Sonos Seminar Series • 1 September 2022

Style transfer of audio effects with differentiable signal processing


Christian J. Steinmetz[1,2]
c.j.steinmetz@qmul.ac.uk


Nick J. Bryan[2]


Joshua D. Reiss[1]

[1]Queen Mary University of London
[2]Adobe Research

arxiv.org/abs/2207.08759

Adobe · Queen Mary University of London · UK Research and Innovation · AI + MUSIC

More people are creating **audio** content

Podcasts

Sound for Video

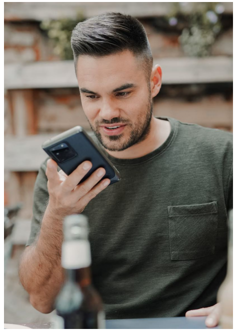
Music

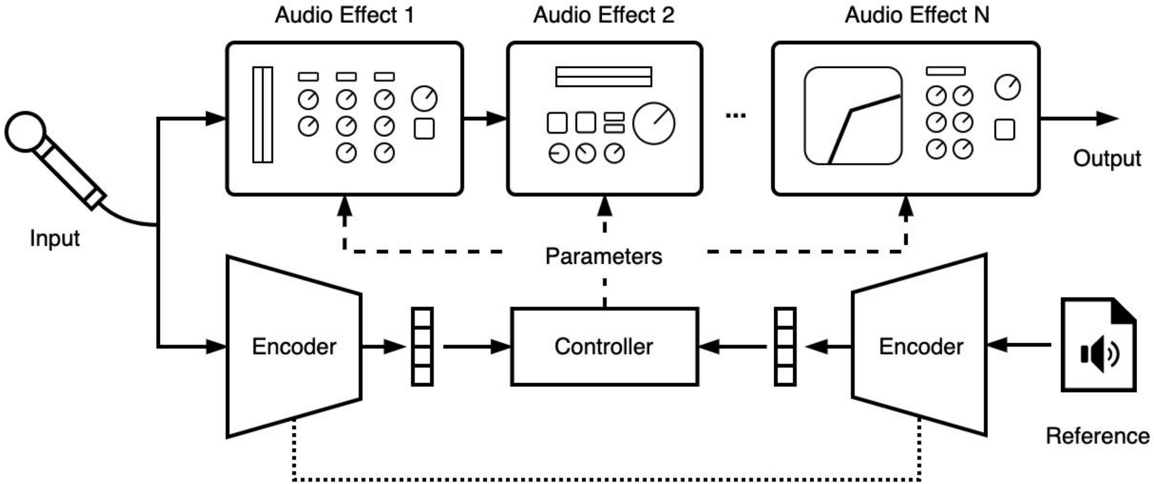
Short-form content

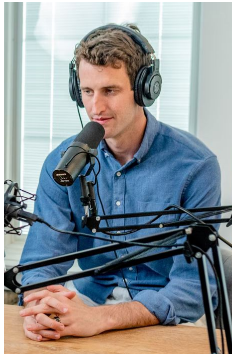
Demand for **high quality audio**

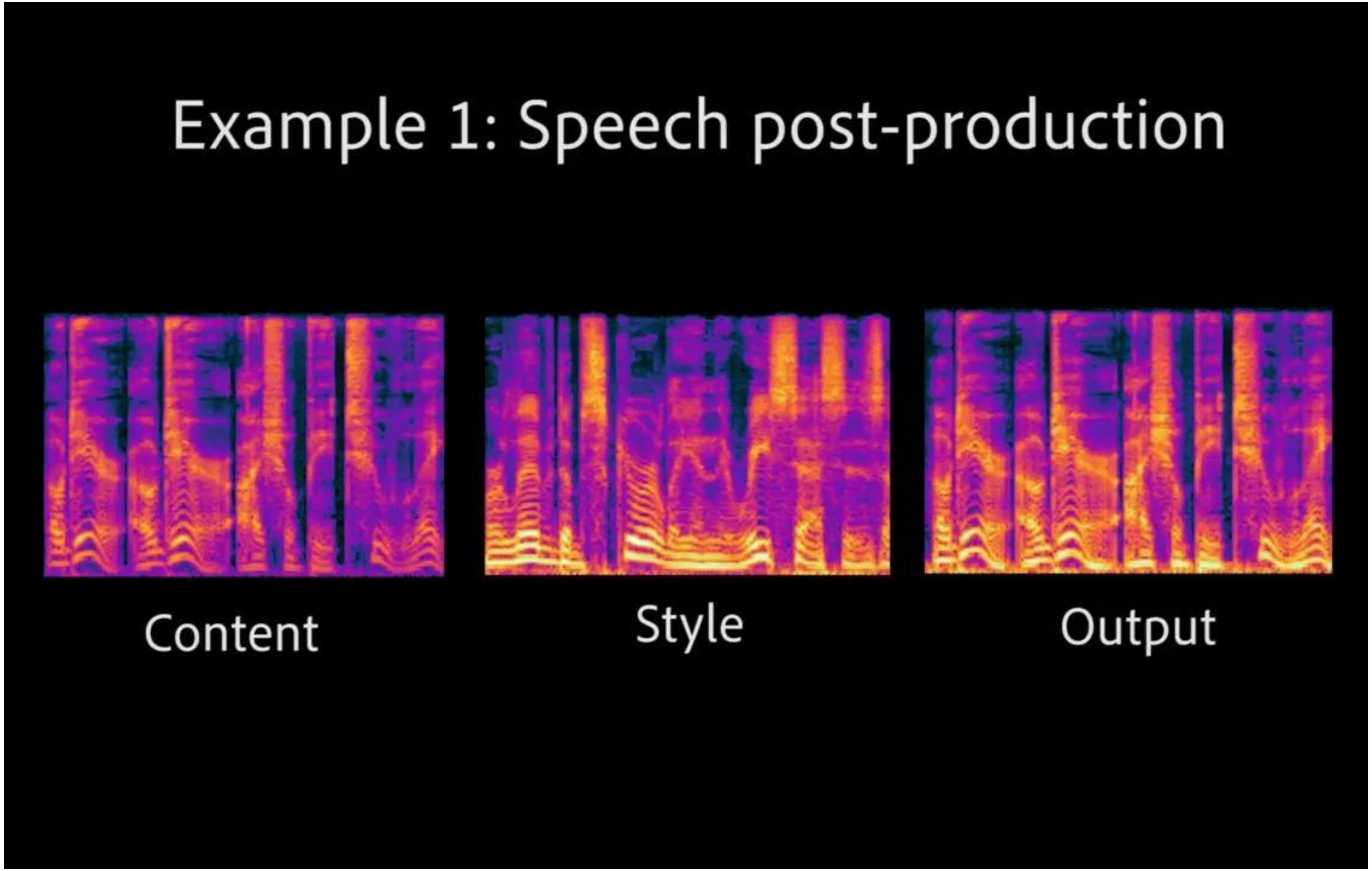
Producing **high quality audio** requires expertise

