DiffVox:

A Differentiable Model for Capturing and Analysing Vocal Effects Distributions

Chin-Yun Yu[1], Marco A. Martínez-Ramírez[2], Junghyun Koo[2], Ben Hayes[1], Wei-Hsiang Liao[2], György Fazekas[1], Yuki Mitsufuji[2]

[1]Queen Mary University of London, UK
[2]Sony AI & Sony Group Coorporation, Japan

Context & Motivation

Sampling audio effect parameters $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$

- Data-driven tasks
  - training effects detector
  - training mixing style transfer systems
  - pretraining audio representations
- Can we have something other than Uniform or Gaussian?
  - Different sampling strategies affect generalisation in neural-network-based filter design*
- Building an effects prior $p(\boldsymbol{\theta}) \rightarrow$ building an effects preset dataset!

* Joseph T Colonel, Christian J Steinmetz, Marcus Michelen, and Joshua D Reiss, “Direct design of biquad filter cascades with deep learning by sampling random polynomials,” in IEEE ICASSP, 2022, pp. 3104–3108.

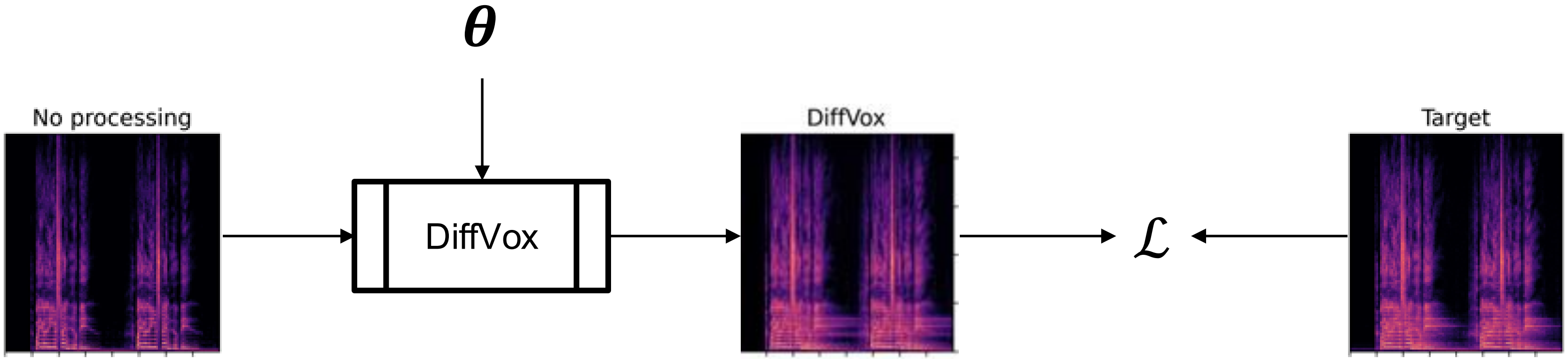
Contributions

- A public dataset with 435 vocal presets
- Efficient differentiable implementation of common audio effects
  - EQ, Compressor, Delay, Reverb
- Multi-scale Loudness Dynamic Range (MLDR) loss and efficient implementation
- A PCA model trained on the dataset

