KLANN: Linearising long-term dynamics in nonlinear audio effects using Koopman networks

Ville Huhtala, Lauri Juvela *Member, IEEE*, Sebastian J. Schlecht, *Senior Member, IEEE*,

***Abstract*—In recent years, neural network-based black-box modeling of nonlinear audio effects has improved considerably. Present convolutional and recurrent models can model audio effects with long-term dynamics, but the models require many parameters, thus increasing the processing time. In this paper, we propose KLANN, a Koopman-Linearised Audio Neural Network structure that lifts a one-dimensional signal (mono audio) into a high-dimensional approximately linear state-space representation with nonlinear mapping, and then uses differentiable biquad filters to predict linearly within the lifted state-space. Results show that the proposed models match the high performance of the state-of-the-art neural models while having a more compact architecture, reducing the number of parameters by tenfold, and having interpretable components.**

I. Introduction

ANALOG audio effects are used for their distinct sound due to nonlinearities in the circuitry. Thus, as music production is digitalised, there is a need for emulations of audio circuitry, which is referred to as virtual analog (VA) modeling in the literature. Black-box VA methods use only the input and output measurements of an audio circuitry under study to infer the system's behavior. Classic methods, such as the Volterra series [1], often lead to complex models whose parameters are difficult to estimate and interpret.

In recent years, several black-box models for modeling audio effects using deep neural networks (DNNs) have been proposed. Previous solutions include convolution networks (CNNs) [2, 3] as well as recurrent neural networks (RNNs) using either a long short-term Memory (LSTM) or a gated recurrent unit (GRU) [4]. Both CNNs and RNNs can operate in real-time [5] but at a relatively high computation cost. Approaches to make CNNs more expressive and lightweight include rapidly growing the dilation factors, increasing the network receptive field [6], and using temporal feature-wise linear modulation [7, 8] to modulate the intermediate features of the CNN [9]. RNNs also remain popular due to their relatively low cost, connection to state-space models [10, 11, 12], and applicability to external control inputs [13].

While most of the recent work has focused on audio effects with relatively short-term dynamics, modeling long-term dynamics remains challenging. Examples of such challenging systems include fuzz pedals [4] and compressors [14]. Recent efforts have made progress towards modeling long-term dependencies [9], but the models are still relatively expensive and hard to interpret.

The Koopman operator can be used to lift a nonlinear problem into an infinite-dimensional space where the problem becomes linear [15]. Koopman networks approximately linearise nonlinear systems by approximating the Koopman operator and can be used to predict the evolution of nonlinear differential equations [16]. The model proposed works by first approximately linearising a nonlinear dynamical system with nonlinear mapping, then solving the approximately linear dynamical system, and finally, nonlinearly mapping the system back to the original state. In audio effect modeling, Koopman networks provide an appealing framework to build on. Specifically, we can lift a one-dimensional audio signal to a high-dimensional approximately linear state-space representation and use classic audio DSP filter structures to model the dynamics on each lifted dimension, similarly to Koopman-based nonlinear system identification [17].

Differentiable biquad filters (second order IIR filters) [18, 19] are a good candidate for implementing the linear predictions between the nonlinear mappings, as they have multiple advantageous properties. First, they are highly expressive in relation to their parameter count and can produce arbitrarily long responses. Second, they are interpretable with common filter analysis techniques and their stability properties are well understood. Third, they are suitable for both frequency domain filtering for fast parallel training on GPUs [20, 21], and fast sequential inference on CPUs [12]. Differentiable biquad filters have been used to model distortion pedals using a Wiener-Hammerstein structure [18] and a series of differentiable parametric equalizers with nonlinearity in between [22] but did not yet match the performance of DNN baselines.

This paper presents KLANN, a Koopman-Linearised Audio Neural Network structure for black-box modeling audio effects. The method comprises 1) a memoryless neural network that lifts the data into a high-dimensional space, 2) a set of differentiable biquad filters that model the approximately linearised dynamics, and 3) another memoryless network that maps the high-dimensional solution back to the audio space. The proposed method is evaluated on two dynamic range compressors and a fuzz distortion effect to test its performance on long-term dynamics. Furthermore, we propose an end-to-end two-stage training scheme where the first stage optimises the time domain loss, and the second stage optimises both the time domain and frequency domain losses.

