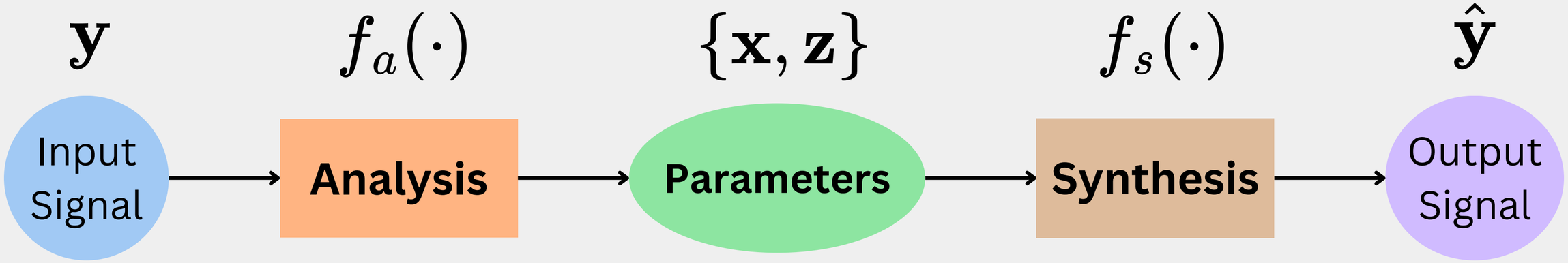
DDSP-Based Neural Audio Synthesis Model with Continuous Timbre Controls

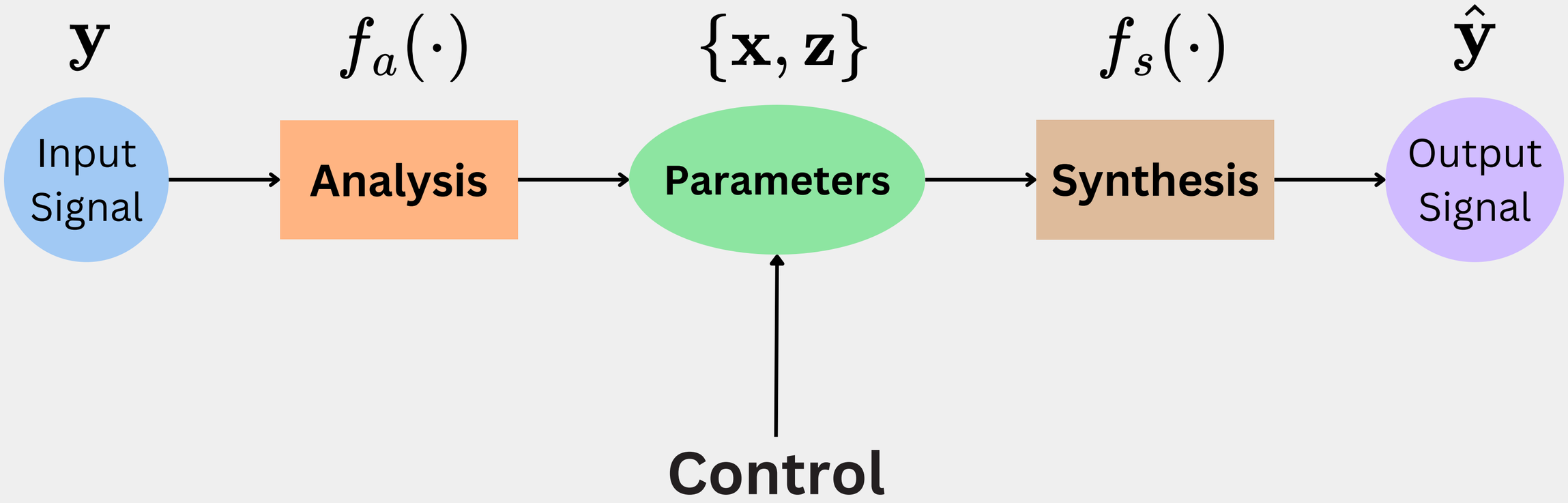
Aayush Kapur
Eto Sun
Junrui Huang
Yanting Zhou

