UNIVERSITY OF AMSTERDAM

MSc Artificial Intelligence

Master Thesis

Representing Audio in a Distribution of Continuous Functions

by

Jan Zuiderveld

10798463

July 16, 2021

36 ECTS
January 2021 — July 2021

Daily supervisor:
Marco Federici MSc

Assessor:
dr. ir. Erik Bekkers

AMSTERDAM MACHINE LEARNING LAB
UNIVERSITY OF AMSTERDAM

Abstract

A recent line of research in computer vision and graphics replaces traditional discrete representations of objects, scene geometry, and appearance with continuous functions parameterized by deep fully-connected networks. These fully connected networks, referred to as *implicit neural representations* (INRs), can be trained to represent any geometric object by mapping input coordinates to structural information at input locations. In contrast with other neural methods for representing geometric objects, representation quality scales with object complexity, independent of resolution. INRs have shown to be very effective at representing images, videos, waveforms, distance functions, radiance fields and various other types of data achieving state-of-the-art results across a variety of tasks such as novel view synthesis, solving differential equations and super-resolution.

INRs can be conditioned by concatenating latent codes to input coordinates, modulating intermediate layer activations or generating network parameters using a hypernetwork. This allows INRs to function as a prior over a distribution of functions. At the time of writing published applications of conditional INRs in the generative domain are scarce, and to our knowledge no research has been done applying conditional INRs in the field of audio synthesis.

In this work we compare different ways to parameterize and condition INRs for the task of representing a distribution of audio waveforms. Aiming to narrow down which have the right amount of flexibility and carry the best inductive bias for modelling distributions of high-frequency one-dimensional continuous functions and their perceptual qualities as audio.

We conclude that conditional INRs show great potential for representing distributions of audio waveforms with perceptual- and absolute fidelity. Previously proposed sinuisoidal INRs with FiLM conditioning signifcantly outperform transposed convolution based architectures with equal parameter budgets. However, the perceptual fidelity is inferior in more uniform datasets due to local waveform inconsistencies, a side effect of the high expressivity of these models, which is amplified when optimizing hyperparameters in short training runs.

To this extent we propose and validate methods for coping with these limitations and overcoming the need to optimize hyperparameters altogether in certain datasets. To foster reproducible research, we published the source code of this research on GitHub.[1]

[1] https://github.com/janzuiderveld/continuous-audio-representations

Acknowledgements

During my time at the University of Amsterdam I met, collaborated with, and learned from bright minds: ranging from classmates to PhD students, post-docs and professors. This work would not have been possible without them - this is an acknowledgement: a thank you, for all those who have given their time and expertise to help me grow into the person that I am today - and for that reason I dedicate this work to them.

I am especially grateful to my supervisors for contributing to my progress during the development of this thesis. To Marco Federici MSc, a great mentor, whose feedback was instrumental in seeing this work to completion. I thank you for the weekly hours of discussion and advice that helped me focus on what mattered. And to dr. ir. Erik Bekkers, who often managed to take time out of his demanding schedule to join our weekly meetings and discuss the fundamentals of this research. It has been a pleasure having you as my supervisors.

The last months were marked by many early mornings and long nights. At times, I felt like I had reached a dead end, but the suggestions of my supervisors helped to push me forward when I felt stuck. Although it was sometimes difficult to stay motivated, the intellectual challenge of combining ideas into a coherent theory and experiencing new discoveries resulted in stimulating moments. I have come to appreciate the unpredictability of scientific research and the way you must be able to adapt to remain productive.

Finally, I want to say I feel lucky for having the opportunity to study Artificial Intelligence at the level provided by the University of Amsterdam, and the open-endedness regarding research directions at Amsterdam Machine Learning Lab which allowed me to pursue a niche that I am passionate about.

Contents

1 Introduction

In generative modelling data is often represented by discrete arrays. Images are represented by two-dimensional grids of RGB values, 3D scenes are represented by three-dimensional voxel grids and audio as vectors of discretely sampled waveforms. However, it is often the case that the true underlying signal is continuous. A recent line of research in computer vision and graphics replaces traditional discrete representations of objects, scene geometry, and appearance with continuous functions parameterized by multilayer perceptrons (MLPs).[1]

These fully-connected networks, referred to as *implicit neural representations* (INRs), can be trained to represent any geometric object by mapping input coordinates to structural information at input locations. In contrast with other neural methods for representing geometric objects, the memory required to parameterize the object is independent of resolution, and only scales with its complexity. A corollary of this is that INRs have infinite resolution, as they can be sampled at arbitrary spatial resolutions. INRs have shown to be very effective at representing images, videos, waveforms, distance functions, radiance fields and various other types of data achieving state-of-the-art results across a variety of tasks such as novel view synthesis[2], solving differential equations[3] and super-resolution.[4]

To represent an audio waveform using an INR we define a function $f : \mathbb{R}^1 \rightarrow \mathbb{R}^1$ mapping time coordinates to waveform amplitude values, parameterized by a MLP. During training this network is supervised on sampled amplitude values of the discrete waveform to be represented.

Then, to apply the concept of INRs in the generative domain we can frame generative modelling as learning a distribution of continuous functions.[5] This can be achieved by introducing conditioning methods, for example by concatenating latent codes to input coordinates[6], modulating intermediate layer activations[7] or generating network parameters using a hypernetwork.[8] This allows INRs to function as a prior over a distribution of functions.

In this work, we combine recent advances of INRs in generative modelling and insights on the characteristics of sound represented

[1] Chen and Hao Zhang, *Learning Implicit Fields for Generative Shape Modeling*, 2019; Mescheder et al., *Occupancy Networks*, 2019; Mildenhall et al., *NeRF*, 2020; Park et al., *DeepSDF*, 2019; Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020; Tancik et al., *Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains*, 2020.

[2] C. Liu et al., *NeLF*, 2021.

[3] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

[4] Xu et al., *UltraSR*, 2021.

[5] Dupont et al., *Generative Models as Distributions of Functions*, 2021.

[6] Chen and Hao Zhang, *Learning Implicit Fields for Generative Shape Modeling*, 2019.

[7] Chan et al., *Pi-GAN*, 2020.

[8] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

in the time domain. We narrow down which conditioning methods have the right amount of expressiveness and what INR parameterizations carry the best inductive bias for modelling distributions of high-frequency one-dimensional continuous functions and their perceptual qualities as audio. We compare these results to the performance of transposed convolution based architectures designed for this purpose in a discrete paradigm. More specifically, we aim to answer the following research question:

Main research question

Are conditional INR's suited for representing distributions of audio waveforms when optimizing for perceptual- and absolute reconstruction fidelity?

To answer the main research question, we address the following subquestions:

Subquestions

- ○ What latent embedding inference method is optimal for exploring the representation characteristics specific to various conditional INR parameterizations?
- ◊ Are previously proposed conditional INR parameterizations capable of representing a distribution of audio waveforms with similar perceptual- and absolute reconstruction fidelity as transposed convolution based architectures designed for this purpose?
- □ Which conditional INR model hyperparameters[9] are of significant influence in perceptual- and absolute reconstruction fidelity when representing different distributions of audio waveforms?
- △ What are the main shortcomings of audio waveform representations learned by conditional INR's, the process required to learn these representations, and how can these shortcomings be addressed?

[9] We consider the following INR model hyperparameters:
- Conditioning methods
- Activation functions
- Latent mapping network depth
- Network shape

By answering these questions we hope to shed light on the potential and amount of work still needed for applying implicit architectures in generative models for novel waveform synthesis with competitive results, and by doing so guiding future research to promising directions within this area.

We conclude that conditional INRs show great potential for representing distributions of audio waveforms with perceptual- and absolute fidelity. We summarize the key contributions of this work:

- ○ Autodecoders are generally better suited for latent embedding inference than autoencoders when exploring the representation characteristics of INR parameterizations.

◊ π-GAN, a previously proposed sinuisoidal INR with FiLM conditioning, signifcantly outperforms previously proposed transposed convolution based architecture with equal parameter budgets in representing distributions of audio waveforms with absolute fidelity.

◊ The perceptual fidelity of representations learned by π-GAN is inferior in more uniform datasets due to local waveform inconsistencies, a side effect of the high expressivity of sinuisoidal INRs with FiLM conditioning.

□ Conditional INR model hyperparameters that reduce the expressiveness of the model can be beneficial in such scenarios. Decreasing network depth while increasing network width, proved to be a very effective adaptation generally improving fidelity.

□ INRs only conditioned via concatenation or without sinusoidal nonlinearities in the input layer are incapable of representing any high frequency content such as audio waveforms.

△ An important shortcoming of conditional sinuisoidal INRs is the high sensitivity to activation scaling hyperparameters. Optimizing these in short training runs introduces expressivity pressure, amplifying local waveform inconsistencies.

△ To this extent we make the following contributions:[10]

1. We propose and validate post hoc methods for taming the expressivity of activation scaling hyperparameters found optimal in short training runs, with success in two out of three tested datasets.
2. We propose and validate a method for removing the need to optimize activation scaling hyperparameters altogether. One out of three datasets shows robust and significant perceptual fidelity gains.

[10] Our source code is publicly available at https://github.com/janzuiderveld/continuous-audio-representations

2 Background

In this chapter, we recapitulate the foundations of which this work is build and motivated upon, spanning across multiple fields of research.

In Section 2.1 we outline intuitions about the perception and origin of sound, and how oscillations are fundamental to its structure.

Then, in section 2.2 we introduce the reader to the current dominant approaches in the field of audio synthesis and their associated inductive biases and limitations.

Finally, in Section 2.3 we discuss Implicit neural representations (INRs), a replacement for traditional discrete representations and argue for their potential as a backbone for neural audio synthesis architectures.

Synthesizing audio has many practical applications in creative sound design for music and film. By using generative architectures, it is possible for artists to "create", explore, and morph between vast amounts of sounds in intuitive ways.[1] However, the high temporal resolution of audio and our perceptual sensitivity to small irregularities in waveforms make neural audio synthesis a complex and computationally intensive task. As such, there is still significant room for improvement in this research area.

[1] E.g. initialize latent embedding by optimizing for some given sound, then explore the space by taking steps in the direction of other given sounds or previously explored regions in the latent space.

Although neural networks are theoretically proven to be universal function approximators in the asymptotic limit[2], integrating inductive biases aligned with the structure and patterns present in the data domain, such as convolution[3], recurrence[4], and self-attention[5] has proven invaluable for the success of neural architectures. Such constraints improve data efficiency and generalization by exploiting symmetries in the structure of data, and can be essential when scaling neural networks. However, current work in neural audio synthesis does not show a lot of awareness in that regard.

[2] Hornik et al., "Multilayer Feedforward Networks Are Universal Approximators", 1989.

[3] LeCun et al., "Backpropagation Applied to Handwritten Zip Code Recognition", 1989.

[4] Sutskever et al., "Generating Text with Recurrent Neural Networks", n.d.

[5] Vaswani et al., *Attention Is All You Need*, 2017.

With this in mind, we briefly outline intuitions how sound is generated by nature, how we perceive it and how oscillations are fundamental to its structure. Then, we elaborate on the current dominant methods in neural audio synthesis and their inductive biases. Finally, we argue that implicit neural representations with periodic activation functions are a promising backbone for neural audio synthesis architectures.

2.1 The nature of sound

Practically everything in the universe, from planets to objects, molecules and atoms, has a natural tendency to vibrate. Elastic forces counteract the movement of particles, resulting in harmonic oscillations between kinetic and potential energy. Neighboring matter forms a propagating medium by synchronizing to these oscillations. The amount of entrainment differs between frequencies and depends on characteristics of the medium.

Accordingly, our auditory perception has evolved to be sensitive to the energy of phase-consistent oscillations at frequencies that are propagated by air molecules[6]. Consequently, human hearing is indifferent to the absolute air pressure or waveform amplitude at specific moments in time, but is purely shaped by the presence of frequencies, like audio representations in the frequency domain.

[6] Moerel et al., "Processing of Natural Sounds in Human Auditory Cortex", 2012; Theunissen and Elie, "Neural Processing of Natural Sounds", 2014.

Images can also be represented in the frequency domain as a combination of oscillations, but when doing so, energies do not correspond to a specific objects, and as such image processing depends less on frequency analysis[7]. Another way to illustrate structural differences between audio and images is by examining the most important components for reconstructing samples using principal component analysis. Figure 2.1 shows the first eight principal components for patches from natural images and slices from speech[8]. This once again shows how periodic patterns are unusual in natural images but a fundamental structure in audio.

[7] However, There are still applications of Fourier analysis in image processing such as image compression and denoising, and optimizing correlation and convolution operations.

[8] Donahue et al., *Adversarial Audio Synthesis*, 2019.

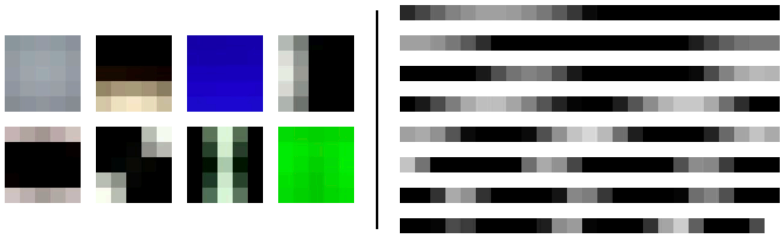


Figure 2.1: The first eight principal components for 5x5 patches from natural images and length-25 audio slices from speech. While the principal components of images generally capture intensity, gradient, and edge characteristics, those from audio form a periodic basis that decompose the audio into constituent frequency bands. Image taken from Donahue et al., Adverserial Audio Synthesis, 2019.

2.2 Neural audio synthesis backbones

Current dominant neural architectures for audio synthesis are based on transposed convolutions or autoregressive architectures. Autoregressive architectures were the first models capable of generating audio with reasonable quality[9], and still produce competitive results, but they are significantly more computationally expensive to train and sample and lack global latent structure. Transposed convolution based architectures have global latent conditioning and

[9] Oord, Dieleman, et al., *WaveNet*, 2016.

efficient parallel sampling, but synthesize less locally consistent waveforms. Both architecture types can generate audio directly as a waveform, but transposed convolution based architectures often rely on spectrogram representations (visual representations of the frequency spectrum over time), which in turn can be transformed to waveforms.

Transposed convolution based architectures that generate waveforms directly with one-dimensional convolutions (e.g. SING[10], Blow[11] and WaveGAN[12]) must precisely align "wave packets"[13] between different convolution steps. Since audio oscillates at many frequencies, all with different periods relative to the stride of the convolutions, these models must learn kernels to cover all variations in phase, which can be a heavy burden in scenarios with more diverse samples.

[10] Défossez et al., *SING*, 2018.

[11] Serrà et al., *Blow*, 2019.

[12] Donahue et al., *Adversarial Audio Synthesis*, 2019.

[13] This term is borrowed from physics, the learned kernels in transposed convolution based architectures look like wave packets, but they are technically not the same.

GANSynth[14] and SpecGAN[15] are transposed convolution based architectures that generate intermediate spectrograms. These models suffer from similar problems when transforming spectrograms to waveforms due to the difficulty of modelling phase information. Additionally, these models must also learn to inhibit artificial frequencies introduced by windowing methods- and frequencies that do not align with the Fourier basis functions used to transform the original waveforms to spectrograms, also known as "spectral leakage".

[14] Engel, Agrawal, et al., *GANSynth*, 2019.

[15] Donahue et al., *Adversarial Audio Synthesis*, 2019.

Autoregressive architectures solve fine-scale waveform coherence by generating waveform amplitude samples one by one, causally. By doing so, they are not constrained by the phase alignment problems described in the last paragraph and are better at synthesizing locally consistent waveforms in more diverse datasets. But this comes at the cost of less global coherence[16], higher memory requirements, slow serial synthesis of amplitude samples and being restricted to objective functions that are calculated per amplitude sample, such as mean squared error (MSE).

[16] The total amount of amplitude measurements in a few seconds of audio is considered very long in the paradigm of sequence modelling.

Our perception of sound is indifferent to absolute amplitude values and shaped purely by the presence of phase-consistent oscillations. As a result, for any sound we perceive many waveform perturbations with similar spectral content exist. By optimizing models to reproduce a small subset of the set of waveforms that we perceive as identical[17] the problem at hand is constrained in unnecessary ways.

[17] E.g. by restricting models to use a limited set of phase matching kernels or objective functions that are calculated per amplitude sample.

We conclude that both transposed convolutions based- and autoregressive architectures, the dominant approaches in neural audio synthesis as of today, show several shortcomings that impede learning in the problem at hand and lack clear inductive biases aligning with the fundamental structure of the origin-, and our perception of sound: oscillations.

2.3 Implicit neural representations

Implicit neural representations (INRs) are neural networks used to approximate low dimensional functions, which can be trained to represent any geometric object by mapping input coordinates to structural information at input locations.[18] In traditional, discrete ways of representing objects, the required memory to store data scales exponentially with its resolution. In contrast, for functional representations, such as INRs, the memory required to parameterize the signal is independent of spatial resolution, and only scales with the complexity of the underlying signal.

[18] See chapter 1 for an extended introduction to the concept of implicit neural representations, and chapter 3 for examples of proposed parameterization details and applications.

Considering these properties, INRs do not seem ideal for representing audio waveforms. The memory scaling of discrete representations relative to resolution are dependent on the dimension of the data; Quadratically for images, cubically for voxel grids but only linearly for waveforms. Additionally, audio waveforms appear as complex signals; high frequency functions with much detail.

However, the realization that our perception and the origin of sound relates strongly to its spectral content can offer a different perspective. Transforming a signal to the frequency domain amounts to finding the composition of different harmonically related sinusoidal waves (the Fourier basis functions) present in the signal. Audio waveforms (and especially musical ones) transformed to the frequency domain appear significantly more simple. This indicates that, with the right *ingredients*, or basis functions, audio waveforms are not overly complex to construct. Under this assumption, functional representations could create significant memory gains for discrete waveform representations, even though they already scale linearly with resolution.

Then, the question arises; what are the right *ingredients* to construct audio waveforms? Bringing us back to the fundamental question in arguably any machine learning research, what *ingredients* create the right inductive biases for our problem?

In the case of an INR parameterized by a MLP, nonlinearities determine the basis functions available for approximating functions. This makes a strong case for the usage of MLPs with a sinusoidal positional encoding of input coordinates in combination with ReLU activation functions[19] (ReLU P.E. MLPs), making the network operate in a sparse Fourier basis, aligning with the oscillatory nature of sound.

[19] Mildenhall et al., *NeRF*, 2020; Tancik et al., *Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains*, 2020.

However, Sitzmann et al. show that due to the fact that ReLU nonlinearities are piecewise linear, INRs parameterized by ReLU P.E. MLPs are significantly less expressive, resulting in difficulties when fitting high frequency components and corrupted representations of the Laplacian of signals. They propose to solve this by

using sinusoidal activation functions throughout representation networks (SIRENs), and report significant gains in the MSE, or absolute fidelity of learned waveform representations[20], see figure 4.7.

SIRENs are a deep composition of sinusoids with large activation scaling, which show very chaotic behaviour. A piecewise linear combination of sinusoids closely approximates how we perceive sound. On the surface, this seems to provide a better inductive bias for sound synthesis. Yet, in this work we focus predominantly on SIRENs as they exhibit better empirical foundations compared to ReLU P.E. MLPs for representing audio, and better expressiveness when representing distributions of functions under feature-wise linear modulation (FiLM) conditioning, as shown in the ablations studies of π-GAN[21], see table 4.1.

[20] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

[21] Chan et al., *Pi-GAN*, 2020.

3 Related work

In this chapter, we give an overview of the development of implicit neural representations, proposed approaches for applications in the generative domain as conditional INRs and the development of neural architectures for audio synthesis.

3.1 Generative implicit neural representations

The concept of implicit neural representations (INRs) was initially proposed in 2007 by Stanley[1]. In this work the weights of very small INRs are sampled using an interactive evolutionary approach guided by aggregated human feedback.[2]

3.1.1 The revival of implicit neural representations

After a long period of silence, in 2019, three different concurrent works proposed using conditional INRs for representing a distribution of three-dimensional shapes: IM-NET[3], DeepSDF[4] and Occupancy Networks.[5] All three papers report results of training on point cloud datasets of a single category of shapes from ShapeNet[6] and reconstructing unseen shapes within the same category, for different categories overlapping between the works. This is achieved by employing the INR as a decoder in a variational autoencoder[7] (VAE).

However, each work reports results of different ShapeNet sets (ShapeNet-Core[8], ShapeNet, 3D-R2N2[9]), and different encoders (PointNet[10], unreported, extended PointNet). Ignoring these differences and judging by metrics overlapping between the papers, IM-NET and Occupancy Networks show similar quality, and both outperform DeepSDF.[11] DeepSDF is the only architecture trained to represent shape surfaces as signed distance functions, whereas IM-NET and Occupancy networks do this by learning a decision boundary.

IM-NET consists of a 6 layer ReLU MLP with an exponentially shrinking amount of hidden units (starting at 2048) and is conditioned by skip connections of the concatenated input and latent vectors until the single last layer. Besides the previously mentioned

[1] Stanley, "Compositional Pattern Producing Networks", 2007.

[2] Interestingly, this served as an experiment highlighting the unique sense of *interestingness* we possess as humans. Stanley believes this sense has been crucial for the rapid technological development seen in our history, and that it is a manifestation of the open-endedness of evolution. He argues that since evolution created biological intelligence in an environment with no explicit goals, this open-endedness could also be a prerequisite for creating general artificial intelligence.

[3] Chen and Hao Zhang, *Learning Implicit Fields for Generative Shape Modeling*, 2019.

[4] Park et al., *DeepSDF*, 2019.

[5] Mescheder et al., *Occupancy Networks*, 2019.

[6] Chang et al., *ShapeNet*, 2015.

[7] Kingma and Welling, *Auto-Encoding Variational Bayes*, 2014.

[8] Häne et al., *Hierarchical Surface Prediction for 3D Object Reconstruction*, 2017.

[9] Choy et al., *3D-R2N2*, 2016.

[10] Qi et al., *PointNet*, 2017.

[11] IM-NET uses roughly 4×10^6, DeepSDF 2×10^6 and Occupancy Networks 1×10^6 parameters.

VAE shape experiments, the authors also report experiments of IM-NET in a generative adversarial network[12] (GAN), and in both generative frameworks trained on MNIST[13]. In terms of diversity and reconstruction quality IM-NET outperforms transposed convolution based counterparts in GAN setups for shapes, and in both generative setups for MNIST.[14] Results of IM-NET and a standard convolution based decoder in an AE setup trained on a translated objects image dataset are shown in Fig 3.1. This result, among others, seems to suggest a strong geometric prior in INRs.

DeepSDF consists of an 8 layer ReLU MLP, with 512 hidden units throughout the network. Conditioned by concatenation, with a single latent skip of concatenated input and latent vectors to the middle of the network. Besides the previously mentioned VAE shape experiments the authors report experiments in an autodecoder setup, a latent embedding inference method proposed in the same paper.[15] In reported experimental results of partially observed shapes in an autodecoder setup DeepSDF consistently outperforms 3D-EPN[16].

Occupancy Networks consist of 5 "full pre-activation ResNet-blocks"[17] with conditional batch normalization[18,19] as the conditioning mechanism. Each of these ResNet blocks consist of 3 fully connected ReLU layers each, resulting in a total of 17 layers in the INR. Besides the VAE shape experiments the authors report promising results in single-image 3D reconstruction and voxel super-resolution.

3.1.2 Fourier encodings in implicit neural representations

Soon after, it became clear that INRs showed difficulties learning high frequency signal components, relating to the phenomenon of *spectral bias*[20].[21] Mildenhall et al.[22] and Tancik et al.[23] concurrently showed that INRs can overcome this by using a sinusoidal positional encoding on the input features (ReLU P.E. MLPs). This approach was extended by Sitzmann et al.[24], who proposed an implementation of INRs with only sinusoidal activation functions. These networks, Sinusoidal representation networks (SIRENs) show significantly more expressivity than ReLU P.E. MLPs, outperforming the latter in representing audio waveforms, see figure 4.7.

Neural Radiance Field (NeRF)[25] is the first work that used INRs as a radiance field, rendering novel viewpoints trained on images with camera coordinates and viewing angles. The authors find that in the basic implementation, optimizing a neural radiance field representation for a complex scene does not converge to a sufficiently high resolution representation and is inefficient in the required number of samples per camera ray. To alleviate this issue the authors proposed using ReLU P.E. MLPs, being one of the first applying this concept in a practical setup.

Generative Radiance Fields (GRAF)[26] builds upon NeRF by

[12] Goodfellow et al., *Generative Adversarial Networks*, 2014.

[13] Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]", 2012.

[14] We compare IM-NET in our baseline experiments, see section 4.6 for more details.

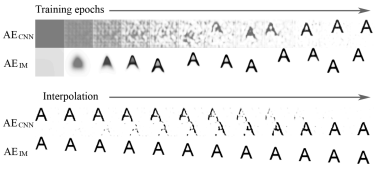


Figure 3.1: CNN-based decoder vs. IM-NET decoder trained in an autoencoder setup with CNN decoder on a synthesized dataset of letter A's on white background.

[15] We do many experiments with this method, see section 4.2 for more details

[16] Dai et al., *Shape Completion Using 3D-Encoder-Predictor CNNs and Shape Synthesis*, 2017.

[17] He et al., *Identity Mappings in Deep Residual Networks*, 2016.

[18] De Vries et al., *Modulating Early Visual Processing by Language*, 2017.

[19] Conditional batch normalization is practically identical to feature wise linear modulation (FiLM), but applies batch normalization as well. See section 4.3.

[20] Rahaman, Baratin, Arpit, Draxler, Lin, F. A. Hamprecht, et al., "On the Spectral Bias of Neural Networks", n.d.

[21] Spectral bias is an implicit regularization inherent to the learning process of gradient descent, prioritizing low-frequency modes in the function space. Potentially an important factor in the contrasting success of overparameterized deep neural networks of with the traditional notions of model complexity.

[22] Mildenhall et al., *NeRF*, 2020.

[23] Tancik et al., *Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains*, 2020.

[24] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

[25] Mildenhall et al., *NeRF*, 2020.

[26] Schwarz et al., "GRAF: Generative Radiance Fields for 3D-Aware Image Synthesis", n.d.

introducing conditioning by concatenation in the architecture, making it the first of its kind to be employed in a generative adversarial setup. Instead of directly generating a 2D image from the input noise $\mathbf{z}_i$, these methods produce an implicit 3D radiance field conditioned on $\mathbf{z}_i$. This radiance field is rendered using volume rendering to produce a 2D image from some camera pose . GRAF improved on previous works in multiview consistence, but is still limited to simple scenes of single objects.

Periodic Implicit GAN (π-GAN)[27] replaces the radiance field representation and conditioning method in GRAF by a SIREN[28] with FiLM[29] conditioning. It improves significantly on GRAF and achieves state-of-the-art results in 3D aware image synthesis. In their ablation studies, the authors show that FiLM conditioning works especially well with SIRENs compared to conditioning by concatenation.[30]

[27] Chan et al., *Pi-GAN*, 2020.

[28] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

[29] Perez et al., *FiLM*, 2017.

[30] All of our experiments include variations of π-GAN adapted for representing a distribution functions with a single input and output dimension. See section 4.6.2 for more information on π-GAN and section 4.3 for a more thorough explanation of FiLM conditioning.

3.1.3 Hypernetworks & implicit neural representations

Hypernetworks generate the parameters of another network.[31,32] Sitzmann et al.[33] were one of the first to propose using hypernetworks for generating INR parameters. The authors report experiments on the reconstruction and generation of small (32×32) images of faces of the celebA dataset[34] in a VAE setup using SIRENs. These experiments functioned as a proof of concept, and do not show results competitive with other methods.

[31] Ha et al., *HyperNetworks*, 2016.

[32] For an introduction to hypernetworks, see section 4.3.

[33] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

[34] Z. Liu et al., *Deep Learning Face Attributes in the Wild*, 2015.

Anokhin et al.[35] proposed an INR based method for image generation achieving state-of-the-art results on some image datasets, but their method depends on learned pixel embeddings making the output bound to a specific resolution.

[35] Anokhin et al., *Image Generators with Conditionally-Independent Pixel Synthesis*, 2020.

Skorokhodov et al.[36] are the first to report truly continuous image generation competitive with traditional transposed convolution based decoders by using of a hypernetwork for generating INRs in a GAN setup. Their proposed models utilize ReLU P.E. MLPs as the generator backbone, and uses the discriminator of StyleGAN2[37]. Special features of their method important for its success include:

[36] Skorokhodov et al., *Adversarial Generation of Continuous Images*, 2020.

[37] Karras, Laine, Aittala, et al., "Analyzing and Improving the Image Quality of StyleGAN", 2020.

