MODULATION EXTRACTION FOR LFO-DRIVEN AUDIO EFFECTS

Christopher Mitcheltree *Christian J. Steinmetz* *Marco Comunità* *Joshua D. Reiss*

Centre for Digital Music, Queen Mary University of London, UK
{c.mitcheltree, c.j.steinmetz, m.comunita, joshua.reiss}@qmul.ac.uk

ABSTRACT

Low frequency oscillator (LFO) driven audio effects such as phaser, flanger, and chorus, modify an input signal using time-varying filters and delays, resulting in characteristic sweeping or widening effects. It has been shown that these effects can be modeled using neural networks when conditioned with the ground truth LFO signal. However, in most cases, the LFO signal is not accessible and measurement from the audio signal is nontrivial, hindering the modeling process. To address this, we propose a framework capable of extracting arbitrary LFO signals from processed audio across multiple digital audio effects, parameter settings, and instrument configurations. Since our system imposes no restrictions on the LFO signal shape, we demonstrate its ability to extract quasiperiodic, combined, and distorted modulation signals that are relevant to effect modeling. Furthermore, we show how coupling the extraction model with a simple processing network enables training of end-to-end black-box models of unseen analog or digital LFO-driven audio effects using only dry and wet audio pairs, overcoming the need to access the audio effect or internal LFO signal. We make our code available and provide the trained audio effect models in a real-time VST plugin[1].

1. INTRODUCTION

In music composition, production, and engineering, audio effects play a key role in altering the sound toward the desired result. Modulation effects such as phaser, flanger, and chorus, are part of a broad family of audio processors based on using a modulation signal to modify the spectrum, loudness, or spatial characteristic of the input audio. The typical modulation signals adopted are periodic (e.g., sinusoidal, sawtooth, triangular) with a frequency below the audible range (20 Hz) and are therefore called low frequency oscillators (LFO). Since oscillators are used to continuously vary the internal parameters of these effects, the exact shape, frequency, and phase of the LFO signal plays a crucial role, affecting the overall timbre and temporal behavior. This is especially evident in analog circuits where imperfections and nonlinearities may cause distortion and quasi-periodicity of the oscillation.

Digital emulation of audio effects is an area of active research [1–3], and many methods have been developed to analyze and emulate effect units. Depending on the degree of prior knowledge and reliance on measurement data, these can be divided into white- [4–7], gray- [8–13], or black-box [14–17] approaches. Most prior work on modulation effects modeling uses complex and time-consuming white-box approaches, obtaining models that are not easily transferable to other designs or LFO-driven effects. There are also examples of gray-box approaches, which strike a balance between general validity of the block-based model [8] and emulation quality of a specific unit [9, 10]. However, the modeling capabilities and robustness of such models is limited by the hand-engineered measurement techniques used to extract the LFO signal [9–12] and assumptions made about the LFO's shape.

Work in [14, 15] proposes recurrent and convolutional neural networks for black-box modeling of time-varying audio effects. Relying only on datasets of dry-wet audio, these are the first end-to-end approaches applied to LFO-driven effects. The method achieves good results with non-causal and non-controllable implementations, but does not explicitly learn the LFO signal and is not evaluated on unseen effects, audio, or LFO shapes.

To address the challenges of modeling a wide range of modulation effects and configurations, we introduce a neural architecture that is trained to extract arbitrary LFO signals from phaser, flanger, and chorus audio effects across varying parameter settings. By training this model on a dataset of guitar signals with basic phaser, flanger, and chorus implementations, we demonstrate our model achieves:

- Accurate modulation extraction from unseen audio sources.
- Extrapolation to complex modulation signals such as quasiperiodic, combined, and distorted LFO signals.
- Generalization across effect implementations for unseen analog and digital phaser and flanger effects.
- End-to-end causal modeling of analog and digital LFO-driven effects without access to the internal LFO signal.

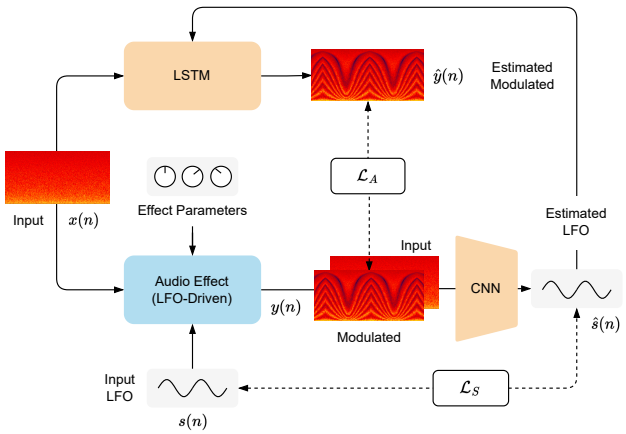

Figure 1: By using the pretrained LFO extraction model (CNN) to analyze input and modulated audio, our proposed system enables training of a black-box neural network model (LSTM) on modulation audio effects without access to the ground truth LFO signal.

[1] https://christhetree.github.io/mod_extraction/

2. BACKGROUND

2.1. Low Frequency Oscillators

In 1964 Robert Moog introduced the first transistor-based voltage controlled oscillator (VCO) and voltage controlled amplifier (VCA) designs [18]. These circuits are at the origin of modular synthesizers and later on led to modulation audio effects like phaser, flanger, and chorus. While VCOs were used to generate pitched sounds, VCAs were responsible for the envelope of synthesized notes. In his designs Moog also included VCOs oscillating at frequencies below 20 Hz, i.e. LFOs, to modulate other signal parameters (e.g., frequency, phase, amplitude) or processing blocks (e.g., panning, cutoff frequency). The most common types of modulations stem from periodic waveforms like sine, sawtooth, triangle, or square, but often extend to more complex shapes.

