Perceptual–Neural–Physical Sound Matching

Han Han, Vincent Lostanlen, Mathieu Lagrange

PERCEPTUAL–NEURAL–PHYSICAL SOUND MATCHING

Han Han, Vincent Lostanlen, and Mathieu Lagrange

Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes, France

ABSTRACT

Sound matching algorithms seek to approximate a target waveform by parametric audio synthesis. Deep neural networks have achieved promising results in matching sustained harmonic tones. However, the task is more challenging when targets are nonstationary and inharmonic, e.g., percussion. We attribute this problem to the inadequacy of loss function. On one hand, mean square error in the parametric domain, known as "P-loss", is simple and fast but fails to accommodate the differing perceptual significance of each parameter. On the other hand, mean square error in the spectrotemporal domain, known as "spectral loss", is perceptually motivated and serves in differentiable digital signal processing (DDSP). Yet, spectral loss is a poor predictor of pitch intervals and its gradient may be computationally expensive; hence a slow convergence. Against this conundrum, we present Perceptual-Neural-Physical loss (PNP). PNP is the optimal quadratic approximation of spectral loss while being as fast as P-loss during training. We instantiate PNP with physical modeling synthesis as decoder and joint time–frequency scattering transform (JTFS) as spectral representation. We demonstrate its potential on matching synthetic drum sounds in comparison with other loss functions.

Index Terms— sound matching, auditory similarity, scattering transform, deep convolutional networks, physical modeling synthesis.

1. INTRODUCTION

Given an audio synthesizer $\boldsymbol{g}$, the task of sound matching [1] consists in retrieving the parameter setting $\boldsymbol{\theta}$ that "matches" a target sound $\boldsymbol{x}$; i.e., such that a human ear judges the generated sound $\boldsymbol{g}(\boldsymbol{\theta})$ to resemble $\boldsymbol{x}$. Sound matching has applications in automatic music transcription, virtual reality, and audio engineering [2, 3]. Of particular interest is the case where $\boldsymbol{g}(\boldsymbol{\theta})$ solves a known partial differential equation (PDE) whose coefficients are contained in the vector $\boldsymbol{\theta}$. In this case, $\boldsymbol{\theta}$ reveals some key design choices in acoustical manufacturing, such as the shape and material properties of the resonator.

Over the past decade, the renewed interest for deep neural networks (DNN's) in audio content analysis has led researchers to formulate sound matching as a supervised learning problem [4]. Intuitively, the goal is to optimize the synaptic weights $\mathbf{W}$ of a DNN $\boldsymbol{f}_{\mathbf{W}}$ so that $\boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x}_n) = \tilde{\boldsymbol{\theta}}_n$ approximates $\boldsymbol{\theta}_n$ over a training set of pairs $(\boldsymbol{x}_n, \boldsymbol{\theta}_n)$. Because $\boldsymbol{g}$ automates the mapping from parameter $\boldsymbol{\theta}_n$ to sound $\boldsymbol{x}_n$, this training procedure incurs no real-world audio acquisition nor human annotation. However, prior publications have pointed out that the approximation formula $\tilde{\boldsymbol{\theta}}_n \approx \boldsymbol{\theta}_n$ lacks a perceptual meaning: depending on the choice of target $\boldsymbol{x}_n$, some deviations $(\tilde{\boldsymbol{\theta}}_n - \boldsymbol{\theta}_n)$ may be judged to have a greater effect than others [5, 6, 7].

The paradigm of differentiable digital signal processing (DDSP) has brought a principled methodology to address this issue [8]. The key idea behind DDSP is to chain the learnable encoder $\boldsymbol{f}_{\mathbf{W}}$ with the known decoder $\boldsymbol{g}$ and a non-learnable but differentiable feature map $\boldsymbol{\Phi}$. In DDSP, $\boldsymbol{f}_{\mathbf{W}}$ is trained to minimize the perceptual distance between vectors $\boldsymbol{\Phi}(\tilde{\boldsymbol{x}}_n) = (\boldsymbol{\Phi} \circ \boldsymbol{g} \circ \boldsymbol{f}_{\mathbf{W}})(\boldsymbol{x}_n)$ and $\boldsymbol{\Phi}(\boldsymbol{x}_n)$ on average over samples $\boldsymbol{x}_n$. Yet, a practical shortcoming of DDSP is that it requires to backpropagate the "spectral loss" $\|\boldsymbol{\Phi}(\tilde{\boldsymbol{x}}_n) - \boldsymbol{\Phi}(\boldsymbol{x}_n)\|_2$ over each DNN prediction $\tilde{\boldsymbol{\theta}}_n$; and so at every training step, since $\mathbf{W}$ is iteratively updated by stochastic gradient descent (SGD).