- Factorizing the output of their HyperNetwork. The authors calculate the final weight matrix $\boldsymbol{W}^\ell$ by defining a shared parameter matrix $\boldsymbol{W}_s^\ell$ that is multiplied by a low rank matrix $\boldsymbol{W}_h^\ell$ produced by a HyperNetwork: $\boldsymbol{W}^\ell = \boldsymbol{W}_s^\ell \odot \sigma\left(\boldsymbol{W}_h^\ell\right)$.
- Utilizing a multi-scale architecture which allows to efficiently represent high-resolution images.[38]

[38] Note that this does not relate to progressive growing of the generator or discriminator. This is not applied in their method, while being crucial in most state-ot-the-art image generation GANs.

Dupont et al. (2021) extends previous INR generative adversarial approaches by using a discriminator independent of resolution as well. The authors achieve this by interpreting image pixels as point clouds, and classifying them using the PointConv framework[39,40].

[39] Wu et al., *PointConv*, 2020.

[40] The authors argue that PointConv layers are suited because they are translation equivariant and permutation invariant. They report that when sampled on a regular grid, PointConv networks closely match the performance of regular CNNs.

The authors mention that sine activations show strong potential for usage in image generators, but they do not report experimental results due to initialization difficulties associated with hypernetwork generated weights.

3.2 Generative modelling of audio

Reasonable audio waveform synthesis was considered unachievable for a long time, and the success of WaveNet[41] in 2016 created a surge in interest of audio waveform modelling. See fig. 3.2 for a chronological overview of advances of raw audio synthesis in musical context.

[41] Oord, Dieleman, et al., *WaveNet*, 2016.

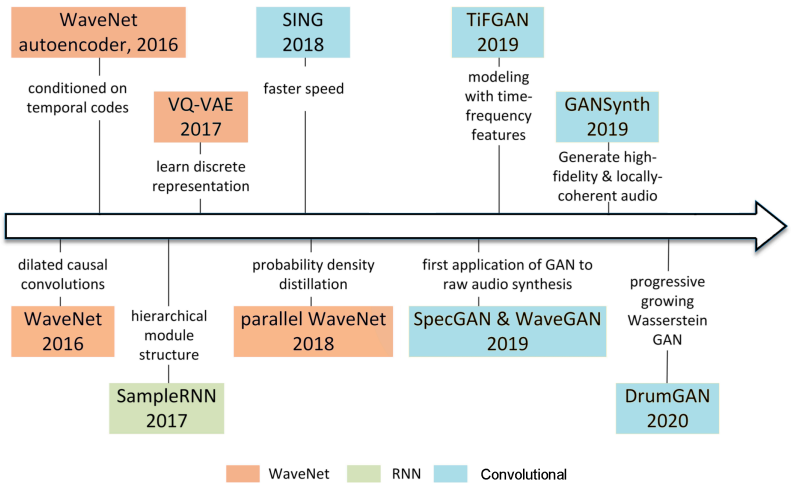


Figure 3.2: Chronological overview of advances of raw audio synthesis in musical context. Different colors represent different model architectures.

WaveNet[42] is an autoregressive model based on PixelCNN[43]. It utilizes dilated causal convolutions to create waveforms. WaveNet can generate novel and highly realistic audio clips due to its local consistency, but the generated sounds can lack long-term consistency, resulting in second-to-second variations. Another downside of the autoregressive nature of the model is that it makes generation computationally heavy.[44]

[42] Oord, Dieleman, et al., *WaveNet*, 2016.

[43] Oord, Kalchbrenner, et al., *Conditional Image Generation with PixelCNN Decoders*, 2016.

[44] See section 2.2 for a more in-depth discussion of autoregressive models applied to neural audio synthesis.

WaveNet autoencoder[45] is WaveNet used as a decoder in an autoencoder setup by introducing a technique that enables the autoregressive decoder to be conditioned on temporal latent embeddings of timbre and pitch. This promotes global consistency and allows better ways to control synthesis, e.g. interpolating between latent embeddings to morph between and create new timbres. However, due to the temporally conditioned latent, synthesis can not be guided in global structure.[46]

[45] Engel, Resnick, et al., *Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders*, 2017.

[46] The NSYNTH dataset is also introduced in this work, which we use different subsets of. See section 5.5 for more information.

Symbol-to-Instrument Neural Generator (SING)[47] is the first successful attempt at applying transposed convolutions to audio waveform generation, resulting in large computational effciency gains, training and generation speed is roughly 32 and 2500 times

[47] Défossez et al., *SING*, 2018.

faster than in WaveNet autoencoder. SING consists of a LSTM[48] whose output is fed to the transposed convolution based decoder for waveform generation. The decoder is initially trained using an autoencoder setup with a convolutional encoder. Afterwards, the LSTM is trained to model the latent sequences conditioned on instrument, pitch, velocity[49] and timbre.

WaveGAN & SpecGAN[50] are transposed convolution based GANs generating audio waveforms (WaveGAN), and in the frequency domain as a spectrogram (SpecGAN), which is transformed to waveforms using Griffin-Lim inversion[51]. WaveGAN is the first successful attempt at applying GANs to audio waveform generation. Both models are minor modifications of DCGAN[52] which was breakthrough in the field of unsupervised image generation. SpecGAN did better covering the diversity of datasets, but WaveGAN achieved better mean opinion scores, possibly caused by noise induced by the lossy Griffin-Lim inversion. The synthesis speed of these models improves upon those of SING due to the fully parallelizable architecture.[53]

[48] Hochreiter and Schmidhuber, "Long Short-Term Memory", 1997.

[49] In musical instrument terminology, velocity refers to speed at which a string is hit, determing it volume and certain timbral characteristics

[50] Donahue et al., *Adversarial Audio Synthesis*, 2019.

[51] Griffin and Lim, "Signal Estimation from Modified Short-Time Fourier Transform", 1984.

[52] Radford et al., *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*, 2016.

[53] We compare WaveGAN in our baseline experiments. For more information on WaveGAN, see section 4.6.3

4 Methodology

In this chapter we formalize the goal of our experiments, propose a conceptual framework to unify the various considered variables in this work, introduce all relevant concepts for our experimental section, and arrange these within the proposed framework.

4.1 Problem formulation

We are interested in learning a distribution of continuous audio waveform representations covering dataset D consisting of N discretely sampled waveforms. Waveforms in datasets D are represented by point sets X_i, consisting of M equally-spaced sampled amplitude values from the corresponding continuous amplitude functions $y_i : \mathbb{R}^1 \to \mathbb{R}^1$:

$$\mathcal{D} = \{X_i\}_{i=1}^{N}, \quad X_i = \left\{ \left(t_j, a_j\right) : a_j = y_i\left(t_j\right) \right\}_{j=1}^{M}. \tag{4.1}$$

Where t_j are time coordinates, and a_j are the corresponding amplitudes or air pressure measurements at these time coordinates.

We represent audio waveforms by directly approximating amplitude functions y_i with continuous functions $f_i : \mathbb{R}^1 \to \mathbb{R}^1$, parameterized by MLPs ϕ_i with sets of (learnable) parameters $\theta_i \in \Theta$.

Generalizing across represented functions f_i, amounts to learning a distribution over all sets of parameters $\{\theta_i\}_{i=1}^{N}$, $p(\theta)$. We assume parameter sets θ_i live in a low-dimensional manifold. To sample parameter sets θ_i from $p(\theta)$, we define:

- A distribution over latent representations $p(\mathbf{z})$, with $\mathbf{z} \in \mathcal{Z}$.
- A function mapping latent representations $\mathbf{z}$ to intermediate latent representations $\mathbf{w} \in \mathcal{W}$, $\mathbf{w} = g(\mathbf{z})$ with $g : \mathcal{Z} \to \mathcal{W}$, parameterized by a neural network ψ with learnable parameters ξ, also known as the latent mapping network.
- A set of parameters $\boldsymbol{\alpha}$ functioning as weights in ϕ_i shared between representations.
- A function defining how intermediate latent representations $\mathbf{w}$ influence shared parameters $\boldsymbol{\alpha}$ to obtain θ_i, $\theta_i = c(\mathbf{w}_i, \boldsymbol{\alpha})$, also known as the conditioning method.

The shared parameters $\boldsymbol{\alpha}$, latent mapping network ψ and conditioning method c together parameterize the conditional distribution $p(\theta|\mathbf{z})$, allowing us to map latent representations to the parameter space $p(\theta) = E_z[p(\theta|\mathcal{Z} = \mathbf{z})]$. By sampling $\mathbf{z}_i \sim p(\mathbf{z})$, obtaining $\mathbf{w}_i$ by mapping $\mathbf{z}_i$ through $g(\mathbf{z}_i)$, conditioning shared parameters $\boldsymbol{\alpha}$ using $\mathbf{w}_i$ and conditioning method $c(\mathbf{w}_i, \boldsymbol{\alpha})$, we obtain parameter set θ_i for ϕ_i. Then, we can evaluate $\phi_i|\theta_i$ at time coordinates t_j to obtain corresponding amplitude approximations $\hat{a}_j$:

$$\mathbf{z}_i \sim p(\mathbf{z}), \quad \theta_i = c(g(\mathbf{z}_i), \boldsymbol{\alpha}), \quad \hat{a}_j = \phi_i(t_j|\theta_i).$$

We optimize $\phi_i|\theta_i$ to represent functions y_i with perceptual- and absolute fidelity.

Perceptual fidelity is measured by the metrics described in section 5.8, absolute fidelity is measured by mean squared error (MSE) between amplitudes a_j and approximations $\hat{a}_j$. All metrics are applied on pairs of point sets sampled from functions $f_i(t_j)$ and $y_i(t_j)$ at $\{t_j\}_{j=1}^{M}$.

4.2 Latent embedding inference methods

In this work we aim to explore characteristics of decoders without any restrictions regarding latent distribution$p(\mathbf{z})$. However, this still requires methods for embedding data in latent spaces. In our baseline experiments (section 6.1) we compare the effects of using autoencoders and autodecoders for inferring latent embeddings $\mathbf{z_i}$ during training on the perceptual- and absolute fidelity of reconstructions f_i.

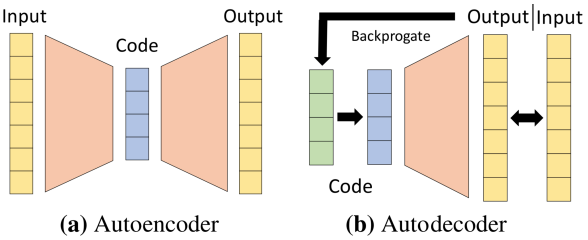
Autoencoders have been around for a long time[1] and are widely used for representation learning as their bottleneck features tend to form natural latent variable representations. Autoencoders consist of an encoder network, which projects signals y_i to a low dimensional latent embedding $\mathbf{z_i}$, and a decoder network which generates the corresponding reconstruction f_i from the inferred latent embedding.

[1] Rumelhart et al., *Parallel Distributed Processing*, 1986.

Autodecoders were introduced recently by Park et al.[2]. They have no encoder network. Latent embeddings $\mathbf{z_i}$ are instead treated as learnable parameters rather than inferred from observations at training. By storing and updating intermediate latent embeddings $\mathbf{z_i}$ with backpropagated training errors during training the decoder can function as an encoder, see fig. 4.1 for a schematic overview.

[2] Park et al., *DeepSDF*, 2019.

Benefits of autodecoders. Due to the direct optimization of latent embeddings $\mathbf{z_i}$, autodecoders generally require less training iterations to reconstruct datasets faithfully. Autodecoders are more parameter efficient by not having an encoder, resulting in less operations per iteration and lower memory requirements during training.



(a) Autoencoder **(b)** Autodecoder

Figure 4.1: Schematic overview of autoencoder and autodecoder architectures. Figure adapted from Park et al., DeepSDF, 2019

Finally, autodecoders can alleviate incompatibility issues between encoders and decoders. This is especially useful when dealing with implicit neural representations (INRs), as these are less trivial to be mirrored to use as an encoder[3].

Drawbacks of autodecoders. A major drawback of autodecoders is that inferring latent embeddings of unseen samples after training requires a multi-iteration optimization process. An encoder can project any sample to a latent embedding in a single forward pass. However, this is not relevant for the experiments in this work as we do not evaluate architectures on reconstructing unseen samples.

Another drawback of autodecoders is that they do not easily function as proper generative models. Park et al. have proposed to enforce a distribution on latent embeddings by using an additional variational objective, similar as in VAEs[4]. However, since autodecoders optimize samples and not parameters of a latent distribution, this is unlikely to result in latent embeddings distributed according to the enforced distibution[5]. Some works have proposed methods which try to fix this, e.g. by enforcing a distribution on the gradients optimizing latent embeddings[6]. However, since we do not enforce any structure on $p(\mathbf{z})$ this is not relevant for our experiments.

[3] E.g. a transposed convolution based decoder mirrors a regular convolution based encoder. INRs are less trivial to be mirrored to as an encoder as they are agnostic to the number of observations and do not require observations on a regular grid. However, good candidates for INR mirroring encoders exist, such as PointNet.

[4] Park et al., *DeepSDF*, 2019.

[5] Dupont et al., *Generative Models as Distributions of Functions*, 2021.

[6] Bond-Taylor and Willcocks, *Gradient Origin Networks*, 2020.

4.3 Neural conditioning methods

As described in section 4.1, we aim to generalize across reconstructed functions f_i by learning a distribution over parameter sets $p(\theta)$ used in MLPs ϕ_i parameterizing functions f_i. Base parameters $\boldsymbol{\alpha}$, latent mapping network ψ and conditioning method c together parameterize conditional distribution $p(\theta|\mathbf{z})$ used to sample parameter sets θ_i.

The amount of expressivity, (implicit) regularization and computational complexity differs strongly between conditioning methods c. Which methods work best dependends on the parameterization of ϕ_i and characteristics of dataset D. In current literature covering INRs in generative settings, the following conditioning methods have seen most exposure:

1. Generating weights of ϕ_i directly with another neural network, a hypernetwork[7].

[7] Dupont et al., *Generative Models as Distributions of Functions*, 2021; Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020; Skorokhodov et al., *Adversarial Generation of Continuous Images*, 2020.

2. Concatenating latent embeddings $\mathbf{z}_i$ to layer inputs in ϕ_i[8].
3. Applying FiLM[9] to layer activations in ϕ_i.[10]

Hypernetworks[11] are considered the most flexible neural conditioning method. A latent mapping network ψ takes a latent embedding $\mathbf{z}_i$ as input and generates all parameters of a *hyponetwork*, in our case ϕ_i[12]. See figure 4.2 for a schematic overview. Framing the typical implementation of the method in the conceptual framework described in section 4.1 we note the following specifics:

1. Intermediate latent representations $\mathbf{w}$ have the same dimensionality as θ_i.
2. Intermediate latent representations $\mathbf{w}$ directly function as parameters in ϕ_i. Resulting in the following conditioning function c: $\theta_i = \mathbf{w}_i$.
3. There are no learnable parameters $\boldsymbol{\alpha}$, shared between representations.

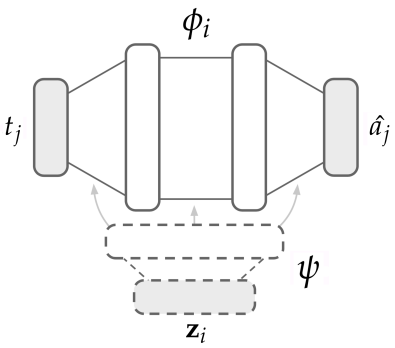


Figure 4.2: Schematic overview a hypernetwork ψ. ψ takes $\mathbf{z}_i$ as input to generate the weights of a hyponetwork ϕ_i. Figure adapted from Dupont et al., Generative Models as Distributions of Functions, 2021.

Conditioning via concatenation can be considered the most ubiquitous neural conditiong method, as it also describes architectures that simply take latent codes as input (such as WaveGAN). Latent embeddings $\mathbf{z}_i$ are concatenated with the input vector $\mathbf{x}_k$ of parameter matrix $\mathbf{W}_k$ of layer k in ϕ_i. Layer activation $\mathbf{y}_k$ is then calculated as follows:

$$\mathbf{y}_k = act_k(\mathbf{W}_k(\mathbf{x}_k||\mathbf{z}_i) + \mathbf{b}_k). \tag{4.2}$$

Where act_k is the activation function, $\mathbf{b}_k$ the bias and $||$ is the concatenation operator. See figure 4.3 for a schematic overview.

Concatenation as a special case of a hypernetwork. If we split weight matrix $\mathbf{W}_k$ into two weight matrices, $\mathbf{W}_{hypo}$ and $\mathbf{W}_{hyper}$, such that $\mathbf{W}_{hyper}$ contains only those weights in $\mathbf{W}_k$ that are applied to $\mathbf{z}_i$, then layer activation $\mathbf{y}_k$ is described as:

$$\mathbf{y}_k = act_k(\mathbf{W}_{hypo}(\mathbf{x}_k) + \underbrace{\mathbf{W}_{hyper}(\mathbf{z}_i) + \mathbf{b}_k}_{\text{bias}}). \tag{4.3}$$

[8] Chen and Hao Zhang, *Learning Implicit Fields for Generative Shape Modeling*, 2019; Park et al., *DeepSDF*, 2019; Schwarz et al., "GRAF: Generative Radiance Fields for 3D-Aware Image Synthesis", n.d.

[9] Perez et al., *FiLM*, 2017.

[10] Chan et al., *Pi-GAN*, 2020; Mescheder et al., *Occupancy Networks*, 2019.

[11] Ha et al., *HyperNetworks*, 2016.

[12] A hypernetwork could also be implemented to only output a subset of parameters of a hyponetwork, as described in equation 4.3, but to our knowledge has never been shown to improve results in the area of INRs.

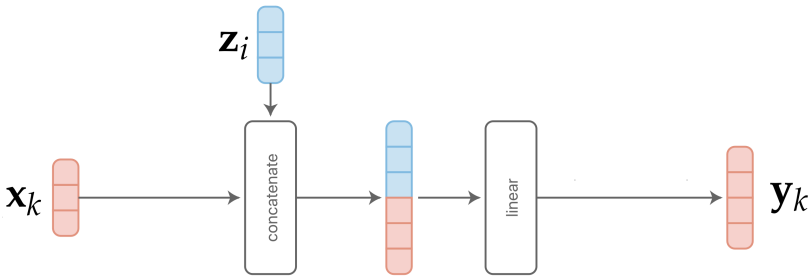


Figure 4.3: Schematic overview of concatenation based conditioned layer. Figure adapted from Perez et al., FiLM, 2017.

Thus, conditioning via concatenation can be viewed as a special case of a hypernetwork, where the latent mapping network ψ is a single affine layer that only predicts the biases of a single layer of a hyponetwork ϕ_i, see figure 4.4. Framing the method in the conceptual framework described in section 4.1 we note the following specifics:

1. Intermediate latent representations $\mathbf{w}$ have the same dimensiononality as the total amount of hidden units in conditioned layers in θ_i.

2. Intermediate latent representations $\mathbf{w}_k$ shift layer activations in layer k of ϕ_i on top of bias controlled by shared parameters $\boldsymbol{\alpha}_k^b$, as described in equation 4.3. Resulting in the conditioning method c shown in equation 4.4.

3. All other parameters θ in ϕ_i are parameterized by $\boldsymbol{\alpha}$, shared between representations.

Conditioning via concatenation applies the following conditioning method:

$$c : \theta_k^b = c(\mathbf{w}_k, \boldsymbol{\alpha}_k^b) = \mathbf{w}_k + \boldsymbol{\alpha}_k^b. \tag{4.4}$$

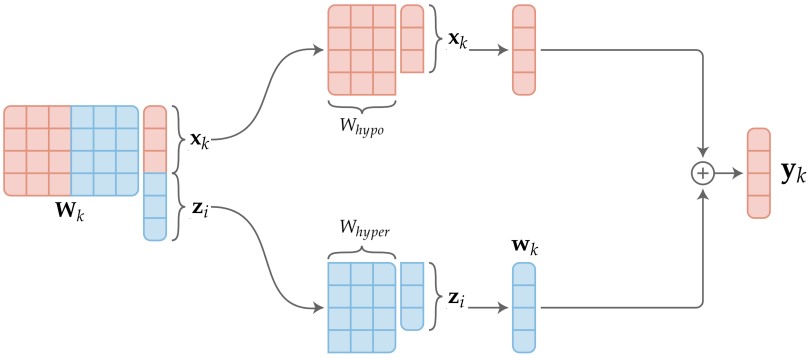


Figure 4.4: Schematic overview of concatenation based conditioning viewed as a special case of a hypernetwork, one that only controls bias. Figure adapted from Perez et al., FiLM, 2017.

Feature-wise linear modulation (FiLM)[13] proposes to apply element-wise[14] scaling and shifting of intermediate layer activations

[13] Perez et al., *FiLM*, 2017.

[14] In the case of convolutional networks, FiLM is applied feature map wise.

based on latent representations $\mathbf{z}_i$, as shown in equation 4.5 and figure 4.5. Framing the method in the conceptual framework described in section 4.1 we note the following specifics:

1. Intermediate latent representations $\mathbf{w}$ have the dimensionality of twice the total amount of hidden units of conditioned layers in θ_i, one half for scaling ($\mathbf{w}^\gamma$) and one half for shifting ($\mathbf{w}^\beta$) layer activations.
2. The conditioning function c is applied as shown in equation 4.5. Note that parameters controlling bias θ_i^b are replaced by $\mathbf{w}_i^\beta$.
3. All other parameters θ in ϕ_i are parameterized by $\boldsymbol{\alpha}$, shared between representations.

$$y_k(\mathbf{x}_k) = act_k(\mathbf{w}_k^\gamma \cdot \mathbf{W_k x}_k + \mathbf{w}_k^\beta). \tag{4.5}$$

Where x_k is the layer's input, z_i is a conditioning input, and γ and β are functions outputting respectively scaling and shifting vectors dependent on $\mathbf{z}_i$.

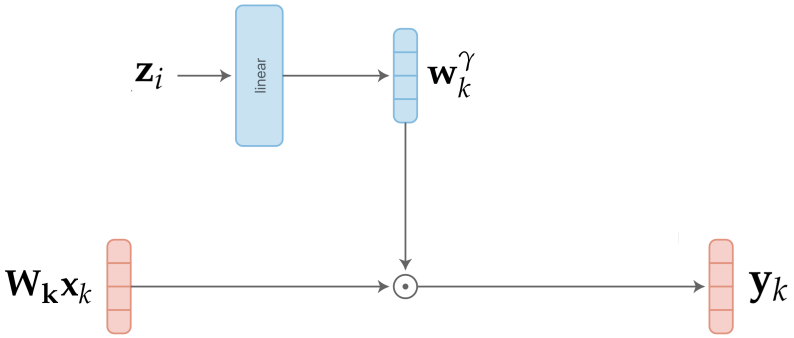


Figure 4.5: Schematic overview of conditional scaling of features. Note this is not a representation of equation 4.5, but a simplified scenario. Figure adapted from Perez et al., FiLM, 2017.

FiLM is a unification of methods that modulate intermediate layer activations, as in normalization layers introduced originally in batch norm[15]. Other implementations include Dynamic Layer Norm for speech recognition[16], Conditional Instance Norm[17], Adaptive Instance Norm[18], and Conditional Batch Norm as used in Occupancy networks[19]

Hypernetworks based approaches have no $\boldsymbol{\alpha}$. In concatenation- and FilM based approaches we assume a significant amount of information of the conditional distribution $p(\theta|\mathbf{z})$ is encoded in the shared parameters $\boldsymbol{\alpha}$. In hypernetwork based approaches there are no shared weights $\boldsymbol{\alpha}$ populating ϕ_i. Consequently, latent mapping network ψ fully parameterizes conditional distribution $p(\theta|\mathbf{z})$. This could be a preferable division of roles in more diverse function distributions. Besides the fact that having no shared parameters results in having more diverse possibilites of functional representations ϕ_i, this could also be argued by the observation that INRs have shown impressive results for representing continuous functions, less so for carrying information of a conditional distribution $p(\theta|\mathbf{z})$.

[15] Ioffe and Szegedy, *Batch Normalization*, 2015.
[16] Kim et al., *Dynamic Layer Normalization for Adaptive Neural Acoustic Modeling in Speech Recognition*, 2017.
[17] Ghiasi et al., *Exploring the Structure of a Real-Time, Arbitrary Neural Artistic Stylization Network*, 2017.
[18] Huang and Belongie, *Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization*, 2017.
[19] Mescheder et al., *Occupancy Networks*, 2019.

Hypernetwork conditioned SIRENs. With preliminary results being strongly in favor of SIRENs, we experimented with small SIRENs generated by hypernetworks. But due to initialization difficulties we were not able to include hypernetwork based parameterizations of $p(\theta|\mathbf{z})$ in our reported experiments. The small size of generated images in the hypernetwork conditioned SIREN experiments of Sitzmann et al.[20] and the general absence of SIRENs in hypernetwork conditioned INR approaches seems to indicate this is a challenging approach.

[20] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

4.4 Nonlinearities in Implicit Neural Representations

Viewing MLPs as architectures learning set of basis functions, ReLU P.E. MLPs learn functions in a sparse fourier basis making them theoretically well suited for representing audio. SIRENs learn functions in a nested sinuisoidal basis. The behaviour of such a basis is less well understood, but it is potentially suited for audio because of its increased expressity compared to a sparse fourier basis.

Emperical results. Sitzmann et al.[21] show the reconstruction performance of SIRENs, ReLU MLPs and ReLU P.E. MLPs on different signals. Comparing performance between SIRENs and ReLU P.E. MLPs, in image reconstruction results shown in figure 4.6, ReLU P.E. MLPs show comparable results to SIRENs (roughly a 20% decrease in peak signal-to-noise ratio, PSNR), while the audio reconstruction results shown in figure 4.7 are signifcantly more in favor of SIRENs (roughly a 90% decrease in MSE).

[21] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

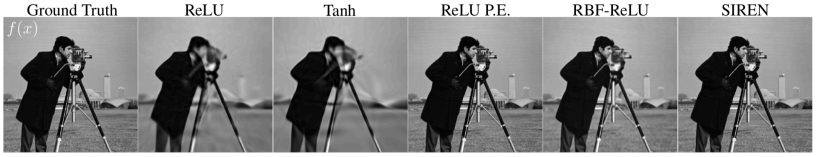


Figure 4.6: Result of reconstruction 5 layer, 256 hidden unit INRs with different activation functions to an image. Figure is taken from Sitzmann et al., Implicit Neural Representations with Periodic Activation Functions, 2020.

4.5 Activation scaling in sinusoidal INRs

Sitzmann et al.[22] introduce an activation scaling factor in every layer of SIRENs, ω_0:

[22] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

$$y_k(\mathbf{x}_k) = \sin\left(\omega_{0,k} \cdot \mathbf{W}_k \mathbf{x}_k + \mathbf{b}\right), \tag{4.6}$$

to match the frequency spectrum of signals to be represented. If we consider a 2 layer SIREN, with no activations in the final layer, represented signals live in a sparse Fourier basis, just like ReLU P.E. MLPs. ω_0 determines the range of frequencies that can be represented. When making the network deeper, we enter the domain of

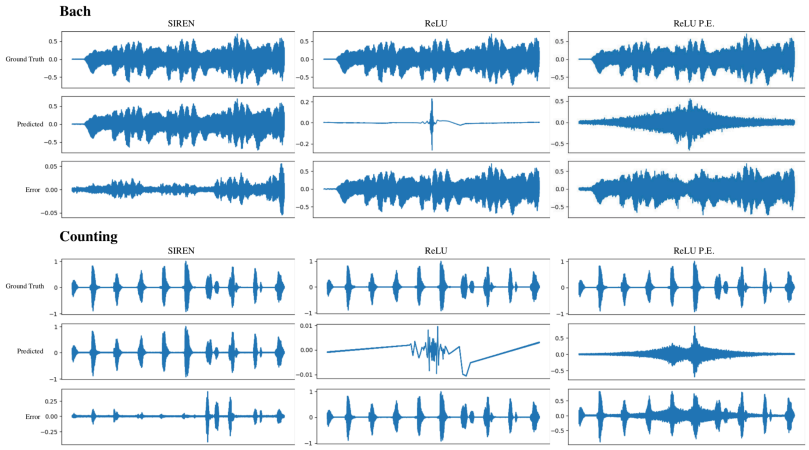


Figure 4.7: Result of reconstruction 5 layer, 256 hidden unit INRs with different activation functions to different audio samples. Figure is taken from Sitzmann et al., Implicit Neural Representations with Periodic Activation Functions, 2020.

nested sines, which behave similar to *damped* sines without any activation scaling. However, with higher ω_0 values nested sines quickly introduce more and higher frequency components.

Consequently, this hyperparameter has significant influence on the inductive bias, smoothness and expressivity of SIRENs and optimal values strongly depend on the data you want to represent and the parameterization of ϕ_i. The image experiment shown in figure 4.6 uses ω_0 values of 30, while the audio experiment shown in figure 4.7 uses ω_0 values of 3000 in the first layer, and 30 in the later ones.