In analog effects [4, 10], non-ideal components can cause distortions from the reference shape as well as deviations that cause quasiperiodicity. There are also cases, like chorus effects, where random LFO signals are adopted. Furthermore, with the prevalence of digital emulations and software synthesizers, modulation signals can achieve an even wider diversity than their analog counterparts. As a result, the extraction of modulation signals from processed audio has applications beyond virtual analog modeling.

2.2. Modulation Effects

Phaser and flanger are examples of modulations affecting the spectrum of a signal, while chorus affects the pitch and timing.

Phaser — Phasing is achieved by using a series of notch or allpass filters [19]. The typical analog implementation uses an even number of first order allpass filters, which have flat magnitude response but phase that varies between 0° and -180°. When two filters are connected in series the phase varies back to 0° and, by mixing the filtered output with the input signal phase cancellations occur at frequencies around the 180° point. Altering the center frequency of the filters creates a characteristic sweeping sound.

Flanger — In a flanger, a delayed copy of the input signal is summed to the dry input itself causing constructive and destructive interference. The delay is periodically modulated but usually kept below ≈ 15 ms. As a result, it is often perceived as a time-varying comb filter. In contrast to phaser effects, where the frequency distance between notches is kept constant on a logarithmic scale, in flanger effects, the distance changes with the delay value.

Chorus — Chorus effects are identical to flangers in implementation, but use multiple delayed and modulated copies of the input signal. Also, by adopting larger delays - around ≈ 30 ms - the output is perceived as a sum of slightly pitch shifted copies of the input, as when multiple instruments or voices are playing in unison. Therefore, there is not a clearly observable modulation of the spectrum compared to phasers and flangers.

2.3. Virtual Analog Modeling of Modulation Effects

Research in virtual analog modeling aims to develop methods that emulate the characteristics and behaviors of a reference unit. These methods can be divided into white-, gray-, or black-box modeling depending on the degree of prior knowledge and type of measurements they rely on. To create accurate simulations, white-box modeling [4–7] requires a thorough understanding of the system, and typically employs differential equations to describe its behavior and numerical methods to solve them. Therefore, such methods are often associated with a time consuming design process and computationally demanding and non-transferable implementations. Circuit analysis together with voltage and current measurements are used to create a state-space model of a phaser effect pedal in [6], while in [7] a similar analysis is used to emulate a bucket brigade delay circuit that is then employed in flanger effect emulation. Phaser, flanger, and chorus are also modeled in [4], where the authors discretize the differential equations of JFET transistors and transconductance amplifiers used in such effects.

To reduce prior knowledge necessary to model a device, gray-box approaches combine a partially theoretical structure with input-output measurements [9, 10, 12]. However, they still require ad hoc measurement and optimization procedures [9, 12] and knowledge of the underlying implementation. A gray-box model of phaser effect pedal is presented in [10], where nonlinear allpass filter blocks are combined with analysis and measurement of the interaction between light dependent resistors and incandescent lamp optocoupler controlling the LFO. This work shows how critical the LFO signal can be in shaping the overall sound of a design. In [9] we have an example of a measurement signal and extraction algorithm specifically designed to capture a phaser's LFO signal.

A similar measurement is adopted in [11], and the extracted LFO signal is used to condition neural networks trained on phaser and flanger effects. A custom extraction algorithm is implemented: the LFO shape (rectified sine) is observed in the output and given to a least-squares solver. Furthermore, custom training data is required, where the test signal is interposed between samples so that the initial LFO phase can be extracted. This work is developed further in [12] by improving the measurement technique.

In black-box approaches, minimal knowledge of the system is required and modeling mostly relies on input-output measurements. A major advantage is that they simplify the process to collecting adequate data. However, these models often lack interpretability and might entail time-consuming optimizations. In [14, 15], we have the only examples of black-box models of time-varying audio effects. Neural networks are successfully trained on many modulation effect types. However, these models are non-causal, non-controllable, and have not been tested on unseen LFO shapes or audio signals different from the training data.

2.4. Effects Recognition and Parameter Estimation

Beyond effect modeling, there has also been research on recognition of audio effects and effect chains, as well as control values from processed audio. Our task of LFO extraction can be viewed as a specific form of audio effect parameter estimation, however, existing works have yet to consider reconstructing the LFO signal itself. Early works focus on audio effects classification [20], while others extend this task to target the identification of specific effect units and their control values [21], including within mixtures [22]. Recently, work has generalized this task to the complete reconstruction of a graph of audio effects and their parameter values [23]. In [24], the authors focus on dynamic range compression, and train neural networks to extract ratio, attack, and release times, and total harmonic distortion from a reference signal. Extracting information from audio recordings for applications in music production and sound synthesis is still at an early stage, and the work presented here also aims to contribute in these directions.

3. METHODOLOGY

We approach the problem of modeling an LFO-driven audio effect in two steps. First, we develop an LFO extraction model which can be trained to reconstruct the modulation signal from dry and wet audio pairs. Then we feed the extracted LFO signal along with the dry audio to an effect model that can be trained to reconstruct the wet audio. Figure 1 visualizes our approach with a block diagram.

