HRTF Generation for Data Demanding Machine Learning Algorithms

Benjamin Tsui

PhD

University of York
Electronic Engineering

January 2023

Abstract

This thesis investigates the application of Machine Learning (ML) techniques to binaural audio research. Whilst there is plenty of work done in this domain currently, much of it is limited by the amount of available Head Related Transfer Function (HRTF) data required to train modern neural network-based ML models, resulting in researchers using a less data-driven approach or finding some workaround with the limited data. This thesis focuses on the generation of enough data to unleash the power of a wide variety of modern ML algorithms. A novel method is presented that can simulate unlimited realistic HRTFs using heads generated from Three-dimensional Morphable Models (3DMMs). The result has led to the creation of the HUman Morphable Model-based Numerically Generated Binaural Impulse Response Database (HUMMNGBIRD) database, created with the first 5000 HRTF sets generated by this method. Principle Component Analysis (PCA) and Variational Auto-Encoder (VAE) reconstruction models were created to investigate the potential of such a large amount of data. The results provide valuable insights into the research directions that could make good use of these types of artificially generated databases in the near future.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my greatest gratitude to my main supervisor, Prof. Gavin Kearney, for his invaluable patience, guidance, and support throughout my studies. I am extremely fortunate to have him in my life, as he has consistently demonstrated an unwavering willingness to support my study. I cannot imagine undertaking this journey with anyone else.

I would also like to thank my mother, who funded my PhD, for her trust and support. As someone who has been fortunate enough to grow up in a financially stable environment, I have always felt a sense of obligation to give back to the world while pursuing my passions. I am grateful to my mother for her generosity and belief in me, even as I pursued a research path that may not have been fully understood by her.

I am also thankful to my second supervisor, Dr William Smith, who provided invaluable guidance on the machine learning aspects of my research and brought the crucial 3DMM model that made the creation of the HUMMNGBIRD database and its related work possible. Now that the database is mostly ready, I am excited to make use of the data we have created and continue our work in the future.

I would also like to thank my dear friends at the AudioLab, specifically Tomasz, Dan, Tom, Cal, Kat, and Jess, for their support and insightful conversations. Their company helps me go through some very difficult times and they have truly made me a better person.

Finally, I would like to express my gratitude to the restaurants in York, especially Upper River, for providing the social interaction that is essential for my well-being as a human being.

Declaration

I, Benjamin Tsui, declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as references.

In addition, I declare that parts of this research have been presented at conferences during the course of the research degrees. The related publications are as follows:

B. Tsui and G. Kearney, "A Head-Related Transfer Function Database Consolidation Tool for High Variance Machine Learning Algorithms," in Audio Engineering Society Convention 145, 2018

B. Tsui, W. A. P. Smith, and G. Kearney, "Low-order spherical harmonic HRTF restoration using a neural network approach," Appl. Sci., vol. 10, no. 17, 2020.

HRTF Generation for Data Demanding Machine Learning Algorithms

Chapter 1

Introduction

Spatial audio allows listeners to perceive the location, movement, and distance of reproduced sound sources over loudspeakers or headphones in a more realistic and immersive way. In contrast to traditional stereo audio, in which the sound can be perceived between the left and right speakers or 'in head' in the case of headphones, spatial audio creates a sense of directionality and depth. It is important for Virtual Reality (VR) and Augmented Reality (AR) applications, where it can be used to create a sense of presence and immersion in a virtual environment, allowing users to feel like they are in the same room as the sound sources. Spatial audio is also used to enhance the listening experience for other digital content, such as movies, music, and video games. The most common and accessible way to experience spatial audio is through headphones, where binaural filters known as Head-Related Transfer Functions (HRTFs), are used. HRTFs describe how sounds change as they travel from an emitting source to a listener's ears from different directions and distances. By convolving (filtering) a sound with HRTFs, listeners can perceive the location, movement, and distance of the sound in 3D space when listening on headphones.

There is ongoing psychoacoustics research into HRTFs to improve the spatial audio experience, in particular in the measurement or simulation of HRTFs. Despite this, HRTFs are still scarce and hard to obtain, making it challenging to apply modern machine learning techniques to HRTF research, where a lot of training data is often required. This thesis shares the work on obtaining HRTF data for modern machine learning algorithms which may potentially lead to new advancements in HRTF research.

1.1 Motivation

HRTFs encapsulate the Interaural Time Difference (ITD), Interaural Level Difference (ILD) and other spectral cues introduced by the ear pinnae and torso from different angles relative to the head in three dimensions [1, 2]. Sound sources can be spatialised by direct convolution with a given HRTF pair representing the intended sound source direction. However, HRTFs are highly personalised due to different head and ear shapes. Using mismatched HRTFs can affect timbre quality, localisation performance and externalisation [3–7]. This leads to a lot of HRTF-related research, like personalised HRTFs, HRTF interpolation and HRTF processing techniques.

At the same time, developments in machine learning have shown great improvement in neural style transfer and data restoration, especially in the image domain, such as noise reduction in images, image inpainting, colourising old photos or videos [8–16]. These models can potentially be applied to HRTF research since HRTF data and images have many characteristics in common, where the data are coherent between one input and the ones next to it. For example, the magnitude of frequency bins in the HRTFs are related to each other. Also, the HRTFs are coherent between one angle and the ones surrounding it. Some examples models include variants of fully connected Neural Networks (NN), Convolution neural network (CNN), Auto-encoder (AE), Convolution Auto-encoder (CAE) [15], Residual Network (ResNet) [17] and Generative Adversarial Networks (GANs) [18]. However, these models often take a data-driven approach, where a high volume of training data is required to develop the model. Currently, the number of available HRTF databases is very limited. There are only around 500 HRTF datasets freely available from databases, including ARI [19], ITA [20], RIEC [21], SADIE I [22], SADIE II [23], CIPIC [24], IRCAM LISTEN [25] and the TU Berlin KU100 database [26]. Compared to the data size used to train image processing machine learning models, which can be in the region of hundreds of thousands of images, HRTF data is lacking in this regard.

The primary way to obtain personalised HRTFs is through physical measurements, where microphones are placed at the ear canal of a subject and the loudspeakers positioned at different angles relative to the head to measure the transfer functions [23, 27]. This measurement process is often tedious and requires substantial setup and calibration. Recent developments have been made in HRTF-based selection based on anthropomorphic data extracted from photographs of the ear [28] or HRTF simulation using 3D head models [29]. However, simulation is computationally expensive and usually requires a lot of processing time [30, 31]. This makes obtaining a large amount of HRTFs with a uniform configuration very difficult. Moreover, the databases created by measurement are often skewed due to the lack of diversity in participants. Furthermore, the complexity of properly setting up the measurement facility means that the diversity of participants is bounded by the geographical location of the setup.

This thesis therefore presents work that led to the creation of the HUman Morphable Model-based Numerically Generated Binaural Impulse Response Database (HUMMNG-BIRD) database, created for use with large data-driven machine learning algorithms. The database uses a novel method that can simulate unlimited realistic HRTFs using heads generated from Three-dimensional Morphable Models (3DMMs). The database is then applied to two ML models to study if the amount of data provides any noticeable improvement.

1.2 Objectives

At the beginning of the PhD, the primary goal was to apply neural network machine learning models to HRTF research. There were two obstacles at that time. The first issue is the size of a dense HRTF set is often too big to use on a deep neural network with a single GPU. Before 2020, even a high-end GPU, like the RT 2080 Ti, could only offer 11GB of memory. The second issue is the number of HRTF sets is limited compared with what a typical deep neural network needs. To further complicate issues, the sampling of the participants from a single HRTF database is often skewed due to the geographical location and practicality, and combining multiple HRTF databases can be challenging as the configuration between different databases can be vastly different. Fast forward to 2022, and the size of an HRTF set is no longer a big problem, as some high-end GPUs can provide up to 24GB of memory. However, HRTF sets are still scarce. Since the number and the quality of the data play an important role in deep neural network training, this work, therefore, focuses on finding a way to gather HRTF sets that could build the foundation for future HRTF research with machine learning models.

This work addresses the following two hypotheses:

1. **That a large number of HRTFs are required for machine learning applications, especially for training neural network models.**

2. **That a feasible alternative to acoustically measured HRTFs can be derived through computational means to facilitate large-scale HRTF data generation for machine learning.**

To address these two hypotheses, the following work has been done:

1. **Propose a robust and flexible way to consolidate HRTF databases**

   A MATLAB toolbox was created for consolidating different HRTF databases. It is capable of finding the common angles across multiple databases with specific angle tolerance at a relatively fast speed, finding abnormal data, and normalising and pre-processing HRTF data. It also includes a flexible plot HRTF angles function that can plot all HRTF measurement points (angle and distance) in three dimensions.

2. **Train a neural network with the consolidated HRTF data and evaluate the performance**

   A novel Neural Network model that can restore the distortion in 1st order Spherical Harmonic interpolated HRTFs is proposed. SH interpolation is one of the most robust methods for HRTF interpolation. However, the SH interpolated HRTFs are only accurate up to a certain spatial aliasing frequency. A novel Neural Network model that can restore the distortion in 1st-order SH interpolated HRTFs is proposed. The training data was consolidated from 7 HRTF databases with the MATLAB consolidation toolbox.

3. **Propose an HRTF interpolation method to obtain more HRTF measurements within a single set when consolidating HRTF databases**

   The Neural Network model that was trained to restore the distortion above the spatial aliasing frequency in 1st order Spherical Harmonic interpolated HRTFs shows that it is possible to improve the SH interpolated HRTFs result with an ML model. However, it also shows that there is room for improvement with more HRTF data.

4. **Since the consolidated HRTF database does not perform as expected due to the quantity and quality of the consolidated data, this thesis**

proposes a method to generate unlimited HRTF sets that are similar to acoustically measured HRTFs using heads generated from Three-dimensional Morphable Models (3DMMs)

In the current ML era, whoever owns the training data or the data pipeline will have the advancement in ML model development. A novel method that can simulate unlimited realistic HRTFs is presented. The main contribution is the tedious work of combining the state-of-the-art Three-dimensional Morphable Models (3DMMs) of human heads with the Boundary Element Method (BEM) to simulate HRTFs from the generated models. During the process, different challenges need to be solved, including finding the right configuration, developing a streamlined process that is most efficient for generating large amounts of data, identifying abnormal data, and optimising the generated data in postprocessing. It led to the creation of the HUman Morphable Model-based Numerically Generated Binaural Impulse Response Database (HUMMNGBIRD), which was created with the first 5000 HRTF sets generated by this method.

5. **After generating over 5000 HRTF sets with the above method, evaluate the performance with a non-neural network machine learning model and a deep neural network model.**

   The model for SH interpolated HRTFs, as presented earlier, processes each HRTF measurement separately rather than utilising the complete HRTF dataset. This approach was necessitated by the limited availability of training data and computational power at the time. When the model was developed, there were few HRTF datasets publically available, and the common measurement angles across these datasets were quite limited, even with considerable tolerances. As a result, the method needed to handle each HRTF measurement individually rather than utilising the entire HRTF dataset. However, considering the development in machine learning and the increased computational power available today, along with the availability of the HUMMINGBIRD HRTF database, treating each HRTF measurement individually is no longer necessary. Employing the entire dataset instead of single HRTF measurements allows for the extraction of valuable information from various HRTF measurements within the HRTF dataset. The HUMMNGBIRD HRTF sets were used to train a few Principal Component Analysis (PCA) and Variational Autoencoder (VAE) models with different numbers of Principal Components (PCs) or latent variables for HRTF restoration and generation. Due to the characteristic of PCA and VAE models, the result can represent a good estimation of performance in future ML applications.

6. **Evaluate the if there is a significant impact on the number of training data with those models.**

   The result shows that the VAE can outperform PCA in extreme conditions. It also shows that the size of the HRTF training set has a noticeable impact on the results of both methods. However, the VAE seems to be more sensitive to the size of the training data than PCA. It also shows that all the models can generate synthetic HRTF sets similarly well. For VAE, a limited number of training data has a more negative impact than having a small of latent variables. This proves that for neural network model training, having a good amount of high-variance HRTF sets is beneficial for the model performance. Both PCA and VAE models can be utilised for dimensionality reduction, indicating that the principal components or latent space can effectively compress the training data into a significantly lower dimension. This reduced dimensionality facilitates future ML-based HRTF research development, as it allows researchers to leverage the latent space for generating synthetic HRTFs by adjusting the PCs or latent variables according to specific applications.

The thesis is structured as follows:

1.3 Thesis Structure

- **Chapter 2: Literature Review**

  This chapter introduces the background knowledge required to understand the remainder of the thesis. It is separated into two sections, HRTFs and ML. The HRTF section provides an overview of the human auditory system and binaural listening. The ML section outlines the fundamental concepts of ML and introduces some commonly used models.

- **Chapter 3: A Head-Related Transfer Function Database Consolidation Tool For High Variance Machine Learning Algorithms**

  This chapter introduces a MATLAB toolbox created for consolidating different HRTF databases. It outlines the functions that come with the toolbox and the complete workflow for HRTF consolidation. Some of the challenges that were faced when creating the toolbox are also shared, especially in finding common angles across different datasets with a flexible tolerance range in a relatively fast manner.

- **Chapter 4.2: Low-order Spherical Harmonic HRTF Restoration using a Neural Network Approach** This chapter introduces a Neural Network model that can restore the distortion in 1st order Spherical Harmonic interpolated HRTFs. The purpose of this work is twofold - Firstly, it is used to test the practicality of using the consolidated HRTF sets to train a data-driven Neural Network model. Secondly, if the model works when consolidating HRTF sets, it can interpolate the mismatched HRTFs across different HRTF databases instead of extracting the ones with common angles. This can provide more HRTFs and more uniform HRTF datasets for future use.

- **Chapter 5: Generating HRTFs with 3D morphable model of the human head** This chapter proposes a method using Three-dimensional Morphable Models (3DMMs) to generate numerous head models and then using the Boundary Element Method (BEM) to simulate HRTFs from the generated models. This method can generate an unlimited amount of clean and uniform HRTFs with different specifications. HUman Morphable Model-based Numerically Generated Binaural Impulse Response Database (HUMMNGBIRD) database, a large HRTF database with over 5000 dense HRTFs datasets, each with a 1-degree Gaussian grid spatial resolution that has 64442 points, was created by using this method.

- **Chapter 6: Preliminary Investigation on the Potential of Using Extra HRTF Datasets in Machine Learning**

  After creating the HUMMNGBIRD database, the newly generated data was applied to several Principal Component Analysis (PCA) models and Variational Autoencoder (VAE) models for HRTF restoration and generation. Since PCA is a more classical statistical model, and VAE is a deep neural network, comparing these two can show the advantage of each method and the impact of the number of data.

- **Chapter 7: Conclusion**

  This chapter concludes the thesis and discusses some potential future directions for the work.

Chapter 2

Literature Review

This chapter presents a brief review of the essential knowledge to understand the work presented in this thesis. It breaks down into two sections: Head related transfer functions (HRTFs) research and Machine Learning (ML). The HRTF section provides an overview of the fundamentals of HRTFs and their usage. It also highlights the challenge of obtaining HRTF measurements. The ML section first outlines some important concepts of ML, and then discusses both traditional and modern machine learning models.

2.1 Head Related Transfer Functions (HRTFs)

HRTFs are a pair of transfer functions that describe the frequency difference between an original sound source and the same sound measured at the eardrum or at the ear canal entrance of a person's ears. They play an important role in Spatial Audio, Virtual Reality (VR), and Augmented Reality (AR) technologies, as they help to provide a high-fidelity immersive experience in virtual environments [4, 5, 32, 33].

HRTFs are often converted from physical Head Related Impulse Response (HRIR) measurements, which is the time-domain version of HRTF. An HRIR directly represents the impulse response measured at a person's eardrum or at the ear canal entrance. HRIRs can also be simulated by the head and ear pinnae models [1, 34, 35]. Further details about different ways to obtain HRTFs are discussed in Section 2.1.3. An example of an HRTF set is shown in Appendix A Figure A.1.

In this thesis, the terminologies related to HRTFs are defined as follows:

HRIR	A single Head Related Impulse Response of a single position
HRTF	A single Head Related Transfer Function of a single position
Acoustically measured HRTFs / HRTF measurements	A single HRTF that was acquired by acoustic measurement of a single position
HRTF set / HRTF dataset	A set of HRTF measurements of the same subject that contains multiple HRTF measurements of different positions
HRTF database	A database that consists of multiple HRTF datasets
Simulated HRTFs	HRTF measurements acquired by acoustic simulation
Synthetic HRTFs	HRTFs acquired by other computational methods other than acoustic simulation

2.1.1 Localisation Cues

In most cases, humans locate sounds with a combination of different localisation cues, such as Interaural Time Differences (ITD), Interaural Level Differences (ILD), and other spectral cues from the pinnae and torso. The shaping of the spectrum introduced by the pinnae is a particularly important cue for individuals in discerning the location of a sound source, especially at a particular height.

Interaural time difference (ITD)

Interaural Time Difference (ITD) refers to the time or phase difference between the sound arriving in the left and right ears. It plays a vital role in localising sound on the horizontal plane. As human ears are separated by on average 18cm [2], when the sound source is at the side, the sound wave will reach the closer ear first before reaching the one further away. This slight delay varies as a function of the angle. ITD is approximately zero in the median plane since the path lengths from the sound source to both ears are identical, which is the case whether the sound is presented from the front or the back or with any elevation in height. ITD will increase when the sound source moves to the side because the distances between the sound source to the left and right ear are no longer the same. ITD can be estimated by the following equation:

$$ITD = \frac{r(\theta + \sin(\theta))}{c} \tag{2.1}$$

where r is the head radius in meters, θ is the angle of the sound source from the median in radian; c is the speed of sound in meters. The equation is commonly referred to as the Woodworth formula in the literature [1, 36, 37].

The human brain calculates the time delay by looking for the phase difference of the sound source between two ears. Thus, ITD has a frequency limit, which is dependent on the phase ambiguity due to the angle of incidence. The maximum ITD frequency for a source at the side of an average head is approximately 0.7kHz [1, 38, 39]. When the frequency gets higher, ambiguity in ITD will be introduced. Some research suggested

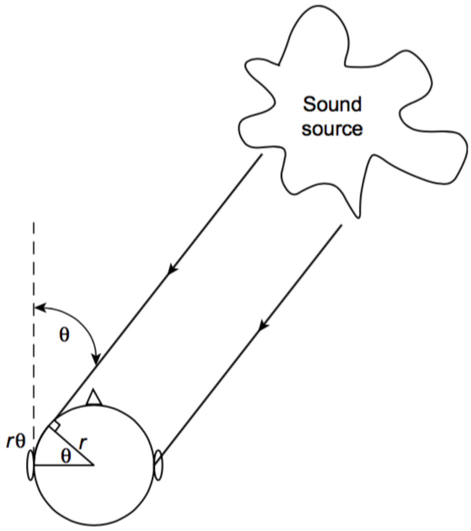


Figure 2.1: Path length around the head on the ITD [2]

that ITDs are frequency dependent, and this could have an effect on localisation [40–42], which the Woodworth formula does not address. On the other hand, Head or source movement may resolve some of the ambiguity [1, 43]. But when the frequency exceeds 1.5kHz, where the wavelength is shorter than the distance between the ears, the phase difference becomes ambiguous, and therefore, ITD is an unreliable cue. In this case, other localisation cues like the Interaural Level Difference (ILD) play a more dominant role in localising sound.

Interaural level difference (ILD)

ILD is another important localisation cue that compares the level differences between the left and right ears. ILD is mainly caused by the shadowing effect of the head, where the sound pressure is higher with the ear nearest to the sound source. Because of this head shadowing effect, ILD does not work well at low frequencies. The level difference only becomes significant once the wavelength is shorter than two-thirds of the head diameter, although some evidence shows the effect could start an octave below at around 600Hz [1, 2, 38]. However, some experiments show that ILD does not act as an effective directional localisation cue until it varies with source direction at about 1.5kHz [2]. In that case, the crossover between ITD and ILD would be completed at about 2.8kHz.

Cone of confusion

ITD and ILD are crucial localisation cues at low and mid-high frequencies. However, if we assume head symmetry, there is no significant difference in ITD and ILD between a front and back position if the source is located at a mirrored angle. Furthermore, a sound source at a similar angle with different elevations can also have the same ITD and ILD. This effect is called the cone of confusion, as shown in Figure 2.2.

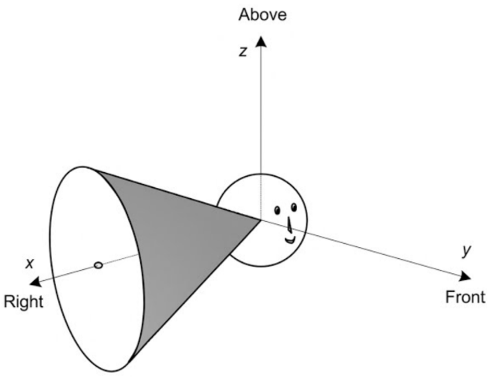


Figure 2.2: Cone of confusion [2]

Any sound sources located on the imaginary cone extending from the human head will have a similar ITD and ILD. However, the cone of confusion rarely gets noticed in daily life. Early in 1940, Wallach hypothesised that head movement might improve sound localisation due to the ITD and ILD changes with head movement [1, 44–46]. Recent studies have shown how head rotation can resolve front-back ambiguity in the horizontal plane. Experiments conducted by Perrett and Noble in 1997, and Rao and Xie in 2005 show that head movement also provides information for vertical movement [1, 47, 48]. This is one reason why current spatial audio applications often rely on head tracking to improve the quality of the localisation effect.

Pinna cues

Other than head tracking, pinnae cues play a critical role in resolving the cone of confusion, particularly when a sound source is static relative to the head. Human beings have a complicated ear pinnae shape, and when sound hits the ear pinnae, it will create a filtering effect before reaching the ear canal. This filtering effect is caused by the delays introduced by the reflections of sounds. When both direct and reflected sounds arrive at the entrance to the ear canal, a comb filter will be created due to the interference effect. This results in directional-dependent peaks and notches at different frequencies that help provide three-dimensional localisation cues. Because the dimensions of the pinna are pretty small, the cues only work at mid to high frequencies, somewhere above 2 to 3 kHz, and are prominent at above 5 to 6 kHz [1, 2, 49, 50]. The pinna shape of human beings is very personal; everyone has different shapes of the pinna. Examples could be found in Chapter 5 Figure 5.2 and 5.3. That is one of the main reasons measuring personal HRTFs plays a critical role in a high-fidelity spatial audio experience [51–54]. Besides getting the personal HRTF by physically measuring the HRIR, some studies suggest getting customised HRTFs by measuring different points of the pinna, or finding the closest match in an HRTF database based on those physical measurements [24, 55–57]. The advantage of this method is that it does not require any professional audio equipment; a picture of the ear pinna can work quite well. But the result may not be as good as a physically recorded HRIR measurement.

There is some research studying the relationship between anthropometric features and HRTFs. The latest work from Stitt presented a parametric pinna model and used the boundary element method (BEM) to simulate the HRTF sets from the model. Then they studied the importance of the anthropometric features related to the generated HRTF sets [58]. As a strong believer in end-to-end machine learning, studying the relationship between specific anthropometric features and HRTFs is beyond the scope of this research. This argument is influenced by the development of facial recognition. Extracting facial features and studying them did not lead to great performance in facial recognition development. Instead, the development of machine learning algorithms with a large amount of training data is the foundation of most facial recognition applications these days. With that being said, there are plenty of HRTF neural network models that use anthropometric measurements as features [59–66]. However, in those cases, understanding the anthropometric measurements is not important, as the model will assign the weight for each anthropometric measurement through training.

Effect of Torso

Some researchers suggest a spectral change below 3kHz due to the scattering and reflection of the torso. Research done by Zhong and Xie further analysed how different clothes change the HRTF. The result shows that different clothes affect high-frequency shoulder reflections above 5kHz, and there are no significant changes at frequencies below 3kHz [1, 24, 67, 68]. Some studies also show no significant differences between tandem and non-tandem head-and-torso movement [69]. Compared to the effect of the head and pinna, the effects of the torso and shoulders on the HRTF are limited but most prominent in the elevation localisation [24].

The effect of the torso changes dynamically when the head angle related to the torso changes in most spatial audio applications. However, most HRTF measurements were made with the measured subject holding in one position. Algazi et al. and Gumerov et al. [70–72] used the “snowman” model to approximate the effect of the torso. Besides approximation, there are substantial studies on the HRTF changes regarding different head-torso angles. Guldenschuh et al. proposed different methods to modify the effect of the torso in the HRTF set [73]. Sontacchi proposed Torso Related Impulse Response (TRIR), which is the torso-relevant parts extracted from the measured HRTFs [74]. Geronazzo proposed Mixed structural modelling that brakes down the HRTFs into the head, torso, and pinna. Apple Inc. has been granted a patent in Spatial audio reproduction based on head-to-torso orientation [75].

2.1.2 Use of HRTFs

HRTFs are commonly used in binaural audio reproduction. By convolving an HRTF pair with any audio signal, people can change their perception of where the sound comes from. This technology is commonly used to reproduce audio in three-dimensional (3D) space.

One reason binaural audio is commonly used in VR, AR, and 360 video is that with the head tracking on modern Head-Mounted Displays (HMDs), the 3D sound field could pan accordingly to counteract the head movement, which hugely improves the realism in the virtual environment. At the same time, head tracking helps reduce front-back confusion, so people can better tolerate the sound reproduced with non-personalised HRTFs. The two technologies help each other to create a better VR experience. Several headphones these days are also embedded with head-tracking functionality, like the ones from manufacturers Apple, Bose, and Sony. Benefiting from these headphones, users can experience some sort of spatial audio and users may find the sound field is widened and externalised, or it can mimic the sound coming from loudspeakers.

Another advantage of binaural audio is its accessibility. There are three main ways to playback audio in 3D: multi-channel loudspeaker arrays, cross-talk cancellation over stereo loudspeaker systems and headphone binaural. A loudspeaker array is costly and challenging to set up. On the other hand, both cross-talk cancellation technology and headphone spatialisation are binaural audio-based technologies which use HRTFs. Both only require a stereo system to reproduce a 3D sound field, which is convenient for most people.

Besides recreating sounds from different directions, HRTFs can sometimes be used reversely. Binaural-based sound localisation is ongoing research, which is used in machine learning quite extensively [76–78]. Some researchers used it on robots to help locate sound by comparing the received sound and HRTFs [79, 80].

2.1.3 Obtaining HRTFs

Currently, there are two common methods to obtain personalised HRTF measurements. One is through physical measurement, where microphones are placed at the ear canal of a subject and the loudspeakers positioned at different angles relative to the head to measure the transfer functions [23, 27]. This measurement process is often tedious for the subject and requires substantial setup and calibration. Another method is to simulate the HRTFs using 3D head or ear models [29]. The simulation process can be computationally expensive and usually requires significant processing time [30, 31]. However, this method is straightforward to set up as long as there is enough computational power. On the other hand, high-resolution head scans for such modelling are required, which again can be cumbersome to achieve.

HRTF measurements

The most straightforward way to obtain HRTFs is through physical measurements, where microphones are placed at the ear canal of a subject, and an excitation signal is played back from a set of loudspeakers positioned at different angles relative to the head [1, 20, 23, 24, 27]. However, high-definition measurements with dense measurement points require a significant amount of time. Participants are usually required to be restrained in an anechoic chamber during the measurements, minimising the movement during the process to obtain the best possible results. Despite the process being time-consuming, tedious and uncomfortable for the participants, this method is easy to scale up once it is properly set up, as long as there are suitable participants willing to participate in the process.

With a similar idea in mind, Zotkin et al. proposed a reciprocity method, which is an inverse method of the conventional measurement procedure [81]. It places micro-loudspeakers in the ear instead of microphones and then uses a spherical mesh structured microphone array to pick up the excitation signal (Figure 2.3. This method has its distinct advantages and disadvantages. One of the greatest advantages lies in its efficiency. In HRTF simulations, including BEM simulations, the reciprocity method is preferred because it enables the calculation of HRTFs for all sound source positions in a single simulation. This streamlining of the process results in a significant time-saving advantage, making it a highly attractive option for simulations. However, this concept is rarely used in physical measurements as it often leads to poor measurement results. This is primarily due to the location of the speakers being close to the eardrums, and therefore the loudness of the measurement signal has to be limited. The lowered signal volume can result in an inadequate signal-to-noise ratio, compromising the quality of the measurements obtained. This trade-off between measurement quality and the necessity of auditory protection makes the reciprocity method a less viable option in physical measurements.

Recently, a VR-based concept has been proposed by Gan et al. This is similar to the conventional method where microphones are placed in the ears, but only one loudspeaker is required for the measurement [69]. Unlike the traditional way, they used the VR HMD to guide people to move their heads for different measurements to get the measurements from different angles.

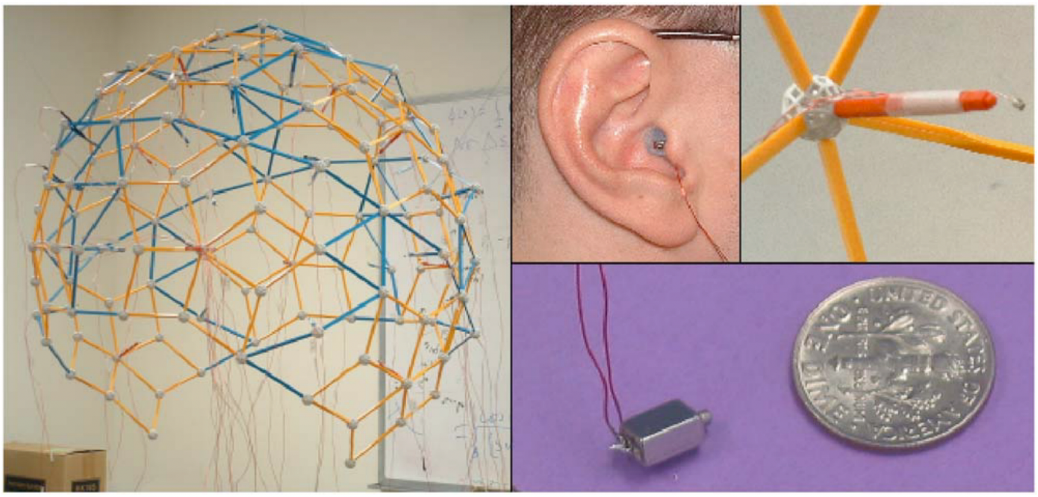


Figure 2.3: HRTFs measurement with the reciprocity method [81]

Most publicly available HRTF databases use the physical measurement method as it is more economical for creating a large number of HRTF sets once it is properly set up. The main issue with these databases is that they are often skewed due to the lack of diversity in participants. Considering the complexity of properly setting up the measurement facility, the diversity of participants is bounded by the geographical location of the setup.

BEM simulation

In addition to physical measurements, HRTFs can be simulated numerically. Compared to other simulation methods like the Finite-Difference Time-Domain (FDTD) [82, 83] and Ultra-Weak Variational Formulation (UWVF) [84, 85], Boundary Element Method

(BEM) simulation is the most common and well-established method for numerical HRTF simulation due to its efficiency. Ziegelwanger et al. created an open-source BEM simulation software, Mesh2HRTF, which became the commonly used software for numerical simulation in the field [86, 87]. Young et al. conducted an acoustic validation study by comparing the HRTFs from acoustic measurement and Mesh2HRTF BEM simulation of a KEMAR head model [30, 88]. They showed that with matched conditions, the just-noticeable perceptual differences are acceptable between the HRTFs. Considering the feasibility of replication studies, this work uses Mesh2HRTF to simulate all the HRTFs.

The most significant advantage of using BEM to simulate HRTFs is its scalability in measurement angles. As discussed in Section 2.1.3, the physical HRTF measurement approach is often time-consuming, tedious and uncomfortable for participants. Due to the fact that any movement from the participants may result in an inaccurate measurement, even with the help of a head tracker or physical support, the experience can be unpleasant if the measurement process takes too long. This is why the number of measurement points of the physically measured HRTF database with humans is often limited, since the more angles measured, the more time will be required. Despite this, there have been a lot of recent advancements in the measurement process, where the measurement is getting quicker and easier. That said, it is still challenging to measure a vast number of source angles relative to the head. BEM simulation does not face the same problem - with a compatible head mesh and enough computational power, an unlimited number of angles of HRTFs can be simulated. BEM simulation is a preferred method to create dense angles of HRTFs on a large scale.

A limitation of using BEM simulation in the past has been the computational cost; depending on the number of measurement angles, range of frequencies and the resolution of the head mesh, a lot of computational power or time to complete one simulation may be required. With the cost of computational power becoming cheaper and large computing clusters becoming more popular, BEM simulation is now a viable method to obtain a large number of HRTFs.

The main challenge these days is to obtain high-quality head models that work for the desired configuration. This is because the resolution of the head and ear model dictates the maximum frequency of the HRTF simulation; in this case, the upper frequency $f_{\max}$ is defined by the maximum length of the edges in the model [31]:

$$f_{\max} = \frac{c}{edge_{\max} \times 6} \tag{2.2}$$

where c is the speed of sound in ms^{-1} and $edge_{\max}$ is the length of the longest edge in meters. At $20°C$, the speed of sound in air is approximately $343ms^{-1}$. In that case, for a simulation that goes up to 20kHz, the maximum length of the edges cannot exceed $2.86mm$. Oftentimes, such a high-resolution model can be challenging and expensive to obtain. A high-resolution 3D scanner like Magnetic Resonance Imaging (MRI), Computed Tomography (CT), or an Artec Space Spider scanner is usually required for such high-resolution scans. Making such scans can be expensive and time-consuming, limiting the number of head models suitable for HRTF BEM simulation.

Obtaining head or ear models by computational methods

To obtain more head models, one way is to manipulate a base mesh by changing the parameters of the mesh, a process known as morpho-acoustics [89, 90]. This method is able to generate HRTFs and can also be used to study the relationship between different parameters and the corresponding HRTFs. The data generated with this method can provide valuable information on how the shape of the ears and heads affect HRTFs, which may lead to better HRTF personalisation methods. The CHEDAR database by Ghorbal et al. created 1253 head meshes by changing a set of 7 parameters of a base mesh and then used Mesh2HRTF to simulate the HRTFs [91]. Stitt et al. created a parametric pinna based on 18 parameters, part of them defined by the CIPIC database [24, 58].

Alternatively, Guezenoc et al. [92] have proposed a statistically based method to create a large dataset of ear shapes and Pinna-Related Transfer Functions (PRTF) generated by random drawings of the ears. 119 three-dimensional left-ear scans were used to create a statistical ear shape model that could generate artificial pinna with Principal Component Analysis (PCA). A synthetic dataset of 1000 ear shapes and matching sets of pinna-related transfer functions (PRTFs) were generated, named WiDESPREaD (Wide Dataset of Ear Shapes and Pinna-Related Transfer Functions Obtained by Random Ear Drawings).

A certain interest has been observed among researchers regarding the investigation of morphable ears and the correlation between ear shapes or anthropomorphic data and HRTFs, as exemplified by studies such as those conducted by Stitt et al. [58], and Pollack et al. [93, 94]. However, the practical implications of understanding the link between ear models and HRTFs for HRTF applications remain somewhat elusive. As an illustration, the study of facial structure has not significantly contributed to today's ML-based facial recognition technologies. Nevertheless, it is important to acknowledge diverse research efforts in this field, even when they may diverge from what is personally believed to be the better direction.

2.1.4 HRTF Data

HRTF format

The most common file format for HRTFs is the Spatially Oriented Format for Acoustics (SOFA). The file format was first suggested by Majdak et al. in 2013 and soon standardised by AES in 2015 [95, 96]. Although it came out later than the one Andreopoulou and Roginska proposed in 2011, the SOFA format is widely adopted due to its ease of use [97]. By using the SOFA format, all HRTF measurements of an individual can be stored in one file amongst other measurement details [98, 99]. It is also capable to store other types of data such as directional room impulse responses [95]. Besides SOFA, some HRTF databases provide the data in other formats. The CIPIC database provides plain text or Matlab format. The SADIE database provides a zipped .wav file of HRIRs.

HRTF databases

A number of HRTF datasets are currently publicly available, created by different research groups. Most of these databases acquired the HRTFs through physical measurement, including ARI [19], ITA [20], RIEC [21], SADIE I [22], SADIE II [23], CIPIC [24], IRCAM LISTEN [25] and the TU Berlin KU100 database [26]. These databases can provide roughly a total of 300 HRTF datasets, depending on the requirement of the application. The biggest drawback of using these databases is the limited number of HRTF sets from each database. At the same time, consolidation of the HRTFs from different databases may not be ideal for some applications due to the different configurations of each database used. For example, the common angles between different databases can be very few, and the measurement quality may vary significantly.

There are HRTF databases that are generated with the measurements of human beings. Brinkmann et al. created the HUTUBS database using BEM simulation with 96 head scans (83 males, 10 females, mean age of 36 years, standard deviation 9 years) [100]. Jin et al. created the SYMARE database that consisted of acoustically measured Head-Related Impulse Responses (HRIRs) for 61 listeners (48 male, 13 female) [90]. Due to the difficulty in obtaining the head scans needed, these HRTF databases face limitations such as a diversity of subjects.

As mentioned in section 2.1.3, some large synthetic datasets have become available by obtaining head or ear models with computational methods, such as the aforementioned CHEDAR [91] and WiDESPREaD [92] databases.

2.1.5 HRTF Interpolation

The idea of HRTF interpolation is to up-sample a sparse HRTF set into a denser one. Considering the challenge of making a dense HRTF measurement set, which was discussed in Section 2.1.3, it allows researchers to reduce the measurement resolution with a sparser angle configuration and up-sample to a dense HRTF set in post-processing.

HRTF interpolation can be done in many different ways, including using inverse-distance weighting and spherical splines in the time or frequency domains or manifold learning [101–105]. It remains unclear which interpolation method provides the best results.

The work in this thesis focuses on Spherical Harmonic (SH) interpolation as it is one of the more elegant methods which has a more standardised procedure and shows promising results [105]. It is considered a global interpolation method that utilises all the HRTFs in an HRTF set.

HRTF sets are recorded in the time domain as HRIRs. These HRIR sets are commonly described in the SH domain due to simplicity and ease of use in Ambisonics for spatial audio reproduction [106]. Since the SH domain describes a continuous spatial representation of the HRIRs, interpolation can be readily achieved. A given HRIR $\mathrm{H}(\theta, \phi)$ can be converted to the spherical harmonic domain at a given spherical harmonic order M by using a re-encoding matrix C with K rows and L columns, where K is the number of SH channels calculated as $K = (M + 1)^2$ and L is the number of HRTF measurements from different angles, where $L \geq K$. The coefficients Y_{mn}^{σ} in the

List of common HRTF databases

HRTF database	Research group	Country	no. of sets	no. of angles	measurement method	distance
CIPIC [24]	Center for Image Processing and Integrated Computing (CIPIC), University of California	U.S.A.	2 mannequins (KEMAR with large and small pinnae), 43 human subjects (27 male, 16 female)	1250	physical measurement	1m
SADIE I [22]	Audio Lab, University of York	U.K.	2 mannequins (KEMAR, KU100), 18 human subjects	1550 on mannequins, 170 on human subjects	physical measurement	1.5m
SADIE II [23]	Audio Lab, University of York	U.K.	2 mannequins (KEMAR, KU100), 31 human subjects (22 male, 5 female, 2 non-binary)	8802 on mannequins, 2818 on human subjects	physical measurement	1.2m
IRCAM Listen [25]	Room Acoustics Team, Institut de Recherche et Coordination Acoustique/ Musique (IRCAM)	France	On paper: 54 human subjects (42 men and 12 women), 3 mannequins (a Neumann KU100, and a Brüel Kjaer type 4100D with and without pinna) On http://recherche.ircam.fr/: 51 subjects	187	physical measurement	1.1m

ARI [107]	The Austrian Academy of Sciences Acoustics Research Institute (ARI)	Austria	More than 200 subjects	1550	physical measurement	1.2m
RIEC [21]	Advanced Acoustic Information Systems Laboratory, Research Institute of Electrical Communication, Tohoku University	Japan	103 human subjects (including 42 male, 11 female), 2 mannequins (SAMRAI, KEMAR)	865	physical measurement	1.5m
Bernschutz [26]	TH Köln University of Applied Sciences Institute of Communication Systems Laboratory for Acoustics, Audio Technology and Audio Signal Processing	Germany	a single KU100 mannequin	Lebedev 2354, Lebedev 2702 or 2 degree Gauss quadrature with 16020 nodes	physical measurement	1m
ITA [20]	Institute of Technical Acoustics, Medical Acoustics Group, RWTH Aachen University	Germany	48 subjects (35 male, 13 female) 45 of them are valid	2304	physical measurement	1m
HUTUBS [108]	Audio Communication Group, Technical University Berlin	Germany	96 subjects (1 repeated FABIAN mannequin, 1 repeated human subject)	440	physical measurement	1.2m

CHASAR [109]	Institute for Hearing Technology and Acoustics, RWTH Aachen University	Germany	26 children (ages 5–10 years old, mean 7.3 years old), consisting of 8 male and 17 females, plus 1 ages 3 years old	2376	physical measurement	1m
SYMARE Acoustic [90]	School of Electrical and Information Engineering, The University of Sydney Department of Electronics, The University of York	Australia U.K.	61 listeners (48 male, 13 female)	393	physical measurement	1m
SYMARE BEM [90]	School of Electrical and Information Engineering, The University of Sydney Department of Electronics, The University of York	Australia U.K.	61 listeners (48 male, 13 female)	393	BEM simulation	1m
CHEDAR [91]	IETR/CentraleSupélec 3D Sound Labs	France	1253 artificial head meshes	2522	BEM simulation	2m, 1m, 50cm, 20cm
WiDESPREaD (PRTF) [92]	FAST Research Team IETR/CentraleSupélec	France	1005 artificial ear meshes based on a dataset of 119 left ear meshes	2562	BEM simulation	2m

Table 2.1: Comparison between different HRTF database

re-encoding matrix C with SH order m and degree n are calculated by

$$Y^{\sigma}_{mn}(\theta,\phi) = \sqrt{(2-\delta_{n,0})\frac{(m-n)!}{(m+n)!}}P_{mn}(\sin\phi) \times \begin{cases} \cos(n\theta), & \text{if } \sigma = +1 \\ \sin(n\theta), & \text{if } \sigma = -1 \end{cases} \tag{2.3}$$

where $\sigma = \pm 1, P_{mn}(\sin\phi)$ are the Legendre functions of order m and degree n, $\delta_{n,0}$ is the Kronecker delta function:

$$\delta_{n,0} \equiv \begin{cases} 1, & \text{for } n = 0 \\ 0, & \text{for } n \neq 0 \end{cases} \tag{2.4}$$

This thesis uses Schmidt Semi-Normalisation (SN3D) in the computation of Y^{σ}_{mn}. For normal SH HRIR use, a mode matching decoding matrix D can be calculated from C with the following pseudo-inverse equation:

$$\mathrm{D} = \mathrm{C}^{-1} = \mathrm{C}^{T}\left(\mathrm{C}\mathrm{C}^{T}\right)^{-1} \tag{2.5}$$

which can be used to inverse the SH HRIR back into the original HRIR [110, 111].