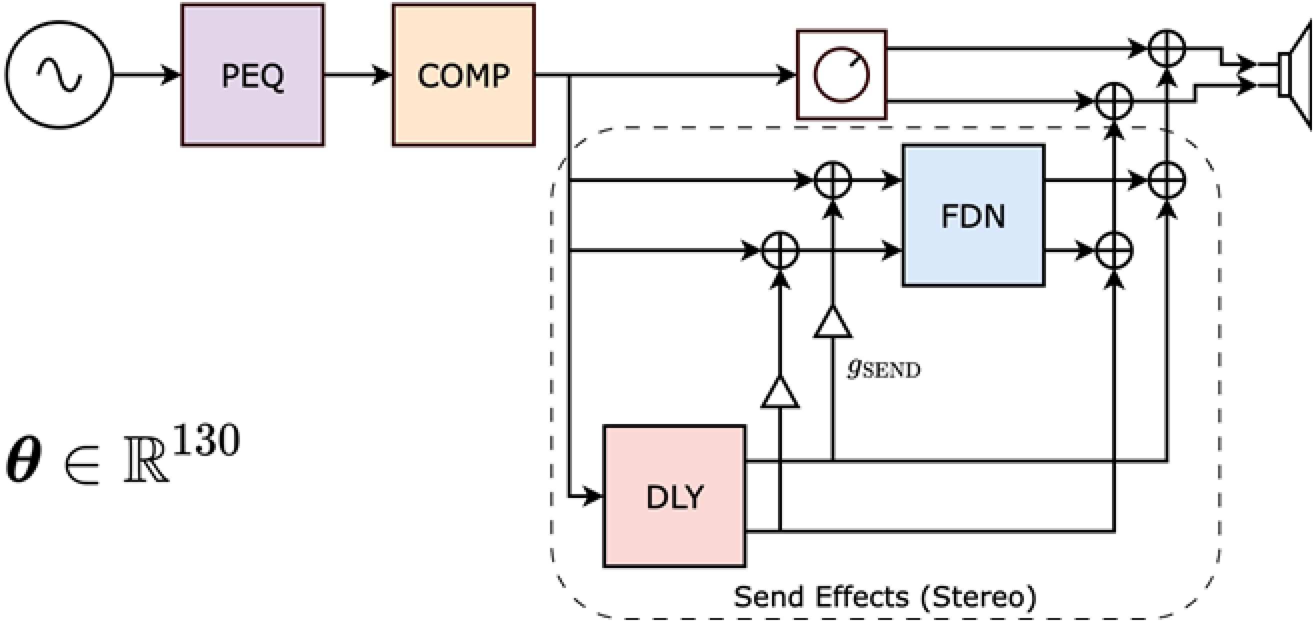
Methodology

How do we retrieve $\boldsymbol{\theta}$?

- **Sound matching** given paired dry and wet mixes
- Finding minimum solution of $\mathcal{L}$ using **gradient descent**
- 365 vocal tracks picked from proprietary western music multi-tracks



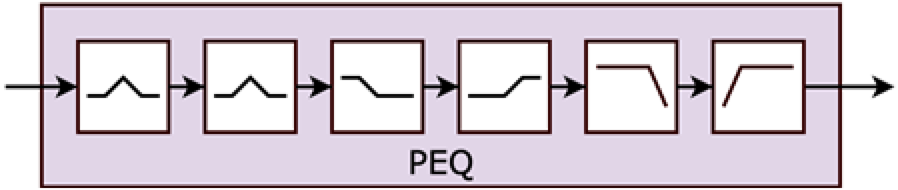
Differentiable **Vo**cal F**x** (DiffVox)



$$\boldsymbol{\theta} \in \mathbb{R}^{130}$$

Six-band parametric equaliser (PEQ)

- Biquad filters
- two peak filters, low-shelf, high-shelf, low-pass, high-pass
- Problem: recursive filters are slow on GPU



Filter acceleration on GPU

- State-space realisation

$$\boxed{\tilde{\mathbf{x}}[n+1] = \mathbf{A}\tilde{\mathbf{x}}[n] + \begin{bmatrix} x[n] \\ 0 \end{bmatrix}}$$

$$y[n] = \mathbf{C}\tilde{\mathbf{x}}[n] + b_0 x[n]$$

$$\mathbf{A} = \begin{bmatrix} -a_1 & -a_2 \\ 1 & 0 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} b_1 - b_0 a_1 & b_2 - b_0 a_2 \end{bmatrix}$$