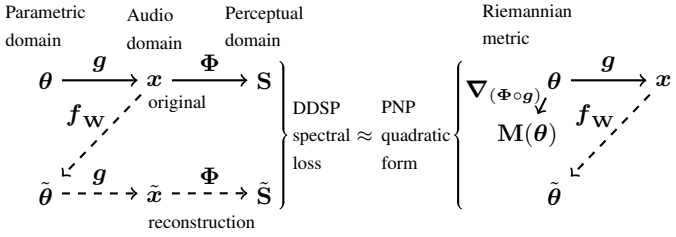

Fig. 1. Graphical outline of the proposed method. Given a known synthesizer $\boldsymbol{g}$ and feature map $\boldsymbol{\Phi}$, we train a neural network $\boldsymbol{f}_{\mathbf{W}}$ to estimate $\tilde{\boldsymbol{\theta}}$ and minimize the "perceptual–neural–physical" (PNP) quadratic form $\langle \tilde{\boldsymbol{\theta}} - \boldsymbol{\theta} | \mathbf{M}(\boldsymbol{\theta}) | \tilde{\boldsymbol{\theta}} - \boldsymbol{\theta} \rangle$ where $\mathbf{M}$ is the Riemannian metric associated to $(\boldsymbol{\Phi} \circ \boldsymbol{g})$. Hence, PNP approximates DDSP spectral loss yet does not need to backpropagate $\boldsymbol{\nabla}_{(\boldsymbol{\Phi} \circ \boldsymbol{g})}(\tilde{\boldsymbol{\theta}})$ at each epoch. Transformations in solid (resp. dashed) lines can (resp. cannot) be cached during training.

In this article, we propose a new learning objective for sound matching, named perceptual–neural–physical (PNP). Our main contribution is to compute the Riemannian metric $\mathbf{M}$ associated to the Jacobian $\boldsymbol{\nabla}_{(\boldsymbol{\Phi} \circ \boldsymbol{g})}$ over each sample $\boldsymbol{\theta}_n$ (see Section 2.1). With $\mathbf{M}(\boldsymbol{\theta}_n)$, we train $\boldsymbol{f}_{\mathbf{W}}$to minimize a locally linear approximation of spectral loss, making PNP comparable to DDSP. Yet, unlike in DDSP, the computation of $\boldsymbol{\nabla}_{(\boldsymbol{\Phi} \circ \boldsymbol{g})}$ is independent from the encoder $\boldsymbol{f}_{\mathbf{W}}$: thus, it may be parallelized and cached during DNN training. A second novelty of our paper resides in its choice of application: namely, differentiable sound matching for percussion instruments. This requires not only a fine characterization of the spectral envelope, as in the DDSP of sustained tones; but also of attack and release transients. For this purpose, we need $\boldsymbol{g}$ and $\boldsymbol{\Phi}$ to accommodate sharp spectrotemporal modulations. Specifically, we rely on original differentiable implementations of the functional transformation method (FTM) for $\boldsymbol{g}$ and the joint time–frequency scattering transform (JTFS) for $\boldsymbol{\Phi}$. [1] [2]

2. METHODS

2.1. Accelerating spectral loss with Riemannian geometry

We assume the synthesizer $\boldsymbol{g}$ and the feature map $\boldsymbol{\Phi}$ to be continuously differentiable. Let us denote by $\mathcal{L}^{\mathrm{DDSP}}$ the "spectral loss"

[1] Companion website: https://github.com/lylyhan/perceptual_neural_physical

[2] Audio examples: https://pnp.cargo.site/

associated to the triplet $(\boldsymbol{\Phi}, \boldsymbol{f}_{\mathbf{W}}, \boldsymbol{g})$. Its value at a parameter set $\boldsymbol{\theta}$ is:

$$\begin{aligned}\mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{DDSP}}(\mathbf{W}) &= \frac{1}{2}\|\boldsymbol{\Phi}(\tilde{\boldsymbol{x}}) - \boldsymbol{\Phi}(\boldsymbol{x})\|_2^2 \\ &= \frac{1}{2}\big\|(\boldsymbol{\Phi}\circ\boldsymbol{g}\circ\boldsymbol{f}_{\mathbf{W}}\circ\boldsymbol{g})(\boldsymbol{\theta}) - (\boldsymbol{\Phi}\circ\boldsymbol{g})(\boldsymbol{\theta})\big\|_2^2 \quad (1)\end{aligned}$$

by definition of $\tilde{\boldsymbol{x}}$ and $\boldsymbol{x}$. Using $\tilde{\boldsymbol{\theta}}$ as shorthand for $(\boldsymbol{f}_{\mathbf{W}}\circ\boldsymbol{g})(\boldsymbol{\theta})$, we conduct a first-order Taylor expansion of $(\boldsymbol{\Phi}\circ\boldsymbol{g})$ near $\boldsymbol{\theta}$. We obtain:

$$\boldsymbol{\Phi}(\tilde{\boldsymbol{x}}) = \boldsymbol{\Phi}(\boldsymbol{x}) + \boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}(\boldsymbol{\theta})\cdot(\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}) + O(\|\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}\|_2^2), \quad (2)$$