However, for HRTF interpolation, another re-encoding matrix $\hat{\mathrm{C}}$ and decoding matrix $\hat{\mathrm{D}}$ with the desired target angles can be calculated with equation 2.3, 2.4 and 2.5, where $\hat{L}$ is replaced as the number of HRTF measurements to be interpolated from the SH HRIRs whilst K remains the same.

The interpolated HRIRs $\hat{\mathrm{H}}(\theta,\phi)$ then can be calculated with the following equation:

$$\hat{\mathrm{H}} = \hat{\mathrm{D}}(\mathrm{C}(\mathrm{H})) \tag{2.6}$$

As an example of SH interpolation based on an application in Chapter 4, consider a case where we have 6 measurement points and wish to interpolate to 2000. The 6 selected sparse HRIRs from the HRIR database are first converted to SH HRIR (1st order SH which has 4 channels) by using the re-encoding matrix C with 4 rows and 6 columns. Then a decoding matrix $\hat{\mathrm{D}}$ is computed using a different re-encoding matrix $\hat{\mathrm{C}}$ with the same number of rows and a different number of columns based on the desired number of HRIRs to interpolate.

The main issue caused by SH HRTF interpolation is that the interpolated HRTFs are only accurate up to the spatial aliasing frequency f_{lim}, approximated by

$$f_{lim} \approx \frac{\mathrm{c}M}{4r(M+1)\mathrm{sin}(\pi/(2M+2))} \approx \frac{\mathrm{c}M}{2\pi r} \tag{2.7}$$

where c is the speed of sound, approximated as 343 m/s at $20°C$ in air, M is the order of the spherical harmonics, and r is the radius of the human head [112]. For 1st order, the spatial aliasing frequency is around 700Hz.

The spectral distortions will not only affect the timbre, they will also degrade the localisation performance since the important cues for identification of source elevation are changed, as shown in Figure 2.4.

To combat this, dual-band Time Alignment (TA) can be employed in the encoding of the HRTFs [113, 114]. Given that ITD is only effective at low frequencies, high-frequency ITD can be removed when undertaking SH encoding. This is achieved by Time Aligning

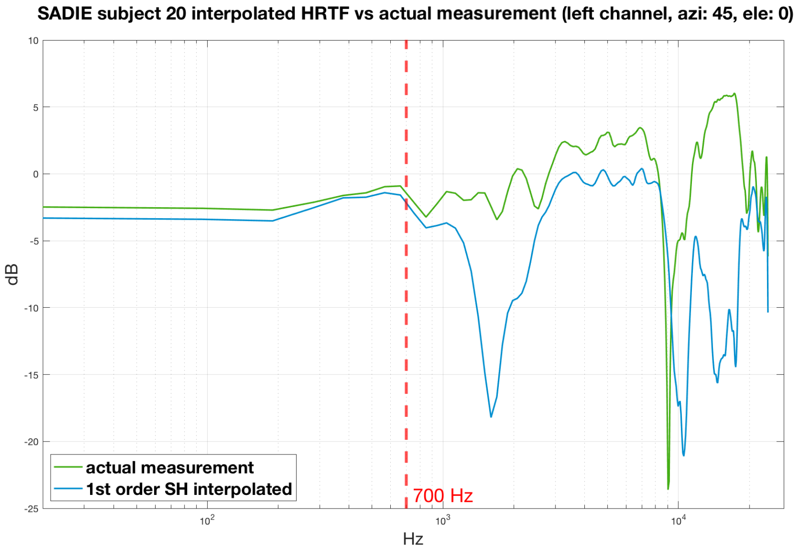


Figure 2.4: Actual ipsilateral HRTF measurement vs SH interpolated HRTF (green: Actual HRTF measurement, blue: SH interpolated HRTF), $M = 1$.

(TA) the HRTFs at high frequencies. By doing this, lower-order SH HRTFs are more effective at preserving high-frequency information, which improves interpolation results. In this study, the crossover frequency was set at 2.5kHz, as suggested in [114].

One of the works presented in this thesis aims to further improve this issue with Machine Learning. Further discussion will be introduced in Chapter 4.

2.1.6 Personalised HRTFs

HRTFs are highly personal, but at the same time, HRTF measurements for every individual are not practical. Thus, a field of research aims to approximate personal HRTFs with simplified methods. There are two main categories for these methods. One is based on simplified head measurements, such as basic head diameter with measuring tape or head or ear scans made with a smartphone camera. The other is based on listening tests or user responses in a spatial audio environment.

Due to the importance of personalising HRTFs, some dedicated research groups have done plenty of work on different ways to achieve this, including different measurement and optimisation methods. A recent paper by Guezenoc et al. [115] reviewed some popular personalisation methods, which are included in the following section. Note that some machine learning-related methods will be discussed in section 2.2.5.

Subjective matching

There is a long history of research in HRTF subjective matching. McMullen et al. used these criteria: externalisation, elevation, and front-back differentiation to estimate a listener's HRTF preference[116]. Iwaya [117] used a Swiss-style tournament listening test to find the best-fitting HRTF. The test was based on the localisation feedback from the participants on the horizontal plane. Andreopoulou1 and Katz, on the other hand, used subjective evaluations for the direction of sound movement on the horizontal and median plane along a fixed radius circle or arc [118]. Based on the result, they found that people's HRTF preferences on the horizontal and median planes may not necessarily be the same. Furthermore, analysing the result by an HRTF data hierarchical clustering of both planes helps to understand the relationship between different HRTFs based on listeners' preferences. Following that paper, they investigated the repeatability of subjective HRTF rating [119]. It shows that the HRTF rating is viable by carefully screening participants through pre-tests. Poirier-Quinot et al. created a VR shooting game by using their own plug-in Anaglyph, to evaluate the player's performance with the best-matched HRTF verse the worst-matched HRTF [4, 5, 120]. They suggested a pre-test method to find out the player's performance with different HRTFs, which showed that there is a significant performance difference between HRTF sets.

Anthropometric parameter matching

The CIPIC database is an example of a database that includes anthropomorphic parameters [24] of head size, shape, shoulder width, torso length etc. Many researchers have used these parameters for HRTF personalisation.

Zotkin et al. proposed a way to find the best match HRTFs, by simplifying and comparing the subject's anthropometric data and finding the closest match in the CIPIC database [121]. Iida et al. used anthropometric data to estimate the notches on the median plane [122]. Later studies were done by Hu et al., who used a simple neural network to predict personalised HRTFs by selected anthropomorphic data [65]. A similar study from Li and Huang used a radial basis function (RBF) neural network to predict individual HRTFs [123]. Both neural network methods show reasonably good results considering the size of the neural network model. In another study, Yao used a neural network to predict listener preference in the CIPIC database [124].

Anthropometric parameters can also be used as an output. A study by He et al. investigated the possibility of estimating anthropometric parameters by features extracted from HRTFs [125]. This study showed the correlation between different anthropometric parameters and HRTF characteristics, which helps to understand the relationship between the two. The most interesting finding from the paper is that ILD yields the best performance for pinnae feature estimation. This contradicts the concept that pinna shape does not contribute much to ILD.

HRTF preference matching by ear shape

Instead of using fixed parameters, an advanced approach is to use the image of the ears to obtain personalised HRTFs. A state-of-the-art approach by Shahid et al. is to separate the task into three sections: image segmentation, feature extraction, and HRTF prediction, each of which is an individual machine learning task that has different options of algorithms shown in Figure 2.5. They then present an algorithm that permutes different combinations of those algorithms [126]. Although the result looks promising in the paper, they do not share their best performance combination, so it is hard to validate their method.

Other attempts try to estimate the ear's 3D model by photos [127], and then use the boundary element method (BEM) to simulate the HRTFs [1, 34, 128]. A similar

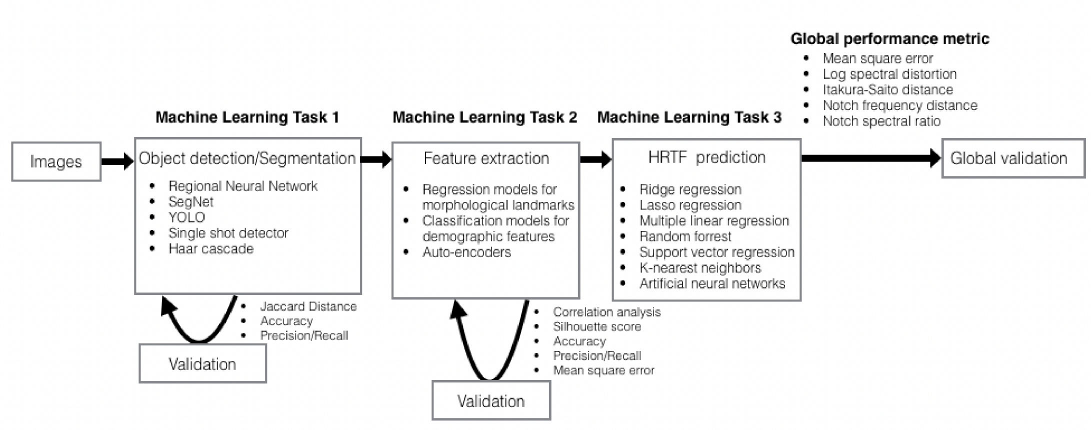


Figure 2.5: The HRTF preference matching pipeline proposed by Shahid et al. [126]

approach uses the LeapMotion controller and the Kinect projector to model the ear pinna in 3D instead of a camera [129]. This seems to perform better than Shahid et al.'s method by comparing the Spectral Distortion (SD) results in the paper. Note that SD is the only attribute in common with both papers but may not be the best way to validate HRTFs, as it is just a subjective evaluation by comparing the predicted HRTFs with the target HRTFs. A recent attempt by Geon Woo Lee shows that it may be possible to use the bottleneck features of CNN auto-encoder for personalised HRTFs [130]. However, the paper only shows that restoring ear images with bottleneck features is possible without making any predictions or adjustments on HRTFs. It shows the dimension reduction of an ear image can be a valuable feature for HRTF estimation, which is also used in the method of Shahid et al.

Personalised HRTF in VR games

Due to the recent growth in VR, researchers have started investigating the importance of personalising HRTFs in VR environments.

Huttunen et al. shared some preliminary ideas about personalised HRTFs on the localisation speed and accuracy in a VR environment [131]. Jenny1 et al. built a Unity plug-in SOFAlizer that allows users to switch HRTFs on-the-fly [132]. Unlike most spatial audio plug-ins for VR that use Ambisonics [133–135], this plug-in uses a direct convolution algorithm instead. This may not be the best for spatial audio reproduction, but it may be helpful for HRTF research as it uses one HRTF at a time. A similar framework HOBA-VR proposed by Michele Geronazzo et al. is a more comprehensive Unity plug-in with headphone equalisation [136]. Unlike SOFAlizer, which uses the nearest HRTF, SpatialPanner interpolates impulse responses from the three HRIRs in the desired region, potentially producing a more accurate and smooth playback result.

Poirier-Quinot et al. built a VR shooting game to evaluate the performance difference between different HRTF set [4, 5, 120]. It shows that player performance can have a significant difference between the best-matched HRTF and the worst-matched HRTF.

HRTF Measurement with XR Headset

Lately, Rudzki et al. proposed an HRTF measurement system that utilises an Extended Reality (XR) headset [137]. By using a single loudspeaker and placing a pair of microphones in the ear canals, HRTF measurements can be made by using the Extended Reality (XR) headset to guide the participant into different positions relative to the loudspeaker. The virtual guiding interface in the Extended Reality (XR) headset will ask the participant to move into the desired position and orient their head according to the visual cues displayed in the headset.

Considering the size of the headset, which is Quest 2 in the paper, HRTF measurements may be affected by the headset. Thus, a method of delivering direction-dependent HRTF correction filters based on the set of KEMAR measurements is proposed. First, HRTF measurements are made on a KEMAR manikin with and without the headset. Then, both measurements are interpolated at a dense regular layout, e.g. 4334-pt Lebedev grid. After that, the time-of-arrival difference and magnitude spectrum difference are calculated. By calculating the time-of-arrival correction and a minimum phase inverse filter for the magnitude difference, the HRTF measurement of the human subject can be corrected using the calculated correction data of the nearest point from the set.

This proposed HRTF measurement method has the flexibility to be utilised to measure sparse HRTFs with any layout. However, the required measurement time will increase linearly with the number of measurement points. Compared to a typical physical measurement setup, this method is much easier to set up and requires significantly less equipment. Also, the procedure does not require an anechoic room due to its post-processing workflow mentioned in the paper. This makes obtaining human HRTF measurements a lot easier in the future.

HRTF personalisation with PCA

PCA is a commonly used technique for data dimensional reduction and analysis [138–141]. Fink and Ray proposed a method to personalise HRTFs by tuning the weights of the principal components [142, 143]. They picked five principal components with the highest standard deviations at 0-degree azimuth and elevation and then asked the participants to tune their weights. After three rounds, the tuned HRTF became very similar to the measured personal HRTF, and the listening test result showed significant improvement with the tuned HRTFs, especially in front-back confusion.

User-to-system HRTF adaptation

In contrast to HRTF personalisation, another concept involves training the user's adaptation to the chosen HRTFs. The idea of user-to-system HRTF adaptation has been extensively explored by Picinali et al. [144]. Their research introduces an innovative strategy that combines both user-to-system and system-to-user adaptation methods. This approach eases the process of HRTF personalisation, as it involves training users to adapt to the best-matched HRTFs. At present, this innovative proposal might be considered the most effective solution for HRTF personalisation.

2.2 Machine Learning

Since the dawn of the modern computer era, it has been a dream to have a machine that can alter its own instruction based on data or experience. This idea led to a substantial amount of work which later became the field of ML, which has gained success in many different fields. One example is the recent developments in image processing, including noise reduction in images, image inpainting, colourising old photos or videos and neural style transfer [8–16]. The main advancement in machine learning these days is the ability to process data in a relatively raw form without excessive feature extraction. The three biggest drivers of this advancement are the development of ML algorithms, data availability and computational scale. Whilst this thesis is trying to solve the data availability issue. It is important to know a little bit about the ML algorithms that the data serves. The following sections will introduce different categories of ML and different types of ML algorithms, from the less data-driven traditional models to the modern deep learning model that often requires a lot of data to train.

2.2.1 Supervised, unsupervised and reinforcement learning

The most common way to categorise machine learning models is based on their goal and reward mechanism to separate them into supervised, unsupervised and reinforcement learning. The categorisation method could be a bit old-fashioned for modern machine learning models. Some modern models can be hard to categorise due to their architectures, such as Generative adversarial networks (GAN) and Auto-Encoder (AE). Based on this categorisation method, some researchers proposed categorising them as Semi-supervised learning and Self-supervised learning. Nevertheless, this conventional categorisation is most useful in providing an understanding of the principle of how machine learning algorithms work.

Supervised learning

Supervised learning is the most common form of machine learning. It requires both input and the target output, where the target output is sometimes called labels. The algorithm will then try to find out the relationship between the input and output data. It is called supervised learning because the algorithm will try to predict the target output based on different ways to process the input data. The target output data acts as a supervisor to check how far apart the estimated result and the actual output data are. Based on the differences, the algorithm will make some adjustments to improve the prediction results. In short, a supervised learning algorithm often tries to minimise the difference between prediction based on the input and target output.

Unsupervised learning

Unsupervised learning aims to find out the underlying structures and relationships between data. Unsupervised learning requires only the input data with no corresponding output target. This is very useful for data analysis and feature extraction. Unsupervised learning can also be separated into two categories, clustering and association.

Clustering: the goal of clustering is to separate data into groups based on their similarities. Hierarchical clustering is commonly used to group similar HRTFs based on their attributes or a user's performance. [97, 118, 145]

Association: Association is to analyse the relationship between data and extract the rules that could explain the relationship. This is commonly used in recommender systems.

Reinforcement learning

Unlike supervised and unsupervised learning which learns from a fixed set of data, reinforcement learning engages with the outside environment, which can be another machine, human, robot etc. By interacting with another system, it tries to learn the actions that could maximise the reward or minimise the punishment. The most famous application is on Go, where AplaGO from Deepmind beat South Korean professional Go player Sedol Lee on Go in 2016 [146]. Besides playing GO, reinforcement learning has also shown good results in ATARI games [147], autonomous driving [148, 149] and robotics [150, 151]).

2.2.2 Symbolic vs Connectionist

Symbolic ML often refers to the ML algorithm that highly relies on prior knowledge and feature extraction. These algorithms usually require less data and computational power. The connectionist ML algorithm refers to the ones that use connected neurons, which will be explained more in the Neural Network section. There are different types of neurons with different hyperparameters that could be changed. By connecting different neurons, different architectures can be formed. The most significant idea behind the connectionist ML algorithm is to use the data in its rawest form if possible. Instead of using the prior knowledge to improve the algorithm's performance, the connectionist ML algorithm often does so by changing the structural or hyperparameters of the architectures based on performance.

Many of the arguments between symbolic ML and connectionist ML are from the 80s, the importance of this discussion is not just about the algorithm but also the philosophical differences between the two approaches. Symbolic ML has been more popular in the past as it shows promising results in some problem domains. However, it also fails to solve well some more complex problems, such as facial recognition and Neutral Language Processing (NLP) problems. Symbolic ML was more popular in the past due to the limited size of digital data and computational power. With the epic growth in data size in the Big Data era and computational power, connectionist ML gained popularity in the late 90s and early 2000s. The discussion between Symbolic ML and connectionist ML is more relaxed these days. Although most of the ML work is done in a connectionist ML principle, it is not uncommon to add some prior knowledge to improve the results. The work mentioned in this thesis leans toward the connectionist ML idea. The goal is to study how far the data-driven approach can go without injecting prior knowledge and what challenges it may face.

2.2.3 Traditional machine learning models

Traditional machine learning, with the Symbolic ML philosophy in mind, is usually based on statistical models. Compared to some recent models, these models typically require less computational power and less data to train. Although most of them are not used in this work, it is still worth introducing some of these models as they shed some insight into the concept of machine learning. Some have significant influences on modern machine learning model design.

Linear regression

Linear regression is one of the most basic machine learning algorithms. The idea is to find the linear relationship between input and output data. The concept of a simple linear regression with one input feature is to define a straight line that could minimise the difference (error) between input and output data, then use that line to predict future results based on the input. There is an extended version of linear regression, sometimes called polynomial regression. Instead of using a straight line, polynomial regression used polynomials to create a curved line. It can find the non-linear relationship between input and output data to a certain extent. By carefully selecting the number of polynomials, it can reduce bias as it has a better chance of describing the training data with minimal error. But when the number of polynomials is more than optimal, it is likely to overfit the data, ending up with incorrect predictions in applications. It is worth noting that linear regression is the cornerstone of modern neural networks, which will be introduced in 2.2.3. A comparison between linear regression and polynomial regression is shown in Figure 2.6.

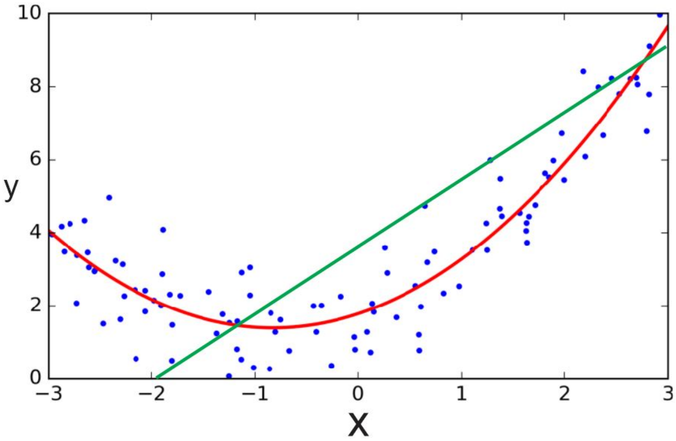


Figure 2.6: Linear regression vs polynomial regression [152]

Logistic regression

Logistic regression is similar to the linear regression algorithm but is used on classification problems. Instead of predicting numbers, it predicts the possibility of binary states by adding a sigmoid function. The sigmoid function converts the linear regression line into an s-curve from zero to one with a steep change so that the algorithm will encourage the resulting output in a binary form. The idea is shown in Figure 2.7. A decision boundary will separate the data into the desired classes. It is worth noting that the concept of adding a function after linear regression is commonly used in neural networks. These functions are often referred to as activation functions. The Sigmoid function was a reasonable choice of activation function during the early days of neural network development. It has fallen out of fashion due to its performance and mathematical complexity compared to other modern options. More on activation functions will be discussed in later sections.

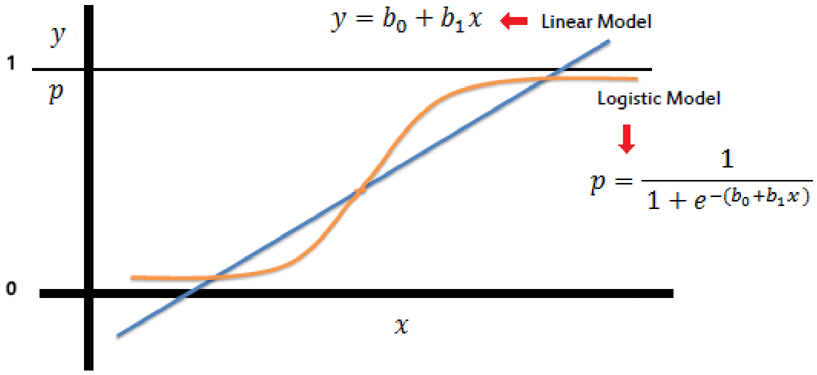


Figure 2.7: Linear regression vs logistic regression [153]

Support vector machine (SVM)

Traditional logistic regression tends to overfit and easily get affected by outlined data. To improve the problem, the SVM was introduced as a large-margin binary classifier that could make more robust predictions. Instead of separating the data with a decision boundary based on the prediction's accuracy, SVM aims to maximise the margin between the decision boundary and the data from different classes, demonstrated in Figure 2.8. SVM was popular in the late 90s as it often outperforms logistic regression when the data is “noisy” because it will not be affected by the outlined data.

SVM is a linear classifier, which means it can only separate the input data linearly. However, some of the input data may be in a distribution that does not have a clear linear boundary. To accommodate this, the kernel method is often used alongside SVM. The idea of the kernel method is to transform the input data into a higher dimension so that the data can be a linearly separable distribution. In execution, to simplify the process, the kernel method does not explicitly transform the input data. Instead, it only uses the pair-wise dot product in the feature space. Thus, it is sometimes called a Kernal trick instead. The concept is shown in Figure 2.9. Worth noting that different kernel functions work well with different training data. However, as this work did not use any kind of SVM, discussing different SVM kernel functions is out of the scope of

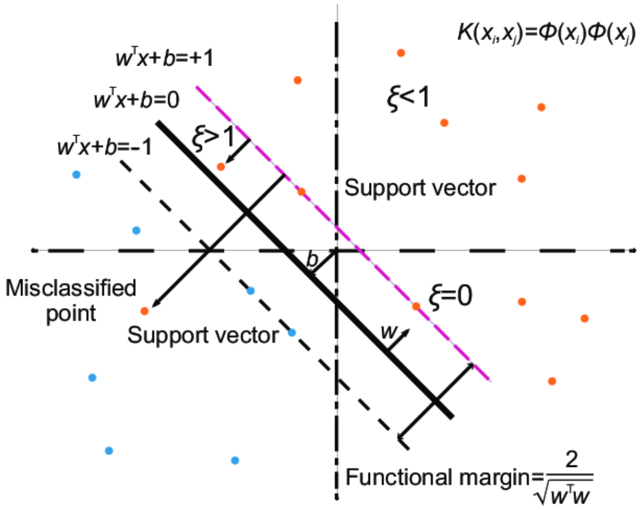


Figure 2.8: Concept of a Support Vector Machine [77]

this thesis.

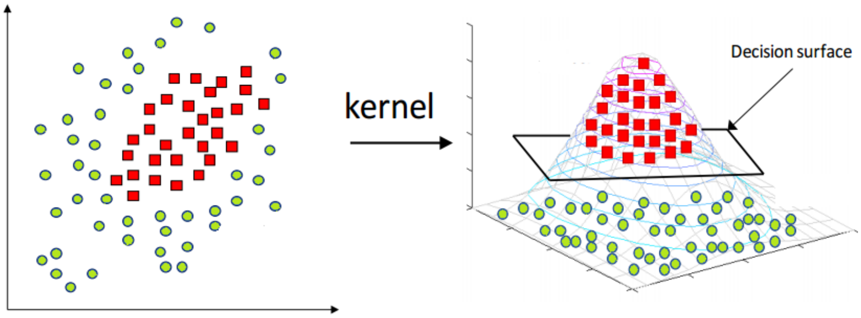


Figure 2.9: The idea of the kernel trick [154]

The use of SVM with the kernel method was popular in the early 90s as it performs well in different applications and can handle raw data relatively well. It marks the turning point from Symbolic ML to Connectionist ML. A basic Neural Network follows the same principle, where the input data is redistributed with some transformation functions and then separated the transformed data with a linear boundary.

The concept of a support vector machine can be used on regression problems, often called support vector regression. Instead of maximising the margin, it tries to find a line or surface that could minimise the error by using the kernel trick.

K-means clustering

K-means clustering is an example of an unsupervised algorithm. This algorithm works by randomly initialising the number of K centroids and then calculating the mean of the data closest to each centroid. The calculated mean will then be the new centroid, and the process is repeated until convergent. K stands for the number of clusters the model targets to find. Figure 2.10 shows the concept of the process.

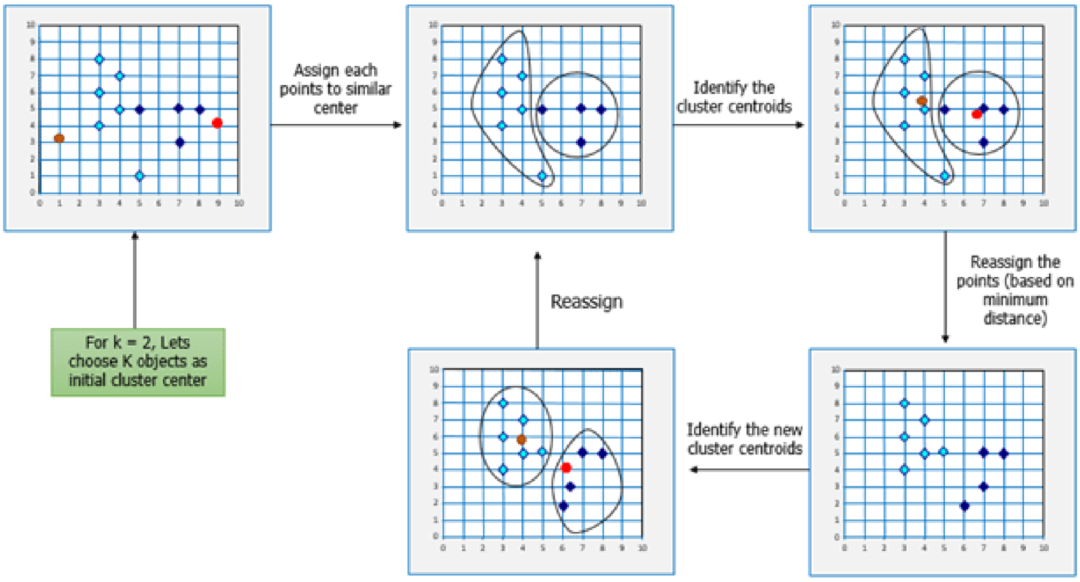

Assign each points to similar center
Identify the cluster centroids
Reassign the points (based on minimum distance)
Identify the new cluster centroids
Reassign
For k = 2, Lets choose K objects as initial cluster center

Figure 2.10: The concept of K-means clustering [155]

K-mean clustering is a simple yet effective clustering algorithm, but sometimes it is hard to define the number K as there is a trade-off between the number of K and how meaningful the clusters are. One way to decide the number is to try with different K values and then find the “elbow” or “knee” point, which represents the best point of the trade-off. Satopaa et al. proposed a way to find the “elbow” or “knee” point robustly [156, 157]. Some examples are shown in Figure 2.11. This concept can be applied to other scenarios when finding the best trade-off is required, like the numbers of Principal Components (PCs) of a Principal Component Analysis (PCA) model in Chapter 6.

Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a statistical tool that can linearly project the input data into many Principal Components (PCs) based on the variance of the data. The lower the PC number captures, the more variance of the data, which represents a combination of the key features of the data. This means that a limited amount of PCs can capture a majority of the variance of the input data. Thus, PCA is commonly used in data analysis and dimensionality reduction. By analysing and comparing the PC, some trends can be found, which may be hard to see in the raw data. Since it does not require any label data, it can also be considered an unsupervised algorithm.

PCA can be used to reconstruct the input data by using a small number of PC as a bottleneck. Since the number of PC is smaller than the input data, the model needs to remove the less relevant information based on the variance of the data. Thus, a lot of the noise in the data will be removed. The most common use case for this type of model is denoising. Alternatively, with such models, by changing the value of the PCs,

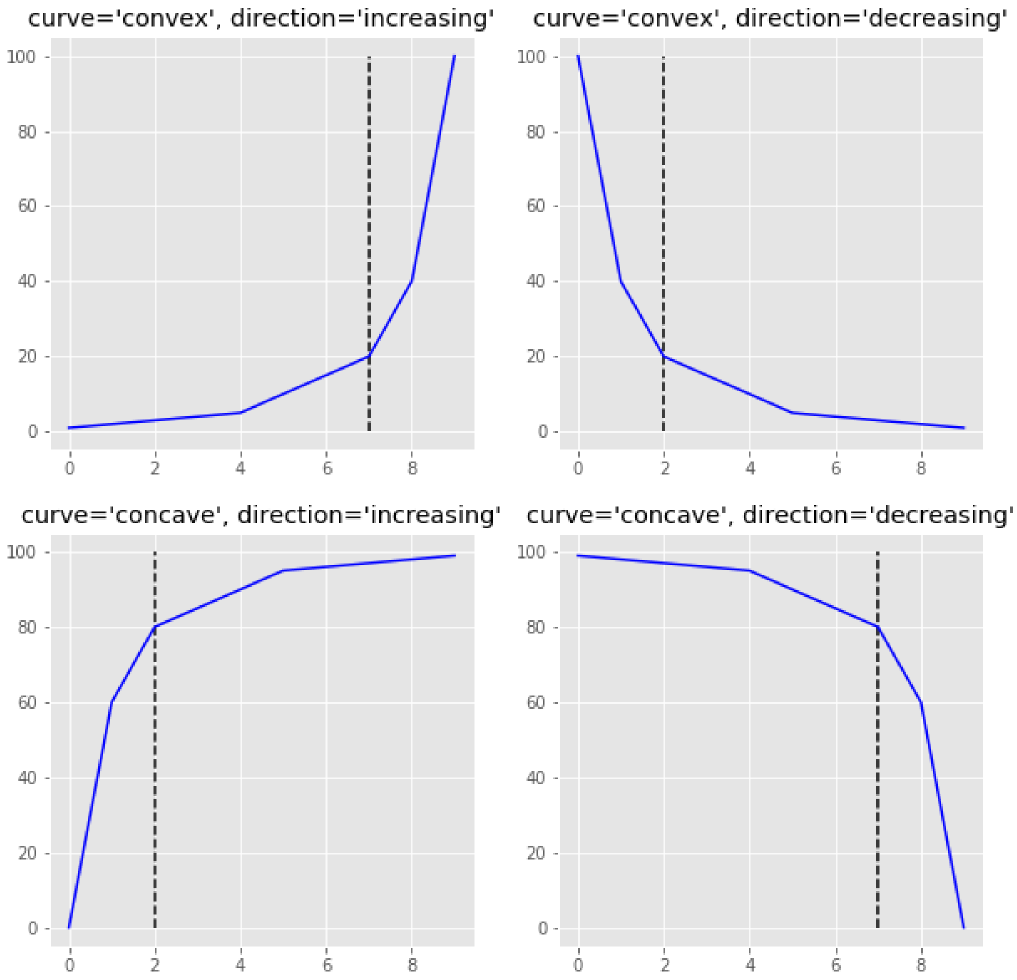


Figure 2.11: Some examples on the "elbow" and "knee" point[157]

the model can synthetically create new data by interpolating the input data with a relatively small number of variables.

Collaborative filtering

Collaborative filtering is an association algorithm that is commonly used as a recommender system. It usually uses matrix factorisation to make predictions [158, 159]. Instead of using a backward propagation method to train the model, collaborative filtering tries to analyse the training data and extract features from it. Consider a movie recommender system as an example - the concept here is to assume people have a different preference to certain genres such as romance, action or sci-fi. Although everyone will have slightly different tastes in movies, diverse groups of people will generally like some types of features and dislike others. By analysing enough people's preferences, the model can make reasonable guesses on what features are in the movie and which type of people may like it.

Neural networks

The concept of neural networks has been around since the dawn of machine learning research. However, due to limited data and computation power in the past, it is less useful than other ML methods (this refers back to the Symbolic vs. Connectionist discussion in Section 2.2.2). A traditional Neural network is usually used in supervised learning, for both classification and regression problems. The concept of a conventional neural network can be considered as a group of logistic regressions that fully connect to each other. By doing this, it can transform the input data into a different distribution and then separate it with a linear function. The most common way to train a neural network is through backpropagation. The idea is that a neural network first makes a prediction, which is sometimes called a forward pass. After that, the prediction will compare with the target output. Based on the differences, backpropagation will perform something called Gradient Descent to adjust the weights in the neural network to improve the prediction. How aggressively the weights are adjusted every time backpropagation is performed is defined by the optimisation algorithm and the learning rate (or a range of learning rates). Changing the optimisation algorithm and its training rate will change the training speed and performance. In theory, a higher learning rate trains faster, but the trade-off is that it may not be able to find the optimal weight of the neural network as it will overshoot the minima in Gradient Descent. However, if the learning rate is too low, the Gradient Descent could get stuck in a local minimum point and fail to find a global minimum point that provides the best performance in theory. Different optimisation algorithms have different rules for gradient descent, each performing better than the others in different tasks [160–164]. In most cases, some trial and error are needed to experiment with various algorithms of optimisation and learning rates.

Figure 2.12 shows an example of a neural network model. It is defined by a few things: input size (i.e. input data size), number of layers, number of neurons in each layer and output size (target data size). Deciding the number of layers and the number of neurons in each layer is a complicated yet essential task. The general idea is that the number of layers is usually decided by the complexity of the problem and the quantity of training data. More layers can solve more complex problems, but also require lots of data to train with and take more time. The general consideration for the number of neurons is based on the input size, output size, the number of training data and also the algorithm of the model. More neurons may increase the model's accuracy, but there is also a chance it may overfit. These guidelines work better with traditional simple neural networks. The modern ones with a more complex architecture often require trial and error to find the proper setup.

Due to the potential complexity of the neural network, it is prone to cause overfitting in the result. The best practice is to add more training data if possible. For this reason,

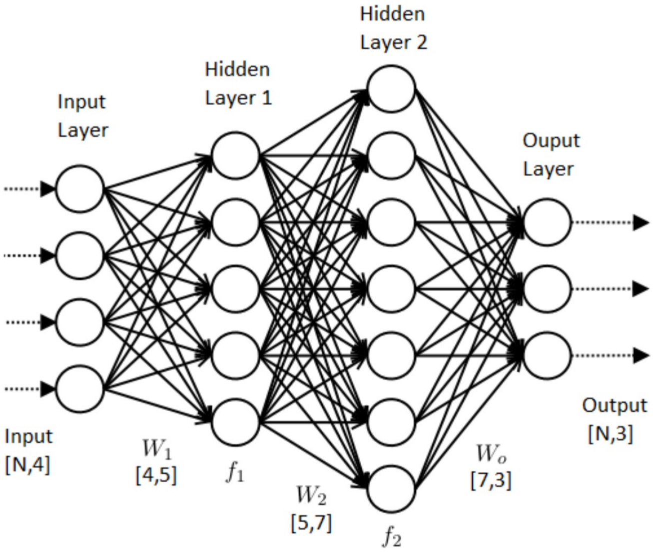


Figure 2.12: A simple neural network [165]

neural networks are often considered a "data-hungry" algorithm. With that being said, when data is limited, using regularisation or dropouts of some neurons during training may be able to rectify some of the problems. The goal of designing a neural network is to find the balance between training results and test results by changing different variables and hyperparameters.

One of the most significant breakthroughs that helped neural networks gain popularity is the use of a rectified linear unit (ReLU) instead of a sigmoid function as an activation function. This improves the training speed and handles the problem of vanishing gradient due to the characteristic of the function. Despite the different variations of ReLU, the idea is to remove the negative value from each neuron, so the output will be zero when the output of the neurons is negative, and the positive results remain unchanged. Due to its simple properties, it is much less computationally expensive than sigmoid or tanh (a scaled sigmoid function), which involves more complex mathematical operations like exponential. Figure 2.13 shows a few examples of activation functions. Leaky ReLu and ELU share the same idea of ReLU, but can reduce vanishing gradient issues as the negative gradients will not be zero. This keeps the gradient “alive” in backpropagation.

A modern neural network has become more complicated, with many more options in hyperparameters and types of neurons. However, the concept of a neural network remains unchanged: connecting different neurons as building blocks to form a sophisticated algorithm that can work well with a specific problem.

2.2.4 Modern machine learning models

Since the high-profile breakthroughs in the early 2000s, neural networks have taken over the machine learning field [146, 167–169]. Most of the modern ML models use some form of neural network. This is due to their limitless possibility for creating different

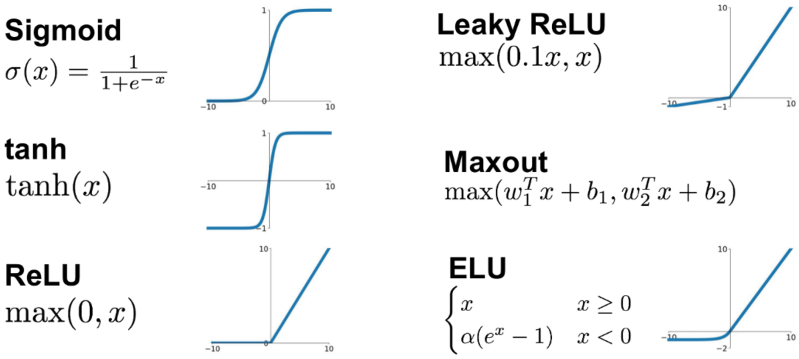


Figure 2.13: comparing different activation functions [166]

types of neurons and architectures. It enables these models to extract higher-level features from the input data, leading to more accurate output when well-designed and trained. On the contrary, due to their complexity and flexibility, they often require lots of training data, and they are hard to find the optimal setup for each model.

In most cases, designing a machine learning system aims to find the balance between bias and variance. High bias often means underfitting; the model can not extract or identify the high-level features to make good predictions, even during training. High variance is the opposite. The model is complex enough, but it fails to have a general understanding of the data. It overfits the training data, thus resulting in poor performance in validation and testing. The modern machine learning model provides the flexibility and scalability to combat underfitting, but it requires enough data to balance out the overfitting issue. Despite some techniques like regularisation or dropout methods that can help reduce the training data required, the best practice is to train with more data if possible. This is why modern machine learning models are considered more "data hungry" compared with some traditional or symbolic approaches. This chapter will focus on architectures that have some major impact on the modern ML era.