ω_0 optimization. Heuristics for optimal values of ω_0 when representing sets of signals regarding set characteristics such as size and diversity, and the parameterization of the conditional distribution $p(\theta|\mathbf{z})$ is unexplored. Results are however very senstive to this hyperparameter. To account for this, we optimize ω_0 magnitudes in all relevant experiments. We minimize the search space by considering only 2 groups of ω_0 values: those in the first layer, and those in later layers.

4.6 Baseline Decoders

In this section we view different previously proposed decoders compared in the baseline comparison of our experiments through the lens of the conceptual framework described in section 4.1. We compare the differences in the parameterization of ϕ_i and the conditional distribution $p(\theta|\mathbf{z})$, and argue why we decided to compare these architectures. For a short introduction and their position among other architectures, see chapter 3

4.6.1 IM-NET

IM-NET[23] parameterizes ϕ_i as a 6 layer MLP with ReLU activation

[23] Chen and Hao Zhang, *Learning Implicit Fields for Generative Shape Modeling*, 2019.

functions and a quadratically shrinking amount of hidden units (starting at 2048). The conditional distribution $p(\theta|\mathbf{z})$ is parameterized as follows:

- Input coordinates are concatenated to $\mathbf{z}$. The latent mapping network ψ takes this concatenated vector as input.
- ψ consists of 5 parralel affine layers with outgoing connections to each layer input of ϕ_i, until the single last layer.
- Outputs of ψ, $\mathbf{w}$, shift layer activations of conditioned layers in ϕ_i, without replacing the bias parameters in ϕ_i. Resulting in the following conditioning method c in layer k:
  $y_k(\mathbf{x})_k = act_k(\mathbf{W}_k\mathbf{x}_k + \mathbf{b}_k + \mathbf{w}_k) \rightarrow \theta_k^b = c(\mathbf{w}_k, \boldsymbol{\alpha}_k^b) = \mathbf{w}_k + \boldsymbol{\alpha}_k^b$.
- Shared parameters $\boldsymbol{\alpha}$ populate all weights native to ϕ_i.

See figure 4.8 for a schematic overview of the network structure as provided by the authors.

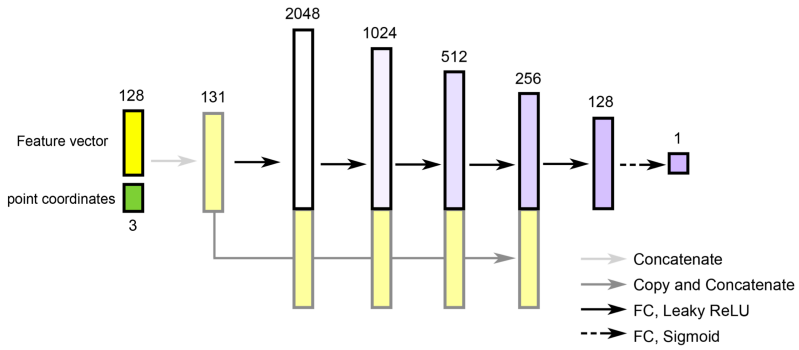


Figure 4.8: Network structure of IM-NET. Figure is taken from Chen and Hao Zhang, Learning Implicit Fields for Generative Shape Modeling, 2019

We considered including DeepSDF or Occupancy networks instead for representing MLP based INRs with ReLU activations. The reported experimental results of IM-NET, Occupancy networks and DeepSDF indicate that IM-NET and Occupancy networks outperform DeepSDF. We decided to drop DeepSDF for its inferior performance, and to continue with IM-NET to create a more diverse comparison, as Occupancy networks' conditioning method is similar to that of π-GAN.

4.6.2 π-GAN

π-GAN[24] is designed to produce implicit 3D radiance fields conditioned on $\mathbf{z}_i$. Ignoring the final parallel layer which integrates ray directions, π-GAN parameterizes ϕ_i as an 8 layer MLP with sine nonlinearities and a constant amount of hidden units (256). The conditional distribution $p(\theta|\mathbf{z})$ is parameterized as follows:

- Latent mapping network ψ consists of a 3 layer MLP with ReLU activations.

[24] Chan et al., *Pi-GAN*, 2020.

- The conditioning mechansism c falls under the category of FiLM. $\mathbf{w}^{\gamma}$ and $\mathbf{w}^{\beta}$ vectors are shared between layers, drastically decreasing the total size of $\mathbf{w}$.
- $\mathbf{w}^{\gamma}$ does not scale layer activations directly, but is summed element-wise with the predefined activation scaling factor ω_0 we know from SIRENs, resulting in the following conditioning method c in layer k:[25]

$$\theta_k^b, \theta_k^{\omega_0} = c(\mathbf{w}, (\boldsymbol{\alpha}_k^b, \omega_{0,k})) = \mathbf{w}_k^{\beta} + \boldsymbol{\alpha}_k^b, \mathbf{w}_k^{\gamma} + \omega_{0,k}.$$

- Shared parameters $\boldsymbol{\alpha}$ populate all weights native to ϕ_i.

See figure 4.9 for a schematic overview of the network structure as provided by the authors.

[25] Technical descriptions in the paper of π-GAN seem to indicate that $\mathbf{w}_k^{\gamma}$ directly replaces $\omega_{0,k}$. This resulted in instable behaviour in preliminary experiments. E. Chan shared parts of their implementation showing the discussed conditioning method in which $\mathbf{w}_k^{\gamma}$ and $\omega_{0,k}$ are summed.

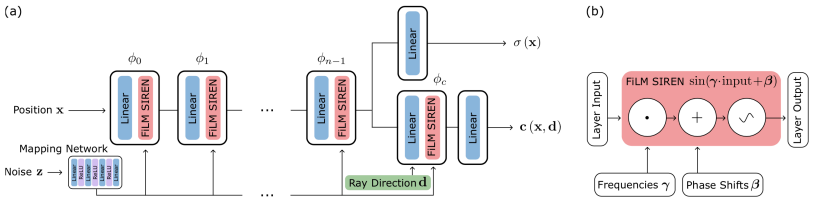


Figure 4.9:
(**a**): Network structure of π-GAN.
(**b**): Schematic overview of a FiLMed-SIREN layer. Figure is taken from Chan et al., Pi-GAN, 2020.

The authors report the effects of replacing the SIREN based parameterization of ϕ_i with a ReLU P.E. MLP, and FiLM conditioning with conditioning via concatenation. Changing either of these significantly reduces FID scores, showing strong potential of combining both methods. See figure 4.1.

Conditioning	Architecture	
	ReLU P.E.	Sine
Concatenation	32.0	21.6
Mapping Network	26.8	**5.15**

Figure 4.1: FID scores on CelebA @ 64 × 64, when comparing π-GAN decoders with different activation functions and conditioning methods.

4.6.3 WaveGAN

The WaveGAN decoder is based off of DCGAN[26]'s decoder, which popularized usage of GANs for image synthesis. This decoder uses transposed convolution to iteratively upsample low-resolution feature maps into a high-resolution image. In the WaveGAN decoder this is modified to work with audio by replacing its two-dimensional 5×5 filters with one-dimensional filters of length 25, and changing the stride factor for all convolutions from 2×2 to 4. These changes result in WaveGAN having the same number of parameters, numerical operations, and output dimensionality as DCGAN. Because DCGAN outputs 64×64 pixel images — equivalent to just 4096 audio samples — one additional layer is added to the model resulting in 16384 samples, slightly more than one second of audio at 16 kHz.

[26] Radford et al., *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*, 2016.

Although it is not very intuitive, WaveGAN can be viewed as a discrete parameterization of ϕ_i & $p(\theta|\mathbf{z})$ largely consisting of shared parameters α. Latent mapping network ψ consists of a single fully connected layer with ReLU nonlinearities, its output, intermediate latent representations $\mathbf{w}$ are reshaped in 128 one-dimensional feature maps of length 16 serving as input maps for the first convolutional layer.

4.7 Objective function

In our experiments we use MSE with addition of MSE between the derivative of target signals approximated with forward finite difference, and the derivative of reconstructions evaluated in the INR (For WaveGAN this is approximated by forward finite difference derivative of the reconstruction):

$$\mathcal{L} = \frac{1}{N}\frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M} \left\|\Phi(t_j, \mathbf{z_i}) - \mathrm{y}_i(t_j)\right\|^2 + \left\|\frac{\delta}{\delta t_j}[\Phi(t_j, \mathbf{z_i})] - \Delta t_j[\mathrm{y}_i(t_j)]\right\|^2 . \tag{4.7}$$

Where N is the batch size, M is the amount of sampled time coordinates per waveform and Δ is the forward finite difference. The addition of a MSE of derivatives of signals is justified by consistent training improvements observed in preliminary experiments.

When optimizing representations of audio waveforms for perceptual fidelity, the problem is constrained unnecessarily by using objective functions that are calculated per amplitude sample such as MSE.[27] However, it turns out π-GAN has strict requirements regarding the local consistency of objective functions, showing improved results for MSE compared to perceptual objectives. See section 6.1.2.

[27] As noted in section 2.2

5 Experimental Setup

In this chapter we specify all details of the experimental setup relevant for interpreting experimental results. Some details necesarry for reproducing these experiments, but not crucial for interpreting results are left out. Those can be found in section A.3.

5.1 Overview

In our experiments we explore the following options regarding the parameterizations of ϕ_i and $p(\theta|\mathbf{z})$:

1. Architecture family of ϕ_i (see section 4.6)
   - Convolutional neural networks
   - Implicit neural representations
2. latent embedding inference methods (see section 4.2)
   - Autoencoder
   - Autodecoder
3. Conditioning methods c (see section 4.3).
   - Concatenation
   - Feature wise linear modulation (FiLM)
4. Activation functions (see section 4.4)
   - ReLU
   - Sine
   - Mixed
5. Latent mapping network ψ depth (see section 5.3)
6. ϕ_i network shape (see section 5.3)
7. Weight regularization (see section 5.4.1)
8. Progressive activation scaling (see section 5.4.2)

The compared parameterizations of ϕ_i and $p(\theta|\mathbf{z})$ in the baseline experiments (section 6.1) differ among parameterizations 1 to 6, as described in section 4.6. In the ablation experiments (section 6.2) we

compare the effect altering 3, 4, 5 and 6 in the best performing INR based parameterization of ϕ_i, π-GAN. In section A.1[1], we report a sequence of SIREN experiments aimed at finding potential solutions to observed shortcomings of resulting representations f_i in previous sections. Finally, we validate potential solutions found in section A.1 in our regular experimental setup in section 6.3.

[1] Although the results of this section resulted in interesting insights, we decided to move this section to the appendix as it is not essential to our research questions.

5.1.1 General remarks

In all experiments in section 6.1, 6.2 and 6.3 we train waveform representations as described in 4.1. We use the objective function as in equation 4.7, to optimize weights of tested neural architectures using gradient descent, see section 5.6 for details regarding the specific optimizers we tested. We test three datasets, described in section 5.5. We did an extensive hyperparameter search in our initial experiments, in later experiments we only optimized a subset of hyperparameters, described in section 5.7.

All reported results are based on three runs with different seeds after 5000 epochs in section 6.1 and 6.2, 10.000 epochs in section 6.3. We compare learned representations f_i to the original waveforms X_i and evaluate the perceptual- and absolute fidelity. Used metrics and selection procedure are described in section 5.8.

Input coordinates for implicit architectures are sampled in the range $[-1, 1]$. Preliminary experiments indicated that subsampling evenly spaced coordinates[2] up to a fraction of $\frac{1}{8}$ did not have significant negative impact on training objective errors observed after 5000 epochs for our datasets. Thus, to even out memory usage between implicit and convolutional network architectures and allow for equal batch sizes, we decided to sample 2000 out of 16000 coordinates for implicit architectures in every iteration during training.

[2] Subsampling evenly spaced coordinates effectively reduces the sampling rate of the wavefile as perceived by the network in every iteration. Uniform random subsampled coordinates also worked okay, but slightly worse than evenly spaced subsampling.

5.2 Baseline experiments

5.2.1 Parameterizations of ϕ_i & $p(\theta|\mathbf{z})$

The following parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ are compared, exact parameter counts are in brackets:

1. WaveGAN[3] [799k]
2. IM-NET[4] [1.1M]
3. π-GAN[5] [790k]

[3] Donahue et al., *Adversarial Audio Synthesis*, 2019.
[4] Chen and Hao Zhang, *Learning Implicit Fields for Generative Shape Modeling*, 2019.
[5] Chan et al., *Pi-GAN*, 2020.

For all parameterizations of ϕ_i & $p(\theta|\mathbf{z})$, the network architecture is changed such that parameter counts are in the order of 10^6. WaveGAN's parameter count is reduced by reducing the number of channels throughout the network by a factor of eight. IM-NET's parameter count is reduced by removing the first, largest hidden layer. π-GAN's parameter count did not have to be reduced.

5.2.2 Latent embedding $\mathbf{z}_i$ inference methods

The decoders are tested with the following latent embedding inference methods:

1. Autoencoder[6]
2. Autodecoder[7]

The autoencoder setups in our experiments use a convolutional encoder designed to match WaveGAN, see section A.3.3 for architectural details. We used this type of encoder as it is compatible with WaveGAN and preliminary experiments indicated a better performance than recurrent encoders (wav2vec[8]), and fully-connected encoders with DFT's as input for INRs.

[6] Rumelhart et al., *Parallel Distributed Processing*, 1986.

[7] Park et al., *DeepSDF*, 2019.

[8] Schneider et al., *Wav2vec*, 2019.

5.3 Ablation experiments

5.3.1 Parameterizations of ϕ_i & $p(\theta|\mathbf{z})$

In the ablation experiments we continue with the best performing INR parameterization and latent embedding inference method found in the baseline experiments. Then, we compare the effects of changing the following specific parameterizations:

1. Activation functions in ϕ_i
   (a) All sine (π-GAN)
   (b) Sine first, others ReLU
   (c) Sine last, others ReLU
   (d) All ReLU
2. Conditioning mechanism c
   (a) FiLM (π-GAN)
   (b) Concat, middle layer
   (c) Concat, all layers
3. Latent mapping network ψ depth
   (a) Three layer latent mapping network (π-GAN)
   (b) Minimal (single layer) latent mapping network
   (c) Extended (five layer) latent mapping network
4. ϕ_i Network shape
   (a) π-GAN (256 hidden units, 8 layers)
   (b) Deep (200 hidden units, 12 layers)
   (c) Wide (380 hidden units, 4 layers)
   (d) Shrinking (840, 420, 210, 105 hidden units)

5.4 Proposed extension experiments

The methods evaluated in this section are developed on SIRENs evaluated in a simplified environment. The validation process is reported in section A.1.

Since these methods significantly increase the amount of iterations needed to converge, we double the total amount of epochs compared to section 6.1 and 6.2, training for 10.000 epochs instead of 5000.

5.4.1 Weight regularization

We use the same weight regularization as tested in section A.1, resulting in the following loss function:

$$\begin{aligned}\mathcal{L} = &\frac{1}{N}\frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M}\left\|\Phi(t_j, \mathbf{z_i})) - \mathrm{y}_i(t_j)\right\|^2 \\ &+ \left\|\frac{\delta}{\delta t_j}[\Phi(t_j, \mathbf{z_i})] - \Delta_{t_j}[\mathrm{y}_i(t_j)]\right\|^2 + \frac{\lambda}{2}\|\mathbf{W}_{\phi_i}\|^2.\end{aligned} \tag{5.1}$$

Note that weight regularization is *only* applied to the weights of ϕ_i, not to those of ψ. In this section we do no new hyperparameter search, but continue with the architecture and previously found optimal ω_0's for π-GAN vanilla to validate if this method can effective applied after a parameter search in the short training regime. For the linearly decreasing weight regularization, we decrease λ to zero between epoch 0 to 5000. In the final 5000 epochs we train without weight regularization.

5.4.2 Progressive activation scaling

Progressive activation scaling is inspired on methods proposed by Hertz et al.[9]. It is implemented as follows:

[9] Hertz et al., *SAPE*, 2021.

256 different entries for the first ω_0's are picked from a uniform distribution in the reported range, sorted low to high, and then split in 8 groups. At the beginning of training, all ω_0's are masked. The first half of total iterations is split in 8 periods (the amount of ω_0 groups). In the first half of each period a new group of ω_0's is linearly unmasked. In the second half of each period nothing changes. In the second half of total iterations, all ω_0's are unmasked and stay that way. [10].

[10] This method was selected by testing on a π-GAN setup with a dataset size of 128 samples. We also tried selecting ω_0's as evenly spaced numbers over a specified interval and treating every ω_0 value as its own group, fading them in one by one. These alterations performed slightly worse.

5.5 Datasets

For all experiments in section 6.1, 6.2 and 6.3 we consider the following three datasets:

1. 1 second, 1024 item, keyboard waveforms in MIDI notes [60, 64] subset of NSYNTH[11]. Balanced for note counts. Recorded at a sampling rate of 16kHz.
2. 1 second, 1024 item, keyboard, mallet and guitar waveforms in MIDI notes [24, 84] subset of NSYNTH. Balanced for note- and instrument counts. Recorded at a sampling rate of 16kHz.
3. 1024 item, keywords 0 to 9 balanced subset of Speech Commands[12]. Recorded at a sampling rate of 16kHz.

[11] Engel, Resnick, et al., *Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders*, 2017.

[12] Warden, *Speech Commands*, 2018.

Preliminary tests of baseline parameterizations of ϕ_i and $p(\theta|\mathbf{z})$ in the specific generative framework described in section 4.1 showed that ϕ_i and $p(\theta|\mathbf{z})$ with a reasonable amount of parameters were uncapable of reconstructing large datasets[13]. Thus, we set the dataset size for the main experiments to be challenging, but not infeasible based on preliminary experiments.

[13] This depends on many factors including the generative framework, how its parameterized, latent embedding size and dataset uniformity

NSYNTH datasets composition

Instrument samples in NSYNTH[14] are tagged for the following qualities: Bright, Dark, Distortion, Fast Decay, Long Release, Multiphonic, Non-Linear, Percussive, Reverb, Tempo-Synced. We filtered all samples that qualified for any of these qualities to create more uniform and less complex datasets.

Then, we selected a dataset subset with the smallest range of subsequent notes within one instrument family cumulating to 1024 items. This turned out to be the keyboard instrument family, MIDI note 60 (C4, 261.63Hz) to 64 (E4, 329.63Hz). The dataset was sampled balancing for note counts. We refer to this dataset as NSYNTH keyboard.

We selected the other split of NSYNTH to contrast with NSYNTH keyboard in pitch- and timbral diversity. Furthermore, we selected MIDI notes 24 (C1, 32.70Hz) to 84 (C6, 1046.50Hz)[15], since the timbres of more extreme notes can sound unnatural to an average listener, which could complicate reconstruction quality judgements (as argued by the authors of GANSynth[16]). Then, we selected three instruments families to introduce more timbral diversity: keyboard, mallet and guitar. The dataset was sampled balancing for note- and instrument family counts. We refer to this dataset as NSYNTH Diverse.

[14]

1. Bright: A large amount of high frequency con- tent and strong upper harmonics.
2. Dark: A distinct lack of high frequency content, giving a muted and bassy sound. Also sometimes described as 'Warm'.
3. Distortion: Waveshaping that produces a distinctive crunchy sound and presence of many harmonics. Sometimes paired with non-harmonic noise.
4. Fast Decay: Amplitude envelope of all harmonics decays substantially before the 'note-off' point at 3 seconds.
5. Long Release: Amplitude envelope decays slowly after the 'note-off' point, sometimes still present at the end of the sample at 4 seconds.
6. Multiphonic: Presence of overtone frequencies related to more than one fundamental frequency.
7. Non-Linear Envelope: Modulation of the sound with a distinct envelope behavior different than the monotonic decrease of the note. Can also include filter envelopes as well as dynamic envelopes.
8. Percussive: A loud non-harmonic sound at note onset.
9. Reverb: Room acoustics that were not able to be removed from the original sample.
10. Tempo-Synced: Rhythmic modulation of the sound to a fixed tempo.

[15] Reported pitch related frequencies are assuming standard A440 tuning. When closely inspecting the DFT's in figure 5.2 there appear to be some samples which are tuned differently.

[16] Engel, Agrawal, et al., *GANSynth*, 2019.

Speech Commands dataset composition

For the Speech Commands dataset we selected keywords zero trough nine, as is done by Donahue et al.[17]. These authors argue that these ten words are interesting for generative purposes because they encompass many phonemes and can be viewed as an audio counterpart of the MNIST dataset of written digits. We refer to this dataset as Speech Commands.

[17] Donahue et al., *Adversarial Audio Synthesis*, 2019.

Dataset analysis

The NSYNTH datasets are considered less complex than the Speech Commands dataset since instrument tones are significantly more spectrally consistent. For keyboard and mallet samples, waveforms can often be reconstructed by mixing a base frequency (which determines the perceived pitch) and its overtones, multiples of the base frequency (the relative magnitude of which is one of the key identifying features of timbre). For string and brass instruments waveforms are more complicated, often containing noise components. Speech waveforms are spectrally even more complex, with energy spread out over different unrelated frequencies.

Figure 5.1 shows the time-domain representations, absolute amplitude magnitudes of every sample for every dataset. Discrete Fourier transform (DFT) magnitudes of every sample for every dataset are shown in figure 5.2. Rows are sorted on frequency with the highest magnitude, low to high. Note that DFT's are calculated over the complete waveforms. In these figures we observe the following:

- Speech Commands' time-domain representations show heterogeneity in onset times.
- Speech Commands DFT's are most spread out and have lower peak frequency magnitudes. This most strongly reflects the fact that Speech Commands' waveforms contain silence as observed in their time-domain representations, but is also caused by their relative spectral incoherence compared to musical, pure tones.
- NSYNTH datasets do not contain significant amounts of silence.
- NSYNTH Keyboard shows most uniform DFT's, showing only peaks at base frequencies and overtones.
- NSYNTH Diverse shows more *speckled* DFT's, with significant magnitudes in frequency bins at different fractions of base frequencies, lower and higher. Energy in lower frequencies can also be observed in time-domain representations showing patterns resembling dashed lines.
- NSYNTH Diverse shows a higher diversity in base frequencies.
- Although NSYNTH Diverse contains many higher base frequencies, NSYNTH Keyboard samples display larger peaks in frequency bins above 1000Hz, due to high magnitude overtones.

5.6 Optimizer

Modern neural networks are typically trained with first-order gradient methods, which can be broadly categorized into two branches: the accelerated stochastic gradient descent family such as SGD with

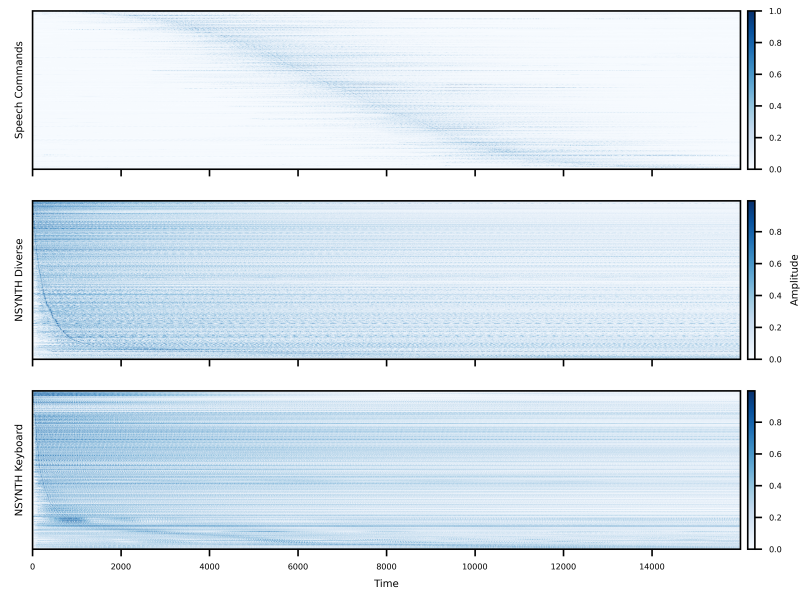


Figure 5.1: Time domain representations for all discretely sampled waveforms $\{X_i\}_{i=1}^{N}$ in all considered datasets D.
Rows represent $\{y_i\}_{i=1}^{N}$, columns represent sampled time coordinates $\{t_j\}_{j=1}^{M}$.
These plots display absolute amplitude values: $|a_j| = |y_i(t_j)|$.
Rows are sorted on their coordinate t_j with maximum absolute amplitude in X_i: $\arg\max |y_i(t_j)|$, low to high.
$\{t_j\}_{j=1}^{M}$

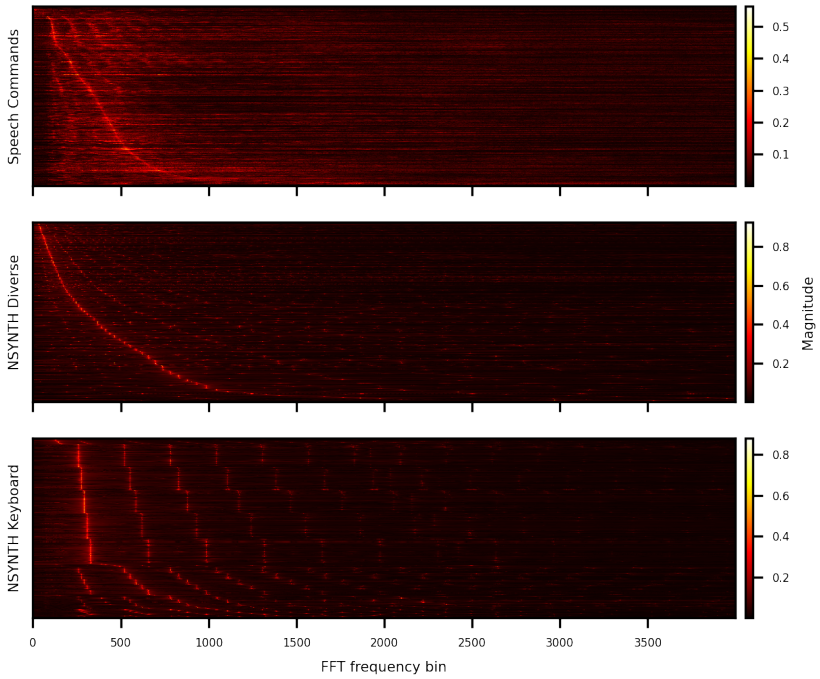


Figure 5.2: Frequency domain representations for all discretely sampled waveforms $\{X_i\}_{i=1}^{N}$ in all considered datasets.
Obtained by taking the absolute magnitudes up to a frequency of 4000Hz of the discrete Fourier transform (DFT) of $\{X_i\}_{i=1}^{N}$.
Rows are sorted on their frequency magnitude with the highest value, low to high.

momentum and the adaptive learning rate methods, such as Adam. SGD methods use a global learning rate for all parameters, while adaptive methods compute an individual learning rate for each parameter. Compared to the SGD family, adaptive methods typically converge fast in the early training phases, but have poor generalization performance. In our baseline decoder comparison we compare Adam[18] and Adabelief[19]. AdaBelief consists of the same algorithm as Adam, but also considers curvature information by scaling update directions by the change in gradient. Preliminary experiments showed consistent, favourable results for Adabelief in all setups.

[18] Kingma and Ba, *Adam*, 2017.

[19] Zhuang et al., *AdaBelief Optimizer*, 2020.

5.7 Hyperparameter search

In all experiments in section 6.1 and 6.2 we optimize SIREN hyperparameters ω_0 first and ω_0 hidden[20] for every combination of dataset D, architecture and latent embedding inference method. We optimize these hyperparameters by running respective experiments in a short training regime of 200 epochs, using Bayesian Search[21] with Hyperband early stopping[22] to guide the search, see section A.3.2 for more details. Preliminary experiments indicated that optimal ω_0 or coordinate multiplier values are very sensitive to many architectural- and data characteristics. Thus, these values are swept using the described procedure in all later experiments. See section A.1.2 for an analysis of optimal ω_0 values.

[20] ω_0 first and ω_0 correspond to parameters controlling all ω_0 values in the first layer, and all ω_0 values in the layers after the first layer.

[21] Snoek et al., *Practical Bayesian Optimization of Machine Learning Algorithms*, 2012.

[22] L. Li et al., *Hyperband*, 2018.

For all experiments in section 6.1 we additionally executed a grid search over different optimizers and learning rates. Resulting optimal parameters are reported in section A.3.2. These hyperparameters are kept constant in all later experiments.

5.8 Evaluation

Capturing the perceptual fidelity of audio reconstructions in a metric is not straightforward. Many audio-to-audio distances exist, each with different sensitivities, indicating the complexity of the problem.

For this work, we decided to start with an extensive selection of metrics based on recent and established research in audio reconstruction and generation and implementation availability. Then, we select metrics for representing background noise presence, sample quality, and overall quality for the NSYNTH datasets combined and the Speech Commands dataset as the main focus of our comparisons. For details on the metric selection procedure, see section 5.9.