where the Jacobian matrix $\boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}(\boldsymbol{\theta})$ contains $P = \dim\boldsymbol{\Phi}(\boldsymbol{x})$ rows and $J = \dim\boldsymbol{\theta}$ columns. The manifold formed by differentiable map $(\boldsymbol{\Phi}\circ\boldsymbol{g})$ and the open set $\boldsymbol{\theta}\subset\mathbb{R}^J$ induces a Riemannian metric $\mathbf{M}$, i.e., an inner product on the tangent space at each point $\boldsymbol{\theta}$:

$$\mathbf{M}(\boldsymbol{\theta})_{j,j'} = \sum_{p=1}^{P}\left(\boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}(\boldsymbol{\theta})_{p,j}\right)\left(\boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}(\boldsymbol{\theta})_{p,j'}\right). \quad (3)$$

The real-valued square matrix $\mathbf{M}(\boldsymbol{\theta})$ defines a positive semidefinite kernel which, once plugged into Equation 2, serves to approximate $\mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{DDSP}}(\mathbf{W})$ in terms of a quadratic form over $(\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta})$:

$$\|\boldsymbol{\Phi}(\tilde{\boldsymbol{x}}) - \boldsymbol{\Phi}(\boldsymbol{x})\|_2^2 = \big\langle\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}\big|\mathbf{M}(\boldsymbol{\theta})\big|\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}\big\rangle + O\big(\|\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}\|_2^3\big). \quad (4)$$

The advantage of the approximation above is that the metric $\mathbf{M}$ may be computed over the training set once and for all. This is because Equation 3 is independent of the encoder $\boldsymbol{f}_{\mathbf{W}}$. Furthermore, since $\boldsymbol{\theta}$ is low-dimensional, we may store $\mathbf{M}(\boldsymbol{\theta})$ on RAM. From this perspective, we define the perceptual–neural–physical loss (PNP) associated to $(\boldsymbol{\Phi}, \boldsymbol{f}_{\mathbf{W}}, \boldsymbol{g})$ as the linearization of spectral loss at $\boldsymbol{\theta}$:

$$\begin{aligned}\mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{PNP}}(\mathbf{W}) &= \frac{1}{2}\big\langle(\boldsymbol{f}_{\mathbf{W}}\circ\boldsymbol{g})(\boldsymbol{\theta}) - \boldsymbol{\theta}\big|\mathbf{M}(\boldsymbol{\theta})\big|(\boldsymbol{f}_{\mathbf{W}}\circ\boldsymbol{g})(\boldsymbol{\theta}) - \boldsymbol{\theta}\big\rangle \\ &= \mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{DDSP}}(\mathbf{W}) + O\big(\|(\boldsymbol{f}_{\mathbf{W}}\circ\boldsymbol{g})(\boldsymbol{\theta}) - \boldsymbol{\theta}\|_2^3\big). \quad (5)\end{aligned}$$

According to the chain rule, the gradient of PNP loss at a given training pair $(\boldsymbol{x}_n, \boldsymbol{\theta}_n)$ with respect to some scalar weight $\mathbf{W}_i$ is:

$$\frac{\partial\mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{PNP}}}{\partial\mathbf{W}_i}(\boldsymbol{\theta}_n) = \Big\langle\boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x}_n) - \boldsymbol{\theta}_n\Big|\mathbf{M}(\boldsymbol{\theta}_n)\Big|\frac{\partial\boldsymbol{f}_{\mathbf{W}}}{\partial\mathbf{W}_i}(\boldsymbol{x}_n)\Big\rangle. \quad (6)$$

Observe that replacing $\mathbf{M}(\boldsymbol{\theta}_n)$ by the identity matrix in the equation above would give the gradient of *parameter loss* (P-loss); that is, the mean squared error between the predicted parameter $\tilde{\boldsymbol{\theta}}$ and the true parameter $\boldsymbol{\theta}$. Hence, we may regard PNP as a perceptually motivated extension of P-loss, in which parameter deviations are locally recombined and rescaled so as to linearly approximate a DDSP objective.

The matrix $\mathbf{M}(\boldsymbol{\theta})$ is constant in $\mathbf{W}$. Hence, its value may be cached across training epochs, and even across hyperparameter settings of the encoder. In comparison with P-loss, the only computational overhead of PNP is the bilinear form in Equation 6. However, this computation is performed in the parametric domain, i.e., in low dimension ($J = \dim\boldsymbol{\theta}$). Hence, its cost is negligible in front of the forward ($\boldsymbol{f}_{\mathbf{W}}$) and backward pass ($\partial\boldsymbol{f}_{\mathbf{W}}/\partial\mathbf{W}_i$) of DNN training.