The remainder of this paper is structured as follows. Section II presents the KLANN structure and our two network variants. In Section III, our models are evaluated and compared against state-of-the-art convolutional and recurrent black-box

Manuscript submitted 12.12.2023.

The authors are with the Acoustics Lab, Department of Information and Communications Engineering, Aalto University, FI-02150 Espoo, Finland (e-mail: ville.huhtala@aalto.fi; lauri.juvela@aalto.fi; sebastian.schlecht@aalto.fi). Sebastian J. Schlecht is also with the Media Lab, Department of Art and Media, Aalto University, FI-02150 Espoo, Finland.

models. The results show that the proposed models match the high performance of DNN baselines while using fewer parameters and having interpretable components. Conclusions and future work are discussed in section IV.

II. Method

Given input and output signals x and y of an audio effect $f(x) = y$ with fixed control parameters, we want to find a function $\hat{f}(x)$ that produces a signal $\hat{y}$ that is perceptually indistinguishable from the real signal y. We propose two DNNs for $\hat{f}(x)$ both consisting of the KLANN structure. In the following, we explain the stages and overall structure.

A. Gated linear unit multilayer perceptron

The nonlinear mappings are gated linear unit multilayer perceptrons (GLU MLP) that consist of varying-sized fully connected (FC) layers using the gated linear unit (GLU) [23] nonlinearity. The input for an FC layer is a signal $X \in \mathbb{R}^{n \times m_{\text{in}}}$ with n samples and m_{in} hidden layer size. Thus, a GLU layer is defined as follows:

$$\text{GLU}(X) = (XW_{\text{lin}} + b_{\text{lin}}) \odot \sigma(XW_{\text{gate}} + b_{\text{gate}}), \quad (1)$$

where W_{lin} and $W_{\text{gate}} \in \mathbb{R}^{m_{\text{in}} \times m_{\text{out}}}$ are the weight matrices, m_{out} is the next hidden layer size, b_{lin} and $b_{\text{gate}} \in \mathbb{R}^{m_{\text{out}}}$ are the bias vectors, σ is the logistic sigmoid function, and $\odot$ is the element-wise product between matrices. The GLU layers increase in size in the first GLU MLP and decrease in the second. The final layer consists of a linear mapping.

B. Differentiable biquad filter

Digital state variable filters (DSVF) [24] are used as the biquad filter type as they can learn any biquad with added interpretability, faster convergence, and better performance when compared to a regular biquad filter [18]. DSVFs are a linear combination of a lowpass, bandpass, and highpass biquad filter with weights m_{LP}, m_{BP}, and m_{HP} controlling the mixing ratio between them, respectively. DSVFs share a cutoff c and a resonance R between its three filter types. The transfer function is defined as follows [24]:

$$H(z) = \frac{m_{\text{LP}} b_{\text{LP}}(z) + m_{\text{BP}} b_{\text{BP}}(z) + m_{\text{HP}} b_{\text{HP}}(z)}{1 + c^2 + 2Rc + (2g^2 - 2)z^{-1} + (1 + c^2 - 2Rc)z^{-2}}, \quad (2)$$

where $b_{\text{LP}}(z) = c^2 + 2c^2 z^{-1} + c^2 z^{-2}$ is the lowpass, $b_{\text{BP}}(z) = c - cz^{-2}$ is the bandpass, and $b_{\text{HP}}(z) = 1 - 2z^{-1} + z^{-2}$ is the highpass filter.

Following previous research [19], the cutoff c is restricted between frequencies $0 < c < \pi$ with the activation $\tan(\pi\sigma(x)/2)$ and a softplus activation is applied to the resonance R to satisfy the stability condition $R > 0$. The cutoff and resonance are initialised as 0 and the weights as 1.