Convolution neural network (CNN)

One of the most important breakthroughs in machine learning is the use of CNN in computer vision. CNN introduced the concept of using convolution in a Neural Network. It works well when the input data are coherent between one input and the ones next to it. Instead of neurons, CNN uses convolutional layers. A convolutional layer convolves the input for the layer with some hyperparameters and passes the output to the next layer. The key hyperparameters are the number of channels in the input matrix, the number of channels produced by the convolution and the size of the convolving kernel. These hyperparameters transform the input data to extract and consolidate the low-level features. Like a neural network, convolutional layers can be stacked on each other to create a more complex model. By creating a more complex model, more abstract features can be learnt. In a traditional setting, each convolutional layer will follow by a pooling layer. This is because each output of convolutional layers represents a precise position of features in the input data, which means it can not encounter any positional change in the input, no matter how minor it is. A pooling layer is a layer that downsamples the output from a convolutional layer by applying pooling to the convolved output data. Downsampling the data and creating a low-resolution version of the data removes the fine detail and only keeps the significant features in the data. The majority of the CNN that use pooling layers use max pooling due to simplicity. By setting the kernel size, the max pooling layer will pool the max value of each kernel in the convolved data. In some modern CNN architectures, pooling layers are replaced by stride convolutions. Stride convolution is a convolution layer that downsamples the input data without the need for a pooling layer. By defining the stride of the convolution, instead of moving the kernel one sample at a time across the input data, the kernel can move across the input data based on the stride number.

In a typical CNN, the output of the last convolutional layer often feeds into a neural network, usually called a fully connected layer in a CNN model. The idea is to extract the features and downsample the input data before feeding them into the fully connected layer. This is useful when the input data are coherent between one input and those next to it, as a neural network often fails to address the relationship between the nearby data.

CNN is excellent in extracting features, especially with image [170], and it is commonly used in the field of music information retrieval (MRI) for analysing spectrograms [171]. Furthermore, Kon and Koike used CNN to estimate acoustic reverberation characteristics from two-dimensional images [172]. Thuillier et al. used CNN on binauralised sound sources to perform simple location classification [173].

Recurrent neural networks (RNN)

Recurrent neural networks are designed for sequential data like Natural Language Processing (NLP), financial data and audio. An RNN uses sequential information by considering previous results for the next prediction so that it can memorise many previous inputs due to its internal memory.

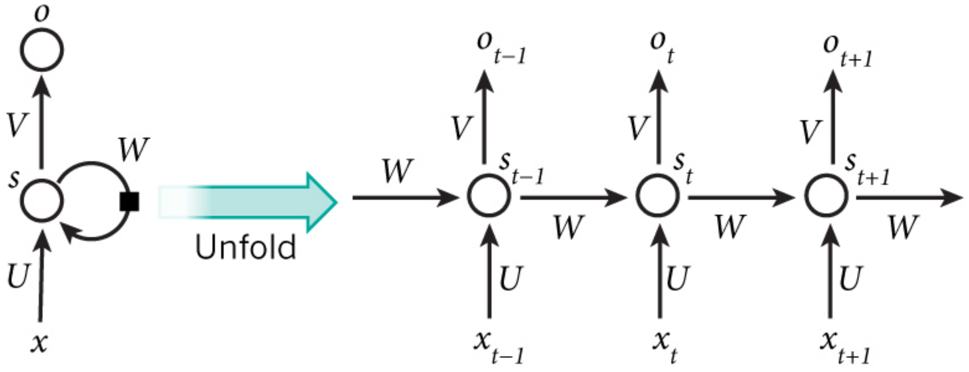


Figure 2.14: Example of a recurrent neural network [174]

A traditional RNN combines the input data with the previous output data of the sequential data. It forms a loop that feeds the result back into the neuron alongside the new input data, as shown in Figure 2.14. The weakness of RNN is that the long-term information often gets lost after a few loops. As early output data needs to sequentially loop many times before getting to the present processing cell, the impact of long-term information is small. This is called the Vanishing Gradient problem. This is a huge problem in some sequential data tasks, like Natural language Processing (NLP), as it often needs some long-term memory to perform well. For example, when a model is trying to generate a paragraph of text, an RNN cannot ”remember” the information it generated a few sentences ago. Thus, it can not generate a coherent paragraph that makes sense to the human reader.

Long short-term memory units (LSTMs) are a type of NN that allows the previously generated output to bypass some units and add them to the input data. As the long-term information no longer needs to be processed in every loop, it can ”remember” the output from longer time steps prior to the current one. For a long time, this has been the most popular method for NLP. However, RNN and LSTM have fallen out of fashion in recent years as these architectures have two main weaknesses. Firstly, the Vanishing Gradient is not fully solved even with LSTM. Secondly, it is almost impossible to apply transform learning in such models. This is a big problem as it means people must train a model from scratch for each task, even if they are similar.

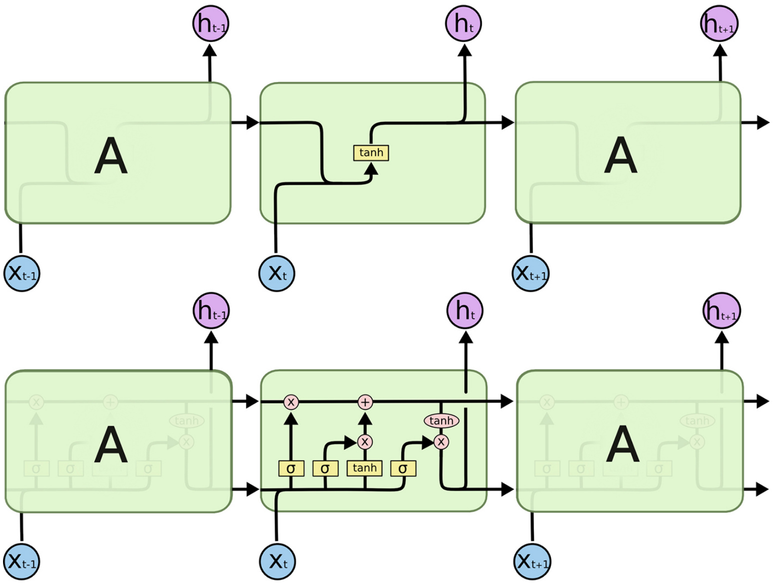


Figure 2.15: A comparison between RNN and LSTM [175]

In 2016, Van den Oord et al. from Deepmind introduced Wavenet, a CNN for text-to-speech problems [176]. This method was state-of-the-art for text-to-speech and sequential data for a short period as it drastically outperformed RNN and LSTM. Instead of a recurrent unit, it uses a concept called causal convolutions to take advantage of the sequential data. However, this method soon fell out of fashion when the Transformer model was introduced in 2017.

Transformer

In 2017, Vaswani et al. from Google Brain published a paper named 'Attention is all you need' [177]. It introduced a new sequential model called the transformer. A year later, a paper from Devlin et al. named 'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding' showcased the transformers model's capability. Since then, transformers have been the most popular concept for NLP and some other sequential ML tasks. The transformers model is quite complicated as a combination of advanced ML ideas, and it often requires a lot of training data [178]. The recent demonstration of Chat GPT gives people a glimpse of what transformer is capable of when combined with Reinforcement Learning with Human Feedback (RLHF) [179–181]. However, since this thesis work did not use anything related to the transformers model, explaining the mechanism of the transformers is out of the scope of this review.

Auto-encoder

Auto-encoder is a model commonly used for dimensional reduction. A simple auto-encoder can be separated into two parts, an encoder and a decoder. They are connected with a few bottleneck features. An encoder tries to reduce the data dimensions into some features with a neural network, sometimes referred to as the latent space. Then the decoder attempts to decode the low-dimension features back into the original input. A diagram of a simple auto-encoder is shown in Figure 2.17. The model aims to match the decoded data with the input data. This is often used as a dimension reduction method similar to PCA. After the Auto-encoder is trained, different outputs can be generated by changing the variables in the latent space.

A convolutional Auto-Encoder (CAE) and a Denoising Auto-Encoder (DAE) are developed based on the same principle. CAE uses convolutional layers instead of neurons. In some cases, it does not require any fully connected layers. A DAE is basically an Auto-Encoder, but the input and output are no longer the same. Instead, the input-output pair are very similar and closely related to each other but different, e.g. noisy images and clear images. Instead of having a low dimensional latent space, it often uses a latent space that is higher than the input and output, so it can extrapolate some features in the encoder and then predict the output. Image inpainting and noise reduction often use CAE or DAE by slightly mismatching input and output, in which the input is the data with noise and the output is the clean data. This forces the model to learn the meaningful representation of the image. Similar techniques have been used in speech enhancement [182].

Variational Auto-encoder (VAE)

A well-trained auto-encoder will create a latent space of the training data. In theory, by changing the latent variables in the latent space, the decoder could generate new output by interpolating the space between the latent variables of the training data. However, the latent space generated by the encoder is not regularised, which means the distribution of the latent space may not have a normal distribution to interpolate new data reliably. This means that the latent space lacks continuity, where the latent variables that are close to each other may not produce similar results. Another issue

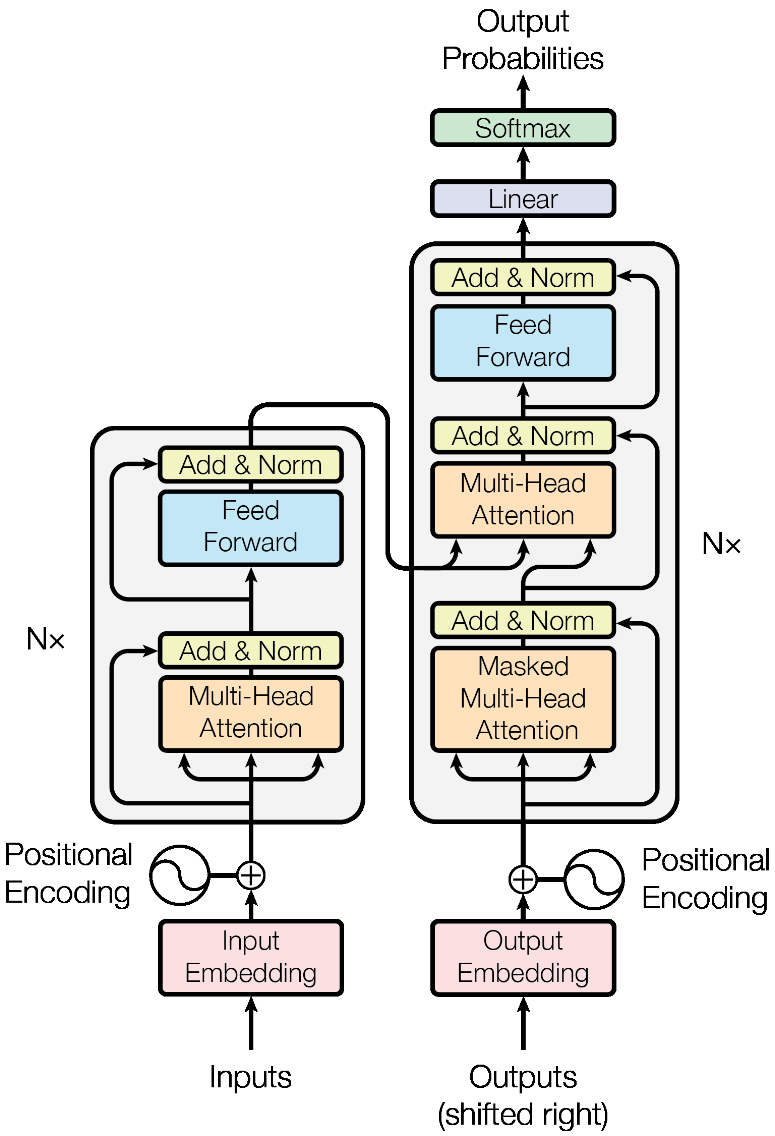


Figure 2.16: A brief idea of a transformer model [177]

with the latent space of an auto-encoder is the completeness of the output from a random latent variable is not guaranteed. This means a randomly sampled point in the latent space may not lead to a meaningful output when decoded. This is similar to overfitting an ML model, where it is not able to generate meaningful new data when changing the latent variables. A special type of auto-encoder was proposed for generative purposes called Variational Auto-Encoder (VAE) [183]. VAE is a generative auto-encoder which could generate new results after training by regularising the latent space. Unlike a typical auto-encoder, a VAE aims to contain the latent variables in a normal distribution. Thus once it is properly trained, by tuning the latent variables in the latent space, new data can be generated by interpolating the dimensionally reduced trained data. In other words, the latent space of VAE is more equally distributed than

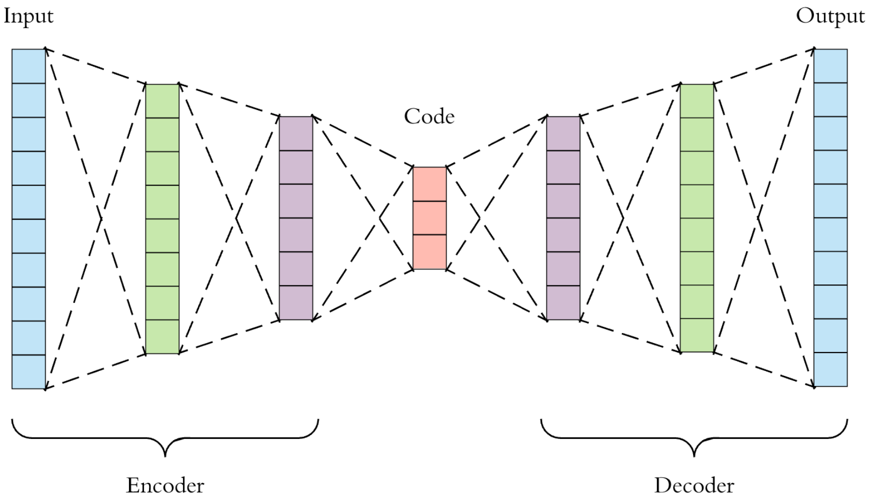


Figure 2.17: The structure of a simple auto-encoder

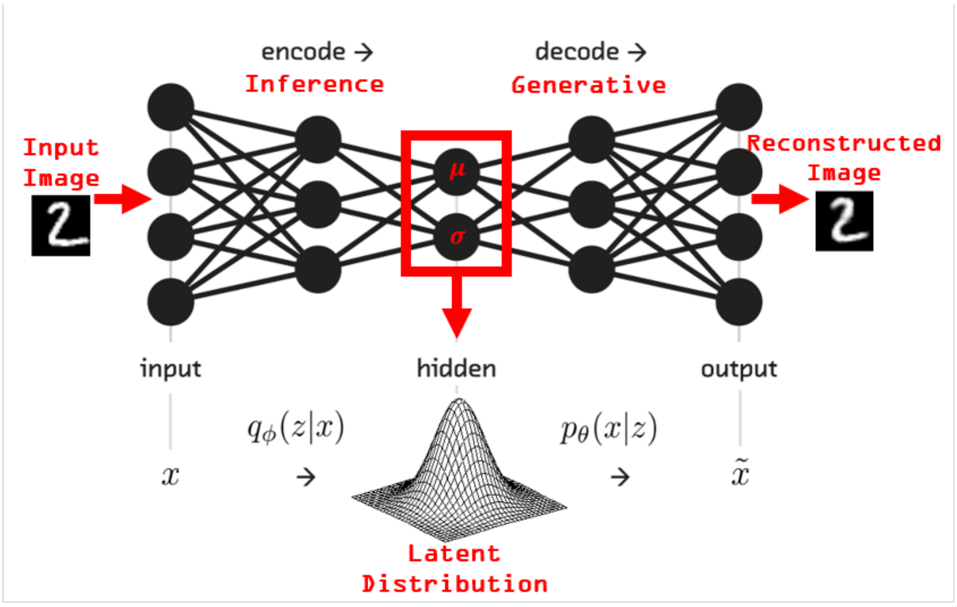


Figure 2.18: The concept of a variational auto-encoder

the latent space of an auto-encoder. Thus it has a smooth transition between different sampled latent variables.

Figure 2.18 shows a typical VAE which works as follows. First, The encoder of the VAE encodes the input as a distribution by converting it into mean and variance. The mean and variance will later be used to measure the difference between two probability distributions with Kullback-Leibler (KL) divergence [183, 184]. Then, a point from the latent space is randomly sampled from the distribution. After that, by providing the point from the latent space, the decoder will reconstruct the sample. Then, the reconstruction loss can be calculated by comparing the model output and the input. The reconstruction loss is commonly calculated by using Binary Cross Entropy (BCE) or Mean Squared Error (MSE). BCE is a more efficient loss function when the input and output data are between 0 and 1, whereas the MSE can be used with any range of input and output data. The final loss function of VAE is the sum of reconstruction loss and the KL divergence loss of the latent space. The KL divergence loss often multiplies with a weight (β) to balance the influence between the reconstruction loss and the KL divergence loss.

Generative models are notoriously hard to train. With GAN, there is mode collapse, with VAE, a similar phenomenon is called posterior collapse. When posterior collapse occurs, the decoder will generate some very similar output no matter what the latent variable is. This is often caused by the imbalance between the reconstruction loss and KL divergence loss. Over the years, different methods were proposed to minimise the chance of posterior collapse happening, including beta-VAE [185], Sigma-VAE [186], cyclical annealing [187] or monotonic annealing [188]. The results of these methods do not guarantee success, especially considering most of these methods are only applied in image or NLP. To keep things simple, this work only focuses on changing the size of the latent and the weighting of KL loss.

Skip Connection

Similar to RNN, a large network with lots of layers can suffer from the Vanishing Gradient problem. Backpropagation is prone to influence the beginning of the backpropagation, which are the layers closer to the output. As a result, the layers closer to the input will have smaller gradients. In some cases, the gradient becomes zero, thus, the early layers do not update at all. So although in theory, deeper networks could solve more complicated problems, in practice, deeper networks could perform worse than shallower networks caused by optimization problems instead of overfitting. Instead of connecting the layer of neurons or convolutional layer sequentially, skip connection provides a parallel path that skips some layers as an alternative path for the gradient with backpropagation. In the skip connection path, the layers that are further away from the output will be closer in backpropagation (Figure 2.19). The most well-known model is the ResNet which was introduced in 2015 (Figure 2.20) [189].

Another well-known model is the UNet, originally designed for Semantic segmentation but later used in some image-to-image models [191]. The idea of UNet is similar to CAE; it encodes the input signal into a latent space and then decodes the latent variables to predict the output. What makes the UNet special is that it has skip connections across the encoder and decoder that skip across the latent space, which gives it the u-shaped architecture.

Generative adversary networks (GAN)

Generative Adversary Networks (GAN) were considered groundbreaking when they were first proposed by Ian Goodfellow in 2014 [18]. GAN soon became the state-of-the-art model in various applications, including style transfer, image sharpening, image inpainting and music generation [192, 193]. The initial idea of the model is to generate meaningful new data from scratch by using two different models, a generative

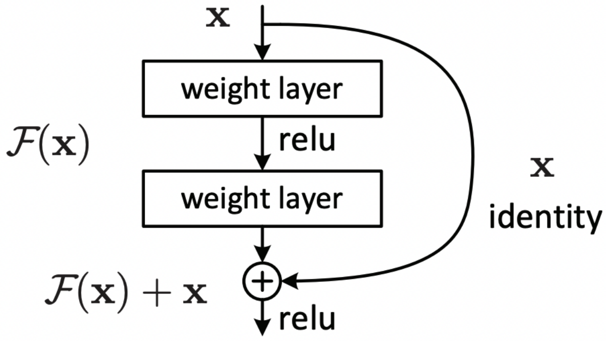

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 2.19: The idea of skip connection [189]

model and a discriminative model. The generative model generates new data, and the discriminative model will estimate the probability that the data came from the training data (real) or the generative model (fake). The problem with GAN is that it is notoriously hard to train. The concept is that GAN is non-convergent; both the generator and discriminator models are trained simultaneously in a zero-sum game. Instead of finding the minima in the gradient, a GAN is trained to find the Nash equilibrium between the generative and discriminative models [18, 194]. In layman's terms, to work, the generative model and the discriminative model can not outperform one another significantly. Especially when the discriminator gets too successful, it will cause diminished gradients, where the generator's gradient vanishes and learns nothing.

Another common problem is mode collapse, where the generator produces a limited variety of output to pass the discriminator. Despite its challenges in training, GAN is a popular generative model as it can generate decent results when appropriately trained. The recent development of the Diffusion Network has become an alternative option to GAN in some tasks and does not require adversarial training. However, due to its complicity and fast-changing development, the discussion of the Diffusion Network is out of the scope of this thesis.

2.2.5 Machine learning based HRTF research

In recent years, machine learning techniques have emerged as promising tools for individualising HRTFs in spatial audio research. The recently published paper from McMullen et al. has summarised the machine learning development in HRTFs for spatial auditory display [196]. There are plenty of machine learning-based HRTF personalisation methods that focus on anthropometric parameters or ear scans [20, 63, 65, 66, 124, 197, 198]. These works lead to some real-world applications from Sony [199], Genelec [200], and Apple [201]. Due to the characteristic of CNNs **??**, it has been utilised for HRTF prediction and interpolation [202–204], while Auto encoders [205], deep belief networks [206], and GANs [207] have also been explored for HRTF individualisation tasks. Anthropometric features are commonly used as input for these machine learning models to predict personalised HRTFs [202, 208]. These methods aim to enhance the realism and immersion of spatial audio experiences by providing more accurate and personalised HRTFs.

Other research has focused on forms of listening tests. A good example is a work presented by Ymamoto et al., who trained a VAE and used a subjective listening test to find the personalised HRTF [209]. Their results show that tuned HRTFs perform better than the best-fitted HRTFs from the CIPIC database [24]. It also demonstrated how to train a neural network with limited data by using each measurement angle as an individual.

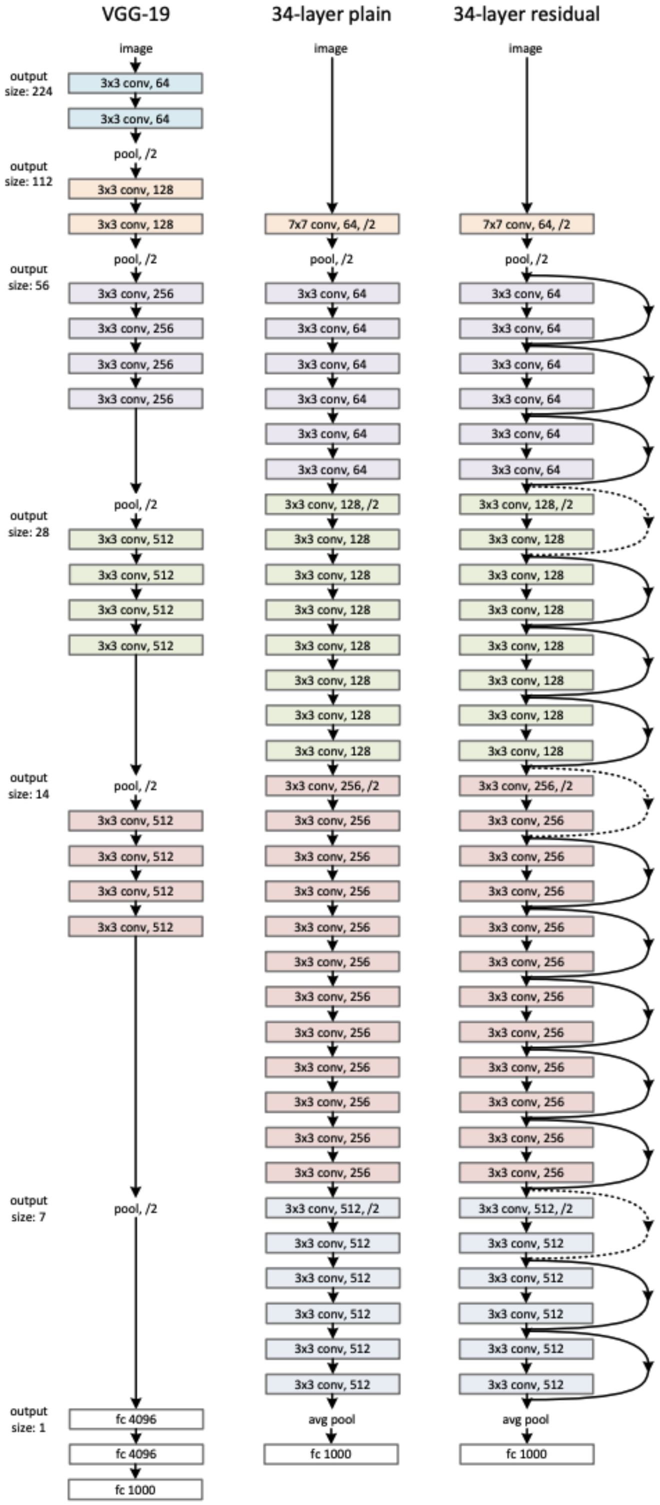


Figure 2.20: Comparing Resnet with other neural network models. Left: the VGG-19 model [190] (19.6 billion FLOPs) as a reference. Middle: a plain network with 34 parameter layers (3.6 billion FLOPs). Right: a residual network with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions. [189]

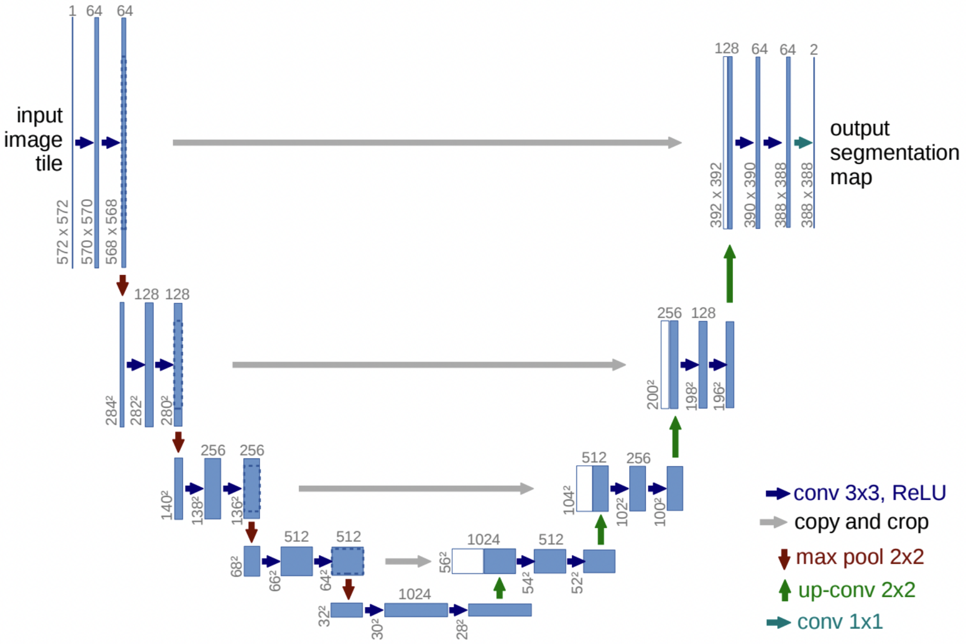


Figure 2.21: The idea of a UNet [191]

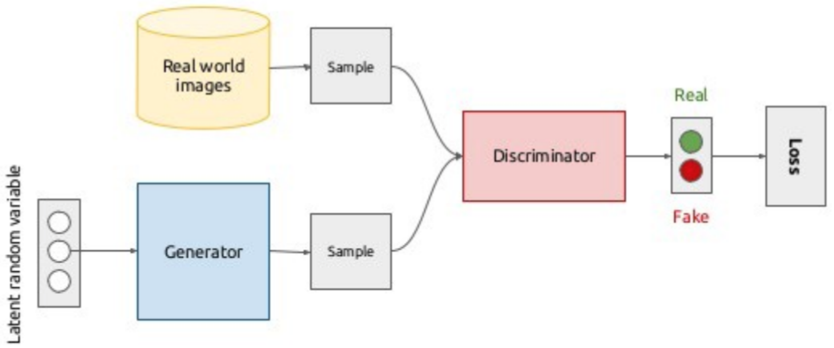


Figure 2.22: The concept of a GAN [195]

2.3 Discussion

To conclude this chapter, a few deliberations and reflections are discussed in this section, which emerged during the research process. These might provide some insight for future development in this domain.

2.3.1 HRTF Data format

When selecting HRTF data for machine learning applications, there are two primary factors to consider: whether to use individual HRTF measurements or entire HRTF datasets and their data formats. For the first consideration, utilising individual HRTF measurements provides greater flexibility, as it reduces the need to consider the measurement positions extensively. However, training a model with individual HRTF measurements may sacrifice vital information from different measurements and the spatial relationship between those positions. On the other hand, employing HRTF datasets typically requires that the number and positions of HRTF measurements be relatively consistent. This constraint may limit the available training data and increase the complexity of data pre-processing. Nevertheless, using HRTF datasets allows for a more comprehensive understanding of the spatial relationships within the data. Other than that, as for understanding the relationship of HRTF measurements in different positions on a sphere in an HRTF dataset, there is ongoing research on spherical CNNs. Spherical CNNs are a type of neural network architecture designed to process data defined on the sphere. They appear to be promising for representing HRTFs, due to their capacity to comprehend the spatial relationships of measurements taken at different points on a sphere. This can lead to improved model performance by preserving the spatial relationship between measurements, which is a key aspect of HRTF data. This could potentially open up new possibilities for HRTF research and could potentially overcome the limitations of conventional machine learning models in dealing with HRTF data.

The second consideration in selecting HRTF data for machine learning (ML) models is the choice of data format and type, with options including time-based, frequency-based, wavelet, and Spherical Harmonics representations. Although end-to-end learning is a common goal in ML development, meaning the model should process raw data without extensive pre-processing, employing clean and feature-extracted data can enhance training speed and, in many cases, improve model performance. Time-based representations, such as HRIRs, provide a straightforward data format that captures impulse response delays and might have a more direct connection to anthropometric information. Conversely, frequency-based data, like HRTFs, offer insights into the spectral characteristics of HRTFs, presenting a clearer understanding of their impact on the sound, as it is more easily interpretable by humans. Wavelet representations deliver both time-frequency information and efficient compression, though they are less commonly used, possibly due to the cumbersome conversion process. Lastly, Spherical Harmonics are well-adapted for depicting HRTF spatial distributions on a sphere, facilitating accurate interpolation and efficient storage, as well as compatibility with Ambisonics. However, Spherical Harmonics can only represent HRTF datasets, which may not be ideal if individual HRTF measurements are preferred for the application. The optimal data format and type for a specific ML model will depend on the problem's nature and the research objectives.

2.3.2 The future of HRTF Personalisation

However, a more practical direction for HRTF personalisation research may involve creating a method that doesn't require detailed head scans, ear scans, or listening tests. This concept is based on three key ideas. First, it would be great to make the process as smooth as possible so that anyone can enjoy spatial audio without too much hassle. Second, some studies suggest that not everyone prefers their own HRTF set, or their preferences might not be reproducible [210, 211]. Lastly, considering that some people don't prefer their own HRTF set, it may be possible that a dynamic, personalised "super" HRTF set could improve a user's virtual reality experience or enhance their performance. This personalised set could be flexible, changing based on different scenarios or needs.

In a lot of reinforcement learning tasks, like Alpha Zero, AlphaStar, and Chat GPT [146, 169, 180, 212], the ML model can improve by interacting with human feedback. One idea involves leveraging natural human feedback in virtual reality (VR) or augmented reality (AR) environments. As users interact with virtual sound sources, their head and eye movements can provide valuable information for personalising HRTFs dynamically. By incorporating Reinforcement Learning with Human Feedback (RLHF), one of the key ideas in Chat GPT, with users' interactions with virtual sound sources and gathering their feedback on the perceived spatial audio quality, an Reinforcement Learning (RL) agent can be trained to optimise HRTF personalisation dynamically [179–181]. As the agent explores different HRTF parameters, it can receive feedback from the user in the form of rewards or penalties, allowing it to learn the optimal HRTF configuration for each individual. This approach could lead to more accurate and immersive spatial audio experiences tailored to individual users. Investigating novel methods for HRTF penalisation that are based on user feedback, machine learning, and reinforcement learning can ultimately result in more accessible and immersive spatial audio experiences for individuals.

2.3.3 HRTF Personalisation with Transformers

The public availability of Chat GPT has brought Transformers into the spotlight [177, 180]. However, there are two main considerations when looking to apply them in HRTF personalisation. Firstly, Transformers were primarily developed to handle sequential data, which does not necessarily match the nature of HRTF data. However, for tasks related to personalised HRTFs, particularly for Ambisonics that uses SH HRTF sets, it is possible to view the HRTF personalisation process as a sequence of HRTF sets. In this case, Transformers could be utilised in such HRTF personalisation tasks.

Secondly, Transformers were designed to work with discrete, tokenised data rather than continuous data. While it is technically possible to treat each HRTF set as discrete, tokenised data, considering that human anthropometric measurements are inherently continuous, it is plausible that a range of potential HRTFs could exist between different HRTF sets. One way around this could be to leverage the latent space created by the Autoencoder or PCA. In this approach, the Autoencoder or PCA transforms the input HRTF datasets into a continuous latent space which is then processed by the Transformer. This can be achieved by using the Autoencoder's encoder or PCA to map the input data into latent vectors and then feed those vectors into the transformer. The transformer will handle the latent vectors as continuous data rather than discrete tokens. Another approach could be to quantise the continuous latent space into a set of discrete tokens, essentially creating a vocabulary for the latent space. This can be done using techniques like Vector Quantisation (VQ) or k-means clustering. Once the continuous data is converted into discrete tokens, it can be fed into the transformer just like any other tokenised data.

Despite the potential of Transformers, it's important to note that some transformer-based language models demonstrate emergent behaviour: they perform at random until reaching a certain scale, after which their performance significantly improves to well above random. This presents a challenge for model evaluation at smaller scales, while deployment at larger scales requires significant computational resources and training data.

2.4 Conclusion

Based on the current state of spatial audio and machine learning, it is clear that HRTFs and machine learning have the potential to revolutionise the way we experience immersive audio. With the rapid growth of VR technology and the increasing demand for realistic, immersive audio experiences, it is important to continue exploring and refining the use of HRTFs in combination with machine learning algorithms. This Chapter outlined the background of HRTFs and some recent developments in machine learning. Some machine learning models, such as CNN, GAN, and Auto-encoder, have already made significant strides in a variety of fields, including computer vision, natural language processing, and image processing. There has also been some research in the area of HRTFs with machine learning that showed promising results [62, 196, 209]. Further exploration and development in this area may lead to significant improvements in the realism and immersion of spatial audio and have a significant impact on the future of VR. This research direction holds great promise for the future of audio technology and has the potential to revolutionise the way people experience sound.

Chapter 3

A HRTF Consolidation Tool For High Variance Machine Learning Algorithms

3.1 Motivation and Challenges

As discussed in Chapter 2, some modern ML models require a significant amount of data to be trained adequately. However, measuring HRTFs is usually undertaken in free-field conditions with either human beings or dummy heads, which can be tedious and time-consuming. Consequently, although there are different HRTF measurement databases available, they are limited in the number of subjects or number of HRTF positions sampled.

To obtain a large number of HRTFs without undertaking extra measurements, one option is to combine current HRTF sets from different databases. However, most HRTF databases have been measured under different settings and have different attributes. As machine learning algorithms usually require uniform input data, consolidating different HRTF databases requires significant preparation and processing to unify measurement attributes.

The most data-demanding way to use the consolidated HRTFs as training features is to use each measurement position as a single or subset of training features. When the number of databases increases, oftentimes, there will be fewer common angles between them, hence a reduction in the number of input features. The number of datasets from the databases (training samples) versus measurement angles (input features) can have a direct effect on the balance between the variance and bias of a model. Finding the combination of datasets that could maximise the number of databases and measurement angles can be challenging. At the same time, considering some ML tasks may allow for a higher margin on the HRTF measurement angles. A minor angle difference between datasets could be tolerable to retain more matched angles. Finding the most common angles across the most number of datasets with a predefined tolerable angle adds another level of complication to the challenge. Lastly, when dealing with numbers of datasets more than two, the time for cross-comparing the angles across datasets increases exponentially. All these requirements may sound straightforward to solve, but it is challenging to make the process robust and less time-consuming. A Matlab tool was built to find the common angles across HRTF sets, extract the HRTFs and save them in different SOFA files accordingly.

Note that besides sample data and feature sizes, the quality and unity of the data are also important. Andreopoulou et al. [213] conducted a comparative analysis of HRTF measurements taken from a single dummy head microphone (Neumann KU-100) in various laboratories worldwide — a study commonly referred to as ”Club Fritz”. The findings indicate noticeable variations between measurements. Research from Pauwels et al. shows that machine learning classifiers can easily identify the measurement setup in HRTFs. They believe that simply mixing (harmonised) datasets to increase the number of training examples will not automatically lead to an increase in robustness. This tool's ability to integrate different HRTF measurements from various databases could be beneficial for researchers when using it mindfully.

3.2 Development of an HRTF Database Consolidation Tool

The developed HRTF database consolidation tool works exclusively with datasets in the Spatially Oriented Format for Acoustics (SOFA), the most common file format for HRTFs [214]. It can store HRTF measurements from different positions with defined attributes in a single file. The file format was first suggested by Majdak et al. in 2013 [95] and subsequently standardised by the AES in 2015 [214]. Since then, this format has been widely adopted by different HRTF databases [20, 21, 24, 25, 215, 216].

This tool is a Matlab program that consists of different functional building blocks. The main function of this toolbox is `preprocess_SOFA.m`. The input variables consist of: SOFA files or related folders (`input_files`), range of tolerance in angle matching (`angle_range`), measurement distance error tolerance (`dist_range`) and the name of output folder (`output_dir`). The function will automatically find the best matching angles, normalise with the proper attributes and save them in a new folder as dictated by the user.

Most of the building blocks can also be used as standalone functions. Details of the functions are documented within each script and their related folder.

3.2.1 Pipeline and File Structure

The data preparation pipeline is as follows:

1. Check SOFA files and compare the files from the same folder to find odd files
2. Organise and fix abnormal SOFA files (may require user's input to decide what to keep if there is an abnormal SOFA file),
3. Find SOFA files that can represent other files with a similar angular distribution (to improve speed).
4. Find angular matches in 'unique' SOFA files (may require user's adjustments if match not found).
5. Find suggested normalisation attributes.
6. Extract measurements, normalise and save as new SOFA files to the desired output folder.

7. Output summary of the new files. (new file names, remaining measurement angles and notes about the file if there is any modification)

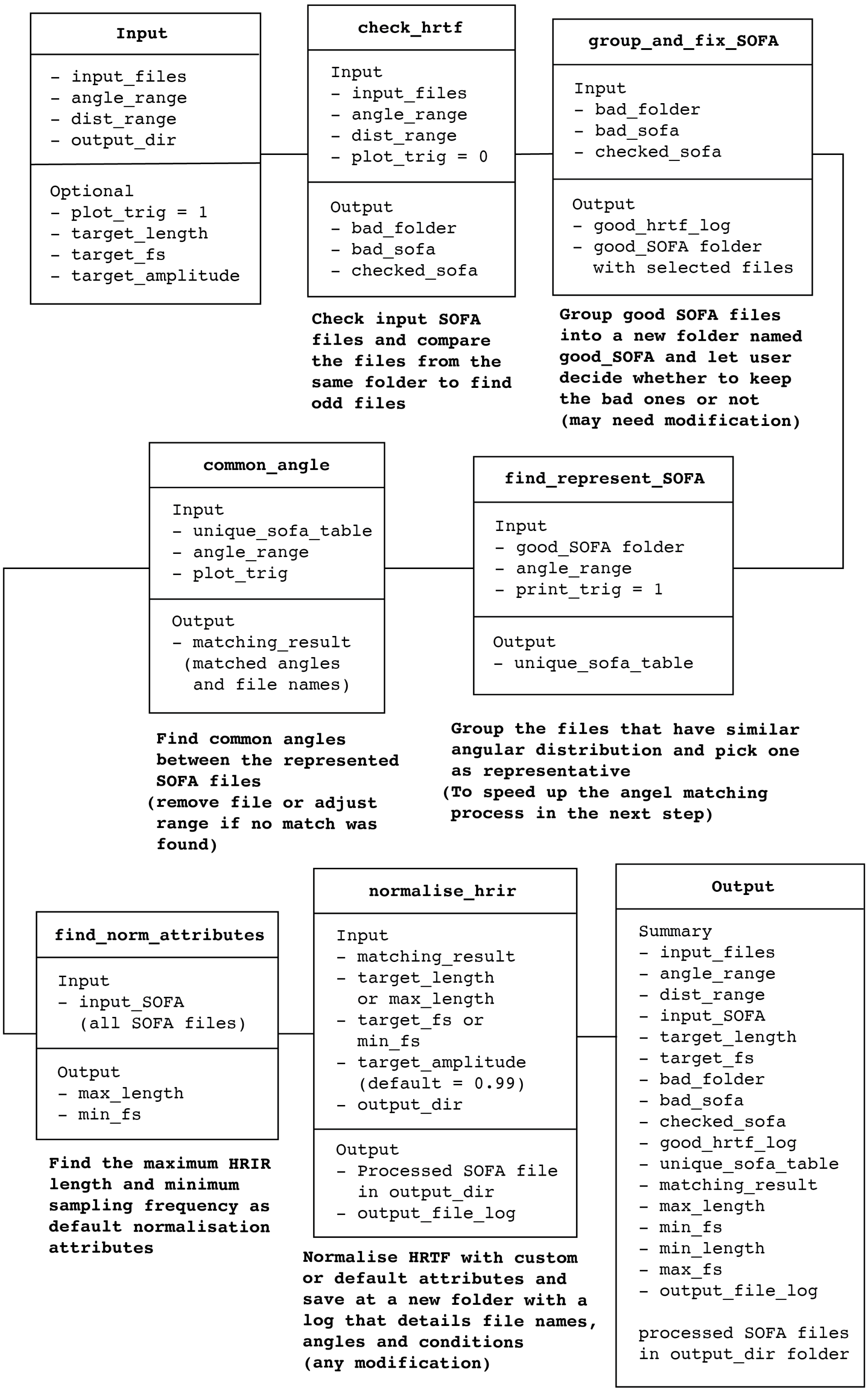

Input
- input_files
- angle_range
- dist_range
- output_dir
Optional
- plot_trig = 1
- target_length
- target_fs
- target_amplitude
check_hrtf
Input
- input_files
- angle_range
- dist_range
- plot_trig = 0
Output
- bad_folder
- bad_sofa
- checked_sofa
Check input SOFA files and compare the files from the same folder to find odd files
group_and_fix_SOFA
Input
- bad_folder
- bad_sofa
- checked_sofa
Output
- good_hrtf_log
- good_SOFA folder with selected files
Group good SOFA files into a new folder named good_SOFA and let user decide whether to keep the bad ones or not (may need modification)
common_angle
Input
- unique_sofa_table
- angle_range
- plot_trig
Output
- matching_result (matched angles and file names)
Find common angles between the represented SOFA files (remove file or adjust range if no match was found)
find_represent_SOFA
Input
- good_SOFA folder
- angle_range
- print_trig = 1
Output
- unique_sofa_table
Group the files that have similar angular distribution and pick one as representative (To speed up the angel matching process in the next step)
find_norm_attributes
Input
- input_SOFA (all SOFA files)
Output
- max_length
- min_fs
Find the maximum HRIR length and minimum sampling frequency as default normalisation attributes
normalise_hrir
Input
- matching_result
- target_length or max_length
- target_fs or min_fs
- target_amplitude (default = 0.99)
- output_dir
Output
- Processed SOFA file in output_dir
- output_file_log
Normalise HRTF with custom or default attributes and save at a new folder with a log that details file names, angles and conditions (any modification)
Output
Summary
- input_files
- angle_range
- dist_range
- input_SOFA
- target_length
- target_fs
- bad_folder
- bad_sofa
- checked_sofa
- good_hrtf_log
- unique_sofa_table
- matching_result
- max_length
- min_fs
- min_length
- max_fs
- output_file_log
processed SOFA files in output_dir folder

Figure 3.2.1 shows a detailed flowchart with the individual function names of each step, input and output that are used in the main script.