Style transfer of audio effects

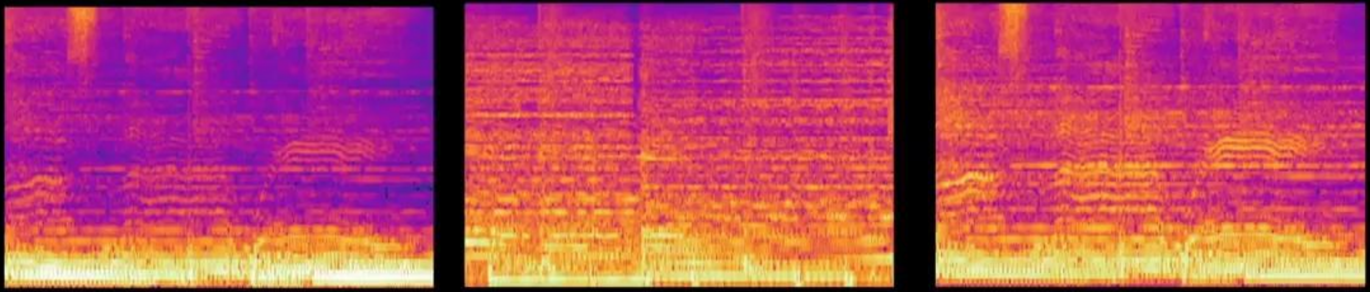







Example 1: Speech post-production
Content
Style
Output

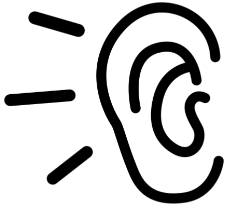
Example 2: Music post-production

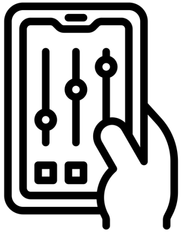


Content　　Style　　Output

Audio production as a three stage process

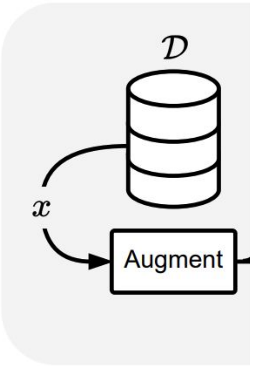
1. **Listen** Perform an acoustic analysis of the input recording
2. **Plan** Establish an acoustic goal (style) considering the context
3. **Execute** Manipulate DSP controls to achieve this goal