3.1. LFO Extraction

The LFO extraction model (LFO-net), shown in Figure 2, is a convolutional neural network (CNN) consisting of sequential convolutional blocks. As input it takes a 2-channel Mel spectrogram of the dry and wet audio. Each block consists of LayerNorm [25] across the frequency and time dimensions, a 2D convolution, Max Pooling, and a PReLU activation [26]. Feature maps are max-pooled only along the frequency dimension and dilated only along the time dimension, similar to how a temporal convolutional network (TCN) operates [27]. As a result, the temporal receptive field of the network grows exponentially with each convolutional block while the frequency resolution decreases exponentially. The final layer of the network is a time-distributed linear layer that estimates the LFO value for the current frame between 0 and 1.

Training — LFO-net is trained using the AdamW optimizer to minimize the L_1 error between the reconstructed modulation signal $\hat{s}$ and the ground truth modulation signal s, each with N timesteps

$$L_1(s, \hat{s}) = \frac{1}{N} \sum_{n=1}^{N} |s(n) - \hat{s}(n)| \tag{1}$$

where n is the time index. We also include terms for the first-order central difference error

$$s'(n) = \frac{s(n+1) - s(n-1)}{2}. \tag{2}$$

as well as the L_1 error of the second-order central difference, which is defined recursively

$$s''(n) = \frac{s'(n+1) - s'(n-1)}{2}. \tag{3}$$

These terms are scaled by α, β, and γ respectively and encourage the network to learn smoother modulation signals. The complete loss function $\mathcal{L}_S$ can be expressed as follows:

$$\mathcal{L}_S = \alpha L_1(s, \hat{s}) + \beta L_1\left(s', \hat{s}'\right) + \gamma L_1\left(s'', \hat{s}''\right) \tag{4}$$

Based on initial testing, we selected $\alpha = 1$, $\beta = 5$, and $\gamma = 10$ to weigh the different terms. In addition, SpecAugment [28] was used for masking both frequency and time dimensions during training to increase robustness.

Post-processing — Since LFO-net imposes no restrictions on the shape of the LFO besides being bounded between 0 and 1, the output can appear noisy or irregular. To improve the quality of the extracted LFO signal we introduce three post-processing steps, shown in Figure 4. First, the signal is smoothed with a 4th order moving average filter. This is followed by "stretching" of the peaks and troughs so they are equal to 0 and 1, respectively. This is achieved by finding the locations of local minima and maxima, and then linearly interpolating consecutive sections to span from 0 to 1. Finally, when training effect models, invalid reconstructed LFO signals where there are too many peaks or troughs or where consecutive peaks or troughs are too close together are thrown out.

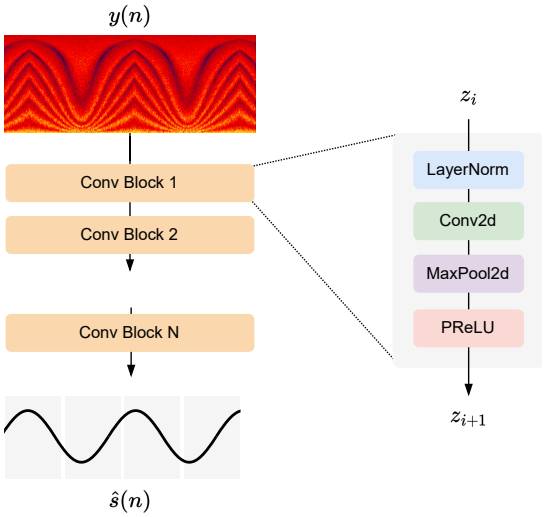

Figure 2: LFO extraction model (LFO-net) diagram.

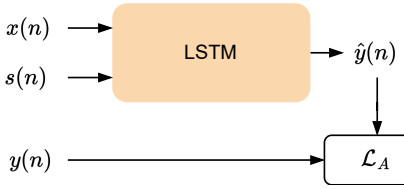

Figure 3: LFO effect modeling block diagram.

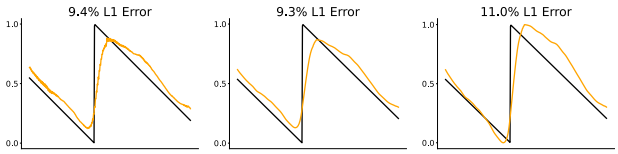

Figure 4: Examples of original (left), smoothed (center), and stretched (right) post-processed modulation signals.

This helps stabilize training by only using examples where the estimated LFO is likely to be an accurate prediction of the ground truth LFO signal. The last two post-processing steps are only used for the unseen effect experiments in Section 4.5.

3.2. Effect Modeling

Our effect model, shown in Figure 3, is based on previous work in black-box modeling of modulation effects [12]. It consists of a long short-term memory (LSTM) network with a time-distributed linear layer that compresses the latent space into a single sample-by-sample value which is added to the input audio and then bounded by a hyperbolic tangent activation. The network takes as input two channels: the dry audio and an LFO conditioning signal.

Training — Training is done on blocks of 1024 samples using truncated backpropagation through time (TBPTT) with 1024 samples of warmup. Once again, the AdamW optimizer is used to minimize $\mathcal{L}_A$: the L_1 loss between the output audio and the ground truth wet audio. We do not train using Error-to-Signal Ratio (ESR) and DC loss as in [29] since in our experiments we found using just the L_1 loss resulted in better results across all metrics.

Table 1: Parameter values for the "fixed params" and "varying params" evaluation configurations.