Introduction

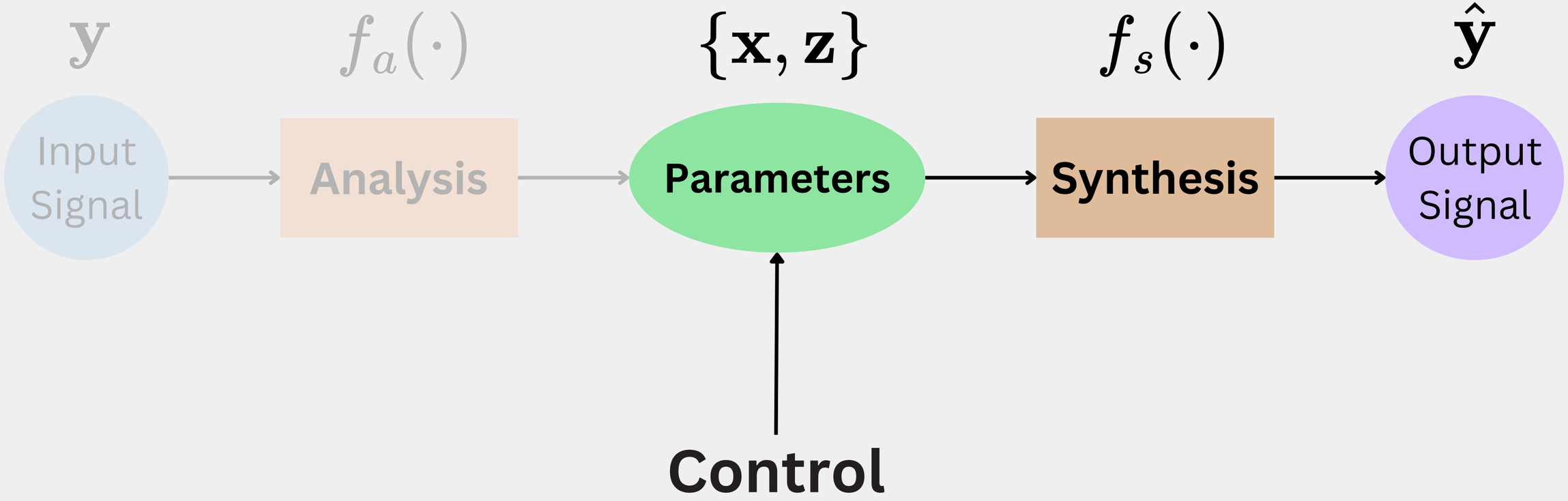
Audio Analysis and Synthesis Framework



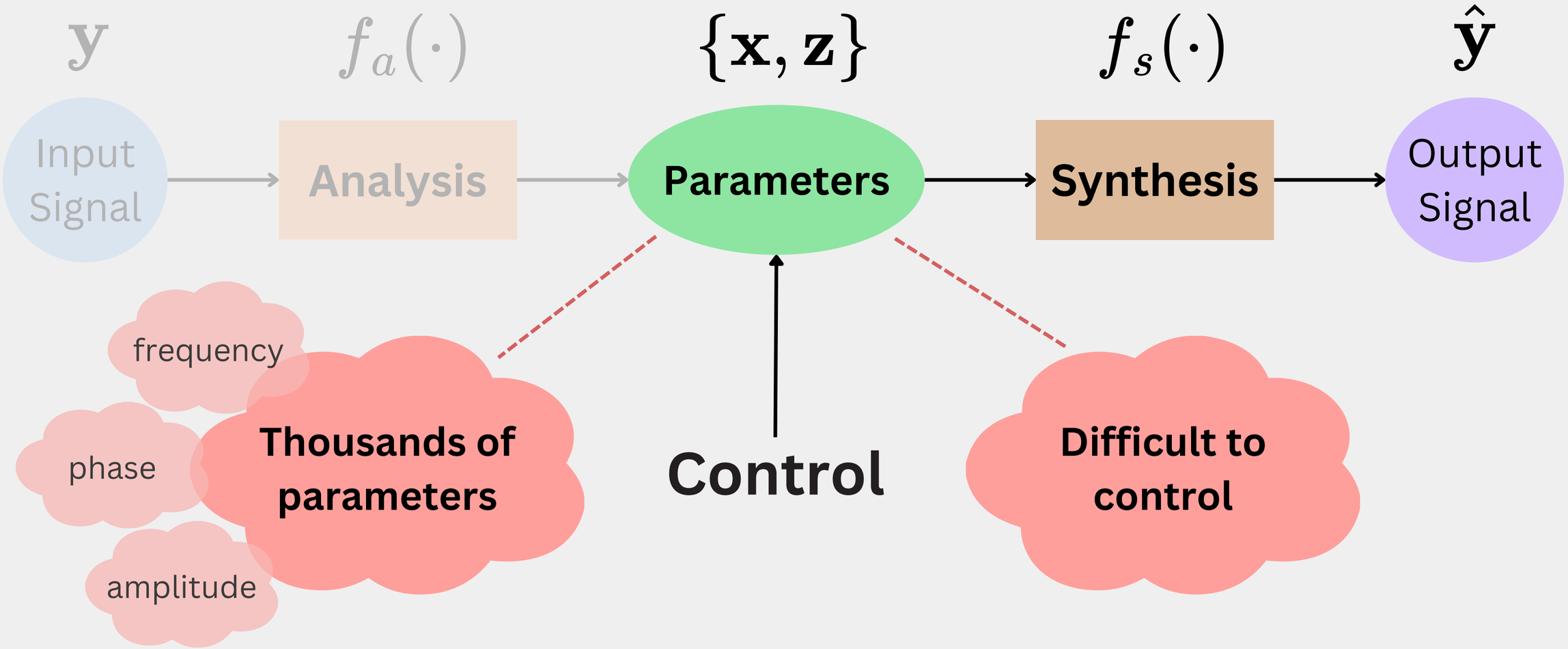
Audio Analysis and Synthesis Framework



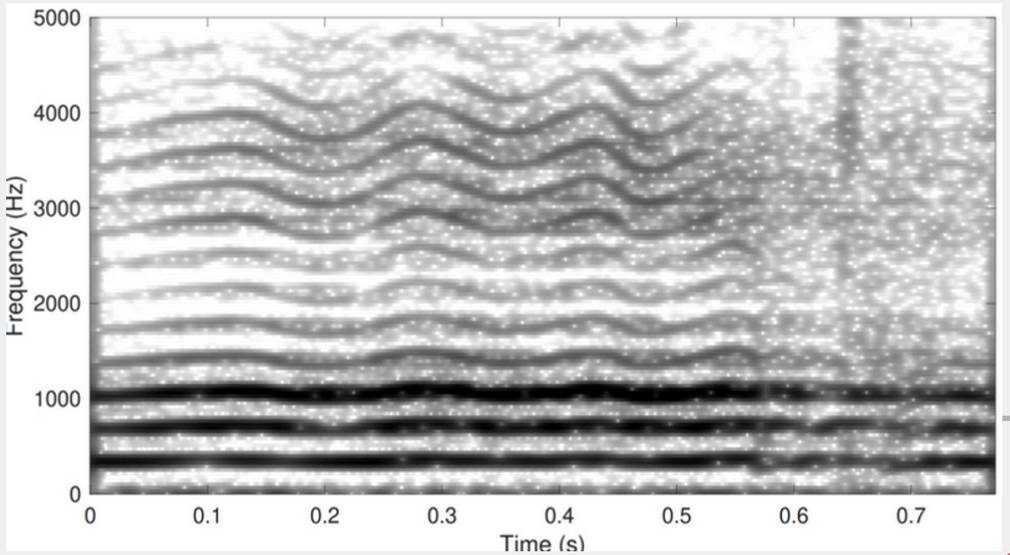
Audio Analysis and Synthesis Framework



Audio Analysis and Synthesis Framework



