ATIAM

VARIATIONAL AUTO-ENCODER

SYNTHESIZER

Machine Learning Project

Adrien BARDET
Alexis ADONIADIS
Bavo VAN KERREBROECK
Cyran AOUAMEUR
Pierre MASSÉ

Phillipe ESLING
Axel CHEMLA

December 25, 2017

Contents

Abstract

The main goal of this project was to develop variational models to find an interesting generative sound synthesis space, where each point of this space corresponds to a machine created sound, coming from the high-level understanding of the input data. For that purpose, we used an unsupervised generative machine-learning model (Variational Auto-Encoder). A meaningful latent space was generated concerning sound synthesis, which can be used for artistic purposes.
To train our VAE, we created toy datasets of various complexities, allowing us to compare many models and to understand what the Auto-Encoder was performing. Once it has satisfied the characteristics one wishes, it can automatically be extended to any type of sound. At the end of this project, we created a real-time synthesizer that can produce sounds depending on parameters encoded in the latent space. Thanks to dimensionality reduction techniques, we can easily explore this high dimensional space and provide simple control to the musician.

Chapter 1

Introduction

1.1 Variational Auto-Encoders

1.1.1 The model

The central component of our project is an unsupervised generative machine-learning model known as a Variational Auto-Encoder [1]. It is based on a two-stage inference/generation procedure that showed great generalization properties and good reconstruction abilities despite its light structure. It is composed of an encoder and a decoder.

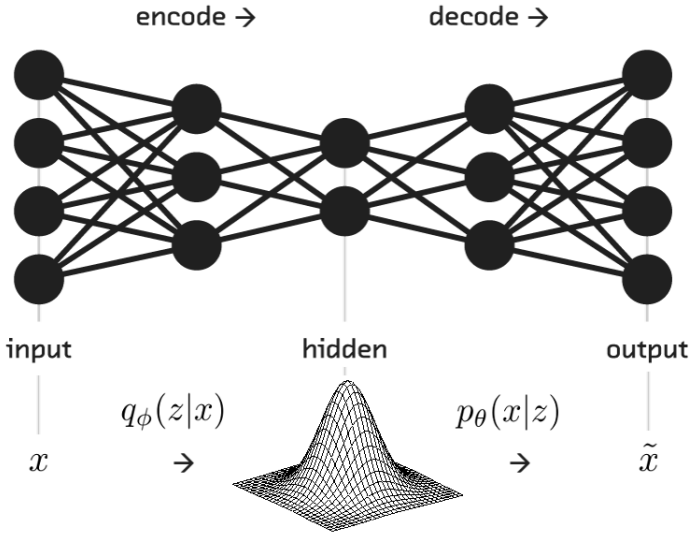


Figure 1.1: Basic scheme of a VAE using neural networks (property of fastforwardlabs.com)

The encoder is a neural network. Its input is a data vector x, its output is a hidden representation z, and it has weights and biases ϕ. Let's say x is a 1025-point spectrum. The encoder 'encodes' the 1025-dimensional data into a latent (hidden) representation space z, which is chosen to be much smaller than 1025 dimensions. For example, one dimension of the latent representation space could encode the pitch and another one could embody the signal to noise ratio. Let's denote the encoder $q_\phi(z|x)$. We note that the latent space is stochastic: the encoder outputs parameters to $q_\phi(z|x)$, which are Gaussian probability densities. Hence, the latent space is modeled by a multidimensional Gaussian distribution.

The decoder is another neural network. Its input is the representation z, it outputs the parameters of the probability distributions of the data, and has weights and biases θ. The decoder is denoted by $p_\theta(x|z)$. The probability distribution of a single frequency bin can be represented using a Gaussian distribution. The decoder gets as input the latent representation of a spectrum z and outputs 1025 Gaussian parameters (mean, variance), one for each frequency bin. The decoder 'decodes' the real-valued numbers in z into 1025 Gaussian parameters (mean, variance). Thus, it generates a multidimensional Gaussian distribution for the output $\mathcal{N}(\mu_X; \sigma_X)$.

$$logP(X) - DKL[Q(z|X) \parallel P(z|X)] = E[logP(X|z)] - DKL[Q(z|X) \parallel P(z)] \tag{1.1}$$

1.1.2 Loss function

It is the evidence lower bound (ELBO) to be maximized. It corresponds to the right term in equation (1.1). As the divergence in left term $DKL[Q(z|X) \parallel P(z|X)]$ is always positive, maximizing the loss would tend to the lower bound of $logP(X)$. As we can see, there are 2 terms in loss functions: $E[logP(X|z)]$ is the reconstruction term, $DKL[Q(z|X) \parallel P(z)]$ is the regularization term. To maximize the loss function, the first has to be maximized while the second minimized.

Reconstruction term

The reconstruction term in loss gives information about how far is the VAE output from its input. It is a maximum likelihood.
For a Gaussian modeled input distribution:

$$E[logP(X|z)] = -\frac{1}{2}\left[log(2\pi\sigma^2) + \frac{[x - p(x)]^2}{\sigma^2}\right] \tag{1.2}$$

For a binary (Bernoulli) modeled input distribution, it is the binary cross entropy:

$$E[logP(X|z)] = -\frac{1}{n}\sum_x [p(x)log(x) + (1 - p(x))log(1 - x)] \tag{1.3}$$

Regularization term

This term is the driver towards a meaningful representation of our VAE input in the latent space. As we modeled the latent space as a normalized Gaussian distribution, the associated divergence term equals:

$$DKL[N(\mu(X), \Sigma(X)) \parallel N(0, 1)] = 0.5.(\sum_k exp(\Sigma(X)) + \mu(X)^2 - 1 - \Sigma(X)) \tag{1.4}$$

The 2 loss function terms can be balanced by introducing a coefficient on the regularization term called β. This coefficient puts more or less impact of regularization in total loss. A small β (near 0) would make reconstruction more important. Once it is trained, the VAE would make good reconstruction of inputs. Whereas a bigger β (e.g. $\beta = 4$) would lower reconstruction quality for a better latent space regularization. Increasing beta > 1 as showed an improved disentanglement of the latent space [2]. Thus, it should generate a more various range of data when decoding.

Chapter 2

Datasets

To create our toy datasets, we have made the following assumptions :
The fundamental frequency of our sum of sinuses and their total energy is constant over all our datasets. Time variations aren't taken into account, all our representations (see Chapter 3) are encoded into a single vector.

2.1 One-factor Harmonic Dataset : Spectral Richness

The first dataset is a very simple series of increasing number of harmonics on a fundamental frequency of 220Hz. The spectral slope fits a $\frac{1}{n}$ function with n : the index of the harmonic. Ten thousand spectra have thus been created. The first spectrum represents a pure sinus. The second one, a pure sinus added with a fraction of its octave. Then this fraction increases until it reaches $\frac{1}{n}$ ($\frac{1}{2}$). Then a fraction of the third harmonic is added until it reaches $\frac{1}{n}$ ($\frac{1}{3}$) and so on.
Spectrums are generated using a 4096 Fast Fourier Transform (FFT) from which we only used the first half of the positive frequencies (1024 frequency bins) as we decided to concentrate on the lower part of the spectrum.