3.2.2 Details

The details about each part of the main function are well documented in the script and folder. This section shares the important information and operations in some of the main building blocks.

Find and Fix Abnormal Files

On occasion, there may be a few abnormal files or measurements inside HRTF databases, such as missing measurements, asymmetric angular distributions or singular measurements that are taken at a different distance due to measurement constraints. These abnormalities may lead to errors when training some algorithms. The proposed tool uses a function to find abnormal files inside the folder as well as abnormal measurements in SOFA files before further processing. Note that these functions are only able to find some simple abnormalities due to the speed constraint.

- `check_hrtf.m` is the general function for finding abnormal data. It outputs two tables to identify problematic areas. One is `bad_folder` for the abnormal SOFA files in folders, and another one is `bad_SOFA` for the abnormal measurements within the SOFA file.

- `check_hrtf_folder.m` compares the consistency with other files inside the folder in four ways: number of measurements, number of channels, HRIR length and sampling rate. The output indicates the name of any problematic SOFA files.

- `check_SOFA.m` checks the individual HRTF measurements inside the SOFA file. It focuses on four aspects: repeated measurement angles inside the file, inconsistency in measurement distance, asymmetrical measurement angle distribution on the left and right side, and asymmetrical measurement angles on the median plane apart from the top and bottom. The output indicates the locations of the problematic measurements.

Find Matched Measurement Angles

The primary distinction among HRTF databases lies in the variations in measurement angles. When integrating various databases, there might be very few overlapping angles. It's commonly agreed that the just noticeable difference for localisation angles in the frontal plane can be as small as 1 degree for human listeners [104, 217–219]. Nonetheless, such a level of precision may not be a necessity for some machine learning systems in their training phase.

The balance between optimising the number of HRTFs and aligning with the most common angles is application-dependent. At times, having a more substantial data set outweighs the need for precision in data location, and at other times, the opposite may be true. For instance, if localisation is treated as a classification issue rather than a regression problem, various HRTF databases with differing neighbouring angles may be considered.

Polar coordinates are commonly used across different datasets to mark HRTF measurement positions. From this, the orthodromic (global) distance can be computed for the distances between measurement points using the `distance` function in Matlab. However, computing the distances between all measurement points with this function can take a long time.

The function `common_angle.m` uses a hybrid method that narrows down the measurement angles by comparing their polar coordinates before calculating their global distances.

Depending on the angular distribution, this hybrid method usually works a lot faster than just using global distance on multiple datasets. The details are documented in the script.

The function includes other matching methods, such as the global distance only or comparing the angle difference only. However, worth noting that using global distance only on more than two datasets will take a long time to finish.

Normalisation Schemes

Alongside measurement angle, other attributes that vary between different databases include sampling frequency, magnitude and HRIR length (FIR filter tap size). Using a combination of datasets with different attributes usually leads to inaccurate results. This tool includes a function (`normalise_hrir`) to normalise them to ensure all the data is within the same range before exporting as a new SOFA file.

To ensure the HRTF frequency response will not be affected by normalisation, the built-in normalisation method handles the process in the frequency domain.

The default normalisation attribute takes the longest HRIR length (to avoid cutting off the tail) and the lowest sample rate from the input datasets (to avoid up-sampling), with a 0.99 magnitude. These attributes can be customised optionally by inputting additional input variables when calling the main function.

3.2.3 Using the Toolbox

The suggested input parameters in the main function `preprocess_SOFA.m` includes SOFA files or related folders (`input_files`), range of tolerance in angle matching (`angle_range`), measurement distance error tolerance (`dist_range`) and the name of output folder (`output_dir`). Before going through the preparation process, the function will first check all the input attributes, especially whether the appointed output folder already exists. If the folder exists, users can decide whether to overwrite it.

During the organising and fixing process, a folder named `good_SOFA` will be created to group all the good and repaired SOFA files in one place. If any problematic SOFA files need modification, it will request further instructions in the command window. Otherwise, the function will pause after grouping all the SOFA files.

By pressing any key to continue, the function will start finding matched angles inside the `good_SOFA` folder. To avoid spending time processing the files with similar angular distributions, the function will first run a quick check to find out which SOFA files share the same distribution, then pick one file to represent each distribution in the matching process. During the angle matching process, there is a chance that the function may not find any match, which then requires users to set a wider tolerance range or remove one of the files. After the matching process, the function will pause again, and plot matched angles of each represented data set on a graph with a uniform distance for better comparison (unless the user has suppressed the plot).

By pressing any key to continue, the function will normalise and save the processed HRTFs as new SOFA files. The function will find the default recommended attribute (maximum HRIR length and minimum sampling frequency), then normalise and save the files at the appointed output folder.

3.2.4 Extra Tool: Plot Measurement Angles

One tool that could benefit all binaural researchers is the `plot_3d_angle.m` function. This function can be used alone, allowing researchers to plot all HRTF measurement points in a 3D plot with a head in the centre as a directional and distance reference. It is similar to the `SOFAplotGeometry` in the original SOFA function [220] but with more flexibility. Firstly, the input not only accepts a pre-loaded SOFA struct but also accepts SOFA files directly or just the azimuth angle, elevation angle and distance. Secondly, this function has the flexibility to plot different measurements on the same plot with different colours and markers. This helps compare the differences between different angular distributions. Finally, the marker's locations on the graph are displayed in polar coordinates instead of the Matlab default Cartesian coordinates, by selecting the marker on the graph, will show the polar coordinates of the selected point in three dimensions.

3.3 Summary

Modern machine learning algorithms often require plenty of training data to unleash their full potential. As discussed in Section 2.1.4, depending on the requirement of the application, there are roughly a total of 300 HRTF datasets available. Considering the limited availability of HRTF sets and the difficulty of obtaining the HRTF measurements, consolidating multiple HRTF datasets is one solution to gather a reasonable amount of training data. However, consolidating HRTF datasets with similar angular distributions and attributes can be a significant task. This Chapter has presented a method to find common angles across different datasets relatively fast, with a workflow that streamlines the data preparation process.

A Matlab tool has been built based on the method proposed in this Chapter, which allows researchers to find similar angles between different databases within a specific range and consolidate those HRTFs with normalised attributes.

The tool is available online:
`https://github.com/Benjamin-Tsui/HRTF_preprocessing`

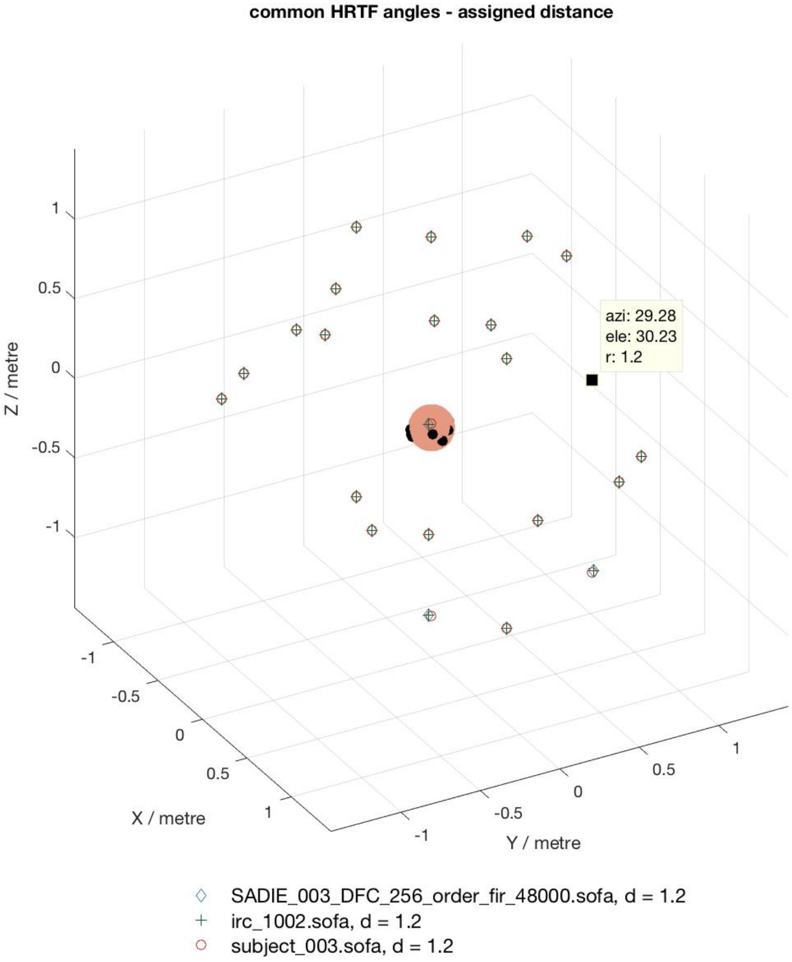


Figure 3.1: Matched angles between SADIE, LISTEN and CIPIC database with 1-degree angle tolerance

Chapter 4

Low-order Spherical Harmonic Interpolated HRTF Restoration using a Neural Network Approach

4.1 Motivation

In Section 2.1.5, the idea of HRTF interpolation has been discussed, which could simplify the measurement process to acquire dense HRTF sets from sparse HRTF measurements. Spherical Harmonic (SH) interpolation was used in this chapter as the process is more robust and standardised compared with other methods, such as linear or barycentric interpolation. SH interpolation leverages spatial continuity in SH, which could be used as a bridge to spatially up-sample a sparse HRTF measurement set to a denser one depending on the number of sparse HRTF measurement points and SH order [221, 222]. Consequently, listeners may perceive timbre differences and weakened localisation performance in practical use.

Recently, developments in machine learning have shown great improvement in neural style transfer and data restoration, especially in the image domain [8, 9]. This chapter investigates whether similar models can be used to restore distorted high-frequency data in SH-interpolated HRTFs.

This chapter is organised as follows: Section 4.2 will cover the relevant information on HRTF SH interpolation for this work. Section 4.3 will discuss the method used in this study, including the data pre-processing workflow, a baseline model and different techniques investigated on top of the baseline model. Section 4.4 evaluates the performance of the model based on perceptual spectral difference and localisation performance.

4.2 Spherical Harmonic HRTF interpolation

As mentioned in section 2.1.5, the main issue caused by SH HRTF interpolation is that the interpolated HRTFs are only accurate up to the spatial aliasing frequency f_{lim}, approximated by Equation 2.7. Whilst Dual-band Time Alignment (TA) can combat this issue to some extent, it is desirable to find if there is any way to improve the interpolated HRTFs further. Porschmann et al. presented Spatial Upsampling by Directional Equalization (SUpDEq) method, by removing direction-dependent temporal and spectral components of the HRTFs, such as frequency-dependent ITDs and ILDs as well as elevation-dependent spectral features, before the SH interpolation. Directional de-equalisation is then applied to the interpolated HRTFs to restore the previously discarded features, resulting in a dense set of interpolated HRTFs [223]. This method requires applying directional equalisation with an appropriate equalisation dataset. In this chapter, with the spirit of the connectionist ML algorithm, a more direct method that requires the least feature extraction is proposed. The input data for the model is the individual left and right HRTF pair from a random angle. The model will then attempt to restore the distorted data from the SH interpolated HRTFs. Other than dual-band Time Alignment (TA), the input HRTF was kept as raw as possible to find out how far the connectionist ML philosophy can be applied to SH HRTF restoration and general HRTF application.

To challenge the full potential of the use of machine learning, this research chooses to use 1st order SH interpolation as it requires the least number of HRIR measurements. Although the minimum number of HRTF measurements for 1st order SH is a Tetrahedron configuration with 4 measurements, an Octahedron configuration with 6 measurements is selected, which has a more stable energy distribution than other arrays for 1st order SH [224]. Interpolation with 1st order SH up to the original number of measurements is undertaken, which, depending on the dataset, can be ¿2000 measurements. More specifically, the 6 selected sparse HRIRs from the HRIR database are first converted to SH HRIR (1st order SH which has 4 channels) by using the re-encoding matrix C with 4 rows and 6 columns. Then a decoding matrix $\hat{\text{D}}$ is computed using a different re-encoding matrix $\hat{\text{C}}$ with the same number of rows and a different number of columns based on the desired number of HRIRs to interpolate. A brief overview of the concept can be found in Figure 4.1.

4.3 Machine learning HRTF Restoration

As discussed in chapter 2.2, research domains like speech recognition, natural language processing, and computer vision have demonstrated that a more general data-driven, connectionist method often beats traditional knowledge-based signal processing methods in the long run, as it can deal with raw data better and the data can keep growing in the future [225]. This has been proven with the development in noise reduction in images, image inpainting, colourising old photos or videos, and neural style transfer [8–16] and these tasks can be considered to be quite similar to restoring distorted SH interpolated HRTFs. Some examples include variants of fully connected NN, CNN, Auto-encoder, CAE [15], ResNet [17] and GANs [18].

Most machine learning models require large amounts of labelled data to produce excellent results. However, there are only a total number of 233 HRTF datasets freely available combined in Spatially Oriented Format for Acoustics (SOFA) format at the time of conducting this work in 2019 [98, 99]. Compared to the data size used to train image processing machine learning models, which can be in hundreds of thousands of images, HRTF data is far too few to generalise well or train sophisticated models. However, even with such limited data, SH interpolated HRTF restoration is potentially achievable

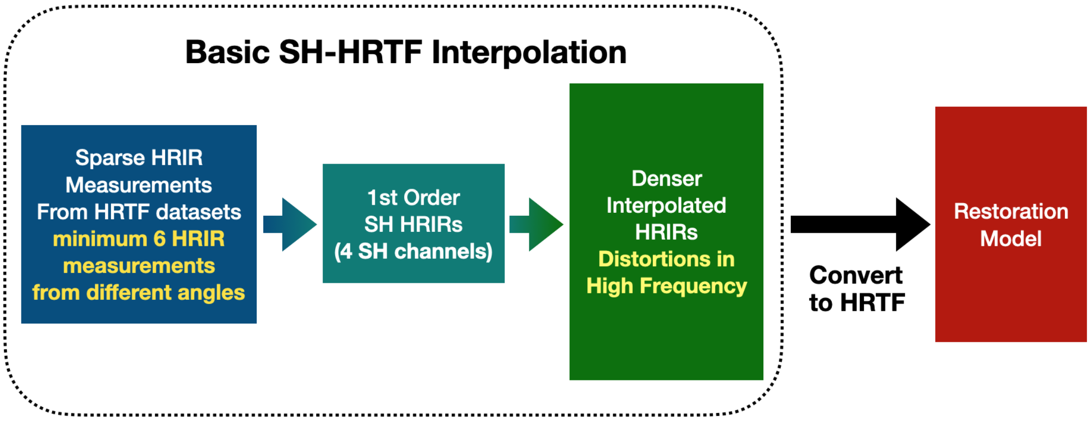

Basic SH-HRTF Interpolation
Sparse HRIR Measurements From HRTF datasets
minimum 6 HRIR measurements from different angles
1st Order SH HRIRs (4 SH channels)
Denser Interpolated HRIRs
Distortions in High Frequency
Convert to HRTF
Restoration Model

Figure 4.1: An overview of the proposed method where a model was trained to reconstruct the distorted high-frequency information in the SH interpolated HRTFs, SH interpolation was done in time-domain before converted into HRTFs to feed into the model

using machine learning algorithms and is now being investigated. The advantage of using a machine learning model is that the result can hopefully be improved in the long run when more labelled data is available in the future.

An overview of the proposed method is shown in Figure 4.1. A subset of HRTFs is selected from a database to represent a sparse HRTF measurement set. These HRTFs are then interpolated in a traditional SH HRTF interpolation manner. After the interpolation, each HRTF measurement feeds into the machine-learning model for restoration. This work chooses the output size for the interpolation process based on the number of HRTF measurements of the original dataset, which is typically over 2000, depending on the dataset. The restored HRTFs output from the machine learning model can then be easily compared with the true HRTF measurements. In this section, we will first discuss the data preparation and format, then introduce the baseline model used in this study before improving it with different enhancement techniques, including weight decay, dropout, early stopping etc.

4.3.1 Data pre-processing

The training and testing data are extracted from different HRTF databases, including ARI [19], ITA [20], RIEC [21], SADIE I [22], SADIE II [23], IRCAM Listen [25], and the Bernschutz KU100 [26] database.

The SH interpolation process takes place in the time domain, as HRIRs are then converted to frequency domain HRTFs after the interpolation process for input to the restoration model. The phase is discarded after the conversion because using complex numbers in conventional NN and CNN can be problematic for certain functions. Whilst there are some alternative methods for using complex numbers that have been proposed, it is questionable if these nontrivial methods are necessary for this project [226–229]. Note that due to the randomness of machine learning models, there is a chance that a model could incorrectly give negative amplitude spectra in the output. To avoid this, the input and output data are scaled to decibels. The output data is then rescaled

	azimuth	elevation
1	90.0	0.0
2	270.0 (or -90.0)	0.0
3	0.0	45.0
4	0.0	-45.0
5	180.0	45.0
6	180.0	-45.0

Table 4.1: Angle selection for training, validation and testing

	azimuth	elevation
1	0.0	0.0
2	180.0	0.0
3	90.0	45.0
4	90.0	-45.0
5	270.0 (or -90.0)	45.0
6	270.0 (or -90.0)	-45.0

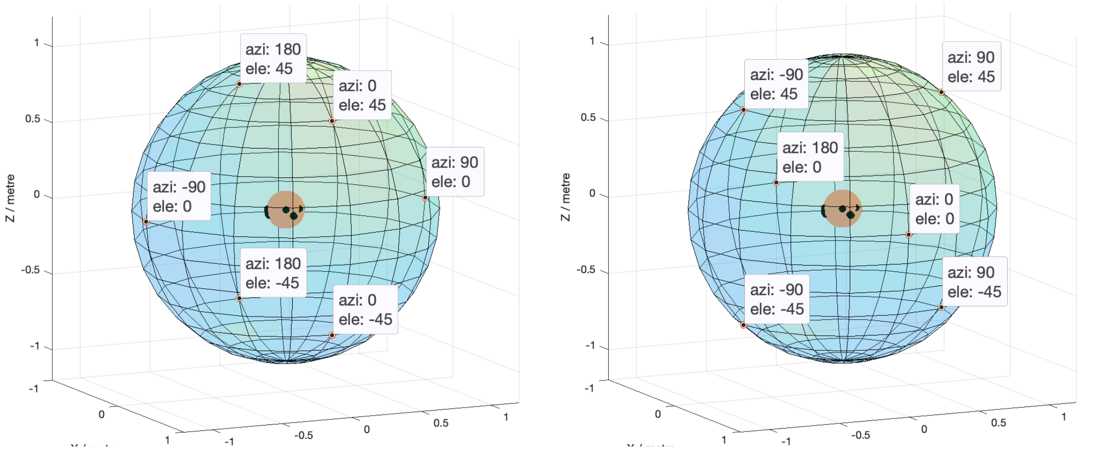
Table 4.2: Angle selection for training and validation only

before converting back to the time domain. All the processing was done with the HRTF consolidation tool proposed in Chapter 3.

Data selection

This work uses 6 measurements with an octahedron configuration for the sparse dataset, which is one of the more challenging cases for SH HRTF interpolation as it involves the second lowest number of measurements in common configurations for 1^{st} order SH [224]. There are two sets of configurations used, one used in both training and testing, and the other one used only in training for data augmentation purposes. The angles are shown in Tables 4.1, 4.2 and 4.2.

Amongst all the popular HRTF datasets, only the SADIE I [22], SADIE II [23], IRCAM Listen [25], and Bernschutz KU100 [26] databases can provide the measurements from these angles. In this work, Subjects 19 and 20 from the SADIE II database are held out for testing and evaluation of the model and are not used in the training. SADIE II Subject 20 was tracked during the training process. This design tries to show how well the model copes with unforeseen HRTF measurements. Furthermore, the work from Andreopoulou et al. [213], known as ”Club Fritz”, conducted a comparative analysis of HRTF measurements from a single dummy head microphone (Neumann KU-100) across various laboratories worldwide, revealing noticeable differences among the measurements. The Bernschutz HRTFs [26], measured from a Neumann KU100 dummy head in the anechoic chamber at Cologne University of Applied Sciences, were also excluded from the training and validation sets but tracked during the training process. This allows us to study the effect of alternate HRTF measurement methods of expected near-match datasets on the existing KU100 measurements in the training data.



(a) Angle selection for training, validation and testing (b) Angle selection for training and validation only

Figure 4.2: PSD of Bernschutz KU100 dataset (Note that -90 = 270 in this figure to give a better sense of direction)

HRTF dataset	Training and validation	Testing
SADIE I	✓	
SADIE II (besides subject 19 and 20)	✓	
IRCAM Listen	✓	
ARI	modified[a]	
ITA	modified[a]	
RIEC	modified[b]	
SADIE II (subject 19 and 20)		✓
Bernschutz KU100		✓

Table 4.3: Angle selection for training and validation only

[a] Positions with an elevation angle at -45° were changed to -30°

[b] Positions with an elevation angle at -45° were changed to -30°, elevation angle at 45° changed to 50°

Pauwels et al. reveal how machine learning can identify the measurement setup in HRTF measurements [230], for all employed source positions, and despite pre-processing aimed at harmonising datasets as well as possible, where any machine learning workflow that involves HRTFs is at risk of generalising badly to other measurement setups. Therefore, cross-dataset testing is of paramount importance. Using Bernschutz KU100 HRTFs as test data for cross-dataset testing could explore if the findings are applicable to HRTF ML applications.

By tracking the loss of the test data during training, overfitting or underfitting with different unforeseen measurements (SADIE II test data) or different measurement methods (Bernschutz KU100) can be observed separately.

Since only the SADIE, IRCAM and Bernschutz datasets have the required measurements for training and validation, it is challenging to produce an accurate model with such a limited variety of HRTF data. Consequently, data augmentation of other HRTF datasets was undertaken to provide some extra data for training and validation. The ARI [19], ITA [20] and RIEC [21] datasets were included with modified angles -

Positions with an elevation angle at -45° were changed to -30°. This modification was also undertaken for the RIEC data set, as well as positions with an elevation angle at 45° changed to 50°. The effect of this data augmentation is demonstrated in Section 4.3.2.

Once all the training and validation HRTFs were concatenated, 50,000 measurements were randomly selected for the training and validation set with an 80:20 ratio, considering practical training time and the limitations of available computer memory (see Section 4.3.2).

To improve the speed and stability of the training process, it is considered good practice to standardise the input data before feeding it into the machine learning model. The standardisation equation is as follows:

$$z = \frac{x - \mu}{\sigma} \tag{4.1}$$

where x is the input data, μ and σ are the mean and standard deviation of the training and validation data, given by:

$$\mu = \frac{\sum x}{N} \tag{4.2}$$

$$\sigma = \sqrt{\frac{\sum (x - \mu)^2}{N}} \tag{4.3}$$

where N is the total number of training and validation data. Note that the same μ and σ should be used for test data.

To summarise, in total there are 230 subjects used in the training, from SADIE I (18 subjects), SADIE II (18 subjects not counting the 2 hold-out data), IRCAM Listen (51 subjects), ARI (60 subjects), ITA (45 subjects) and RIEC (38 subjects). Subjects 19 and 20 from SADIE II and the KU100 measurement from Bernschutz are held out as test sets. Data was standardised before training.

4.3.2 Baseline Model

As mentioned in Section 4.3, there are numerous ML (Machine Learning) models throughout the literature that have the potential for SH interpolated HRTF restoration as they have shown some promising results in similar tasks in the visual domain. Here we aim to find a model that has a simple architecture whilst able to produce viable results. The reason to use a simple model is based on the consideration of the limited number of HRTF datasets - a simpler model is less likely to over-fit the training data. For comparison, all the models in this work were trained with 500 epochs. The majority were trained with an NVIDIA Quadro P4000M GPU with 32GB of Computer RAM (Random-access memory). For the models with an extensive amount of data in subsection 4.3.3, an NVIDIA GeForce RTX2080 Ti with 40GB of Computer RAM was used.

The proposed model can be seen as a simplified version of an inception module of an Inception Network [231]. Separate models for left and right channels are trained individually, whilst the input of the model takes both channels to provide additional information which improves the results as shown in Table 4.4. Here, the Overall Mean is the average MSE loss across all training, validation and test results and the Test Mean

Comparison of results between mono and stereo inputs (lower is better)						
Model	Overall Mean	Training	Validation	SADIE 20	Bernschutz	Test Mean
Baseline (mono)	50.66	31.40	33.94	62.27	75.04	68.65
Baseline (stereo)	44.46	28.17	30.17	52.34	67.17	59.75

Table 4.4: Comparison between stereo input and mono input with the baseline model demonstrating that stereo input performs better than mono input.

is the average MSE loss of SADIE subject 20 and Bernschutz KU100 test data. The phase difference between two channels is handled with dual-band Time Alignment (TA) as discussed in Section 2.1.5. The model input size is 2 × 256 (left and right channels of the interpolated HRTFs with the length of 256 samples per channel), and the output is 256 (either left or right channel).

The model proposed here uses a combination of a Convolutional Auto-Encoder (CAE) and a Denoising Auto-Encoder (DAE) [232]. Preliminary test results showed that the DAE is better with the main contour of the frequency response (Figure 4.3) and CAE is better with the finer details (Figure 4.4). The combination of the two yields positive results. Similar results are also observed in research with image [233].

The results from the convolutional CAE and DAE are concatenated and passed through a fully connected layer for the voting process.

The complete model is shown in Figure 4.5. Note that batch normalisation is performed after each convolution layer and transposed convolution layer, with the exception of the very last transposed convolution layer so the output of the CAE should have a similar magnitude to the DAE. The models are built and trained with PyTorch [234, 235] using smooth L1 loss with Adam optimiser (learning rate: 0.000001, beta 1: 0.9, beta 2: 0.999).

Different loss functions were compared, and the results are shown in Table 4.5. Note that all the losses are calculated with the HRTF values in dB. The mean square error (MSE) loss, also known as the L2 loss, is given by the following equation:

$$\text{L2 Loss} = \sum_{i=1}^{n} (y_{\text{label}} - y_{\text{predicted}})^2 \tag{4.4}$$

where y_{label} is the target value of the output and $y_{\text{predicted}}$ is the prediction from the model. The MSE loss performs worse in the test data but slightly better in the training and validation data. L1 loss, also known as the mean absolute error (MAE), is given by the following equation:

$$\text{L1 Loss} = \sum_{i=1}^{n} |y_{\text{label}} - y_{\text{predicted}}| \tag{4.5}$$

MAE showed key improvements with the SADIE Subject 20 dataset and slight improvements with the Bernschutz KU100 test data. A reason L1 loss outperforms MSE loss might be because L1 loss is usually less sensitive to outliers, such as the case here when there are HRTFs from different databases. Smooth L1 loss is a combination of L1 loss

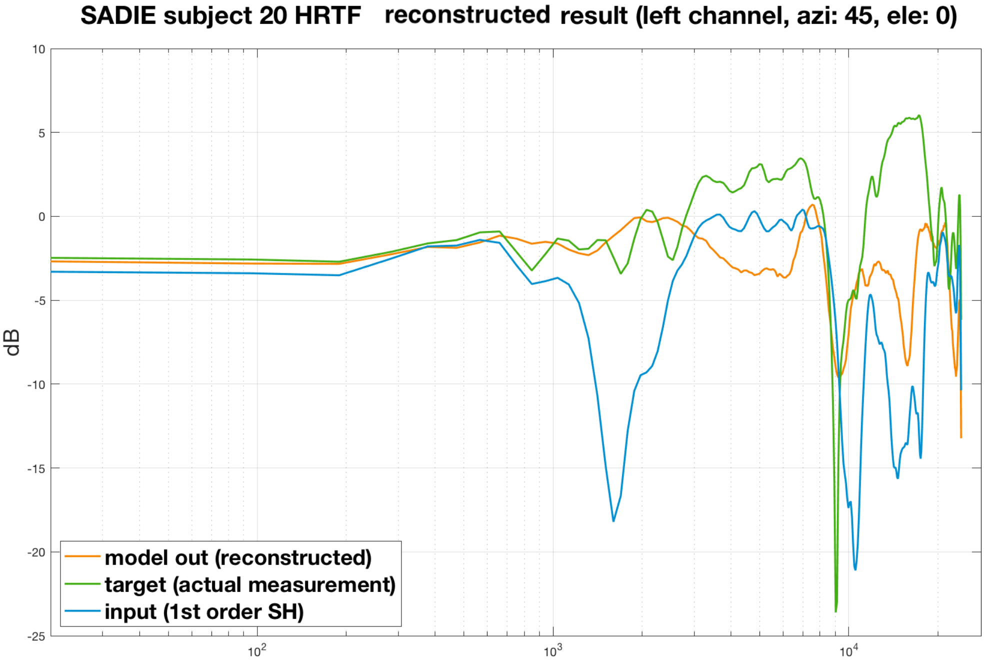


Figure 4.3: Restoration result example for ipsilateral HRTF response with DAE only (orange: model restored output, green: actual HRTF measurement, blue: SH interpolated HRTF.)

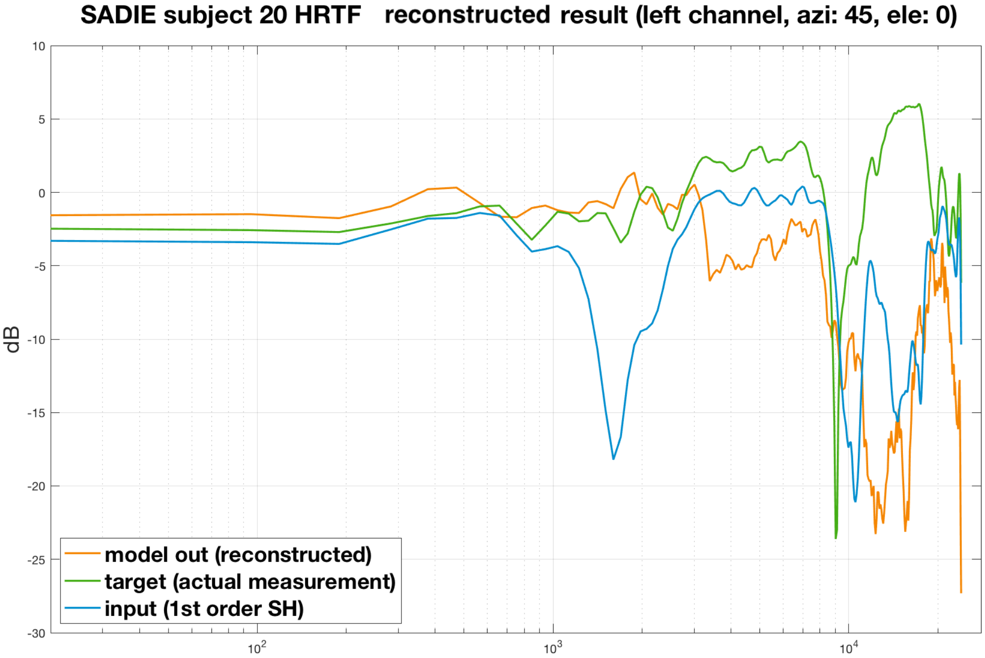


Figure 4.4: Restoration result example for ipsilateral HRTF response with CAE only (orange: model restored output, green: actual HRTF measurement, blue: SH interpolated HRTF).

MSE with different loss functions (lower the better)						
Model	Overall Mean	Training	Validation	SADIE 20	Bernschutz	Test Mean
Baseline (MSE loss)	44.97	27.38	29.43	58.14	64.92	61.53
Baseline (L1 loss)	44.04	28.16	30.14	53.66	64.18	58.92
Baseline (Smooth L1 loss)	44.46	28.17	30.17	52.34	67.17	59.75

Table 4.5: Comparison between different loss functions demonstrating that Smooth L1 loss performs the best in SADIE II test data.

and MSE loss expressed with the following equation:

$$\text{Smooth L1 Loss} = \begin{cases} 0.5(y_{\text{label}} - y_{\text{predicted}})^2, & \text{if } |y_{\text{label}} - y_{\text{predicted}}| < 1 \\ |y_{\text{label}} - y_{\text{predicted}}| - 0.5, & \text{otherwise} \end{cases} \tag{4.6}$$

For an error below 1, Smooth L1 loss performs as the MSE loss function; and for above 1 it performs as an L1 loss. Compared to L1 loss, this method has a continuous derivative at zero, so it provides a smoother gradient when the error gets smaller than 1. The result of Smooth L1 loss further improves the SADIE 20 dataset, but there is a trade-off with the Bernschutz dataset. Considering the real-world application, it is more practical to optimise for unforeseen HRTF measurements of different human subjects instead of different measurement methods with the same artificial head model. The same principle holds in the further optimisation techniques in the upcoming tests. Therefore, the models in this work use Smooth L1 loss as the loss function. Note that this loss function only compares the difference between the model output and target. Whilst they can indicate a model's performance to some extent, they may not represent human perceptual response, although they are easier to back-propagate than perceptual models. Nonetheless, the proposed model will be further evaluated with perceptual models in Section 4.4.

The baseline model has been trained for 500 epochs with a batch size of 8. The reason for using a small batch size is because prior research shows that a larger batch size may produce worse performance [236, 237].

Figures 4.6 and 4.7 show the MSE change during the training. The blue and orange lines are the training and validation results respectively,

The green and red lines are the test sets of Subject 20 from the SADIE II database and Bernschutz KU100 measurements accordingly. The results show that the training and validation results are trending downward while the two test sets flatten out after the first 20 epochs. This indicates that the models are overfitting the training data. Over-fitting is normal in this case considering the limited number of HRTF datasets in the training data (230 in total).

The effects of data augmentation are shown in Table 4.6, where it can be seen that there are some drawbacks with the training and validation data, and a minor drawback with the SADIE Subject 20 test data. However, the extra data provides significant improvement with the Bernschutz interpolation. This demonstrates that the extra

Compare the results with and without extra data(lower the better)						
Model	Overall Mean	Training	Validation	SADIE 20	Bernschutz	Test Mean
Baseline (without extra data)	45.28	18.29	20.55	51.33	90.93	71.13
Baseline (with extra data)	44.46	28.17	30.17	52.34	67.17	59.75

Table 4.6: Mean Squared Error (MSE) with and without data from ARI, ITA and RIEC demonstrating that using additional data improves the result with the Bernschutz test data significantly.

variety of measurements helps the model generalise better across different measurement methods.

The most ideal way to reduce over-fitting is to train with more data. However, given the limited HRTF measurements available, different regularisation techniques can be utilised to improve the baseline result, which will be discussed in this section.

Figures 4.6 and 4.7 show there is some difference between the SADIE hold-out test data and Bernschutz KU100 data although the difference is less on the left channel. However, the average MSE test error of the two channels is approximately the same (left: 59.874, right: 59.633). Further investigations are required to establish the cause of the differences in the left and right channels.

On the other hand, according to Figures 4.6 and 4.7, it is interesting to find that the results with the Bernschutz KU100 data also suffer from over-fitting. As there are different KU100 HRTFs in the training data, and the KU100 measurements in the SADIE II database are very similar to the Bernschutz KU100 measurements, it is unexpected to see the model perform quite poorly when comparing the training and validation results. More oddly, in the later sessions, it shows that the Bernschutz KU100 measurements do not seem to benefit from any regularisation methods. One plausible hypothesis is that the current model only trained with 6 HRTF databases which represent 6 different measurement setups. The model requires a larger variety of inputs to be able to generalise across different measurement setups and methods.

4.3.3 Model Enhancement

Smaller Model

To address the over-fitting problem, an effective method is to decrease the complexity of the model, by decreasing the number of parameters. A smaller model was trained with convolution and transposed convolution layer pair removed in CAE and a NN layer removed from the fully connected NN (Figure 4.5 with the yellow highlighted layers removed). The result does not seem to have a significant improvement on the SADIE II Subject 20 dataset. However, it substantially increased the MSE with the Bernshutuz data from 67.17 to 89.35 (Table 4.7 Model B).

Compare the results from different models (lower the better)						
Model	Overall Mean	Training	Validation	SADIE 20	Bernschutz	Test Mean
A. Baseline	44.46	28.17	30.17	52.34	67.17	59.76
B. Smaller model	45.51	19.54	21.52	51.62	89.35	70.48
C. With weight decay	46.04	28.59	30.96	52.75	71.89	62.32
D. With dropout	46.47	29.14	30.08	54.52	72.15	63.33
E. With weight decay and dropout (proposed model)	45.48	29.85	30.61	**47.21**	74.23	60.72
F. With weight decay and dropout (early stopped at 111 epoch)	49.78	40.92	41.00	**47.18**	70.04	58.61
G. Baseline trained with extra data	39.36	19.74	20.09	59.87	57.72	58.80
H. With weight decay and dropout and trained with extra data	41.44	22.49	22.07	59.69	61.50	60.60
I. Bigger model with weight decay and trained with extra data	**31.38**	**7.83**	**10.61**	56.88	**50.22**	**53.55**

Table 4.7: Comparison of MSE between different models. The table shows that the bigger model with weight decay and trained with extra data (Model I) performs the best with the Bernschutz test data, and also generalises better with different measurement methods. However, for the SADIE II Subject 20 test data, the proposed model (Model E) and the early stopped proposed model (Model F) perform the best across all models.

Model with weight decay

Weight decay is also known as L^2 parameter regularisation or ridge regression. This is a common regularisation method for reducing over-fitting in training. The idea of weight decay is to penalise the large weights in order to simplify the model and reduce over-fitting [238, 239]. This work uses a weight decay rate of 0.001 as an experiment to see the effect of this regularisation method. In theory, a higher weight decay rate should have a stronger regularisation effect and 0.001 is a reasonable value to start testing with weight decay [240, 241].

The result does not seem to have any positive impact on the SADIE II Subject 20 dataset, and there is the main drawback with the Bernschutz KU100 data as the error increases from 67.17 to 71.89 (Table 4.7 Model C).

Model with dropout

Dropout randomly “drops out” a percentage of nodes in the neural network during training. The idea is to avoid co-adaption between nodes by never guaranteeing that any pair will both be used during the training process to avoid the model over-relying on a few nodes within a layer [242–244]. It can also be seen as randomly sampling from the exponential number of possible narrow sub-networks during training, then providing an average of the performance of all these combinations in test time or application. Note that the dropout layer can only apply on fully connected layers but not convolutional layers. The model uses a 20 per cent dropout ratio on the second to fourth fully connected layer.