Parallel associative scan

non-parallelisable

$$\tilde{\mathbf{x}}[n+1] = \mathbf{A}\left(\mathbf{A}\left(\mathbf{A}\left(\ldots\right) + \mathbf{x}[n-2]\right) + \mathbf{x}[n-1]\right) + \mathbf{x}[n]$$

parallelisable

$$(*, \tilde{\mathbf{x}}[n+1]) = (\mathbf{A}, \mathbf{x}[0]) \oplus (\mathbf{A}, \mathbf{x}[1]) \oplus \cdots \oplus (\mathbf{A}, \mathbf{x}[n])$$

$$(\mathbf{U}, \mathbf{x}) \oplus (\mathbf{V}, \mathbf{z}) \mapsto (\mathbf{VU}, \mathbf{Vx} + \mathbf{z})$$

Guy E. Blelloch, “Prefix sums and their applications,” Tech. Rep. CMU-CS-90-190, School of Computer Science, Carnegie Mellon University, Nov. 1990.

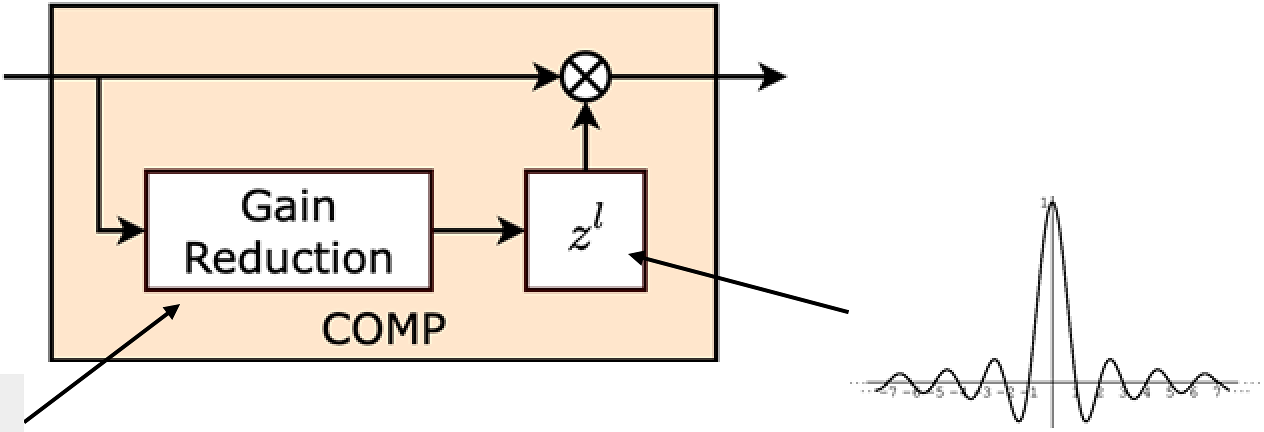
Diagonalised state-space

$$\mathbf{A} = \mathbf{P}\begin{bmatrix}\lambda_1 & 0 \\ 0 & \lambda_2\end{bmatrix}\mathbf{P}^{-1}, \quad \lambda_1 \neq \lambda_2$$

$$\boxed{\mathbf{P}^{-1}\tilde{\mathbf{x}}[n+1] = \begin{bmatrix}\lambda_1 & 0 \\ 0 & \lambda_2\end{bmatrix}\mathbf{P}^{-1}\tilde{\mathbf{x}}[n] + \mathbf{P}^{-1}\begin{bmatrix}x[n] \\ 0\end{bmatrix}}$$

scalar multiplication

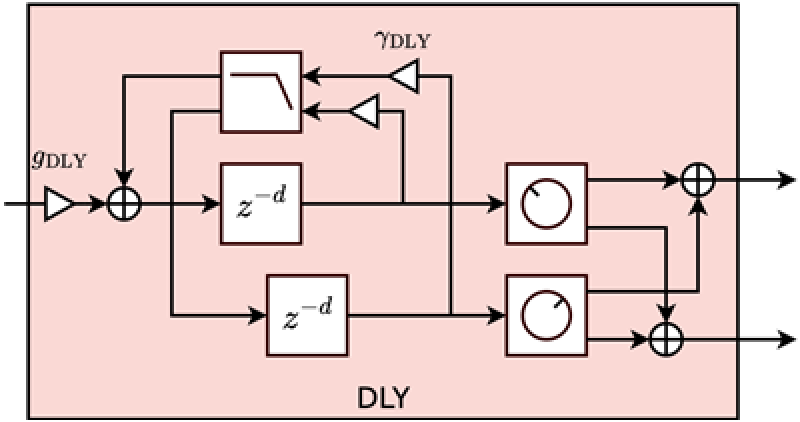
Feed-Forward Compressor and Expander (COMP)




```
pip install torchcomp
```