Filtering is performed in frequency domain [22, 21] to speed up the training process, and the overlap-add method can also be used but at the cost of accuracy [20, 19]. Frequency domain filtering is carried out individually for each biquad. First, a biquad filter is frequency-sampled by using the discrete Fourier transform (DFT) on the numerator, and the denominator of its transfer function given in Eq. (2) [25]. This yields an FIR filter that is a truncated version of the biquad filter response in the time domain. Next, the DFT of the input signal for a biquad is taken, and the signal is multiplied with the FIR filter. Lastly, the inverse discrete Fourier transform (IDFT) returns the filtered signal to the time domain. If the overlap-add method is used, the overlapping segments are summed after the IDFT. The DFTs and IDFTs are zero-padded to a length of a power of two to employ an efficient fast Fourier transform (FFT) given by $2^{\lceil \log_2(2N-1) \rceil}$ [25], where N represents the length of the input signal in samples. N also dictates the length of the truncated biquad filter. Thus, N should be picked high enough to minimise time-aliasing [19]. Inference is done recursively in the time domain.

C. Parallel and parallel-series structures

Block diagrams of two networks using the KLANN structure are shown in Fig. 1. Both networks use two GLU MLPs to nonlinearly lift the input audio into a K-dimensional state-space and to restore the outputs of the lifted state-space into the original state, respectively. K corresponds to the number of biquads. The parallel KLANN shown in Fig. 1a has only biquad filters in the lifted state-space, making it fully linear, whereas the parallel-series KLANN shown in Fig. 1b does not assume the state-space to be fully linear as each biquad is connected to the next one using a single FC layer with the GLU nonlinearity followed by a linear mapping. Thus, the parallel-series KLANN can learn more complex lifted state-spaces compared to the parallel KLANN.

Both networks are interpretable as the biquad filters are the only part with memory. However, some interpretability is lost in the parallel-series KLANN as all nonlinearities do not occur in the GLU MLPs. Nevertheless, the biquad filters and the nonlinear mappings can be analyzed to determine the learned time dependencies and nonlinearities, respectively. Reference implementation is provided at the accompanying website [1].

D. Loss function

A combination of time and frequency domain losses are used for training. The mean squared error (MSE) is used as the time domain loss, and the multi-resolution short-time Fourier transform (MR STFT) [26, 27] is used as the frequency domain loss. MR STFT is defined as follows:

$$\text{L}_{\text{MR}} = \frac{1}{M} \sum_{m=1}^{M} \left[\text{L}_{\text{sc}}(\hat{y}, y) + \text{L}_{\text{sm}}(\hat{y}, y) \right], \quad (3)$$

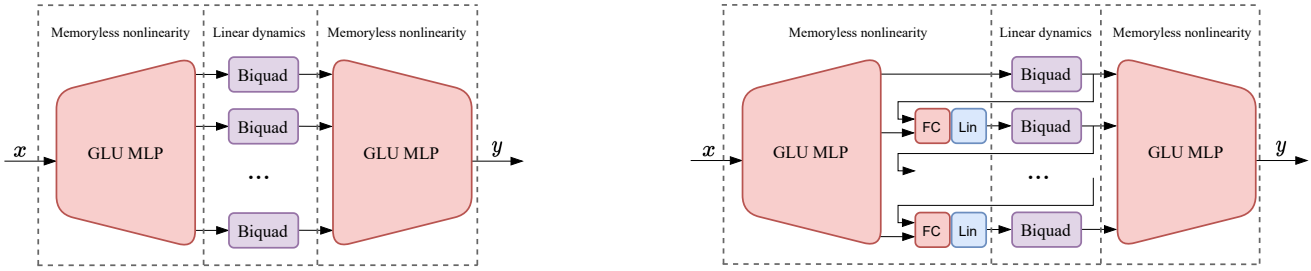
where M is the resolution of an STFT, and L_{sc} and L_{sm} denote the spectral convergence and the spectral log-magnitude losses, respectively. These are defined as [28]:

$$\text{L}_{\text{sc}}(\hat{y}, y) = \frac{\| |\text{STFT}(y)| - |\text{STFT}(\hat{y})| \|_F}{\| |\text{STFT}(y)| \|_F}, \quad (4)$$

$$\text{L}_{\text{sm}}(\hat{y}, y) = \frac{1}{P} \| \log |\text{STFT}(y)| - \log |\text{STFT}(\hat{y})| \|_1, \quad (5)$$

where $\|\cdot\|_F$ and $\|\cdot\|_1$ denote the Frobenius and L_1 norms, respectively, and P is the number of elements in an STFT magnitude.