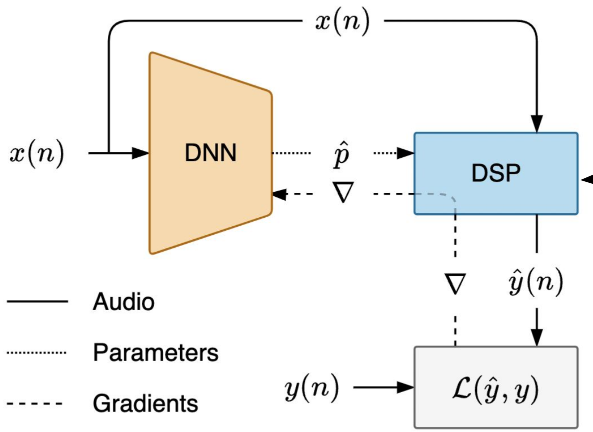


Learning audio production by example

Self-Supervised Data Generation



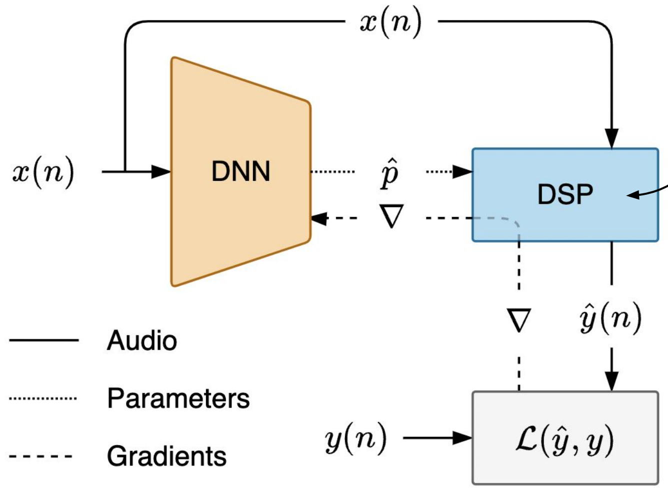
Differentiable signal processing

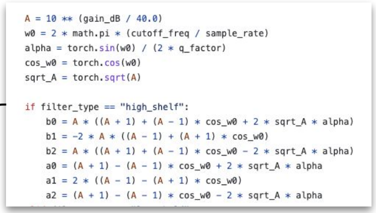


Backprop through DSP operations

- Leveraging existing DSP tools and knowledge
- High quality audio processing with few artifacts
- Human understandable outputs that can be adjusted
- Efficient and can easily run in real-time on CPU

1 Automatic differentiation





```
A = 10 ** (gain_dB / 40.0)
w0 = 2 * math.pi * (cutoff_freq / sample_rate)
alpha = torch.sin(w0) / (2 * q_factor)
cos_w0 = torch.cos(w0)
sqrt_A = torch.sqrt(A)

if filter_type == "high_shelf":
    b0 = A * ((A + 1) + (A - 1) * cos_w0 + 2 * sqrt_A * alpha)
    b1 = -2 * A * ((A - 1) + (A + 1) * cos_w0)
    b2 = A * ((A + 1) + (A - 1) * cos_w0 - 2 * sqrt_A * alpha)
    a0 = (A + 1) - (A - 1) * cos_w0 + 2 * sqrt_A * alpha
    a1 = 2 * ((A - 1) - (A + 1) * cos_w0)
    a2 = (A + 1) - (A - 1) * cos_w0 - 2 * sqrt_A * alpha
```

Explicitly define signal processing operations in autodiff framework

PyTorch TensorFlow

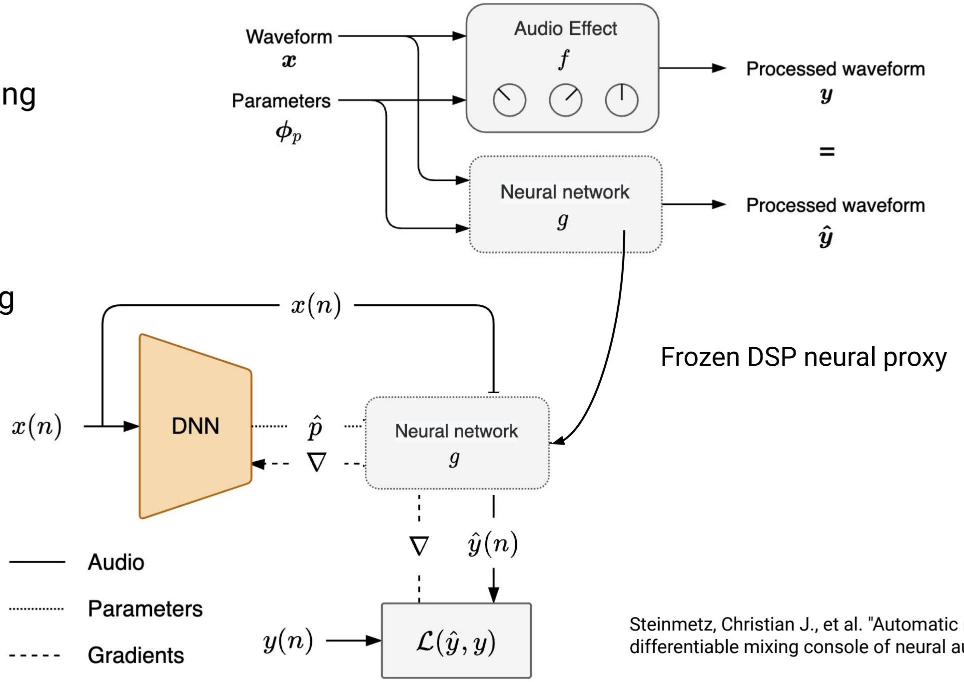
Engel, Jesse, et al. "DDSP: Differentiable digital signal processing." *ICLR* (2021).

