NEURAL WA VESHAPING SYNTHESIS

Ben Hayes, Charalampos Saitis, George Fazekas
Centre for Digital Music, Queen Mary University of London
{b.j.hayes, c.saitis, g.fazekas}@qmul.ac.uk

ABSTRACT

We present the Neural Waveshaping Unit (NEWT): a novel, lightweight, fully causal approach to neural audio synthesis which operates directly in the waveform domain, with an accompanying optimisation (FastNEWT) for efficient CPU inference. The NEWT uses time-distributed multilayer perceptrons with periodic activations to implicitly learn nonlinear transfer functions that encode the characteristics of a target timbre. Once trained, a NEWT can produce complex timbral evolutions by simple affine transformations of its input and output signals. We paired the NEWT with a differentiable noise synthesiser and reverb and found it capable of generating realistic musical instrument performances with only 260k total model parameters, conditioned on F0 and loudness features. We compared our method to state-of-the-art benchmarks with a multi-stimulus listening test and the Fréchet Audio Distance and found it performed competitively across the tested timbral domains. Our method significantly outperformed the benchmarks in terms of generation speed, and achieved real-time performance on a consumer CPU, both with and without FastNEWT, suggesting it is a viable basis for future creative sound design tools.

1. INTRODUCTION

Synthesisers are indispensable tools in modern music creation. Over the last six decades, their evolving sonic affordances have defined uncountable musical aesthetics and cultures, enabling composers, sound designers, and musicians to interact with human auditory perception in previously impossible ways.

The recent proliferation of deep neural networks as audio synthesisers is further expanding the capabilities of these tools: realistic instrument performances can be synthesised from simple, low dimensional control signals [1–3]; the timbre of one instrument can be convincingly transferred to another [1,3,5]; instruments can be morphed and interpolated along nonlinear manifolds [6,7]; and sounds can be manipulated using high level descriptors of perceptual characteristics [7,9]. Yet despite their impressive abilities, these systems have not been widely adopted in music creation workflows.

We argue that this is largely a pragmatic issue. Modern music production centres around the digital audio workstation (DAW), with software instruments and signal processors represented as real-time plugins. These allow users to dynamically manipulate and audition sounds, responsively tweaking parameters as they listen or record. Neural audio synthesisers do not currently integrate elegantly with this environment, as they rely on deep neural networks with millions of parameters, and are often incapable of functioning in real-time on a CPU.

In this work we move towards integrating the benefits of neural audio synthesis into creative workflows with a novel, lightweight architecture built on the principles of digital waveshaping synthesis [10]. Our model implicitly learns a bank of continuous differentiable waveshapers, which are applied to an exciter signal. A control module learns to generate time-varying timbres by dynamically shifting and scaling the learnt waveshaper’s input and output. As the waveshapers encode information about the target timbre, our model can synthesise convincing audio using an order of magnitude fewer parameters than the current state-of-the-art methods.

This paper is laid out as follows. In section 2 we discuss related work on neural audio synthesis and waveshaping. Section 3 introduces our architecture, and we outline our training methodology in section 4. In section 5 we present and discuss evaluations of our model in comparison to the current state of the art methods [1,3]. Finally, we conclude with suggestions for future work in section 6. We provide full source code¹ and encourage readers to listen to the audio examples in the online supplement².

2. RELATED WORK

2.1 Neural Audio Synthesis

Audio synthesis with deep neural networks has received considerable attention in recent years. Autoregressive models such as WaveNet [11] and SampleRNN [12] defined a class of data-driven, general-purpose vocoder, which was subsequently expanded on with further probabilistic approaches, including flow-based models [13–15] and generative adversarial networks [16–19]. These models allow realistic synthesis of speech, and applications to musical audio [6,20,21] have yielded similarly impressive results. A parallel stream of research has focused on controllable musical audio synthesis [1,3,7,8,22], in which
models are designed to provide control affordances that may be of practical use. Such controls have included MIDI scores \([2, 22]\), semantic or acoustical descriptors of timbre \([7, 8]\), and F0/loudness signals \([1, 3]\). The representations of timbre learnt by these models have also been observed to show similarities to human timbre perception \([23]\).