[1] https://github.com/ville14/KLANN



(a) Parallel KLANN. (b) Parallel-series KLANN.

Fig. 1: Block diagrams of the two proposed KLANN model variants. The KLANN model is a sequence of memoryless nonlinear lifting, multiple linear filters, and a nonlinear combination. The inputs to the FC layer in the parallel-series KLANN are concatenated.

E. Training

The model parameters are updated in shuffled mini-batches. For each mini-batch, the last 1024 samples are used to calculate the loss to allow biquad filters time to stabilise. The audio effect under study determines the stabilisation time. The exact time is not required, but it has to be longer than the time-dependency of the audio effect under study.

The training process is divided into two stages. In the first stage, a model is trained with the MSE, and in the second stage, the model is fine-tuned using the sum of the MSE and the weighted MR STFT with the weight set to 0.001. The MR STFT has different window lengths, hop sizes, and DFT sizes. Window lengths 600, 240, and 100, hop sizes 120, 50, and 25, and DFT sizes 1024, 512, and 256 are used. Adam [29] is used with the initial learning rate set to 0.001, and the batch size is set to 50 on both stages. 2000 and 1000 epochs are allocated for the first stage and the second stage, respectively.

F. Dataset

All models are trained on three audio effects: a digital compressor, an analog compressor, and a digital fuzz pedal called the MCompressor[2], the LA-2A[3], and the Face Bender[4], respectively. The dataset is from [9], which is constructed using the SignalTrain dataset [14] for modeling the LA-2A and recordings from the IDMT-SMT-Guitar dataset [30] are used for modeling the digital audio effects. All audio is sampled at 44.1 kHz. The audio is divided into mini-batches using a sliding window with a hop size of 1024 samples. This way, the loss function is calculated utilising all possible samples. If the overlap-add filtering method is used, the mini-batch length is a multiple of N. Otherwise, N sets the length of the mini-batches. In this work, we use a single long FFT for filtering instead of over-lap add, since the biquad filters remain constant in any given minibatch.

The audio effects are trained with set configurations. The MCompressor is trained with 1 ms attack and 1000 ms release times. The LA-2A dataset has fixed attack and release times with an average attack time of 10 ms and a release time of about 60 ms for 50% of the release, and anywhere from 1 to 15 seconds for the rest. The time-constant of the Face Bender is unknown. Therefore, the MCompressor is trained using N=65536 samples, and both the LA-2A and the Face Bender are trained using N=32768 samples. This allows the biquad filters about 1.46 s and 0.74 s to stabilise, respectively.

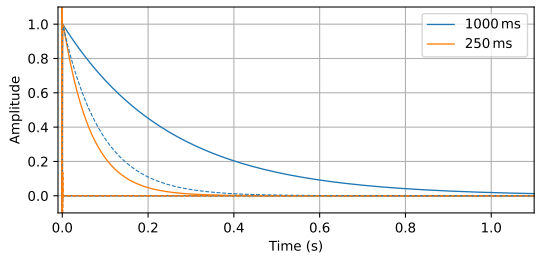


Fig. 2: Learned normalised IRs of a small parallel model for the MCompressor with 1000 ms and 250 ms release times. The first few samples are ignored before normalisation. Most responses decay within a few milliseconds.

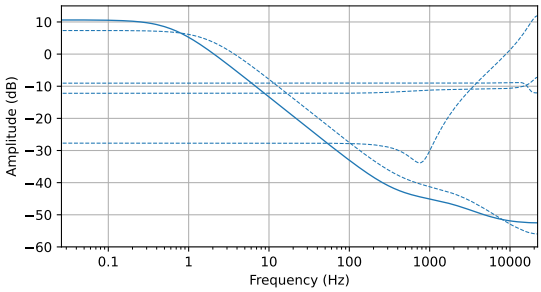


Fig. 3: Learned corresponding magnitude responses of the small parallel KLANN model for the MCompressor with 1000 ms release time shown in Fig. 2.