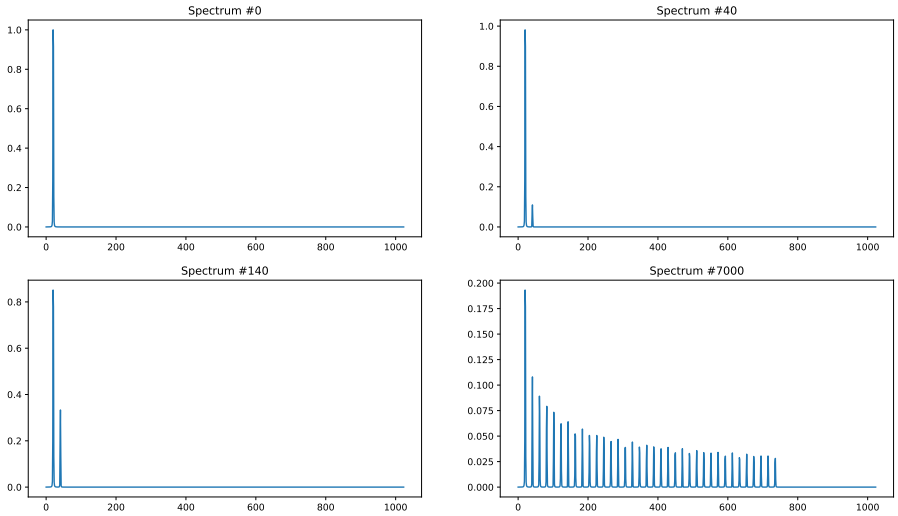


Figure 2.1: Spectral richness toy-dataset's spectra

This dataset was very useful to exhibit characteristics from our VAE. As it is very simple, subtle differences in regularization or dimensionality reduction were enabled to be seen.

2.2 Four- and five factor rich and filtered Dataset

A second dataset adds further complexity by increasing the number of parameters used to generate the samples. Furthermore, samples were created using the 'pyo' python library allowing an easy verification with eventual reconstructed output from the VAE.

The datasets with 4 and 5 parameters generate sinusoidal components using respectively additive and Frequency Modulation (FM) synthesis with a respectively Biquad and Moog ladder filter operation afterwards. All datasets are created with sample-rate (sr) 44100 samples/second and fundamental frequency (f0) equal to 110. Labels for each spectrum were generated using the range index of each factor of variation as listed below.

The four-factor dataset is created using following formula and parameters:

- Inharmonicity j: range(0,1,0.1)
- # Harmonics k : range(0,20,1)
- Filter resonance Q: range(1,11,1)
- Filter frequency f: range(1,11,1)

$$s(n) = BiQuadfunc(sin(2\pi(2k + j)f0t), f(sr)/200, Q) \tag{2.1}$$

Given the above factors of variations, this dataset resulted in a (1025,20 000) matrix of spectrum with a (4,20 000) matrix of associated labels.

The five-factor dataset is created using following formula and parameters:

- Carrier j: range(1,5,0.5)
- Ratio k: range(0.1,0.4,0.05)
- Index r: range(0,30,4)
- Filter resonance Q: range(1,11,1)
- Filter frequency f: range(1,11,1)

$$s(n) = Moogfunc(FM([f0 - j * f0/10, f0, f0 + j * f0/10], k, r), f(sr)/200, Q) \tag{2.2}$$

Given the above factors of variations, this dataset resulted in a (1025,44 800) matrix of spectrum with a (4,44 800) matrix of associated labels.

Further experiments were executed using normalized (values between 0 and 1), decibel and noisy versions of the above datasets. Noise was simply an added using an additional factor of variation controlling the balance between the original signal and noise power.

2.3 Normalizing the input data ?

As linear magnitude spectra can reach values of amplitude up to around 800 and as low as 10^{-14}, normalizing the data between 0 and 1 has been a question throughout the project. We will show a result from a normalized input trained VAE.
The main issue we revealed with normalized input magnitude spectra is that the VAE performs a global mean and projects every input on the same multidimensional output point. We plotted the mean and variance on each dimension for 100 spectra. (figure 2.2)

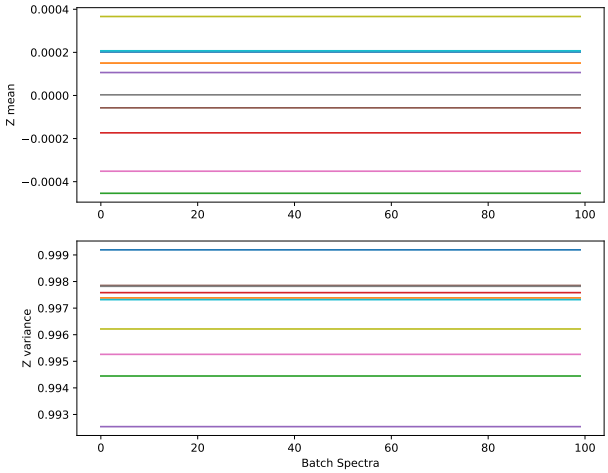


Figure 2.2: Normalized input - Latent space's means and variances over a 100 batch

This training was done with a 10-dimensions latent space. On figure 2.2, one color represents the means and variances of one dimension. On the horizontal axis, we find the spectrum index from the 100-spectra batch forwarded in the VAE. As we can see, sampling the latent space won't be interesting as the output is the same for every spectrum.
Training the same VAE with the same parameters, but with the input **not normalized** gives us back the expected expressiveness of the VAE. The means and the variances vary with the input (see figure 2.3), interesting latent space exploration and fine reconstruction are enabled.

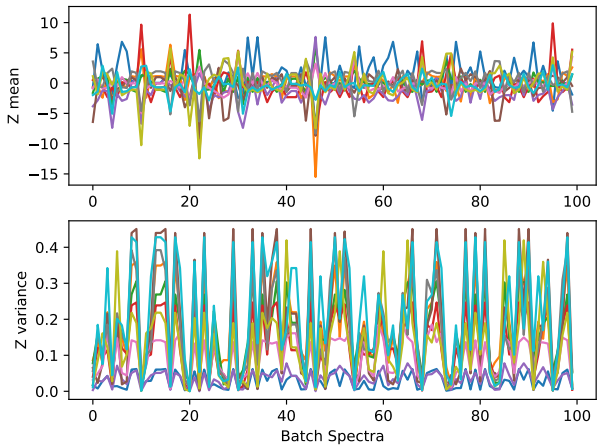


Figure 2.3: Unnormalized input - Latent space's means and variances over a 100 batch

Chapter 3

Testing Different Sound Representations

This chapter will briefly describe the different sound representations and their configurations used as inputs for our VAE model. Quantitative and qualitative performances will be assessed. Experiments in this chapter mainly use the 4- and 5-factor dataset, directly analyzing the relations between multiple factors of variation, their representation and the VAE performance.