A recent category of model, \([1, 3, 24]\) unified under the conceptual umbrella of differentiable digital signal processing (DDSP) \([1]\), has enabled low-dimensional, interpretable control through strong inductive biases to audio synthesis. Whereas generalised neural vocoders must learn from scratch to produce the features that typify audio signals, such as periodicity and harmony, DDSP methods utilise signal processing components designed to produce signals exhibiting such features. These components are expressed as differentiable operations directly in the computation graph, effectively constraining a model’s outputs to a subspace defined by the processor’s capabilities.

DDSP methods fall into two groups: those where the network generates control signals for a processor, and those where the network is trained to be a signal processor itself. The DDSP autoencoder \([1]\) falls into the first category as it generates control signals for a spectral modelling synthesiser \([25]\). The neural source-filter (NSF) approach \([3, 24, 26]\) is in the second category. It learns a nonlinear filter that transforms a sinusoidal exciter to a target signal, guided by a control embedding generated by a separate encoder. In other words: the control module “plays” the filter network.

The NSF filter network transforms its input through amplitude distortion, as each activation function acts as a nonlinear waveshaper. A given layer’s ability to generate a target spectrum is thus bounded by the distortion characteristics of its activation function. For this reason, neural source-filter models are typically very deep: Wang et al.’s simplified architecture \([24]\) requires 50 dilated convolutional layers, and Michelashvili & Wolf’s musical instrument model \([3]\) consists of 120 dilated convolutional layers – 30 for each of its four serial generators.

Our method avoids the need for such depth by learning continuous representations of detailed waveshaping functions as small multilayer perceptrons. These functions are optimised such that their amplitude distortion characteristics allow them to produce spectral profiles appropriate to the target timbre. This allows our model to accurately transform an exciter signal considerably more efficiently, whilst still exploiting the benefits of the network-as-synthesiser approach.

### 2.2 Digital Waveshaping Synthesis

In waveshaping synthesis \([10]\), timbres are generated using the amplitude distortion properties of a nonlinear shaping function \(f: \mathbb{R} \to \mathbb{R}\), which is memoryless and shift invariant. Due to its nonlinearity, \(f\) is able to introduce new frequency components to a signal \([2, 7]\). When a pure sinusoid \(\cos a\omega n\) is used as the input to \(f\), only pure harmonics are introduced to the signal. An exciter signal with multiple frequency components, conversely, would result in intermodulation distortion, generating components at frequencies \(a\omega_1 \pm b\omega_2, \forall a, b \in \mathbb{Z}^+\), for input frequencies \(\omega_1\) and \(\omega_2\). This would result in inharmonic components if \(\omega_1\) and \(\omega_2\) are not harmonically related.

The shaping function \(f\) is designed to produce a specific spectral profile when excited with \(\cos a\omega n\). This is achieved as a weighted sum of Chebyshev polynomials of the first kind, which possess the property that the \(k\)th polynomial \(T_k\) directly transforms a sinusoid to its \(k\)th harmonic: 

\[
T_k(\cos a\omega n) = \cos k(a\omega n)
\]

With a function specified in this way, we can define a simple discrete time waveshaping synthesiser

\[
x[n] = N[n]f(a[n] \cos \omega n),
\]

where \(a[n]\) is the distortion index and \(N[n]\) is a normalising coefficient. As the frequency components generated by a nonlinear function vary with input amplitude, varying the distortion index over time allows us to generate evolving timbres, whilst the normalising coefficient allows us to decouple the frequency content and overall amplitude envelope of the signal.

### 3. NEURAL WAVESHAPING SYNTHESIS

Our model acts as a harmonic-plus-noise synthesiser \([25]\). This architecture separately generates periodic and aperiodic components and exploits an inductive bias towards harmonic signals. Fig. \([1]\) illustrates the overall architecture of our model.

#### 3.1 Control Encoder

We condition our model on framewise control signals extracted from the target audio with a hop size of 128. We project these to a 128-dimensional control embedding \(z\) using a causal gated recurrent unit (GRU) of hidden size 128 followed by a time distributed dense layer of the same size. We leave the exploration of the performance of alternative sequence models to future work.

#### 3.2 NEWT: Neural Waveshaping Unit

The shaping function \(f\) of a waveshaping synthesiser can be fit to only a single instantaneous harmonic spectrum. The spectral evolution afforded by the distortion index \(a[n]\) is thus usually unrelated to the target timbre. This is a limitation of the Chebyshev polynomial method of shaping function design. Here, we propose to instead learn a shaping function \(f_\theta\) parameterised by a multilayer perceptron (MLP). As demonstrated in recent work on implicit neural representations \([28, 29]\), MLPs with sinusoidal activations dramatically outperform ReLU MLPs in learning continuous representations of detailed functions with arbitrary support. We therefore use sinusoidal activations in \(f_\theta\), which enables useful shaping functions to be learnt by very compact networks. Here, we use 64 parallel shaper MLPs, each with 4 layers, with a hidden size of 8 neurons.