According to Table 4.7 Model D, this method produces worse results with the test data compared to the baseline model, especially with the hold-out SADIE II data, but performs slightly better with the validation data. However, such slight differences may be introduced by the randomness of machine learning training. According to the result, dropout seems to have a more negative impact on regularisation compared to weight decay. In theory, it is possible to increase the dropout ratio or use a different configuration to increase the regularisation effect. However, according to the current result, tuning the dropout ratio does not seem to be a promising approach to improving the overall performance.

Combining weight decay and dropout

Combining weight decay and dropout shows the best result in the hold-out SADIE II data despite there being a noticeable trade-off with the Bernschutz dataset. This result in Table 4.7 Model E indicates that by combining weight decay and dropout, the model can generalise better across different unforeseen HRTF subjects (SADIE hold out) but not the measurement method (Bernschutz). As this model performs the best with the SADIE II test data, this will be the proposed model to be further analysed in Section 4.4.

It is interesting to find that the combination of the two different methods shows a large difference in results, but not with either method individually. It is not clear whether the improvement comes from the combination of the techniques or it is from the cumulative regularisation power. This could be an individual research topic to be investigated in the future.

Early stopping

Early stopping is one of the regularisation methods that sometimes is not considered good practice in machine learning training because it breaks the principle of orthogonalisation and makes hyper-parameter tuning difficult [245]. Another reason this method is controversial is that the result can be hard to reproduce and compare across different models. However, according to the learning curve from the model combining weight decay and dropout in Figure 4.8 and 4.9, early stopping should perform slightly better with the test data, especially with the test set from SADIE II. In order to demonstrate the effect, the proposed model was retrained and stopped the training at 111 epochs as it is the lowest point in Figure 4.8 and 4.9.

The result in Table 4.7 Model F shows that there is a very slight improvement with the SADIE II Subject 20 test set and a more noticeable improvement with the Bernschutz KU100 test data.

Training with more data

To shorten the training time to compare across different methods and considering the limited size of RAM, the models discussed above were trained with 50,000 randomly sampled HRTFs from different angles of the training and validation HRTF sets. However, the best way to reduce over-fitting is to increase the size of the training set. To investigate what the model is capable of with more data, a baseline model was trained with 633,000 HRTF measurements. Training this amount of data can take a lot of time per epoch. To speed up the process, the batch size for training increased to 32, whilst the validation and test sets remained the same at 8 for better comparison.

Two models, the baseline model (Table 4.7 Model G) and the model with weight decay and dropout were trained with extra data (Table 4.7 Model H). The baseline model trained with extra data showed major improvement with the Bernschutz dataset, alongside the training and validation sets. However, there is also a noticeable trade-off with the SADIE II Subject 20 test data. It is believed that the improvement in the Bernschutz dataset is the result of the model having more examples of the KU100 HRTFs with different measurements at different angles so that it can generalise the measurement method better. The trade-off in the SADIE II Subject 20 test data may be caused by the increased batch size, which can induce worse performance [236, 237]. Nevertheless, as the extra data is within the same distribution, it is quite unlikely it could provide any noticeable performance improvement with unforeseen HRTF measurement subjects.

Bigger model

Considering the current results and the limited number of labelled data, training a bigger model is against normal machine learning practices.

To demonstrate the potential capability of the proposed method and insight for future research, a slightly deeper model was also trained with extra data. The goal here was to minimise the training and validation error as much as possible, neglecting the trade-off in test datasets' results. To balance out the model size and training time, only the convolution neural network was changed. An extra convolution layer and transposed convolution layer pair were added to the convolution model.

As this model only focused on the test and validation results, dropout and weight decay regularisation methods were lifted, which defeats the purpose of using a bigger neural network. The model was trained with 633,000 HRTFs with a batch size of 32 for training similar to section 4.3.3 as bigger models usually work better with more data.

Compared to the baseline model, the training time of each epoch from the bigger model (Model I) increased from 6 minutes to 26 minutes with the setup discussed in section 4.3.2. The model was trained with 500 epochs, and the results are shown in Table 4.7. As expected, the model provides a huge improvement in training and validation, but not much improvement in the SADIE II test data. On the other hand, it provides the best performance for the Bernschutz data. Comparing the results of the smaller model, baseline model and the bigger model, it seems like the bigger model has better performance with the Bernschutz data which indicates it generalises better across different measurement methods.

Summary

According to the results in Table 4.7, it is clear that there is room for improvement through some enhancements of the baseline model. The baseline model with weight decay and dropout (Model E) and baseline model with weight decay, dropout and early stopping (Model F) provides the best results with SADIE II test data, yet the bigger model with weight decay (Model I) generalises better across different measurement methods.

However, as mentioned in Section 4.3.2, this work focuses on optimising for unforeseen HRTF measurements of different human subjects instead of different measurement methods with the same artificial head model, therefore the baseline model with weight decay and dropout (Model E) is the proposed model in this work.

On the other hand, according to the trends seen across the smaller model (Model B), the baseline model (Model A) and the bigger model (Model I), the performance for unforeseen measurement methods like the Bernschutz KU100 test data is expected to improve when the model size increases.

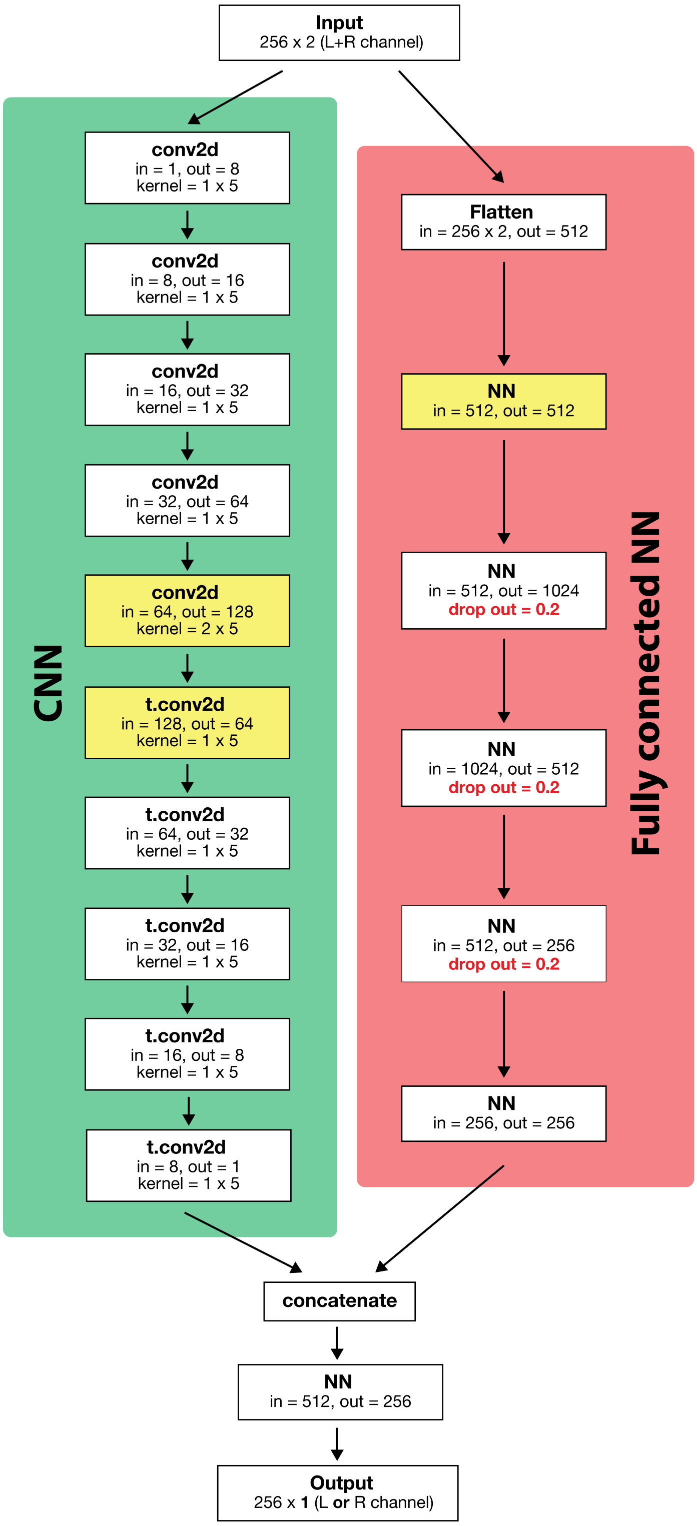


Figure 4.5: Baseline model, smaller model (smaller model removes the layers highlighted in yellow) and proposed model (proposed model uses drop out in some NN layers, shown in red. Amongst other techniques discussed in Section 4.3.3)

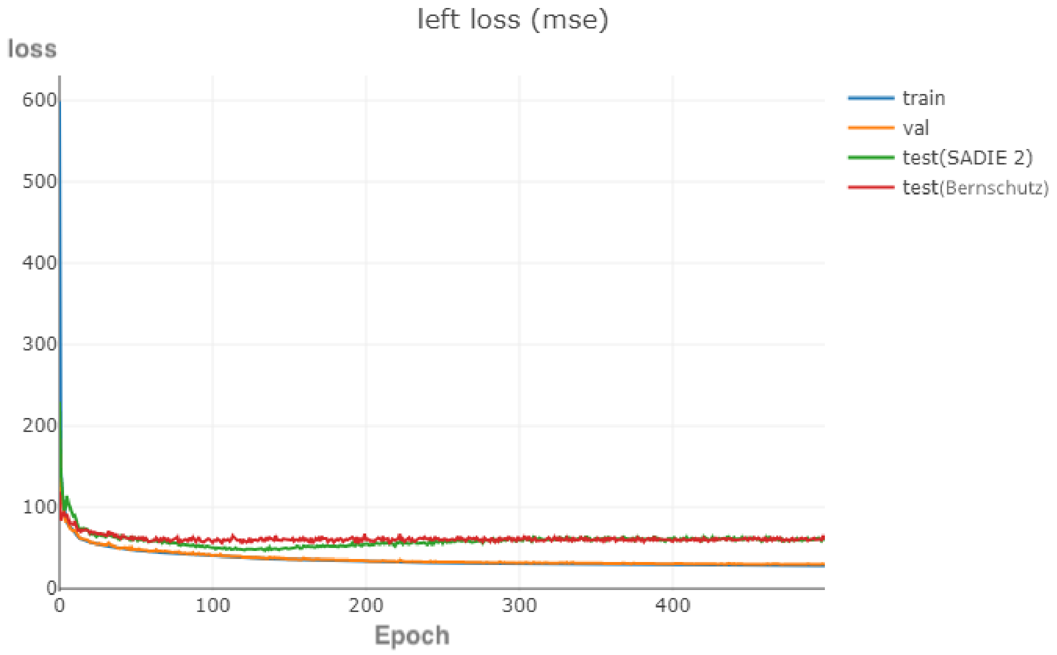


Figure 4.6: Left MSE during training (blue: training loss, orange: validation loss, green: hold out test data Subject 20 from SADIE II, red: Bernschutz KU100), which shows that the model is over-fitting the training data

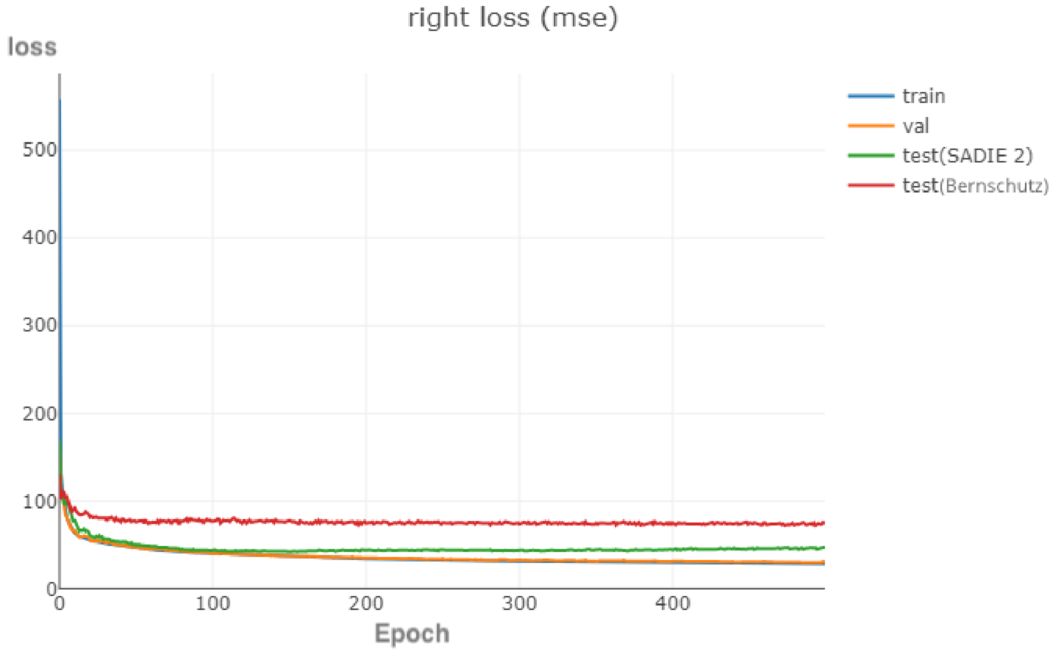


Figure 4.7: Right MSE during training (blue: training loss, orange: validation loss, green: hold out test data Subject 20 from SADIE II, red: Bernschutz KU100), demonstrating that the model is over-fitting the training data

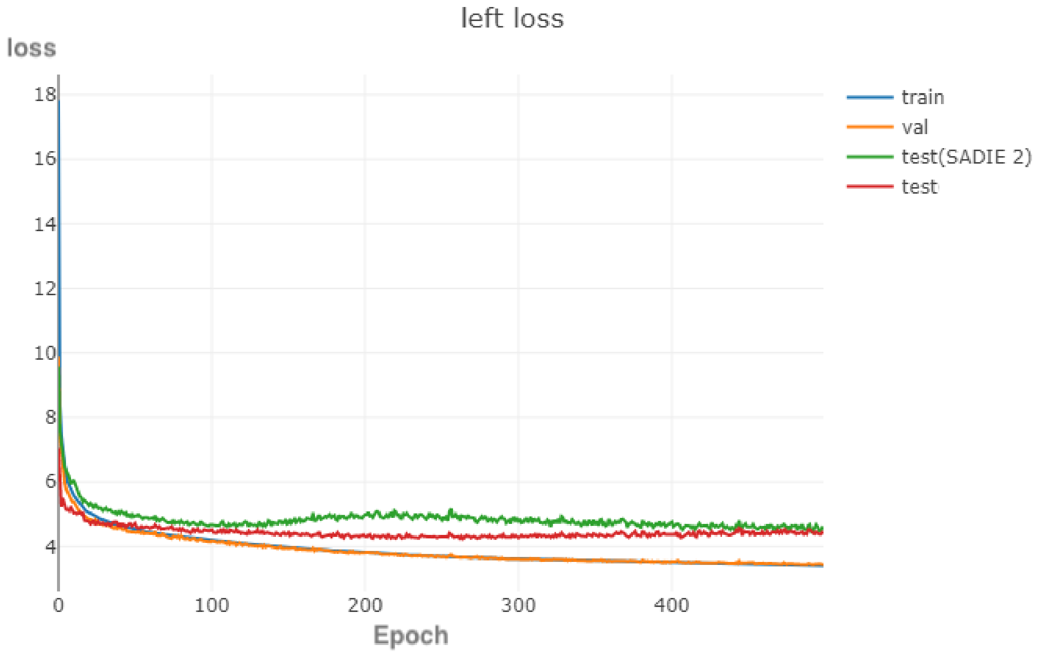


Figure 4.8: Left channel L1 loss during training (blue: training loss, orange: validation loss, green: hold out test data Subject 20 from SADIE II, red: Bernschutz KU100), which shows that the lowest point in the curve is at around 110 epoch

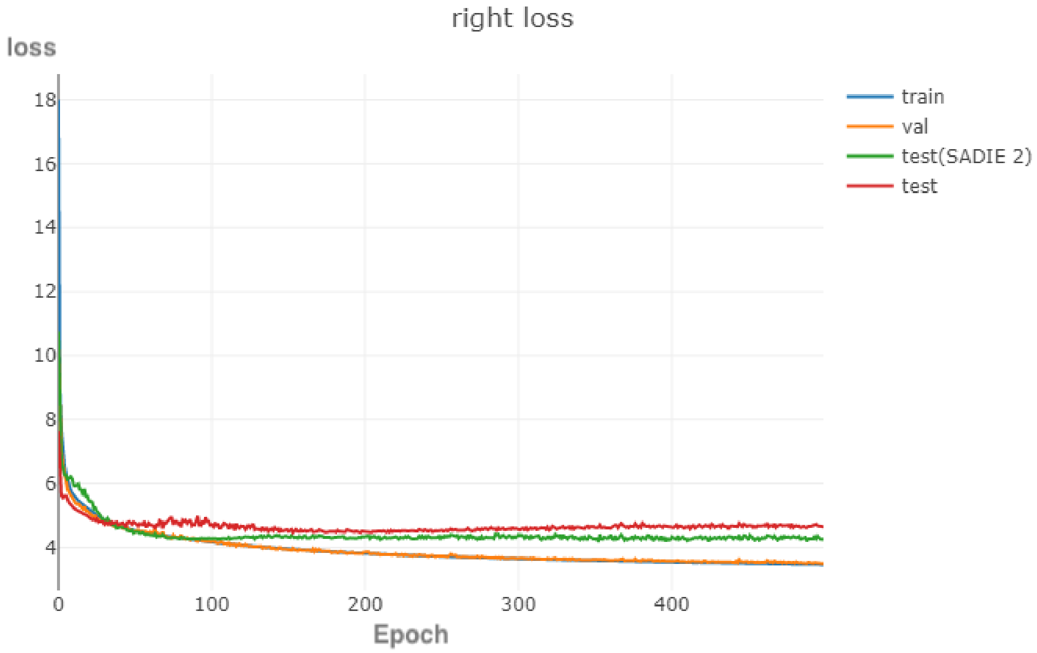


Figure 4.9: Right channel L1 loss during training (blue: training loss, orange: validation loss, green: hold out test data Subject 20 from SADIE II, red: Bernschutz KU100), which shows that the lowest point in the curve is at around 110 epoch

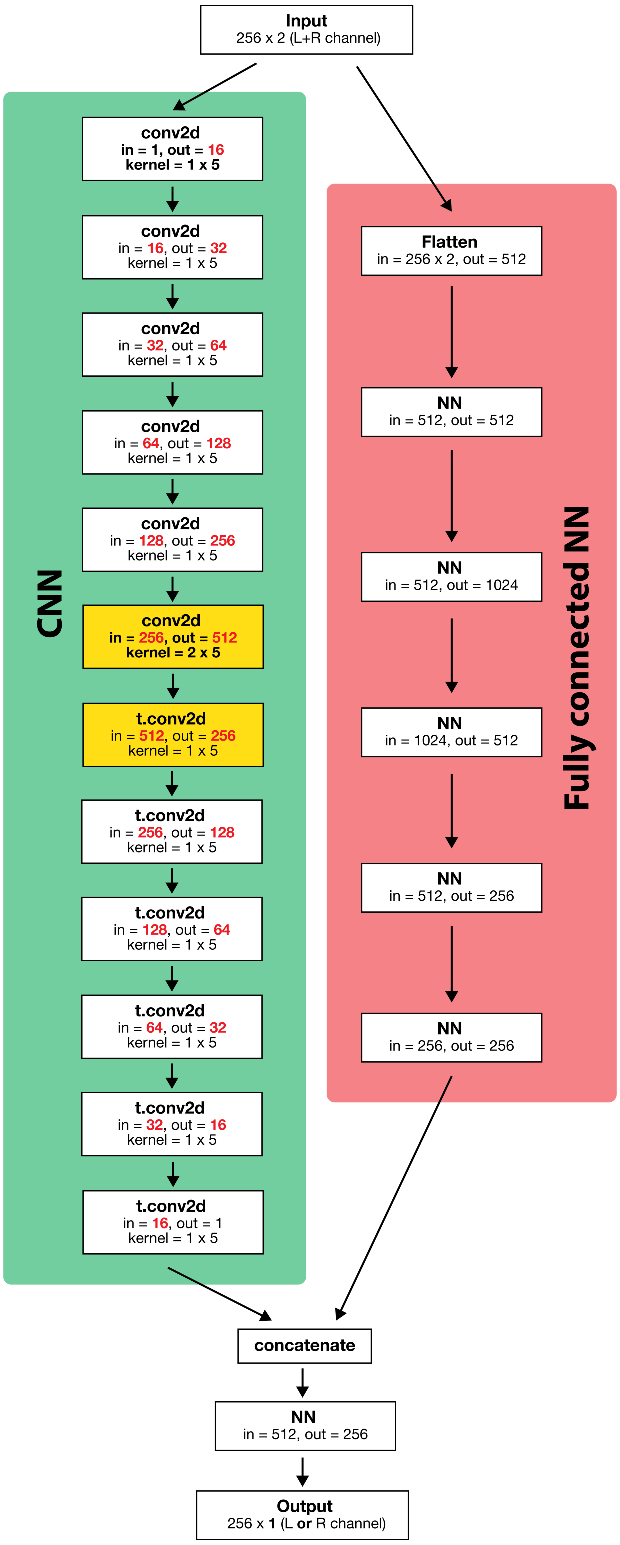


Figure 4.10: Wider and deeper model (highlighted the main difference compared to the proposed model)

4.4 Evaluation

As a proof of concept, this work did not engage in comparison with other methods. Given that none of the proposed models were in their optimal state, nor did they exhibit evidence of readiness for real-world applications, such comparisons may not have provided meaningful insights. However, a basic evaluation focusing on perceptual differences and localisation performance does offer some additional perspective on this method. In this section, the results of the proposed model, including weight decay and dropout, are further analysed for perceptual difference and localisation performance. Perceptual models based on these two criteria are used in order to provide more robust results for benchmarking. This work utilised the Perceptual Spectral Difference (PSD) model [246], alongside May's model and Baumgartner's model in the Auditory Modelling Toolbox (AMT) [247–249] for evaluation. These models were selected due to their well-established nature and their suitability for the objectives of this work at the time of its publication.

Recently, Engel et al. proposed advanced methods for evaluating HRTF sets [250]. Despite their potential benefits, these methods were not integrated into this work, given its state at the time. This acknowledgement serves not only as a recognition of the continuous evolution of the field but also as an anticipation for potential future work which might incorporate these advancements. Considering the explorative nature of this thesis, it was a conscious choice to limit the evaluation tools used to optimise clarity. This study could serve as a solid foundation for future exploration and a bridge towards the adoption of emerging techniques.

4.4.1 Perceptual Spectral Difference

To formally estimate the perceptual performance, the results were further analysed with a Perceptual Spectral Difference (PSD) model [246]. This model calculates the difference between two binaural signals or HRTFs and presents a more accurate perceptual comparison of spectral differences as PSD. It was the state-of-the-art method to evaluate the difference between two binaural signals or HRTFs when this work started. This work compares the difference between before and after the restoration process with the actual HRTF measurements.

The comparison between mean PSD before and after reconstruction with different HRTF datasets is shown in Table 4.8 and Figure 4.11 with the minimum and maximum PSD plotted. The results show that the model provides significant improvement in PSD across all datasets as the mean PSD is lower. However, the minimum and maximum in Figure 4.11 shows that the model seems to introduce higher PSD error in some cases. As for most applications that use HRTFs, the smoothness across all angles is more crucial than the average performance. A more detailed plot across different angles is shown in Figure 4.13, 4.14, 4.15. Further analysis with the box plot in Figure 4.12 shows that although the model may introduce more extreme outliers with unforeseen HRTF measurement subjects (SADIE Subject 19 and 20), the model still improves the majority of the HRTFs and reduces the Interquartile Range (IQR) in the result.

According to Figure 4.11 and 4.12, Subject 19 from the SADIE II database has a worse maximum PSD and many outliers. To further investigate the cause of the result, Figure 4.13 shows the PSD before and after the comparison of different angles in

	SADIE 18 (training data)	SADIE 19 (hold out)	SADIE 20 (hold out)	Bernschutz KU100
PSD (sones) (SH input)	3.03	3.05	2.84	2.57
PSD (sones) (model output)	1.93	2.12	1.96	1.61
Frontal azimuth mean error (SH input)	20.81	25.36	30.00	39.67
Frontal azimuth mean error (model output)	19.29	17.47	15.84	18.98
Sagittal RMS error (deg) (SH input)	40.7	38.6	37.5	38.1
Sagittal RMS error (deg) (model output)	44.3	43.7	39.3	41.4
Sagittal quadrant error (%) (SH input)	11.5	9.1	7.6	7.1
Sagittal quadrant error (%) (model output)	24.8	25.2	14.7	12.4

Table 4.8: Predicted model performance with various HRTF sets

Subject 19 from the SADIE II database. The left is the SH interpolated HRTFs before restoration and the right is the one after being processed with the ML model. The figure shows that most of the high PSD results were introduced in the lower frontal region. It is unclear what caused the increased error, but one hypothesis is that the abnormality was caused by the shadow effect from the knees, as the SADIE II HRTFs along with most of the other databases are measured with subjects sitting on a chair.

However, Figure 4.14 shows the PSD before and after comparison with the holdout Subject 20 from the SADIE II database. Besides a small area of minor PSD increment in the very low frontal region, the restored result shows there is no significant abnormality in any region. To have a deeper understanding of the cause of the abnormality in Subject 19 holdout data, extra tests with more HRTF sets are required.

Figure 4.15 shows the PSD before and after comparison with the Bernschutz KU100 data. The model seems to perform better with the unforeseen measurement method, as it shows improvement in the PSD at all angles. It is worth noticing that the lower frontal region in the figures does not have any oddly high PSD results, perhaps because KU100 is a head-only dummy head model.

4.4.2 Localisation performance

In machine learning, having a quick and robust objective benchmark is essential for developing and comparing the performance of various models. An ideal benchmark should be reliable globally, enabling researchers from around the world to compete towards the same goal. Rather than a perceptual test, which can be utilised before deploying the model in real-world applications, a computational model would be more suitable for providing consistent and reproducible results. This approach ensures that the evaluation process is less subjective, less time-consuming, and less resource-intensive, facilitating faster progress in model development and comparison across the research community. The well-established May's model and Baumgartner's model in the Auditory Modelling Toolbox (AMT) [247–249] were selected for evaluating the localisation performance. May's model was developed for frontal azimuth localisation on the horizontal plane, and Baumgartner's model was developed for the sagittal plane. May's model provides an estimation of the azimuth localisation error, whilst Baumgartner's model provides estimations of Root-Mean-Square (RMS) error and quadrant error in percentage. RMS error represents the RMS of the sagittal localisation estimation versus the ground truth angle. The quadrant error in percentage was defined as the percentage of sagittal localisation errors that were larger than 90°.

Table 4.8 showed the localisation results of the proposed model. The frontal azimuth localisation means error from May's model show improvement in all HRTF datasets. However, for the frontal sagittal plane with Baumgartner's model, results in RMS error and quadrant errors indicate all the reconstructed HRTFs seem to perform worse in the frontal sagittal plane localisation.

The current model suffers in localisation tasks may because it used smooth L1 loss as the loss function. The smooth L1 loss only focuses on the magnitude difference at each frequency point. There may not be any meaningful connection between those magnitude differences and localisation performance. Therefore, it is no surprise that the model failed to optimise the HRTFs restoration for the localisation performance. Future models could add some types of localisation error into the loss function to improve the localisation results.

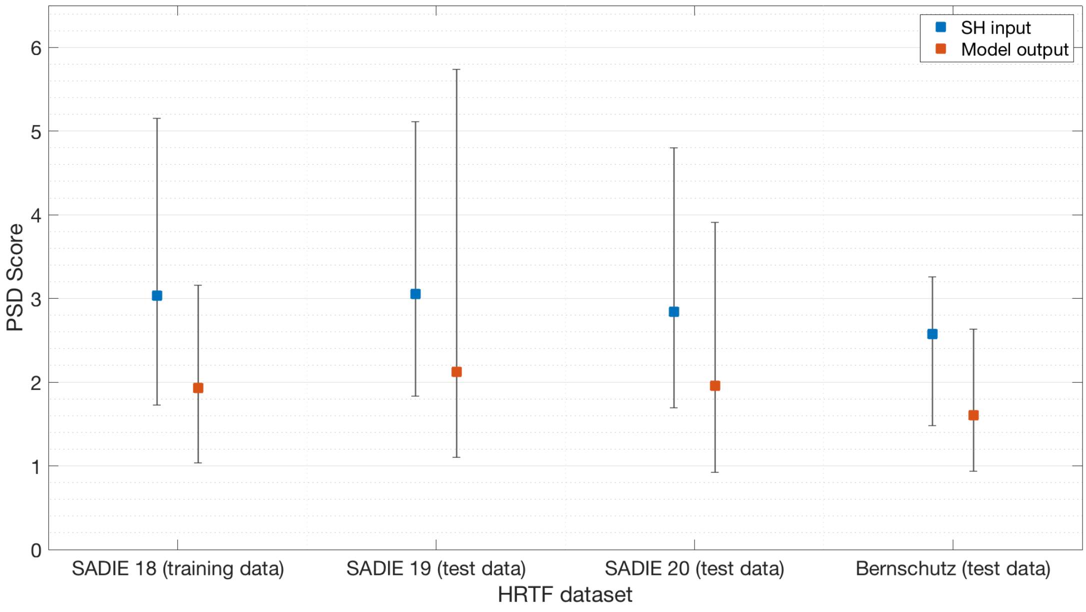


Figure 4.11: Minimum, maximum, and average PSD across different angles in different data sets, which shows the model significantly improved the mean PSD from the non-reconstructed interpolated data.

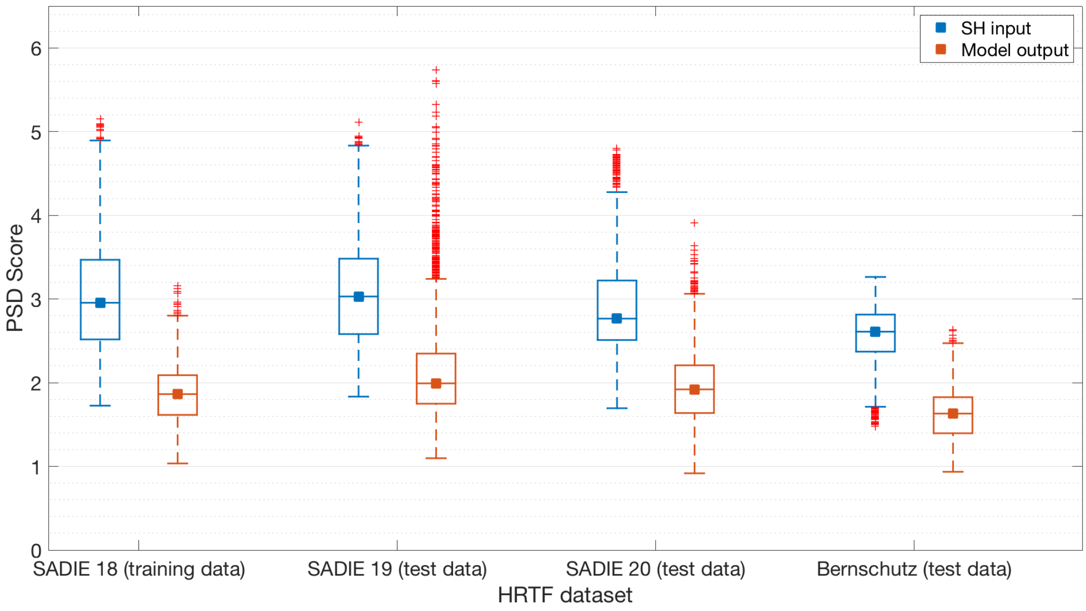
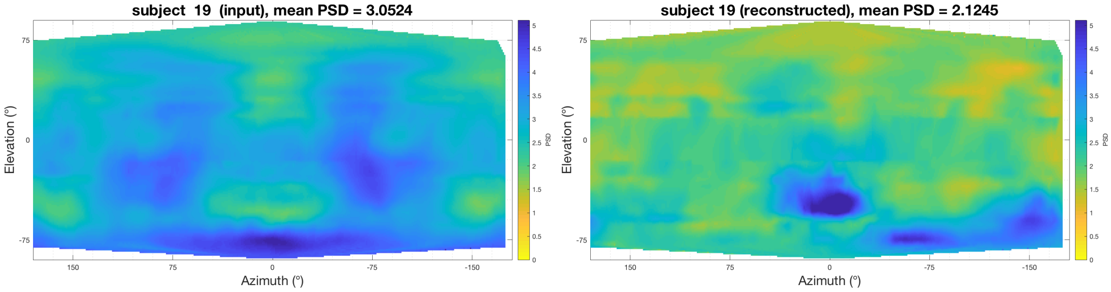


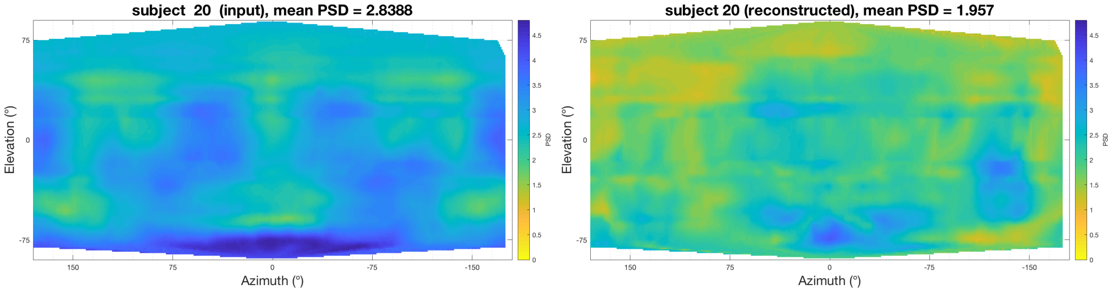
Figure 4.12: PSD median and box plot with whiskers with maximum 1.5 IQR, which shows the model reduces the interquartile range (IQR) significantly.



(a) before reconstruction

(b) after reconstruction

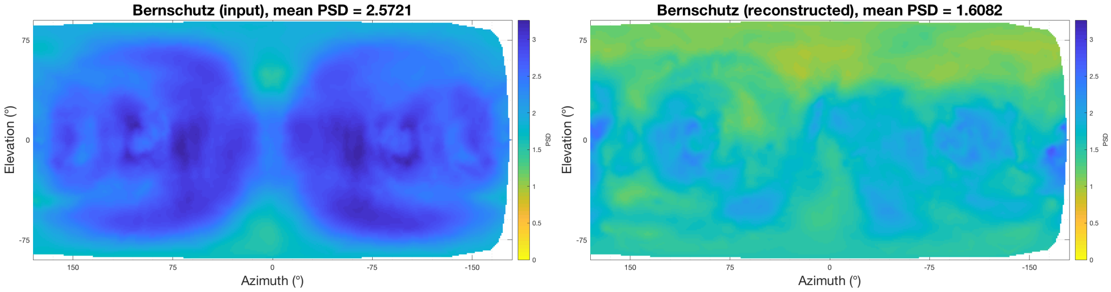
Figure 4.13: Comparing the PSD of Subject 19 from the SADIE II database before and after reconstruction which shows a noticeable improvement in most regions, besides the very low frontal region.



(a) before reconstruction

(b) after reconstruction

Figure 4.14: Comparing the PSD of Subject 20 from the SADIE II database before and after reconstruction shows a noticeable improvement in most regions.



(a) before reconstruction

(b) after reconstruction

Figure 4.15: Comparing the PSD of the Bernschutz KU100 dataset before and after reconstruction shows a noticeable improvement in most regions.

4.5 Discussion

This work aims to prove the hypothesis that machine learning can be used to restore distorted interpolated HRTFs. To draw a convincing argument, this Chapter picked one of the more challenging situations based on 1^{st} order SH and 6 measurements. Models with higher order SH and more measurements should have better performance than the current one as less data needs to be restored.

The results show that a simple ML model can be used to restore distorted SH interpolated HRTFs, although the current state of this model is far from optimised for this application. It is believed that there will be a significant improvement if more HRTF measurements are available for training in the future. Under the current situation, one way to improve the model is through hyper-parameter tuning, including the parameters for regularisation.

An alternative method that may be possible to reduce over-fitting is to use data augmentation. HRTF measurements are expensive and tedious and therefore it is not very likely there will be a huge increase in HRTF measurement data in the near future. To augment the current dataset, one possible way is to use more different sparse HRTF configurations or different SH orders to train the model. Table 4.6 shows that even if the extra data is not perfect, it is possible that it can still improve the model performance in some cases.

Similar to data augmentation, noise injection is a different regularisation method that has been shown to work better than weight decay in some cases [251–253]. By picking the right parameters, it is believed that it could generalise better across measurement methods as the model could focus on the general information across various HRTF measurements as opposed to the artefacts introduced by different measurement methods.

Another problem that was observed from the current model is the localisation performance. Although it shows some improvement in the horizontal plane, the sagittal error needs improvement. As discussed in Section 4.4, it is believed that the smooth L1 loss function only compares the difference in each frequency and fails to capture other useful metrics of HRTFs. Potentially, some custom loss functions can be implemented to improve the model. Gatys et al. [254] trained a separate ML model for content loss and implemented a special function for style loss. Similar methods, like training a localisation model as a localisation loss function, may be able to solve the problem. Furthermore, with the recent success in generative adversarial networks (GAN), it should be possible to build a GAN based on localisation performance [16, 18, 255–257]. However, as a GAN can be unstable to train and it usually requires a lot of tuning, it may not be the most effective way for SH interpolated HRTF restoration.

This work also tries to show a general insight into using machine learning for HRTFs reconstruction. According to Table 4.7, Table 4.8 and the discussion in session 4.4, to apply the idea in real-life applications, optimising the model performance for specific narrative tasks should yield better performance. With a more specific application in mind, not just the parameters of the model can be changed; the model can also be trained with cleaner or more particular data specialised for the task.

Given the swift advancements in the machine learning field, both in terms of algorithms and hardware, it is important to understand that this work could be deemed obsolete due to the emergence of more efficient methods and powerful hardware. With the state of current machine learning development, it is possible to fit an entire HRTF set into a single GPU memory. This allows work to be trained using an entire HRTF dataset as output as long as enough uniform HRTF sets are available. With the sparse HRTF set employed directly as input, this could yield a more accurate result and faster performance speed.

4.6 Conclusion

HRTF interpolation in the SH domain often suffers from distortion in the high frequencies. With the recent development in machine learning algorithms, this paper has shown that it is possible to restore the distorted SH interpolated HRTFs with an ML model. Although the proposed method suffers from over-fitting, it still shows improvements in perceptual difference and localisation performance. It is believed that with more training data, the model performance can be vastly improved. However, HRTF measurements can be difficult and time-consuming to obtain. In the next Chapter, Chapter 5, the HUman Morphable Model-based Numerically Generated Binaural Impulse Response Database (HUMMNGBIRD) HRTF database comprising over 5000 HRTF sets was created. It uses a novel method that can simulate unlimited realistic HRTFs with different configurations. With that being said, even without the extra data, the model has the potential to work well for some narrative use cases in real-world applications with some extensive hyperparameter tuning. For future reference, the acronym of the models created in this research is Lo-SHaRe Model (Low-order Spherical Harmonic Restoration Model). Supporting data and code are available at GitHub: `https://github.com/Benjamin-Tsui/SH_HRTF_Restoration`.

Chapter 5

Generating HRTFs with a 3D morphable model of human heads

5.1 Motivation and Challenges

As mentioned in the previous chapters, ML models can often improve with more data. The SH HRTF Restoration model introduced in the last chapter shows that it suffers from overfitting, likely due to the lack of training data. The results from the last chapter indicated that the number of HRTFs by consolidating the currently available HRTF data is not quite enough. Also, with some cleaner, more uniform training data, the model could be more accurate and more robust. However, there are a few challenges to generating a huge amount of HRTF sets. Firstly, it is hard to gather a large number of participants for HRTF measurements or gather enough head scans for HRTF simulations. Secondly, it is ideal to have a dense grid of measurements that is future-proof, which can take up any other ML tasks in the future. It is difficult for humans to withhold the relatively long process of HRTF measurements for a dense grid. It is also time-consuming to do the measurements on a large scale. Thirdly, as discussed in Section 2.1.3, the diversity of participants for HRTF measurement is often bounded by the geographical location of the setup. Fourthly, at the time this work started, there was no efficient way to streamline the HRTF simulation process.

This chapter proposes a method using Three-dimensional Morphable Models (3DMMs) to generate numerous head models and then using the Boundary Element Method (BEM) software Mesh2HRTF [87] to simulate HRTFs from the generated models. 3DMMs are powerful statistical tools for representing the 3D shapes and textures of an object class. Recent developments of such models can generate human heads that include the face, cranium, ears, eyes, teeth and tongue. This paper used one of the 3DMMs developed by [258] to generate head models and simulate HRTFs with those heads. The 3DMM has been constructed based on sampling of a wide range of human heads and faces, of different races and ages. After pre-processing the generated head models, the Boundary Element Method (BEM) software Mesh2HRTF [87] was used to simulate the HRTF sets. With the Viking computer cluster provided by the University of York, on average, it takes 36 to 44 hours to simulate each 1-degree Gaussian grid HRTF set. This work established a streamlined workflow to generate an unlimited amount of different HRTF sets.

