Approximately Linear Audio Consistency Autoencoders

Learning linearity via implicit regularization

Bernardo Torres · Manuel Moussallam* · Gabriel Meseguer-Brocal*

LTCI, Télécom Paris, Institut Polytechnique de Paris · *Deezer Research

Audio processing in the latent space

- Perform downstream tasks on compressed representations for compute efficiency

The optimization trap: compression vs. structure

- AEs trained to **maximize compression** and **minimize reconstruction error**
- Latent space has **no structure**

Audio processing in the latent space

- Perform downstream tasks on compressed representations for compute efficiency

The optimization trap: compression vs. structure

- AEs trained to **maximize compression** and **minimize reconstruction error**
- Latent space has **no structure**

Cheap fix: train a task specific module

- Bandwidth extension, upmixing (mono-to-stereo) [*Bralios et al. WASPAA 2025*]
- Source separation [*Bindi et al. ICASSP 2023*] [*Li et al. ICASSP 2025*] [*Omran et al. ISMIR 2024*]
- Re-bottleneck: ordered channels, semantic alignment, low-pass filtering [*Bralios et al. WASPAA 2025*]

Audio processing in the latent space

- Perform downstream tasks on compressed representations for compute efficiency

The optimization trap: compression vs. structure

- AEs trained to **maximize compression** and **minimize reconstruction error**
- Latent space has **no structure**

Cheap fix: train a task specific module

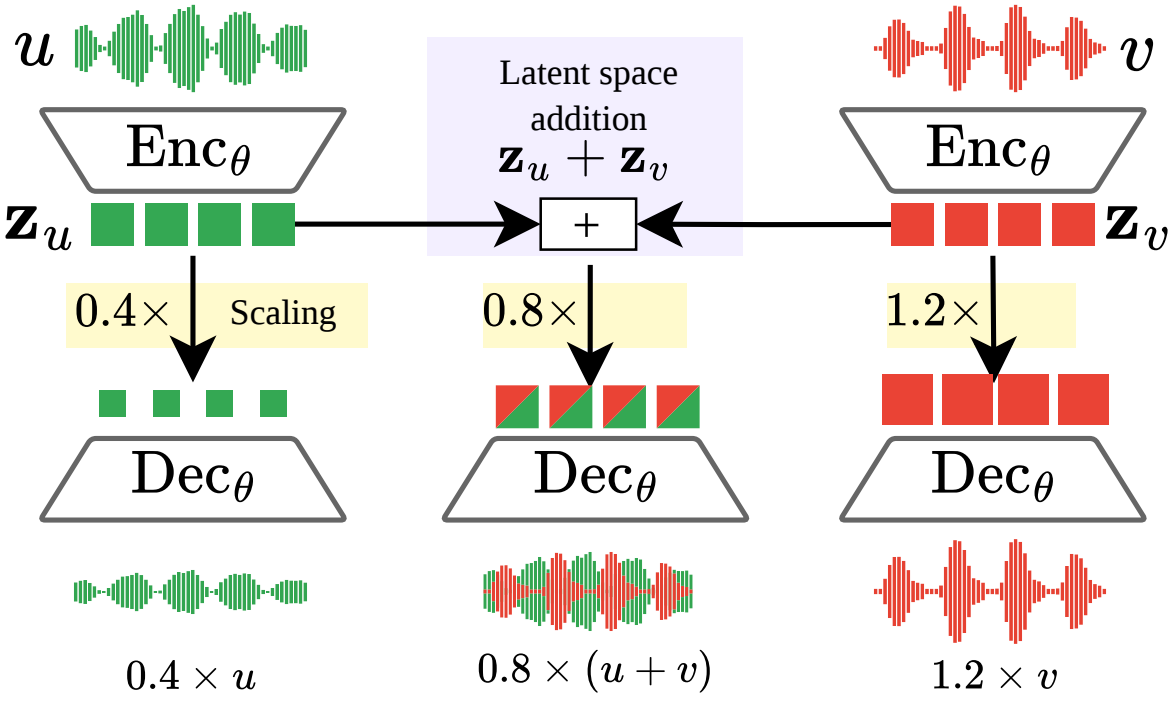
- Bandwidth extension, upmixing (mono-to-stereo) [*Bralios et al. WASPAA 2025*]
- Source separation [*Bindi et al. ICASSP 2023*] [*Li et al. ICASSP 2025*] [*Omran et al. ISMIR 2024*]
- Re-bottleneck: ordered channels, semantic alignment, low-pass filtering [*Bralios et al. WASPAA 2025*]