5.8.1 Considered metrics

1. time-domain MSE
2. time-domain derivative MSE

3. time-domain MSE + derivative MSE
4. Signal-to-Noise Ratio (SNR)
5. Segmented SNR (SegSNR)
6. discrete Fourier transform (DFT) magnitude MSE
7. DFT magnitude wasserstein distance
8. DFT angular phase MSE
9. DFT magnitude pos/neg difference
10. Log-spectral distance (LSD)[23]
11. Multi resolution short-time Fourier transform (STFT) MSE
12. Multi resolution STFT wasserstein distance
13. Multi resolution STFT pos/neg difference
14. CSIG, CBAK, COVL[24]
15. PESQ[25]
16. FAD[26]
17. CDPAM[27]

[23] Gray and Markel, "Distance Measures for Speech Processing", 1976.

[24] Hu and Loizou, "Evaluation of Objective Quality Measures for Speech Enhancement", 2008.
[25] Rix et al., "Perceptual Evaluation of Speech Quality (PESQ)-a New Method for Speech Quality Assessment of Telephone Networks and Codecs", 2001.
[26] Kilgour et al., *Fr\'echet Audio Distance*, 2019.
[27] Manocha et al., *CDPAM*, 2021.

For NSYNTH datasets we use CSIG for representing signal quality, multi resolution STFT MSE for representing background noise presence and CDPAM for overall quality. For the SPEECHCOMMANDS dataset we use LSD for representing signal quality and overall quality and inverse CSIG representing background noise level.

CSIG is a composite of objective measures (spectral subtractive, subspace, statistical-model based, and Wiener algorithms) combined using multiple linear regression analysis to correlate highly with subjective quality ratings of signal distortion. The subjective quality ratings were obtained using the ITU-T P.835 methodology designed to evaluate the quality of enhanced speech along three dimensions: signal distortion, noise distortion, and overall quality.

Multi resolution STFT MSE is calculated by averaging STFT magnitude MSE's as shown in equation 5.2 with hamming windowing for N=4 window sizes: {400, 800, 1600 3200}:

$$Multi\,STFT\,MSE = \frac{1}{N}\sum_{i}^{N} |||STFT_i(X_i)| - |STFT_i(\{\phi_i(t_j)\}_{j=1}^{M})|||^2. \tag{5.2}$$

CDPAM is a contrastive learning based perceptual audio similarity metric parameterized as a deep neural network. It is trained on a dataset of audio similarity judgements based on triplet comparisons, asking subjects: "Is A or B closer to reference C?".

The log-spectral distance (LSD) is a distance measure (expressed in dB) between the power spectrum ($P(\omega)$) of waveforms. $P(\omega)$ is commonly defined as the Fourier transform of its autocorrelation function, where ω is the frequency. It can be approximated in discrete signals as follows:

$$P(\omega) \approx \frac{1}{M} \sum_{\tau=1}^{M} |STFT(\omega, \tau)|^2$$

Where M is the amount of frames in the resulting STFT transformation. LSD is then calculated as by combining all frequencies F as follows:

$$LSD = \frac{1}{F} \sum_{i=1}^{F} \left[\log \frac{P(\omega)}{\hat{P}(\omega)} \right]^2 . \tag{5.3}$$

For evaluating the absolute fidelity of amplitude function representations f_i), we measure mean squared error (MSE) between signals and their reconstructions as in equation 5.4:

$$MSE = \frac{1}{N} \frac{1}{M} \sum_{i=1}^{N} \sum_{j=1}^{M} \left\| \Phi(t_j, \mathbf{z_i}) - \mathrm{y}_i(t_j) \right\|^2 . \tag{5.4}$$

With the difficulties of measuring perceptual fidelity of audio waveforms in mind, we also report qualitative waveform analyses of representative samples of the NSYNTH Keyboard dataset (where helpful).

5.9 Metric selection

To select metrics that are sensitive to the characteristics of background noise, sample degradation and overall perceived quality in reconstructions of the models and data in our interest, we rate reconstructions of the baseline models and INR ablations[28] on background noise presence, sample quality, and overall quality for the three datasets.

[28] The INR decoder examined in ablation experiments was chosen based on MSE reconstruction scores for every dataset.

Then, we select a metric for representing background noise presence, sample quality and overall quality for the NSYNTH datasets combined and the Speech Commands dataset based on correlation coefficients between metrics and the respective ratings. Selected metrics are used for analysis and reported in all later sections of this chapter.

5.9.1 Subjective ratings of perceptual fidelity

Subjective ratings of overall quality, sample quality and background noise level of reconstructed samples for all tested parameterizations of ϕ_i in section 6.1 and 6.2, for all datasets are shown in figure 5.3.

For every combination of dataset and parameterization of ϕ_i, we selected the best training run as measured by MSE. Since differences between reconstructions can be subtle, ratings are relative to reconstructions of other architectures. Ratings are given by us, for a single sample. This sample was selected for every dataset by validating that the sample was representative for the performance of every parameterization of ϕ_i. A sample was determined to be representative for the performance of a decoder for a dataset if 10 different, randomly picked reconstructed samples of the same decoder for the same dataset were on par in terms of overall quality, sample quality and background noise level.

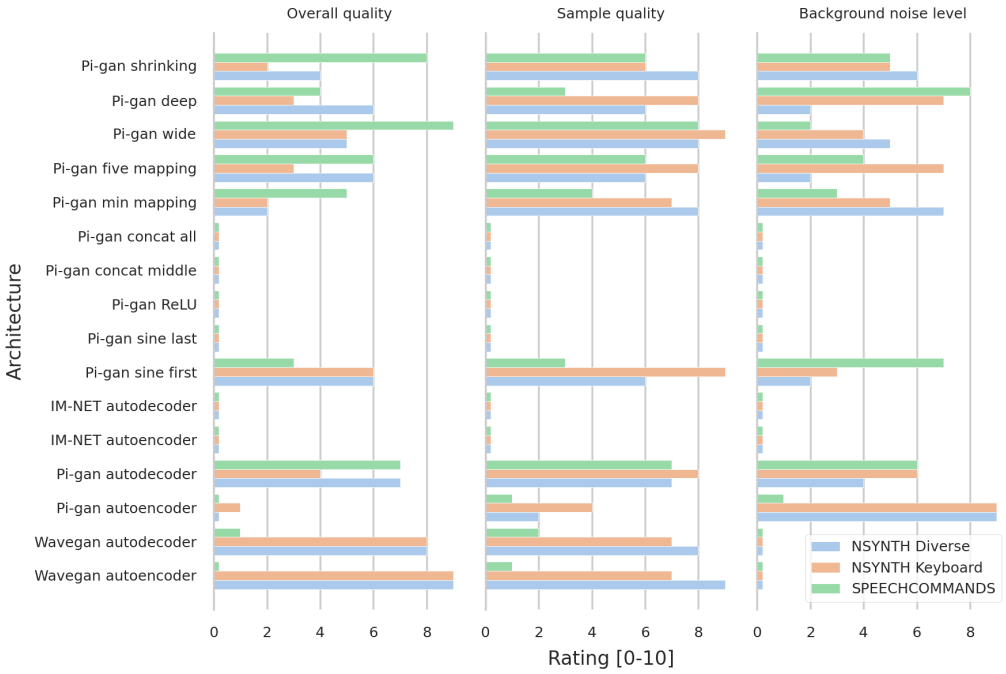


Figure 5.3: Subjective rating of reconstructed samples for all datasets for overall quality, sample quality and background noise level between 0 and 10. For quality ratings, higher is better. For background noise, higher is more noise. Ratings are relative to other setup reconstruction ratings within each dataset. All reported architectures are employed in an autodecoder setup where not specified.

1. WaveGAN autoencoder:
   (a) Best overall quality and background noise levels for both NSYNTH datasets.
   (b) Best sample quality for the NSYNTH diverse dataset.
   (c) Mostly silent reconstructions for the Speech Commands dataset.
2. WaveGAN autodecoder:
   (a) Performs very close to its autoencoder counterpart.
   (b) Comparing to WaveGAN autoencoder, reconstructions have slightly more clarity, but also contain modest high-pitched noise at higher volumes.
   (c) Produces audible, but barely legible signals for the Speech Commands dataset.

3. π-GAN autodecoder is the only setup capable of reconstructing legible words in the Speech Commands dataset.
4. π-GAN wide: best overall quality, sample quality and background noise level[29] for the Speech Commands dataset.
5. π-GAN sine first and π-GAN wide: best sample quality for the NSYNTH keyboard dataset.
6. All implicit architectures without sine activations in the first layer or without FiLM conditioning produce reconstructions with barely any audible sound for all datasets.
7. All implicit architectures that do produce audible sounds, introduce background noise in reconstructions to some extent, which is not the case for WaveGAN architectures.

[29] Ignoring setups that did not produce reasonable results.

5.9.2 Metric correlation

Because of the used rating method[30], Spearman's rank correlation coefficient is used for metric correlation calculations as it assesses monotonic relationships (whether linear or not). Absolute Spearman's rank correlation coefficients of all considered metric scores and ratings of representative samples are calculated within datasets. Resulting correlation coefficients are reported in figure 5.4. The NSYNTH datasets are aggregated by averaging correlation coefficients. If the sign of a correlation coefficient between the NSYNTH datasets is inconsistent, this coefficient is set to zero.

[30] Ratings were given relative to other reconstructions and should not be interpreted in an absolute manner, but rather as values to rank against each other.

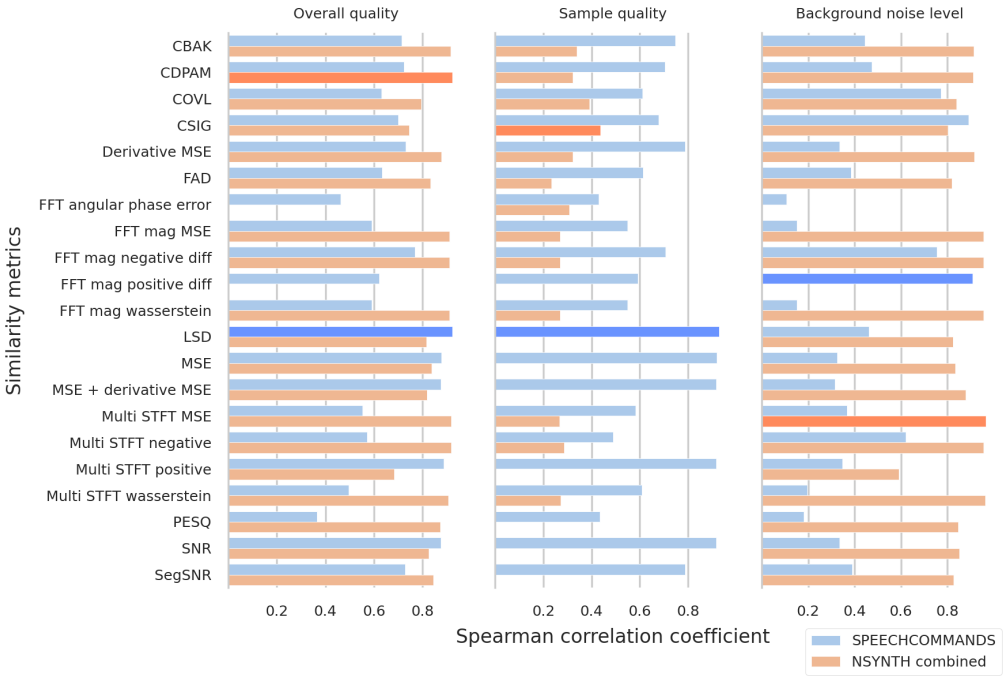


Figure 5.4: Absolute Spearman's rank correlation coefficients of all considered metrics with subjective ratings for overall quality, sample quality and background noise level. Metrics with the highest correlation within their dataset and rating category are colored darker. Missing bars indicate inconsistent correlation directions between datasets.

As shown in figure 5.4, the following metrics correlate best with our qualities of interest:

1. NSYNTH combined:
   (a) CDPAM correlates best with overall quality ratings.
   (b) CSIG correlates best with sample quality ratings.
   (c) Multi resolution STFT MSE correlates best with background noise level ratings.
2. Speech Commands:
   (a) LSD correlates best with overall quality and sample quality.
   (b) FFT magnitude positive difference correlates best with background noise level ratings.
   (c) CSIG's absolute Spearman correlation coefficient is not significantly lower than the more experimental FFT magnitude positive difference metric (0.909 versus 0.893).

We decide to continue with CSIG for representing background noise levels in Speech Commands. Note that CSIG is designed to correlate with sample quality, where a higher score is better. The correlation coefficient of CSIG and background noise level judgements in Speech Commands is + 0.893, which indicates that a higher CSIG score in Speech Commands *positively* correlates with more background noise. This is the opposite of what is expected.

5.9.3 Discussion

- Selected background noise metric for Speech Commands is CSIG *inversed*. This is not a scientifically backed metric, results should be interpreted with caution.
- Ratings of perceptual fidelity are given by a single person. For more robust results, these should be rated by more people. Small differences in ratings could be subject to noise.
- The ranking of subjective ratings of perceptual fidelity is mostly consistent with the ranking of selected perceptual fidelity evaluation metrics reported in section 6.1 and 6.2. This indicates that the quality of reconstructions within dataset and parameterization of ϕ_i combinations and the selected perceptual fidelity evaluation metrics are consistent.
- Without consistency in quality of reconstructions within dataset and parameterization of ϕ_i combinations the selection procedure for representative samples to be rated[31] would have been infeasible.
- CBAK, COVL, CSIG and PESQ are each designed for measuring speech waveform quality and specifically to represent one of the perceptual fidelity categories we consider: background noise level, overall-, sample- and overall quality, respectively. These metrics have been around for $\pm$ 15 years, and are still used in research as of today. Still, they correlate relatively poorly with our

[31] The selection procedure for representative samples to be rated is described in the beginning of this chapter.

subjective ratings of perceptual fidelity in their respective categories. Assuming that similar perceptual fidelity degradation can be caused by different kinds of low-level waveform perturbations, this seems to indicate that these metrics are sensitive to other perturbations than those introduced in by INRs in our experiments.

- To our knowledge no previous research has reported similar observations as in statement 5.9.3. Thus, considering the distinct characteristics of INR's compared to other audio reconstruction/synthesis methods, we hypothesize that INRs can introduce unique low-level waveform perturbations. When researching perceptual fidelity of INR-based waveform synthesis, extra care should be taken when picking evaluation metrics.

6 Results and Discussion

In this chapter we examine the ability of several implicit and convolutional network architectures conditioned on latent embeddings on the task of reconstructing different sets of audio waveforms as described in section 4.1. We report the results of the experiments as described in chapter 5 and discuss them.

In section 6.1 we evaluate two different previously proposed architecures to a convolutional architecture as a baseline, and compare the effect of different latent embedding inference methods.

Then, in section 6.2, we conduct a series of ablation experiments on the best performing conditional INR decoder to find an answer to what architecture charecteristics contribute how much to reconstruction performance. Then we try to find a concrete relation between ω_0 values troughout siren networks and output signal frequencyes.

Finally, in section 6.3, we compare the effect of progressive activation scaling and weight regularization as described in section 5.4 for countering high frequency noise observed in reconstructions of earlier experiments.

6.1 Baseline experiments

In this section we aim to answer the following questions:

- ○ What latent embedding inference method is optimal for exploring the representation characteristics specific to various conditional INR parameterizations?
- ◊ Are previously proposed conditional INR parameterizations capable of representing a distribution of audio waveforms with similar perceptual- and absolute reconstruction fidelity as transposed convolution based architectures designed for this purpose?
- △ What are the main shortcomings of audio waveform representations learned by these conditional INRs, and how can these be explained theoretically?

We report experimental results on the task of learning distributions of continuous audio waveform representations as described in section 4.1. We compare the perceptual- and absolute fidelity of

learned representations f_i of different recently proposed parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ (IM-NET, π-GAN and WaveGAN, see section 4.6), combined with diferent latent embedding inference methods (autoencoder and autodecoder, see section 4.2) on the datasets described in section 5.5.

Observations. We summarize the most important observations regarding the questions we aim to answer in this section:

- ○ Autoencoders perform worse than autodecoders in all compared setups, and that these negative effects are more significant for π-GAN.
- ◊ In autodecoder setups π-GAN outperforms the other architectures in absolute fidelity on all datasets.
- ◊ IM-NET shows results close to silence.
- △ For perceptual fidelity WaveGAN slightly outperforms π-GAN, which shows local waveform inconsistencies in NSYNTH datasets, but fails to reconstruct Speech Commands reasonably.

Conclusions. We summarize the key findings of our analysis of the experimental results reported in this section:

- ○ Autodecoders are optimal for exploring the representation characteristics of INR parameterizations.
- ◊ π-GAN is better at representing a distribution of audio waveforms than transposed convolution based architectures regarding absolute reconstruction fidelity, but π-GAN is worse regarding perceptual fidelity in less diverse datasets, due to local inconsistencies in reconstruced waveforms.
- △ Local waveform inconsistencies are a main shortcoming of INRs, and periodic nonlinearities play an important role in this, as well as their success.

6.1.1 Results

See table 6.1 for the perceptual fidelity scores for NSYNTH datasets and table 6.2 for the perceptual fidelity scores for Speech Commands. Absolute fidelity scores for all datasets are reported in table 6.3. Figure 6.1 shows the first 200 samples of a waveform from the NSYNTH keyboard dataset and reconstructions of all baseline setups.[1]

[1] Note that this is a small section of a single waveform. Although the described differences between architectures' reconstructions are present in most waveforms we examined, conclusions should be made with caution since the majority of waveform samples remain unexamined.

Perceptual fidelity evaluation

Considering tables 6.1 and table 6.2, we make the following observations regarding the perceptual fidelity of f_i for the different parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ and latent embedding inference methods:

1. WaveGAN autodecoder:
   - Best perceptual fidelity evaluations across all metrics for NSYNTH keyboard
   - best CSIG (sample quality) and Multi resolution STFT MSE (background noise level) scores for NSYNTH diverse.
2. π-GAN autodecoder:
   - best LSD (overall- and sample quality) scores for Speech Commands.
   - best CDPAM (overall quality) scores for NSYNTH diverse.
   - Perceptual fidelity evaluations close to WaveGAN autodecoder, except for Multi resolution STFT MSE (background noise level).
3. IM-NET: Overall- and sample quality scores of all IM-NET architectures in the NSYNTH datasets are close to those of silence.
4. Autodecoder setups consistently outperform autoencoder setups.

	CDPAM: Overall quality		CSIG: Sample quality		Multi STFT MSE: Noise	
Architecture	Diverse	Keyboard	Diverse	Keyboard	Diverse	Keyboard
WaveGAN AE	0.67 $\pm$ 0.3	0.67 $\pm$ 0.46	-0.09 $\pm$ 2.05	-3.1 $\pm$ 3.09	0.09 $\pm$ 0.07	0.1 $\pm$ 0.06
IM-NET AE	1.83 $\pm$ 0.02	1.60 $\pm$ 0.01	-4.24 $\pm$ 0.14	-5.76 $\pm$ 0.36	0.15 $\pm$ 0.0	0.12 $\pm$ 0.0
π-GAN AE	0.72 $\pm$ 0.25	0.64 $\pm$ 0.21	-2.01 $\pm$ 2.03	-2.98 $\pm$ 2.63	0.22 $\pm$ 0.09	0.21 $\pm$ 0.04
WaveGAN AD	0.43 $\pm$ 0.21	**0.43** $\pm$ 0.27	**1.6** $\pm$ 1.4	**-1.66** $\pm$ 2.61	**0.06** $\pm$ 0.05	**0.06** $\pm$ 0.04
IM-NET AD	1.81 $\pm$ 0.0	1.61 $\pm$ 0.0	-4.14 $\pm$ 0.04	-6.76 $\pm$ 0.16	0.13 $\pm$ 0.0	0.1 $\pm$ 0.0
π-GAN AD	**0.35** $\pm$ 0.24	0.48 $\pm$ 0.18	1.38 $\pm$ 1.84	-1.97 $\pm$ 2.45	0.06 $\pm$ 0.02	0.14 $\pm$ 0.04
Silence	1.65 $\pm$ 0.0	1.77 $\pm$ 0.0	-1.31 $\pm$ 0.0	-4.62 $\pm$ 0.0	0.16 $\pm$ 0.00	0.16 $\pm$ 0.00
White noise	1.06 $\pm$ 0.27	1.06 $\pm$ 0.32	-5.43 $\pm$ 3.14	-9.31 $\pm$ 2.59	2.27 $\pm$ 0.06	2.32 $\pm$ 0.04

Table 6.1: Evaluation metrics CDPAM, CSIG and multi resolution STFT MSE scores of all baseline setups for NSYNTH datasets. For CDPAM and multi resolution STFT MSE a lower score is better, for CSIG representing sample quality in NSYNTH datasets a higher score is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

Architecture	LSD: Overall / Sample quality	CSIG: Noise
WaveGAN AE	0.69 $\pm$ 0.44	**0.51** $\pm$ 0.74
IM-NET AE	0.72 $\pm$ 0.0	-1.38 $\pm$ 0.02
π-GAN AE	0.48 $\pm$ 0.21	0.78 $\pm$ 1.1
WaveGAN AD	0.38 $\pm$ 0.27	0.88 $\pm$ 1.03
IM-NET AD	0.49 $\pm$ 0.0	-0.68 $\pm$ 0.04
π-GAN AD	**0.27** $\pm$ 0.12	0.59 $\pm$ 1.26
Silence	0.74 $\pm$ 0.0	-2.05 $\pm$ 0.0
White noise	7.27 $\pm$ 0.07	-1.24 $\pm$ 1.93

Table 6.2: LSD $\times 10$ and CSIG scores of all baseline setups for the Speech Commands dataset. For both metrics lower is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

Absolute fidelity evaluation

Considering table 6.3, we make the following observations regarding the absolute fidelity of reconstructed samples f_i for the different parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ and latent embedding inference methods:

1. π-GAN autodecoder: best absolute fidelity for all datasets.
2. Autodecoder setups consistently outperform autoencoder setups.
3. IM-NET: MSE close to silence for all datasets.

Architecture		NSYNTH Keyboard		NSYNTH Diverse		Speech Commands	
WaveGAN	AE	25.68	± 2.59	12.50	± 0.38	6.64	± 0.59
IM-NET	AE	74.35	± 0.13	44.12	± 0.04	6.73	± 0.01
π-GAN	AE	13.21	± 0.11	28.89	± 0.86	2.75	± 0.09
WaveGAN	AD	8.67	± 0.83	4.05	± 0.42	2.22	± 0.55
IM-NET	AD	70.35	± 0.23	41.19	± 0.08	6.23	± 0.02
π-GAN	AD	**4.48**	± 1.02	**1.32**	± 0.3	**0.85**	± 0.01
Silence		78.74	± 0.0	46.19	± 0.0	7.48	± 0.0
White noise		1094.74	± 1313.56	1061.49	± 1212.03	1023.18	± 1111.56

Table 6.3: Mean and standard deviation of MSE $\times 10^3$ statistics of batches (128 samples) of synthesized samples in the 5000th epoch of all models with best performing hyperparameters in baseline setups. Silence and white noise ($\mu = 0, \sigma = 1$) errors reported for reference.

Qualitative waveform analysis | NSYNTH keyboard

Considering figure 6.1, we make the following observations regarding the qualities of the reconstructed waveforms of the different parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ and latent embedding inference methods:

1. WaveGAN reconstructions are significantly smoother.

2. Autoencoder setups seem to have more difficulty reaching more extreme amplitudes.

3. The first 25 samples of the original waveform are silent, π-GAN autodecoder's reconstruction is able to reconstruct this best. For the rest of the waveform, WaveGAN autodecoder's reconstructions show the most resemblence to the original due to their smoothness.

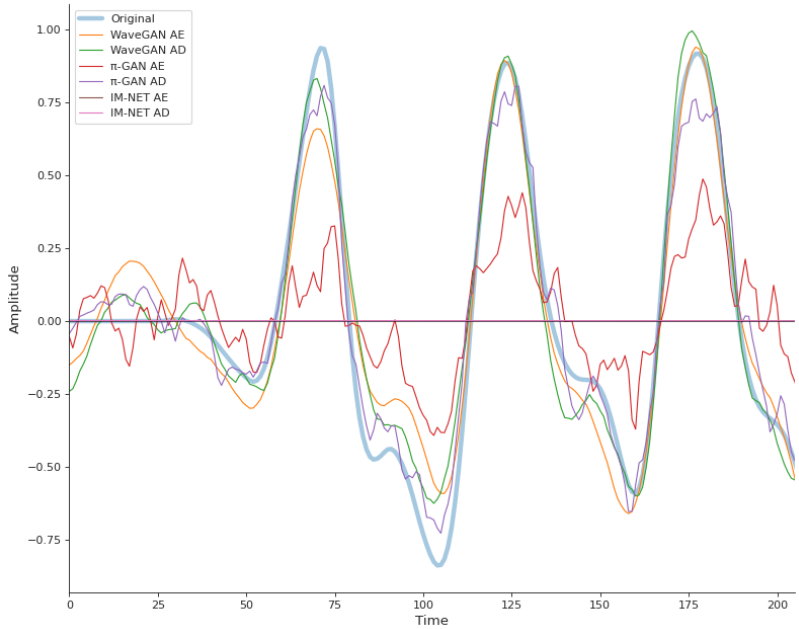


Figure 6.1: First 200 timepoints of a waveform validated to be representative as described in section 5.9.1 of the NSYNTH keyboard dataset and reconstructions of baseline setups.

6.1.2 Discussion

Interpreting results of all baseline experiments in the review above, the following key findings emerge regarding the effectiveness of autoencoders and autodecoders for exploring characteristics of $\boldsymbol{\phi_i}$&$\boldsymbol{p(\theta|\mathbf{z})}$, the capabilities of previously proposed INRs and their main shortcomings:

- ○ **Autodecoders are optimal for exploring characteristics of $\boldsymbol{\phi_i}$.** All parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ employed in an autoencoder setup are consistently less expressive, and perform worse than those employed in an autodecoder setup (see observations 2 and 2. But, this effect is not of the same magnitude for every combination of dataset and decoder. Comparing performance of autodecoders to autoencoders for WaveGAN, all scores are reduced with equal proportions for all datasets, indicating autoencoders slow learning. While for π-GAN scores degrade with a significantly higher ratio for NSYNTH Diverse and Speech Commands. We hypothesize this indicates compatibility issues with the encoder, explained as follows:

  (a) The convolutional encoder used in autoencoder setups learns significantly faster with more consistent gradients between samples X_i, in more uniform datasets, similar to observations in statement /labelenum:base:wave:slow.

  (b) π-GAN delivers noisy gradients. See statement 0b.

  (c) In more diverse datasets less consistent gradients between samples X_i combined with the noisy backpropagation of π-GAN prohibit the encoder from learning, leaving both the decoder and encoder without a learning signal.

  (d) For NSYNTH Keyboard, the most uniform dataset, the convolutional encoder receives relatively consistent gradients. Eventually enough to start forming somewhat meaningful latent embeddings, after which the training of both the encoder and decoder can make progress.

  Thus, we argue that autodecoders are optimal for exploring the representation characteristics of decoder networks.

- ◊ **IM-NET is not suited for representing audio waveforms.** In line with the results shown in figure 4.7, INRs without sine activations struggle to represent signals with high detail. On top of that, results in section 6.2 indicate that FiLM modulation is also essential for π-GAN in the task at hand. IM-NET has to do without both these parameterizations, which make it signifcantly less expressive.

- △ **π-GAN closely follows the objective function.** π-GAN strongly outperforms all other architectures in absolute fidelity, which is directly optimized by the objective function. This shows π-GAN is very expressive. However, MSE does not correlate well with the

perceptual qualities we aim to optimize. This indicates that, when optimizing for perceptual fidelity, π-GAN might benefit from an objective function that does align better with these qualities.

△ **π-GAN requires locally consistent objectives.** We tried many of the considered metrics that were differentiable as objective functions, but none of them resulted in reconstruction quality close to that of the objective function in equation 4.7. An in-depth comparison of various established spectral and perceptual audio similarity distances by Turian et al.[2] maps their output for several signal perturbations. The authors find that many exhibit roughness, resulting in inconsistent gradients within small perturbations. Still, many audio synthesis architectures in previous literature are reported to benefit from these spectral and perceptual objective functions[3]. These observations indicate that π-GAN is significanty less robust against label noise, it has more strict requirements regarding the local consistency and convexity of loss landscapes.

[2] Turian and Henry, *I'm Sorry for Your Loss*, 2020.

[3] Défossez et al., *SING*, 2018.

△ We see two potential sources, likely to be mutually reinforcing, for π-GAN's dependence on local consistency in loss functions:

(a) **Low dimensional input of INRs**: for any neural network to learn efficiently in supervised setups, close to identical input needs to result in consistent gradient directions[4]. If this is not the case, backpropagated errors will cancel each other out. Since the input of INR's are low dimensional geometric coordinates, it processes relatively many close to identical inputs. Assuming that gradients of loss functions which show inconsistencies with respect to small signal perturbations also show inconsistencies within geometrically close measurements of a signal, it is expected that such loss functions seriously impede learning for INR's.