3.1 Representations

During the project, we did not take temporal variations of spectra or audio into account. Indeed, working with spectrograms or modulation spectra instead of spectra severely complicates the analysis and the training of the VAE. An interesting next step for our research would be to include temporal variations and adapt the VAE model accordingly (e.g. using convolutional networks or increasing the dimensionality of input and output layers).

Discrete Fourier Transform with and without phase

The choice for magnitude spectra is an evident choice due to its sparse representation of audio signals. Spectra were calculated using a 4096-point discrete Fourier transform and taking the first 1025 positive frequencies per input. For the first part of our experiments we normalized the magnitudes between 0 and 1. As results improved drastically without normalization, some results presented here are generated using unnormalized data. However, we performed additional experiments including the phase component as well, resulting in 2050 samples per input feeding our VAE. The formulas for the spectrum, magnitude and phase are presented below.

$$X_k = \sum_{n=0}^{N-1} x_n . e^{-i2\pi kn/N}, \quad |X_k| = \sqrt{Re(X_k)^2 + Im(X_k)^2}, \quad arg(X_k) = atan(\frac{Im(X_k)}{Re(X_k)}) \tag{3.1}$$

Modified Discrete Cosine Transform

As a third and final representation, we calculated the modified discrete cosine transform (MDCT) based on the IV-type discrete cosine transform. This transform was chosen due to its good compression properties (large energy in low frequency components), ease of computation, ease of reconstruction and inclusion of a (brute) phase component in the sign of transformed coefficients. The used formula is presented below for which reconstruction is achieved using overlap-add of subsequent windows. The MDCT was calculated with N = 2048 resulting in input of 1024 samples.

$$X_k = \sum_{n=0}^{2N-1} x_n . cos\big[\frac{\pi}{N}(n + \frac{1}{2} + \frac{N}{2})(k + \frac{1}{2})\big] \tag{3.2}$$

3.2 Evaluation

This section will compare the results for spectra, spectra with phase and MDCT. During our experiments and for the results in following subsections we modeled our spectra as independent identically distributed Gaussian distributions using a basic VAE with following parameters unless stated otherwise:

1 hidden layer with 400 elements	$\beta = 1$	No warm-up period	Batch sizes $= 100$
Learning rate = 0.0001	600 epochs	Z dimensions fitted to data (i.e. 4 or 5)	Relu activation functions

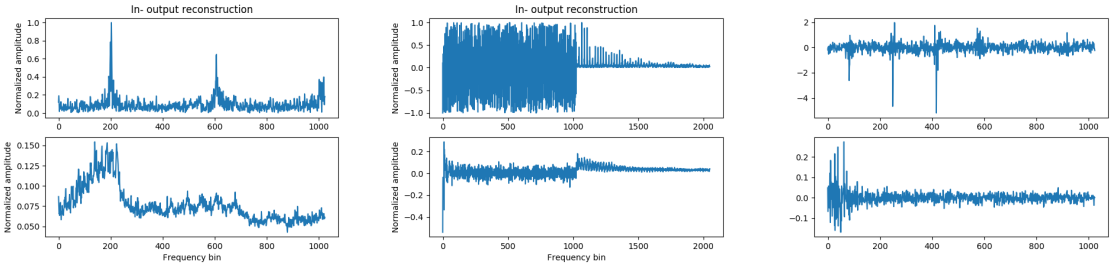
Overall, we obtained interesting and well-regularized latent space representations of our data while reconstruction quality was relatively poor.

MDCT representations suffered the most in reconstruction which could be explained given the VAE can be interpreted as an additional compression layer. With a large part of the energy contained in the low frequencies and a large spread in the absolute values, the VAE model had great difficulties reconstructing higher frequencies.

Spectra with and without phase information were poorly reconstructed as well and for which possible causes are the high complexity of the datasets with a relatively simple VAE and a small number of latent dimensions, inappropriate normalization Our model generalized strongly as reconstructed input data had little variation overall. However, organization of the data in the latent space showed good regularization and clustering properties allowing distinction between factors of variation in our dataset.

3.2.1 Comparing in- and output

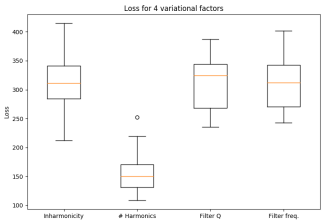
Reconstruction tends toward the lower frequencies as shown on the following figures. With mean energy of our dataset situated around lower frequencies, it demonstrated the generalization properties of the VAE model.



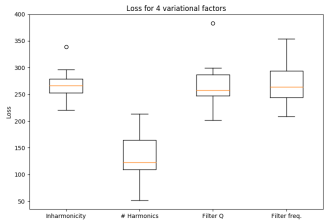
(a) Spectrum (b) Spectrum + phase (c) MDCT

Figure 3.1: Reconstructing input

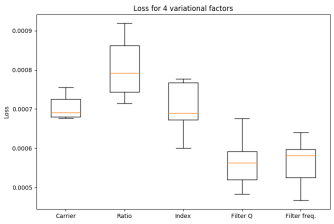
To get an idea of the VAE modeling behavior with regards to the variational factors in our dataset, we calculated mean loss for our input dataset keeping one factor value constant. For example, given we have 10 increments of the number of harmonics in dataset 2, the below box-plot shows the distribution of the mean loss of each harmonic number increment while varying the remaining factors inharmonicity, filter Q and filter frequency. Different representations had no influence on the loss distribution per factor and the below figures show how the number of harmonics for dataset 2 and the filter Q and filter frequency for dataset 3 were best recognized by the VAE model. It must be noted that figures with 4 factors were calculated using unnormalized data, further tests are needed to confirm this result for broader conditions.



(a) Spectrum + phase, 4 factors



(b) Spectrum + phase 4 factors

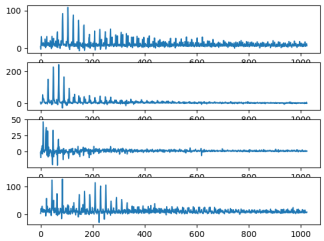


(c) MDCT 5 factors

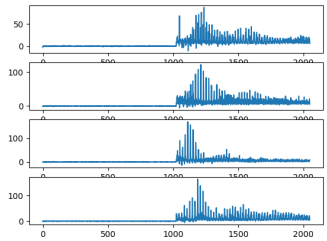
Figure 3.2: Mean squared error loss versus factors of variation

3.2.2 Sampling the Latent Space

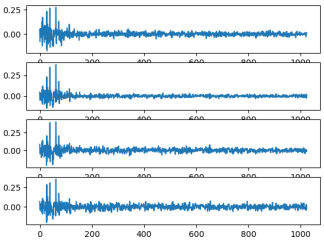
To show how our latent space decodes, the following figures perform random sampling over the latent space with values between -4 and 4. It is again clear how the reconstructed output tends heavily towards the lower frequencies. Additional images showing mesh grids of latent space sampling are shown in appendix A.