2 Neural proxy

(1) Pretraining

(2) Training

(3) Inference

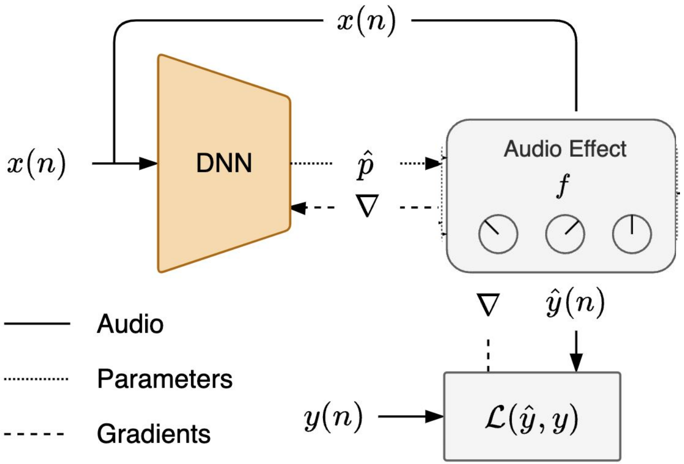


Steinmetz, Christian J., et al. "Automatic multitrack mixing with a differentiable mixing console of neural audio effects." ICASSP, 2021.

3 Neural proxy hybrid

(2) Training

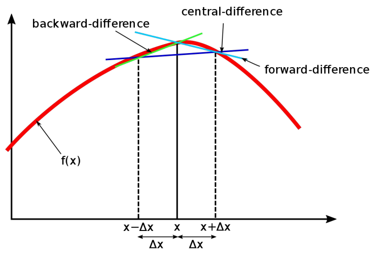
(3) Inference

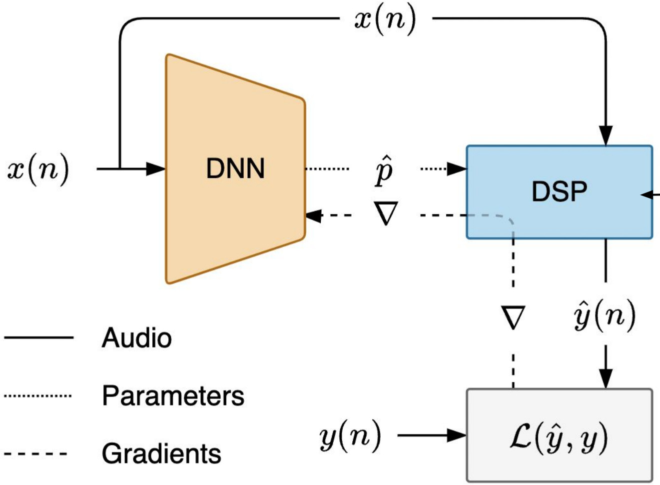


Use original DSP during inference

4 Gradient approximation

Finite differences (FD)





$$\frac{\hat{h}(x, p_i)}{p_i} = \frac{h(x, p + \varepsilon\Delta^P) - h(x, p - \varepsilon\Delta^P)}{2\varepsilon\Delta_i^P}, \qquad (2)$$

where ε is a small, non-zero value and $\Delta^P \in \mathbb{R}^P$ is a random vector sampled from a symmetric Bernoulli distribution ($\Delta_i^P = \pm 1$) [46].

Simultaneous perturbation stochastic approximation (SPSA)

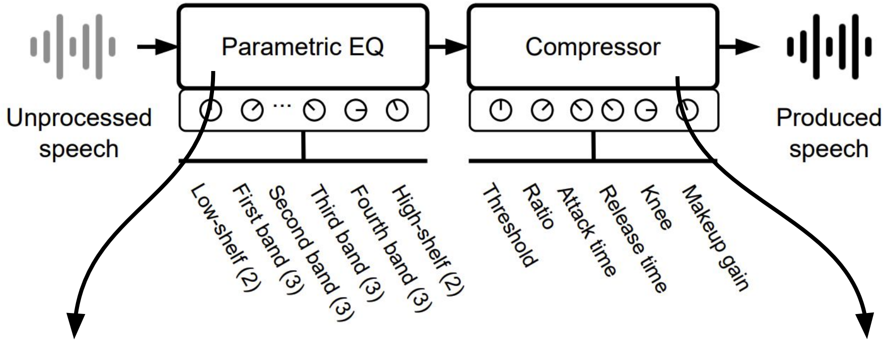
Martínez Ramírez, Marco A., et al. "Differentiable signal processing with black-box audio effects." ICASSP, 2021.

Differentiable signal processing

1. Automatic differentiation
2. Neural proxy
3. Neural proxy hybrid
4. Gradient approximation

No existing comparison of these approaches in a unified setup.

Automatic differentiation audio effects



Nercessian, Shahan. "Neural parametric equalizer matching using differentiable biquads." Proc. Int. Conf. Digital Audio Effects (eDAFx-20). 2020.