Expensive fix: retrain the autoencoder and enforce desired property

- Source disentanglement [*Bie et al. ICASSP 2025*]

This work: **linearity!**

Linear autoencoders



How can we "learn linearity" in a high-compression, high-fidelity autoencoder?

Definition: approximately linear latent space

Property 1 - Homogeneity

Decoder

$$\mathrm{Dec}_\theta(a \cdot \boldsymbol{z}_x) \approx a \cdot \mathrm{Dec}_\theta(\boldsymbol{z}_x)$$

Encoder

$$\mathrm{Enc}_\theta(a \cdot \boldsymbol{x}) \approx a \cdot \mathrm{Enc}_\theta(\boldsymbol{x})$$

Property 2 - Additivity

Decoder

$$\mathrm{Dec}_\theta(\boldsymbol{z}_u + \boldsymbol{z}_v) \approx \mathrm{Dec}_\theta(\boldsymbol{z}_u) + \mathrm{Dec}_\theta(\boldsymbol{z}_v)$$

Encoder

$$\mathrm{Enc}_\theta(\boldsymbol{u} + \boldsymbol{v}) \approx \mathrm{Enc}_\theta(\boldsymbol{u}) + \mathrm{Enc}_\theta(\boldsymbol{v})$$

Proposed method: train a linear decoder with only data augmentation

Algorithm (intuition)

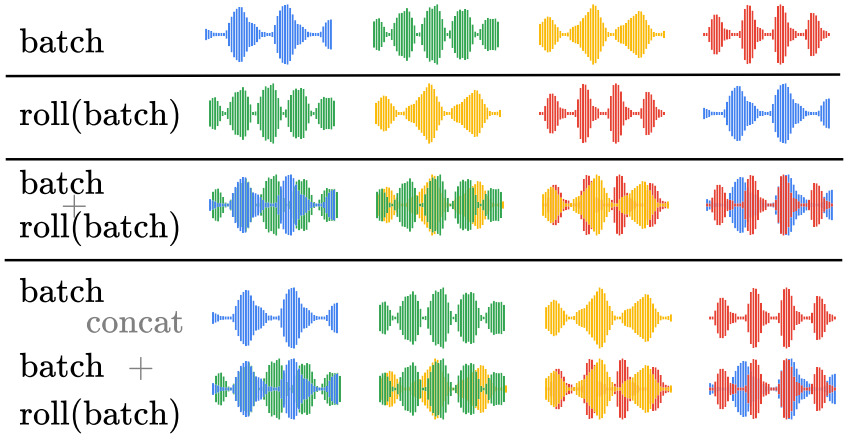
Require: Dataset $\mathcal{D}$, Encoder E, Decoder D

1: **for** each training step **do**
2: Sample two audio signals $\boldsymbol{u}, \boldsymbol{v} \sim \mathcal{D}$
4: Get individual latents: $\boldsymbol{z}_u \leftarrow \mathrm{Enc}(\boldsymbol{u})$, $\boldsymbol{z}_v \leftarrow \mathrm{Enc}(\boldsymbol{v})$
5: **Sum latents:** $\boldsymbol{z}' \leftarrow \boldsymbol{z}_u + \boldsymbol{z}_v$
6: **Sample gain:** $\boldsymbol{a} \sim \mathcal{U}(a_{\min}, a_{\max})$
7: Decode scaled mix: $\hat{\boldsymbol{x}} \leftarrow \mathrm{Dec}(\boldsymbol{a} \cdot \boldsymbol{z}')$
3: Get target by mixing in audio space: $\boldsymbol{x} \leftarrow \boldsymbol{a} \cdot (\boldsymbol{u} + \boldsymbol{v})$
8: Loss: $\mathcal{L}(\hat{\boldsymbol{x}}, \boldsymbol{x})$
9: **end for**

Decode a **sum of latents** scaled by a **gain** and reconstruct the **scaled mix in the input space**.