		LFO Parameters			Effect Parameters					
Effect	Config.	Shape	Phase	Rate	Center Freq.	Min. Delay	Delay Width	Feedback	Depth	Mix
Phaser	Fixed	Cos.	0 - 2π	0.5 - 3.0 Hz	440 Hz	-	-	0.25	1.0	1.0
	Varying				70 - 18k Hz	-	-	0.0 - 0.7	0.25 - 1.0	1.0
Flanger	Fixed	All	0 - 2π	0.5 - 3.0 Hz	-	1 ms	4 ms	0.25	1.0	1.0
	Varying				-	0 - 1 ms	2.5 - 10 ms	0.0 - 0.7	0.25 - 1.0	1.0
Chorus	Fixed	All	0 - 2π	0.5 - 3.0 Hz	-	20 ms	10 ms	0.25	1.0	1.0
	Varying				-	11 - 30 ms	2.5 - 10 ms	0.0 - 0.7	0.25 - 1.0	1.0

4. EXPERIMENTS

4.1. Modulation Extraction

Most phaser, flanger, and chorus implementations do not allow defining an arbitrary LFO signal. As a result, in order to be able to train the LFO extraction model with effects using arbitrary LFO signals, we implement our own flanger/chorus effect directly in PyTorch so that it can run on GPU and be integrated into our data pipelines. We use six different LFO shapes: cosine (cos), triangle (tri), rectified cosine (rect. cos), inverse rectified cosine (inv. rect. cos), sawtooth (saw), and inverse sawtooth (inv. saw). The LFO parameters of the module are phase, rate, and shape and the effect parameters are min. delay, delay width, feedback, depth, and mix. For the flanger effect we set the minimum delay to 0-1 ms whereas for a chorus effect we set it to 10-20 ms. We also use a modified version of the phaser provided in `Pedalboard`[2], which allows us to specify the LFO phase, while its shape remains restricted to a cosine waveform. Its LFO parameters are phase and rate and its effect parameters are center frequency, feedback, depth, and mix.

Dataset — We use the fourth subset of the IDMT-SMT-Guitar [30] dataset, which contains 64 short electric guitar pieces grouped by genre. Each piece has been recorded at a fast and a slow tempo using three different guitars. We remove the two bars of synchronization tones at the beginning of each piece and split into 75% training and 25% validation sets across the 64 unique songs. This results in 154 min of audio in the training set and 50 min of audio in the validation set. We generate LFO signals with random phase, shape, and rate between 0.5 and 3 Hz and then apply the three audio effects to random 2-second chunks of the dataset while uniformly sampling the effect parameters within their usable ranges.

Training — The input to LFO-net is a Mel spectrogram with 1024 FFT size, 256 sample hop length, 256 Mel bins, and a sample rate of 44.1 kHz. The model consists of 6 convolutional blocks, each with 64 channels, a kernel size of 5 by 13, and a frequency max-pooling and temporal dilation factor of 2. As a result, the receptive field of the network along the time axis spans 2 seconds and outputs 345 frames given 88200 input samples. SpecAugment of 25% is applied during training to both the frequency and time axes. The model contains 1.3 M parameters.

Evaluation — During evaluation of LFO-net, we smooth the signal using a 4th order moving average filter and keep phase, shape, and rate of the LFO signal random. We define two different effect parameter configurations to compare against: "fixed params" and "varying params", summarized in Table 1. We evaluate on 1000 random 2-second non-silent chunks of the dataset. As a baseline, we assume an experienced audio engineer could correctly guess the shape of the LFO signal, whether it's going up or down, and the approximate rate of modulation from listening to the wet audio. We define this as an LFO signal with the correct shape, a random phase error of up to 50%, and a random rate error of up to 25%.

[2] https://github.com/spotify/pedalboard

4.2. Unseen Audio Sources

We evaluate the LFO-net on five unseen datasets processed with the `Pedalboard` phaser and our flanger/chorus implementation using the same setup described in the previous experiment (Section 4.1). These datasets are guitar, bass/double bass, and keyboard audio from MedleyDB 2.0 [31], drums from the IDMT-SMT-Drums [32] dataset, and vocals from VocalSet [33].

4.3. Quasiperiodic, Combined, and Distorted Modulations

Irregular LFO shapes can greatly expand the creative possibilities of an effect and are commonplace in virtual synthesizers. Furthermore, the internal LFOs of analog audio effects are imperfect and can drift or become distorted. As a result, we test the ability of our LFO extraction model to generalize to irregular LFO shapes. We generate quasiperiodic LFO signals by randomly stretching each cycle of a periodic modulation by 10 - 33.33%. We generate combined LFO signals by swapping out random cycles of a periodic modulation with a different shape. We try combining all six shapes randomly together and the four symmetrical shapes (no sawtooth and inverse sawtooth). Finally, we distort LFO signals via exponentiation, which makes different sections of the signal more concave or convex. Figure 5 shows examples of these three types of irregular LFO signals. We then evaluate on the test dataset using the "fixed params" evaluation configuration.

4.4. Latent Space Visualization

In order to see whether the model learns meaningful representations in its latent space we generate three different visualizations. We perform inference on 200 samples from the validation dataset while changing one variable and keeping all others fixed. A single 64-dimensional latent vector is obtained by taking the average across the output frames of the final convolutional block in the model. We then produce a 2d visualization of the vectors using principle component analysis (PCA). We explore how the rate and shape of the LFO signal are encoded as well as the difference between the phaser, flanger, and chorus effects.

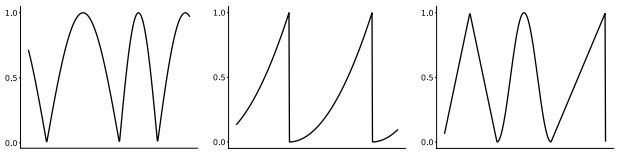

Figure 5: Examples of quasiperiodic (left), distorted (center), and combined (right) LFO shapes.