Audio Analysis and Synthesis Framework



$\{\mathbf{x}, \mathbf{z}\}$

$f_s(\cdot)$

$\hat{\mathbf{y}}$

Parameters → **Synthesis** → Output Signal

frequency

phase

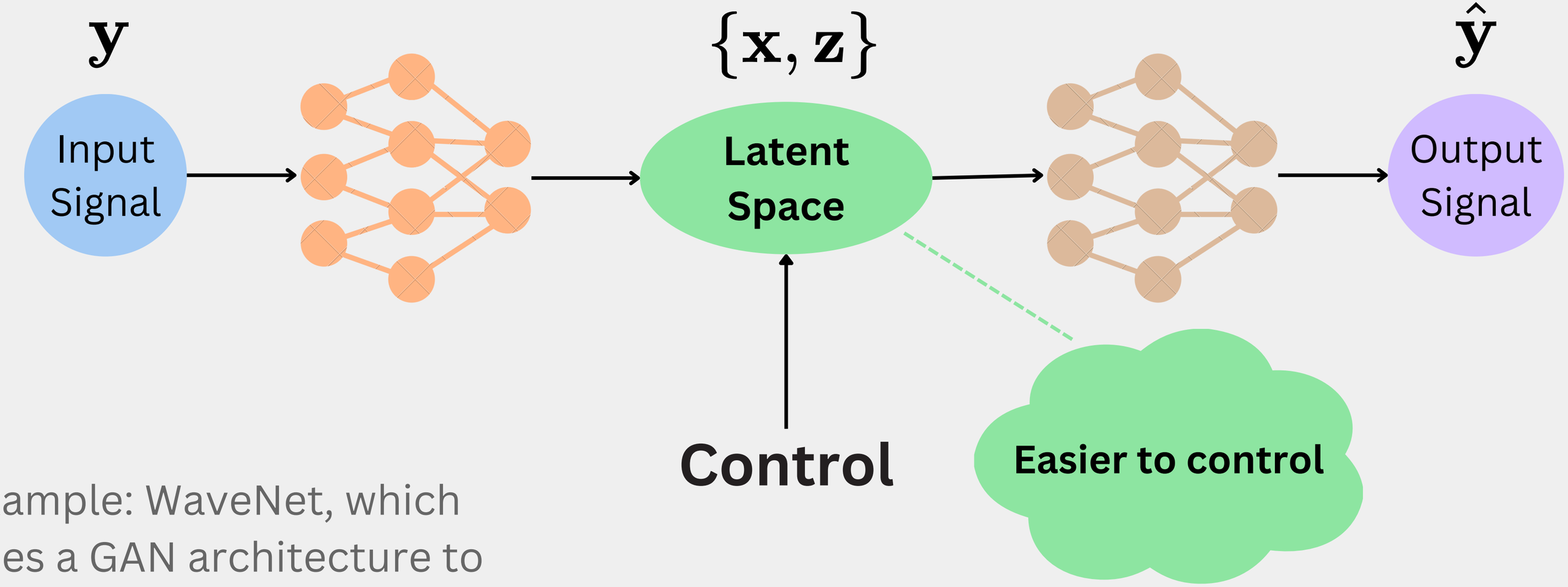
amplitude

Thousands of parameters

Control

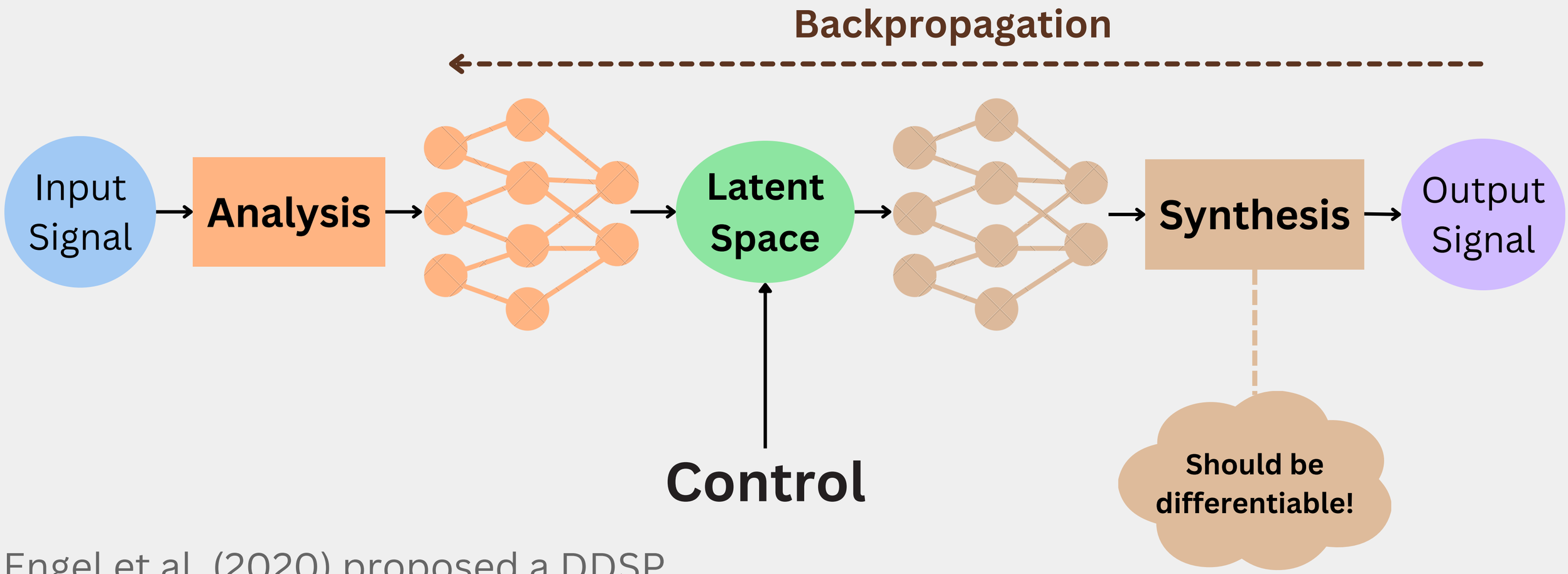
Difficult to control

Neural Audio Synthesis Model