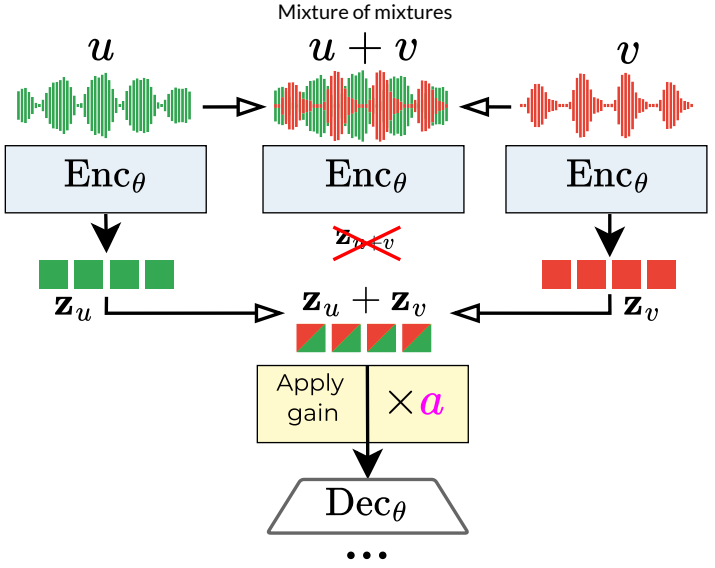
Implicitly enforcing additivity using random mixes on the batch

(1) Create artificial mixes from random training samples efficiently in the batch.



†Half of the batch are artificial mixes, the other half are regular samples.

(2) Decode the sum of latents when input is an artificial mix.



Experiments on diffusion autoencoders

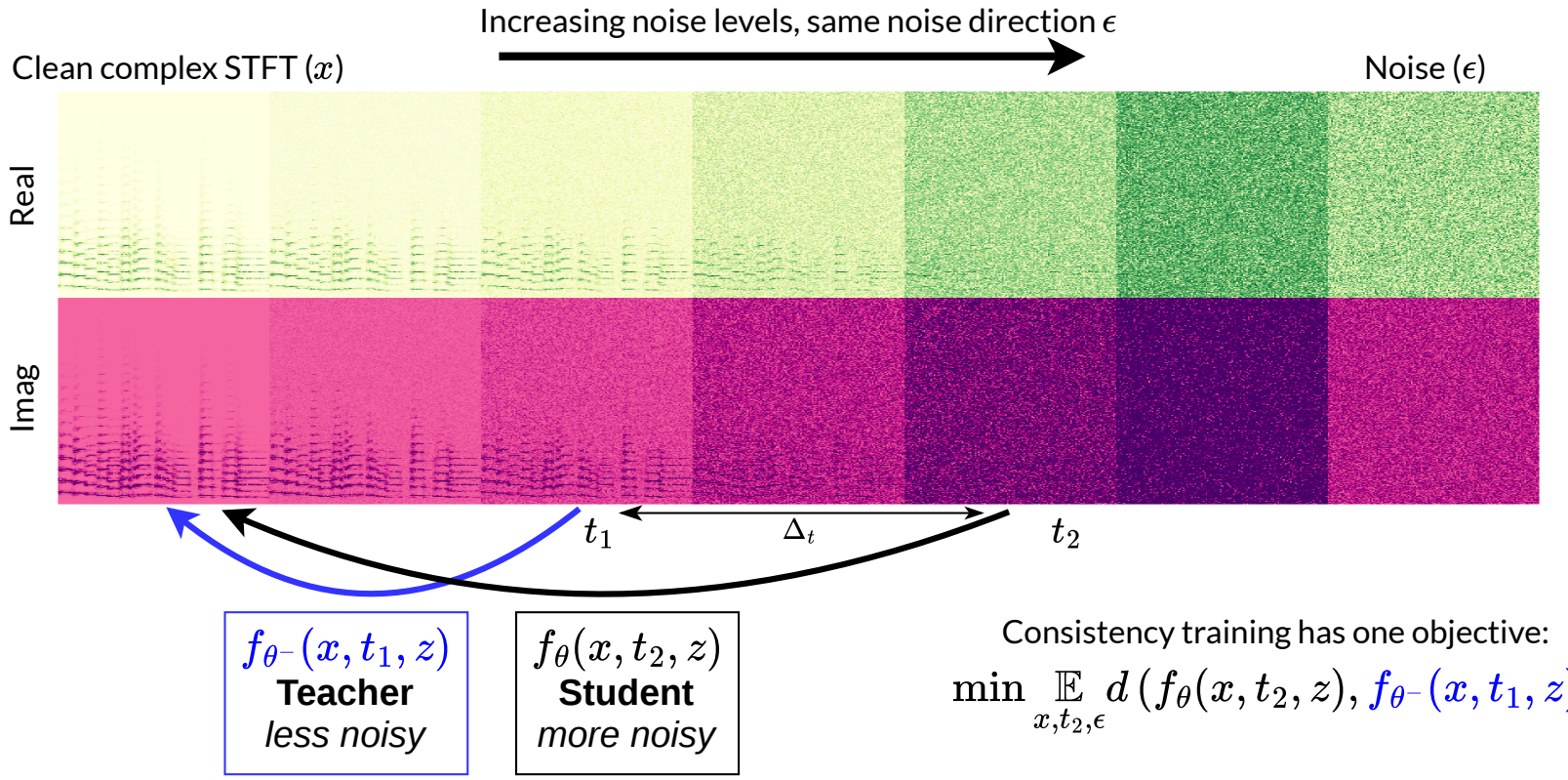
- The decoder is a **diffusion model conditioned on the latent z.**[1]
- Trained on a **single objective** -> no two-stage training.
- Efficient compression, fill details at inference time (e.g. high-frequency texture)