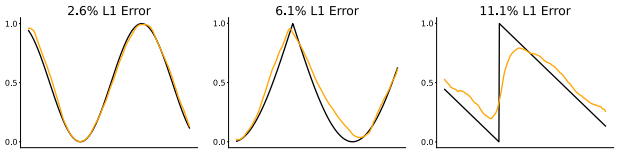

Figure 6: Examples of 3%, 6%, and 11% extracted LFO L_1 errors.

4.5. Unseen Analog and Digital Effects

Our final experiment evaluates whether LFO-net can be applied to unseen analog and digital effects and then be used to condition and train an effect emulation model. We use the EGFxSet dataset [34], which consists of five-second long recordings of single electric guitar notes processed with an MXR Phase 45 phaser pedal, a Mooer E-Lady flanger pedal, and a Boss CE-3 chorus pedal. Referencing the datasheets of these effects, we established that all three effects use a rounded triangle LFO shape. We peak normalize the input chunks of audio since the volume levels differ significantly between the wet and dry audio pairs. We apply all post-processing steps described in Section 3.1 during training and inference when extracting the LFO signal (8th order moving average for smoothing) and use a 70/18/12% train-val-test split.

For digital effects we use the MeldaProduction MPhaser and MFlanger plugins[3]. These effects give the user control over the LFO signal and enable combined and irregular LFO signals to be drawn in the user interface. We test two scenarios. First, we consider modeling a phaser and flanger effect with an irregular LFO signal and then with a quasiperiodic LFO signal. For the irregular case we define a skewed sinusoidal LFO shape as shown in Figure 7 at a frequency of 0.75 Hz. For the quasiperiodic case we start with a triangle shape and automate the rate of the LFO from 0.5 Hz to 2.0 Hz and back every 4 seconds. We apply both effects to 8 minute training, 2.5 minute validation, and 2 minute test sets from the fourth subset of the IDMT-SMT-Guitar dataset. During post-processing for the irregular case, we omit step 2 (stretching) to preserve the original shape of the extracted LFO signal.

Since we do not have access to the internal LFO signal for these effects, we first confirm visually that the LFO extraction model is able to output similar LFO signals when applied to these unseen effects. We then use it to train one effect model LSTM with 64 hidden units for each of the seven analog and digital effect configurations defined previously that learns to reconstruct the wet audio given the dry audio and the extracted LFO signal. As a baseline we also train effect models conditioned on a randomly generated LFO with a triangle shape and 0-25% frequency error for the analog effects, a triangle shape and random frequency between 0.5 - 2.0 Hz for the quasiperiodic experiment, and a cosine shape and 0-25% frequency error for the irregular LFO signal experiment.

[3] https://www.meldaproduction.com/effects/free

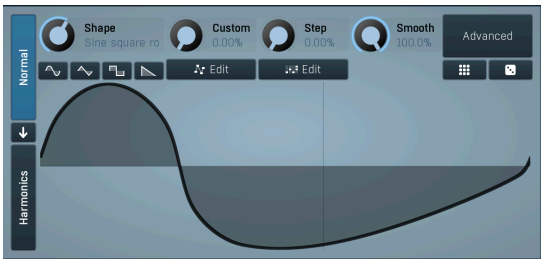

Figure 7: Skewed sinusoidal LFO shape used in the Melda Phaser and Flanger irregular LFO effect modeling experiments.

5. RESULTS

5.1. Modulation Extraction

Table 2 summarizes the ability of the model to extract LFO signals from the test dataset. Figure 6 provides a visual reference for the reconstruction quality corresponding to different L_1 error values. We find that an L_1 error of less than 5% corresponds to very accurate extraction with less than 10% error still being acceptable. We notice that the model struggles most with the asymmetrical sawtooth shapes. This is likely due to the waveform containing sharp edges, which can be difficult to reconstruct. We also observe that the model is better at extracting the LFO from the phaser, and worse at extracting the LFO from the chorus. This matches our intuition since the phaser is limited to a cosine LFO shape and because the chorus effect contains the largest varying delay which results in the greatest change in the wet audio spectrogram compared to the flanger. Finally, there is no difference in model performance when the parameters are fixed or varying across their entire usable ranges, thus highlighting the learning capabilities of the proposed LFO model architecture. The baseline consistently results in very large errors due to the fact that small differences in phase and frequency can cause the baseline and ground truth signal to drift apart. We also experimented with extracting the LFO signal from just the wet audio (no dry audio channel) and found that this resulted in an approximately 3% increase in the L_1 error.

5.2. Unseen Audio Sources

We find the model generalizes well to unseen data processed with our three training effects. From Table 3 we see that LFO-net performs just as well or even better on the unseen guitar, bass, and keys datasets. Performance on vocals is also only marginally worse. We expect drums to be the most challenging to extract LFO signals from due to the less tonal and dense onsets and the results match this intuition with extraction ability becoming worse for the flanger and chorus effects on the drums dataset. Varying parameters also results in a very small reduction in performance compared to fixed parameters.

5.3. Quasiperiodic, Combined, and Distorted

The quasiperiodic, distorted, and combined LFO signal results are contained in Tables 4 and 5. The ability to extract quasiperiodic signals is only slightly reduced when compared to periodic signals with the chorus and asymmetrical shapes appearing more challenging than the flanger and symmetrical shapes. This implies the system could be used to obtain an LFO signal for non-periodic audio effects.