To enable our model to fully exploit the distortion characteristics of \(f_\theta\), we replace the distortion index \(a[n]\) and normalising coefficient \(N[n]\) with affine transforms before
Figure 1. The full architecture of our neural audio synthesiser. All linear layers and MLPs are time distributed. Convolution is denoted * and applied by multiplication in the frequency domain. Blocks with dashed outlines operate at the same coarse time steps as the control signal, whilst those with solid outlines operate at audio rate.

Figure 2. A block diagram depicting the structure of the neural waveshaping unit (NEWT). Blocks with dashed outlines operate at control signal time steps, whilst solid blocks operate at audio rate.

and after the shaping function. The parameters of these transforms, denoted $\alpha_a$ and $\beta_a$ for the distortion index and $\alpha_N$ and $\beta_N$ for the normalising coefficient, are generated by a separate MLP (depth 4, width 128, ReLU activations with layer normalisation [30]) which takes $z$ as input, and then upsampled to audio rate. The output of a single NEWT in response to exciter signal $y[n]$ is thus given by:

$$x[n] = \alpha_N f_\alpha(\alpha_a y[n] + \beta_a) + \beta_N.$$  

In this way, the NEWT disentangles two tasks: it learns a synthesiser parameterised by $(\alpha_a, \alpha_N, \beta_a, \beta_N)$, and it learns to “play” that synthesiser in response to a control signal $z$. Fig. 2 illustrates the structure of the NEWT. In practice, we use multiple such units in parallel. We can implement this efficiently using grouped 1-dimensional convolutions with a kernel size of 1 — essentially a bank of parallel time-distributed dense layers.

3.3 FastNEWT

The NEWT is an efficient approach to generating time-varying timbres, but its reliance on grouped 1-dimensional convolutions best suits it to GPU inference. Many use-cases for our model do not guarantee the availability of a GPU, and so efficient CPU inference is of crucial importance. For this reason, we propose an optimisation called the FastNEWT: as each learnable shaping function simply maps $\mathbb{R} \rightarrow \mathbb{R}$, it can be replaced by a lookup table of arbitrary resolution. Forward passes through $f_\theta$ are then simply replaced with the $O(1)$ operation of reading values from an array and calculating an interpolation.

To produce a FastNEWT, we sample $f_\theta$ across a closed interval. The sampling resolution and interval are tunable parameters of this operation, and represent a trade-off between memory cost and reconstruction quality. Here, we opt for a lookup table of 4096 samples over the interval $[-3, 3]$, using a naïve implementation with linear interpolation. Like the rest of our model, this is implemented using PyTorch operations, and so we treat this as an upper bound on the computational cost of the FastNEWT. In practice, an implementation in a language with low level memory access would confer performance improvements.

3.4 Harmonic Exciter

To reduce the resolution required of the shaping functions, we produce our exciter with a harmonic oscillator bank generating up to 101 harmonics, truncated at the Nyquist frequency. The outputs of this oscillator bank are passed through a time distributed linear layer, acting as a mixer which provides each NEWT channel with a weighted mixture of harmonics. Thus, the $i$th output channel of the exciter module is given by:

$$y_i[n] = \sum_{k=1}^{K} A(k\omega) w_{ik} \cos k\omega n + b_i,$$  

where the antialiasing mask $A(k\omega)$ is 1 if $-\pi < k\omega < \pi$ and 0 otherwise.

3.5 Noise Synthesiser

In spectral modelling synthesis [25], audio signals are decomposed into a harmonic portion and a residual portion. The residual portion is typically modelled by filtered noise, with filter coefficients varying over time according to the spectrum of the residual. Here, we use an MLP (depth 4, hidden size 128, ReLU activations with layer normalisation) to generate 256-tap FIR filter magnitude responses conditioned on $z$. We apply a differentiable window-design method like that used in the DDSP model [1] to
apply the filters to a white noise signal. First, we take the inverse DFT of these magnitude responses, then shift them to causal form, and apply a Hann window to the impulse response. We then apply the filters to a white noise signal by multiplication in the frequency domain.

