the working musician," in *Proceedings of the 22nd International Congress on Acoustics*, 2016.

[3] M. Ducceschi and L. Gabrielli, "The 1st DAFx Parameter Estimation Challenge: A Benchmark for Plate Reverb System Identification," in *Proceedings of the 29th International Conference on Digital Audio Effects (DAFx-26)*, 2026.

[4] M. A. M. Ramírez, E. Benetos, and J. D. Reiss, "Modeling plate and spring reverberation using a DSP-informed deep neural network," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020, pp. 241–245.

[5] J. Engel, L. H. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *International Conference on Learning Representations*, 2020.

[6] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Neural spline flows," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., Dec. 2019, no. 675, pp. 7511–7522.

[7] A. Tejero-Cantero, J. Boelts, M. Deistler, J.-M. Lueckmann, C. Durkan, P. J. Gonçalves, D. S. Greenberg, and J. H. Macke, "Sbi: A toolkit for simulation-based inference," *Journal of Open Source Software*, vol. 5, no. 52, p. 2505, Aug. 2020.

[8] K. Cranmer, J. Brehmer, and G. Louppe, "The frontier of simulation-based inference," *Proceedings of the National Academy of Sciences*, vol. 117, no. 48, pp. 30 055–30 062, Dec. 2020.

[9] D. S. Greenberg, M. Nonnenmacher, and J. Macke, "Automatic Posterior Transformation for Likelihood-Free Inference," in *International Conference on Machine Learning*, May 2019.

[10] P. Lemos, A. Coogan, Y. Hezaveh, and L. Perreault-Levasseur, "Sampling-based accuracy testing of posterior estimators for general inference," in *International Conference on Machine Learning*. PMLR, 2023, pp. 19 256–19 273.