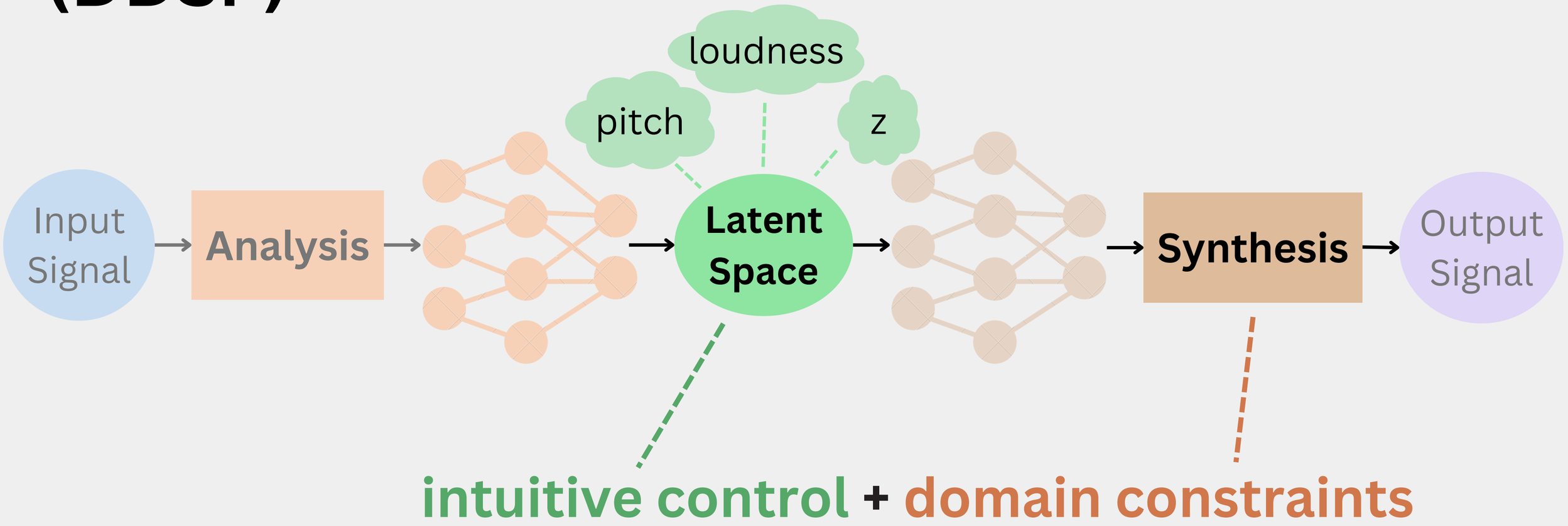
Example: WaveNet, which uses a GAN architecture to generate raw audio (Oord et al. 2016).

Differentiable Digital Signal Processing (DDSP)



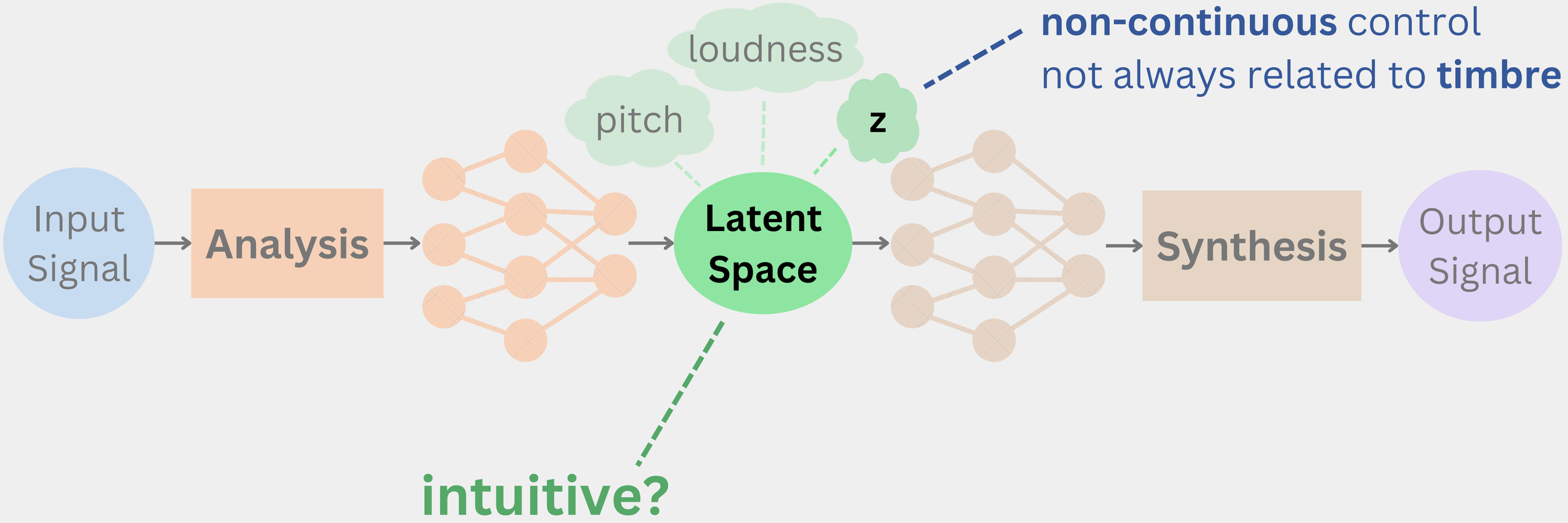
Engel et al. (2020) proposed a DDSP framework using an AutoEncoder architecture.