3.6 Learnable Reverb

To model room acoustics, we apply a differentiable convolutional reverb to the signal. We use an impulse response \( c[n] \) of length 2 seconds, initialised as follows:

\[
c[n] = \begin{cases} 
    \mathcal{N}(0; 1e-6), & \text{if } n > 1, \\
    0, & \text{if } n = 0.
\end{cases}
\]

(4)

\( c[n] \) is trainable for \( n \geq 1 \), whilst the 0th value is fixed at 0. The reverberated signal \( (c * x)[n] \) is computed by multiplication in the frequency domain, and the output of the reverb is summed with the dry signal.

4. EXPERIMENTS

Our model can be trained directly through maximum likelihood estimation with minibatch gradient descent. Here we detail the training procedure used in our experiments.

4.1 Loss

We trained our model using the multi-resolution STFT loss from [18]. A single scale of the loss is defined as the expectation of the sum of two terms. The first is the spectral convergence \( L_n \) (Eqn. 5) and the second is log magnitude distance \( L_m \) (Eqn. 5), defined as:

\[
L_n(x, \hat{x}) = \frac{\|\text{STFT}_m(x)\| - \|\text{STFT}_m(\hat{x})\|_{F}}{\|\text{STFT}_m(x)\|_{F}}
\]

and

\[
L_m(x, \hat{x}) = \frac{1}{m} \|\log \|\text{STFT}_m(x)\| - \log \|\text{STFT}_m(\hat{x})\|\|_1
\]

respectively, where \( \| \|_{F} \) is the Frobenius norm, \( \| \|_{1} \) is the L1 norm, and \( \text{STFT}_m \) gives the short-time Fourier transform with analysis window of length \( m \) for \( m \in \{512, 1024, 2048\} \). We used the implementation of this loss provided in the auraloss library [31].

4.2 Data

We collated monophonic audio files from three instruments (violin, trumpet, & flute) from across the University of Rochester Music Performance (URMP) dataset [32], and for each instrument applied the following preprocessing. We normalised amplitude across each instrument subset, for each instrument applied the following preprocessing.

5.1 Benchmarks

Two DDSP architectures were used as benchmarks: the “full” model used to train a solo violin synthesiser in the original paper, and the “tiny” model described in the paper’s appendices. Each model was trained for 30k iterations as recommended in the supplementary materials. We denote these \( \text{DDSP-full} \) and \( \text{DDSP-tiny} \), respectively. HTP consists of four distinct Parallel WaveGAN [18] generators operating at increasing timescales. We trained each for 120k iterations, as recommended in the original paper. Table 1 lists the total trainable parameter counts of all models under comparison.

5. EV ALUATION & DiscussON

To evaluate the performance of our model across different timbres, we trained a neural waveshaping model for each instrument subset. We denote these models \( \text{NWS} \), specifying the instrument where relevant. After training, we created optimised models with \( \text{FastNEWT} \), denoted \( \text{NWS-FN} \), and included these in our experiments also.

5.2 Fréchet Audio Distance

The Fréchet Audio Distance (FAD) is a metric originally designed for evaluating music enhancement algorithms [34], which correlates well with perceptual ratings of audio quality. It is computed by fitting multivariate Gaussians to embeddings generated by a pretrained VGGish model [35]. This process is performed for both the set under evaluation,
Table 2. Fréchet Audio Distance scores for all models using background embeddings computed across each instrument’s full dataset. Bold type indicates the best performance in a column and italics the second best.

<table>
<thead>
<tr>
<th>Model</th>
<th>Fréchet Audio Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Flute</td>
</tr>
<tr>
<td>Test Data</td>
<td>0.463</td>
</tr>
<tr>
<td>HTP</td>
<td>6.970</td>
</tr>
<tr>
<td>DDSP-full</td>
<td>3.091</td>
</tr>
<tr>
<td>DDSP-tiny</td>
<td>3.673</td>
</tr>
<tr>
<td>NWS</td>
<td><strong>2.704</strong></td>
</tr>
<tr>
<td>NWS-FN</td>
<td>2.717</td>
</tr>
</tbody>
</table>

Table 3. Real-time factor computed by synthesising four seconds of audio in a single forward pass across all benchmarks. Statistics computed over 100 runs. Bold type indicates the best performance in a column and italics the second best.