(a) Spectrum



(b) Spectrum + phase



(c) MDCT

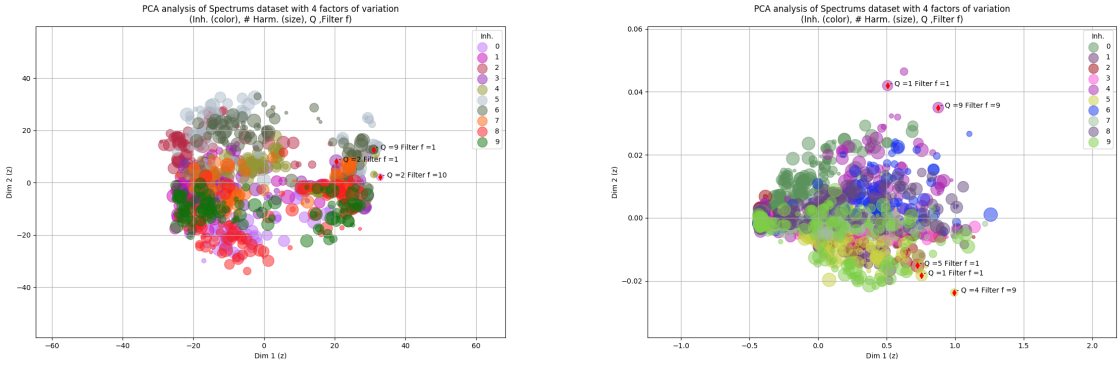
Figure 3.3: Random sampling latent space

Chapter 4

Latent Space Regularization and Dimensionality Reduction

4.1 PCA analysis and exploration

The VAE regularizes better data information when the number of latent dimensions is more important ; however this can be impracticable with user control. We can fix this by applying manifold analysis such as PCA [3] to get the most relevant 2D-embedding from an higher-dimensioned space. It also allows us to analyze the distribution of our datasets in the latent space. The results are presented below. We can clearly see how for unnormalized data and a high latent space dimension, we achieve a big spread of our dataset in the latent space and a clear clustering for the inharmonicity variational factor after only 200 epochs of training. Normalized data performs worse although clustering around the inharmonicity variational factor can still be noticed while other factors such as filter Q and frequency are not clustered as shown by the labeled data in figure 4.1. One explanation for the improved performance for unnormalized data could be the greater ranges resulting in more accurate calculations in the VAE model as means and variances for normalized data are very small. Unnormalized data also gave much better reconstruction results for spectra. An additional PCA plot for dataset 3 is shown in appendix A.



(a) PCA(15 dim -▷ 2)

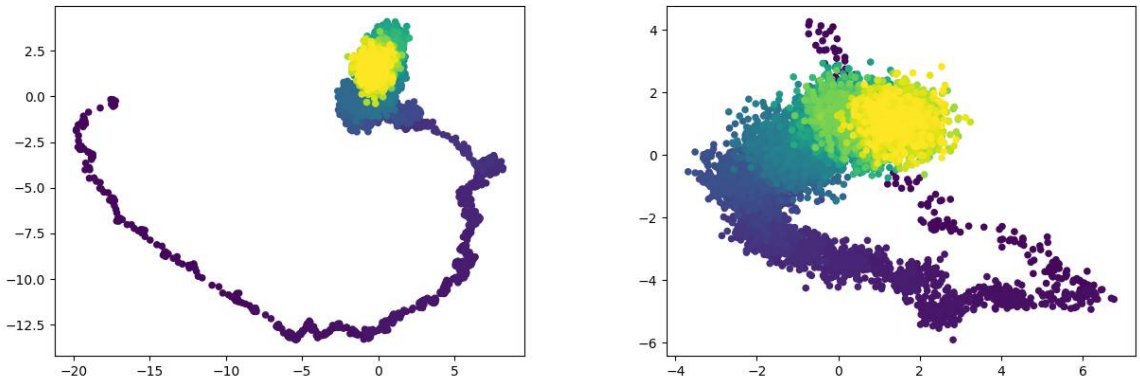
(b) PCA(4 dim -▷ 2)

Figure 4.1: Dimensionality reduction with PCA algorithm

4.2 The β parameter

β represents the scalar multiplying the regularization term, KL loss in our case. A β VAE is a VAE where $\beta > 1$. Increasing its value encourages the model to disentangle the latent space [2]. As the

prior is an isotropic Gaussian, the representation of the latent space is rewarded the most when its components are independent. A disentanglement measure was proposed by [2] which allows to test automatically various parameters including the β hyper-parameter, so that one can extract the combination which maximizes disentanglement.

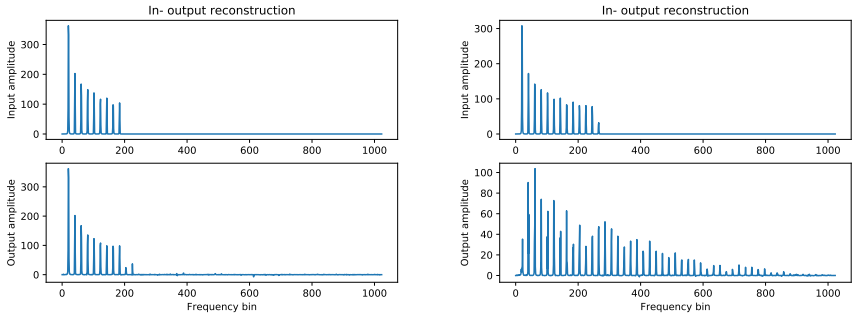


(a) $\beta = 1$ (b) $\beta = 3$

Figure 4.2: Impact of β factor seen with a PCA on the latent space (10 dimensions)

We have chosen to visually test the disentangled quality of our latent space. As a first stage of analysis, $\beta = 1$ was the optimal value reported. When β increases, the latent variables tend to fit more and more a reduced centered gaussian.