Chin-Yun Yu, Christopher Mitcheltree, Alistair Carson, Stefan Bilbao, Joshua D. Reiss, and György Fazekas, “Differentiable all-pole filters for time-varying audio systems,” in DAFx, 2024, pp. 345–352.

Ping-pong delay



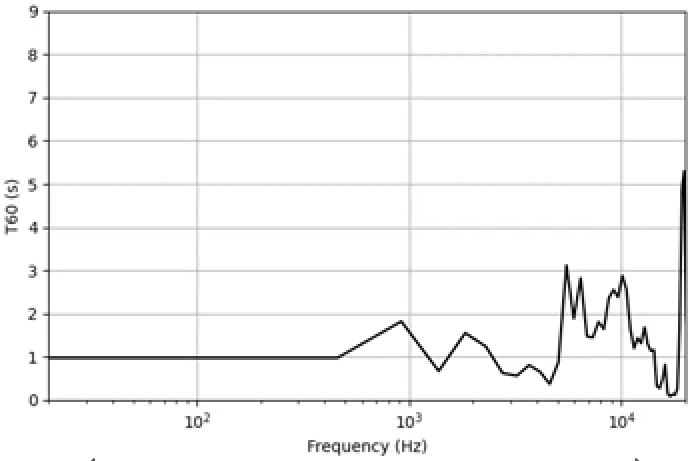
$$H_{\mathrm{odd}}(z) = \frac{z^{-d}}{1 - \gamma_{\mathrm{DLY}} H_{\mathrm{LP}}(z) z^{-2d}}$$

$$\approx \gamma_{\mathrm{DLY}}^{-1} \sum_{k=1}^{\left\lfloor \frac{N_{\mathrm{DLY}}-d}{2d} \right\rfloor} \left(\gamma_{\mathrm{DLY}} H_{\mathrm{LP}}(z) \eta^{\frac{\angle z}{2\pi} N_{\mathrm{DLY}}} z^{-d} \right)^{2k-1}$$

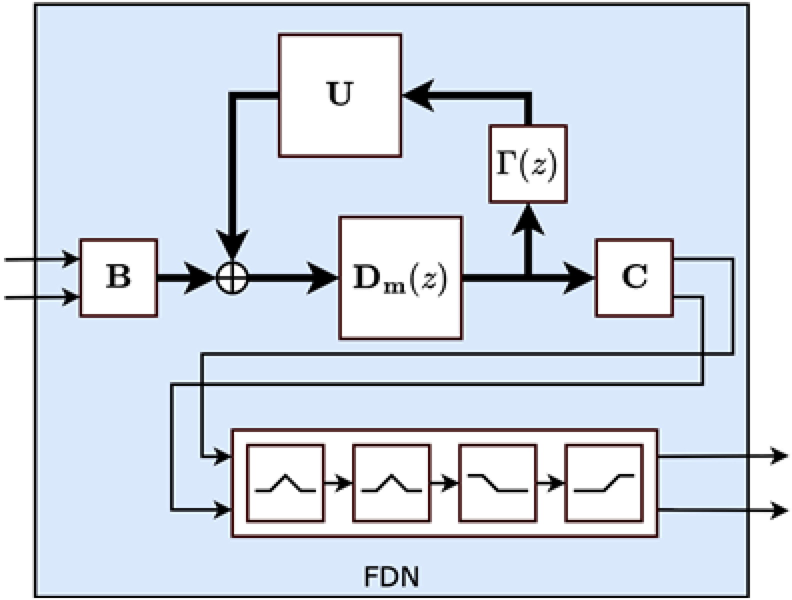
N_{DLY} = FIR truncation length, $0 \leq \eta \leq 1$