2.2. Damped least squares

The principal components of the Jacobian $\boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}(\boldsymbol{\theta})$ are the eigenvectors of $\mathbf{M}(\boldsymbol{\theta})$. We denote them by $\boldsymbol{v}_j$ and the corresponding eigenvalues by σ_j^2: for each of them, we have $\mathbf{M}(\boldsymbol{\theta})\boldsymbol{v}_j = \sigma_j^2\boldsymbol{v}_j$. The $\boldsymbol{v}_j$'s form an orthonormal basis of $\mathbb{R}^J$, in which we can decompose the parameter deviation $(\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta})$. Recalling Equation 5, we obtain an alternative formula for PNP loss:

$$\mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{PNP}}(\mathbf{W}) = \frac{1}{2}\sum_{j=1}^{J}\sigma_j^2\left|\langle(\boldsymbol{f}_{\mathbf{W}}\circ\boldsymbol{g})(\boldsymbol{\theta}) - \boldsymbol{\theta}\big|\boldsymbol{v}_j\rangle\right|^2 \quad (7)$$

The eigenvalues σ_j^2 stretch and compress the error vector along their associated direction $\boldsymbol{v}_j$, analogous to the magnification and suppression of perceptually relevant and irrelevant parameter deviations. In practice however, when σ_j^2 cover drastic ranges or contain zeros, as presented below in Section 4.3, the error vector is subject to extreme distortion and potential instability due to numerical precision errors. These scenarios, commonly referred to as $\mathbf{M}$ being ill-conditioned, can lead to intractable learning objective $\mathcal{L}_{\theta}^{\mathrm{PNP}}$.

Reminiscent of the damping mechanism introduced in Levenberg-Marquardt algorithm when solving nonlinear optimization problems, we update Equation 5 as

$$\mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{PNP}}(\mathbf{W}) = \frac{1}{2}\big\langle\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}\big|\mathbf{M}(\boldsymbol{\theta}) + \lambda I\big|\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}\big\rangle \quad (8)$$

The damping term λI up-shifts all eigenvalues of $\mathbf{M}$ by a constant positive amount λ, thereby changing its condition number. At the limit of $\lambda\to 0$, $\mathcal{L}_{\theta}^{\mathrm{PNP}}$ reduces to a quadratic form which is asymptotically equivalent to spectral loss as P-loss approaches zero. At the limit of $\lambda\to\infty$, $\mathbf{M}$ is negligible in front of λI thus $\mathcal{L}_{\theta}^{\mathrm{PNP}}$ boils down to P-loss. Alternatively, Equation 8 may also be viewed as a L2 regularization with coefficient λ.

To further address potential convergence issues, λ may be scheduled or adaptively changed according to epoch validation loss. We adopt delayed gratification mechanism to decrease λ by a factor of 5 when epoch validation loss is going down, and fix λ otherwise.

3. APPLICATION TO DRUM SOUND MATCHING

3.1. *Perceptual:* Joint time–frequency scattering (JTFS)

The joint time–frequency scattering transform (JTFS) is a nonlinear convolutional operator which extracts spectrotemporal modulations in the constant-Q scalogram [9, 10]. Its kernels proceed from a separable product between two complex-valued wavelet filterbanks, defined over the time axis and over the log-frequency axis respectively. After convolution, we apply pointwise complex modulus and temporal averaging to each JTFS coefficient. These coefficients are known as scattering "paths" p. We apply a logarithmic transformation to the feature vector $\mathrm{JTFS}(\boldsymbol{x}_n)$ corresponding to each sound $\boldsymbol{x}_n$, yielding

$$\mathbf{S}_{n,p} = \boldsymbol{\Phi}(\boldsymbol{x}_n)_p = (\boldsymbol{\Phi}\circ\boldsymbol{g})(\boldsymbol{\theta}_n)_p = \log\left(1 + \frac{\mathrm{JTFS}(\boldsymbol{x}_n)_p}{\varepsilon}\right), \quad (9)$$

We set $\varepsilon = 10^{-3}$, which is the order of magnitude of the median value of JTFS across all examples $\boldsymbol{x}_n$ and paths p.

The multiresolution structure of JTFS is reminiscent of spectrotemporal receptive fields (STRF), and thus may serve as a biologically plausible predictor of neurophysiological responses in the primary auditory cortex [11]. At a higher level of music cognition, a recent study has shown that Euclidean distances in $\boldsymbol{\Phi}$ space predict auditory judgments of timbre similarity within a large vocabulary of instrumental playing techniques, as collected from a group of professional composers and non-expert music listeners [12].

We use the GPU implementation of [13] to compute JTFS with the same parameters as [12]: $Q_1 = 12$, $Q_2 = 1$, and $Q^{\mathrm{fr}} = 1$ filters per octave respectively. We set the temporal averaging to $T = 3$

seconds and the frequential averaging to $F = 2$ octaves; hence a total of $P = 20762$ paths. We refer to [14] for further details on the ability of $\nabla_{(\boldsymbol{\Phi} \circ \boldsymbol{g})}$ to extract "mesostructures" in nonstationary audio signals.