Differentiable Digital Signal Processing (DDSP)



Engel et al. (2020)

Motivation


very **abstract**
non-continuous control
not always related to **timbre**
loudness
pitch
z
Input Signal
Analysis
Latent Space
Synthesis
Output Signal
intuitive?

Introduction

- **Audio Synthesis**

  Neural audio synthesis model that utilizes deep learning to generate audio signals
  [Oord et al. 2016, Eagle et al. 2018]

- **DDSP framework**

  Integrates Digital signal processing knowledge with neural network training including back-propagation of parameters.
  [Engel et al. 2020]

- **Timbre**

  Multidimensional perceptual attribute that, together with pitch and loudness, affects the "quality" or "texture" of a sound.

Motivation

- It is difficult to understand how modifications in the latent space translate to specific changes in timbre.
- We want to create a model where modifications in the latent space directly translate to specific timbre changes.

Application:
A need for finer and meaningful timbre control, potential use for composers and sound designers.

Problem defined

Develop a deep learning model with controllable features in the latent space for timbre control based on the differentiable digital signal processing (DDSP) framework.

Such that,

- Quality of audio synthesis should be good.
- Control over timbre
- User experience: A human user should feel that the sound aligns with what they expect for that particular combination of control factors.

Problem Formulation

Develop a deep learning model based on the DDSP framework that have:

- **controllable features** in the latent space for specific timbre control
- **continuous control** in the latent space
- **high perceived quality** of our reconstruction signals

Application

A need for finer and meaningful timbre control, potential use for composers and sound designers.

Constraints

- Concentrate on synthesizing one specific type of sound: harmonic sinusoids and noise (domain knowledge from DSP)
- Model's latent space should be capable of representing specific timbre dimensions for further control

Dataset and Resources

NSynth: audio dataset

305,979 musical notes, each with a unique pitch, timbre, and envelope.

Each sample is a four second, monophonic 16kHz audio snippet.

can be either: acoustic or electronic (for instruments), or synthetic

We worked with:

All acoustic flute sound samples.
GPU used: RTX 3070 laptop

80% training set, 5% validation set, and 15% test set

Methodology

Baseline Structure

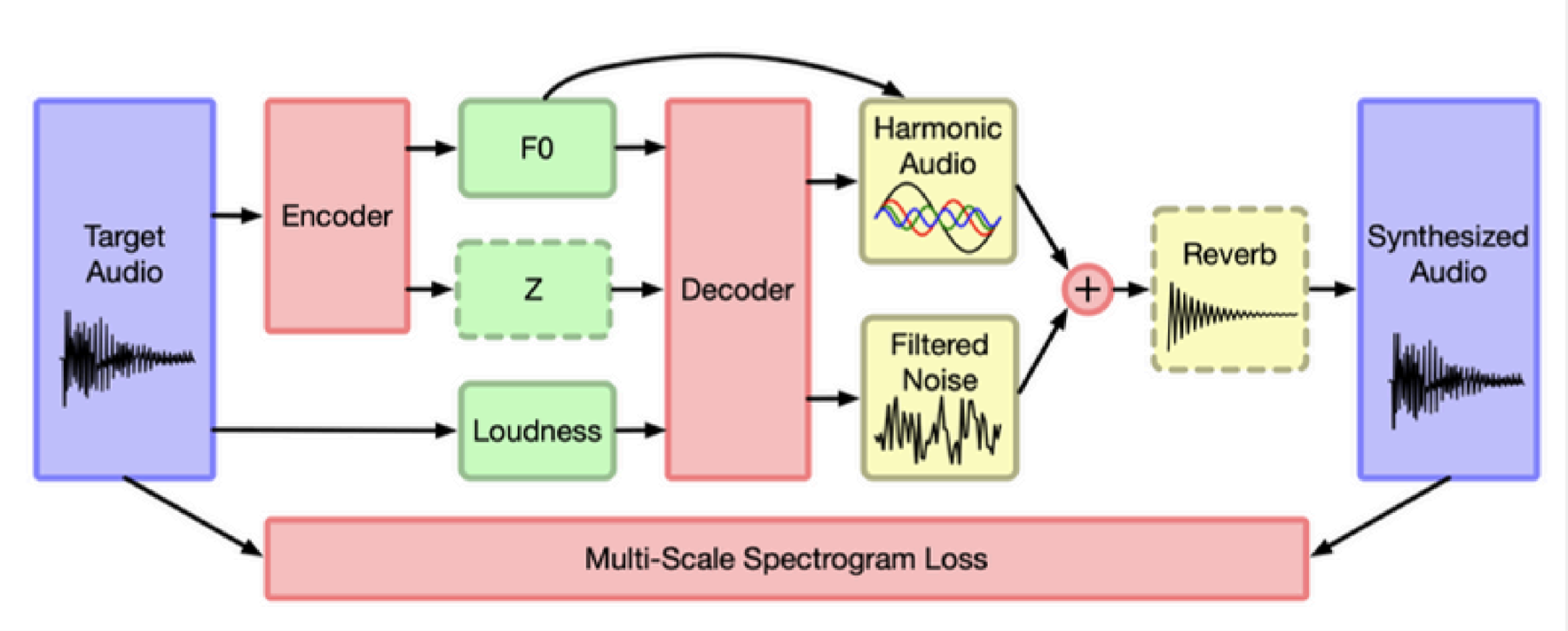
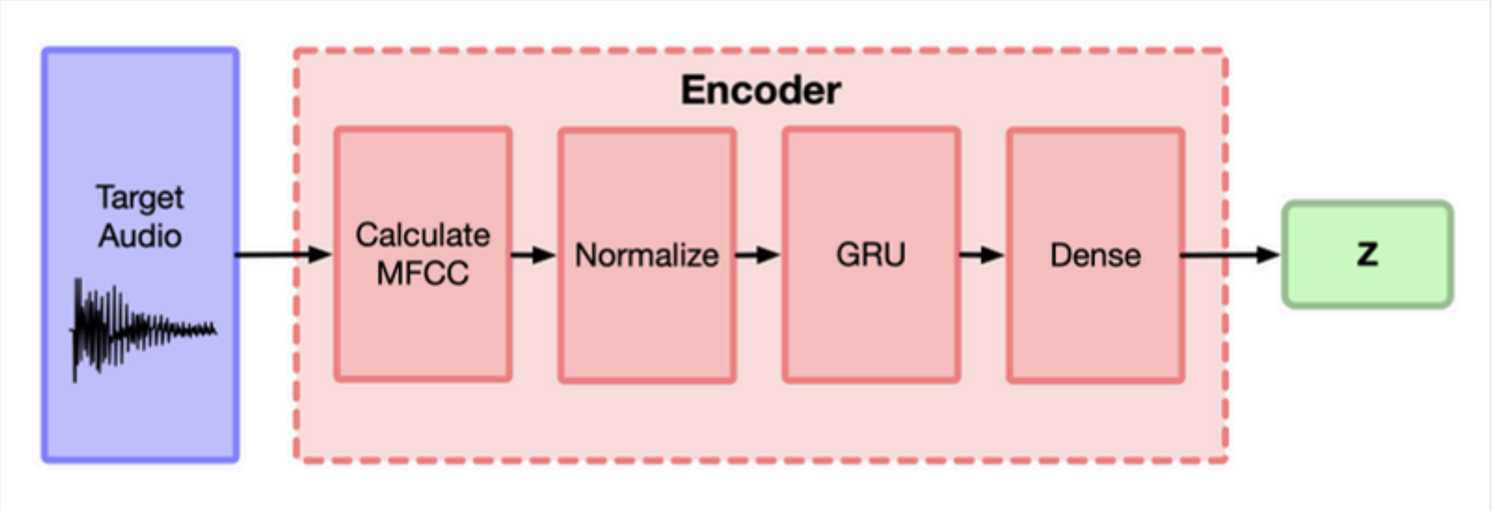