Feedback delay network reverb (FDN)

$\boldsymbol{\Gamma}(e^{j\omega})$



49 points from 0 to π



Optimisation

- Losses
  - Multi-scale STFT (MSS)
  - Multi-scale loudness dynamic range (MLDR)
    - Nercessian et al. "A direct microdynamics adjusting processor with matching paradigm and differentiable implementation." DAFx. 2022.
  - Square error $(1 - \eta)^2$
- Preprocessing
  - Normalised to 18 -dB LUFS
  - Split songs into 12-second chunks
- 20 ~ 40 minutes on RTX 3090 per song
- 70 vocal tracks from MedleyDB for comparison

Results

Listening samples

No effects

Target

DiffVox

Sound matching performance

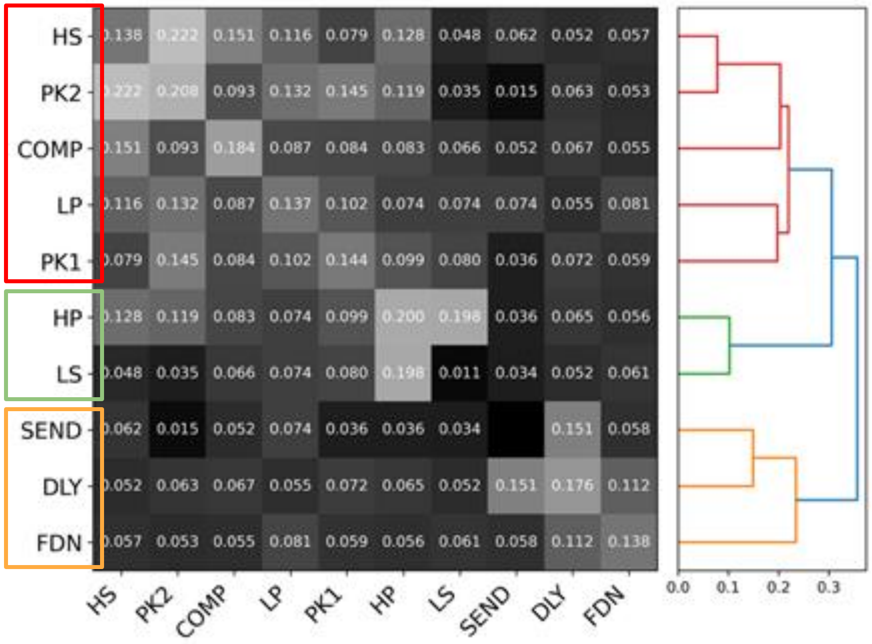
Dataset	Configuration	MSS ↓ l/r	MSS ↓ m/s	MLDR ↓ l/r	MLDR ↓ m/s
Internal	No processing	1.44	2.39	1.82	2.08
	DiffVox	**0.76**	**0.94**	**0.34**	**0.41**
	⊢ w/o Approx.	0.78	0.95	0.38	0.44
MedleyDB	No processing	1.27	2.16	1.00	1.35
	DiffVox	0.75	0.98	**0.39**	**0.45**
	⊢ w/o Approx.	0.77	1.00	0.42	0.48
	w/o FDN	0.79	1.14	0.48	0.62
	w/o DLY	0.76	0.99	0.40	0.47
	w/o DLY&FDN	**0.61**	**0.90**	0.82	1.17