3.2. *Neural:* Deep convolutional network (convnet)

EfficientNet is a convolutional neural network architecture that balances the scaling of the depth, width and input resolution of consecutive convolutional blocks [15]. Achieving state-of-the-art performance on image classification with significantly less trainable parameters, its most light-weight version EfficientNet-B0 also succeeded in benchmarking audio classification tasks [16]. We adopt EfficientNet-B0 as our encoder $\boldsymbol{f}_{\mathbf{W}}$, resulting in 4M learnable parameters. We append a linear dense layer of $J = \dim \boldsymbol{\theta}$ neurons and a 1D batch normalization before tanh activation. The goal of batch normalization is to gaussianize the input, such that the activated output is capable of uniformly cover the normalized prediction range. The input to $\boldsymbol{f}_{\mathbf{W}}$ is the log-scaled CQT coefficients of each example, spanning 10 octaves with 12 filters per octave.

3.3. *Physical:* Functional transformation method (FTM)

We are interested in the perpendicular displacement $\mathbf{X}(t, u)$ on a rectangular drum face, which can be solved from the following partial differential equation defined in the Cartesian coordinate system $\boldsymbol{u} = (u_1, u_2)$.

$$\left(\frac{\partial^2 \mathbf{X}}{\partial t^2}(t, \boldsymbol{u}) - c^2 \nabla^2 \mathbf{X}(t, \boldsymbol{u})\right) + S^4\left(\nabla^4 \mathbf{X}(t, \boldsymbol{u})\right) + \frac{\partial}{\partial t}\left(d_1 \mathbf{X}(t, \boldsymbol{u}) + d_3 \nabla^2 \mathbf{X}(t, \boldsymbol{u})\right) = 0 \tag{10}$$

In addition to the standard traveling wave equation in the first above parenthesis, the fourth-order spatial and first-order time derivatives incorporate damping factors induced by stiffness, internal friction in the drum material and air friction in the external environment, rendering the solution a closer simulation to reality. Specifically, α, S, c, d_1, d_3 designate respectively the side length ratio, stiffness, traveling wave speed, frequency-independent damping and frequency-dependent damping of the drum. Even though real world drums are mostly circular, a rectangular drum model is equally capable of eliciting representative percussive sounds in real world scenarios. The circular drum model simply requires a conversion of Equation 10 into the Polar coordinate system. We bound the four sides of this l by $l\alpha$ rectangular drum at zero at all time. For simplicity, we simulate the excitation by setting the initial condition at $t_0 = 0$ to be $\mathbf{X}(t_0, \boldsymbol{u} = (0.4l, 0.4l\alpha)) = 0.03$ meters, and 0 otherwise.

We implement generator $\boldsymbol{g}$ as a PDE solver to this high-order damped wave equation, namely the functional transformation method (FTM) [17, 18]. FTM solves the PDE by transforming the equation into its Laplace and functional space domain, where an algebraic solution can be obtained. It then finds the time-space domain solution via inverse functional transforms, expressed in an infinite modal summation form

$$\boldsymbol{x}(t) = \mathbf{X}(t, \boldsymbol{u}) = \sum_{m \in \mathbb{N}^2} K_m(\boldsymbol{u}, t) \exp(\sigma_m t) \sin(\omega_m t) \tag{11}$$

The coefficients $K_m(\boldsymbol{u}, t)$, σ_m, ω_m are derived from the original PDE parameters in the following ways.

$$\omega_m^2 = (S^4 - \frac{d_3^2}{4})\Gamma_{m_1,m_2}^2 + (c^2 + \frac{d_1 d_3}{2})\Gamma_{m_1,m_2} - \frac{d_1^2}{4} \tag{12}$$

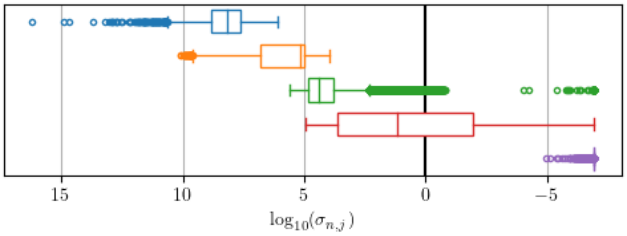

Fig. 2. Distributions of the five sorted eigenvalues of $\mathbf{M}(\boldsymbol{\theta}_n)$. For the sake of comparison between PNP and P-loss, the bold line indicates the eigenvalues of the identity matrix (see Equation 6).

$$\sigma_m = \frac{d_3}{2}\Gamma_{m_1,m_2} - \frac{d_1}{2} \tag{13}$$

$$K_m(\boldsymbol{u}, t) = y_u^m \delta(t) \sin(\frac{\pi m_1 u_1}{l}) \sin\left(\frac{\pi m_2 u_2}{l\alpha}\right) \tag{14}$$

where $\Gamma_{m_1,m_2} = \pi^2 m_1^2 / l^2 + \pi^2 m_2^2 / (l\alpha)^2$, and y_u^m is the m^{th} coefficient associated to the eigenfunction $\sin(\pi m \boldsymbol{u}/l)$ that decomposes $y_u(\boldsymbol{u})$.