If we continue to increase β, reconstruction gets undesired. The regularization term gets to strong compared to the reconstruction term, in the loss. (figure 4.3)



(a) $\beta = 1$ (b) $\beta = 4$

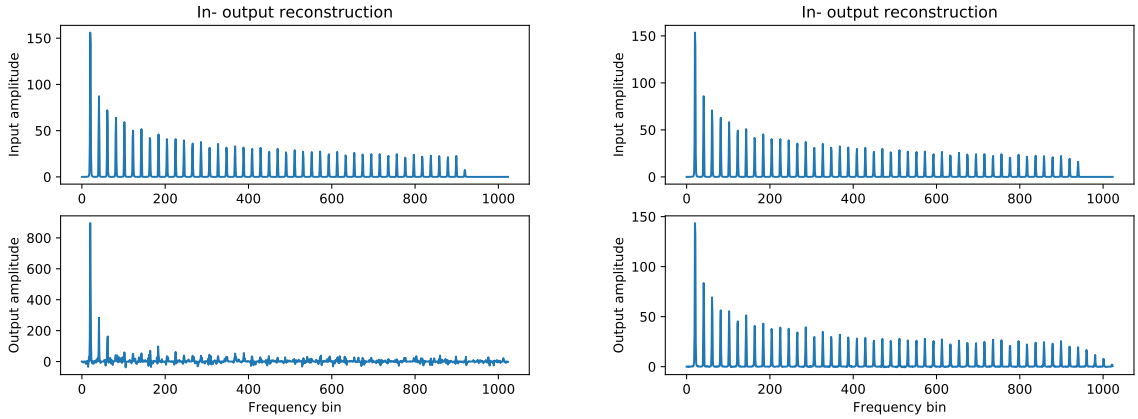
Figure 4.3: β effect on reconstruction

Our interest is on the quality of the sampling of the latent space, and it also gets very noisy with $\beta = 4$ (as shown in figure A.5) and doesn't fit the evolution of the toy dataset. Whereas with $\beta = 1, 2 \; and \; 3$, we can see in figure A.4, that the evolution of the spectra on two randomly taken dimensions is very close to the dataset's evolution, spectral richness increases as the horizontal dimension increases and low frequency spectral richness increases as the vertical dimension increases. The information is still entangled on 10 dimensions but as we can see with figure 4.2, PCA disentangles the 10 dimensions into 2 dimensions where the spectral richness path is clearly observable. Following an inverse PCA, and listening to this 2D path decoded using our real time implementation was really promising.

4.3 Warm-up

Warm up is a technique to introduce the beta parameter smoothly throughout the epochs. The variational regularization term has this negative effect of inhibiting some latent units at the beginning of the training (which stay inactive throughout the training,[2]). Warm up should counteract this effect and improve the ratio between fine reconstruction and efficient regularization. With the 1-factor toy dataset and with a $\beta = 1$, warm up introduces noise in the output so that reconstruction and latent space sampling is very poorly done.

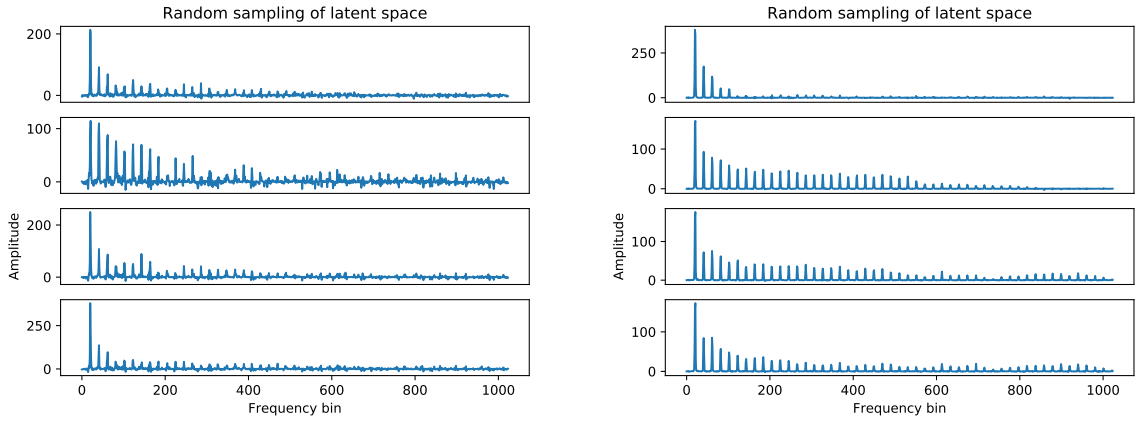
Figure 4.4 shows how reconstruction is badly done with warm up, as opposed to without warm up (for $\beta = 1$ still). The sampling of the latent space is also affected on its noisiness by warm up. We have tried many other combinations ($\beta = 1, 2, 3\ and\ 4$; different latent space dimensions, normalized/unnormalized) but warm up never improves the performance. Warm up is usually useful for deep VAEs, this may be the reason for its ineffectiveness on our single hidden layer model. For the 4- and 5- factors toy dataset, the effect of warm up was also negative or unnoticeable.



(a) With warm up

(b) Without warm up

Figure 4.4: Warm up effect on reconstruction



(a) With warm up

(b) Without warm up

Figure 4.5: Warm up effect on latent sampling

Ultimately, very interesting latent spaces have been generated, using unnormalized magnitude spectra training data, with a low number of latent dimensions (10 or less), small values of β (3 or less) and no warm up. These were exploited and listened to, thanks to our real time generation implementation.

Chapter 5

Sound Generation and Latent Space Trajectories

In order to fulfill the high-level goal of the project, it was necessary to develop a way to use the VAE's learned latent space in order to guide sound synthesis. This was achieved by training the VAE on a spectral representation, chosen both for its capacity to describe the sound's timbral properties and the relative simplicity with which it can be integrated in a real-time synthesis implementation.

5.1 Spectral Representation

The main complication of using a spectral representation for sound generation is the need to treat either complex values or both magnitude and phase information simultaneously. Furthermore, signal reconstruction is extremely phase-sensitive, so different methods had to be tested in order to determine the most appropriate for real-time synthesis. Two general algorithms were tested: first with a sinusoidal oscillator bank, and second using an inverse short-term Fourier transform. The former takes magnitude and phase information and controls oscillators whose frequencies are set relative to the reference fundamental frequency used to generate the VAE's training dataset. The main advantage of this approach is the relative ease with which the oscillators' frequencies can be adjusted with respect to a given played note while controlling the spectral characteristics with the latent space trajectories. However, for a large number of oscillators (e.g. the 1024 frequency bins we were usually working with), interferences between adjacent sinusoidals become significant as soon as there are any discrepancies between the magnitude and phase information. For this, the inverse Fourier transform only introduces additional noise, although the short-term implementation consequently creates irregular and inharmonic reconstructions.
Synthesis was thus tested using phase information trained concatenated to the magnitude spectrum, as explored above, as well as using an adapted Griffin-Lim algorithm to approximate the phase from a VAE trained only on magnitudes. For the short-term Fourier transform, the phase and magnitude are first used to calculate a complex spectrum whose real and imaginary parts are sent separately.