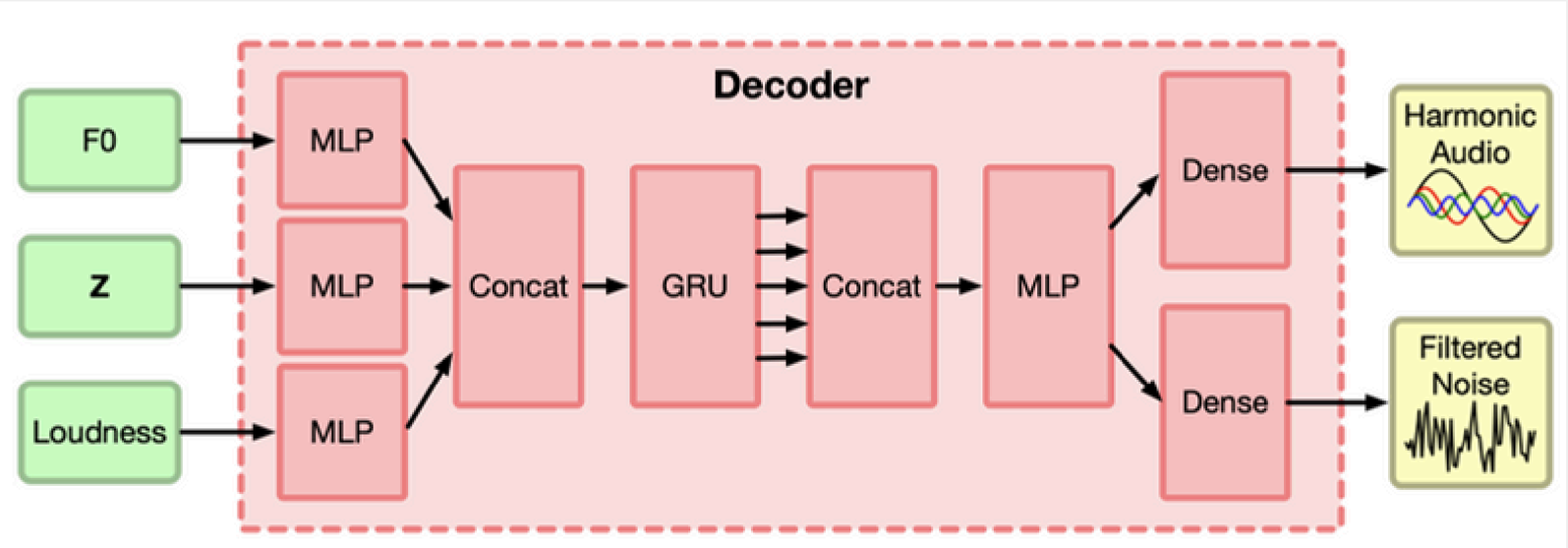
Fig. Autoencoder architecture

Engel et al. (2020) proposed a DDSP framework using an AutoEncoder architecture.