III. Evaluation

The parallel KLANN and the parallel-series KLANN are evaluated with two configurations: small and large. The small configuration has hidden layer sizes 3, 4, and 5 for the first GLU MLP and 5 biquad filters. The large configuration has hidden layer sizes 5, 10, and 15 for the first GLU MLP and 15

[2]https://www.meldaproduction.com/MCompressor
[3]https://www.uaudio.com/blog/la-2a-collection-tips-tricks/
[4]http://distorqueaudio.com/plugins/face-bender.html

TABLE I: Model sizes and training results. The best results are highlighted.

				Face Bender		MCompressor		LA-2A	
Model	**Layer sizes**	**#biquads**	**Params.**	**ESR_{dB} ↓**	**MR STFT ↓**	**ESR_{dB} ↓**	**MR STFT ↓**	**ESR_{dB} ↓**	**MR STFT ↓**
Wiener-Hammerstein	20, 20, 20	2	911	-20.836	0.920	-11.476	0.686	-9.003	1.073
small parallel	3 ,4, 5 - 5 ,4, 3	5	291	-21.768	0.620	-26.421	0.331	-14.424	0.661
large parallel	5, 10, 15 - 15, 10, 5	15	1701	-26.792	0.441	-29.228	0.314	-15.946	0.600
small parallel-series	3, 4, 5 - 5, 4, 3	5	435	-31.511	0.411	-30.808	0.307	-14.740	0.665
large parallel-series	5, 10, 15 - 15, 10, 5	15	2205	**-45.773**	**0.240**	**-36.192**	**0.282**	-16.869	0.582
GCNTF-1	-	-	38.9k	-34.931	0.308	-33.488	0.302	-19.433	0.539
GCNTF-3	-	-	71.1k	-41.125	0.264	-32.339	0.304	**-20.183**	**0.512**
GCNTF-250	-	-	74.3k	-34.8035	0.348	-30.254	0.311	-18.3785	0.586
GCNTF-2500	-	-	48.2k	-34.515	0.327	-35.632	0.293	-19.621	0.550
LSTM-32	32	-	4513	-39.497	0.374	-30.938	0.309	-15.501	0.669
LSTM-96	96	-	38.1k	-39.880	0.257	-30.030	0.317	-15.488	0.655

TABLE II: Impact of the training process. Single-stage training uses the sum of the MSE and the MR STFT, and two-stage training uses first the MSE and then the sum of the MSE and the MR STFT. The MR STFT is weighted by 0.001.

			Face Bender		MCompressor		LA-2A	
Model	**Loss function**	**Epochs**	**ESR_{dB} ↓**	**MR STFT ↓**	**ESR_{dB} ↓**	**MR STFT ↓**	**ESR_{dB} ↓**	**MR STFT ↓**
small parallel-series	single-stage	3000	-22.083	0.525	-29.323	0.311	-14.588	0.652
	two-stage	2000, 1000	-31.511	0.411	-30.808	0.307	-14.740	0.665
GCNTF-3	single-stage	3000	-41.076	0.252	-32.415	0.297	-17.572	0.513
	two-stage	2000, 1000	-41.125	0.264	-32.339	0.304	-20.183	0.512
LSTM-32	single-stage	3000	-41.357	0.260	-30.411	0.309	-12.846	0.680
	two-stage	2000, 1000	-39.497	0.374	-30.938	0.309	-15.501	0.669

biquad filters. The second GLU MLPs have the hidden layer sizes reversed. The GLU layer in the parallel-series KLANN has a hidden layer size of 5.

We compare state-of-the-art convolutional and recurrent networks called the GCNTF [9] and LSTM [4], respectively. We use the implementation provided in [9]. Both networks are trained using the two-stage training scheme similar to our models. The batch size for the GCNTF and LSTM models are set to 8 and 50, respectively. We also compare a differentiable Wiener-Hammerstein model with two DSVF filters proposed in [18].