Table 2: LFO extraction evaluation metrics.

		L_1 Error (%)		
Effect	LFO Shape	Fixed	Varying	Baseline
Phaser	Cosine	1.8%	2.1%	32%
Flanger	Cosine	1.9%	1.9%	32%
	Triangle	2.2%	2.3%	27%
	Rect. Cosine	2.2%	2.1%	28%
	Inv. Rect. Cos.	1.9%	2.0%	28%
	Saw	4.5%	4.5%	27%
	Inv. Saw	4.9%	4.7%	27%
	All	2.9%	2.9%	28%
Chorus	Cosine	3.6%	2.9%	32%
	Triangle	3.1%	3.3%	27%
	Rect. Cosine	2.7%	2.9%	28%
	Inv. Rect. Cos.	2.9%	2.9%	28%
	Saw	8.0%	6.9%	27%
	Inv. Saw	8.5%	7.3%	27%
	All	4.7%	4.3%	28%
All	All	3.1%	3.1%	29%

Table 3: LFO extraction metrics for unseen datasets.

		L_1 Error (%)			
Dataset	Params	Phaser	Flanger	Chorus	All
MDB Guitar	Fixed	1.8%	2.8%	4.7%	3.1%
	Varying	1.8%	2.8%	4.9%	3.2%
MDB Bass	Fixed	1.9%	2.4%	4.3%	2.9%
	Varying	2.3%	2.6%	4.7%	3.2%
MDB Keys	Fixed	1.8%	2.5%	4.2%	2.8%
	Varying	2.3%	2.5%	4.0%	2.9%
IDMT Drums	Fixed	1.9%	5.3%	12.2%	6.5%
	Varying	2.7%	5.8%	11.3%	6.6%
Vocalset	Fixed	2.8%	4.3%	5.4%	4.2%
	Varying	2.7%	4.2%	5.8%	4.2%

Distorted inverse rectified cosine, saw, and inverse saw are also difficult for LFO-net to extract. We believe this is because the inverse rectified cosine shape becomes closer to a square wave at the troughs when exponentiated which results in a constant delay and less sweeping patterns in the spectrum to analyze. Similarly, the saw and inverse saw shapes become even more jagged at the corners, thus making reconstruction more challenging, especially at higher LFO rates. Finally, we found that LFO-net is better at reconstructing random combinations of the LFO shapes when the asymmetrical ones are omitted. We believe this is due to the harsh discontinuities that can be introduced by combining sawtooth waves with the other symmetrical waves. Our results indicate that the model can extract symmetrical modulation shapes well, even when each period consists of a different shape.

Table 4: LFO extraction metrics for quasi. and distorted signals.

		L_1 Error (%)			
Effect	LFO Shape	Quasi.	Base.	Dist.	Base.
Flanger	Cosine	3.3%	32%	3.4%	33%
	Triangle	3.6%	28%	2.4%	30%
	Rect. Cosine	3.7%	28%	1.9%	32%
	Inv. Rect. Cos.	3.3%	29%	8.1%	28%
	Saw	5.8%	27%	13%	32%
	Inv. Saw	6.5%	28%	13%	31%
	All	4.5%	29%	6.7%	31%
Chorus	Cosine	4.7%	32%	4.6%	33%
	Triangle	5.3%	28%	3.1%	30%
	Rect. Cosine	4.9%	28%	3.6%	32%
	Inv. Rect. Cos.	4.3%	29%	8.7%	28%
	Saw	10%	27%	16%	32%
	Inv. Saw	11%	28%	16%	31%
	All	7.0%	29%	8.5%	31%
Both	All	5.8%	29%	7.6%	31%

Table 5: LFO extraction metrics for combined modulations.

		L_1 Error (%)	
Effect	LFO Shapes	Combined	Baseline
Flanger	Symmetrical	4.7%	33%
	All	9.4%	34%
Chorus	Symmetrical	6.1%	33%
	All	11.2%	34%
Both	Symmetrical	5.4%	33%
	All	10.3%	34%

5.4. Latent Space Visualization

The latent space visualizations for changing LFO shape, effect, and rate are shown in Figures 8, 9, and 10, respectively. The latent space decouples for all three visualizations with the relationship between different LFO shapes being encoded by the distance of their clusters in the latent space. Opposite pairs of shapes (i.e. saw / inverse saw and rect. cos. / inv. rect. cos.) are separated by a large distance and similar shapes like triangle and cosine are close together. Similarly, the three different effects decouple in the latent space with chorus and flanger having more overlap since they are identical in implementation, but with different delay amounts. We expect the phaser effect to be the most distinct since it is a unique implementation. Finally, the LFO rate visualization displays a clear relationship between the frequency of the LFO and position in the latent space, with high frequencies becoming more densely clustered together.