, $\mathbf{b}_k = [b_{0,k}, b_{1,k}, b_{2,k}]$ and $\mathbf{a}_k = [a_{0,k}, a_{1,k}, a_{2,k}]$,

$$H_k(e^{j\omega}) = \frac{\mathrm{DFT}(\mathbf{b}_k)}{\mathrm{DFT}(\mathbf{a}_k)} = \frac{\sum_{m=0}^{2} b_{m,k}\, e^{-j\omega m}}{\sum_{n=0}^{2} b_{n,k}\, e^{-j\omega n}}. \tag{1}$$

Estimate IIR filter response with DFT and apply as a frequency domain FIR filter

$$y_L[n] = \begin{cases} \alpha_A y_L[n-1] + (1-\alpha_A)x_L[n] & x_L[n] > y_L[n-1] \\ \alpha_R y_L[n-1] + (1-\alpha_R)x_L[n] & x_L[n] \le y_L[n-1] \end{cases}, \tag{3}$$

$$y_L[n] = \alpha y_L[n-1] + (1-\alpha)x_L[n]. \tag{4}$$

This can be approximated with a FIR (frequency domain) filter

Training details

Models

RB-DSP	Rule-based DSP
cTCN	Conditional TCN
NP	Neural Proxy
NP-HH	Neural Proxy Half-hybrid
NP-FH	Neural Proxy Full-hybrid
SPSA	Gradient approximation
AD	Automatic differentiation

Audio domain loss
Multi-resolution STFT

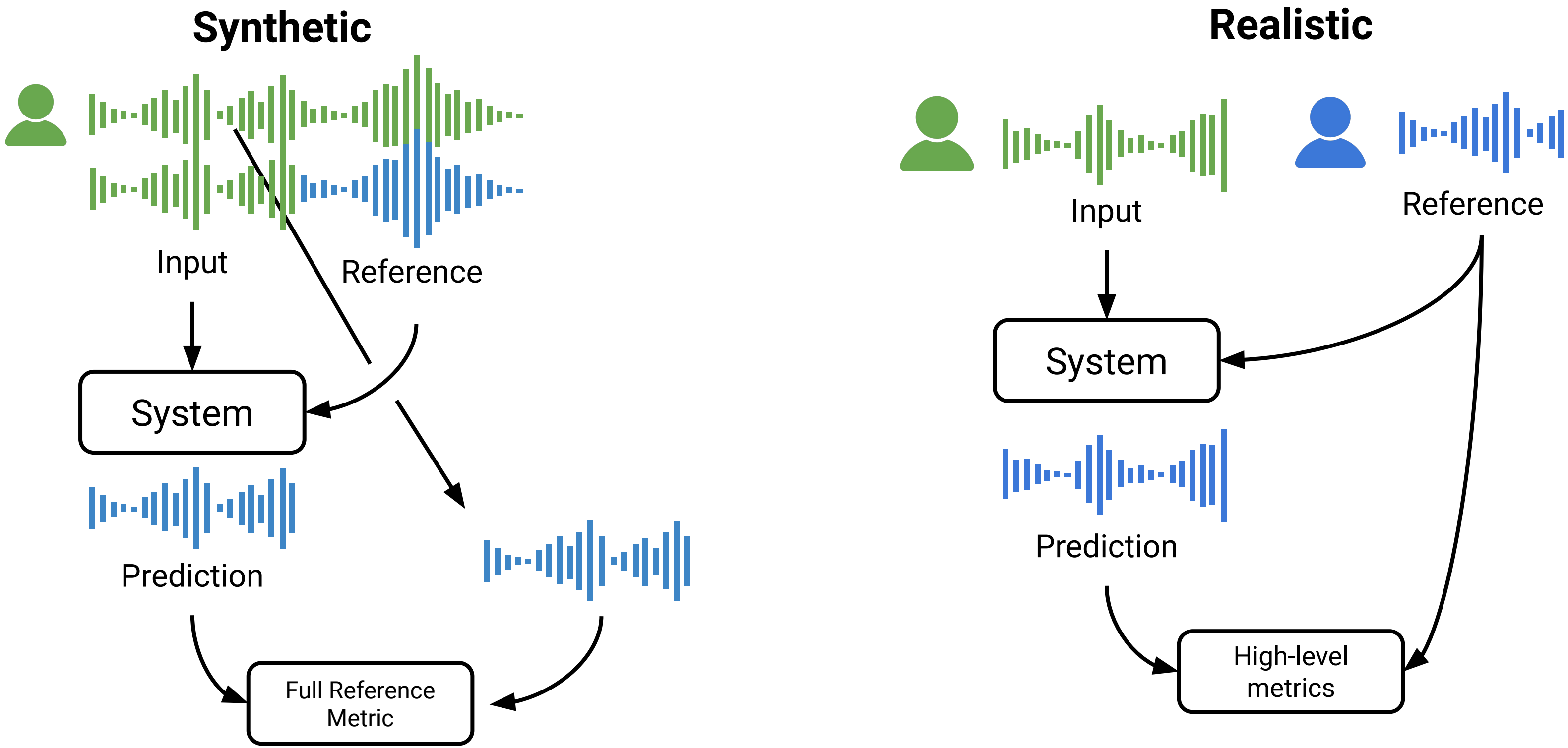
Training Datasets
Speech (LibriTTS)
Music (MTG-Jamendo)