Without losing connections to the acoustical manufacturing of the drum yet better relating $\boldsymbol{g}$'s input with perceptual dimensions, we reparametrize the PDE parameters $\{S, c, d_1, d_3, \alpha\}$ into $\theta = \{\log \omega_1, \tau_1, \log p, \log D, \alpha\}$, detailed in Section 3.4 of [19]. We prescribe sonically-plausible ranges for each parameter in θ, normalize them between -1 and 1, uniformly sample in the hyper-dimensional cube, and obtain a dataset of 100k percussive sounds sampled at 22050 HZ. The train/test/validation split is $8:1:1$.

In particular, fundamental frequency ω_1, duration τ_1 falls into ranges $[40, 1000]$ Hz and $[0.4, 3]$ seconds respectively. Inhomogeneous damping rate p, frequential dispersion D and aspect ratio α ranges are $[10^{-5}, 0.2]$, $[10^{-5}, 0.3]$, and $[10^{-5}, 1]$.

4. RESULTS

4.1. Baselines

We train f_W with 3 different losses - multi-scale spectral loss [20], parameter loss, and PNP loss. We use a batch size of 64 samples for spectral loss, and 256 samples for P-loss and PNP loss. The training proceeds for 70 epochs, where around 20% of the training set is seen at each epoch. We use Adam optimizer with learning rate 10^{-3}. Table 1 reports the training time per epoch on a single Tesla V100 16GB GPU.

4.2. Evaluation with JTFS-based spectral loss

We propose to use the L2 norm of JTFS coefficients error averaged over test set for evaluation. As a point of reference, we also include the average multi-scale spectral error, implemented as in Section 4.1. One of the key distinctions between Euclidean JTFS distance and multi-scale spectral error is the former's inclusion of spectro-temporal modulations information. Meanwhile unlike mean squared parameter error, both metrics reflect the perceptual closeness instead of parametric retrieval accuracy for each proposed model.

4.3. Discussion

Despite being the optimal quadratic approximation of spectral loss, it is nontrivial to apply the bare PNP loss form as Equation 5 in

Loss	$\mathbf{\Phi}$	Pitch	JTFS distance (avg. on test set)	MSS (avg. on test set)	Training time per epoch
P-loss	—	Known	$\mathbf{22.23} \pm 2.17$	$\mathbf{0.31} \pm 0.013$	49 minutes
$\mathcal{L}_\theta^{\text{DDSP}}$	$\mathbf{\Phi}_{\text{MSS}}$	Known	31.86 ± 0.332	0.335 ± 0.005	54 minutes
$\mathcal{L}_\theta^{\text{PNP}}$	$\mathbf{\Phi}_{\text{JTFS}}$	Known	23.58 ± 0.877	0.335 ± 0.005	49 minutes
$\mathcal{L}_\theta^{\text{DDSP}}$	$\mathbf{\Phi}_{\text{JTFS}}$	—	—	—	est., > 1 day
P-loss	—	Unknown	61.91 ± 6.26	1.02 ± 0.094	53 minutes
$\mathcal{L}_\theta^{\text{DDSP}}$	$\mathbf{\Phi}_{\text{MSS}}$	Unknown	138.95 ± 37.12	1.59 ± 0.307	59 minutes
$\mathcal{L}_\theta^{\text{PNP}}$	$\mathbf{\Phi}_{\text{JTFS}}$	Unknown	$\mathbf{61.21} \pm 1.207$	$\mathbf{0.97} \pm 0.019$	49 minutes

Table 1. Report of average JTFS distance and MSS metrics evaluated on test set. Six models are trained with two modalities: 1. the inclusion of pitch retrieval i.e. regressing $\theta = \{\tau, \log p, \log D, \alpha\}$ vs. $\theta = \{\log \omega_1, \tau, \log p, \log D, \alpha\}$, and 2. the choice of loss function: P-loss, MSS loss, or PNP loss with adaptive damping mechanism. The best performing models with known and unknown pitch are P-loss and PNP loss respectively. Training with MSS loss is more time consuming than training with P-loss or PNP loss. Training with differentiable JTFS loss is unrealistic in the interest of time.

experimental settings. On one hand, $\mathbf{\Phi} \circ \boldsymbol{g}$ potentially has undesirable property that exposes the Riemannian metric calculations to numerical precision errors. On the other hand, extreme deformation of the optimization landscape may lead to the same numerical instability facing stochastic gradient descent with spectral loss. We report on a few remedies that helped stabilize learning with PNP loss, and offer insights on future directions to take.