The error-to-signal ratio in decibels (ESR_{dB}) and the MR STFT values are used to evaluate the models. The ESR_{dB} is calculated as follows:

$$\mathrm{ESR}_{\mathrm{dB}} = 10 \log_{10} \left| \frac{\sum_{n=0}^{N-1} |y_n - \hat{y}_n|^2}{\sum_{n=0}^{N-1} |y_n|^2} \right|, \tag{6}$$

where y and $\hat{y}$ are the target signal and the output signal of the model, respectively.

Results of the training are shown in Table I, and audio examples are provided in the accompanying website[5]. The large parallel-series model performs the best between our models. It achieves the smallest ESR_{dB} and MR STFT values in modeling the Face Bender and it models the MCompressor with similar accuracy to the best GCNTF model. However, the GCNTF-3 model achieves the best accuracy in modeling the LA-2A. Nevertheless, we found it hard to distinguish audible differences between our model and the GCNTF model. Even the small parallel model produced convincing results, although it lacks modeling of some high-frequency content in the Face Bender.

Table II shows the impact of the training process for the small parallel-series, GCNTF-3, and the LSTM-32 models. Two-stage training is beneficial when the MSE value, which is good at modeling lower frequencies, is better optimized in the first stage with no MR STFT loss present as in the case with the Face Bender for the small parallel-series model. The LA-2A for the GCNTF-3 and LSTM-32 models also see improvements. However, the Face Bender was modeled better with single-stage training for the LSTM-32 model.

Fig. 2 shows two learned impuse responses (IRs) of a small parallel model trained on the MCompressor with 1000 ms and 250 ms release times. Fig. 3 shows the corresponding magnitude responses of the 1000 ms release time case. An exponentially decaying IR that matches the actual release time was learned for both cases. These are plotted as solid lines and the other learned biquads are plotted as dashed lines. The parallel-series model also learns similar IRs.

IV. Conclusion

This work presents a Koopman-based neural network structure for black-box modeling audio effects. Two networks were proposed, called the parallel KLANN and the parallel-series KLANN. Our models use significantly fewer parameters than the state-of-the-art DNN models while still producing perceptually convincing results. Conditioning the network on variable control positions remains as future work.

[5]https://ville14.github.io/KLANN-examples/

REFERENCES

[1] T. Hélie, "Volterra series and state transformation for real-time simulations of audio circuits including saturations: Application to the Moog ladder filter," *IEEE transactions on audio, speech, and language processing*, vol. 18, no. 4, pp. 747–759, 2009.

[2] E.-P. Damskägg, L. Juvela, E. Thuillier, and V. Välimäki, "Deep learning for tube amplifier emulation," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 471–475.

[3] M. A. M. Ramírez and J. D. Reiss, "Modeling nonlinear audio effects with end-to-end deep neural networks," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 171–175.

[4] A. Wright, E.-P. Damskägg, V. Välimäki *et al.*, "Real-time black-box modelling with recurrent neural networks," in *22nd international conference on digital audio effects (DAFx-19)*, 2019, pp. 1–8.

[5] A. Wright, E.-P. Damskägg, L. Juvela, and V. Välimäki, "Real-time guitar amplifier emulation with deep learning," *Applied Sciences*, vol. 10, no. 3, p. 766, 2020.

[6] C. J. Steinmetz and J. D. Reiss, "Efficient neural networks for real-time modeling of analog dynamic range compression," in *Audio Engineering Society Convention 152*. Audio Engineering Society, 2022.

[7] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "FiLM: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[8] S. Birnbaum, V. Kuleshov, Z. Enam, P. W. W. Koh, and S. Ermon, "Temporal FiLM: Capturing long-range sequence dependencies with feature-wise modulations." *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[9] M. Comunità, C. J. Steinmetz, H. Phan, and J. D. Reiss, "Modelling black-box audio effects with time-varying feature modulation," in *Proc. ICASSP*. IEEE, 2023, pp. 1–5.