5.5. Unseen Analog and Digital Effects

We are able to use LFO-net to model unseen analog and digital audio effects using the effect model described in Section 3.2. Figure 11 shows some examples of the extracted LFO signals from the different effects. For the EGFx analog effects dataset, we see best results on the phaser effect, followed by the chorus, and then the

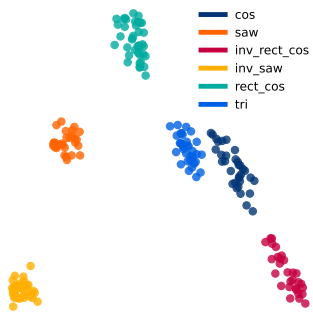

Figure 8: LFO shape

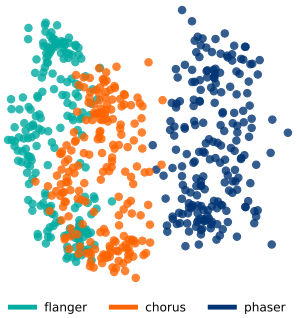

Figure 9: LFO effect type

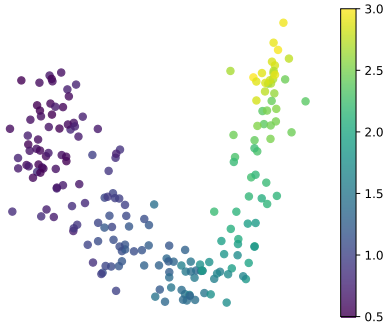

Figure 10: LFO rate

Table 6: Unseen effect evaluation results.

		Audio Error		Baseline Error	
Effect	LFO Shape	L_1 (%)	ESR	L_1 (%)	ESR
EGFx Phaser	Analog Tri.	3.5%	0.42	6.1%	0.78
EGFx Flanger	Analog Tri.	5.8%	0.94	5.9%	0.95
EGFx Chorus	Analog Tri.	5.0%	0.59	6.6%	0.82
Melda Phaser	Quasi. Tri.	1.4%	0.21	2.7%	0.61
	Irregular	0.76%	0.08	3.0%	0.78
Melda Flanger	Quasi. Tri.	2.3%	0.13	5.3%	0.51
	Irregular	2.9%	0.18	5.2%	0.45

flanger, which is not able to be modeled effectively. We found this dataset to be challenging due to large differences in power supply noise between dry and wet audio pairs, making it difficult to interpret the error metrics and forcing the LSTM to learn to model these differences as well. Despite this, the phaser is able to be modeled and sounds close to the wet audio from informal listening. We provide audio samples in the supplemental material.

The chorus effect is not modeled very well, but in our initial experiments we found that the LSTM effect model is unable to learn chorus effects, even when presented with the ground truth LFO signal, due to the long delays they make use of. As a result, we are surprised to see that the chorus model performs better than the baseline and is sometimes able to match the volume envelope of the wet audio. We also notice that the flanger appears to have two modulations occurring in its spectrogram. LFO-net is able to reliably extract one of them, but this is insufficient for modeling the effect. We believe extracting multiple modulations from audio is a natural future research direction to continue this work on.

For the Melda digital effects we see that both the irregular and quasiperiodic phaser and flanger effects are able to be captured successfully by the effect model. Our informal listening tests also confirm that they sound close to the target wet audio. The baseline model is able to capture the effects to an extent, but struggles especially with the quasiperiodic and irregular phaser LFO signals. The difference in the final ESR highlights the importance of providing an accurate LFO signal to the effect model.

We plot extracted LFO signals from unseen audio effects in Figure 11. A similar LFO shape to the one shown in Figure 7 is extracted for the flanger, but for the phaser it is extracted as two individual rounded peaks, one taller than the other. Since the irregular phaser is able to be modeled with a lower ESR than the irregular flanger, this indicates that this may be an artifact of the internal implementation of the Melda phaser, or that the exact LFO shape may not be required to successfully model an LFO-driven effect. We consider this another interesting future research direction.

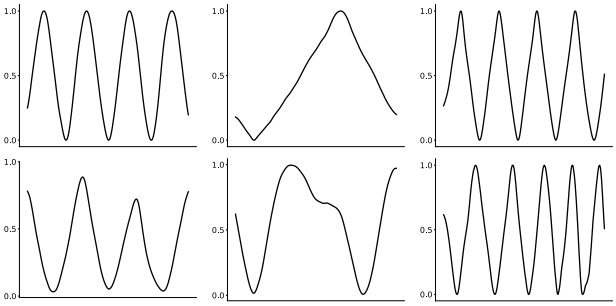

Figure 11: Extracted LFO patterns from unseen audio effects.
Top row: EGFx Phaser, Flanger, Chorus
Bottom row: Melda Phaser Irregular, Flanger Irregular, Quasi.

6. CONCLUSIONS

In this work, we propose a system that extracts arbitrary LFO signals from processed audio for multiple LFO-driven audio effects (phaser, flanger, and chorus), parameter settings, and instrument configurations. Our approach does not impose any restrictions on the LFO shape, which allows our neural network architecture to generalize to quasiperiodic, combined, and distorted modulation signals. We test our pretrained network on LFO extraction from a multitude of unseen audio sources, including guitar, bass, keyboards, drums, and singing voice. We show through a visualization of the latent space that the network learns meaningful representations of the different modulation shapes, rates, and effects. Finally, we demonstrate that our pretrained extraction network enables end-to-end modeling of unseen analog and digital LFO-driven audio effects when coupled with a simple processing network, overcoming the need for cumbersome and hand-engineered LFO measurement methods. We find that asymmetrical and discontinuous LFO shapes, such as saw waveforms, are the most difficult to extract and that the effect model cannot learn LFO-driven effects that make use of larger delays or contain multiple modulations. We make our code available and provide the trained audio effect models in a real-time VST plugin.

7. ACKNOWLEDGMENTS

Funded by UKRI and EPSRC as part of the "UKRI CDT in Artificial Intelligence and Music", under grant EP/S022694/1.

8. REFERENCES

[1] Jyri Pakarinen and David T. Yeh, "A review of digital techniques for modeling vacuum-tube guitar amplifiers," *Computer Music Journal*, vol. 33, no. 2, 2009.