Effects
6-band parametric EQ
Dynamic range compressor

Experiments

1. **Synthetic production style transfer**
   (matching input and reference)
2. **Realistic production style transfer**
   (non-matching input and reference)
3. **Audio production representations**
   (audio production style classification)
4. **Computational complexity**

Audio production style transfer



Evaluation metrics

General similarity (full reference)	**PESQ**	Perceptual evaluation of speech quality
	STFT	Multi-resolution STFT error
Spectral balance (EQ) (high-level features)	**MSD**	Large window log-mel spectrogram error
	SCE	Spectral centroid error
Dynamics (Compression) (high-level features)	**RMS**	Root mean square energy error
	LUFS	Perceptual loudness error

Synthetic audio production style transfer

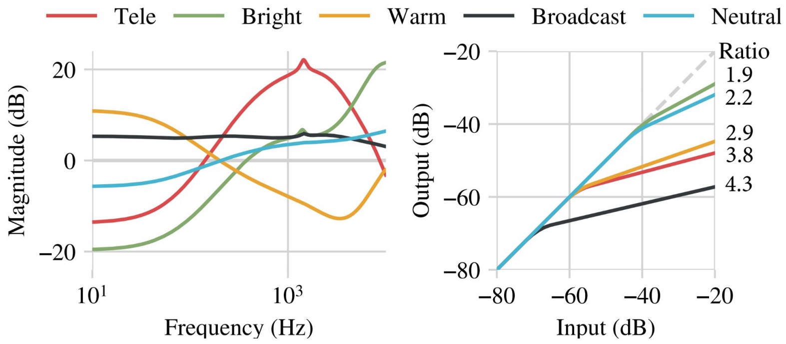
out-of-domain datasets

	Speech																	
	LibriTTS						DAPS						VCTK					
Method	**PESQ**	**STFT**	**MSD**	**SCE**	**RMS**	**LUFS**	**PESQ**	**STFT**	**MSD**	**SCE**	**RMS**	**LUFS**	**PESQ**	**STFT**	**MSD**	**SCE**	**RMS**	**LUFS**
Input	3.765	1.187	2.180	687.5	6.983	2.426	3.684	1.179	2.151	641.7	6.900	2.314	3.672	1.254	2.008	815.4	7.783	2.532
RB-DSP	3.856	0.943	1.955	410.3	4.204	1.674	3.787	0.917	1.882	399.7	3.705	1.481	3.709	1.101	1.911	657.6	5.039	2.018
cTCN 1	4.258	0.405	0.887	128.4	2.237	1.066	4.185	0.419	0.884	124.6	2.098	1.006	4.181	0.467	0.891	173.8	2.651	1.165
cTCN 2	4.281	0.372	**0.833**	117.5	1.927	0.925	**4.224**	**0.391**	**0.841**	113.9	1.886	0.913	4.201	**0.441**	**0.856**	163.8	2.431	1.086
NP	3.643	0.676	1.405	265.0	2.812	1.340	3.605	0.685	1.362	249.2	2.732	1.350	3.651	0.737	1.300	321.7	3.166	1.453
NP-HH	3.999	1.038	2.179	440.2	5.472	2.679	3.903	1.022	2.113	451.9	5.104	2.535	3.951	1.044	1.930	591.5	5.194	2.651
NP-FH	3.945	1.058	2.088	404.9	6.820	3.197	3.891	1.037	2.045	395.4	6.754	3.117	3.894	1.087	1.934	514.0	7.065	3.363
SPSA	4.180	0.635	1.406	219.5	3.263	1.600	4.099	0.645	1.379	213.6	2.989	1.511	4.023	0.730	1.359	301.6	3.535	1.737
AD	**4.310**	**0.388**	0.882	**111.5**	**1.828**	**0.823**	4.222	0.416	0.895	**109.0**	**1.758**	**0.799**	**4.218**	0.481	0.924	**152.7**	**2.317**	**1.006**