First and foremost, our preliminary experiments show that training PNP loss without damping $\lambda = 0$ subjects to convergence issues due to the high condition numbers in empirical $\mathbf{M}$s as illustrated in Section 2.2. Fig. 2 shows the sorted eigenvalue distribution of all $\mathbf{M}$s in test set, where $\mathbf{M}$s are rank-2,3 or 4 matrices with eigenvalues ranging from 0 to 10^{20}. This could be an implication that entries of $\boldsymbol{\theta}$ contain implicit linear dependencies in generator $\boldsymbol{g}$, or that local variations of certain $\boldsymbol{\theta}$ fail to linearize differences in the output of $\boldsymbol{g}$ or $\mathbf{\Phi} \circ \boldsymbol{g}$. As an example, the aspect ratio α influences the modal frequencies and decay rates via [19, Equations 12–13], where in fact $(1/\alpha + 1/\alpha^2)$ could be a better choice of variable that linearizes $\boldsymbol{g}$.

To address $\mathbf{M}$s' ill conditions we attempted at numerous damping mechanisms to update λ: namely, constant λ, scheduled λ decay, and adaptive λ decay. The intuition is to have $\mathcal{L}_\theta^{\text{PNP}}$ start in the parameter loss regime and move towards the spectral loss regime while training. The best performing model is achieved with adaptive λ decay (see Section 2.2). We propose to divide λ by a factor of 5 if the model breaks the best epoch validation loss record and keep it the same otherwise. In practice, we initialize λ to match the largest empirical $\sigma_j^2 \approx 10^{20}$, and then adaptively decay it to 3×10^{14} in 20 epochs. This indicates that $\boldsymbol{f}_{\mathbf{W}}$ is able to learn with damped PNP loss if λ is large enough to compensate for rank deficiency in $\mathbf{M}$.

The diagonal elements of $\mathbf{M}(\boldsymbol{\theta})$ can be regarded as both the applied weights' magnitudes and proxies for the perceptual importance of θ's accuracy. Inspecting the results of $\tau = \boldsymbol{\theta}[i_\tau]$ regression, we observe in Fig. 3 that in comparison with P-loss model, PNP model improves retrieval accuracy for sounds inducing larger perceptual difference to changes in τ (in blue), at the expense of lowered accuracy for the opposite (in yellow). This suggests a trade-off behavior aligned with PNP loss' weighting scheme.

We believe that more of PNP loss' mathematical potential can be exploited in the future, notably in cases where parameterization without domain-specific knowledge renders the failure of P-loss, and its use in hybrid optimization schemes. We plan to investigate the scalability of each loss function under reparameterizations, as well as other damping schemes and optimizers. The current update mechanism, originated from the Leverberg-Marquardt algorithm, aims to improve the conditioning of a matrix inversion problem in the

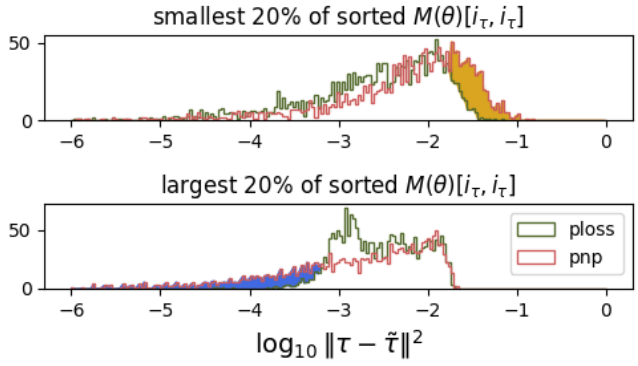

Fig. 3. Histogram of log squared τ estimation error for perceptually significant (largest 20% $\mathbf{M}(\boldsymbol{\theta})[i_\tau, i_\tau]$) and less significant (smallest 20% $\mathbf{M}(\boldsymbol{\theta})[i_\tau, i_\tau]$) sounds. The yellow and blue regions indicate the weighting-induced difference in retrieval accuracy.

Gauss-Newton algorithm. However when used jointly with stochastic gradient descent, each λ update may change the optimization landscape drastically. The resulting optimization behavior is thus not fully understood. We consider interfacing nonlinear least squares solver with SGD and forming a hybrid learning scheme in future work.

5. CONCLUSION

Knowledge on human auditory perception aiding data-driven approaches to machine listening tasks have been exemplified in a multitude of applications [21]. In this article we presented another case of this synergy named Perceptual-Neural-Physical (PNP) autoencoding, a bilinear form learning objective for sound matching task. In our application, PNP optimizes the retrieval of physical parameters from sounds in a perceptually-motivated metric space, enabled by differentiable implementations of physical model and computational proxy of neurophysiological construct of human auditory system.

We demonstrated PNP's mathematical relationship to spectral loss and parameter loss. Using this formulation, we motivated and established one way of interpolating between optimizing in parameter and spectral loss regimes. We presented damping mechanisms to facilitate its learning under ill-conditioned empirical settings and provided future plans for further exploiting its mathematical potential.

6. REFERENCES

[1] Andrew Horner, "Wavetable matching synthesis of dynamic instruments with genetic algorithms," *Journal of the Audio Engineering Society*, vol. 43, no. 11, pp. 916–931, 1995.