[4] Chapelle et al., *Semi-Supervised Learning*, 2006, smoothness assumption of supervised learning: If two points x_1, x_2 are close, then so should be the corresponding outputs y_1, y_2.

(b) **Periodic activation functions**: periodic activation functions are nonmonotonic, classically viewed as an undesired characteristic for activation functions[5]. Nonmonotonic activation functions create geometrically local inconsistent gradients at stationary points, points where the derivative is zero. In the case of periodic activation functions, stationary points appear repeatedly. The density of stationary points within the range of intermediate layer activations is determined by the input scaling factor ω_0. Thus, it is expected that with high ω_0's, required for modelling signals with fine details, gradients in π-GAN can quickly get noisy as they travel deeper in the network.[6] This does not have to be a problem (larger batch sizes help), but it makes it significantly harder to learn with noisy loss functions.

[5] There have been several nonmonotonic activation functions proposed in recent work (Swish, Mish) often reported to outperform ReLU on a battery of tasks

◊ **π-GAN handles diverse datasets well.** π-GAN autodecoder is the only setup capable of achieving proper representations for the Speech Commands dataset (see observation 3), WaveGAN autodecoder shows the best representations for NSYNTH Keyboard,

[6] **Stationary points introduce noise.** The hypothesis that stationary points in layer activations create noisy signals aligns with the following observations:

i. π-GAN benefits from larger batch sizes.
ii. π-GAN requires low learning rates.
iii. Averaging the output of multiple π-GANs trained with different seeds significantly reduces background noise. This indicates that the noise distribution is centered around zero, as expected if introduced by stationary points.

and for NSYNTH Diverse metric scores are not unanimous, both architectures perform similar. This aligns perfectly with the diversity of samples within the different datasets. NSYNTH Keyboard is composed to be uniform, NSYNTH Diverse to contain a higher diversity of timbres and pitches. Speech Commands is spectrally most complex, and sample onset times are unaligned, see figures 5.2 and 5.1, and section 5.5. This shows that when learning to represent a given dataset, WaveGAN thrives when the dataset is more spectrally uniform and requires it to be aligned[7]. While π-GAN is significantly more expressive, and does not require alignment. However, this expressivity comes at a cost: less smoothness in representations, resulting in audible noise.

[7] Generative architectures generally perform significantly better when signals in datasets are aligned.

◊ **π-GAN shows less noise in NSYNTH diverse,** It is possible this has to do with the specific timbral characteristics of instrument samples in NSYNTH Diverse, noisy components to be specific[8] in e.g. guitar samples. We hypothesize that under certain circumstances, noisy components in a dataset are able to "absorb" noise introduced by periodic activation functions, making them less present in other parts of represented signals. This would be caused by similar principles as causing the benefits of noise injection in convolutional GANs[9], although it is harder to make sense of this with per sample reconstruction supervision as in our experiments.

[8] Keyboard samples contain significantly less noisy components. They are introduced by snare plucking and effects such as overdrive.

[9] Feng et al., *On Noise Injection in Generative Adversarial Networks*, 2021; Karras, Laine, and Aila, *A Style-Based Generator Architecture for Generative Adversarial Networks*, 2019.

△ **Weak oscillatory bias in sine compositions.** The observed expressivity (when exposed to a smooth learning signal) and relatively non-smooth reconstructions of π-GAN indicate that the composition of sine activations does not contain a strong oscillatory bias, at least not in the presence of high activation scalings required represent our datasets.

■ **Full π-GAN only ϕ_i that fits parameter budget.** Out of all three compared architectures, π-GAN is the only architecture which did not have to be changed to fit the parameter budget we enforced. As architecture hyperparameters can be carefully tuned, this could affect results. For IM-NET this does not seem the case, preliminary experiments on smaller datasets showed similar results between full and trimmed versions of IM-NET.

■ **Hypernetwork comparison absence.** Having a setup with hypernetwork based conditioning would provide a more complete picture, but because of difficulties with initialization instabilities and time constraints we decided to leave this for future work.

6.2 Ablation experiments

In this section we aim to answer the following questions:

△ What are the main shortcomings of the process required to learn audio waveform representations by conditional INRs?

□ Which conditional INR model hyperparameters are of significant influence in perceptual- and absolute reconstruction fidelity when representing different distributions of audio waveforms?

Based on the absolute fidelity of all architectures and latent embedding inference methods, π-GAN in combination with an autodecoder was chosen to continue experiments with in this section. We report the experimental results on the task of learning distributions of continuous audio waveform representations as described in section 4.1. We compare the perceptual- and absolute fidelity of learned representations f_i on the datasets described in section 5.5 of several ablations of π-GAN:

- Activation function
- Conditioning mechanism
- Latent mapping network
- Network shape

See section 5.3 for a more detailed overview of the ablations' specific parameterizations.

Observations. We summarize the most important observations regarding the questions we aim to answer in this section:

□ Sine activations in the first layer and FiLM conditioning are essential for representing any distribution of audio waveforms.

□ Replacing the sine activations with ReLUs in all but the first layer results in improved representation fidelity for NSYNTH keyboard and similar results for Speech Commands.

□ Minimizing the amount of layers in the mapping network improves the fidelity of NSYNTH Keyboard and Speech Commands representations.

□ Decreasing the depth and increasing the width of π-GAN results in the overall best fidelity of reconstructions for NSYNTH Keyboard and Speech Commands, and close to the best fidelity for NSYNTH Diverse.

□ The original π-GAN parameterization outperforms all ablations in NSYNTH Diverse's fidelity.

Conclusions. We summarize the key findings of our analysis of the experimental results reported in this section:

△ Fidelity of learned representations is highly sensitive to expressivity of parameterizations.

△ Optimizing ω_0's in a short training regime creates a pressure towards highly expressive ω_0's, forming a significant shortcoming in the process required to learn good representations of distributions of audio waveforms with limited computational resources.

- When dealing with pressure towards highly expressive ω_0's in more spectrally uniform datasets, less expressive parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ thrive.
- π-GAN Wide shows optimal representation fidelity due to expressity limiting factors, a more convex optimization landscape and more fine-grained FiLM control.

6.2.1 Results: Activation function

First, we examine the effect of changing the activation function in π-GAN. We evaluate the performance of π-GAN when all sine activations are replaced by ReLU's, and when all but the first or the last are replaced by ReLUs. See table 6.4 for the perceptual fidelity scores for NSYNTH datasets, and table 6.5 for the perceptual fidelity scores for Speech Commands. Absolute fidelity scores for all datasets are reported in table 6.6.

Perceptual fidelity evaluation

Considering tables 6.4 and table 6.5, we make the following observations regarding the perceptual fidelity of reconstructed samples f_i for the different parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ and latent embedding inference methods:

1. All architectures without a sine activation in the first layer drastically drop performance for Overall- and sample quality metrics, resulting in scores close to silence.
2. Sine first, others ReLU, outperforms π-GAN in the NSYNTH keyboard dataset across all metrics by a small margin.
3. For the NSYNTH diverse and Speechcommands datasets π-GAN consistently outperforms Sine first, others ReLU by a small margin.

	CDPAM: Overall quality		CSIG: Sample quality		Multi STFT MSE: Noise	
Architecture	Diverse	Keyboard	Diverse	Keyboard	Diverse	Keyboard
π-GAN	**0.35** $\pm$ 0.24	0.48 $\pm$ 0.18	**1.38** $\pm$ 1.84	-1.97 $\pm$ 2.45	**0.06** $\pm$ 0.02	0.14 $\pm$ 0.04
- Sine first	0.43 $\pm$ 0.25	**0.47** $\pm$ 0.19	0.69 $\pm$ 1.68	**-1.76** $\pm$ 2.07	0.09 $\pm$ 0.04	0.12 $\pm$ 0.05
- Sine last	1.81 $\pm$ 0.0	1.61 $\pm$ 0.0	-4.14 $\pm$ 0.04	-6.76 $\pm$ 0.16	0.13 $\pm$ 0.0	0.1 $\pm$ 0.0
- All ReLU	1.8 $\pm$ 0.0	1.38 $\pm$ 0.0	-4.63 $\pm$ 0.05	-6.82 $\pm$ 0.14	0.12 $\pm$ 0.0	**0.09** $\pm$ 0.0
Silence	1.65 $\pm$ 0.0	1.77 $\pm$ 0.0	-1.31 $\pm$ 0.0	-4.62 $\pm$ 0.0	0.16 $\pm$ 0.00	0.16 $\pm$ 0.00
White noise	1.06 $\pm$ 0.27	1.06 $\pm$ 0.32	-5.43 $\pm$ 3.14	-9.31 $\pm$ 2.59	2.27 $\pm$ 0.06	2.32 $\pm$ 0.04

Table 6.4: Evaluation metrics CDPAM, CSIG and multi resolution STFT MSE scores of activation ablation setups for NSYNTH datasets. For CDPAM and multi resolution STFT MSE a lower score is better, for CSIG representing sample quality in NSYNTH datasets a higher score is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

Absolute fidelity evaluation

Considering table 6.6, we make the following observations regarding the absolute fidelity of reconstructed samples f_i for the different parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ regarding activation functions:

Architecture	LSD: Overall + Sample quality	CSIG: Noise
π-GAN	**0.27** $\pm$ 0.12	0.59 $\pm$ 1.26
- Sine first	0.31 $\pm$ 0.17	1.39 $\pm$ 1.06
- Sine last	0.49 $\pm$ 0.0	**-0.68** $\pm$ 0.04
- All ReLU	0.45 $\pm$ 0.0	-0.67 $\pm$ 0.04
Silence	0.74 $\pm$ 0.0	-2.05 $\pm$ 0.0
White noise	7.27 $\pm$ 0.07	-1.24 $\pm$ 1.93

Table 6.5: LSD $\times 10$ and CSIG scores of all activation ablation setups for the Speech Commands dataset. For both metrics lower is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

1. π-GAN: best absolute fidelity for NSYNTH Diverse and Speech Commands.

2. π-GAN Sine first: best absolute fidelity for NSYNTH Keyboard.

Architecture	NSYNTH Diverse		NSYNTH Keyboard		Speech Commands	
π-GAN	**1.32**	$\pm$ 0.3	4.48	$\pm$ 1.02	**0.85**	$\pm$ 0.01
- Sine first	3.1	$\pm$ 1.57	**4.29**	$\pm$ 0.48	1.25	$\pm$ 0.03
- Sine last	34.45	$\pm$ 0.0	44.93	$\pm$ 0.0	2.42	$\pm$ 0.0
- All ReLU	38.57	$\pm$ 0.0	49.1	$\pm$ 0.0	2.39	$\pm$ 0.0
Silence	46.19	$\pm$ 0.0	78.74	$\pm$ 0.0	7.48	$\pm$ 0.0
White noise	1061.49	$\pm$ 12.03	1094.74	$\pm$ 13.56	1023.18	$\pm$ 11.56

Table 6.6: Mean and standard deviation of MSE $\times 10^3$ statistics of batches (128 samples) of synthesized samples in the 5000th epoch of all models with best performing hyperparameters in activation ablation setups. Silence and white noise ($\mu = 0, \sigma = 1$) errors reported for reference.

6.2.2 Results: Conditioning mechanism

Next, we examine the effect of changing the conditioning mechanism of π-GAN. We evaluate the performance of π-GAN when the network is conditioned by latent concatenation instead of FiLM. We compare concatenation in the first and middle layer, as in DeepSDF[10], and concatenation in every layer until the second-last layer, as in IM-NET[11]. Tables are moved to the appendix becuase do not bring new information next to the observations made below. See table 1 for the perceptual fidelity scores for NSYNTH datasets, and table 3 for the perceptual fidelity scores for Speech Commands. Absolute fidelity scores for all datasets are reported in table 3.

[10] Park et al., *DeepSDF*, 2019.

[11] Chen and Hao Zhang, *Learning Implicit Fields for Generative Shape Modeling*, 2019.

Perceptual- and absolute fidelity evaluation

Considering tables 1, 2 and **??** we see that for all datasets, all architectures without FiLM conditioning drastically drop performance for absolute fidelity, overall-, and sample quality metrics, resulting in scores close to silence.

6.2.3 Results: Latent mapping network

Next, we examine the effect of changing the depth of the latent mapping network implemented in π-GAN. We evaluate the performance of π-GAN when the latent mapping network is deeper than in the original implementation of π-GAN, five layers instead of three, and when the latent mapping network is minimized, consisting of only one layer. Note that in this comparison the total parameter count of the architectures is not kept constant, which is the case for all other experiments. See table 6.7 for the perceptual fidelity scores for

NSYNTH datasets, and table 6.8 for the perceptual fidelity scores for Speech Commands. Absolute fidelity scores for all datasets are reported in table 6.9.

Perceptual fidelity evaluation

- For the NSYNTH keyboard dataset:
  1. Minimal mapping layers result in a significant improvement for sample quality and background noise level.
  2. Five mapping layers result in a slight improvement in overall quality, where minimal mapping layers result in a slight decrease in overall quality, but differences are insignificant.
- For Speech Commands minimizing the amount of latent mapping layers shows positive results for all metrics.
- For the NSYNTH diverse dataset reconstructions of the original latent mapping network of π-GAN show best results across all metrics, but differences with five mapping layers are insignificant.

	CDPAM: Overall quality		CSIG: Sample quality		Multi STFT MSE: Noise	
Architecture	Diverse	Keyboard	Diverse	Keyboard	Diverse	Keyboard
π-GAN	**0.35** $\pm$ 0.24	0.48 $\pm$ 0.18	**1.38** $\pm$ 1.84	-1.97 $\pm$ 2.45	**0.06** $\pm$ 0.02	0.14 $\pm$ 0.04
- Min mapping	0.56 $\pm$ 0.26	0.49 $\pm$ 0.21	0.08 $\pm$ 1.66	**-0.69** $\pm$ 2.67	0.14 $\pm$ 0.05	**0.11** $\pm$ 0.03
- Five mapping	0.36 $\pm$ 0.24	**0.47** $\pm$ 0.17	1.07 $\pm$ 1.98	-1.93 $\pm$ 2.49	0.06 $\pm$ 0.02	0.14 $\pm$ 0.03
Silence	1.65 $\pm$ 0.0	1.77 $\pm$ 0.0	-1.31 $\pm$ 0.0	-4.62 $\pm$ 0.0	0.16 $\pm$ 0.00	0.16 $\pm$ 0.00
White noise	1.06 $\pm$ 0.27	1.06 $\pm$ 0.32	-5.43 $\pm$ 3.14	-9.31 $\pm$ 2.59	2.27 $\pm$ 0.06	2.32 $\pm$ 0.04

Table 6.7: Evaluation metrics CDPAM, CSIG and multi resolution STFT MSE scores of latent mapping network ablation setups for NSYNTH datasets. For CDPAM and multi resolution STFT MSE a lower score is better, for CSIG representing sample quality in NSYNTH datasets a higher score is better. Silence and white noise ($\mu = 0$, $\sigma = 1$) scores reported for reference.

Architecture	LSD: Overall / Sample quality	CSIG: Noise
π-GAN	0.27 $\pm$ 0.12	0.59 $\pm$ 1.26
- Min mapping	**0.22** $\pm$ 0.13	**-0.52** $\pm$ 1.19
- Five mapping	0.28 $\pm$ 0.12	0.19 $\pm$ 1.26
Silence	0.74 $\pm$ 0.0	-2.05 $\pm$ 0.0
White noise	7.27 $\pm$ 0.07	-1.24 $\pm$ 1.93

Table 6.8: LSD $\times 10$ and CSIG scores of all latent mapping network ablation setups for the Speech Commands dataset. For both metrics lower is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

Absolute fidelity evaluation

Considering table 6.9, we make the following observations regarding the absolute fidelity of reconstructed samples f_i for the different latent mapping network depths:

1. Minimizing mapping layers has a positive effects on absolute fidelity of reconstructions in NSYNTH Keyboard and Speech Commands, but a negative effect for NSYNTH Diverse.
2. Five latent mapping layers on average performs similar to three mapping layers (regular π-GAN), but shows more consistent results between training runs.

Architecture	NSYNTH Diverse		NSYNTH Keyboard		Speech Commands	
π-GAN	**1.32**	$\pm$ 0.3	4.48	$\pm$ 1.02	0.85	$\pm$ 0.01
- Min mapping	6.57	$\pm$ 0.17	**3.83**	$\pm$ 0.28	**0.67**	$\pm$ 0.0
- Five mapping	1.33	$\pm$ 0.07	4.18	$\pm$ 0.29	0.91	$\pm$ 0.01
Silence	46.19	$\pm$ 0.0	78.74	$\pm$ 0.0	7.48	$\pm$ 0.0
White noise	1061.49	$\pm$ 12.03	1094.74	$\pm$ 13.56	1023.18	$\pm$ 11.56

Table 6.9: Mean and standard deviation of MSE $\times 10^3$ statistics of batches (128 samples) of synthesized samples in the 5000th epoch of all models with best performing hyperparameters in mapping network ablation setups. Silence and white noise ($\mu = 0, \sigma = 1$) errors reported for reference.

6.2.4 Results: Network shape

Finally, we test the effects of network shape ablations on waveform reconstruction evaluations. Parameter counts are kept constant. See table 6.10 for the perceptual fidelity scores for NSYNTH datasets, and table 6.11 for the perceptual fidelity scores for Speech Commands. Absolute fidelity scores for all datasets are reported in table 6.12.

Perceptual fidelity evaluation

Considering tables 6.10 and table 6.11, we make the following observations regarding the perceptual fidelity of reconstructed samples f_i for the different network shape parameterizations of ϕ_i & $p(\theta|\mathbf{z})$:

- For the NSYNTH keyboard and Speech Commands datasets, π-GAN Wide architecture outperforms original and other network shape ablation architectures acrosss all metrics (except for background noise levels in Speech Commands)
- π-GAN Shrinking showing best background noise levels for Speech Commands, with a slight decrease in overall- and sample quality compared to the original network shape.
- For the NSYNTH diverse dataset, all changes to the original network shape result in worse reconstruction evaluations across all metrics, although differences with π-GAN Deep are small.

	CDPAM: Overall quality		CSIG: Sample quality		Multi STFT MSE: Noise	
Architecture	Diverse	Keyboard	Diverse	Keyboard	Diverse	Keyboard
π-GAN	**0.35** $\pm$ 0.24	0.48 $\pm$ 0.18	**1.38** $\pm$ 1.84	-1.97 $\pm$ 2.45	**0.06** $\pm$ 0.02	0.14 $\pm$ 0.04
- Wide	0.56 $\pm$ 0.28	**0.36** $\pm$ 0.2	1.28 $\pm$ 1.37	**-0.05** $\pm$ 2.73	0.11 $\pm$ 0.05	**0.08** $\pm$ 0.02
- Deep	0.35 $\pm$ 0.22	0.52 $\pm$ 0.21	0.97 $\pm$ 1.93	-2.23 $\pm$ 2.47	0.06 $\pm$ 0.02	0.14 $\pm$ 0.05
- Shrinking	0.59 $\pm$ 0.28	0.52 $\pm$ 0.22	0.99 $\pm$ 1.25	-0.2 $\pm$ 2.65	0.13 $\pm$ 0.05	0.1 $\pm$ 0.03
Silence	1.65 $\pm$ 0.0	1.77 $\pm$ 0.0	-1.31 $\pm$ 0.0	-4.62 $\pm$ 0.0	0.16 $\pm$ 0.00	0.16 $\pm$ 0.00
White noise	1.06 $\pm$ 0.27	1.06 $\pm$ 0.32	-5.43 $\pm$ 3.14	-9.31 $\pm$ 2.59	2.27 $\pm$ 0.06	2.32 $\pm$ 0.04

Table 6.10: Evaluation metrics CDPAM, CSIG and multi resolution STFT MSE scores of network shape ablation setups for NSYNTH datasets. For CDPAM and multi resolution STFT MSE a lower score is better, for CSIG representing sample quality in NSYNTH datasets a higher score is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

Architecture	LSD: Overall / Sample quality	CSIG: Noise
π-GAN	0.27 $\pm$ 0.12	0.59 $\pm$ 1.26
- Wide	**0.2** $\pm$ 0.11	0.53 $\pm$ 1.3
- Deep	0.33 $\pm$ 0.15	0.89 $\pm$ 1.27
- Shrinking	0.29 $\pm$ 0.12	**0.33** $\pm$ 1.3
Silence	0.74 $\pm$ 0.0	-2.05 $\pm$ 0.0
White noise	7.27 $\pm$ 0.07	-1.24 $\pm$ 1.93

Table 6.11: LSD $\times 10$ and CSIG scores of all network shape ablation setups for the Speech Commands dataset. For both metrics lower is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

Absolute fidelity evaluation

Considering table 6.12, we make the following observations regarding the absolute fidelity of reconstructed samples f_i for the different network shape parameterizations of ϕ_i & $p(\theta|\mathbf{z})$:

1. π-GAN Wide shows best absolute fidelity for NSYNTH Keyboard and Speech Commands[12], in line with its reconstructions' perceptual fidelity.
2. For the NSYNTH diverse dataset, all changes to the original network shape result in worse absolute fidelity, although differences with π-GAN Deep are small.

[12] This holds for all experiments with Speech Commands in this work

Architecture	NSYNTH Diverse		NSYNTH Keyboard		Speech Commands	
π-GAN	**1.32**	$\pm$ 0.3	4.48	$\pm$ 1.02	0.85	$\pm$ 0.01
- Wide	6.55	$\pm$ 0.16	**1.64**	$\pm$ 0.03	**0.51**	$\pm$ 0.01
- Deep	1.39	$\pm$ 0.12	6.19	$\pm$ 0.57	1.33	$\pm$ 0.32
- Shrinking	8.58	$\pm$ 1.7	5.57	$\pm$ 0.09	1.01	$\pm$ 0.08
Silence	46.19	$\pm$ 0.0	78.74	$\pm$ 0.0	7.48	$\pm$ 0.0
White noise	1061.49	$\pm$ 12.03	1094.74	$\pm$ 13.56	1023.18	$\pm$ 11.56

Table 6.12: Mean and standard deviation of MSE $\times 10^3$ statistics of batches (128 samples) of synthesized samples in the 5000th epoch of all models with best performing hyperparameters in network shape ablation setups. Silence and white noise ($\mu = 0, \sigma = 1$) errors reported for reference.

6.2.5 Qualitative waveform analysis

Considering figure 6.2 (and figure 10 and 9 in the Appendix), we make the following observations regarding the qualities of the reconstructed waveforms in the NSYNTH keyboard dataset of the different parameterizations of ϕ_i & $p(\theta|\mathbf{z})$:

- π-GAN Wide's reconstruction in general resembles the waveform best, consistent with results in table 6.4. Sine first, others ReLU reconstruction comes very close.
- Sine first, others ReLU reconstruction is most smooth, although not as smooth as waveGAN reconstructions shown in figure 6.1. The reconstruction of π-GAN Wide comes very close in terms of smoothness.
- The minimal mapping layers' reconstruction in general resembles the waveform slightly better than the π-GAN and five layer mapping network reconstructions, consistent with the results in table 6.7. See figure 10 in the Appendix.
- Ablations without FiLM conditioning show results close to silence, consistent with the results in table 6.4. See figure 9 in the Appendix.

6.2.6 Discussion

Interpreting results of all ablation experiments in the review above, the following key findings emerge regarding influential hyperparameters and the shortcomings of INRs:

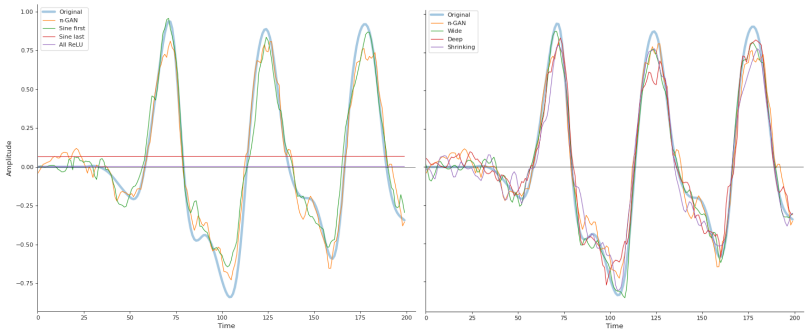


Figure 6.2: First 200 timepoints of a waveform validated to be representative as described in section 5.9.1 of the NSYNTH keyboard dataset and reconstructions of activation function (left) network shape (right) ablation setups.

△ **Fidelity is highly sensitive to expressivity of parameterizations.** The experimental results of WaveGAN, π-GAN Wide, and π-GAN Sine first confirm intuitions that less expressive architectures tend to require more uniformity in datasets for good fidelity in representations[13]. But if enough uniformity is present they perform better than more expressive architectures. This indicates that representation fidelity is highly sensitive to parameterization flexibility.

[13] To answer what types of diversity demand which amounts of expressiveness is hard to answer with just three datasets, but further discussed in section 6.3.3 statement 6.3.3

△ **Increased expressivity comes with increased fitting speed** Comparing the training curves of parameterizations of π-GAN parameterizations with different associated amounts of flexibility[14], we confirm intuitions that increased expressivity also comes with significantly increased fitting speed.[15]

[14] We compared training curves of parameterizations of π-GAN versus π-GAN Wide and π-GAN Sine first.
[15] In section A.1 we reproduce this behaviour in a simplified environment on a (unconditioned) SIREN with different ω_0's.

△ **Expressivity pressure in ω_0 search.** The activation input scaling parameters ω_0 have large effects on the expressivity of π-GAN; they strongly influence the balance between being able to represent fine detail and retaining smoothness in representations, see section A.1. Our hyperparameter selection procedure described in section 5.7 compares performance of ω_0's in a short training regime of 200 epochs to find good solutions in the sensitive search space of these parameters within computational limits. By doing so, we optimize ω_0's for early training speed. By statement 6.2.6, this creates a bias towards highly expressive ω_0's.

△ **ω_0's are a blessing and curse.** Fidelity of reconstructions is highly sensitive to expressivity, see statement 6.2.6. Thus, by statement 6.2.6, the hyperparameter selection procedure is likely to have had negative effects on representation quality in all our experiments in section 6.1 and 6.2. Thus, we argue that, with computational limitations, the combination of the large effects of ω_0's on the expressivity of π-GAN, more expressivity leading to faster training and the high sensitivy of fidelity to expressivity of parameterizations is a main shortcoming in the process required to learn good representations of distributions of audio waveforms with conditional INRs. Activation scaling is a key ingredient to the succes of SIRENs, but also creates difficult situations with limited resources: ω_0's are a blessing and curse.

- **Shallow latent mapping nets increase early training speed.** Optimal ω_0's for minimal latent network depth sit on the lower bound of all compared setups. We hypothesize that although a deeper latent mapping network is expected to be more expressive in the long run, in the beginning of training a randomly initialized deeper latent mapping network will slow down latent optimization due to noisier gradients. Thus, a shallower latent mapping net will facilitate more expressive behavior in a short training regime, leveling the playing field and allowing less expressive, in the longer term optimal, ω_0's to be selected. We think this is the main reason that the performance of minimal mapping setups outperforms others in NSYNTH Keyboard and Speech Commands.

- **Large ω_0 inconsistencies impede learning.** Found optimal first ω_0's for π-GAN Sine first in the short ω_0 sweeps are 260, 350 and 930 respectively for Speech Commands, NSYNTH keyboard and NSYNTH Diverse. This is significantly lower than average, see figure 3. We initially expected that optimal first ω_0's would be significantly higher to compensate for having no sine activations in later layers. Looking closer at optimal ω_0's in other parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ a trend about optimal ω_0's becomes clear. ω_0's in the first layer are roughly between 3-7 times the magnitude of ω_0's in later layers. Larger inconsistencies in magnitude seem to impede learning.[16] With the effective hidden ω_0's magnitude being small in π-GAN Sine first, it can be expected that optimal first ω_0's are lower than on average.

- **FiLM conditioning and first layer sine activations essential.** In line with theory, previous reported results[17], and the baseline experiments our experiments show that first layer sine activations are essential for representing any distribution of audio waveforms. However, we did not expect to FiLM conditioning to be as essential. In preliminary experiments on smaller datasets of up to 256 waveforms SIRENs conditioned via concatenation learned reasonable representations. These findings support the notion that the combination of FiLM conditioning with SIRENs is greater than the sum of its parts.

- **Explaining the success of less expressive π-GANs.** The ω_0 selection procedure induces suboptimal expressivity pressure, see statement 6.2.6. This is likely an important factor for the success of less expressive variants of π-GAN, as inherently less expressive architectures (e.g. π-GAN Wide and π-GAN Sine first) limit the maximum expressivity, minimizing the negative effects of expressivity pressure.

- **Explaining the success of π-GAN Wide.** Analyzing the network shape ablation results they align nicely with previous observations and neural network theory. π-GAN Wide outperforms all other ablations on Speech Commmands[18] and NSYNTH

[16] Wide and Shrinking network shapes are the most significant outliers to this rule, see section A.1.2 for a further analysis of trends in optimal ω_0's.