5.2 Real-Time Implementation

The real-time sound synthesis setup was developed in Max/MSP as a Max for Live device communicating with a Python script over an OSC (Open Sound Control) connection in order to facilitate MIDI control of the latent space trajectories. Continuous control messages are used to set the value of the latent variable by assigning one controller number to each latent space dimension, before being sent to the VAE's decoder in order to generate the corresponding spectrum. For the oscillator bank setup, MIDI notes adjust the playing frequencies as described above, while simply acting as a gate for the short-term Fourier transform's output (the re-pitching process being somewhat more complicated in the latter).
Further improvements can be anticipated with respect to these generation methods, as many

clicks, interferences, and irregularities persist, most likely due to inaccuracies in determining the phase spectrum (VAE regeneration is non-deterministic by definition, and with no explicit cross-regularization between the amplitude and phase spectra there is no guarantee that a regenerated spectrum will be physically accurate).

5.3 Latent Space Exploration

The latent space can thus be explored sonically by controlling the latent variable through continuous controllers while playing a note, which translates to a dynamic evolution in the sound's timbre. The training trends analyzed in the previous chapter are globally confirmed perceptively: maximum spectral richness is found around the origin (zero-point), and different latent dimensions organize the evolution of this richness in different ways.
Depending on the structure of the dataset used for training the VAE, some dimensions can correspond to certain mathematical sound generation parameters, while others encode undetermined yet perceptible timbral evolutions, and some seem altogether "useless" in that they do not control any sonically significant characteristics. Starting with a parametrically "weak" dataset and leading to a larger latent space seems to increase the likelihood of finding unexpected evolutions, but also generates more "dead" dimensions.