Consistency autoencoder (CAE)

Music2Latent[2] — single-step decoding, 64× compression at 44.1 kHz.

[1] Preechakul, K., Chatthee, N., Wizadwongsa, S., & Suwajanakorn, S. (2022). Diffusion Autoencoders: Toward a Meaningful and Decodable Representation. *Cvpr.long*, 10609–10619. https://doi.org/
[2] Pasini, M., Lattner, S., & Fazekas, G. (2024). Music2Latent: Consistency Autoencoders for Latent Audio Compression. *ISMIR*, 111–119. https://doi.org/

Complex STFT consistency training

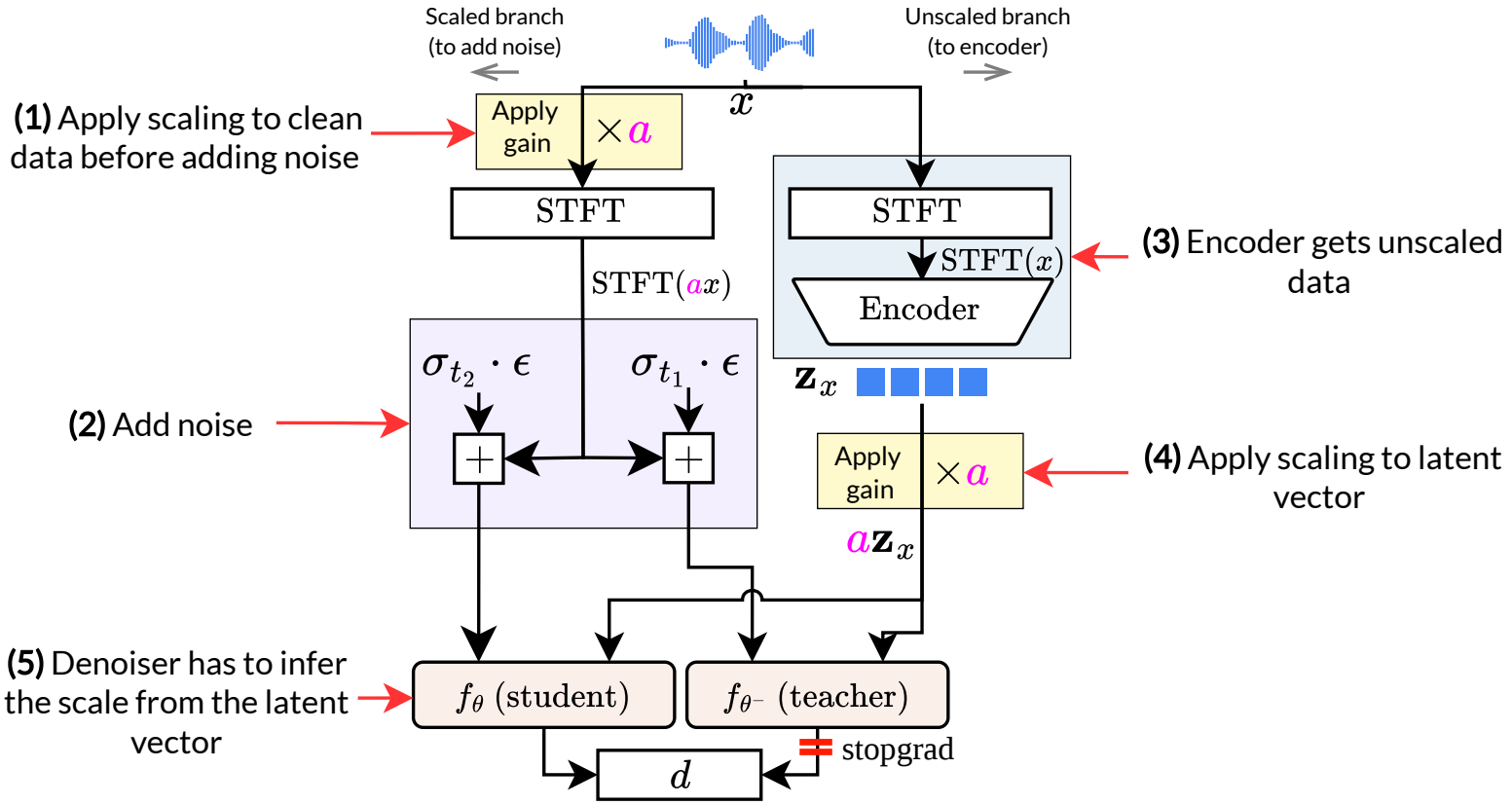


Consistency training has one objective:

$$\min \mathop{\mathbb{E}}_{x, t_2, \epsilon} d\left(f_\theta(x, t_2, z), f_{\theta^-}(x, t_1, z)\right)$$

- Single step inference possible for consistency training

Implicit gain regularization for CAE training



Training details

Training data

- **MTG-Jamendo** - full tracks
- **MoisesDB** - isolated instruments
- **M4Singer** - vocals · **E-GMD** - drums · **DNS-Challenge** - speech

Setup

- **64×** compression · **44.1 kHz** · 2s segments