[10] J. D. Parker, F. Esqueda, and A. Bergner, "Modelling of nonlinear state-space systems using a deep neural network," in *Proc. DAFx*, 2019.

[11] A. Peussa, E.-P. Damskägg, T. Sherson, S. Mimilakis, L. Juvela, A. Gotsopoulos, and V. Välimäki, "Exposure bias and state matching in recurrent neural network virtual analog models," in *Proc. Int. Conference on Digital Audio Effects (DAFx)*, 2021, pp. 284–291.

[12] A. Gu, K. Goel, and C. Re, "Efficiently modeling long sequences with structured state spaces," in *Proc. ICLR*, 2022.

[13] L. Juvela, E.-P. Damskägg, A. Peussa, J. Mäkinen, T. Sherson, S. I. Mimilakis, K. Rauhanen, and A. Gotsopoulos, "End-to-end amp modeling: from data to controllable guitar amplifier models," in *Proc. ICASSP*, 2023, pp. 1–5.

[14] S. H. Hawley, B. Colburn, and S. I. Mimilakis, "SignalTrain: Profiling audio compressors with deep neural networks," *arXiv preprint arXiv:1905.11928*, 2019.

[15] B. O. Koopman, "Hamiltonian systems and transformation in Hilbert space," *Proceedings of the National Academy of Sciences*, vol. 17, no. 5, pp. 315–318, 1931.

[16] B. Lusch, J. N. Kutz, and S. L. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics," *Nature communications*, vol. 9, no. 1, p. 4950, 2018.

[17] A. Mauroy and J. Goncalves, "Koopman-based lifting techniques for nonlinear systems identification," *IEEE Transactions on Automatic Control*, vol. 65, no. 6, pp. 2550–2565, 2020.

[18] B. Kuznetsov, J. D. Parker, and F. Esqueda, "Differentiable IIR filters for machine learning applications," in *Proc. Int. Conf. Digital Audio Effects (eDAFx-20)*, 2020, pp. 297–303.

[19] S. Lee, H.-S. Choi, and K. Lee, "Differentiable artificial reverberation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 2541–2556, 2022.

[20] L. Juvela, B. Bollepalli, J. Yamagishi, and P. Alku, "GELP: GAN-excited linear prediction for speech synthesis from mel-spectrogram," in *Proc. Interspeech*, 2019, pp. 694–698.

[21] S. Nercessian, "Neural parametric equalizer matching using differentiable biquads," in *Proc. Int. Conf. Digital Audio Effects (eDAFx-20)*, 2020, pp. 265–272.

[22] S. Nercessian, A. Sarroff, and K. J. Werner, "Lightweight and interpretable neural modeling of an audio distortion effect using hyperconditioned differentiable biquads," in *Proc. ICASSP*, 2021, pp. 890–894.

[23] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *International conference on machine learning*. PMLR, 2017, pp. 933–941.

[24] A. Wishnick, "Time-varying filters for musical applications," in *Proc. 17 Int. Conf. Digital Audio Effects (DAFx-14)*, Erlangen, Germany, September 2014, pp. 69–76.

[25] C. J. Steinmetz, N. J. Bryan, and J. D. Reiss, "Style transfer of audio effects with differentiable signal processing," *Journal of the Audio Engineering Society*, vol. 70, no. 9, pp. 708–721, 2022.

[26] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *Proc. ICASSP*. IEEE, 2020, pp. 6199–6203.

[27] C. J. Steinmetz and J. D. Reiss, "auraloss: Audio focused loss functions in pytorch," in *Digital music research network one-day workshop (DMRN+ 15)*, 2020.

[28] S. Ö. Arık, H. Jun, and G. Diamos, "Fast spectrogram inversion using multi-head convolutional neural networks," *IEEE Signal Processing Letters*, vol. 26, no. 1, pp. 94–98, 2018.

[29] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015.

[30] C. Kehling, J. Abeßer, C. Dittmar, and G. Schuller, "Automatic tablature transcription of electric guitar recordings by estimation of score-and instrument-related parameters." in *DAFx*, 2014, pp. 219–226.