Appendix A

Additional Figures

Latent space's generated spectra for spectra, spectra + phase and MDCT data

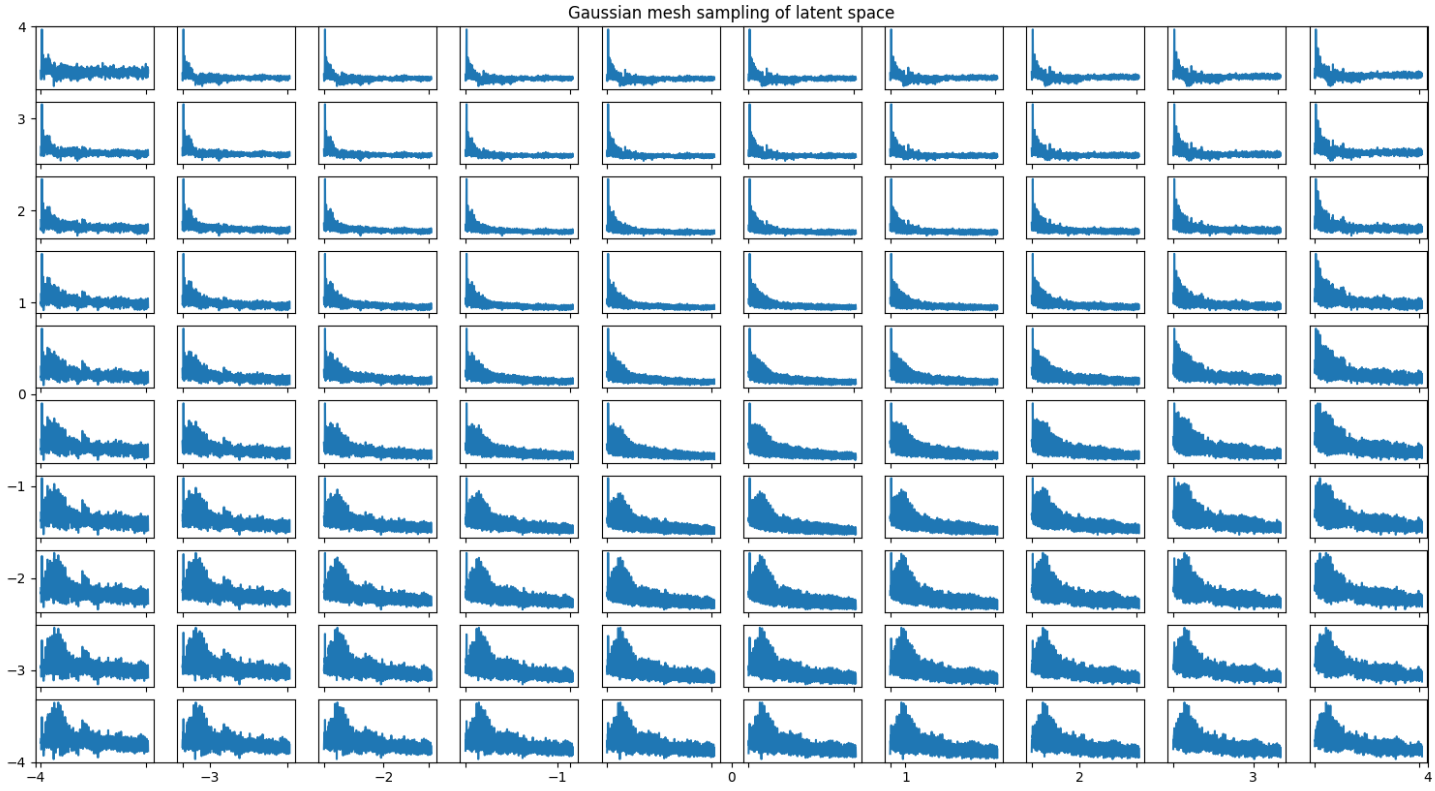


Figure A.1: Gaussian mesh sampling (z in [-4,4]) for spectra data

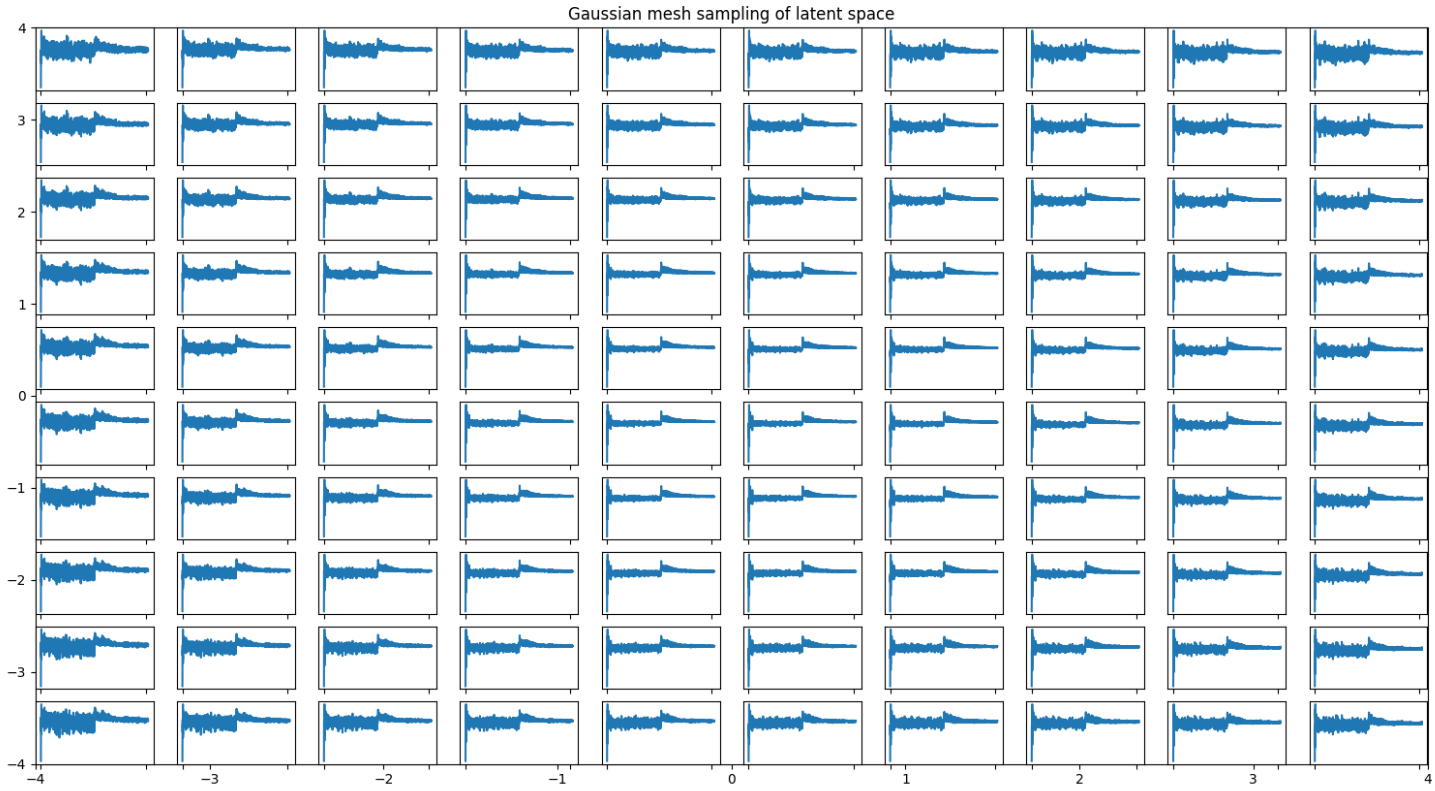


Figure A.2: Gaussian mesh sampling (z in [-4,4]) for spectra + phase data

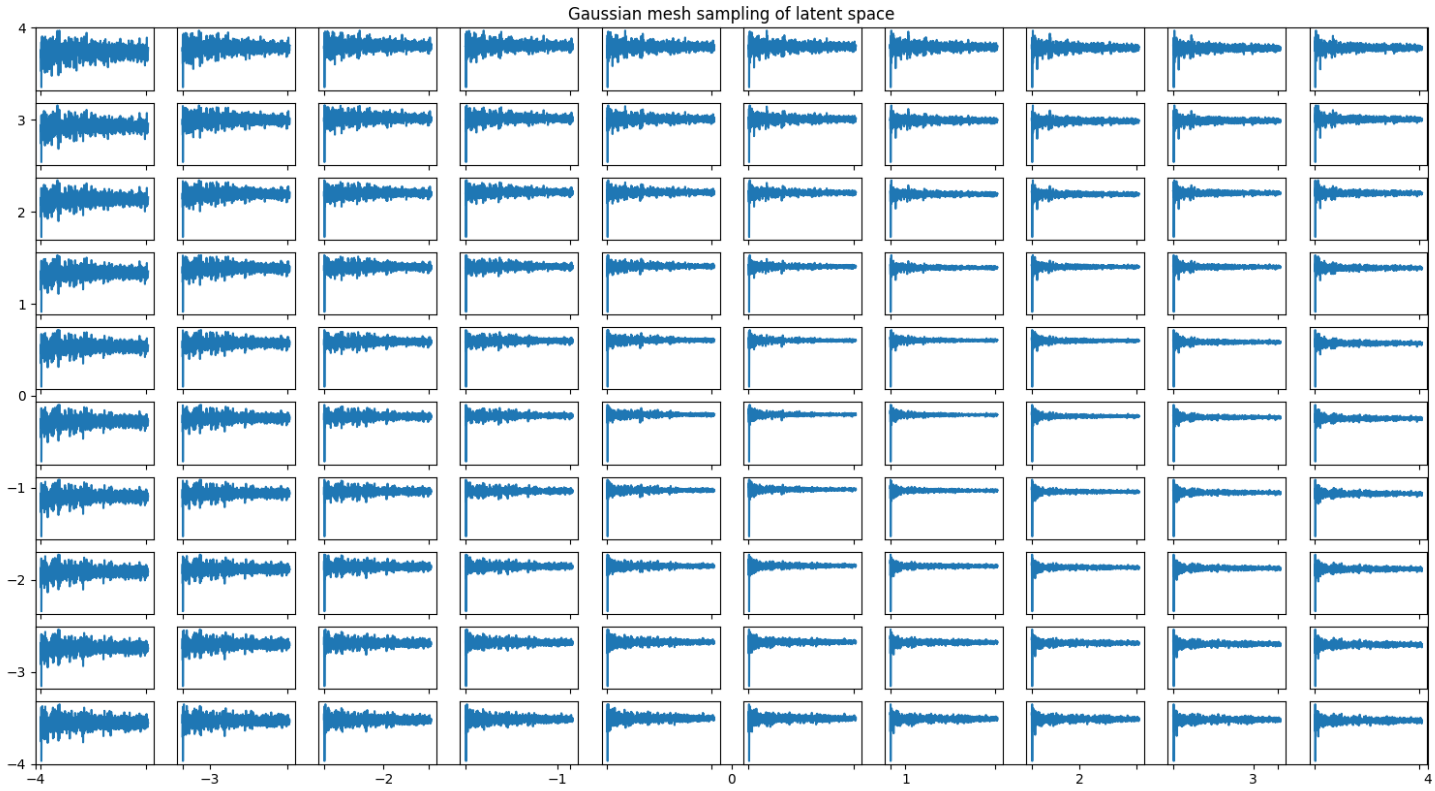


Figure A.3: Gaussian mesh sampling (z in [-4,4]) for MDCT data

Latent space's generated spectra : $\beta = 3$ - Very similar results for $\beta = 1$ *or* 2