As explained in 1.2, the primary contribution of this work lies in merging state-of-the-art Three-dimensional Morphable Models (3DMMs) of human heads [258] with the Boundary Element Method (BEM) [87], enabling the simulation of HRTFs from these generated models. This approach, arguably, is the sole pathway to generating a large, uniform HRTF database with diverse human heads. Both the 3DMMs of human heads [258] and the BEM software Mesh2HRTF [87] have been thoroughly discussed and evaluated in their respective papers. In contrast to the work by Stitt et al. [58], no morphology is incorporated in this process. Although a script could be developed to extract such features, the time cost and the further slowdown of an already intensive process do not justify this course of action. It is also believed that even in the worst-case scenario, where future research might find the generated HRTF unreliable, it should still prove invaluable for pre-training ML models. This work is essentially a proof of concept, and it is understood that improvements are achievable as more advanced head models become available or better BEM simulation workflows are developed.

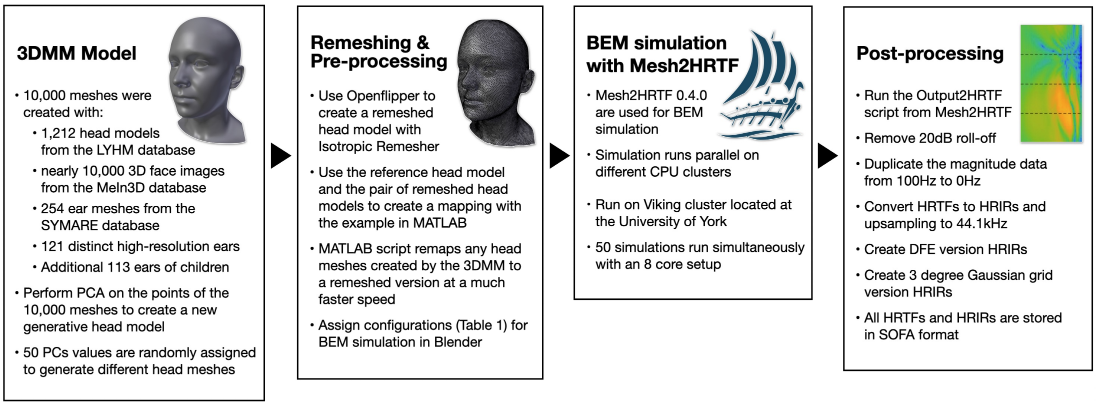
This chapter is organised as follows: Section 5.2 will discuss the method used in this study, including the 3DMM, pre-processing workflow, BEM simulation with Mesh2HRTF set-up, post-processing workflow and output files. Section 5.3 evaluates the generated HRTFs and its similarity to actual HRTF measurements. Section 5.4 concludes the Chapter.

5.2 Method

The method can be broken down into four parts, as shown in Figure 5.1. Firstly is to generate head models with 3DMM. Secondly, the head model will be remeshed and pre-processed for BEM simulation. Then, the processed head model will feed into the Mesh2HRTF software for BEM simulation. The Mesh2HRTF version that is used in this work is Mesh2HRTF 0.4.0, as it was the most up-to-date version at the beginning of the project [87]. Finally, post-processing will be performed to check for errors, create uniform HRTF and HRIR results, and apply Diffuse Field Equalisation (DFE). The generated HRTFs and their equivalent HRIRs will be stored in SOFA format alongside the head model and its 3DMM latent variables.

5.2.1 Generating Head Models with 3DMM

Three-dimensional Morphable Models (3DMMs) are statistical models that represent different 3D shapes and textures of an object class. These models are commonly used in computer vision, computer graphics, and medical imaging [259–262]. Ploumpis et al. created a 3DMM of the human head [258], which is used in this work to generate human head models. The 3DMM was created with a combined total of 1,212 head models, nearly 10,000 face models and 254 ear models. The head model templates contain the ears from the Liverpool-York Head model (LYHM) [263]. The Large-scale Face Model (LSFM) utilised nearly 10,000 face scans from the large MeIn3D database captured during a special exhibition in the Science Museum, London [264]. The ear models are comprised of left and right ears of 10 individuals from the SYMARE database [90], 113 ears of children acquired by CT scan, and 121 distinct high-resolution ears of 64 individuals. The 64 individuals consisted of 32 men and 32 women ranging from 20 to



3DMM Model

- 10,000 meshes were created with:
  - 1,212 head models from the LYHM database
  - nearly 10,000 3D face images from the MeIn3D database
  - 254 ear meshes from the SYMARE database
  - 121 distinct high-resolution ears
  - Additional 113 ears of children
- Perform PCA on the points of the 10,000 meshes to create a new generative head model
- 50 PCs values are randomly assigned to generate different head meshes

Remeshing & Pre-processing

- Use Openflipper to create a remeshed head model with Isotropic Remesher
- Use the reference head model and the pair of remeshed head models to create a mapping with the example in MATLAB
- MATLAB script remaps any head meshes created by the 3DMM to a remeshed version at a much faster speed
- Assign configurations (Table 1) for BEM simulation in Blender

BEM simulation with Mesh2HRTF

- Mesh2HRTF 0.4.0 are used for BEM simulation
- Simulation runs parallel on different CPU clusters
- Run on Viking cluster located at the University of York
- 50 simulations run simultaneously with an 8 core setup

Post-processing

- Run the Output2HRTF script from Mesh2HRTF
- Remove 20dB roll-off
- Duplicate the magnitude data from 100Hz to 0Hz
- Convert HRTFs to HRIRs and upsampling to 44.1kHz
- Create DFE version HRIRs
- Create 3 degree Gaussian grid version HRIRs
- All HRTFs and HRIRs are stored in SOFA format

Figure 5.1: Full pipeline for the proposed method, from head models that are generated from 3DMM to the HRTFs of the head models

70 years old. High-resolution ears are obtained by scanning the inner and outer areas of both ears of the individuals with a light-stage apparatus. The LYHM and MeIn3D databases have a wide variety of ages from under 5 to above 80 and are nearly balanced in gender. Combining the data, 10,000 meshes were created, and Principal Component Analysis (PCA) was then performed on points of the meshes to create a new generative head model. For further detail on the creation of the 3DMM of the human head, the reader is directed to [258].

The final model used in this research has 50 Principal Component (PC) values which can be randomly assigned to generate different head meshes.

The models that are used in this work are head only. Full head and torso models were not used due to the consideration of simulation time since the extra vertices on the torso will take a significantly longer time for each HRTF simulation. Moreover, some studies indicate that the torso's reflection and shadowing effect can be added back in when necessary [67, 68, 265].

5.2.2 Remeshing and Pre-processing

The Mesh2HRTF software provides a Multi-level Fast Multipole Method Boundary Element Method (ML-FMM BEM) with reciprocity simulation, which is useful for simulating a large amount of HRTFs due to its reasonably fast speed. The original output from the 3DMM has an uneven vertices distribution that is not ideal for BEM simulation. Some areas of the mesh do not require high-resolution detail, e.g. the back of the head and cheek area, and their edges may exceed the maximum edge length for the target maximum frequency. Although these edges have very little to no effect on the results as they are not on the ears, it is still good practice to remesh the model, so it has fairly even edges. Isotropic Remesher in the software Openflipper allows the user to set the maximum edge length and remesh as uniformly as possible to the target edge length. Due to the number of head meshes involved in this work and to make the remeshing process more efficient and robust, a reference head model is fed into Openflipper to create its equivalent remeshed head model instead of remeshing each

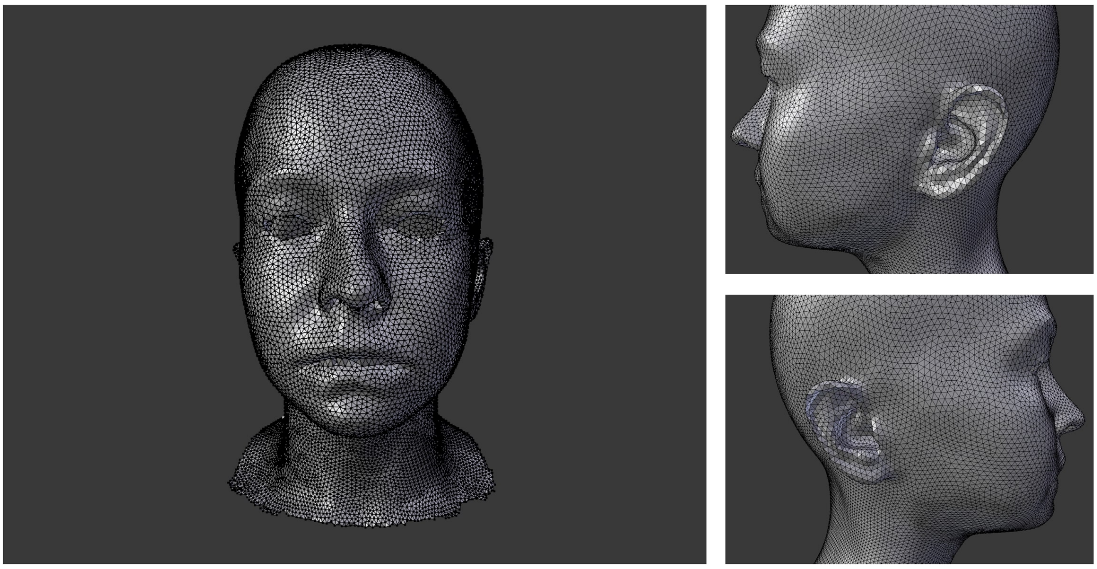


Figure 5.2: 3DMM generated head mesh example 1

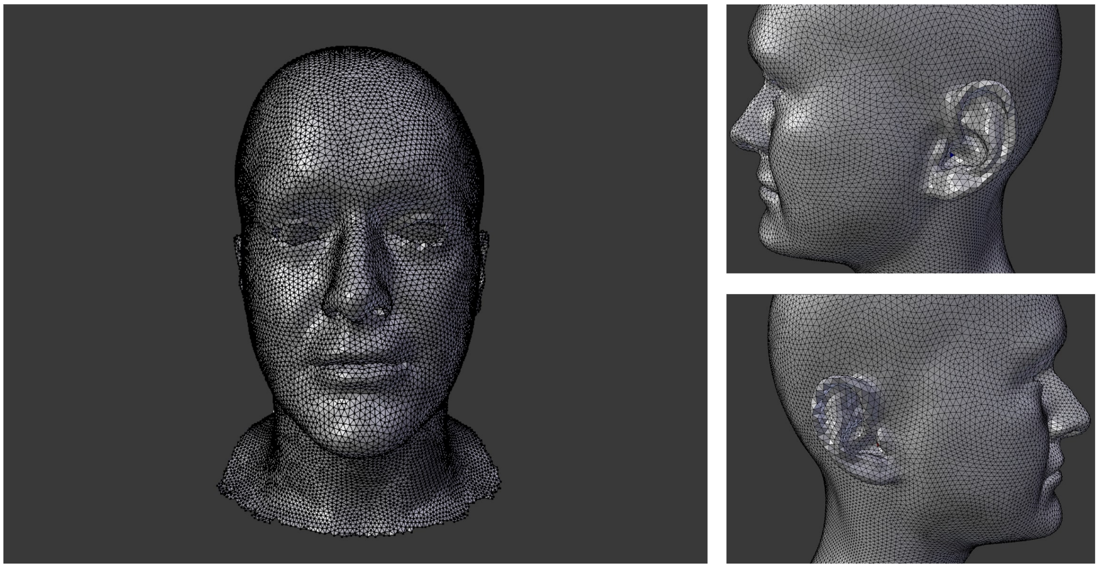
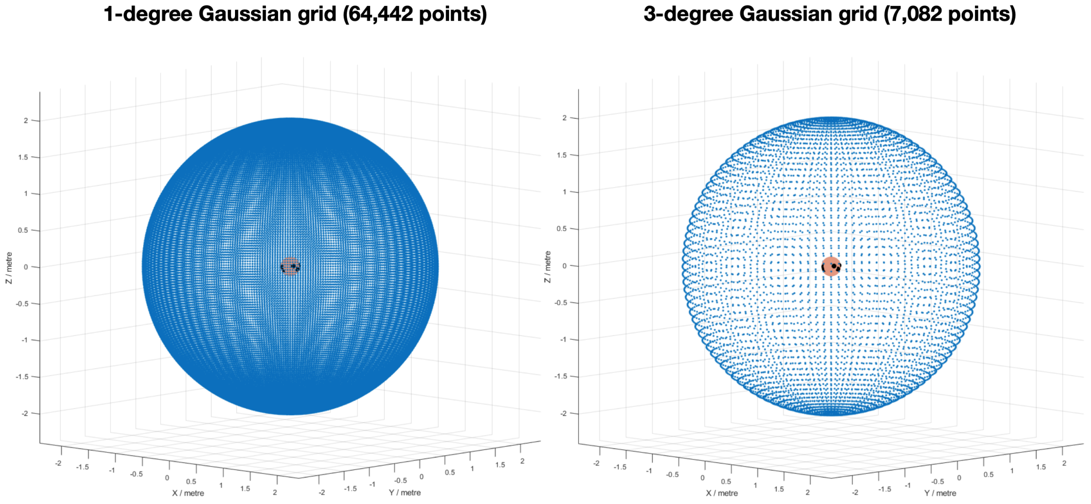


Figure 5.3: 3DMM generated head mesh example 2

and every head model with Openflipper. By using the reference head model and the pair of remeshed head models, a MATLAB script remaps any head meshes created by the 3DMM to a remeshed version at a much faster speed.

All meshes will be first generated with the 3DMM, then passed through the remapping MATLAB script to apply the mapping for an evenly distributed mesh. After remeshing, the final mesh has 165,406 vertices and 55,138 faces. All triangles in the mesh have an edge length no longer than 2.8mm, as the HRTFs generated in this work aim to reach 20kHz (according to Equation 2.2), and the angles are larger than 15 degrees to avoid



1-degree Gaussian grid (64,442 points)

3-degree Gaussian grid (7,082 points)

Figure 5.4: 1-degree Gaussian grid (64,442 points) and 3-degree Gaussian grid (7,082 points) with a radius of 2 meters

collapse in simulation [90]. Reciprocity simulation is used, where instead of mimicking the most common physical measurement method that has microphones placed in the ear canal of a subject and a set of loudspeakers positioned at different angles relative to the head, speakers are placed in the left and right ear canal, and a set of microphones positioned at different angles relative to the head. This method is rarely used in physical measurement due to the consideration of the location of the speakers being close to the eardrums, so the loudness of the measurement signal is often limited and results in a poor signal-to-noise ratio. However, this method is more commonly used in BEM simulation as it can obtain HRTFs from all angles simultaneously. Mesh2HRTF allows users to select two triangular faces at the left and right ear as speaker locations. Due to the characteristic of the head models generated with 3DMM, different elements of a head share the same vertices across all head models. Thus, by locating the index of the vertices of the triangular face that is at the opening of the ear canal, those indexes can be applied to all the head models to determine the speaker locations. Figure 5.5 and 5.6 show the triangular faces that were selected as the speaker locations. To have a dense angle of HRTFs measurements, the output angles of the BEM simulation are in a 1-degree Gaussian grid (64,442 angles) with a radius of 2 meters. 3-degree Gaussian grid (7,082 angles) HRTFs will be extracted from the 1-degree Gaussian grid output after post-processing. The distributions are shown in Figure 5.4.

The Mesh2HRTF software provides a Blender plugin called Mesh2Input to pre-process the head mesh for BEM simulation with a given configuration. It allows the user to set different parameters for different simulations, including the speed of sound, the desired grid of output angles, output frequency step, maximum output frequency, and CPU configuration. This work used a slightly modified version of the original plug-in that includes the minimum output frequency.

Considering most binaural applications are used indoors, the HUMMNGBIRD database aims to simulate HRTFs that are around a common room temperature of

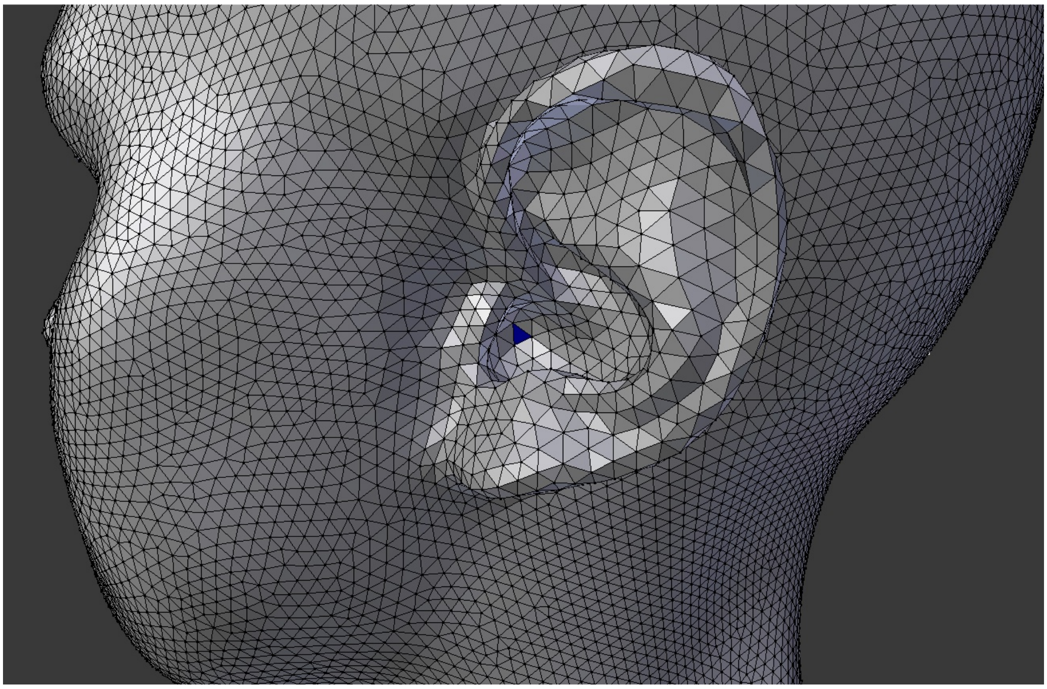


Figure 5.5: The triangular face that was selected as the speaker location at the opening of the left ear canal (in blue)

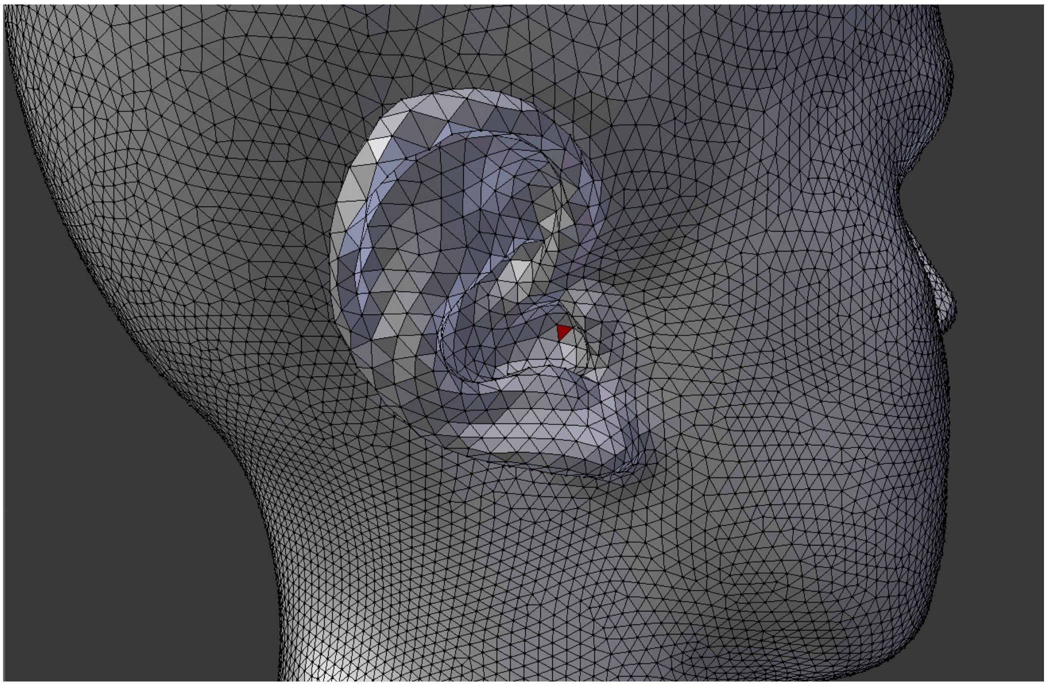


Figure 5.6: The triangular face that was selected as the speaker location at the opening of the right ear canal (in red)

Python Variables	Value
frequencyStepSize	100
maxFrequency	20000
minFrequency	100
cpuFirst	1
cpuLast	4
numCoresPerCPU	2
pictures	False
ear	‘Both ears’
evaluationGird1	a 1-degree Gaussian grid created for the HUMMNGBIRD database
method	‘4’
reciprocity	True
sourceXPosition	‘0’
sourceYPosition	‘0’
sourceZPosition	‘0’
speedOfSound	‘343’
densityOfMedium	‘1.21’
unit	‘m’
frequencyDependency	False
nearFieldCalculation	False

Table 5.1: Configurations of the Python Variables for the Mesh2Input Blender plug-in

$20°C$ in dry air, where the speed of sound is $343ms^{-1}$, and the density of medium is $1.21kgm^{-3}$. The output frequency is set from 100Hz to 20kHz with a 100Hz step. This is a balance between computational time and frequency resolution. Table 5.1 lists the configuration setup in the Mesh2Input Blender plug-in. The CPU configuration is discussed in Section 5.2.3.

The entire process, from generating head models to the Mesh2Input pre-processing, is automated by a bash script. 1000 head meshes are created and then pre-processed each time. The process takes about 32 to 36 hours with a laptop equipped with an Intel Xeon E3-1505M v6 CPU, 32GB DDR4 2400MHz ECC SoDIMM RAM, NVIDIA Quadro P4000 8GB, 1TB SSD PCIe TLC OPAL2 SSD and output to a RAID 500 HD 7200RPM hard disk.

5.2.3 BEM Simulation with Mesh2HRTF

The Mesh2HRTF software allows the simulation to run parallel on different CPU clusters. For a single simulation, the more CPUs are used, the shorter the simulation takes. However, for batch simulation, the fewer CPUs used for each batch simulation, the more efficient each CPU can be. This means that despite taking more time for each simulation, the total time will be shorter when multiple simulations are run in parallel. For example, when running 12 simulations on a 24 CPU core computer setting, a 2-core setup will take less time than a 6-core setup to finish all 12 simulations: a 3-degree Gaussian grid simulation (7320 points) takes 112028 seconds (about 31.12 hours) with 2 cores and 4014 seconds (about 11.25 hours) with 6 cores. In that case, 12 simulations can be done in about 31.12 hours with 2 cores but require 33.75 hours with six cores.

Viking is a computer cluster located at the University of York that aims to provide high-performance computing for various projects. It contains 42TB of memory and 7024 CPU cores with an estimated performance of 435.2 TFLOPS. Considering the fair share of this resource, this work utilises the Viking computer cluster for the BEM simulation only. All the pre-processing and post-processing are done on a local PC. Due to the constraints of the computer cluster setup, the simulations for this work use an 8-core setup, which takes 36 to 44 hours for each 1-degree Gaussian grid HRTF set. 50 simulations can run simultaneously, resulting in approximately two weeks to finish 500 simulations.

5.2.4 Post-processing

The Mesh2HRTF software provides a MATLAB script named Output2HRTF to post-process the BEM simulation raw output data into HRTFs. The default Output2HRTF setting for the post-processing has a 20dB roll-off between every 10^n to 10^{n+1} Hz and the magnitude data starts from 100Hz to 20kHz in 100Hz steps. This is because the Mesh2HRTF 0.4.0, does not provide a “reference” option in the Blender plug-in to generate Measurement Equalised HRTFs despite the Output2HRTF MATLAB script providing the feature [1]. This option was later added in version 1.0. Measurement Equalised HRTFs equalise the measured or simulated HRIRs with respect to the free field sound pressure at the centre of the head with the head absent, given by

$$H(\theta, \phi, f) = \frac{P(\theta, \phi, f)}{P_0(f)} \tag{5.1}$$

where H is the Measurement Equalised HRTFs, P represents the measured HRTFs, P_0 represents the free field sound pressure in the frequency domain at the centre of the head with the head absent. θ and ϕ are the azimuth and elevation angle, respectively, and f is the complex value of the sound pressure in the frequency domain.

To calculate P_0 and equalise the simulated HRTFs, some modifications need to be made manually in the Output2HRTF MATLAB script to enable the feature without the Blender plugin. The magnitude spectra plots of the result are shown in Figure 5.7.

After running the Output2HRTF script, since the BEM simulation could not simulate anything at 0Hz, there is a cut-off at 100 Hz, as shown in Figure 5.7. To solve that, the magnitude data from 100Hz is duplicated to 0Hz (Figure 5.8).

In order to make the simulated HRIRs compatible with typical headphone response design target curves, the HUMMNGBIRD database provides a diffuse-field compensated version of HRIRs [266, 267]. These HRTFs have been equalised with respect to the non-directional magnitude response component calculated for each of the HRTF sets. The compensated HRTFs are calculated using Equation 5.2, where H_{dfe} is the Diffuse Field Equalised HRTF set, H represents the Measurement Equalised HRTFs, M represents the total number of HRTFs in each set, θ and ϕ are the azimuth and elevation angles, f is frequency bin and sa_i is a solid angle corresponding to each simulated HRTF direction expressed in steradians.

$$H_{dfe}(\theta, \phi, f) = \frac{H(\theta, \phi, f)}{\sqrt{\frac{1}{4\pi}\sum_{i=1}^{M} |H(\theta_i, \phi_i, f)|^2 sa_i}} \tag{5.2}$$

The DFE filter is calculated based on the average of both left and right ear signals, as shown in Figure 5.9.

5.2.5 Output Files

The database offers different HRTFs and HRIRs in the SOFA file format, as shown in Table 5.2. The two HRTFs_simulated are the sofa files that are straight from the Output2HRTF script without any extra post-processing. HRTF_raw is the sofa file with the cut-off at 100Hz removed. HRIR_raw is the equivalent sofa file of the HRTF_raw file in both 1-degree and 3-degree Gaussian grids. HRIR_dfe is the diffuse field equalised version of the HRIR_raw file in both 1-degree and 3-degree Gaussian grids. The 3D_Model is the 3D Head mesh in Blender format. The latent_var .mat file is the MATLAB file that stores the equivalent latent PC variables to generate the head mesh with the 3DMM model.

5.3 Evaluation

Whilst Young et al. validated the BEM simulation process of Mesh2HRTF, a question remains as to whether the HRTFs simulated with the head models from 3DMM are similar to the HRTFs from actual human beings. To evaluate this objectively, Principal Component Analysis (PCA) is used to examine the result numerically. This idea comes from Middlebrooks et al [138]., Mokhtari et al. [139] and Kistler et al. [140], who have conducted studies on PCA of HRTFs. One common use of PCA is to calculate the PCs, which are the eigenvectors that characterise how the data are distributed. Inspired by their research, by performing PCA on each database individually and comparing the PCs between them, the similarity of characteristics of the HRTFs shared across different datasets can be observed. This means that if the HUMMNGBIRD HRTF datasets have PCs similar to other HRTF datasets, it is likely that the generated HRTFs are on par with the HRTFs from other HRTF databases. This could indicate that the HRTFs simulated with the head models from 3DMM are similar to the HRTFs from actual human beings. To evaluate newly generated HRTFs, PCA was performed, and the PCs were compared with a variety of HRTF databases: SADIE II [23], CIPIC [24], IRCAM LISTEN [25], CHASAR [109], SYMARE acoustic measurement [90], SYMARE BEM simulated [90] and RIEC [21]. These databases were chosen for specific reasons: The SADIE II, CIPIC, and LISTEN databases are well-established and commonly used in HRTF research, which is a good baseline. As the 3DMM that this work used to generate head models includes children, the CHASAR database is selected as it provides the HRTF measurements of children of the age of 3-10 years. SYMARE acoustic measurement and the SYMARE BEM simulated databases were used to compare the difference between HRTF measurements and BEM simulation, as they have HRTFs from the measurements and BEM simulation from the same human subjects. The version of SYMARE BEM simulated HRTFs used in this study are without torsos and

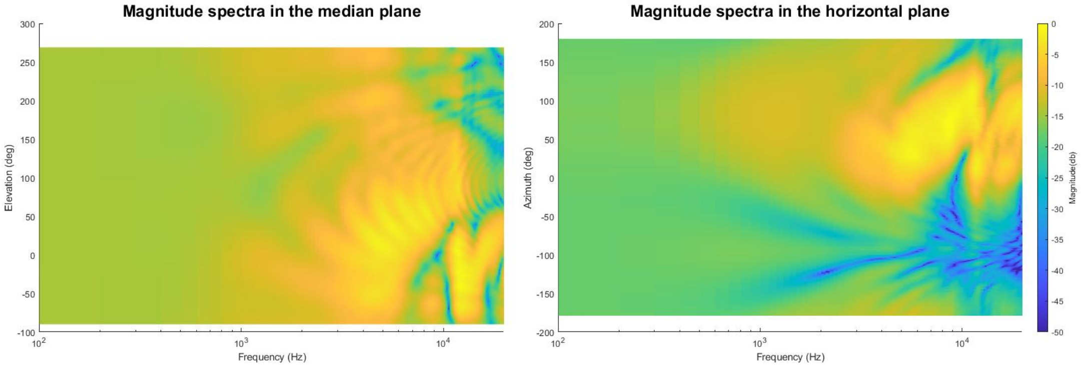


Figure 5.7: Magnitude spectra plots of an HRTF set from Output2HRTF. It shows the magnitude spectra in both the median and horizontal planes. The cut-off at 100Hz removed has not been removed, thus the magnitude data start from 100Hz to 20kHz in 100Hz steps

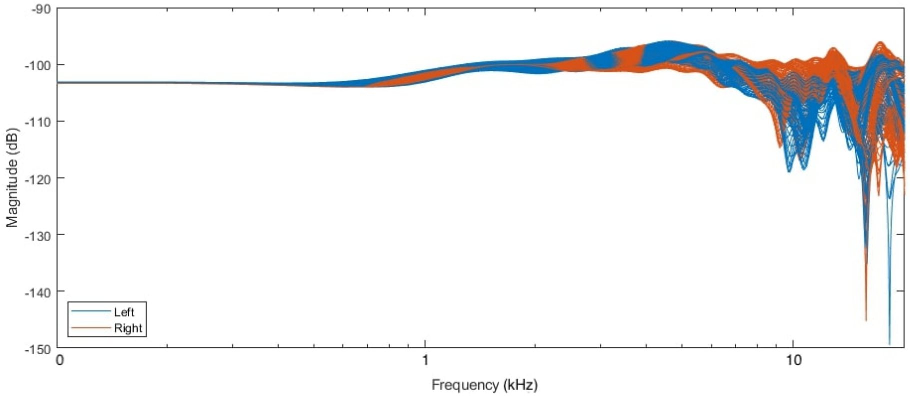


Figure 5.8: A HRTF plot of an HRTF set from Output2HRTF with the magnitude data from 100Hz is duplicated to 0Hz

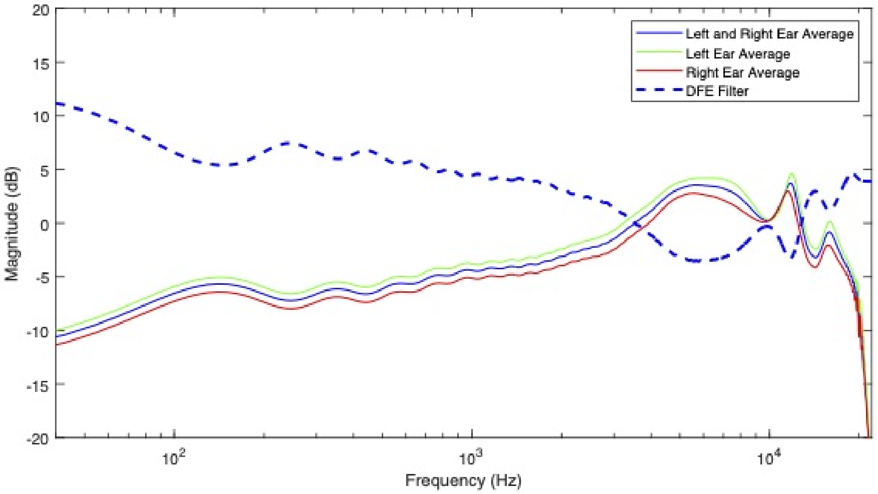


Figure 5.9: Left, right, both ear average responses and the Diffuse Field Equalisation (DFE) filter response.

File name	Data	Grid	Points	Size	Length
HRTF_simulated _1deg_H1.sofa	Original HRTF output from Output2HRTF	1-degree Gaussian	64,442	340 MB	200 Frequency steps (100Hz to 20kHz)
HRTF_simulated _GeneralTF _1deg_H1.sofa	Original GeneralTF HRTF version output from Output2HRTF	1-degree Gaussian	64,442	340 MB	200 Frequency steps (100Hz to 20kHz)
HRTF_raw_1deg_H1.sofa	HRTF with cut-off at 100Hz removed	1-degree Gaussian	64,442	340 MB	441 taps
HRIR_raw_1deg_H1.sofa	HRIR with cut-off at 100Hz removed	1-degree Gaussian	64,442	370 MB	441 taps
HRIR_raw_3deg_H1.sofa	HRIR with cut-off at 100Hz removed	3-degree Gaussian	7,082	40 MB	441 taps
HRIR_dfe_1deg_H1.sofa	HRIR with cut-off at 100Hz removed and diffuse field equalised	1-degree Gaussian	64,442	370 MB	441 taps
HRIR_dfe_3deg_H1.sofa	HRIR with cut-off at 100Hz removed and diffuse field equalised	3-degree Gaussian	7,082	43 MB	441 taps
3d_Model_H1.blend	3D Head mesh in blender format	n/a	n/a	7 MB	n/a
latent_var_H1.mat	Latent variables for generating the head mesh with 3DMM	n/a	n/a	4 KB	n/a

Table 5.2: List of the output files

only go up to 16kHz, as the SYMARE BEM simulated HRTFs with torso only go up to 5kHz, which makes it hard to compare the difference in high-frequency data. This could also be used as a benchmark for the BEM-simulated HRTFs without torsos. RIEC is an HRTF database from Japan, suggesting there may be more HRTF measurements from Asian people. Since one of the goals of this work is to provide an HRTF database that could cover different races, the RIEC database is selected to find out if it yields different results compared to other databases.

5.3.1 PCA on HRTF Datasets

Most HRTF databases present HRTFs in the time domain as HRIRs due to ease of use in audio rendering. In this thesis, as the intention is to perform PCA in different frequencies to understand the characteristics of HRTFs, the Fast Fourier Transform (FFT) was performed to convert the minimum phase HRIRs to HRTFs. To best replicate the PCA method from Middlebrooks et al. [138], Mokhtari et al. [139], and Kistler et al. [140], the HRTF sets are rearranged into a 2D matrix where the row number is equal to: subjects × positions × channel, and the column number equal to the signal length, which is the frequency bins of the HRTFs. In this study, the frequency bins range from 0Hz to 20kHz in 100Hz steps. HRTFs are then converted to decibels, and the average magnitudes of the HRTF sets are subtracted. The reason for subtracting the average magnitudes is twofold: to eliminate the artefacts induced by the measurement or simulation setup across the HRTFs, and to perform mean normalisation. Mean normalisation is a crucial step before using the data with PCA, as it can make sure the mean of the processed data will be close to zero. This ensures that the first principal component describes the direction of maximum variance.

Figure 5.10 shows the results of the first five PCs across frequency. To better understand the HUMMNGBIRD datasets, three types of HUMMNGBIRD HRTFs are used in the PCA. HUMMNGBIRD DFE HRIR is the post-processed DFE HRIR, the HUMMNGBIRD RAW HRIR is the HRIR without DFE, and the HUMMNGBIRD RAW HRTF is the raw HRTF from BEM simulation with the cut-off at 100Hz removed. All three are basically identical, except for the slight difference in PC5 between 1kHz and 7kHz. All the HUMMNGBIRD HRTFs are in a 1-degree Gaussian grid. Only 100 sets of HRTFs from the HUMMNGBIRD database are used to match the approximate number of HRTF sets from the other databases. The plots show two red vertical lines at 2kHz and 16kHz, which are for comparing the results from the Mokhtari et al. paper [139].

As shown in Figure 5.10, all eigenvectors of the PCs from different databases have a similar shape. The exceptions are in PC2, where both BEM-simulated HRTFs have a wider and lower peak, and the CHASAR has a more graduate slope in the mid-frequency. However, the HUMMNGBIRD results in PC2 to PC5 unanimously shifting to higher frequencies, with a difference of around 2kHz. The BEM-simulated HRTF from SYMARE has a similar issue in PC2 and PC3, even when compared with the acoustic-measured HRTFs. This indicates this difference may be caused by the BEM simulation. In the SYMARE paper, Jin et al. [90] suspect that the most likely explanation is that the value used for the speed of sound in the simulations ($343ms^{-1}$) was larger than the actual speed of sound for the measurements. Kreuzer et al. [268]

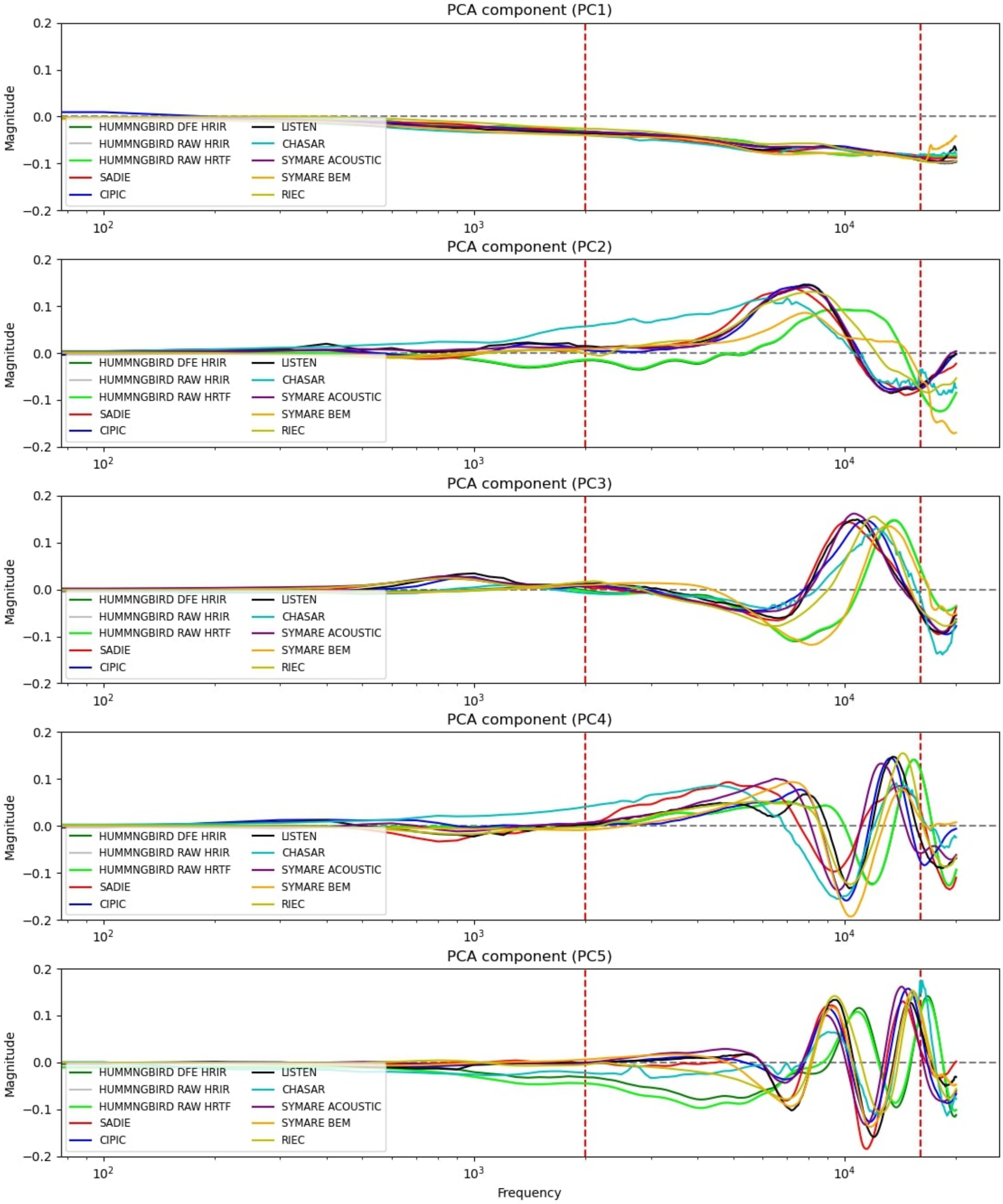


Figure 5.10: Comparing the first five PCs of different HRTF databases. The name of the database that each colour presented is in the bottom left legend. Three types of HUMMNGBIRD HRTFs are used in the plots. HUMMNGBIRD DFE HRIR is the post-processed DFE HRIR, the HUMMNGBIRD RAW HRIR is the HRIR without DFE, and the HUMMNGBIRD RAW HRTF is the raw HRTF from BEM simulation with a cut-off at 100Hz removed. The two red vertical lines at 2kHz and 16kHz are for comparing the results from the Mokhtari et al. [139] paper. Note that the SYMARE BEM simulated HRTFs only go up to 16kHz.

support their hypothesis and show that temperature has an effect on the simulation result. With higher temperatures, where the speed of sound is faster, the peak and notches will shift towards higher frequencies. In the paper by Young et al. [30], to validate the BEM simulation of KEMAR with physical acoustic measurement, the speed of sound and air density for the BEM simulation was set to $338ms^{-1}$ and $1.24kgm^{-3}$ to match the ambient air temperature of $11°C$ in an anechoic chamber. This implies that HRTF measurements from the other databases may be measured in an environment with a different speed of sound and air density, likely lower than $20°C$, on which the HUMMNGBIRD simulation is based. Another observation is that in PC2 to PC4, the results from RIEC are also slightly shifted to higher frequencies. It is unclear if the different races would have any specific effect on the HRTFs generally. Further research could be conducted in the future, but this is beyond the scope of this thesis. As for the effect on the torso, by comparing the SYMARE and HUMMNGBIRD BEM simulated results among other HRTFs in the low to mid frequencies, it appears that the HUMMNGBIRD HRTFs are flatter from 500Hz to 1150Hz in PC3. Considering the range and location, the difference may be caused by the missing torso. Finding a novel method to add back the influence of the torso would be a future advancement.