- 800K steps on 1× L40S GPU (∼8 days)
- Only positive gains [0-1]

Baselines

- **Music2Latent-pub** [1]: Public weights
- **Music2Latent-ours** [1]: Retrained on same data, with mixes, without our implicit conditioning
- **Stable Audio VAE** [2]

[1] Pasini, M., Lattner, S., & Fazekas, G. (2024). Music2Latent: Consistency Autoencoders for Latent Audio Compression. *ISMIR*, 111–119. https://doi.org/
[2] Evans, Z., Parker, J. D., Carr, \. C., Zukowski, Z., Taylor, J., & Pons, J. (2025). Stable Audio Open. *ICASSP*, 1–5. https://doi.org/

Reconstruction quality and homogeneity (MusicCaps)

MSS = Multi-Scale Spectral Loss
KAD = Kernel Audio Distance
Homogeneity = equivariance to scaling (gain)

Model	Reconstruction MSS ↓	Reconstruction SNR ↑	KAD ↓	Decoder homogeneity MSS ↓	Encoder homogeneity error ↓
Music2Latent-pub	1.14	1.85	**5.69**	2.52	12.13
Stable Audio VAE	**0.72**	**7.32**	6.27	3.03	4.59
Music2Latent-ours	0.98	3.09	6.53	2.27	8.52
Lin-CAE (ours)	1.01	3.19	6.19	**1.37**	**0.69**