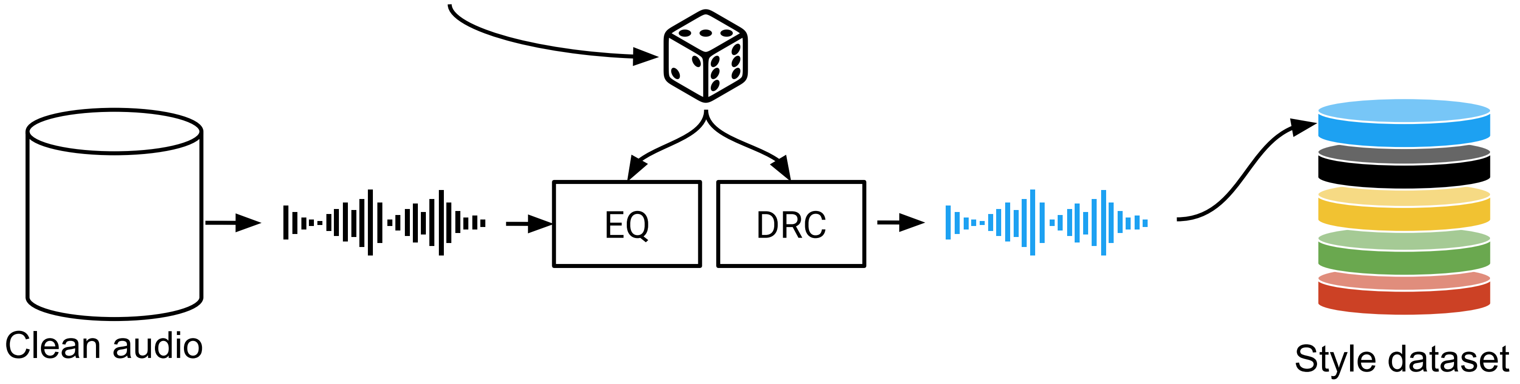
Table 1. Synthetic production style transfer with models trained using LibriTTS. Held-out speakers from the LibriTTS dataset are used, while utterances from DAPS and VCTK come from datasets never seen during training. Lower is better for all metrics except PESQ.

Production style generation

For evaluating realistic style transfer



Styles are defined by distributions in the parameter space of the parametric EQ and dynamic range compressor.

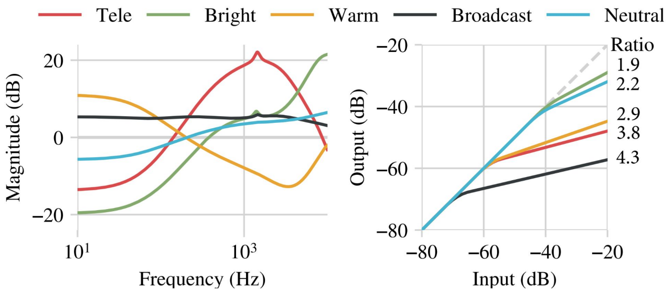


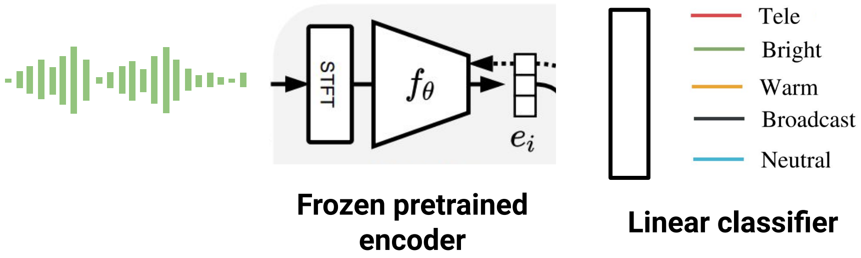
Realistic audio production style transfer

	DAPS				MUSDB18			
Method	**MSD**	**SCE**	**RMS**	**LUFS**	**MSD**	**SCE**	**RMS**	**LUFS**
Input	10.4	1041.7	10.4	3.6	8.4	2607.4	9.4	3.8
RB-DSP	8.9	517.2	5.8	2.4	6.5	**915.4**	8.6	3.7
NP-HH S	9.8	636.6	12.9	5.9	7.0	1512.7	10.0	4.5
SPSA	8.0	360.1	**5.1**	2.5	5.5	1297.0	4.6	2.1
AD	**7.8**	**278.1**	5.2	**2.4**	**4.8**	947.6	**3.8**	**1.7**

Table 3. Realistic production style transfer average performance of all pairwise configurations from five predefined styles with speech from DAPS using the model trained on LibriTSS and music from MUSDB18 using the model trained on MTG-Jamendo.

Learning audio production representations





DAPS (Speech)						
Features	Telephone	Bright	Warm	Broadcast	Neutral	**Avg**
Random Mel	0.87	0.78	0.73	0.39	0.00	0.55
OpenL3	0.19	0.61	0.08	0.10	0.18	0.23
CDPAM	**1.00**	**1.00**	0.79	0.25	0.63	0.73
NP-HH S	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
SPSA	0.95	0.98	**1.00**	0.89	0.95	0.96
AD	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
MUSDB18 (Music)						
Features	Telephone	Bright	Warm	Broadcast	Neutral	**Avg**
Random Mel	0.80	0.98	0.62	0.17	0.00	0.51
OpenL3	0.32	0.66	0.20	0.17	0.30	0.33
CDPAM	0.89	0.95	0.66	0.00	0.06	0.51
NP-HH S	**0.98**	**1.00**	0.92	**0.59**	**0.60**	**0.82**
SPSA	**0.98**	**1.00**	0.90	0.26	0.00	0.63
AD	**0.98**	**1.00**	**0.95**	0.54	0.50	0.79

Table 4. Class-wise F1 scores for five-class style prediction with linear classifiers trained on top of audio representations for speech and music using a single linear layer.