[17] Chan et al., *Pi-GAN*, 2020; Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

[18] π-GAN Wide shows best absolute fidelity for Speech Commmands across all experiments in this work.

Keyboard[19] and performs almost identical to the best setup in NSYNTH Diverse, the original formulation of π-GAN, except for absolute fidelity. We hypothesize that π-GAN Wide turned out to be the optimal setup for the following reasons:

1. Lower compositional depth of sine nonlinearities results in a less expressive architecture, as argued in 6.2.6
2. Wide networks tend to have more convex optimization landscapes than deeper networks[20], something that π-GAN should benefit from argued in statement 6.1.2.
3. FiLM conditioning parameters γ and β are shared over layers. Due to the extra width, FiLM controls more features and due to the decreased depth every FiLM parameter is shared between fewer features.

[19] π-GAN Wide shows best overall- and sample quality for NSYNTH Keyboard across all experiments in this work.

[20] Nguyen and Hein, "The Loss Surface of Deep and Wide Neural Networks", n.d.

6.3 Proposed extension experiments

In this section we aim to answer the following question:

- How can the main shortcomings in audio waveform representations learned by conditional INRs and the process required to learn these representations be addressed?

In section 6.1 we identify local waveform inconsistencies perceived as noisy components as the main shortcoming of audio waveform representations learned by conditional sinusoidal INRs. We argue this is caused by the high density of stationary points within these networks. In section 6.2 we identify the process of optimizing ω_0's in a short training regime to have amplifying effects on the amount of local waveform inconsistencies. We argue this happens because this process enforces a selection pressure towards more expressive ω_0's.

In section A.1 we validate these observations in a simplified and controlled environment, then prototype methods to cope with expressivity pressure post hoc and methods to circumvent having to optimize ω_0's altogether.

To answer if the shortcomings of audio waveform representations learned by conditional INRs can be adressed we test the methods validated in section A.1:

- **Weight regularization:** coping with expressivity pressure post hoc:
  - Train previously optimized ω_0 parameterizations with weight regularization.
  - Train previously optimized ω_0 parameterizations with linearly decreasing weight regularization.
- **Progressive activation scaling:** circumventing the ω_0 optimization process:

  - Train with progressively introducing nodes with higher activation scalings (ω_0's) in the first SIREN layer.

Observations. We summarize the most important observations regarding the questions we aim to answer in this section:

- For NSYNTH Keyboard and Speech Commands, applying constant weight regularization generally improves perceptual fidelity of reconstructions robust to the introduced hyperparameter λ. For optimal values of λ, reconstructions show the best, or close to the best results across all experiments reported in this work.
- Progressive weight regularization shows less robustness to λ and generally lower perceptual- and absolute fidelity scores.
- Progressive activation scaling shows very robust results with respect to hyperparameters, but is less robust with respect to dataset characteristics for perceptual fidelity. Overall- and sample quality in Speech Commands is improved for all ω_0 ranges that did not crash early due to NaN losses, resulting in the best scores across all experiments in this work, but these metrics underperform for other datasets. Absulute fidelity shows less gains, but is more robust to dataset characteristics.
- Qualitative waveform analyisis suggests improved smoothness and resemblence compared to WaveGAN for optimal weight regularization hyperparameters.

Conclusions. We summarize the key findings of our analysis of the experimental results reported in this section:

- Progressive activation scaling is an effective method for circumventing any hyperparameter tuning, but the method is not robust to a wide variety of dataset characteristics.
- Constant weight regularization proves to be an effective method for countering associated shortcomings of audio waveform representations learned by conditional INRs. It is more robust to dataset characteristics than progressive activation scaling, but less robust with respect to its introduced hyperparameter λ.
- Progressive weight regularization is an inferior method compared to constant weight regularization.
- NSynth Diverse does not benefit from any of the proposed methods, however it is unclear whether this is due to diversity in base frequencies and timbres, the higher presence of noisy components, lack of silence, or a combination of these.

6.3.1 Results: Weight regularization

We examine the effect of applying weight regularization to π-GAN after optimizing ω_0's in a short training regime. Weight regularization is applied for different values of λ as described in section

5.4, resulting in the objective function shown in equation A.1. See table 6.13 for the perceptual fidelity scores for NSYNTH datasets, and table 6.14 for the perceptual fidelity scores for Speech Commands. Absolute fidelity scores for all datasets are reported in table 6.15. Figure 6.3 shows the first 200 samples of a waveform from the NSYNTH keyboard dataset and reconstructions of the proposed weight regularization setups.

Perceptual fidelity evaluation

Considering tables 6.13 and table 6.14, we make the following observations regarding the perceptual fidelity of reconstructed samples f_i for the different weight regularization implementaions and values of λ:

- Results for constant and progressive weight regularization in Speech Commands are very robust, all tested values of λ consistently improve results, except for runs that crashed due to NaN losses.
- Results for constant weight regularization in NSYNTH keyboard are robust, a large range of values of λ perform better or similar in sample quality and background noise levels.
- For the NSYNTH Keyboard dataset, constant weight regularization with $\lambda = 10$ shows best background noise levels over all experiments in this work.
- For Speech Commands, constant weight regularization with $\lambda = 1$ shows best overall- and sample quality over all weight regularization setups.

	CDPAM: Overall quality		Multi STFT MSE: Noise		CSIG: Sample quality	
	Diverse	Keyboard	Diverse	Keyboard	Diverse	Keyboard
π-GAN	**0.35** $\pm$ 0.24	**0.48** $\pm$ 0.18	**0.06** $\pm$ 0.02	0.14 $\pm$ 0.04	**1.38** $\pm$ 1.84	-1.97 $\pm$ 2.45
Prog.^{50} W_ϕ reg.	nan $\pm$ nan	nan $\pm$ nan	nan $\pm$ nan	nan $\pm$ nan	nan $\pm$ nan	nan $\pm$ nan
Prog.^{30} W_ϕ reg.	nan $\pm$ nan	0.72 $\pm$ 0.08	nan $\pm$ nan	0.06 $\pm$ 0.02	nan $\pm$ nan	-2.92 $\pm$ 0.13
Prog.^{10} W_ϕ reg.	1.16 $\pm$ 0.03	0.76 $\pm$ 0.05	0.07 $\pm$ 0.0	0.04 $\pm$ 0.0	-4.04 $\pm$ 0.32	-2.33 $\pm$ 0.32
Const.^{30} W_ϕ reg.	1.11 $\pm$ 0.01	0.75 $\pm$ 0.01	0.08 $\pm$ 0.0	0.05 $\pm$ 0.0	-3.8 $\pm$ 0.51	-2.02 $\pm$ 1.69
Const.^{10} W_ϕ reg.	1.11 $\pm$ 0.05	0.71 $\pm$ 0.05	0.07 $\pm$ 0.0	**0.04** $\pm$ 0.0	-4.34 $\pm$ 0.27	**-1.04** $\pm$ 0.05
Const.^{1} W_ϕ reg.	1.18 $\pm$ 0.04	0.78 $\pm$ 0.04	0.08 $\pm$ 0.0	0.05 $\pm$ 0.01	-4.12 $\pm$ 0.2	-2.81 $\pm$ 0.24
Silence	1.65 $\pm$ 0.0	1.77 $\pm$ 0.0	0.16 $\pm$ 0.00	0.16 $\pm$ 0.00	-1.31 $\pm$ 0.0	-4.62 $\pm$ 0.0
White noise	1.06 $\pm$ 0.27	1.06 $\pm$ 0.32	2.27 $\pm$ 0.06	2.32 $\pm$ 0.04	-5.43 $\pm$ 3.14	-9.31 $\pm$ 2.59

Table 6.13: Evaluation metrics CDPAM, CSIG and multi resolution STFT MSE scores of proposed weight regularization setups for NSYNTH datasets. For CDPAM and multi resolution STFT MSE a lower score is better, for CSIG representing sample quality in NSYNTH datasets a higher score is better. Silence and white noise ($\mu = 0$, $\sigma = 1$) scores reported for reference.

Absolute fidelity evaluation

Considering table 6.15, we make the following observations regarding the absolute fidelity of reconstructed samples f_i for the different weight regularization implementaions and values of λ:

- For the NSYNTH Keyboard dataset, constant weight regularization shows improved absolute fidelity for all tested values of λ.

	LSD: Overall / Sample quality Speechcommands_	CSIG: Noise Speechcommands_
π-GAN	0.27 $\pm$ 0.12	0.59 $\pm$ 1.26
Prog.^{50} W_ϕ reg.	nan $\pm$ nan	nan $\pm$ nan
Prog.^{30} W_ϕ reg.	0.13 $\pm$ 0.0	-0.4 $\pm$ 0.0
Prog.^{10} W_ϕ reg.	0.13 $\pm$ 0.0	-0.26 $\pm$ 0.0
Const.^{30} W_ϕ reg.	0.1 $\pm$ 0.03	1.79 $\pm$ 0.28
Const.^{10} W_ϕ reg.	0.12 $\pm$ 0.01	**-0.58** $\pm$ 0.2
Const.^{1} W_ϕ reg.	**0.06** $\pm$ 0.0	0.18 $\pm$ 0.44
Silence	0.74 $\pm$ 0.0	-2.05 $\pm$ 0.0
White noise	7.27 $\pm$ 0.07	-1.24 $\pm$ 1.93

Table 6.14: LSD $\times 10$ and CSIG scores of all proposed weight regularization setups for the Speech Commands dataset. For both metrics lower is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

- For the NSYNTH Keyboard dataset, progressive weight regularization with a starting value of $\lambda = 10$ shows best absolute fidelity over all experiments in this work.
- Other datasets show lower absolute fidelity after applying any weight regularization method with any value of λ.

Architecture	NSYNTH Diverse		NSYNTH Keyboard		Speech Commands	
π-GAN	**1.32**	$\pm$ 0.3	4.48	$\pm$ 1.02	**0.85**	$\pm$ 0.01
Prog.^{50} W_ϕ reg.	nan	$\pm$ nan	nan	$\pm$ nan	nan	$\pm$ nan
Prog.^{30} W_ϕ reg.	nan	$\pm$ nan	9.21	$\pm$ 7.79	16.41	$\pm$ 0.0
Prog.^{10} W_ϕ reg.	4.78	$\pm$ 0.02	**0.76**	$\pm$ 0.09	16.41	$\pm$ 0.0
Const.^{30} W_ϕ reg.	10.52	$\pm$ 0.56	2.17	$\pm$ 0.52	11.19	$\pm$ 5.24
Const.^{10} W_ϕ reg.	4.53	$\pm$ 0.22	0.85	$\pm$ 0.1	14.86	$\pm$ 1.46
Const.^{1} W_ϕ reg.	4.27	$\pm$ 0.2	1.0	$\pm$ 0.29	3.71	$\pm$ 0.0
Silence	46.19	$\pm$ 0.0	78.74	$\pm$ 0.0	7.48	$\pm$ 0.0
White noise	1061.49	$\pm$ 12.03	1094.74	$\pm$ 13.56	1023.18	$\pm$ 11.56

Table 6.15: Mean and standard deviation of MSE $\times 10^3$ statistics of batches (128 samples) of synthesized samples in the 5000th epoch of all models in proposed weight regularization setups. Silence and white noise ($\mu = 0, \sigma = 1$) errors reported for reference.

6.3.2 Results: Progressive activation scaling

We examine the effect of the proposed method to the original formulation of π-GAN. Results specified as using original ω_0's used previously found optimal ω_0 in searches to simulate the effectiveness of the proposed method as a post hoc measure against the expressivity pressure observed in the short training regime. To validate the effectiveness and robustness of the proposed method as a replacement for optimizing ω_0's in a short training regime the other setups did not use previous ω_0 search results. See table 6.16 for the perceptual fidelity scores for NSYNTH datasets, and table 6.17 for the perceptual fidelity scores for Speech Commands. Absolute fidelity scores for all datasets are reported in table 6.18. Figure 11 shows the first 200 samples of a waveform from the NSYNTH keyboard dataset and reconstructions of the proposed progressive activation scaling setups.

Perceptual fidelity evaluation

Considering tables 6.16 and table 6.17, we make the following observations regarding the perceptual fidelity of reconstructed samples f_i for the different progressive activation scaling ranges:

- For Speech Commands, overall- and sample quality is significantly improved for all ω_0 ranges that did not crash early due to NaN losses.
- Background noise levels for NSYNTH keyboard are improved when progressively introducing activation scalings for all ω_0 ranges that did not crash early due to NaN losses.
- Overall- and sample quality is significantly worse when progressively introducing activation scalings in NSYNTH Keyboard and NSYNTH Diverse datasets.
- For Speech Commands, progressively introducing activation scalings up to 10.000, with hidden activation scalings of 300 shows best overall- and sample quality of across all experiments in this work.

	CDPAM: Overall quality		Multi STFT MSE: Noise		CSIG: Sample quality	
	Diverse	Keyboard	Diverse	Keyboard	Diverse	Keyboard
π-GAN	**0.35** $\pm$ 0.24	**0.48** $\pm$ 0.18	**0.06** $\pm$ 0.02	0.14 $\pm$ 0.04	**1.38** $\pm$ 1.84	**-1.97** $\pm$ 2.45
Prog. orig. ω_0	1.16 $\pm$ 0.01	0.87 $\pm$ 0.01	0.08 $\pm$ 0.0	0.11 $\pm$ 0.01	-1.6 $\pm$ 0.3	-3.07 $\pm$ 0.34
Prog.$^{3000}_{30}$ ω_0	nan $\pm$ nan	nan $\pm$ nan	nan $\pm$ nan	nan $\pm$ nan	nan $\pm$ nan	nan $\pm$ nan
Prog.$^{3000}_{300}$ ω_0	1.19 $\pm$ 0.03	nan $\pm$ nan	0.1 $\pm$ 0.01	nan $\pm$ nan	-3.38 $\pm$ 0.83	nan $\pm$ nan
Prog.$^{3000}_{1000}$ ω_0	1.18 $\pm$ 0.03	0.82 $\pm$ 0.01	0.09 $\pm$ 0.0	0.1 $\pm$ 0.02	-2.19 $\pm$ 0.52	-2.46 $\pm$ 0.32
Prog.$^{10000}_{30}$ ω_0	1.21 $\pm$ 0.0	0.84 $\pm$ 0.01	0.08 $\pm$ 0.0	**0.07** $\pm$ 0.01	-2.39 $\pm$ 0.13	-2.73 $\pm$ 0.39
Prog.$^{10000}_{300}$ ω_0	1.25 $\pm$ 0.02	0.89 $\pm$ 0.02	0.11 $\pm$ 0.0	0.11 $\pm$ 0.02	-5.28 $\pm$ 0.18	-5.27 $\pm$ 0.33
Silence	1.65 $\pm$ 0.0	1.77 $\pm$ 0.0	0.16 $\pm$ 0.00	0.16 $\pm$ 0.00	-1.31 $\pm$ 0.0	-4.62 $\pm$ 0.0
White noise	1.06 $\pm$ 0.27	1.06 $\pm$ 0.32	2.27 $\pm$ 0.06	2.32 $\pm$ 0.04	-5.43 $\pm$ 3.14	-9.31 $\pm$ 2.59

Table 6.16: Evaluation metrics CDPAM, CSIG and multi resolution STFT MSE scores of different progressive activation scaling setups for NSYNTH datasets. For CDPAM and multi resolution STFT MSE a lower score is better, for CSIG representing sample quality in NSYNTH datasets a higher score is better. Silence and white noise ($\mu = 0$, $\sigma = 1$) scores reported for reference.

	LSD: Overall / Sample quality	CSIG: Noise
	Speechcommands_	Speechcommands_
π-GAN	0.27 $\pm$ 0.12	0.59 $\pm$ 1.26
Prog. orig. ω_0	0.06 $\pm$ 0.0	0.76 $\pm$ 0.06
Prog.$^{3000}_{30}$ ω_0	nan $\pm$ nan	nan $\pm$ nan
Prog.$^{3000}_{300}$ ω_0	nan $\pm$ nan	nan $\pm$ nan
Prog.$^{3000}_{1000}$ ω_0	0.05 $\pm$ 0.0	0.82 $\pm$ 0.05
Prog.$^{10000}_{30}$ ω_0	0.05 $\pm$ 0.0	**-0.21** $\pm$ 0.03
Prog.$^{10000}_{300}$ ω_0	**0.04** $\pm$ 0.0	0.88 $\pm$ 0.03
Silence	0.74 $\pm$ 0.0	-2.05 $\pm$ 0.0
White noise	7.27 $\pm$ 0.07	-1.24 $\pm$ 1.93

Table 6.17: LSD $\times$10 and CSIG scores of all progressive activation scaling setups for the Speech Commands dataset. For both metrics lower is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

Absolute fidelity evaluation

Considering table 6.18, we make the following observations regarding the absolute fidelity of reconstructed samples f_i for the different parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ and different progressive activation scaling ranges:

1. For the NSYNTH Keyboard dataset, progressively introducing activation scalings up to 10.000 shows improved absolute fidelity compared to the original π-GAN scores, robust to a wide range of hidden $omega_0$ values.
2. Other datasets show lower absolute fidelity when applying progressive activation scaling in any range.

Architecture	NSYNTH Diverse		NSYNTH Keyboard		Speech Commands	
π-GAN	**1.32**	$\pm$ 0.3	4.48	$\pm$ 1.02	**0.85**	$\pm$ 0.01
Prog. orig. ω_0	4.58	$\pm$ 0.44	6.84	$\pm$ 2.15	3.62	$\pm$ 0.3
Prog.$^{3000}_{30}$ ω_0	nan	$\pm$ nan	nan	$\pm$ nan	nan	$\pm$ nan
Prog.$^{3000}_{300}$ ω_0	4.73	$\pm$ 0.3	nan	$\pm$ nan	nan	$\pm$ nan
Prog.$^{3000}_{1000}$ ω_0	4.99	$\pm$ 0.03	6.21	$\pm$ 3.25	2.43	$\pm$ 0.24
Prog.$^{10000}_{30}$ ω_0	4.19	$\pm$ 0.15	**1.71**	$\pm$ 0.31	2.66	$\pm$ 0.16
Prog.$^{10000}_{300}$ ω_0	5.5	$\pm$ 0.04	3.62	$\pm$ 1.11	1.46	$\pm$ 0.11
Silence	46.19	$\pm$ 0.0	78.74	$\pm$ 0.0	7.48	$\pm$ 0.0
White noise	1061.49	$\pm$ 12.03	1094.74	$\pm$ 13.56	1023.18	$\pm$ 11.56

Table 6.18: Mean and standard deviation of MSE $\times 10^3$ statistics of batches (128 samples) of synthesized samples in the 5000th epoch of all models with best performing hyperparameters in proposed progressive latent scaling setups. Silence and white noise ($\mu = 0$, $\sigma = 1$) errors reported for reference.

Qualitative waveform analysis

Considering figure 6.3 (and figure 11 in the appendix), we make the following observations regarding the qualities of the reconstructed waveforms f_i for the different weight regularization implementations with different values of λ and progressively introducing activation scalings in different ranges:

- Setups with a small amount of weight regularization resemble the waveform very well, the best reconstructions resemble the waveform closer than WaveGANs reconstruction.
- It is clearly visible that larger values of λ make the decoder less expressive.
- All waveforms are significantly smoother than those in previous results.
- Waveforms reconstructed with a higher range of progressive activation scalings still show chaotic behaviour, while those reconstructed with a lower range of progressive activation scalings are not expressive enough.

6.3.3 Discussion

Interpreting results of all proposed extension experiments in the review above, the following key findings emerge regarding the effectiveness of the tested methods to counter shortcomings of audio waveform representations learned by conditional INRs:

- **Expressiveness demanding factors in datasets.** Across all experiments in this work for NSYNTH Diverse, none improved upon absolute- or perceptual fidelity of the original formulation of π-GAN, neither did the methods tested in this section. Most of the experiments suppressed expressiveness in some way, except for π-gan Deep and π-gan with five latent mapping layers[21], which perform as good as π-GAN. This shows that it demands most expressiveness of all three datasets, however it is unclear whether this is due to diversity in base frequencies and timbres, the higher presence of noisy components, lack of silence, or a combination

[21] We argue more latent mapping layers mostly increases expressivity combined with short training ω_0 optimization, in which it causes extra expressivity by slowing early training progress, making the search process find higher optimal values for ω_0.

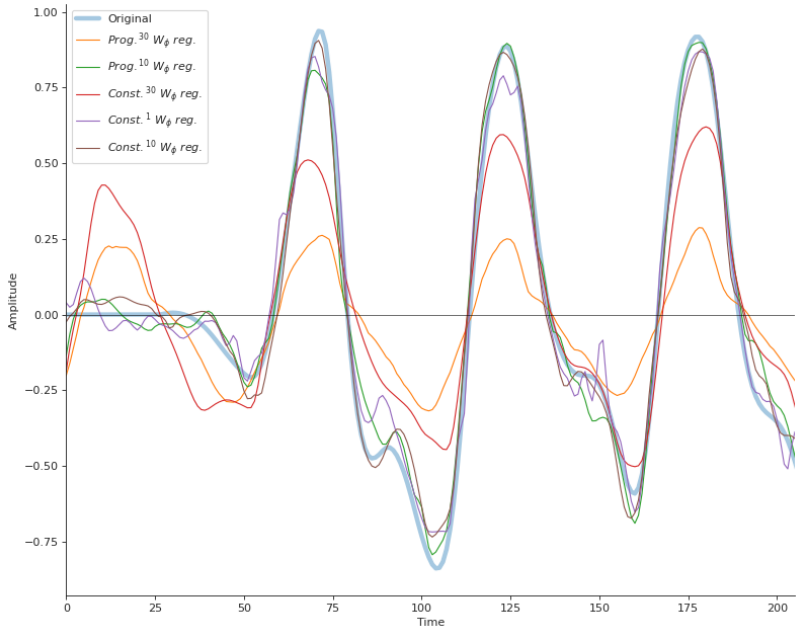


Figure 6.3: First 200 timepoints of a waveform validated to be representative as described in section 5.9.1 of the NSYNTH keyboard dataset and reconstructions of weight regularization setups.

of these. To get a more clear view of this, experiments with more fine-grained differences between datasets are required.[22]

[22] The differences between NSYNTH Diverse and NSYNTH Keyboard (demanding the least amount of expressivity) must be explained by the diversity in base frequencies and timbres and the presence of noisy components, while the differences between NSYNTH keyboard and Speech commands must be explained by heterogeneity in onset times, differences in the amount of silence and spectral incoherence.

- **Constant weight regularization** proves to be a good method to cope with the expressivity of conditional INRs with short training optimized ω_0's. The introduced hyperparameter λ is quite robust, showing similar results for large ranges in all tested datasets. For NSYNTH Keyboard and Speech Commands, reconstructions generally improve in terms of perceptual fidelity and show the best, or close to the best results for optimal values of λ across all experiments reported in this work. For NSYNTH keyboard this also holds for absolute fidelity.

- **Progressive weight regularization** compared to constant weight regularization shows results with less robustness to λ and generally lower perceptual- and absolute fidelity scores making the method inferior to constant weight regularization. However, in NSYNTH Keyboard for optimal values of λ it shows the highest perceptual fidelity across all experiments reported in this work. The inferior robustness to values of λ compared to constant weight regularization contrasts with the results of experiments with SIRENs reconstructing single waveforms in section A.1.3. We hypothesize this contrast is caused by the relatively more complex loss landscape of conditional INRs and the fact that we did not run experiments for different amounts of epochs in which λ is decreases to zero. We did not do this because if it would be necessary, the introduced hyperparameter complexity of this method would defeat its own purpose.

- **Progressive activation scaling** shows strong and robust results

with respect to hyperparameters. For perceptual fidelity of Speech Commands, overall- and sample quality is significantly improved for all ω_0 ranges that did not crash early due to NaN losses, resulting in the best scores across all experiments in this work. However, the methods perceptual fidelity scores are less robust to dataset characteristics, as both NSYNTH datasets generally show lower perceptual fidelity. Absolute fidelity is slightly higher for NSYTNH Keyboard and slightly lower for Speech Commands, compared to the optimized ω_0 π-GAN setups. Note that we did not experiment with different amounts of epochs in which the ω_0's are introduced, indicating a high likelihood of robustness with respect to this parameter as well. Progressive activation scaling is robust with respect to its hyperparameters, thus an effective method for circumventing any hyperparameter tuning, but it is not robust to a wide variety of dataset characteristics.

- **Oscillatory bias in less expressive sinusoidal INRs.** Figure 6.3 shows how sinusoidal INRs with larger amounts of weight regularization have trouble inhibiting periodic behaviour in the base frequency, indicating a strong oscillatory bias. This can be explained by the fact that nested sines with no activation scaling behave very similar to regular sines. With large weight regularization the activation scaling is minimized by reduced network parameters and the network starts to exhibit behaviour similar to having a sparse fourier basis in the range of base frequencies in the signal. As shown in the inhibition difficulties in figure 6.3, this results in a strong oscillitory bias that is detrimental to reconstruction performance in waveforms with dynamic spectral content, because the global support of sinusoids requires the network to supress any non-static spectral content. Nested sinusoids with larger amounts of input scaling quickly introduce more and higher frequencies. The input scalings optimal in our experiments result in functions biased for high amounts of extremely high frequency functions, because this provides more flexibility for suppressing arbitrary function regions essential for representing non-static spectral content.

7 Conclusion and future work

In this work we explored the potential of applying implicit neural representations in generative modelling of audio waveforms, abandoning the classical way to represent audio as discretized vectors and instead parameterize individual data points by continuous functions. Implicit neural representations are an emerging paradigm, at the time of writing published applications in the generative domain are scarce, and to our knowledge no research has been done applying it in the field of audio synthesis.

To this extent, we studied the effects of various parameterizations of implicit architectures, conditioning mechanisms and methods for inferring latent embeddings on several (perceptual) quality metrics of learned representations. We compared results across three datasets, two containing musical notes and one containing speech, relating dataset characteristics to reconstruction performance. We proposed and validated two methods to cope with deficiencies observed in our experiments.

Predominantly focussing on sinusoidal representation networks conditioned using feature wise linear modulation, we discussed both the power and inherent limitations of this implicit generative architecture compared to convolutional alternatives, touching upon topics such as expressivity, label noise robustness and encoder compatibility.

7.1 Conclusion

We conclude that conditional INRs show great potential for representing distributions of audio waveforms with perceptual- and absolute fidelity. To foster reproducible research, we published the source code of this research on GitHub.[1] We summarize the key findings of this work:

[1] https://github.com/janzuiderveld/continuous-audio-representations

- Convolutional encoders significantly impede learning for sinusoidal INR's conditioned using feature wise linear modulation (FiLM) due to incompatibility between the architectures. By learning representations in an autodecoder setup there are no introduced bottlenecks or biases external to the decoder. We argue

this is optimal for exploring the representation characteristics of decoder networks.

◊ Sinusoidal INR's conditioned using FiLM (π-GAN) exhibit exceptional expressivity, making them suited for modelling distributions of high-frequency one-dimensional continuous functions such as audio. This INR parameterization outperforms convolutional architectures with equal parameter counts in absolute fidelity of learned representations in all tested datasets. But, the perceptual fidelity of representations is inferior to those of transposed convolution based architectures in more uniform datasets as it introduces local waveform inconsistencies in reconstructions.

□ INR parameterizations without sine activations in the first layer and/or with conditioning only by concatenation are ill-suited for representing a distribution of audio waveforms.

• Sinusoidal INR's conditioned using FiLM have strict requirements for local consistency in loss landscapes. Based on our experimental results and neural network theory we derive this is caused by noisy signal propagation within these networks. We argue that the periodicity of sine activations (resulting in recurrent non-monotonic regions) is the origin of this deficiency, and that large activation scaling values (ω_0's) and high compositional depth have an amplifying effect on it.

□ Our experiments show that exactly these three factors (sine activations, ω_0's and compositional depth) are most influential in the expressivity, smoothness and final performance of INR's conditioned using FiLM.

□ When fine-tuning the considered model hyperparameters of sinusoidal INR's conditioned using FiLM, they outperform the baseline convolutional decoder in selected perceptual fidelity metrics across all tested datasets. We develop several heuristics for determining model hyperparameters according to dataset characteristics.

△ We show that representation fidelity in sinusoidal INR's is very sensitive to ω_0 values. When dealing with limited computing resources it is infeasible to optimize ω_0's over full training runs. Optimizing ω_0's in a short training regime can be problematic as it induces pressure on expressivity, which can have negative effects on perceptual fidelity of representations. To this extent we make the following contributions:

1. We propose and validate post hoc methods for taming the expressivity of activation scaling hyperparameters found optimal in short training runs, with success in two out of three tested datasets.
2. We propose and validate a method for removing the need to optimize activation scaling hyperparameters altogether. One

out of three datasets shows robust and significant perceptual fidelity gains.

We will conclude this work with a brief description of ideas for future work.

7.2 Future work

Finally, we propose several ideas for future work in the domain of implicit audio synthesis and explain our intuitions:

- Considering the noise introduced by the high density of stationary points in sinusoidal representation network layers with high ω_0's (see section 6.1.2) resulting in noisy representations and gradients, and possibly reinforcing sensitivity to label noise, we propose to apply gradient clipping, as this could allow more stable learning and higher learning rates.