[2] Jordie Shier, Kirk McNally, George Tzanetakis, and Ky Grace Brooks, "Manifold learning methods for visualization and browsing of drum machine samples," *Journal of the Audio Engineering Society*, vol. 69, no. 1/2, pp. 40–53, 2021.

[3] Philippe Esling, Naotake Masuda, Adrien Bardet, Romeo Despres, Axel Chemla, et al., "Universal audio synthesizer control with normalizing flows," in *Proceedings of the International Conference on Digital Audio Effects (DAFX)*, 2019.

[4] Leonardo Gabrielli, Stefano Tomassetti, Carlo Zinato, and Francesco Piazza, "End-to-end learning for physics-based acoustic modeling," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 160–170, 2018.

[5] Naotake Masuda and Daisuke Saito, "Synthesizer sound matching with differentiable DSP," in *Proceedings of the International Society on Music Information Retrieval (ISMIR) Conference*, 2021, pp. 428–434.

[6] Martin Roth and Matthew Yee-king, "A comparison of parametric optimization techniques for musical instrument tone matching," *Journal of the Audio Engineering Society*, May 2011.

[7] Matthew Yee-King, Leon Fedden, and Mark d'Inverno, "Automatic programming of vst sound synthesizers using deep networks and other techniques," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, pp. 150–159, 2018.

[8] Jesse Engel, Lamtharn (Hanoi) Hantrakul, Chenjie Gu, and Adam Roberts, "DDSP: Differentiable Digital Signal Processing," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[9] Joakim Andén, Vincent Lostanlen, and Stéphane Mallat, "Joint time–frequency scattering," *IEEE Transactions on Signal Processing*, vol. 67, no. 14, pp. 3704–3718, 2019.

[10] Mathieu Andreux, Tomás Angles, Georgios Exarchakis, Roberto Leonarduzzi, Gaspar Rochette, Louis Thiry, John Zarka, Stéphane Mallat, Joakim Andén, Eugene Belilovsky, Joan Bruna, Vincent Lostanlen, Muawiz Chaudhary, Matthew J. Hirn, Edouard Oyallon, Sixin Zhang, Carmine Cella, and Michael Eickenberg, "Kymatio: Scattering transforms in Python," *Journal of Machine Learning Research*, vol. 21, no. 60, pp. 1–6, 2020.

[11] Taishih Chi, Powen Ru, and Shihab A Shamma, "Multiresolution spectrotemporal analysis of complex sounds," *The Journal of the Acoustical Society of America*, vol. 118, no. 2, pp. 887–906, 2005.

[12] Vincent Lostanlen, Christian El-Hajj, Mathias Rossignol, Grégoire Lafay, Joakim Andén, and Mathieu Lagrange, "Time–frequency scattering accurately models auditory similarities between instrumental playing techniques," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2021, no. 1, pp. 1–21, 2021.

[13] John Muradeli, Cyrus Vahidi, Changhong Wang, Han Han, Vincent Lostanlen, Mathieu Lagrange, and George Fazekas, "Differentiable time-frequency scattering in kymatio," in *Proceedings of the International Conference on Digital Audio Effects (DAFX)*, 2022.

[14] Cyrus Vahidi, Han Han, Changhong Wang, Mathieu Lagrange, György Fazekas, and Vincent Lostanlen, "Mesostructures: Beyond spectrogram loss in differentiable time-frequency analysis," *arXiv preprint arXiv:2301.10183*, 2023.

[15] Mingxing Tan and Quoc Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the International conference on Machine Learning (ICML)*. PMLR, 2019, pp. 6105–6114.

[16] Neil Zeghidour, Olivier Teboul, Félix de Chaumont Quitry, and Marco Tagliasacchi, "LEAF: A learnable frontend for audio classification," *ICLR*, 2021.

[17] L. Trautmann and Rudolf Rabenstein, *Digital Sound Synthesis by Physical Modeling Using the Functional Transformation Method*, Springer, 2003.

[18] M. Schäfer, M. Werner, and R. Rabenstein, "Physical modeling in sound synthesis: Vibrating plates," in *Proc. 26th International Congress on Sound and Vibration (ICSV26)*, Montreal, Canada, Jul. 2019, pp. 1–8.

[19] Han Han and Vincent Lostanlen, "wav2shape: Hearing the Shape of a Drum Machine," in *Proceedings of Forum Acusticum*, 2020, pp. 647–654.

[20] Christian J. Steinmetz and Joshua D. Reiss, "auraloss: Audio focused loss functions in PyTorch," in *Digital Music Research Network One-day Workshop (DMRN+15)*, 2020.

[21] Laurie M. Heller, Benjamin Elizalde, Bhiksha Raj, and Soham Deshmukh, "Synergy between human and machine approaches to sound/scene recognition and processing: An overview of ICASSP special session," in *Proceedings of the IEEE International Conference on Audio, Speech, and Signal Processing*. 2023, IEEE.