Computational complexity

Method	Train step (s)	CPU	GPU	Parameters	Interpretable
RB-DSP	-	0.004	-	0	–
cTCN 1	0.438	0.132	0.002	174 k	-
cTCN 2	0.642	0.268	0.005	336 k	-
NP	0.434	0.277	0.005	336 k	✓
NP-HH	0.434	**0.003**	-	0	✓
NP-FH	0.434	**0.003**	-	0	✓
SPSA	0.413	**0.003**	-	0	✓
AD	**0.301**	0.006	**0.001**	0	✓

Table 5. Runtime comparison across differentiation methods including seconds taken for a single training step, and real-time factor for inference on CPU (Intel Xeon CPU E5-2623 v3 @ 3.00GHz) and GPU (GeForce GTX 1080 Ti).

Differentiation approaches performance

1. **Rule-based DSP baseline** outperformed by learned approaches
2. **Neural proxy hybrid** approaches do not perform well
3. **Gradient approximation** performs second best but struggles with instability
4. **Automatic differentiation** performs best overall but is only an approximation of effects

Contributions

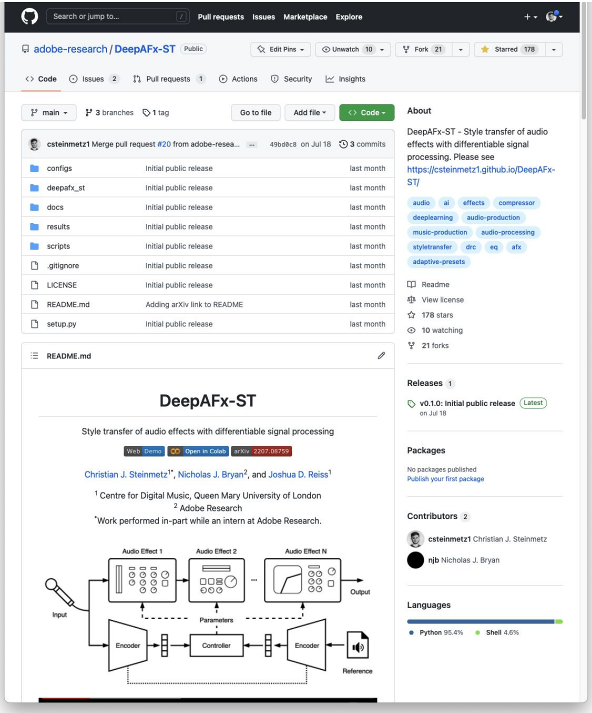
1. The first audio effects style transfer method to integrate audio effects as differentiable operators, optimized end-to-end with an audio-domain loss

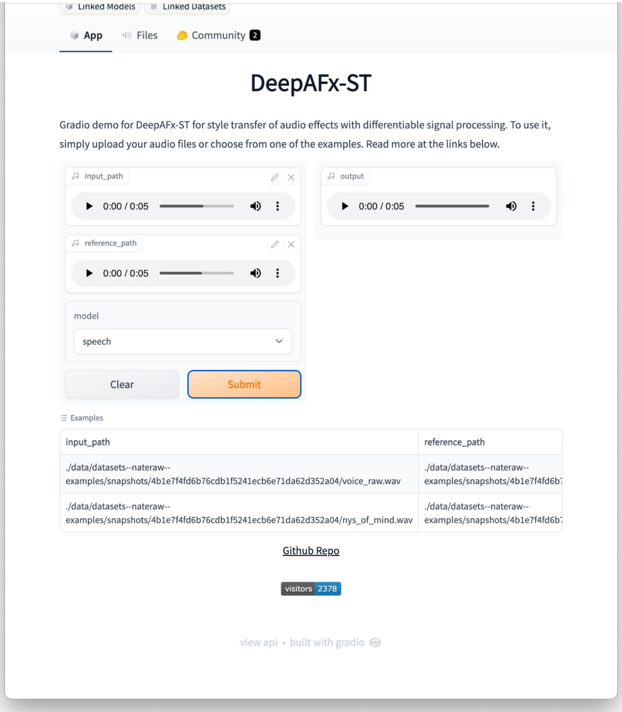
2. Self-supervised training that enables automatic audio production without labeled or paired training data

3. A benchmark of five differentiation strategies for audio effects, including compute cost, engineering difficulty, and performance

4. The development of novel neural proxy hybrid methods, and a differentiable dynamic range compressor.

Resources





github.com/adobe-research/DeepAFx-ST

huggingface.co/spaces/nateraw/deepafx-st

Future directions

1. Extend this approach with more differentiable effects (e.g. reverb, distortion, etc)
2. Improved methods for training neural proxy (hybrids)
3. Methods for handling dynamic construction of the processing chain
4. Adapt this approach for multichannel use cases (e.g. multitrack mixing)
5. Zero-shot adaptation to a new set of audio effects (can I use the plugins in my DAW?)

Sonos Seminar Series • 1 September 2022

Style transfer of audio effects with differentiable signal processing

Christian J. Steinmetz[1,2]
c.j.steinmetz@qmul.ac.uk

Nick J. Bryan[2]

Joshua D. Reiss[1]

[1]Queen Mary University of London
[2]Adobe Research

arxiv.org/abs/2207.08759

Adobe · Queen Mary University of London · UK Research and Innovation · AIM AI + MUSIC