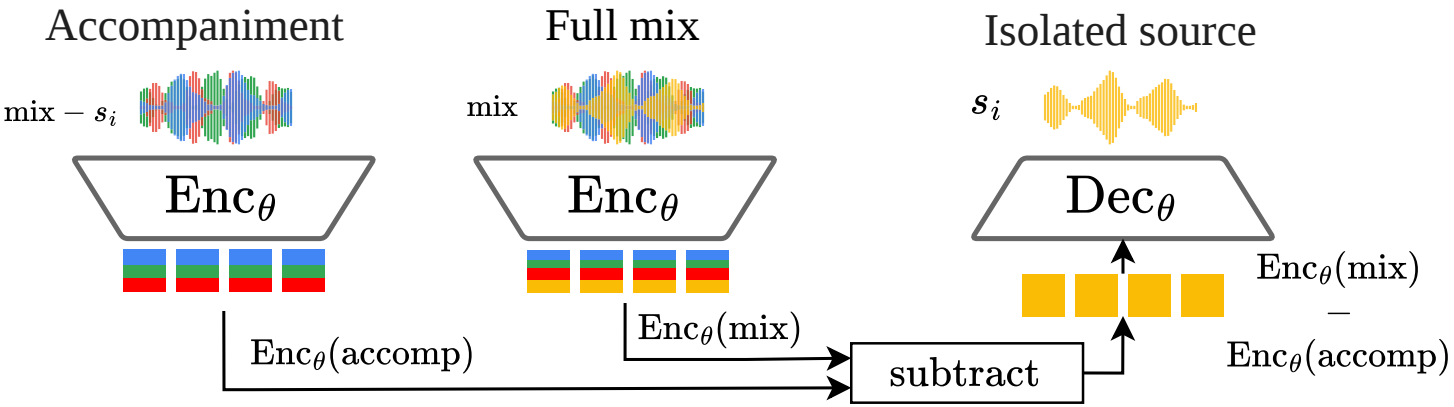
- Lin-CAE matches reconstruction quality
- Homogeneity error measures how well scaling of the latents “disturbs” the reconstruction
- Lin-CAE achieves much better homogeneity than baselines
- Emergent encoder additivity

Does the latent space preserve additivity? (MUSDB-18HQ)

Model	Decoder Additivity		Encoder
	MSS ↓	SNR ↑	Additivity Error ↓
Music2Latent-pub	5.01	−0.79	2.82
Stable Audio VAE	5.38	−12.58	<u>1.71</u>
Music2Latent-ours	5.21	−0.48	2.73
Lin-CAE (ours)	**0.99**	**1.22**	**0.60**

- **Trained on 2-source mixes** — evaluated on real **4-source mixtures** (bass + drums + other + vocals)
- **Decoder additivity MSS: 0.99 vs. 5+ for all baselines**
- Matches reconstruction values
- Emergent encoder additivity

Oracle source separation via latent arithmetic (MUSDB-18HQ)



Oracle source separation via latent arithmetic (MUSDB-18HQ)

Model	Reconstruction MSS ↓	Separation MSS ↓			
		Bass	Drums	Other	Vocals
Music2Latent-pub	1.16	95.34	28.08	6.78	5.10
Stable Audio VAE	**0.57**	323.26	87.80	3.65	3.08
Music2Latent-ours	0.97	39.56	12.26	5.30	3.49
Lin-CAE (ours)	1.00	**1.58**	**1.16**	**1.21**	**1.23**
Ablations					
– Additivity	1.39	6.61	2.37	3.44	2.87
– Homogeneity	0.96	23.61	7.84	2.98	3.08

- **Lin-CAE separation quality matches its own reconstruction**
- Much better than all baselines
- Removing either property degrades linearity. Homogeneity helps more than additivity.

Conclusion

Trained a generative autoencoder with **approximately linear behavior** in the latent space.

- With **no additional loss functions**
  - Only random mixes as data augmentation
- With **no loss of reconstruction quality**
- Applications:
  - Remixing audio in the latent space
  - Latent audio processing

Demos & paper

Demos

Paper

Models: `pip install lin-cae`