- Taking the inferior label noise robustness observed in sinusoidal representation networks conditioned using feature wise linear modulation into account, methods for dealing with noisy loss signals would be another promising avenue. This would allow the usage of spectral and perceptual reconstruction losses, which correlate much better with perceptual qualities than MSE. Combined with the expressivity of sinusoidal representation networks conditioned using feature wise linear modulation this could greatly improve perceptual qualities of represented distributions. One might hope that gradient clipping can also aid in mitigating the detrimental effects of locally inconsistent loss functions on learning. However, it has been proven that in classification problems standard gradient clipping does not *in general* provide robustness to label noise[2].

- The compositional nature of SIRENs makes it difficult to analyze representation characteristics in a signal processing context. Multiplicative filter networks[3] (MFNs) can be viewed as linear function approximators over an exponential number of Fourier or Gabor basis functions. This establishes a connection of the network architecture with the traditional Fourier and Gabor wavelet transforms, which are extensively studied in literature and widely used in many application domains, especially audio.

  Comparisons with MFNs would be useful to indicate if the compositional depth of SIRENs form a substantial benefit. In the experiments reported by the authors, MFNs show similar performance as SIRENs in parameter equal experiments, and larger gains in performance when increasing network depth and width. We indeed found SIRENs to lack in terms of scaling gains. Combined with the intuition that Fourier-based MFNs contain a strong oscillatory bias, the architecture is a promising avenue for future

[2] Menon et al., "CAN GRADIENT CLIPPING MITIGATE LABEL NOISE?", 2020.

[3] Fathony et al., "MULTIPLICATIVE FILTER NETWORKS", 2021.

research in the area of implicit audio synthesis. The authors report however that SIRENs are more biased towards smoother regions in the represented function, where in our experiments, the lack of smoothness in representations of waveforms is arguably the foremost downside of SIRENs.

- Another idea which looks promising in the light of our results, still unexplored in current INR literature, is that of splitting the classically fully-connected MLP parametrizing INR's in multiple parallel subnetworks. This could reduce the propagation of noise induced at stationary points and cancel noise in output by aggregating multiple independent noise sources. This argumentation is supported by multiple research directions.

  Assuming similar principles are shared between channels in convolutional neural networks and subnetworks in INRs, several publications[4] show that increasing the amount of kernels in Resnets has profound effects on reducing chaotic behaviour of loss landscapes.

  Research in global optimality conditions of Haeffele & Vidal[5] concludes the following: *"A final implication of our analysis is that neural networks which generate the output by taking the sum of multiple parallel subnetworks are highly conducive to efficient optimization."*[6]

  The concept of combining the outputs of parallel subnetworks also has clear analogies to the field of ensemble methods, which could provide further theoretical foundations relating to this idea. For example, recent work of Fort et al[7] shows how different randomly initialized CNN's trained for classification problems tend to explore diverse modes in function space, resulting in improved robustness. We found that averaging waveform reproductions of independently trained conditional INR's worked well in mitigating noisy components, indicating that similar principles might apply.

[4] H. Li et al., *Visualizing the Loss Landscape of Neural Nets*, 2018; Hongyang Zhang et al., *Deep Neural Networks with Multi-Branch Architectures Are Less Non-Convex*, 2018.

[5] Haeffele and Vidal, "Global Optimality in Neural Network Training", 2017.

[6] The provided minimally sufficient conditions to guarantee that local minima are globally optimal in this paper include that the output of every parallel subnetwork is a positively homogeneous function of the network parameters, which does not hold when applying periodic activations. However, the authors indicate that any networks' optimization landscape would benefit from consisting of a linear combination of multiple parallel subnetworks.

[7] Fort et al., *Deep Ensembles*, 2020.

Bibliography

Anokhin, Ivan, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhenkov (2020). *Image Generators with Conditionally-Independent Pixel Synthesis*. arXiv: 2011.13775 [cs]. URL: http://arxiv.org/abs/2011.13775 (visited on 02/17/2021) (cit. on p. 21).

Bond-Taylor, Sam and Chris G. Willcocks (2020). *Gradient Origin Networks*. arXiv: 2007.02798 [cs]. URL: http://arxiv.org/abs/2007.02798 (visited on 11/16/2020) (cit. on p. 27).

Chan, Eric R., Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein (2020). *Pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis*. arXiv: 2012.00926 [cs]. URL: http://arxiv.org/abs/2012.00926 (visited on 02/16/2021) (cit. on pp. 9, 17, 21, 28, 33, 38, 65).

Chang, Angel X., Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. (2015). *ShapeNet: An Information-Rich 3D Model Repository*. arXiv: 1512.03012 [cs]. URL: http://arxiv.org/abs/1512.03012 (visited on 06/26/2021) (cit. on p. 19).

Chapelle, Olivier, Bernhard Schölkopf, and Alexander Zien, eds. (2006). *Semi-Supervised Learning*. Adaptive Computation and Machine Learning. Cambridge, Mass: MIT Press. 508 pp. (cit. on p. 56).

Chen, Zhiqin and Hao Zhang (2019). *Learning Implicit Fields for Generative Shape Modeling*. arXiv: 1812.02822 [cs]. URL: http://arxiv.org/abs/1812.02822 (visited on 03/17/2021) (cit. on pp. 9, 19, 28, 32, 38, 60).

Choy, Christopher B., Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese (2016). *3D-R2N2: A Unified Approach for Single and Multi-View 3D Object Reconstruction*. arXiv: 1604.00449 [cs]. URL: http://arxiv.org/abs/1604.00449 (visited on 06/26/2021) (cit. on p. 19).

Dai, Angela, Charles Ruizhongtai Qi, and Matthias Nießner (2017). *Shape Completion Using 3D-Encoder-Predictor CNNs and Shape Synthesis*. arXiv: 1612.00101 [cs]. URL: http://arxiv.org/abs/1612.00101 (visited on 06/26/2021) (cit. on p. 20).

Défossez, Alexandre, Neil Zeghidour, Nicolas Usunier, Léon Bottou, and Francis Bach (2018). *SING: Symbol-to-Instrument Neural Generator*. arXiv: 1810.09785 [cs, eess, stat]. URL: http://arxiv.org/abs/1810.09785 (visited on 03/25/2021) (cit. on pp. 15, 22, 56).

Deng, Li (2012). "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]". In: *IEEE Signal Processing Magazine* 29.6, pp. 141–142 (cit. on p. 20).

De Vries, Harm, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron Courville (2017). *Modulating Early Visual Processing by Language*. arXiv: 1707.00683 [cs]. URL: http://arxiv.org/abs/1707.00683 (visited on 03/25/2021) (cit. on p. 20).

Donahue, Chris, Julian McAuley, and Miller Puckette (2019). *Adversarial Audio Synthesis*. arXiv: 1802.04208 [cs]. URL: http://arxiv.org/abs/1802.04208 (visited on 11/18/2020) (cit. on pp. 14, 15, 23, 38, 41).

Dupont, Emilien, Yee Whye Teh, and Arnaud Doucet (2021). *Generative Models as Distributions of Functions*. arXiv: 2102.04776 [cs, stat]. URL:

http://arxiv.org/abs/2102.04776 (visited on 02/12/2021) (cit. on pp. 9, 27).

Engel, Jesse, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts (2019). *GANSynth: Adversarial Neural Audio Synthesis*. arXiv: 1902.08710 [cs, eess, stat]. URL: http://arxiv.org/abs/1902.08710 (visited on 11/10/2020) (cit. on pp. 15, 41).

Engel, Jesse, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi (2017). *Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders*. arXiv: 1704.01279 [cs]. URL: http://arxiv.org/abs/1704.01279 (visited on 03/25/2021) (cit. on pp. 22, 41).

Fathony, Rizal, Devin Willmott, Anit Kumar Sahu, and J Zico Kolter (2021). "MULTIPLICATIVE FILTER NETWORKS". In: p. 11 (cit. on p. 77).

Feng, Ruili, Deli Zhao, and Zhengjun Zha (2021). *On Noise Injection in Generative Adversarial Networks*. arXiv: 2006.05891 [cs, stat]. URL: http://arxiv.org/abs/2006.05891 (visited on 07/02/2021) (cit. on p. 57).

Fort, Stanislav, Huiyi Hu, and Balaji Lakshminarayanan (2020). *Deep Ensembles: A Loss Landscape Perspective*. arXiv: 1912.02757 [cs, stat]. URL: http://arxiv.org/abs/1912.02757 (visited on 06/24/2021) (cit. on p. 78).

Ghiasi, Golnaz, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens (2017). *Exploring the Structure of a Real-Time, Arbitrary Neural Artistic Stylization Network*. arXiv: 1705.06830 [cs]. URL: http://arxiv.org/abs/1705.06830 (visited on 03/26/2021) (cit. on p. 30).

Goodfellow, Ian J., Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). *Generative Adversarial Networks*. arXiv: 1406.2661 [cs, stat]. URL: http://arxiv.org/abs/1406.2661 (visited on 03/26/2021) (cit. on p. 20).

Gray, A. and J. Markel (1976). "Distance Measures for Speech Processing". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 24.5, pp. 380–391 (cit. on p. 45).

Griffin, D. and Jae Lim (1984). "Signal Estimation from Modified Short-Time Fourier Transform". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 32.2, pp. 236–243 (cit. on p. 23).

Ha, David, Andrew Dai, and Quoc V. Le (2016). *HyperNetworks*. arXiv: 1609.09106 [cs]. URL: http://arxiv.org/abs/1609.09106 (visited on 03/26/2021) (cit. on pp. 21, 28).

Haeffele, Benjamin D. and Rene Vidal (2017). "Global Optimality in Neural Network Training". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI: IEEE, pp. 4390–4398 (cit. on p. 78).

Häne, Christian, Shubham Tulsiani, and Jitendra Malik (2017). *Hierarchical Surface Prediction for 3D Object Reconstruction*. arXiv: 1704.00710 [cs]. URL: http://arxiv.org/abs/1704.00710 (visited on 06/26/2021) (cit. on p. 19).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). *Identity Mappings in Deep Residual Networks*. arXiv: 1603.05027 [cs]. URL: http://arxiv.org/abs/1603.05027 (visited on 06/26/2021) (cit. on p. 20).

Hertz, Amir, Or Perel, Raja Giryes, Olga Sorkine-Hornung, and Daniel Cohen-Or (2021). *SAPE: Spatially-Adaptive Progressive Encoding for Neural Optimization*. arXiv: 2104.09125 [cs]. URL: http://arxiv.org/abs/2104.09125 (visited on 07/05/2021) (cit. on pp. 40, 92).

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long Short-Term Memory". In: *Neural computation* 9, pp. 1735–80 (cit. on p. 23).

Hornik, Kurt, Maxwell Stinchcombe, and Halbert White (1989). "Multilayer Feedforward Networks Are Universal Approximators". In: *Neural Networks* 2.5, pp. 359–366 (cit. on p. 13).

Hu, Yi and Philipos Loizou (2008). "Evaluation of Objective Quality Measures for Speech Enhancement". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 16, pp. 229–238 (cit. on p. 45).

Huang, Xun and Serge Belongie (2017). *Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization*. arXiv: 1703.06868 [cs]. URL: http://arxiv.org/abs/1703.06868 (visited on 03/26/2021) (cit. on p. 30).

Ioffe, Sergey and Christian Szegedy (2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. arXiv: 1502.03167 [cs]. URL: http://arxiv.org/abs/1502.03167 (visited on 07/04/2021) (cit. on p. 30).

Karras, Tero, Samuli Laine, and Timo Aila (2019). *A Style-Based Generator Architecture for Generative Adversarial Networks*. arXiv: 1812.04948 [cs, stat]. URL: http://arxiv.org/abs/1812.04948 (visited on 11/02/2020) (cit. on p. 57).

Karras, Tero, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila (2020). "Analyzing and Improving the Image Quality of StyleGAN". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA: IEEE, pp. 8107–8116 (cit. on p. 21).

Kilgour, Kevin, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi (2019). *Fr\'echet Audio Distance: A Metric for Evaluating Music Enhancement Algorithms*. arXiv: 1812.08466 [cs, eess]. URL: http://arxiv.org/abs/1812.08466 (visited on 03/26/2021) (cit. on p. 45).

Kim, Taesup, Inchul Song, and Yoshua Bengio (2017). *Dynamic Layer Normalization for Adaptive Neural Acoustic Modeling in Speech Recognition*. arXiv: 1707.06065 [cs]. URL: http://arxiv.org/abs/1707.06065 (visited on 03/26/2021) (cit. on p. 30).

Kingma, Diederik P. and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs]. URL: http://arxiv.org/abs/1412.6980 (visited on 03/21/2021) (cit. on p. 44).

Kingma, Diederik P. and Max Welling (2014). *Auto-Encoding Variational Bayes*. arXiv: 1312.6114 [cs, stat]. URL: http://arxiv.org/abs/1312.6114 (visited on 03/17/2021) (cit. on p. 19).

LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel (1989). "Backpropagation Applied to Handwritten Zip Code Recognition". In: *Neural Computation* 1.4, pp. 541–551 (cit. on p. 13).

Li, Hao, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein (2018). *Visualizing the Loss Landscape of Neural Nets*. arXiv: 1712.09913 [cs, stat]. URL: http://arxiv.org/abs/1712.09913 (visited on 06/29/2021) (cit. on p. 78).

Li, Lisha, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar (2018). *Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization*. arXiv: 1603.06560 [cs, stat]. URL: http://arxiv.org/abs/1603.06560 (visited on 01/31/2021) (cit. on pp. 44, 94).

Liu, Celong, Zhong Li, Junsong Yuan, and Yi Xu (2021). *NeLF: Practical Novel View Synthesis with Neural Light Field*. arXiv: 2105.07112 [cs]. URL: http://arxiv.org/abs/2105.07112 (visited on 06/23/2021) (cit. on p. 9).

Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang (2015). *Deep Learning Face Attributes in the Wild*. arXiv: 1411.7766 [cs]. URL: http://arxiv.org/abs/1411.7766 (visited on 06/27/2021) (cit. on p. 21).

Manocha, Pranay, Zeyu Jin, Richard Zhang, and Adam Finkelstein (2021). *CDPAM: Contrastive Learning for Perceptual Audio Similarity*. arXiv: 2102.05109 [cs, eess]. URL: http://arxiv.org/abs/2102.05109 (visited on 04/16/2021) (cit. on p. 45).

Menon, Aditya Krishna, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar (2020). "CAN GRADIENT CLIPPING MITIGATE LABEL NOISE?" In: p. 26 (cit. on p. 77).

Mescheder, Lars, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger (2019). *Occupancy Networks: Learning 3D Reconstruction in Function Space*. arXiv: 1812.03828 [cs]. URL: http://arxiv.org/abs/1812.03828 (visited on 03/24/2021) (cit. on pp. 9, 19, 28, 30).

Mildenhall, Ben, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng (2020). *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*. arXiv: 2003.08934 [cs]. URL: http://arxiv.org/abs/2003.08934 (visited on 03/24/2021) (cit. on pp. 9, 16, 20).

Moerel, Michelle, Federico De Martino, and Elia Formisano (2012). "Processing of Natural Sounds in Human Auditory Cortex: Tonotopy, Spectral Tuning, and Relation to Voice Sensitivity". In: *The Journal of Neuroscience* 32.41, pp. 14205–14216. pmid: 23055490 (cit. on p. 14).

Nguyen, Quynh and Matthias Hein (n.d.). "The Loss Surface of Deep and Wide Neural Networks". In: (), p. 10 (cit. on p. 66).

Oord, Aaron van den, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu (2016). *WaveNet: A Generative Model for Raw Audio*. arXiv: 1609.03499 [cs]. URL: http://arxiv.org/abs/1609.03499 (visited on 03/25/2021) (cit. on pp. 14, 22).

Oord, Aaron van den, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu (2016). *Conditional Image Generation with PixelCNN Decoders*. arXiv: 1606.05328 [cs]. URL: http://arxiv.org/abs/1606.05328 (visited on 03/25/2021) (cit. on p. 22).

Park, Jeong Joon, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove (2019). *DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation*. arXiv: 1901.05103 [cs]. URL: http://arxiv.org/abs/1901.05103 (visited on 02/05/2021) (cit. on pp. 9, 19, 26–28, 39, 60).

Perez, Ethan, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville (2017). *FiLM: Visual Reasoning with a General Conditioning Layer*. arXiv: 1709.07871 [cs, stat]. URL: http://arxiv.org/abs/1709.07871 (visited on 02/16/2021) (cit. on pp. 21, 28, 29).

Qi, Charles R., Hao Su, Kaichun Mo, and Leonidas J. Guibas (2017). *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*. arXiv: 1612.00593 [cs]. URL: http://arxiv.org/abs/1612.00593 (visited on 06/26/2021) (cit. on p. 19).

Radford, Alec, Luke Metz, and Soumith Chintala (2016). *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*. arXiv: 1511.06434 [cs]. URL: http://arxiv.org/abs/1511.06434 (visited on 03/26/2021) (cit. on pp. 23, 34).

Rahaman, Nasim, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville (2019). "On the Spectral Bias of Neural Networks". In: *International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, pp. 5301–5310 (cit. on p. 86).

Rahaman, Nasim, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A Hamprecht, Yoshua Bengio, and Aaron Courville (n.d.). "On the Spectral Bias of Neural Networks". In: (), p. 10 (cit. on p. 20).

Rix, A.W., J.G. Beerends, M.P. Hollier, and A.P. Hekstra (2001). "Perceptual Evaluation of Speech Quality (PESQ)-a New Method for Speech Quality Assessment of Telephone Networks and Codecs". In: *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*. 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221). Vol. 2, 749–752 vol.2 (cit. on p. 45).

Rumelhart, David E., James Mcclelland, and James L. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1. Foundations* (cit. on pp. 26, 39).

Schneider, Steffen, Alexei Baevski, Ronan Collobert, and Michael Auli (2019). *Wav2vec: Unsupervised Pre-Training for Speech Recognition*. arXiv: 1904.05862 [cs]. URL: http://arxiv.org/abs/1904.05862 (visited on 07/05/2021) (cit. on p. 39).

Schwarz, Katja, Yiyi Liao, Michael Niemeyer, and Andreas Geiger (n.d.). "GRAF: Generative Radiance Fields for 3D-Aware Image Synthesis". In: (), p. 13 (cit. on pp. 20, 28).

Serrà, Joan, Santiago Pascual, and Carlos Segura (2019). *Blow: A Single-Scale Hyperconditioned Flow for Non-Parallel Raw-Audio Voice Conversion*. arXiv: 1906.00794 [cs, eess, stat]. URL: http://arxiv.org/abs/1906.00794 (visited on 06/25/2021) (cit. on p. 15).

Sitzmann, Vincent, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein (2020). *Implicit Neural Representations with Periodic Activation Functions*. arXiv: 2006.09661 [cs, eess]. URL: http://arxiv.org/abs/2006.09661 (visited on 03/15/2021) (cit. on pp. 9, 17, 20, 21, 27, 31, 65, 85, 90, 93).

Skorokhodov, Ivan, Savva Ignatyev, and Mohamed Elhoseiny (2020). *Adversarial Generation of Continuous Images*. arXiv: 2011.12026 [cs]. URL: http://arxiv.org/abs/2011.12026 (visited on 02/02/2021) (cit. on pp. 21, 27).

Snoek, Jasper, Hugo Larochelle, and Ryan P. Adams (2012). *Practical Bayesian Optimization of Machine Learning Algorithms*. arXiv: 1206.2944 [cs, stat]. URL: http://arxiv.org/abs/1206.2944 (visited on 03/21/2021) (cit. on pp. 44, 94).

Stanley, Kenneth O. (2007). "Compositional Pattern Producing Networks: A Novel Abstraction of Development". In: *Genetic Programming and Evolvable Machines* 8.2, pp. 131–162 (cit. on p. 19).

Sutskever, Ilya, James Martens, and Geoffrey Hinton (n.d.). "Generating Text with Recurrent Neural Networks". In: (), p. 8 (cit. on p. 13).

Tancik, Matthew, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng (2020). *Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains*. arXiv: 2006.10739 [cs]. URL: http://arxiv.org/abs/2006.10739 (visited on 02/02/2021) (cit. on pp. 9, 16, 20).

Theunissen, Frédéric and Julie Elie (2014). "Neural Processing of Natural Sounds". In: *Nature reviews. Neuroscience* 15, pp. 355–66 (cit. on p. 14).

Turian, Joseph and Max Henry (2020). *I'm Sorry for Your Loss: Spectrally-Based Audio Distances Are Bad at Pitch*. arXiv: 2012.04572 [cs, eess]. URL: http://arxiv.org/abs/2012.04572 (visited on 02/24/2021) (cit. on p. 56).

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). *Attention Is All You Need*. arXiv: 1706.03762 [cs]. URL: http://arxiv.org/abs/1706.03762 (visited on 06/24/2021) (cit. on p. 13).

Warden, Pete (2018). *Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition*. arXiv: 1804.03209 [cs]. URL: http://arxiv.org/abs/1804.03209 (visited on 03/29/2021) (cit. on p. 41).

Wu, Wenxuan, Zhongang Qi, and Li Fuxin (2020). *PointConv: Deep Convolutional Networks on 3D Point Clouds*. arXiv: 1811.07246 [cs]. URL: http://arxiv.org/abs/1811.07246 (visited on 02/14/2021) (cit. on p. 21).

Xu, Xingqian, Zhangyang Wang, and Humphrey Shi (2021). *UltraSR: Spatial Encoding Is a Missing Key for Implicit Image Function-Based Arbitrary-Scale Super-Resolution*. arXiv: 2103.12716 [cs]. URL: http://arxiv.org/abs/2103.12716 (visited on 06/23/2021) (cit. on p. 9).

Zhang, Hongyang, Junru Shao, and Ruslan Salakhutdinov (2018). *Deep Neural Networks with Multi-Branch Architectures Are Less Non-Convex*. arXiv: 1806.01845 [cs, stat]. URL: http://arxiv.org/abs/1806.01845 (visited on 06/29/2021) (cit. on p. 78).

Zhuang, Juntang, Tommy Tang, Yifan Ding, Sekhar Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James S. Duncan (2020). *AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients*. arXiv: 2010.07468 [cs, stat]. URL: http://arxiv.org/abs/2010.07468 (visited on 03/21/2021) (cit. on pp. 44, 94).

Appendix

A.1 Periodic activation scaling analysis

In this section we evaluate the severity of the negative effects of the expressivity pressure created by optimizing in a short training regime. Then, we test methods to counter this expressivity pressure post hoc and methods to circumvent having to optimize ω_0's altogether.

Experiments in the periodic activation scaling analysis section consider SIRENs for reconstructing a single waveform. These experiments are used to validate theories about the interaction between different SIREN parameterizations and output characteristics. SIRENs are parameterized as in the audio experiments of Sitzmann et al.[8]:

[8] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

1. 5 layers
2. 256 hidden units
3. First ω_0: 3000
4. Hidden ω_0: 30
5. Optimizer: Adam
6. Learning rate: 5×10^{-5}

We assume that the behavior of a SIREN reconstructing a single waveform is indicative for the behavior of a FiLMed SIREN reconstructing a set of waveforms.

We fit using MSE as the objective function :

$$\mathcal{L} = \frac{1}{M} \sum_{j=1}^{M} \left\| \Phi(t_j, \mathbf{z_i})) - \text{Amp}_i(t_j) \right\|^2 .$$

We test the effect of L2 weight regularization implemented as follows:

$$\mathcal{L} = \frac{1}{M} \sum_{j=1}^{M} \left\| \Phi(t_j)) - \text{Amp}_i(t_j) \right\|^2 + \frac{\lambda}{2} \|\mathbf{W}_\phi\|^2$$

A.1.1 Reconstructing mixed sines with different ω_0's

As noted by Sitzmann et al.[9], the weights of a SIREN can be interpreted as angular frequencies, networks with larger weights output

[9] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

higher frequencies. $omega_0$, or γ in the case of π-GAN, multiplies the weights network weights by a constact factor, directly influencing the potential frequency output of the network and its flexibility at finer scales.

Considering the results of all SIREN based parameterizations of ϕ, we see perceived background noise in reconstructions (see figure 5.3) and rough waveform shapes (figure 6.1, 6.2) we hypothesize that these parameterizations suffer from being too expressive, see statement 6.2.6. We suspect that an important reason for this is that the ω_0 selection procedure (see section 6.1) which optimizes for reconstruction performance in a minimal amount of iterations (equation 4.7 after 200 epochs), favouring short term progress, resulting in suboptimal expressiveness, see statement 6.2.6. We validate if optimal ω_0's in a short amount of iterations indeed result in suboptimal final performance.

We consider fitting a target signal consisting of two summed sine waves of 1000 Hz and 6000 Hz with a duration of 1 second, normalized to have an amplitude between -1 and 1. Fitting is done at observations sampled at 16000 Hz. Results after 200 and 1000 iterations (after which models are converged) are shown in figure 1 and 2. Reported average MSE is calculated over the densely sampled signal and reconstructions, as plotted in the figures.

We observe the following:

1. First ω_0 values with lower magnitudes showing extremely suboptimal results in early training can turn out to be optimal in late training.
2. Higher magnitudes of first ω_0 take significanty less iterations to reconstruct higher frequency components.
3. Higher magnitudes of first ω_0 are quicker to overfit on sparsely sampled target signal observations by introducing frequency components higher than those present in the target signal.
4. SIRENs show spectral bias[10]. lower frequency signal components are approximated first.
5. SIRENs with lower first ω_0 magnitudes show less variation between runs.

Observation 2 and 3 can be viewed the underlying reason for observation 1. These results are in line with the hypothesis that our ω_0 selection procedure for the experiments in section 6.1 and 6.2 can be one of the reasons for the observed background noise in respective experiments.

[10] Rahaman, Baratin, Arpit, Draxler, Lin, F. Hamprecht, et al., "On the Spectral Bias of Neural Networks", 2019.

A.1.2 ω_0 search space

Knowing the importance of ω_0 magnitudes and the fact that optimal results in early training are not guaranteed to result in optimal final

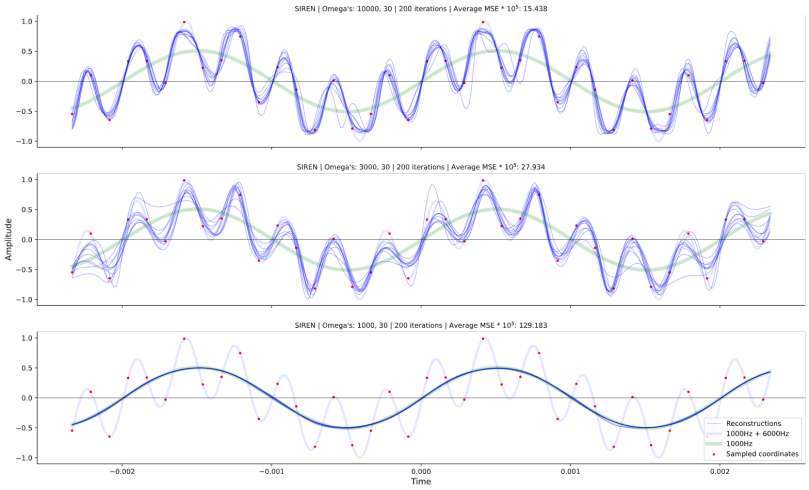


Figure 1: SIRENs fitted to two mixed sine waves of 1000Hz and 6000Hz after 200 iterations for different magnitudes of ω_0 first for 10 different seeds with average MSE .

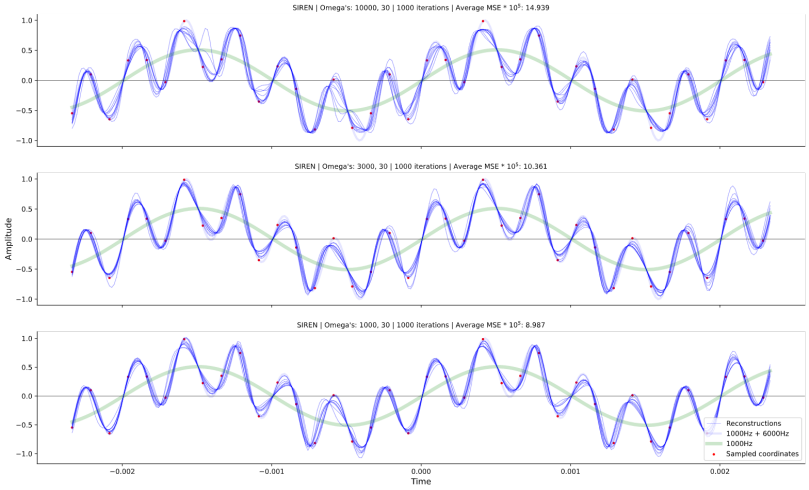


Figure 2: SIRENs fitted to two mixed sine waves of 1000Hz and 6000Hz after 1000 iterations for different magnitudes of ω_0 first.

performance, the question arises how to select ω_0 magnitudes when training until convergence for many values is unfeasible. Considering the direct relationship between ω_0 magnitudes and spectral content, a promising avenue seems to be to determine ω_0 magnitudes by analyzing the frequency spectra of signals in the dataset you aim to reconstruct. A robust analysis of the relation between frequency spectra in a dataset and optimal ω_0 magnitudes would require training on many datasets with diverse spectral contents for a Wide range of ω_0 magnitudes until convergence, which is out of the scope of this research.