Methodology

Baseline Structure





Methodology

New Model -- Extra timbre descriptor

- **Added timbre as an additional feature to the model such that a better embedding can be learned.**
- **Timbre is a time-invariant feature so it needs to be augumented before concatenation.**

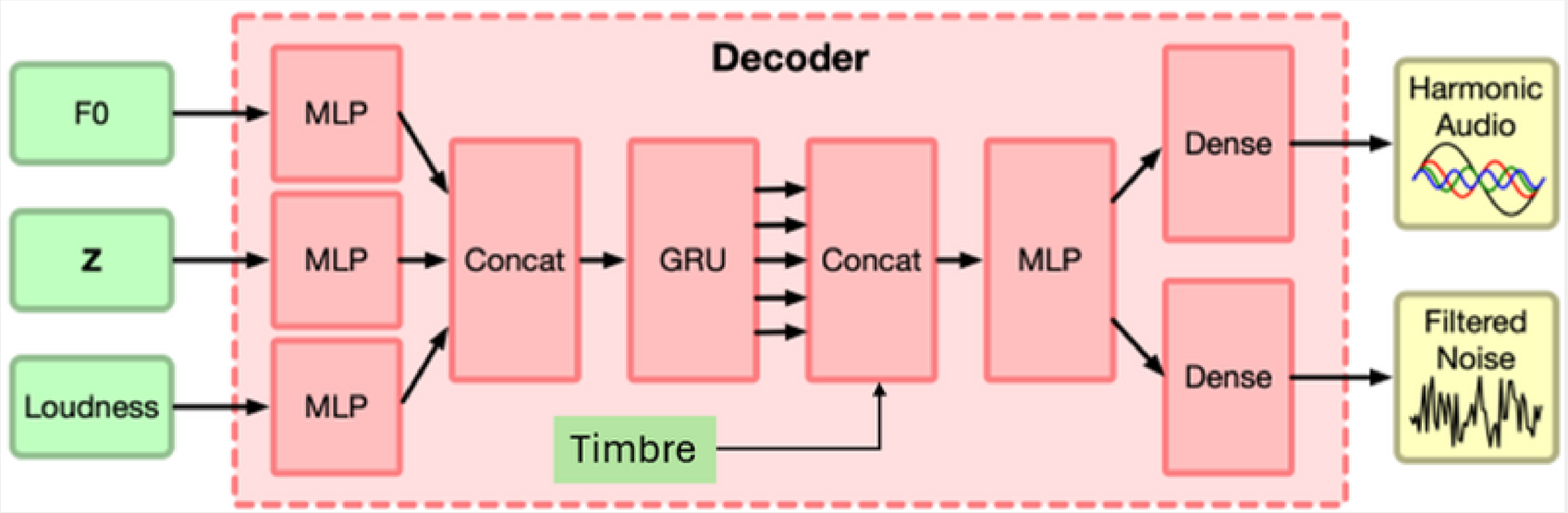


Fig. Decoder structure

Methodology

New Model -- Extra timbre descriptor

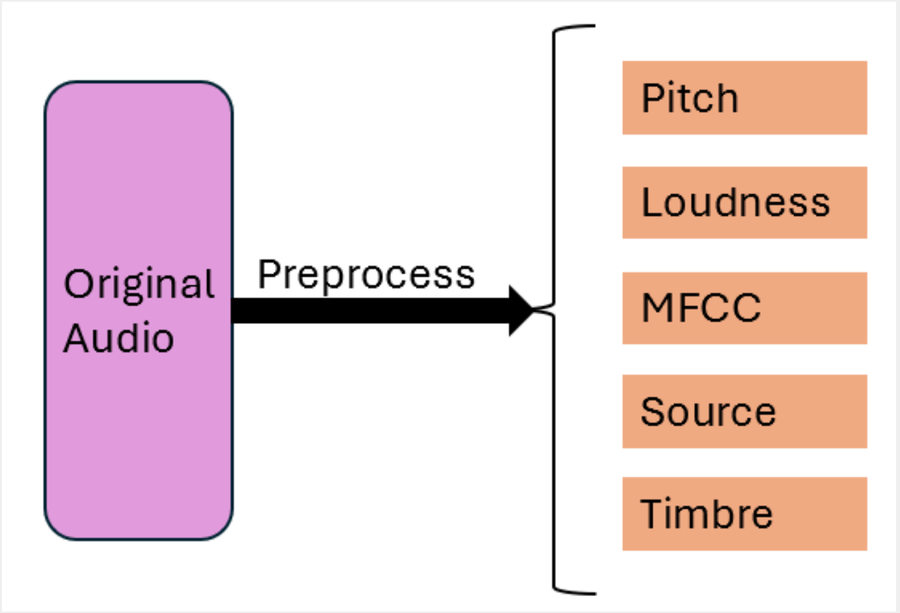
- Timbre feature includes:
  - Spectral centroid
  - Spectral flatness
  - Temporal centroid

$$\mu_1(t_m) = \sum_{k=1}^{K} f_k \cdot p_k(t_m).$$

$$\mathrm{SFM}(t_m) = \frac{\left(\prod_{k=1}^{K} a_k(t_m)\right)^{1/K}}{\frac{1}{K}\sum_{k=1}^{K} a_k(t_m)}.$$

$$tc = \frac{\sum_{n=n_1}^{n=n_2} t_n \cdot e(t_n)}{\sum_{n} e(t_n)},$$

- Python Library “Librosa” is used to extract them.
- Extraction done on preprocessing step of the orignal audios.
- Include timbre information into loss function.