Parameter correlation analysis

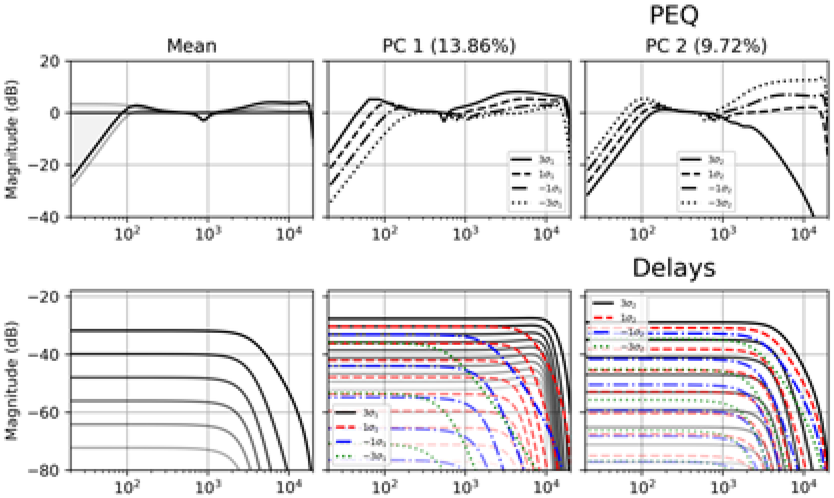
Parameter-wise

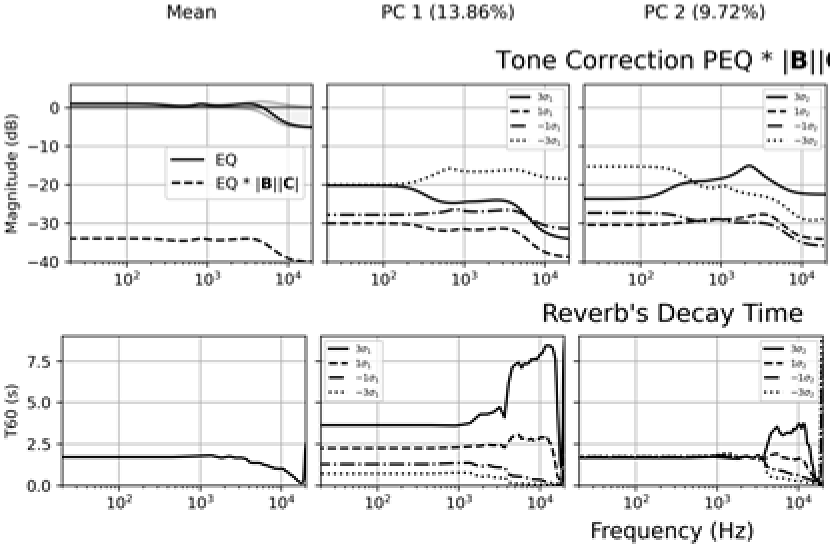
Parameter 1	Parameter 2	SCC Internal	SCC MedleyDB
f_{LP}	$\gamma(e^{i\frac{44}{48}\pi})$	0.60	0.32
g_{PK2}	Q_{PK2}	-0.60	-0.10
d	γ_{DLY}	-0.58	-0.20
f_{LP}	$\gamma(e^{i\frac{43}{48}\pi})$	0.56	0.35
CT	make-up	-0.55	0.06
f_{LP}	$\gamma(e^{i\frac{45}{48}\pi})$	0.53	0.19
ET	ER	-0.52	-0.30
d	g_{DLY}	-0.51	-0.02
γ_{DLY}	$f_{\mathrm{DLY.LP}}$	0.49	0.41
$g_{\mathrm{FDN.PK2}}$	$f_{\mathrm{FDN.PK2}}$	-0.46	-0.47

Effect-wise



Principal component analysis





Conclusions

- **Spatial effects are crucial** for achieving good matching performance
- Correlation and principal component analysis match some common beliefs
    - compressor threshold ↔ make-up gain
    - higher gain → smaller Q
    - PC 1: reverberation
    - PC 2: brightness / spectrum shape
    - etc.
- The PCA weights are not gaussian-distributed

Future works

1. Improving effects implementations
   - Convert cascaded biquads to **parallel biquads**
   - Introduce soft-bypass design (dry/wet ratio)
   - Efficient time-domain differentiable FDN
   - Automation (parameter-varying)
   - Multi-track mixes
   - .etc
2. Improving analysis
   - Non-linear dimension reduction models
   - Weighted PCA analysis
3. Preset generation

DAFx25 Tutorial 2, Balázs Bank

Q&A

Sound samples

Code

Demo