However, if we assume that relations between optimal ω_0's in the short training- and optimal $\omega 0$'s in the training until convergence regime are similar and optimal ω_0's in the short training regime are never of lower magnitude, analysis of the results of our ω_0 sweeps is useful. Heuristics for relations between optimal ω_0's in the short training regime can then be used for shaping the upper limit of the search space of optimal $\omega 0$'s in the training until convergence regime.

To see if certain ϕ parameterizations affect optimal ω_0's in the short training regime, we plot the 5% best perfoming ω_0 values for all autodecoder setups which have first- and hidden ω_0 as parameters and learned at a reasonable speed, see figure 3. We observe the following:

- Alterations to network shape influence optimal ω_0 values.
- Minimizing the mapping network influences optimal ω_0 values.
- A deeper mapping network does not make a significant difference for optimal ω_0 values.

Then, we continue with the setups that show similar optimal ω_0's across datasets: π-GAN vanilla and π-GAN five mapping. We again plot the first ω_0 and hidden ω_0, this time of the top 30% of scores of both setups combined, for every dataset, see figure 4. Dots are scaled by score. We observe the following:

1. For NSYNTH Keyboard, first ω_0 magnitudes do not matter much.
2. For Speech Commands, first ω_0 needs to be between $\pm$ 500 and $\pm$ 1000.
3. For NSYNTH Diverse, first ω_0 needs to be between $\pm$ 1600 and $\pm$ 2000.

→ **More spectrally diverse datasets require larger first ω_0's.**
→ **Optimal hidden ω_0 magnitudes never exceed $\pm$ 700**

4. For NSYNTH datasets, a higher first ω_0 demands a higher hidden ω_0. For Speech Commands this trend is less obvious.

→ **More spectrally coherent[11] datasets demand a more specific ratio between first ω_0 and hidden ω_0**

[11] Coherent as in containing pure tones, not as in uniform.

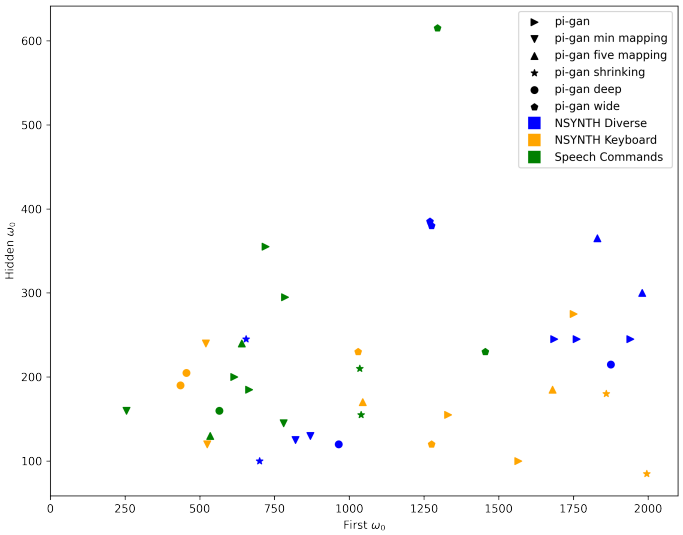


Figure 3: Scatterplot of first Ω_0 and hidden Ω_0 of the top 5% best objective error for for all autodecoder setups which have first- and hidden ω_0 as parameters and learned at a reasonable speed.

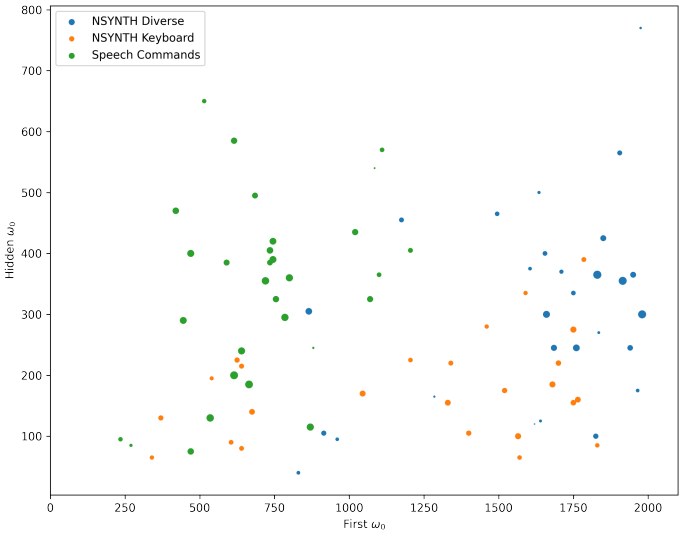


Figure 4: Scatterplot of first ω_0 and hidden Ω_0 of the top 30% best objective error of all runs in the ω_0 sweeps of π-GAN autoencoder, autodecoder, minimal mapping and five mapping setups. Dots are scaled by score.

Relating these observations to the spectral anayisis of the datasets described in section 5.5, they can be used as an indicator for determining the upper limit of search spaces of ω_0's in the training until convergence regime for parameterizations of ϕ similar to π-GAN. Keep in mind these observations are based on only three datapoints (NSYNTH Keyboard, NSYNTH Diverse and Speech Commands). Within these three samples, there are many potentially important factors unvaried, such as dataset size, audio length and subsampling ratio during training.

A.1.3 Coping with expressiveness pressure

Considering the scenario where we search optimal ω_0's in the short training regime, we can think about methods to cope with associated expressiveness pressure. The weights of a SIREN can be interpreted as angular frequencies[12]. Thus, to suppress higher frequencies in SIREN representations it might help to enforce weight regularization.

[12] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

We continue with the optimal $\omega_0's$ found after 200 iterations in the previous section $(10000, 30)$, and apply weight regularization as in equation A.1. We observe that convergence takes longer with weight regularization, and train for 2000 iterations instead of 1000 in the previous experiment. The experiment is executed for different values of λ: 0.1, 0.3, 1, see figure 5.

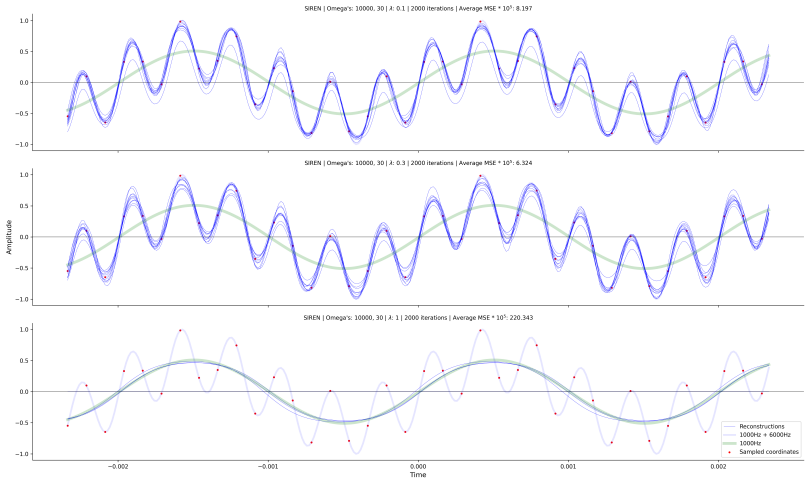


Figure 5: SIRENs fitted to two mixed sine waves of 1000Hz and 6000Hz after 200 iterations for different values of λ for 10 different seeds with average MSE .

- For λ is 1, SIREN representations are still silent after 2000 iterations
- Results are less sensitive to λ's value than to the values of the ω_0's.
- For λ's below 1, reconstructions after 2000 iterations significantly outperform optimal ω's in previous setups.

In this simplified scenario, weight regularization shows the effects we hoped for. It seems that, if λ is not set too high, it succesfully counters the flexibiltity pressure when searching for optimal ω's in the short training regime and imporoves upon earlier results. Although this method introduces yet another hyperparameter and increases required training iterations, the higher robustness of reconstruction quality for λ's compared to ω's is a step forward.

Finally, we test if progressively reducing weight regularization and progressively introducing nodes with higher activation scalings (ω_0's) in the first SIREN layer can aid in making reconstructions more robust towards hyperparameter values. We hope to find that these methods will reduce overfitting by forcing SIRENs to initially use lower frequencies to approximate a signal. Then, when the SIREN is *allowed* to use higher frequencies, these will only be introduced if needed.

First, we again train for 2000 iterations and progressively decrease λ linearly to zero in different amounts of iterations to gauge the effect of this. For these experiments we selected a λ starting value of 1.5. See figure 6 for results.

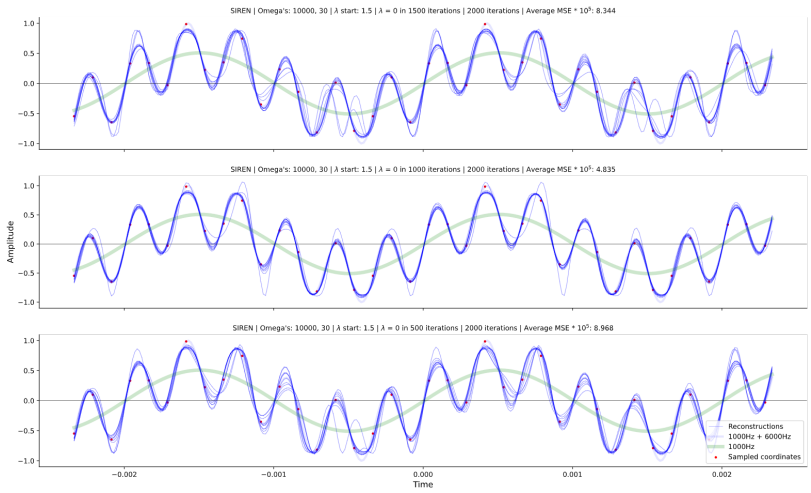


Figure 6: SIRENs fitted to two mixed sine waves of 1000Hz and 6000Hz after 200 iterations for different values of λ for 10 different seeds with average MSE .

- Progressively reducing weight regularization is not very sensitive to the amount of iterations where λ is decreasing.
- Results show significantly more variance when there are a relativively large or small amount of training iterations left without weight regularization.
- Progressively reducing weight regularization with an equally amount of iterations where λ is decreasing versus where λ is zero is optimal within tested runs.

Then, we fix the amount of iterations in which λ decreases to zero and test for different values of λ to test robustness of reconstructions with respect to this parameter. See figure 8 for results.

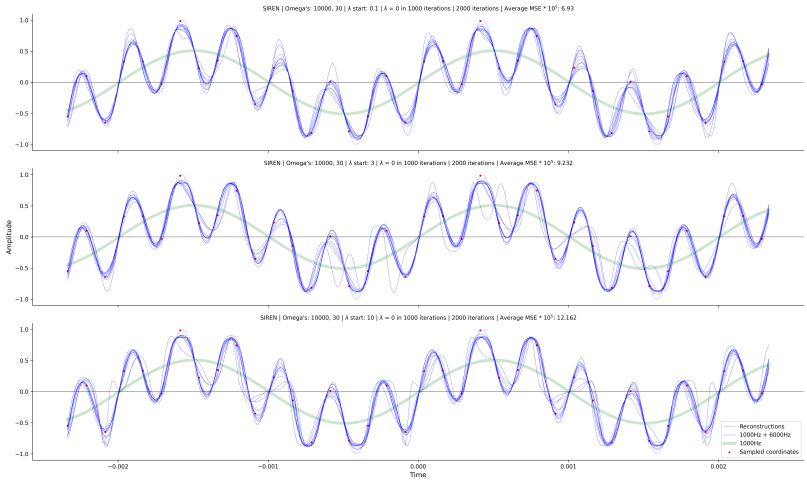


Figure 7: SIRENs fitted to two mixed sine waves of 1000Hz and 6000Hz after 200 iterations for different values of λ for 10 different seeds with average MSE .

- Results are more robust with respect to λ compared to non progressive weight regularization, although extreme values of λ still results in degraded results.
- Reconstructions are slightly lower quality and contain more variation compared to constant weight regularization.
- The extra robustness comes at the cost of another hyperparameter; in how many iterations to decrease λ to zero.

Finally, we once again train SIRENs for 2000 iterations and progressively introduce nodes with higher activation scalings (ω_0's) in the first SIREN layer, without weight regularization. This method was inspired by recent work by Hertz et al.[13], where the authors propose to apply a similar technique in ReLU P.E. MLPs (ReLU MLPs with a Fourier based positional encoding in the first layer).

[13] Hertz et al., *SAPE*, 2021.

256 different entries for the first ω_0's are picked from a uniform distribution between 1 and 10000, sorted, and then divided in 8 groups. At the beginning of training, all ω_0's are masked. The first half of iterations is split in 8 periods (the amount of ω_0 groups). in the first half of each period a new group of ω_0's is linearly unmasked, starting with the lowest magnitudes. In the second half of each period the mask stays the same. In the second half of the predetermined amount of iterations, all ω_0's are unmasked and stay that way[14]. See figure 8 for results.

[14] This method was selected by testing on a π-GAN setup with a dataset size of 128 samples. We also tried selecting ω_0's as evenly spaced numbers over a specified interval and treating every ω_0 value as it's own group.

- Reconstructions do not improve upon previous results from weight regularization and are on par with searching optimal ω_0's in a short training regime.
- Reconstruction quality is relatively consistent for a large range of ω_0's, making this approach require almost no hyperparameter tuning for the considered scenarios.

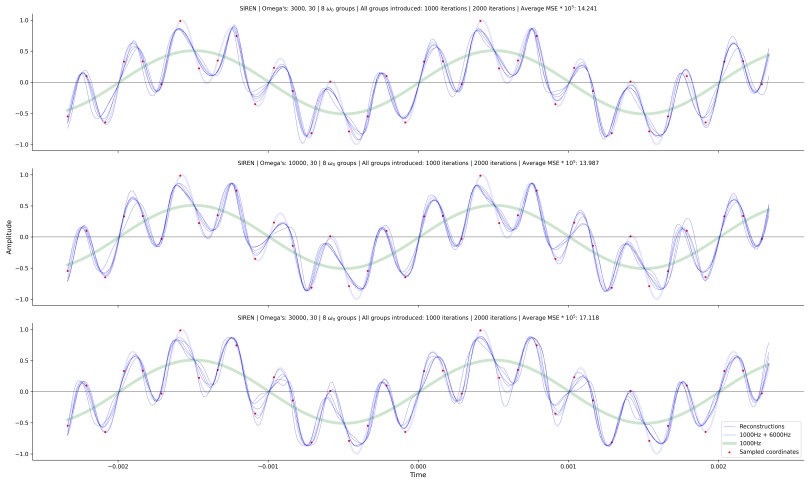


Figure 8: SIRENs fitted to two mixed sine waves of 1000Hz and 6000Hz after 200 iterations for different values of λ for 10 different seeds with average MSE .

Discussion

- progressively introducing nodes with higher activation scalings (ω_0's) in the first SIREN layer most robust solution, might be able to circumvent hyperparamter search altogether, but does not show optimal reconstruction quality
- Constant weight regularization shows best reconstruction quality.
- Progressive weight regularization sits in between the latter two methods for robustness and reconstruction quality.

A.2 Unsuccessful experiments

Among others, the following ideas were evaluated in prelimiary experiments, but resulted in inferior performance.

1. Using double precision
2. Many loss functions, including perceptual loss functions such as multi resolution STFT and CDPAM.
3. Recurrent encoder (wav2vec)
4. ω_0's as trainable parameters
5. Hypernetwork: tried with initialization methods as described by Sitzmann et al.[15].
6. Film conditioning: scaling ω_0 with γ.
7. Progressive activation scaling: reported method was first validated on a π-GAN setup with a dataset size of 128 samples. We also tried selecting ω_0's as evenly spaced numbers over a specified interval and treating every ω_0 value as its own group, fading them in one by one. These alterations performed slightly worse.

[15] Sitzmann et al., *Implicit Neural Representations with Periodic Activation Functions*, 2020.

A.3 Reproducability details

A.3.1 Hardware

All experiments were executed on single Titan RTX GPU with 24 gigabytes of GDDR6 memory.[16]

[16] We would like to thank SURFsara for providing the GPU nodes on the lisa system.

A.3.2 Hyperparameter search results

Optimal learning rate and optimizer results are uniform within setups for implicit and convolutional architectures:

- WaveGAN autoencoder and autodecoder:
  - Learning rate: 1×10^{-4}
  - Optimizer: Adabelief
- π-GAN and IM-NET autoencoder and autodecoder:
  - Learning rate: 1×10^{-5}
  - Optimizer: Adabelief

  These hyperparameters are kept constant in all experiments.

Hyperparameter search details

Fixed hyperparameters

To minimize the hyperparameter space to explore, some parameters were fixed throughout the hyperparameter search based on results in preliminary experiments:

- Batch size: 128
  - Preliminary experiments indicated positive effects for larger batch sizes. 128 was the maximum power of 2 that fitted in memory on a single Titan RTX.
- Latent embedding size: 256
  - WaveGAN showed significant performance degradation in latent embedding sizes below 256 in preliminary experiments.

Hyperparameter search procedure

1. For implicit architectures; execute a ω_0 or coordinate multiplier search, with fixed learning rate at 1×10^{-5} and Adabelief[17] as the optimizer[18], for 200 epochs, using Bayesian Search[19] with Hyperband[20] early stopping. For π-GAN decoder setups:
   - ω_0 first $[1, 2000]$
   - ω_0 hidden $[1, 2000]$

   For IM-NET decoder setups:
   - Coordinate multiplier $[1, 3000]$

[17] Zhuang et al., *AdaBelief Optimizer*, 2020.

[18] A learning rate of 1×10^{-5} and Adabelief were optimal in prelimiary experiments.

[19] Snoek et al., *Practical Bayesian Optimization of Machine Learning Algorithms*, 2012.

[20] L. Li et al., *Hyperband*, 2018.

2. Grid search (with previously best found ω_0 or coordinate multiplier, if applicable) for 5000 epochs over the following hyperparameters, three times:

   (a) Learning rate: $\{1 \times 10^3, 1 \times 10^4, 1 \times 10^5, 1 \times 10^6\}$

   (b) Optimizer: {Adam, Adabelief}

All ω_0 optimization runs were executed for 5 hours, after which the process was executed.

A.3.3 Baseline experiments

Encoder paramterization

In autoencoder setups we use a convolutional encoder parametrized as follows:

1. Conv1d(1, 128, 320, 4)
2. ReLU()
3. MaxPool1d(4)
4. Conv1d(128, 128, 3)
5. ReLU()
6. MaxPool1d(4)
7. Conv1d(128, 256, 3)
8. ReLU()
9. MaxPool1d(4)
10. Conv1d(256, 512, 3)
11. ReLU()
12. MaxPool1d(4)
13. AvgPool1d(14)
14. Linear(512, 256)

Resulting in 714k parameters, roughly equivalent to the decoders. A first-layer filter size of 320 is used, corresponding to 20ms in 16kHz audio.

A.4 Additional results and observations

A.4.1 Baseline experiments

1. IM-NET autoencoder: best CSIG (background noise level) scores for Speech Commands[21].

2. Autoencoder architectures:

   (a) WaveGAN outperforms other architectures for the NSYNTH diverse dataset.

   (b) π-GAN outperforms other architectures for the NSYNTH keyboard dataset, except for Multi resolution STFT MSE (background noise level).

   (c) WaveGAN: MSE close to silence for Speech Commands dataset.

[21] Note that CSIG scores representing background noise level in the Speech Commands dataset are believed to be better for lower scores, since CSIG showed a positive correlation with background noise level ratings in the Speech Commands dataset. while for CSIG representing sample quality in NSYNTH datasets a higher score is better, as in the original literature. Thus, these results should be interpreted with caution.

(d) π-GAN: best MSE for NSYNTH keyboard and Speech Commands datasets.

A.4.2 Ablation experiments

Architecture	CDPAM: Overall quality Diverse	Keyboard	CSIG: Sample quality Diverse	Keyboard	Multi STFT MSE: Noise Diverse	Keyboard
π-GAN	**0.35** $\pm$ 0.24	0.48 $\pm$ 0.18	**1.38** $\pm$ 1.84	**-1.97** $\pm$ 2.45	**0.06** $\pm$ 0.02	0.14 $\pm$ 0.04
- Concat middle	1.82 $\pm$ 0.0	1.6 $\pm$ 0.01	-8.32 $\pm$ 0.08	-10.19 $\pm$ 0.27	0.13 $\pm$ 0.0	**0.1** $\pm$ 0.0
- Concat all	1.81 $\pm$ 0.02	1.6 $\pm$ 0.0	-8.41 $\pm$ 0.17	-10.77 $\pm$ 0.59	0.13 $\pm$ 0.0	0.1 $\pm$ 0.0
Silence	1.65 $\pm$ 0.0	1.77 $\pm$ 0.0	-1.31 $\pm$ 0.0	-4.62 $\pm$ 0.0	0.16 $\pm$ 0.00	0.16 $\pm$ 0.00
White noise	1.06 $\pm$ 0.27	1.06 $\pm$ 0.32	-5.43 $\pm$ 3.14	-9.31 $\pm$ 2.59	2.27 $\pm$ 0.06	2.32 $\pm$ 0.04

Table 1: Evaluation metrics CDPAM, CSIG and multi resolution STFT MSE scores of conditioning ablation setups for NSYNTH datasets. For CDPAM and multi resolution STFT MSE a lower score is better, for CSIG representing sample quality in NSYNTH datasets a higher score is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

Architecture	LSD: Overall / Sample quality	CSIG: Noise
π-GAN	**0.27** $\pm$ 0.12	0.59 $\pm$ 1.26
- Concat middle	0.49 $\pm$ 0.0	**-2.0** $\pm$ 0.04
- Concat all	0.49 $\pm$ 0.0	-1.55 $\pm$ 0.07
Silence	0.74 $\pm$ 0.0	-2.05 $\pm$ 0.0
White noise	7.27 $\pm$ 0.07	-1.24 $\pm$ 1.93

Table 2: LSD $\times 10$ and CSIG scores of all conditioning ablation setups for the Speech Commands dataset. For both metrics lower is better. Silence and white noise ($\mu = 0, \sigma = 1$) scores reported for reference.

Architecture	NSYNTH Diverse		NSYNTH Keyboard		Speech Commands	
π-GAN	1.32	$\pm$ 0.3	4.48	$\pm$ 1.02	0.85	$\pm$ 0.01
- Concat middle	38.57	$\pm$ 0.0	49.13	$\pm$ 0.01	2.39	$\pm$ 0.0
- Concat all	38.58	$\pm$ 0.01	49.1	$\pm$ 0.0	2.39	$\pm$ 0.0
Silence	46.19	$\pm$ 0.0	78.74	$\pm$ 0.0	7.48	$\pm$ 0.0
White noise	1061.49	$\pm$ 13.56	1094.74	$\pm$ 12.03	1023.18	$\pm$ 11.56

Table 3: Mean and standard deviation of MSE $\times 10^3$ statistics of batches (128 samples) of synthesized samples in the 5000th epoch of all models with best performing hyperparameters in conditioning ablation setups. Silence and white noise ($\mu = 0, \sigma = 1$) errors reported for reference.

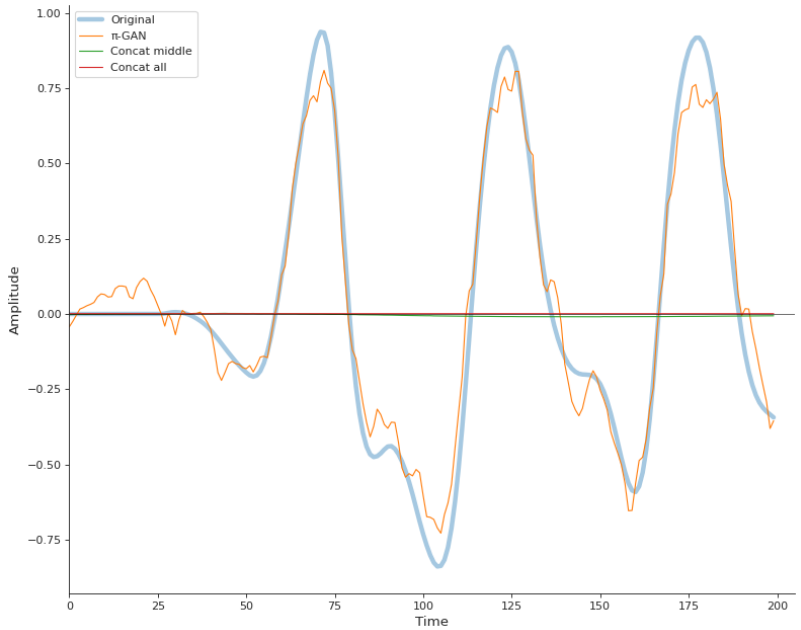


Figure 9: First 200 timepoints of a waveform validated to be representative as described in section 5.9.1 of the NSYNTH keyboard dataset and reconstructions of conditioning ablation setups.

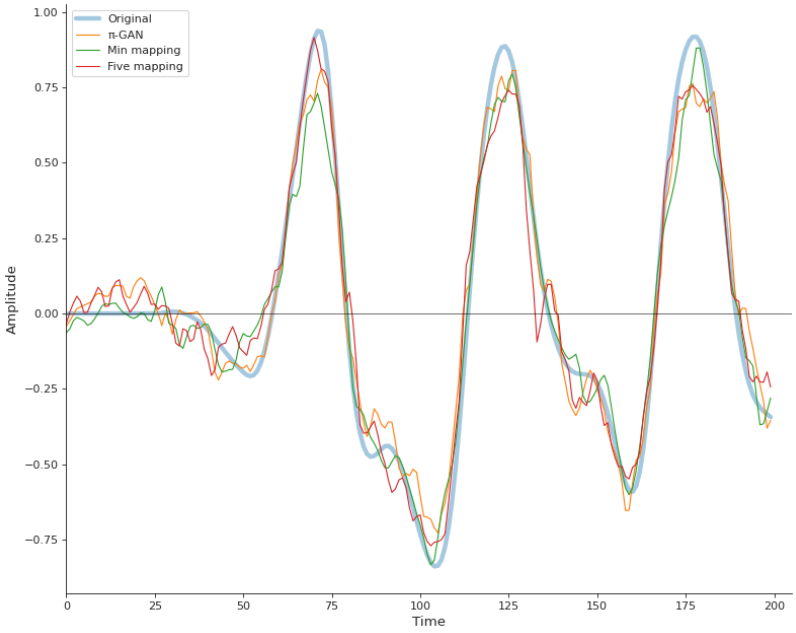


Figure 10: First 200 timepoints of a waveform validated to be representative as described in section 5.9.1 of the NSYNTH keyboard dataset and reconstructions of latent mapping network ablation setups.

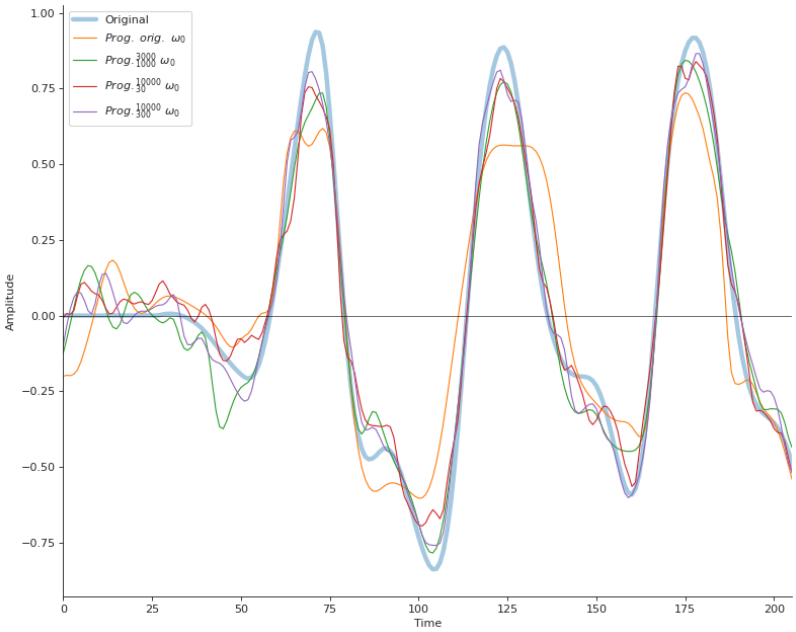


Figure 11: First 200 timepoints of a waveform validated to be representative as described in section 5.9.1 of the NSYNTH keyboard dataset and reconstructions of progressive activation scaling setups.