	SADIE	CIPIC	LISTEN	CHASAR	SYMARE ACOUSTIC	SYMARE BEM	RIEC	Mean	Max
PC1	0.994	0.991	0.991	0.991	0.993	0.985	0.989	0.991	0.994
PC2	0.909	0.894	0.898	0.804	0.903	0.856	0.880	0.878	0.909
PC3	0.805	0.805	0.799	0.741	0.823	0.979	0.915	0.838	0.979
PC4	0.684	0.921	0.905	0.716	0.843	0.728	0.924	0.817	0.924
PC5	0.752	0.779	0.744	0.800	0.732	0.739	0.858	0.772	0.858
Mean of all PCs	0.829	0.878	0.867	0.810	0.859	0.857	0.913	0.859	0.933

Table 5.3: The maximum value of the normalised cross-correlation between the HUMMNGBIRD DFE HRIRs and the other databases. (Higher the better, with 1 representing a completely correlated relationship, and 0 indicating no correlation)

To further understand if the plots are in a similar shape other than the frequency shift, Table 5.3 shows the maximum value of the normalised cross-correlation between the HUMMNGBIRD DFE HRIRs and the other databases with 1 representing a completely correlated relationship, and 0 indicating no correlation. The mean across all databases shows that PC1 to PC4 are over 80% correlated. The mean of PC1 to PC5 shows that the HUMMNGBIRD database is above 80% similar to all databases. The maximum correlation across all databases is above 90% for PC1 to PC4 and about 85% for PC5. This shows that the character of the HUMMNGBIRD HRTFs is similar to at least one database.

5.4 Conclusion

This Chapter has demonstrated a method for generating a significant amount of clean and uniform HRTF data from head models generated by 3DMM. More importantly, the datasets provide, by far, the most fairly distributed HRTFs in gender, age, and race. Although physical HRTF measurements are still preferred in many spatial audio applications, the HUMMNGBIRD database provides an alternative for those who could benefit from a more diverse HRTF database. In addition to this, the method shown in this paper has the potential to generate other kinds of data for specific HRTF research, for example, near-field HRTFs, HRTFs with different head and torso orientations, HRTFs with different speed of sound or density of air, different measurement grids, and HRTFs of specific head models.

At the time of writing, the HUMMNGBIRD HRTF database is still in the process of refinement and finding the optimal way to distribute such a huge amount of data. If there are any inquiries regards to the HUMMNGBIRD HRTFs, please feel free to contact via elec-hummngbird@york.ac.uk .

Chapter 6

Preliminary Investigation into the Potential of Using HUMMNGBIRD HRTF Datasets in Machine Learning

6.1 Motivation

To understand the power of the significant number of HRTF sets generated for the HUMMNGBIRD database, a few PCA models and VAEs have been implemented. These models are often used for dimensional reduction, where the input data are projected into a latent space that is smaller than the input size. PCA is a more traditional statistical technique, and VAE is a neural network. Both of these models are outlined in Chapter 2 section 2.2.3 and 2.2.4. The purpose of implementing dimensional reduction is that since the input and target output are the same, it is a fair way to evaluate the performance of the model. At the same time, by testing different configurations, it can show how small the latent space could be whilst being able to reconstruct a set of HRTFs. It can also find out if the quality and the size of the HUMMNGBIRD database are good enough to generate synthetic HRTF sets with the latent space.

To reduce the size of the HRTF data so the VAE models can be trained with a single GPU, the input and the target output of the VAE are 5810 HRTFs extracted from the 1-degree HUMMNGBIRD HRTF datasets. The 5810 HRTF datasets are in a Lebedev quadrature configuration. 5810 is the 131st Levedev order which is equivalent to a 32nd-order SH configuration. Based on Equation 2.7, the interpolated HRTFs are approximately accurate up to the spatial aliasing frequency 20,562 Hz when $c = 343$ and $r = 0.85$, which is above the maximum human hearing frequency.

This Chapter is organised as follows: Section 6.2 will cover relevant setup detail that is used in the experiments. It will also discuss some of the choices and potential changes that can be made for different applications. Section 6.3 will discuss the details and results of the performed experiments. Section 6.4 discusses the results across all the experiments. Section 6.5 concludes the chapter.

6.2 Setup

All the models are trained with the DFE HRTFs. To have more uniform data for comparison, all the HRIRs from all databases were upsampled to 48 kHz if necessary after the FFT was performed to convert the HRIR to HRTF with a 256 length from 0 to 24 kHz.

With a big database, there must also come great computational power. To get good results from the modern ML model, the number of training data, computational power and algorithm need to come hand in hand. When training the SH interpolation models in Chapter 4.2, an NVIDIA RTX2080 Ti was used, which only has 11 GB VRAM. The 1-degree SOFA file from HUMMNGBIRD costs around 374 MB, and the extracted HRTF dataset with 5810 HRTFs in Numpy (.npy) format costs 23 MB. The ML model uses 5810 positions extracted from each HRTF set. Thus, the data size is 5810 positions × 2 channels (left and right) × HRTF length. The HRTF length used in the model depends on the setting, in most cases, the input data only goes up to 20 kHz, and the data point representing the lowest frequency was removed since it represents 0 Hz which is not much use for the ML model. Thus, the HRTF length in most cases is 213. This means the input size is 2,206,680 in total (5810 × 2 × 213). Such large input data means the ML model could be deeper and wider. Despite the CNN architecture being fairly efficient, the basic CNN model that was trained in this Chapter requires 11 GB of memory with a batch size of 8 and kernel size of 3. Increasing the batch size and kernel size will cost more computational power. A VAE, on the other hand, can be double the size of the CNN, and it involves some linear layer which demands more memory than a convolutional layer, a conservative model takes about 21 GB of memory. Thus an NVIDIA RTX 3090, which provides 24 GB VRAM enabled the option to use a straightforward approach with a large amount of data. This is also the reason that the previous SH restoration model is no longer needed. There is no need to perform SH restoration as part of the pre-processing and restore each HRTF individually when it is possible to process the entire HRTF set at once. It is also believed that the HRTFs from different angles could contribute to a better interpolation result.

6.2.1 PCA

Due to the size of the data, a typical PCA requires a computer with TB of memory to process all the HRTF datasets. Incremental Principal Components Analysis proposed by Ross et al. [269] allows PCA to process the HRTF sets in batches. This work used the IPCA implementation from the sklearn Python library due to its robustness. Three IPCA models were created with 3, 30, and 100 PCs accordingly. Comparing the results with different numbers of PCs can hopefully show how they affect the reconstruction and generative result. These numbers are selected for the following reasons: The 100 PCs are set because the IPCA process was run in a PC with 32GB of RAM, as the batch size of the IPCA can not be smaller than the number of PC, 100 PCs is the setup that can be reliably run on that PC. Using a 3 PCs setting is to test the performance of an extreme condition. The distribution of the 3PCs can be plotted in a 3D plot in debugging.

Figure 6.1 shows that 100 PCs can explain about 80% of the variance. The first two PCs capture significantly more variance compared to the other PCs. The knee point is

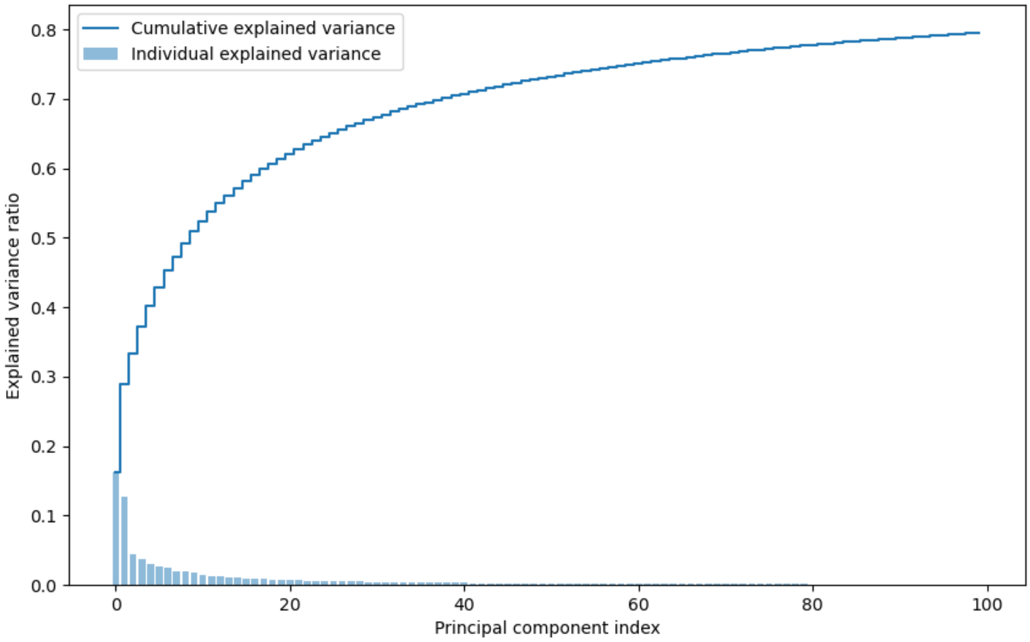


Figure 6.1: Explained variance of 100 PC

at around 21 PC, which has a good trade-off between the number of latent variables and performance. The explained variance with 21 PC is 62.1%. As it is common practice to use a number that is a square of 2 as the latent variable, 32 is chosen as the middle ground between 3 and 100 PC, where the explained variance is 67.7%. The idea is to use this to show the transition between small and large numbers of PCs. To investigate if the number of data affects a PCA model, a separate 32 PCs model was created with only 500 HUMMNGBIRD training data.

6.2.2 VAE generative model

The input and output of the VAE are both the 5810 positions extracted from the 1-degree HUMMNGBIRD HRTF set. The encoder and decoder layers of the VAE are mostly constructed with a convolutional layer. The convolutional layer changes the dimension that represents the HRTFs position. Thus, the encoder reduces the channel size from 5810 until it reaches 1. Then a few linear layers are used to reduce the data further into the desired latent space size. After that, the MSE loss and KL loss are calculated. With the reparameterisation trick, the final latent variables are developed. The latent variables then feed into the decoder. The decoder used in this work is a mirror of the encoder. Changing the size of the latent space changes the bottleneck of the VAE, which forces the model to find a more efficient way to represent the input data in a small latent space. However, if the latent space is too small, the model may throw away too much data and compromise the performance significantly. For preliminary investigation purposes, three VAEs were trained, one with 3 latent variables, one with 32 latent variables and one with 32 latent variables that were trained with 500 HRTFs only.

Parameters	Value
Epochs	500
Batch Size	8
Convolution Layer Kernal	9
Convolution Layer Padding	4
Pooling Layer Kernal	11
Pooling Layer Padding	5
KL Beta	0.1
Learning Rate	0.0001 (0.001 for the one trained with 500 HRTFs)

Table 6.1: Hyperparameters for the VAE models

The 32 latent variables one that trained with 500 HRTFs is to show the significance of the number of data. It will compare with the 32 PCs PCA model to see how different models behave differently with a limited amount of data. All the models are optimised with slightly different hyperparameters that are listed in table 6.1 All the models are trained with 500 epochs with a batch size of 8. All the models trained with 5000 HRTF sets have a learning rate of 0.0001. The one trained with 500 HRTF sets uses a learning rate of 0.001 as it can not converge after 500 epochs with a learning rate of 0.0001. For the models that were trained with 5000 HRTF sets, each model took about 160 hrs to train. For the 32 latent variables, VAE that trained with 500 HRTFs took about 24 hours to train. Each model is slightly optimised to make sure the performance can reflect the potential of the model reasonably. Due to the time constraint and lack of specific narrative objectives for the models, the models by no means represent their best performance.

6.2.3 Evaluation Protocol

To evaluate the effectiveness of the model, the mean squared error (MSE) between the input and output data is first calculated. However, MSE alone may not be sufficient for accurately assessing the model's performance, as it can be deceptive in certain situations. For example, when the number of latent variables used in the model is not sufficient to capture the variance in the data, the model may produce very similar outputs even though the latent variables are different. This may result in a low MSE even though the model is not useful. To better evaluate the variance of the model's output, the standard deviation of 100 validation data outputs was calculated. To compare the standard deviation of the input and output data, the standard deviation of HRTFs was plotted in Figure 6.3. Despite the darker means of a higher standard deviation, the more grey area indicates a better variance across the entire data. To visualize the distribution of the standard deviation, the histogram of the standard deviation for the input and model output is shown in Figure 6.4. By comparing the histograms, we can see how well the model captures the variance of the input data. Note that all of the HRTF plots shown in this analysis are for the left channel of the horizontal plane HRTFs. The left and right channels of the horizontal, median, and frontal planes are included in Appendix A.

Compare the reconstruction MSE between PCA and VAE (in dB)		
Model	PCA	VAE
100 PC / latent variables	3.7575	N.A.
32 PC / latent variables	5.3541	8.3956
3 PC / latent variables	9.2225	9.5363
32 PC / latent variables (trained with 500 HRTF sets only)	5.7453	9.7339

Table 6.2: Use Mean Squared Error (MSE) to compare the reconstruction of the validation HRTF sets with PCA and VAE models

6.3 Experiment results

6.3.1 PCA Training and Reconstruction

According to Table 6.2, the number of PCAs affects the MSE significantly. However, the number of training data does not seem to affect the 32P PCA model much. To exam the reconstructed output further, figure 6.2 shows the reconstructed output of one of the validation data from the PCA models on the right column. According to the plots, although there are minor differences between the results, all of them look reasonable compared with the input on the top left. Figure 6.3 shows the standard deviation of 100 validation HRTF sets from the PCA models, where the performance difference of each PCA model is more noticeable. The bigger the grey area in the plot, means the model is capable of adapting to different inputs. The performance of the PCA model decreased when the PCs got lower. However, the number of training data doesn't seem to have a huge effect on the result with 32 PCs. Figure 6.4 shows the histogram of the standard deviation. It shows that the 100 PCs and 32 PCs model has a similar trend with the input data. However, the 3 PCs model seems to perform differently. It is unclear whether this is important, especially when considering the result in figure 6.2 it seems like all the models are capable of producing decent results.

6.3.2 VAE Training and Reconstruction

Based on the information provided, it seems that the results from Table 6.2 indicate that the MSE between different VAE models is relatively similar. However, the VAE model with 32 latent variables that were trained with 500 HRTF sets performed worse than the VAE model with 3 latent variables. This suggests that the performance of the VAE model, a deep neural network model, is heavily influenced by the size of the training data. In addition, Figure 6.2 shows that the VAE model with 32 latent variables produces sharper results compared to the 32 latent variables model trained with 500 HRTF sets and the one with 3 latent variables. The results in Figure 6.4 and Figure 6.3also confirm that the VAE model with 32 latent variables trained with 500 HRTF sets performs the worst. These findings suggest that the size of the training data and the number of latent variables in the VAE model can significantly impact its performance.

6.3.3 Comparing Results Between PCA and VAE

By comparing the PCA and VAE results in Table 6.2 and Figure 6.2, it is clear that PCA outperforms VAE. This is expected, as oftentimes, traditional statistical models outperform neural network model that is not optimised. The biggest challenge of using a neural network model is that there are a lot of hyper-parameters that can be tuned and different architectures that can be experimented with. On the other hand, neural network models often perform better in challenging tasks. As an example, by comparing the performance between a PCA model with 3 PCs and a VAE with 3 latent variables in Table 6.2, the difference in MSE is slight. The histogram of the standard deviation from Figure 6.4 shows that the VAE with 3 latent variables looks more similar to the input. This indicates that with more hyper-parameter tuning, a VAE with 3 latent variables might outperform the PCA result with 3 PCs. Figure 6.3 shows the stand deviation of validation results. Compared with the PCA model with 3 PCs, the VAE with 3 latent variables seems to have more grey area, which indicates the VAE model is more capable than the PCA model. However, VAE is more sensitive to the size of the training data. The 32 PCs PCA models that were trained with 5000 HRTF sets and 500 HRTF sets look very similar. However, the 32 latent variables VAE that were trained with 5000 HRTF sets and 500 HRTF sets look very different. The one trained with 500 HRTF sets is arguably worse than the 3 latent variables VAE result. Similar results are shown in Figure 6.3.

The VAE with 3 latent variables has great potential to outperform the PCA model, considering how close the MSE and the histogram from Figure 6.4 are. More objective measurements and data visualisation methods are needed to further evaluate and compare the performance of these models.

6.3.4 PCA and VAE for synthetic HRTF sets

Figure 6.5 shows the left channel of randomly generated HRTFs on a horizontal plane from the PCA and VAE models. There are three randomly generated HRTF sets displayed for each model in this figure, and a total of nine HRTF sets for each model can be found in Appendix A. All of the HRTFs appear to be realistic. This prompts the question of whether the distribution shown in Figure 6.4 holds any significance.

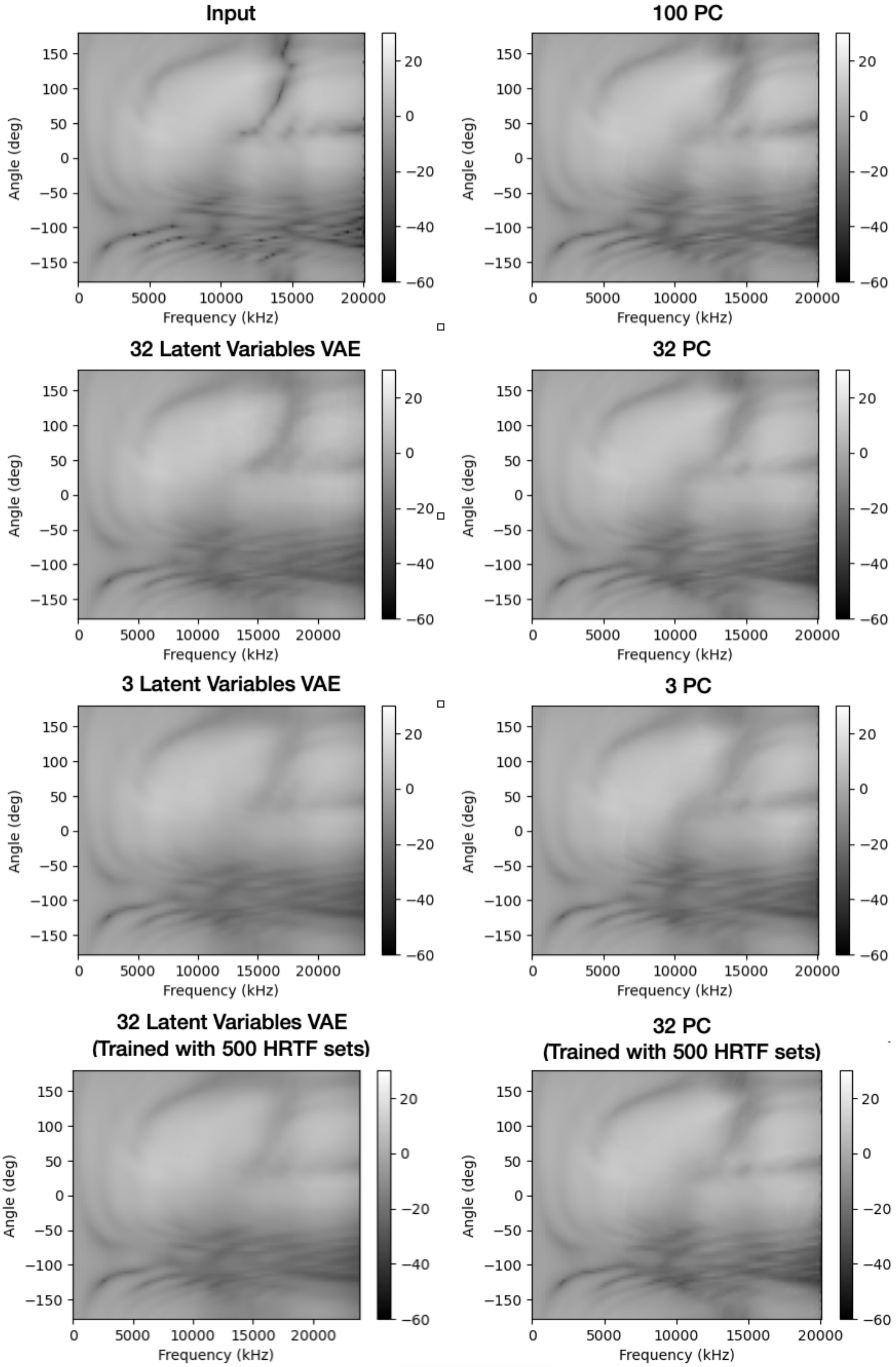


Figure 6.2: Input and reconstructed output of 100 validation HRTF sets from different models

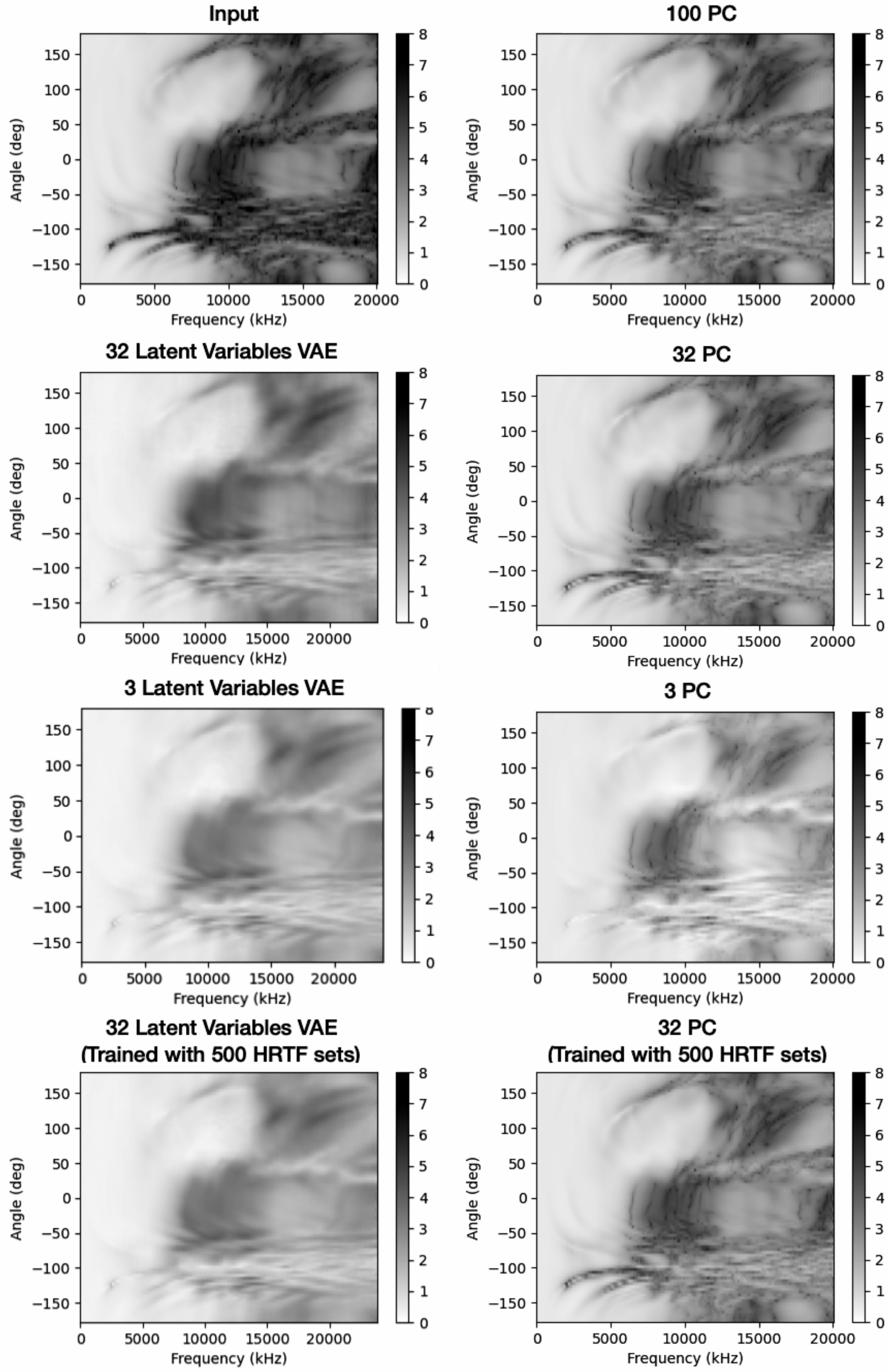


Figure 6.3: Standard deviation of the input and reconstructed output of 100 validation HRTF sets from different models

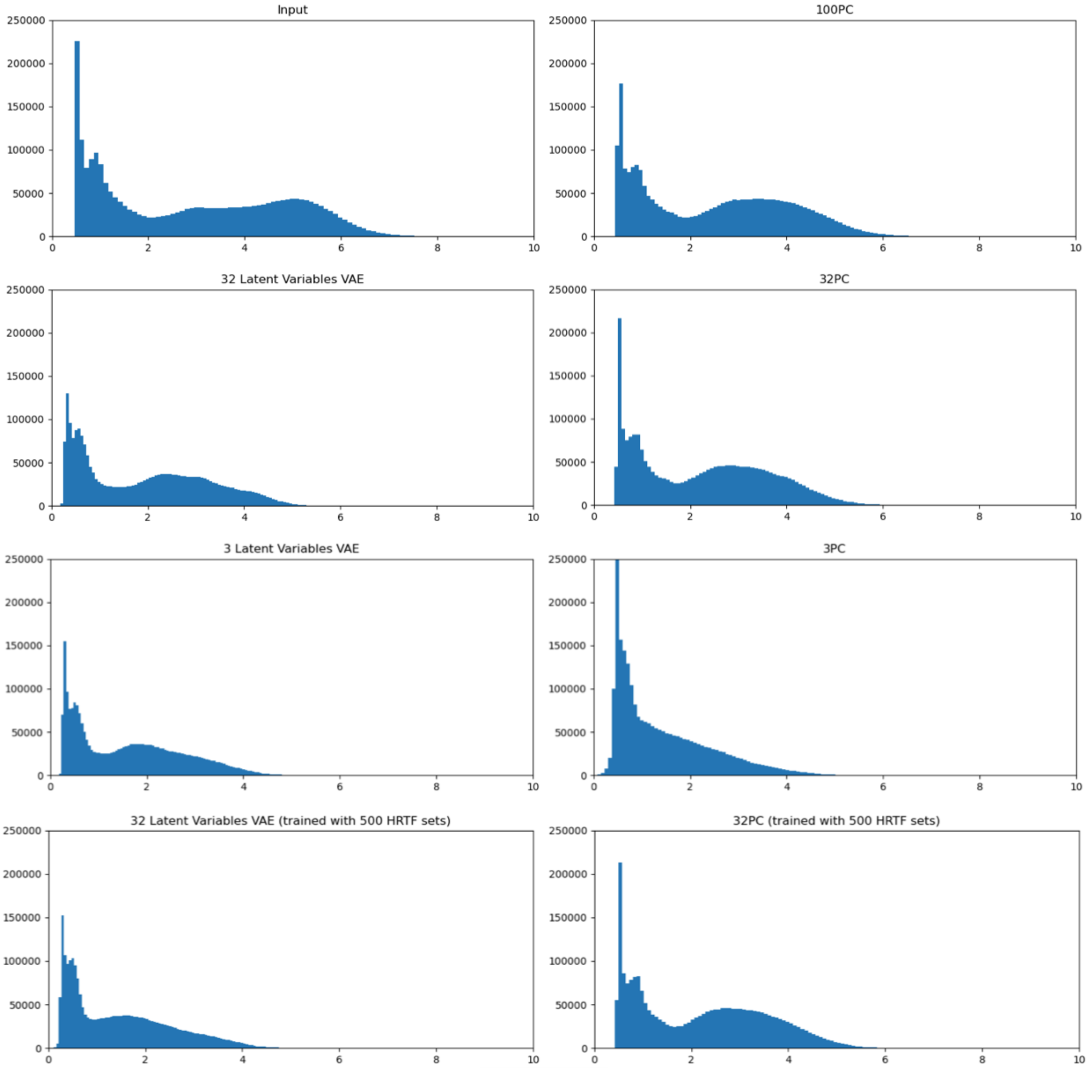


Figure 6.4: Histogram of the standard deviation of the input and reconstructed output of 100 validation HRTF sets from different models

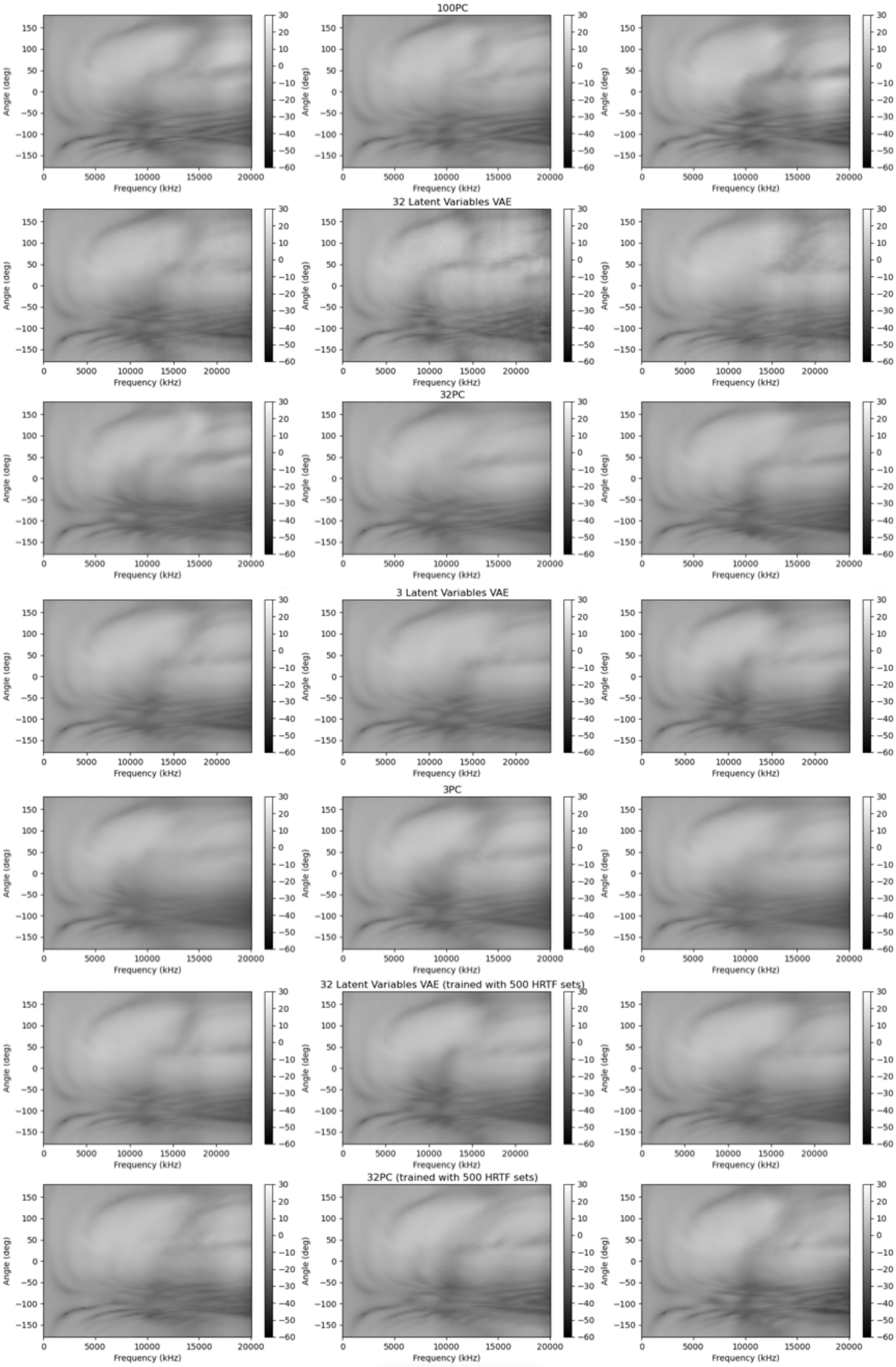


Figure 6.5: Random generated synthetic HRTF sets from different models, three samples are randomly generated from each model

Model	PCA 100	PCA 3	PCA 32	PCA 32 500hrtf	VAE 3	VAE 32	VAE 32 500hrtf
Mean PSD (sones)	**0.00694**	0.06829	0.03562	0.05593	0.15461	0.28077	0.56806
Min PSD (sones)	**0.00024**	0.03830	0.01649	0.03206	0.05257	0.08162	0.25777
Max PSD (sones)	0.16220	**0.14424**	0.18704	0.20184	0.35121	0.69740	0.93997
PSD Range(sones)	0.16197	**0.10594**	0.17055	0.16978	0.29864	0.61579	0.68220
Frontal azimuth mean error (deg)	16.446	16.136	**15.317**	15.321	17.165	15.631	15.44
Sagittal RMS Error (deg)	**29.3**	30.7	30.2	30.0	33.8	33.1	33.4
Sagittal quadrant Errors (%)	**4.3**	5.8	4.9	4.9	9.4	8.7	8.8

Table 6.3: Predicted model performance with different models

In a manner similar to Chapter 4.4, the results were evaluated objectively through a Perceptual Spectral Difference (PSD) model, as well as May's and Baumgartner's models within the Auditory Modelling Toolbox (AMT) [246–249]. The PSD model was utilised to compare the difference between the reconstructed outputs and the inputs, given that both PCA and VAE can be interpreted as models which compress input data into a smaller scale before subsequently decompressing it to reconstruct the initial information. The PSD model should be able to evaluate the amount of perceptual difference introduced in this process. Meanwhile, May's and Baumgartner's models were employed to estimate localisation performance. Each model was trained with either 500 or 5000 HRTF training sets. It was theorised that PCA with 100 PCs should offer superior performance and thus be the benchmark. However, it was later found that this was not consistently the case. It should be noted that none of the VAE models were considered optimised. As such, delving too deeply into these results may not be beneficial, but they should still provide valuable insights for determining the direction of future work.

Table 6.3 provides an overview of the PSD and localisation estimation for all models. The mean and max PSD for all models were observed to be significantly below 1. This suggests that perceptual differences for human listeners should not be noticeable, despite the data loss from the compression and decompression processes. The key takeaway is that according to the PSD, both PCA and VAE models perform worse when the training data is limited. Nevertheless, given the significant general difference between PCA and VAE, it is challenging to determine which is more sensitive to the volume of training data. Interestingly, the mean PSD of the VAE with 3 latent variables significantly outperformed the VAE with 32 latent variables. This may be attributed to the reduced number of neurons in the VAE model when the latent space is decreased. Models with fewer neurons, being less data-hungry, could exhibit this behaviour. However, it remains unclear why the max PSD and PSD range of PCA with 3 PCs significantly outperformed PCA with 100 PCs, especially when PCA with 32 PCs was found to perform worse than PCA with 100 PCs. However, it is once again important to note that, as the max PSD for all models was observed to be significantly below 1, the differences between these models are minimal, and any slight difference could potentially be attributed to randomised noise in the data.

In terms of localisation evaluation, PCA appears to perform better, particularly in sagittal quadrant errors, which represent the percentage of significant sagittal localisation errors. Generally, as the number of PCs or latent variables decreases, sagittal localisation performance deteriorates. This trend is largely echoed in the case of Azimuth error, but it remains unclear why PCA with 100 PCs performed worse than PCA with 32 PCs. Finally, the size of the training data appears to have minimal, if any, impact on localisation performance.

6.4 Discussions

Although the PCA model results seem to outperform the VAE results, VAE provides more flexibility for future applications. The VAE presented in this Chapter is for preliminary investigation purposes, despite each model having been slightly optimised to ensure it can represent a reasonable performance for comparison. It is believed that there is plenty of room to improve, especially when the model is optimised for a specific task like HRTF interpolation, HRTF restoration or HRTF personalisation. There are two ways to use a trained VAE model. The decoder of a VAE will always generate something HRTF-like as long as the latent variables are within a reasonable range. By changing the encoder of the VAE, different inputs can be used. For example, for HRTF personalisation, the input can be an ear scan or human feedback; for HRTF interpolation, the input can be a set of sparse HRTF sets; for HRTF restoration, the input can be a noisy or distorted HRTF measurement. By back-propagating through the new model with the decoder's weights and biases fixed, a new model with a different input can be created, which almost guarantees the output will be something HRTF-like. Another way of seeing it is that the latent space is a compression of the entire HUMMNGBIRD HRTF database. HRTF sets can be fetched with the right latent variables. Thus, with a well-developed latent space, future research can solely focus on finding the latent variables based on different inputs for different applications.

Another use of the VAE is for transfer learning. In the case where the output of the model should be like some other datasets instead of the HUMMNGBIRD database, transfer learning can be applied to the trained VAE model. By using transfer learning, the required training time and the training data size will be significantly less. For example, for HRTF interpolation or restoration, if the desired output should look like another database, by training the model with a few samples from the target database, the output of the model should look a bit more like the ones from the target database. This method is sometimes also referred to as neural network pretraining, which is one of the common uses of Autoencoder [48, 270, 271].

Applying transfer learning on the trained VAE with acoustic HRTF measurement requires HRTF interpolation that aligns the acoustically measured HRTF set with the HUMMNGBIRD HRTF configuration. A potentially more appropriate application of the trained VAE model lies in retaining the trained decoder section while replacing the encoder to fit the configuration of the desired acoustic-measured HRTF set. By keeping the weights of the decoder's neurons fixed and training only the new encoder, the model can learn to project the acoustic-measured HRTF set into the trained VAE latent space. After proper training, the desired acoustically measured HRTF set will be the input, with the output being the best-matched HRTF sets decoded from the latent space. If necessary, transfer learning can be carefully applied to the newly trained VAE to fine-tune the model, despite the possibility that it might negatively impact the model's performance.

6.5 Conclusion

In this chapter, the HUMMINGBIRD database was used to evaluate and compare the performance of PCA and VAE models. These models are commonly used for dimensionality reduction and data reconstruction. The results of the analysis revealed that PCA generally outperformed VAE in terms of accuracy. However, in a more challenging scenario with a latent space of 3 variables, VAE demonstrated an advantage over PCA. This suggests that VAE may be more suitable for handling specific tasks when a small latent space is needed. According to the plots in 6.4, it was observed that the performance of VAE with 32 latent variables was more sensitive to the amount of training data than that of PCA with 32 PCs. This observation was made despite the fact that 6.3 indicates both models' performance was compromised when the training data were limited. This means that VAE requires a larger amount of data to learn and generalise effectively, while PCA is able to achieve good performance with a smaller amount of training data.

It is worth noting that neural networks, such as the VAE, have the ability to be modified and tailored for specific applications through changes to the model architecture, techniques, and hyperparameter tuning. Transfer learning can also be applied to trained models to alter the output with less training data and time. However, due to time constraints, this study concludes at this point. Further research and experimentation are needed to understand the strengths and limitations of these models fully and to identify the best approaches for different applications.

Chapter 7

Conclusion

The main contribution of this thesis is to gather the HRTF data required for machine learning. Chapter 2 introduced the necessary information about HRTFs and ML. It also highlighted some of the challenges of using ML in HRTF research. Chapter 3 presented the attempt to gather HRTF sets through consolidation. A MATLAB toolbox was created for the pre-processing, data clean-up, finding matched angles across multiple HRTF databases, and HRTF angle visualisation. Chapter 4.2 presented a novel way to restore the distorted HRTF information from SH-interpolation through a neural network model. This demonstrated training an ML model with the consolidated HRTF data with the consolidation toolbox created in Chapter 3. The goal was to use the model to improve HRTF interpolation results to be used in future HRTF consolidations without removing the HRTF measurements with mismatched angles. However, the result is not good enough due to the lack of HRTF training data and the nonuniform data across different databases. This became the motivation for creating a large HRTF database with a uniform configuration and non-skewed data. Chapter 5 documented the creation of the HUMMNGBIRD database, a large HRTF database with 5000 BEM simulated HRTF sets. The head models of the HRTF sets were created with a 3DMM sampled from multiple databases, which have a wide variety of ages from under 5 to above 80 and are nearly balanced in gender. This method allows researchers to create unlimited HRTF sets with different configurations in the future. Chapter 6 applied the HUMMINGBIRD database in PCA models and VAE. The results show that PCA, a traditional statistical model outperforms VAE in most cases. However, in a more challenging scenario, when the latent space is down to 3 variables, VAE seems to have an advantage against PCA. However, the performance of a VAE is more sensitive to the amount of training data. The number of training data affects the performance noticeably. With that being said, one of the benefits of using neural networks is their flexibility to modify the model for specific applications. For different specific tasks, neural networks often have room to improve with different architectures, techniques and hyperparameter tuning. Due to the time constraints of the PhD, this is the point where this thesis concludes.