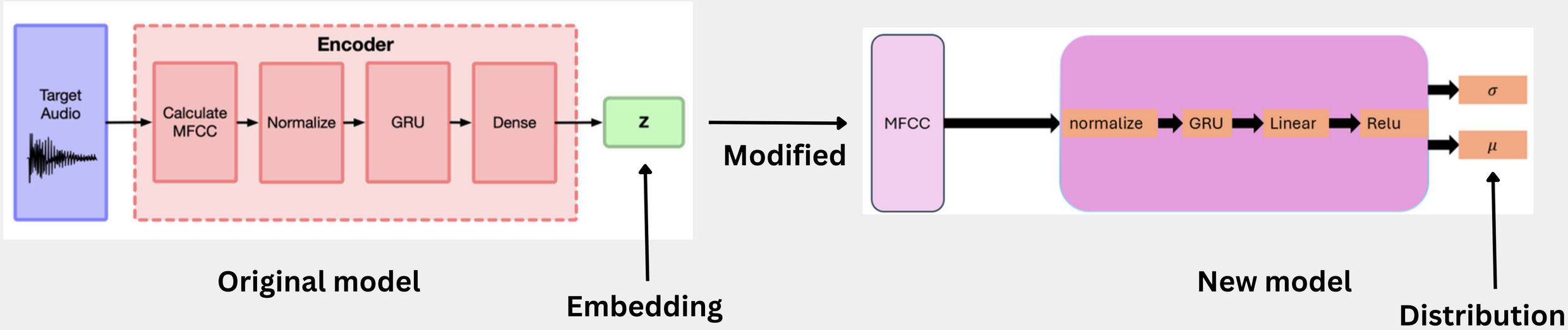
Methodology

New Model -- VAE

- **Loss function:**

$$L_{\text{vae}}(x) = \mathbb{E}_{\hat{x}\sim p(x|z)}[S(x,\hat{x})] + \beta \times D_{KL}[q_\theta(z|x)\|p(z)]$$

$$S(x,y) = \sum_{n\in\mathcal{N}}\left[\frac{\|\text{STFT}_n(x) - \text{STFT}_n(y)\|_F}{\|\text{STFT}_n(x)\|_F} + \log\left(\|\text{STFT}_n(x) - \text{STFT}_n(y)\|_1\right)\right]$$

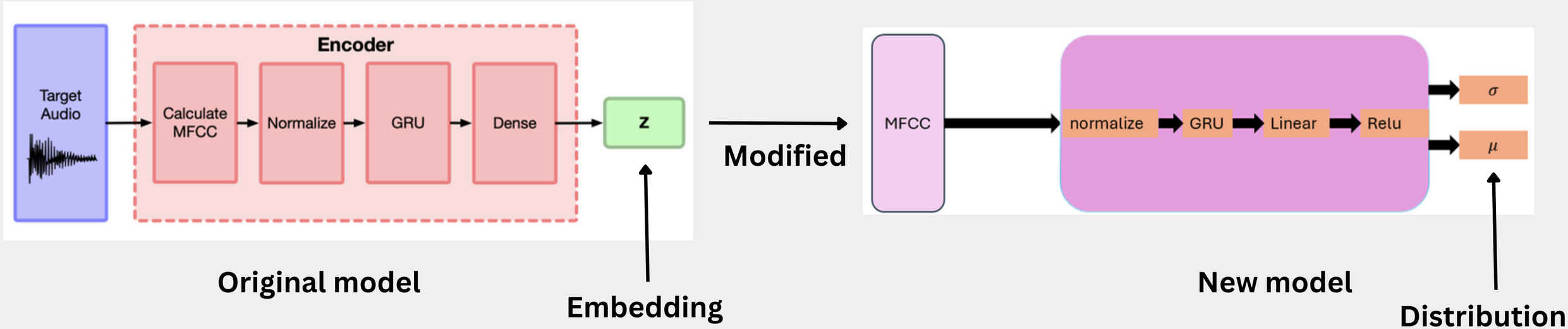


Antoine & Philippe (2021) proposed the loss function

Methodology

New Model -- VAE

- Regularization: Add dropout layer after GRU (RNN)



Metrics

Performance evaluation metrics

- **L1 distance of pitch**
- **L1 distance of loudness**
- **L1 distance of spectral centroid**
- **L1 distance of spectral flatness**
- **L1 distance of temperal centroid**

Timbre

Results

Table 1. Performance metrics of alternative baseline models vs new models with VAE

model type	Baseline Model	Baseline Model + Timbre	VAE Model	VAE Model + Timbre
L1 pitch distance (note)	5.5369	5.6877	5.219	1.3089
L1 loudness distance	0.4164	0.4199	0.4017	1.2368
L1 spectral centroid distance (note)	0.4656	0.4299	0.3202	1.7548
L1 spectral flatness distance	0.0054	0.0044	0.0034	0.0002
L1 temporal centroid distance (sec)	0.0827	0.0703	0.0703	0.0948

Conclusions and Future work

Added timbre and changed autoencoder into VAE

To satisfy constraints:

- Only focus on synthesizing flute sound

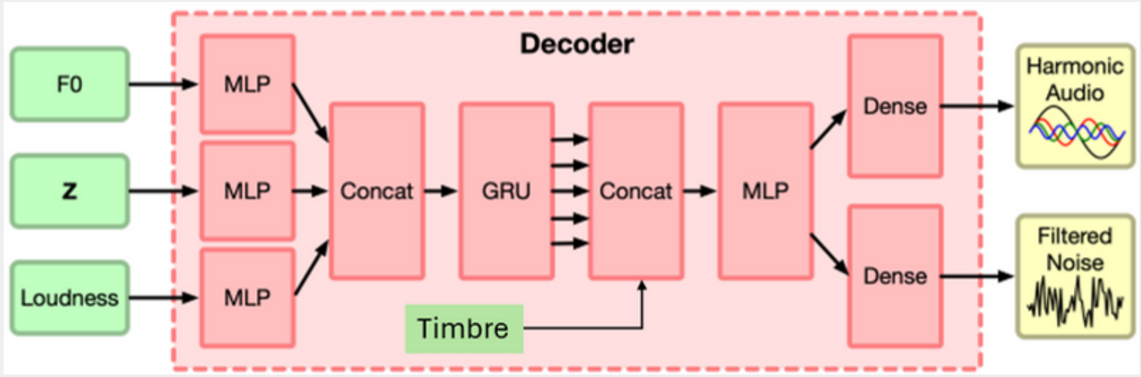
Pros and Cons for proposed models

- VAE without timbre: all metrics improved; but pitch difference still high.
- VAE + timbre: much better performance on pitch and spectral flatness

Future work:

- Try incorperating timbre into encoder
- Domain adaptation

  acoustic, electronic, and synthetic



End

Thank you!

Do you have any questions?