[2] David T. Yeh, Jonathan S. Abel, and Julius O. Smith, "Automated physical modeling of nonlinear audio circuits for real-time audio effects—Part I: Theoretical development," *TASLP*, vol. 18, no. 4, 2009.

[3] Vesa Välimäki, Federico Fontana, Julius O. Smith, and Udo Zölzer, "Introduction to the special issue on virtual analog audio effects and musical instruments," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, 2010.

[4] Antti Huovilainen, "Enhanced digital models for analog modulation effects," in *DAFx*, 2005.

[5] Julian Parker, "A simple digital model of the diode-based ring-modulator," in *DAFx*, 2011, vol. 14.

[6] Felix Eichas, Marco Fink, Martin Holters, and Udo Zölzer, "Physical modeling of the mxr phase 90 guitar effect pedal," in *DAFx*, 2014.

[7] Jaromír Mačák, "Simulation of analog flanger effect using bbd circuit," in *DAFx*, 2016.

[8] Julius O. Smith, "An allpass approach to digital phasing and flanging," in *ICMC*, 1984.

[9] Roope Kiiski, Fabián Esqueda, and Vesa Välimäki, "Time-variant gray-box modeling of a phaser pedal," in *DAFx*, 2016.

[10] Champ Darabundit, Russell Wedelich, and Pete Bischoff, "Digital grey box model of the uni-vibe effects pedal," in *DAFx*, 2019.

[11] Alec Wright and Vesa Välimäki, "Neural modelling of periodically modulated time-varying effects," in *DAFx*, 2020.

[12] Alec Wright and Vesa Välimäki, "Neural modeling of phaser and flanging effects," *Journal of the Audio Engineering Society*, vol. 69, no. 7/8, 2021.

[13] Joseph T. Colonel, Marco Comunità, and Joshua D. Reiss, "Reverse engineering memoryless distortion effects with differentiable waveshaper," in *153rd AES*, 2022.

[14] Marco A. Martínez Ramírez, Emmanouil Benetos, and Joshua D. Reiss, "A general-purpose deep learning approach to model time-varying audio effects," in *DAFx*, 2019.

[15] Marco A. Martínez Ramírez, Emmanouil Benetos, and Joshua D. Reiss, "Deep learning for black-box modeling of audio effects," *Applied Sciences*, vol. 10, no. 2, 2020.

[16] Christian J. Steinmetz and Joshua D. Reiss, "Efficient neural networks for real-time modeling of analog dynamic range compression," in *152nd AES*, 2022.

[17] Marco Comunità, Christian J. Steinmetz, Huy Phan, and Joshua D. Reiss, "Modelling black-box audio effects with time-varying feature modulation," in *ICASSP*, 2023.

[18] Robert A. Moog, "Voltage controlled electronic music modules," *Journal of the Audio Engineering Society*, vol. 13, no. 3, 1965.

[19] William M. Hartmann, "Flanging and phasers," *Journal of the Audio Engineering Society*, vol. 26, no. 6, 1978.

[20] Michael Stein, Jakob Abeßer, Christian Dittmar, and Gerald Schuller, "Automatic detection of audio effects in guitar and bass recordings," in *128th AES*, 2010.

[21] Marco Comunità, Dan Stowell, and Joshua D. Reiss, "Guitar effects recognition and parameter estimation with convolutional neural networks," *Journal of the Audio Engineering Society*, vol. 69, no. 7/8, 2021.

[22] Reemt Hinrichs, Kevin Gerkens, Alexander Lange, and Jörn Ostermann, "Convolutional neural networks for the classification of guitar effects and extraction of the parameter settings of single and multi-guitar effects from instrument mixes," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2022, no. 1, 2022.

[23] Sungho Lee, Jaehyun Park, Seungryeol Paik, and Kyogu Lee, "Blind estimation of audio processing graph," in *ICASSP*, 2023.

[24] Di Sheng and György Fazekas, "A feature learning siamese model for intelligent control of the dynamic range compressor," in *IJCNN*. IEEE, 2019.

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

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *ICCV*, 2015.

[27] Colin Lea, Michael D. Flynn, Rene Vidal, Austin Reiter, and Gregory D. Hager, "Temporal convolutional networks for action segmentation and detection," in *CVPR*, 2017.

[28] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *Interspeech 2019*, Sep 2019.

[29] Alec Wright, Eero-Pekka Damskägg, and Vesa Välimäki, "Real-time black-box modelling with recurrent neural networks," in *DAFx*, 2019, pp. 1–8.

[30] Christian Kehling, Jakob Abeßer, Christian Dittmar, and Gerald Schuller, "Automatic tablature transcription of electric guitar recordings by estimation of score-and instrument-related parameters," in *DAFx*, 2014.

[31] Rachel M. Bittner, Julia Wilkins, Hanna Yip, and Juan P. Bello, "Medleydb 2.0: New data and a system for sustainable data collection," *ISMIR Late Breaking Demo*, p. 36, 2016.

[32] Christian Dittmar and Daniel Gärtner, "Real-time transcription and separation of drum recordings based on nmf decomposition," in *DAFx*, 2014.

[33] Julia Wilkins, Prem Seetharaman, Alison Wahl, and Bryan Pardo, "Vocalset: A singing voice dataset," in *ISMIR*, 2018.

[34] Hegel Pedroza, Gerardo Meza, and Iran R. Roman, "Egfxset: Electric guitar tones processed through real effects of distortion, modulation, delay and reverb," in *ISMIR Late Breaking Demo*, 2022.