<table>
<thead>
<tr>
<th>Model</th>
<th>Real-time Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GPU Mean</td>
</tr>
<tr>
<td>HTP</td>
<td>0.105</td>
</tr>
<tr>
<td>DDSP-full</td>
<td>0.038</td>
</tr>
<tr>
<td>DDSP-tiny</td>
<td>0.032</td>
</tr>
<tr>
<td>NWS</td>
<td>0.004</td>
</tr>
<tr>
<td>NWS-FN</td>
<td><strong>0.003</strong></td>
</tr>
</tbody>
</table>

suggesting that FastNEWT has little, if any, impact on the perceptual quality of the synthesised audio. Across flute and trumpet trials our models were rated similarly to the benchmarks. In the first violin trial, our models’ ratings were similar to those of DDSP-tiny, whilst in the second they were lowest overall. These ratings are concordant with FAD scores: our model performs competitively on trumpet and flute whilst struggling somewhat with violin.

To examine the influence of melodic stimuli on participants’ ratings, we performed Wilcoxon’s signed-rank test between scores given for each instrument’s two stimuli, for each synthesis model. For example, scores given to DDSP-full for stimulus Flute 1 were compared to scores given to DDSP-full for Flute 2. Out of fifteen tests, significant differences ($p < .001$) were observed in two: between trumpet stimuli for both DDSP-full and HTP. No other significant effects were observed ($\alpha = 0.05$).

To examine the effect of synthesis model, we performed Friedman’s rank sum test on ratings from each trial. For flute stimuli, no significant effects were found. Significant effects were observed for both trumpet stimuli, although Kendall’s $W$ suggested only weak agreement between raters (Trumpet 1: $Q = 27.45, p < 0.001, W = 0.38$; Trumpet 2: $Q = 14.18, p < 0.01, W = 0.20$). Both violin stimuli also resulted in significant effects with moderate agreement between raters (Violin 1: $Q = 42.28, p < 0.001, W = 0.59$; Violin 2: $Q = 37.95, p < 0.001, W = 0.53$). Post-hoc analysis was performed within each trial using Wilcoxon’s signed-rank test with Bonferroni $p$-value correction. Significant differences (corrected threshold $p < .005$) were observed for Trumpet 1, Violin 1, and Violin 2. These are illustrated as brackets in Fig. 3.

5.4 Real-time Performance

We evaluated the real-time performance of our model in two scenarios. In both cases we took measurements on a GPU (Tesla P100-Pcie 16GB) and a CPU (Intel i5 1038NG7 2.0GHz) and used the real-time factor (RTF) as a metric. The RTF is defined as

$$RTF := \frac{t_p}{t_e},$$

(8)
6. CONCLUSION

In this paper, we presented the NEWT: a neural network structure for audio synthesis based on the principles of waveshaping [10]. We also present full source code, pretrained checkpoints, and an online supplement containing audio examples. Our architecture is lightweight, causal, and comfortably achieves real-time performance on both GPU and CPU, with efficiency further improved by the FastNEWT optimisation. It produces convincing audio directly in the waveform domain without the need for hierarchical or adversarial training. Our model is also capable of many-to-one timbre transfer by extracting F0 and loudness control signals from the source audio. Examples of this technique are provided in the online supplement, where we also offer insight into the specific shaping functions learned by the NEWT.

In evaluation with a multi-stimulus listening test and the Fréchet audio distance our model performed competitively with state-of-the-art methods with over $20 \times$ more parameters on trumpet and flute timbres, whilst performing similarly to a comparably sized DDSP benchmark on violin timbres. We suspect the lower scores on violin timbres were due to the greater proportion of harmonic energy at higher frequencies in these sounds. The NEWT may thus have failed to learn shaping functions capable of producing these high harmonics without introducing aliasing artefacts. Using deeper or wider MLPs inside the NEWT may allow more accurate shaping functions to be learnt, whilst retaining efficient inference with FastNEWT. Future work will investigate this and other differentiable antialiasing strategies, including adaptive oversampling [39]. We will also explore extending our model to multi-timbre synthesis.

7. ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers at ISMIR for their thoughtful comments. We would also like to thank our colleague Cyrus Vahidi for many engaging and insightful discussions on neural audio synthesis. This work was supported by UK Research and Innovation [grant number EP/S022694/1].

8. REFERENCES