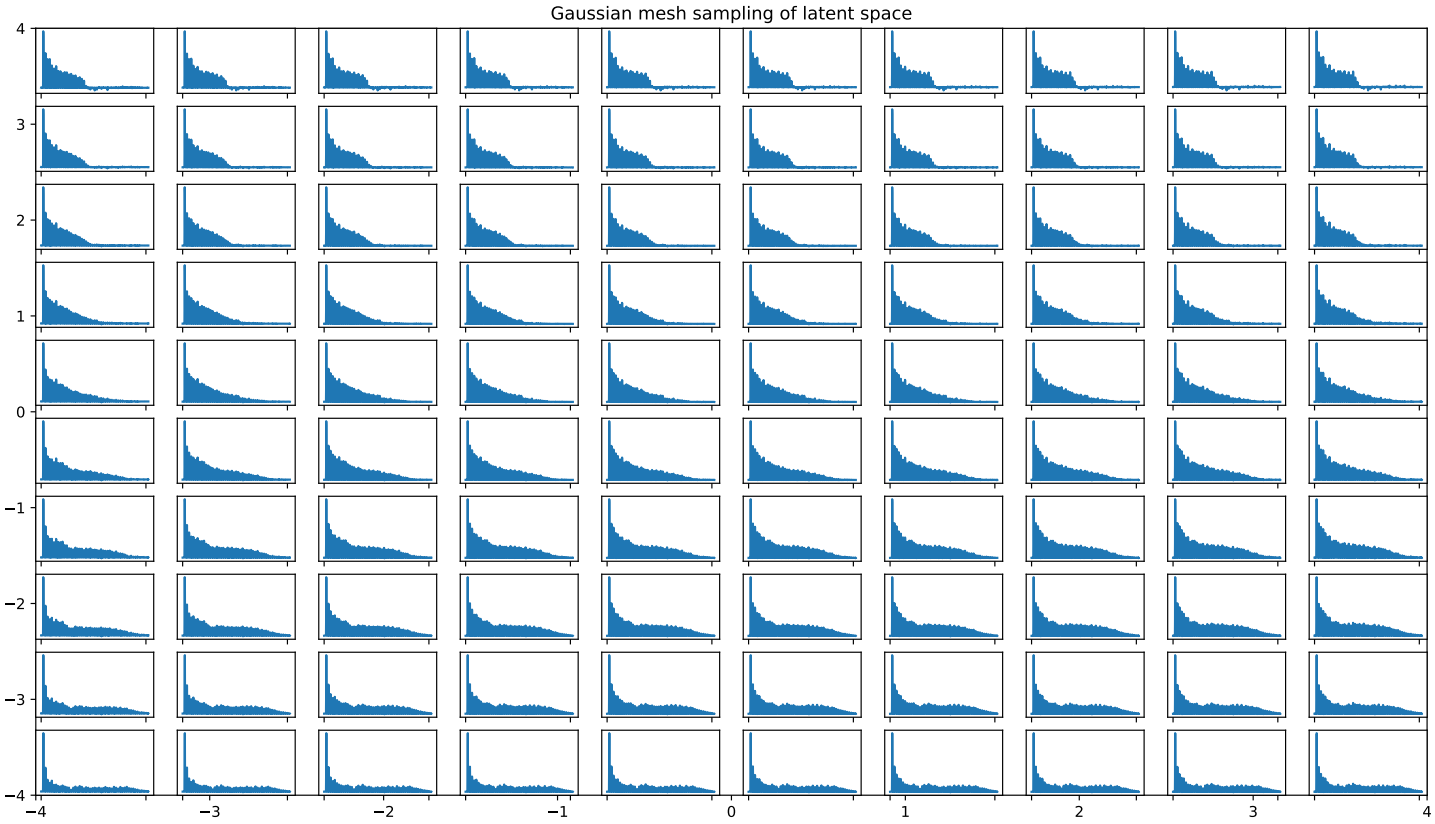


Figure A.4: $\beta = 3$ - Latent space's sampling over two random dimensions

Latent space's generated spectra : $\beta = 4$

Figure A.5: $\beta = 4$ non-optimal hyper parameter - Latent space's sampling over two random dimensions

PCA for dataset 3 (FM and Moog filter)

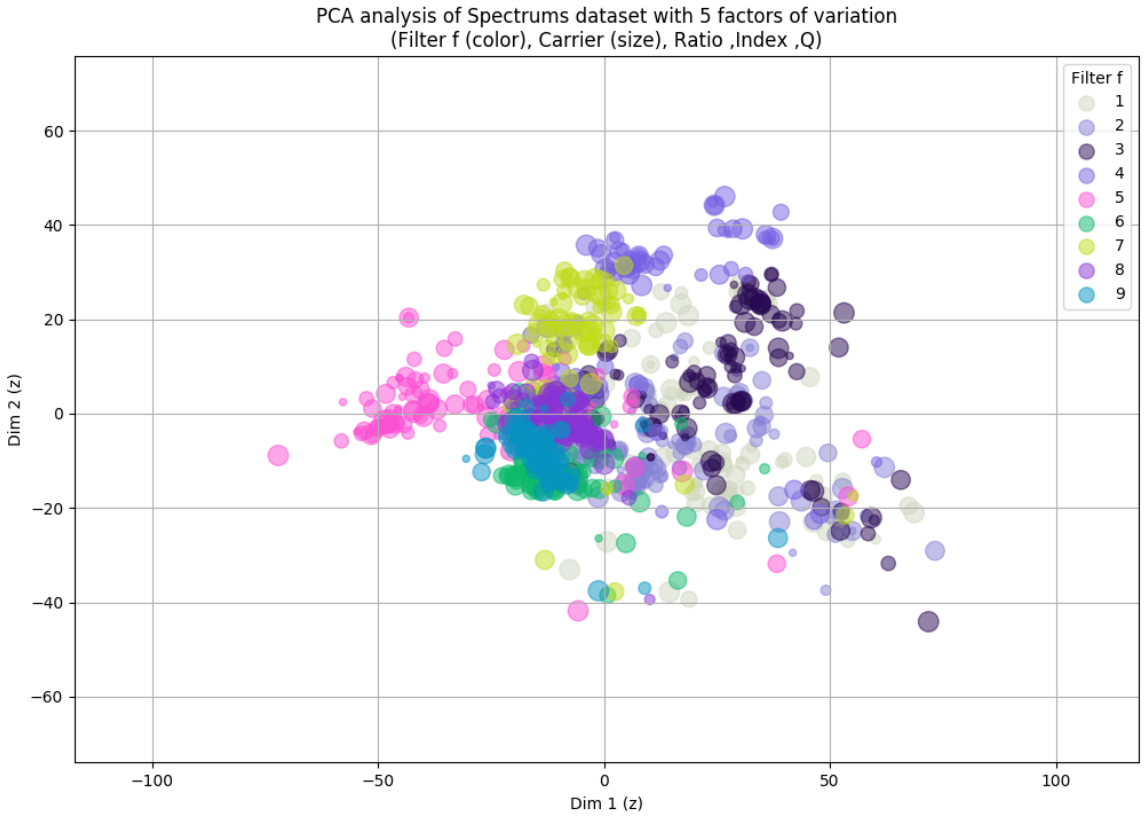


Figure A.6: PCA(15 -▷ 2) latent space spectrum dataset 3 (unnormalized)

Bibliography

[1] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. 12 2013.

[2] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, Alexander Lerchner, and Google Deepmind. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. *Iclr*, 2017.

[3] M. E. Tipping and Christopher Bishop. Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society, Series B*, 21/3, 1 1999.