7.1 Restatement of Research Hypotheses

The hypotheses originally stated in Section 1.2 can now be restated as follows:

1. That a large number of HRTFs could be beneficial for machine learning applications, especially for training neural network models.

The results from Chapter 4 show that increasing the training data of a neural network from 50,000 HRTFs to 633,000 HRTFs improves the performance of the model significantly. The results from Chapter 6 further prove that increasing the training data from 500 HRTF sets to 5000 HRTF sets has a positive effect on the results with both PCA and VAE. This shows that no matter whether the ML model uses HRTFs or HRTF sets, having more training data will be beneficial for the model's performance. By comparing the results between PCA and VAE, it shows that the VAE is more sensitive to the amount of training data. This partly confirms the hypothesis that neural network models could benefit from the extra data more than other ML models like PCA.

However, even training with limited data, which is 50,000 HRTFs for the work in Chapter 4 and 500 HRTF sets for the work in Chapter 6, all the models showcased in this work still perform reasonably. An examination of the PSD results in Chapter 6 reveals that the max PSD values between the input and equivalent reconstructed outputs are all significantly below 1. This suggests that even unoptimised models should provide adequate performance for general use. In this case, the time and computational cost associated with model training and data generation must be considered, as the expense may surpass the benefit. Therefore, instead of stating that a large number of HRTFs are required for machine learning applications, it is more accurate to restate it as a large number of HRTFs could be, but not always beneficial for machine learning applications, especially for training neural network models.

2. That a feasible alternative to measured HRTFs can be derived through computational means to facilitate large-scale HRTF data generation for machine learning.

The HUMMNGBIRD HRTF sets presented in Chapter 5 show that an alternative to measured HRTFs can be derived through computational means. By using 3DMM to generate synthetic head models and then using BEM simulation to generate HRTFs, unlimited HRTF sets can be created. Chapter 6 shows that the HUMMNGBIRD HRTF sets work well with machine learning models like PCA and VAE. The only issue with the HUMMNGBIRD HRTF sets is the slight high-frequency shift. However, it is believed that by adjusting the configurations in the BEM simulation, the issue can be solved in future iterations.

7.2 Future Work

7.2.1 HRTF Consolidation Tools

When this work was published, this tool was fully functional for consolidating and pre-processing HRTF sets for binaural research and machine learning. It has been tested on all the HRTF sets that were available at that time. However, since then, plenty of new HRTF datasets have been published. Alongside that, the SOFA version has been updated a few times. Maintaining the tool and keeping it up-to-date in such a fast-developing field is challenging. However, future work will look to add more features to the toolbox that could serve broader types of studies and algorithmic development.

Graphical User Interface (GUI): The current version uses the command window to operate the tool, including selecting different options during the process. A GUI interface should allow researchers to have an understanding of what functions are available and provide better insight into the results. A GUI also allows for adding more features in the future whilst remaining user-friendly. It should be more future-proof as it can present the HRTF dataset information more clearly before starting the process.

Visual-based Angle Selection: Angle selection could give researchers the convenience of selecting their desired output data. After the matched angles are found and plotted, researchers would be able to choose the measurements they want to save instead of outputting all matches. Selecting the desired matched angles before outputting them into a new file provides extra flexibility, faster processing time, and saves storage space from redundant data.

Visual-based Attributes Readjustment: Finding matches between HRTF databases can be difficult, especially with a narrow range of tolerance. When no match is found, researchers may want to remove a few databases instead of increasing the capacity to preserve high precision or increasing the range of tolerance to find more matches. Plotting the angle distribution of each represented file or the different results between different combinations could help the user make an informative decision. In addition, adding this feature should provide more possibility to adapt some new HRTF datasets.

Readjusting Parameters After Matching: Depending on the input parameters, the matching results may not always be ideal, such as too few matched angles or their distribution not broad enough. Sometimes, changing input parameters may generate better outcomes. To bypass most of the time-consuming processes, after the function finds and plots all the matched angles, researchers would have a chance to readjust the input parameters based on the current result. Combined with the GUI, users may also quickly estimate the new input parameters by just comparing the azimuth and elevation angle differences.

Extract Ear Parameters The relationship between ear parameters and HRTFs is popular in binaural research. Although the output summary includes the original file names, finding their ear parameters can still be challenging as they are stored in different formats depending on the database. However, it is an excellent bonus if this tool can extract the ear parameters from some popular datasets to make this tool more useful.

7.2.2 Low-order Spherical Harmonic HRTF Restoration using a Neural Network Approach

Although the results show that a simple ML model can be used to restore distorted SH interpolated HRTFs, the state of this model is far from optimised for applications. It is believed that there will be a significant improvement if more HRTF measurements are available for training in the future. One way to improve the model without extra data is through hyper-parameter tuning, including the parameters for regularisation. Noise injection is a different regularisation method that has been shown to work better than weight decay in some cases [251–253]. By picking the right parameters, it is believed that it could generalise better across measurement methods as the model could focus on the general information across various HRTF measurements as opposed to the artefacts introduced by different measurement methods.

Another problem that was observed from the current model is the localisation performance. Although it shows some improvement in the horizontal plane, the sagittal error needs further consideration. As discussed in Section 4.4, it is believed that the smooth L1 loss function only compares the difference in each frequency and fails to capture other useful metrics of HRTFs. Potentially, some custom loss functions can be implemented to improve the model. Gatys et al. [254] trained a separate machine-learning model for content loss and implemented a special function for style loss. Similar methods, like training a localisation model as a localisation loss function, may be able to improve the problem. Furthermore, with the recent success in generative adversarial networks (GAN), it should be possible to build a GAN based on localisation performance [16, 18, 255–257]. However, as a GAN can be unstable to train and it usually requires a lot of tuning, it may not be the most effective way for SH interpolated HRTF restoration.

In order to obtain more HRTF sets to train the model, the HUMMNGBIRD dataset was created after this work. However, with uniform data from the HUMMNGBIRD HRTF, and the development in computational power, this model may no longer be the best way for HRTF interpolation. This work is suitable when the HRTF sets used in training have different distributions. This is because the SH interpolation process and a model that focuses on one HRTF at a time can adapt different distributions of HRTF sets. At that time, there was no single HRTF database with sufficient data to train the model. With the HUMMNGBIRD HRTF sets that share the same distributions, there is a more straightforward way for HRTF interpolation.

Maintaining knowledge of the latest advancements in this rapidly evolving field is crucial. Equally important is the ability to discard outdated ideas, despite any sunk costs. Nevertheless, these processes serve as significant stepping stones, enabling the creation of a vast HRTF database. Consequently, this database allows for the utilisation of more advanced algorithms and hardware technologies.

7.2.3 Generating HRTFs with a 3D morphable model of human heads

The HUMMNGBIRD project, with its applications shown in Chapter 6, shows that numerically generated HRTFs alongside ML can be powerful. The biggest issue with the current database is the high-frequency shift. The shift could be caused by the BEM simulation configuration, including the speed of sound and density of air and the variance in head size. There are plans in place to generate a new batch of HRTF sets without the frequency shift by testing different configurations, adjusting the scaling of the head model, adapting the latest Mesh2HRTF version and process, and trying the different mesh grading methods. There is also a plan to speed up the generating process by optimising the script, specifically in post-processing. By optimising the process, different kinds of HRTF data can be generated for specific HRTF research, for example, near-field HRTFs, HRTFs with different head and torso orientations, HRTFs with different speed of sound or density of air, different measurement grids, and HRTFs of specific head models. The extraction of anthropomorphic data from the head model could potentially prove beneficial for some researchers, facilitating more comprehensive acoustic analyses.

7.2.4 Preliminary Investigation on the Potential of Using Extra HRTF Dataset in Machine Learning

The preliminary result shows that in an extreme case, VAE can outperform PCA. VAE, a neural network model, also provides more flexibility in modifying the model for specific applications. For example, considering the latent space can be small as 3 latent variables, by swapping the encoder of VAE with different models and inputs, different tasks can be performed. By changing the encoder and input into sparse HRTF sets, the model can perform HRTF interpolation. By changing the encoder and input with ear scans or user feedback, the model can potentially perform HRTF personalisation tasks. It is all about how to let the encoder navigate the latent space to find the best HRTF set as output. At the same time, the result shows that there is room to improve the quality and variance of the decoder output. Considering the trade-off between the HRTF set quality and the size of the latent space. Having a specific and narrow task in mind would be important to determine what to do next. Given the time and resources required for optimising an ML model for particular applications, additional experimentation beyond the scope of this thesis is required to determine the appropriate direction for future work.

7.3 Closing Remarks

In this thesis, the progress of developing training data for machine learning ML was presented. It is widely believed that the quantity and quality of training data are crucial for building a useful ML model. The research conducted throughout this PhD aimed to lay the foundation for applying more data-intensive ML models in future HRTF research. Although the results of this research have not yet provided a complete solution to the problem, there is hope that with further improvements to the HUMMNGBIRD database, this goal can be achieved. Given the advancements in other ML fields, such data may be the key to driving innovation in HRTF research. The experiment with PCA and VAE demonstrated that it is potentially possible to compress a lot of HRTF sets into a small latent space. With a well-developed latent space, future research can focus on methods to navigate the latent space for specific HRTF sets, which should be easier and more robust compared with some conventional methods. Using such latent space can also guarantee the output HRTF sets are HRTF-like. Despite the limitations of this research, the progress made in developing training data for machine learning and the potential for using techniques like PCA and VAEs to compress HRTF data into a small latent space offers promising avenues for future research. It is hoped that these findings will serve as a foundation for more advanced ML models in HRTF research and may potentially lead to new advancements in this field.

Appendix A

Supplementary Plots from the PCA and VAE models

A.1 Reconstruction Results of Subject 5001 from the HUMMNGBIRD

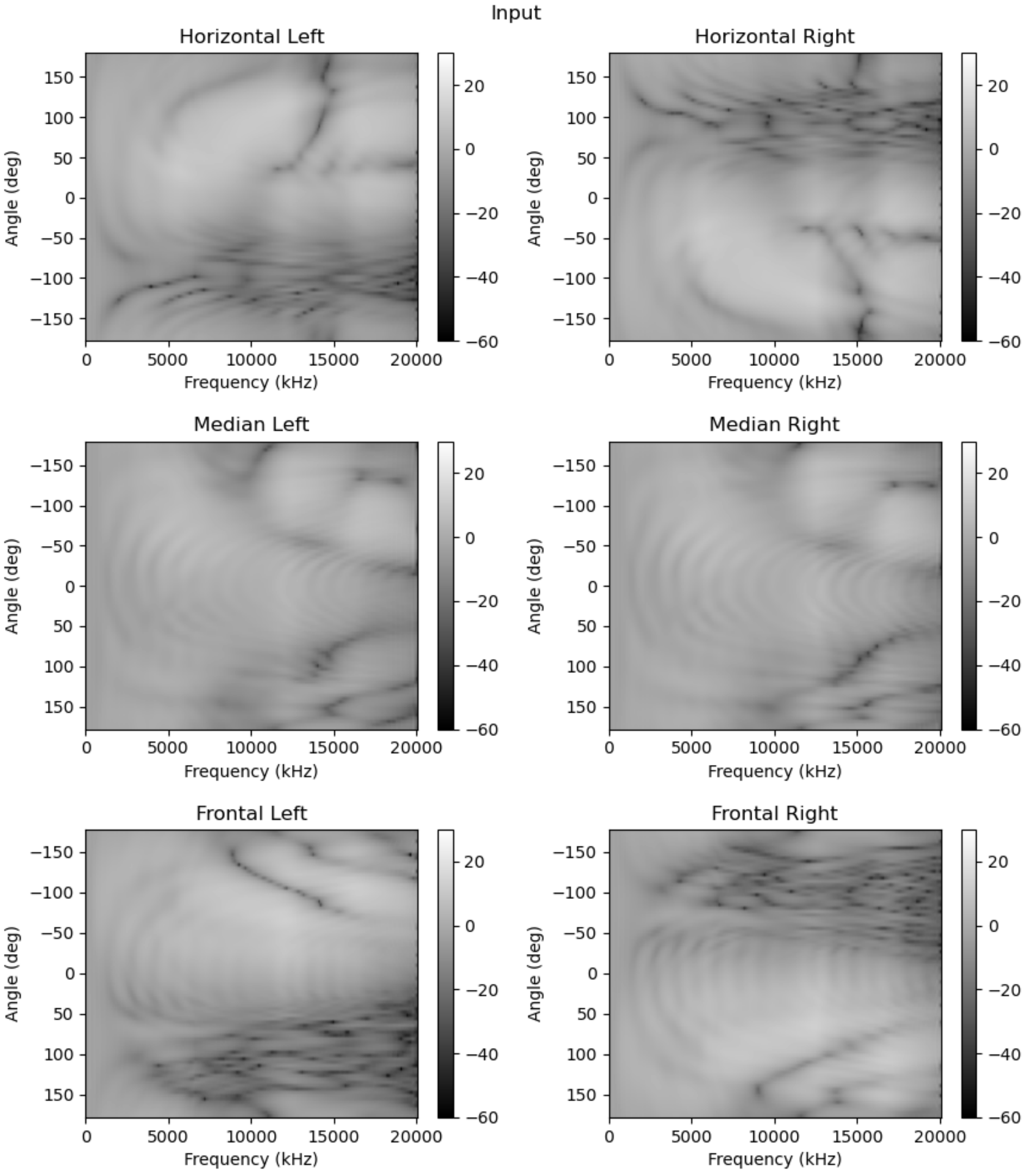


Figure A.1: Input data, Subject 5001 from the HUMMNGBIRD

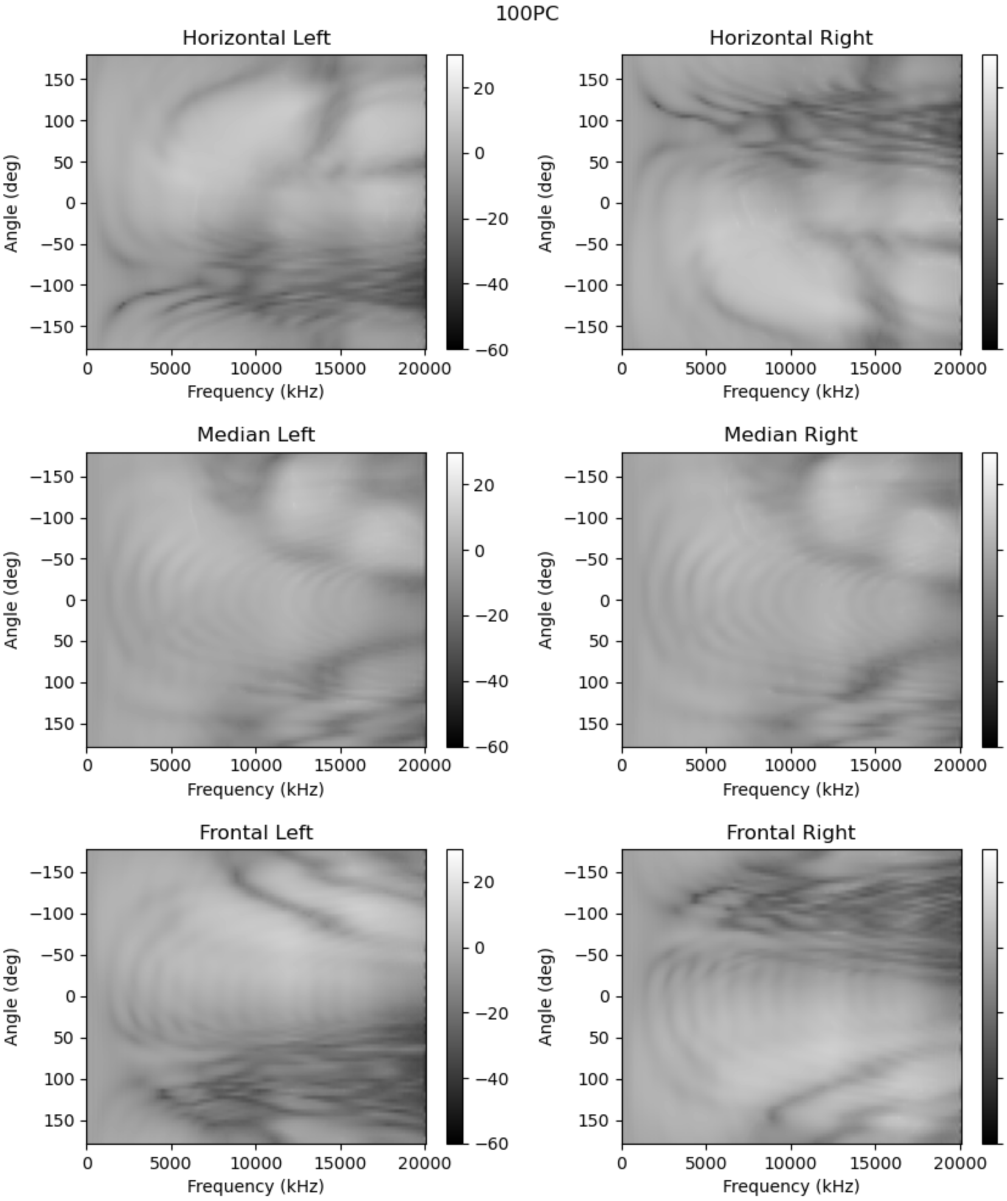


Figure A.2: Restoration output from the PCA model with 100 PCs trained with 5000 HRTF sets

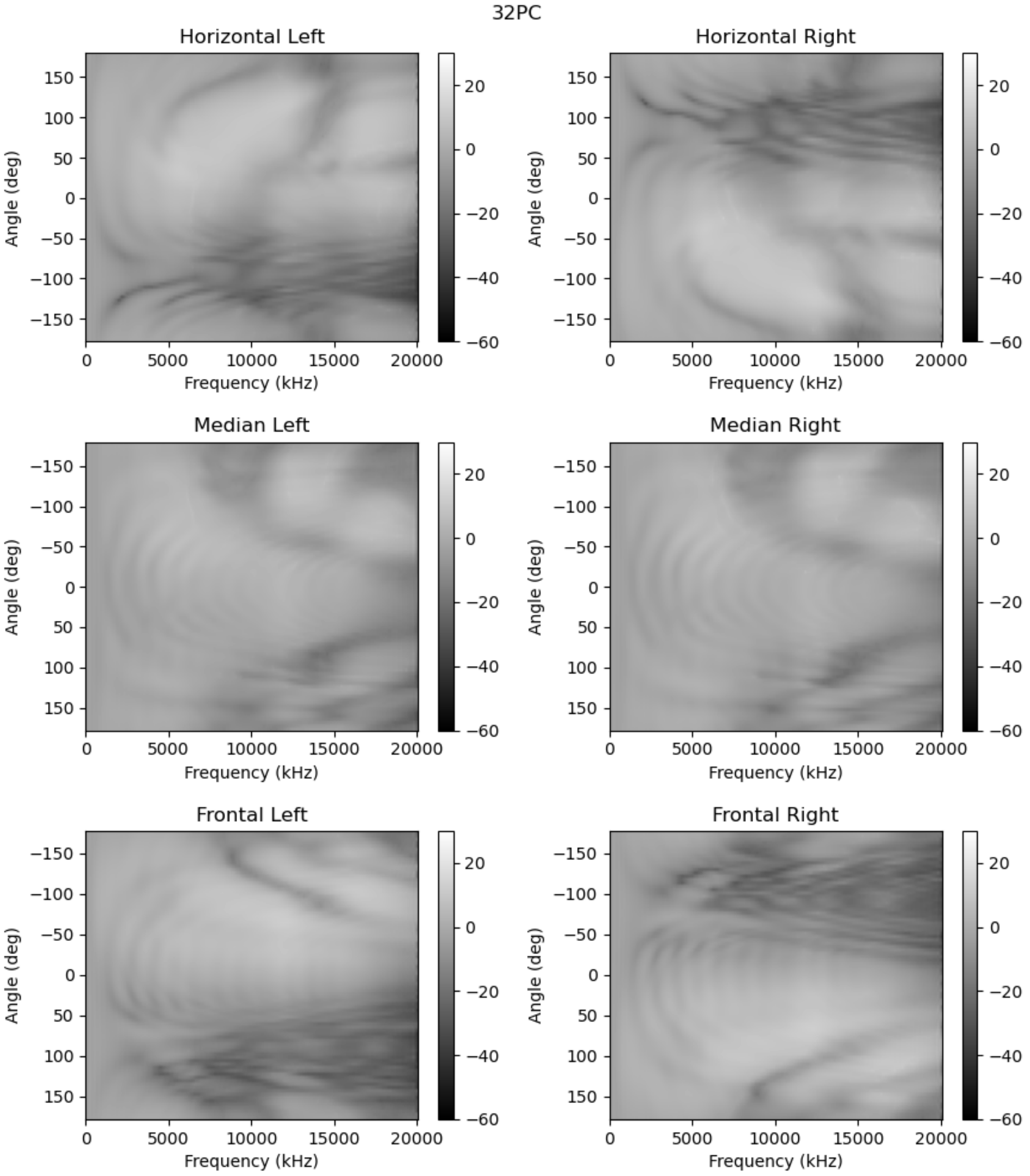


Figure A.3: Restoration output from the PCA model with 32 PCs trained with 5000 HRTF sets

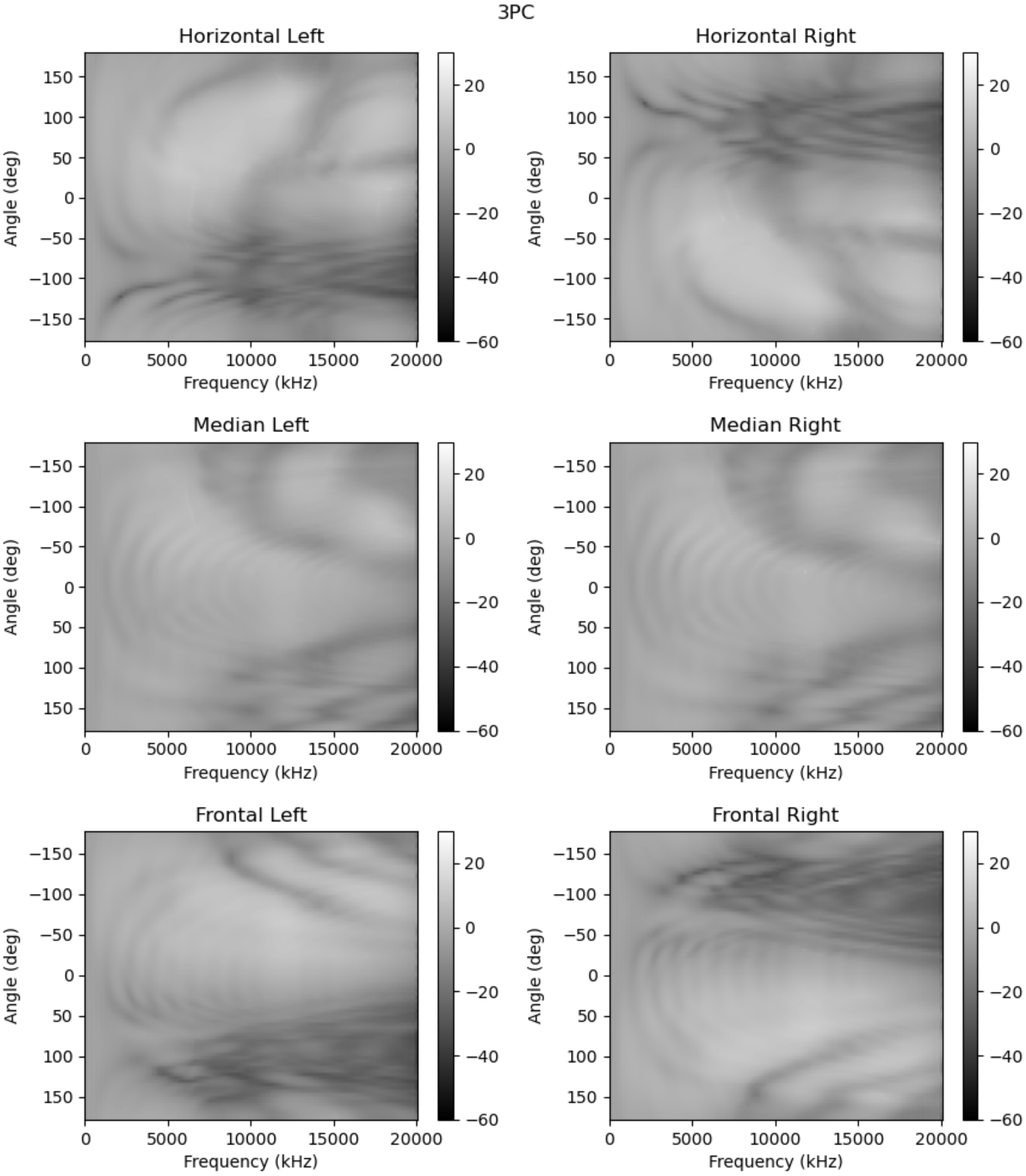


Figure A.4: Restoration output from the PCA model with 3 PCs trained with 5000 HRTF sets

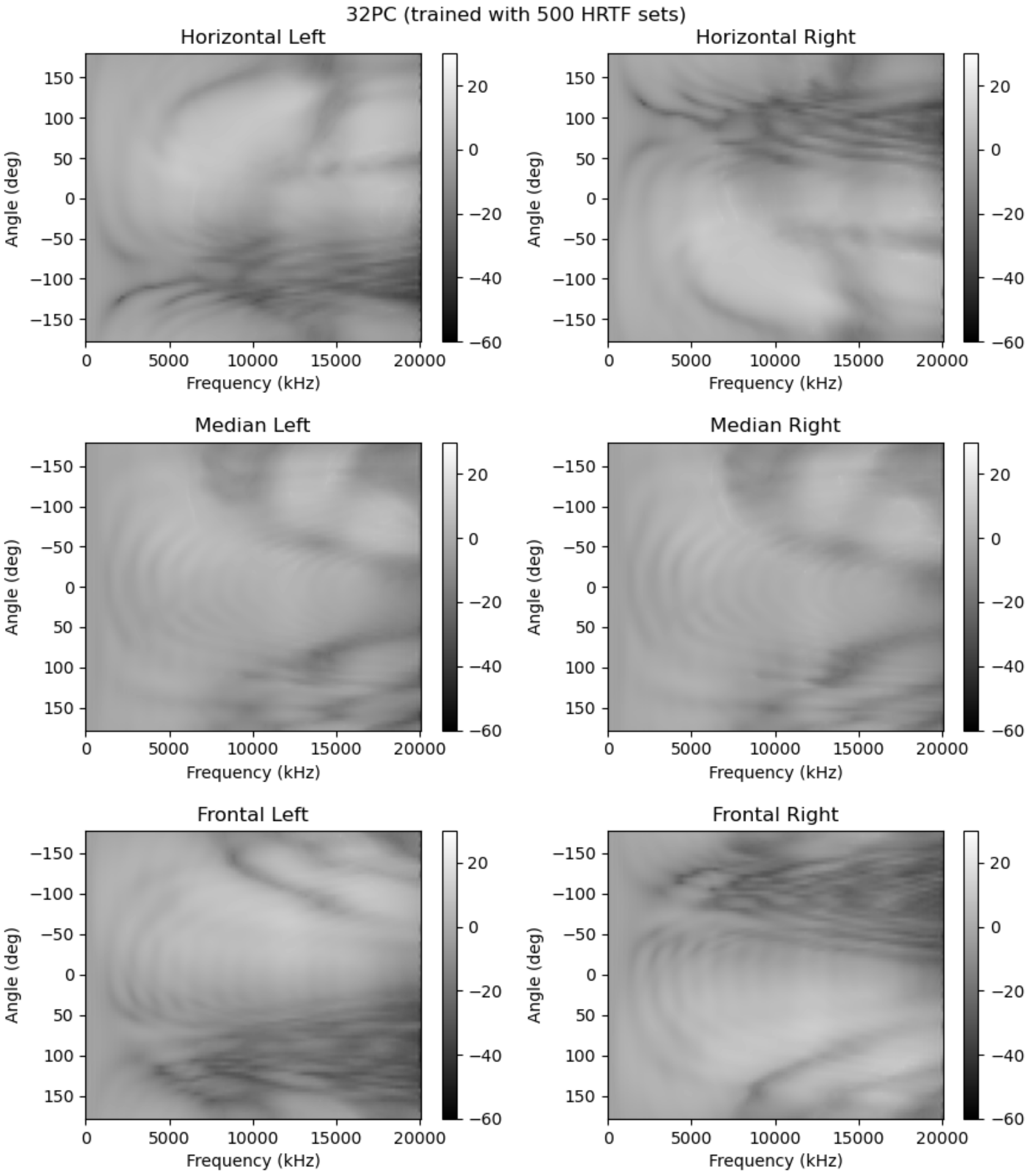


Figure A.5: Restoration output from the PCA model with 32 PCs trained with 500 HRTF sets only

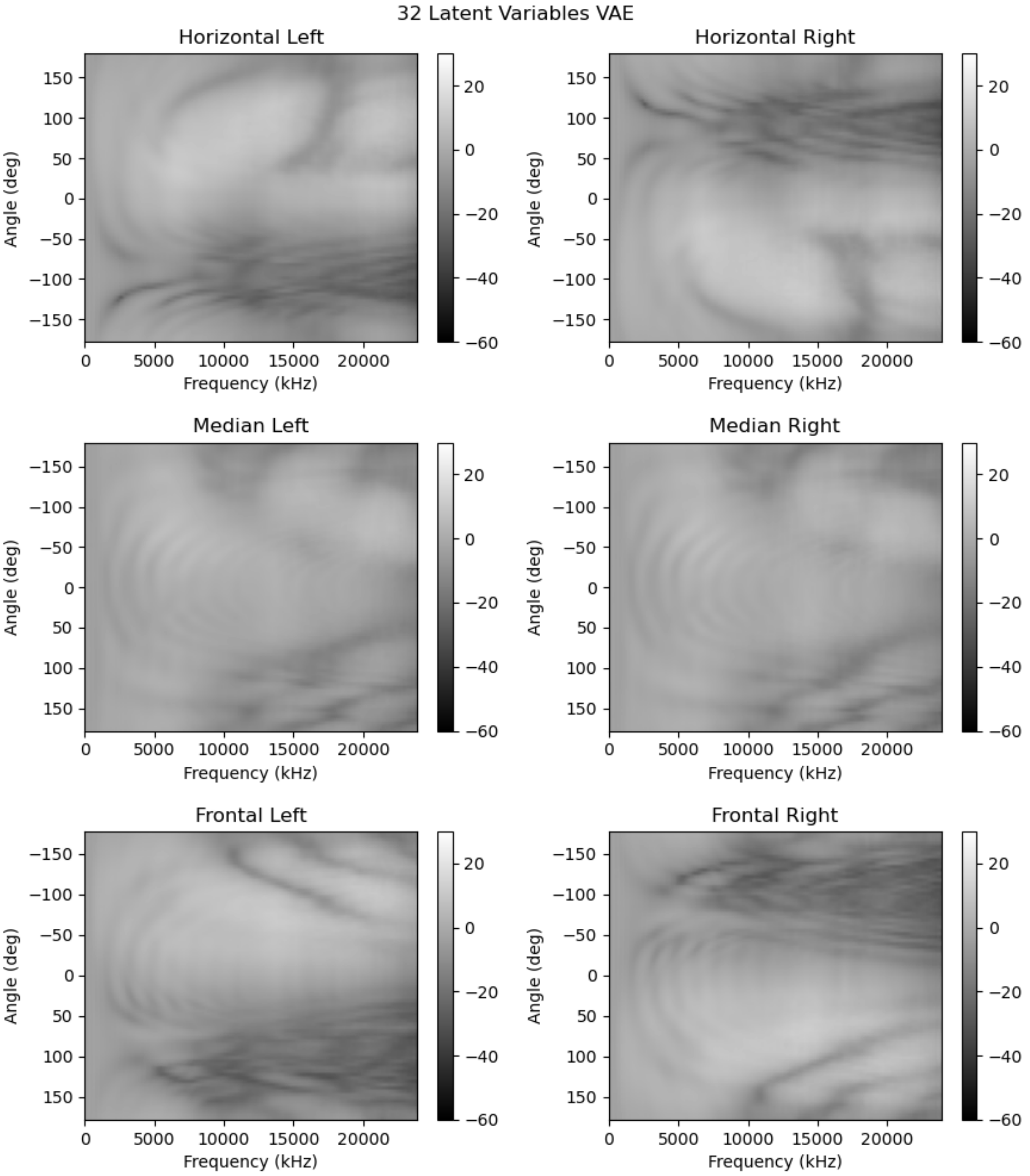


Figure A.6: Restoration output from the VAE model with 32 latent variables trained with 5000 HRTF sets

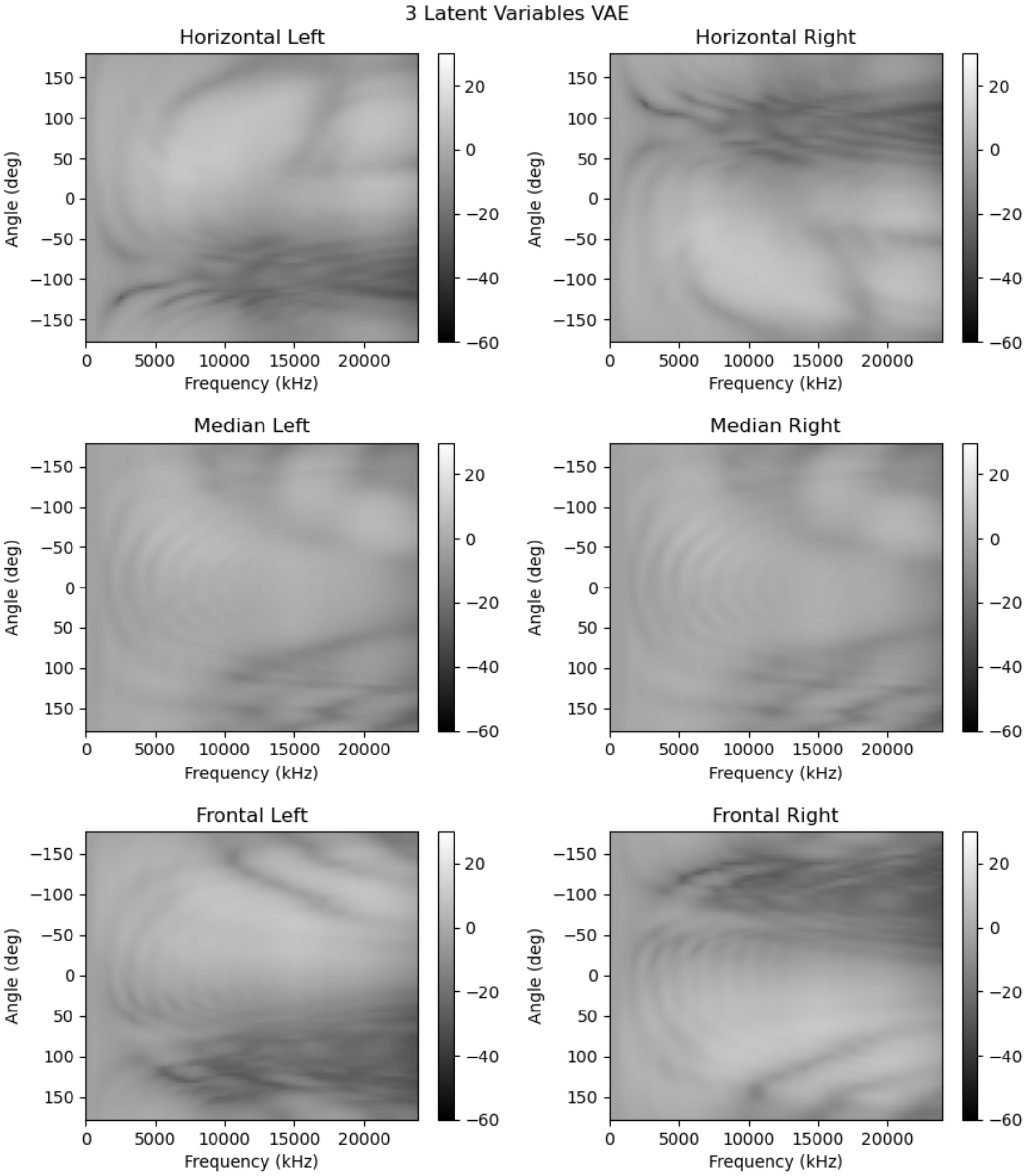


Figure A.7: Restoration output from the VAE model with 3 latent variables trained with 5000 HRTF sets

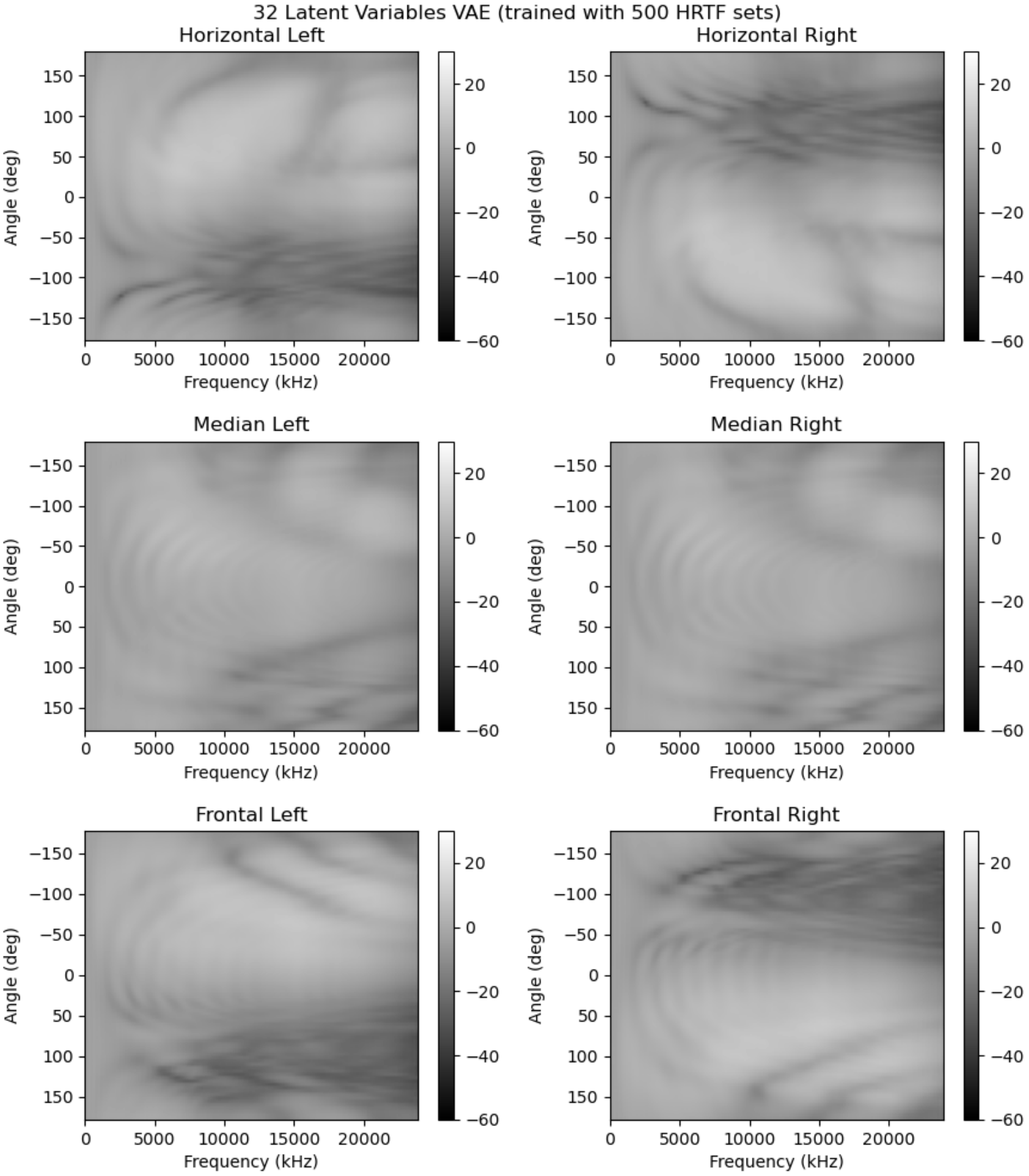


Figure A.8: Restoration output from the VAE model with 32 latent variables trained with 500 HRTF sets

A.2 Standard Deviation of 100 HRTF Validation Results from the HUMMNGBIRD

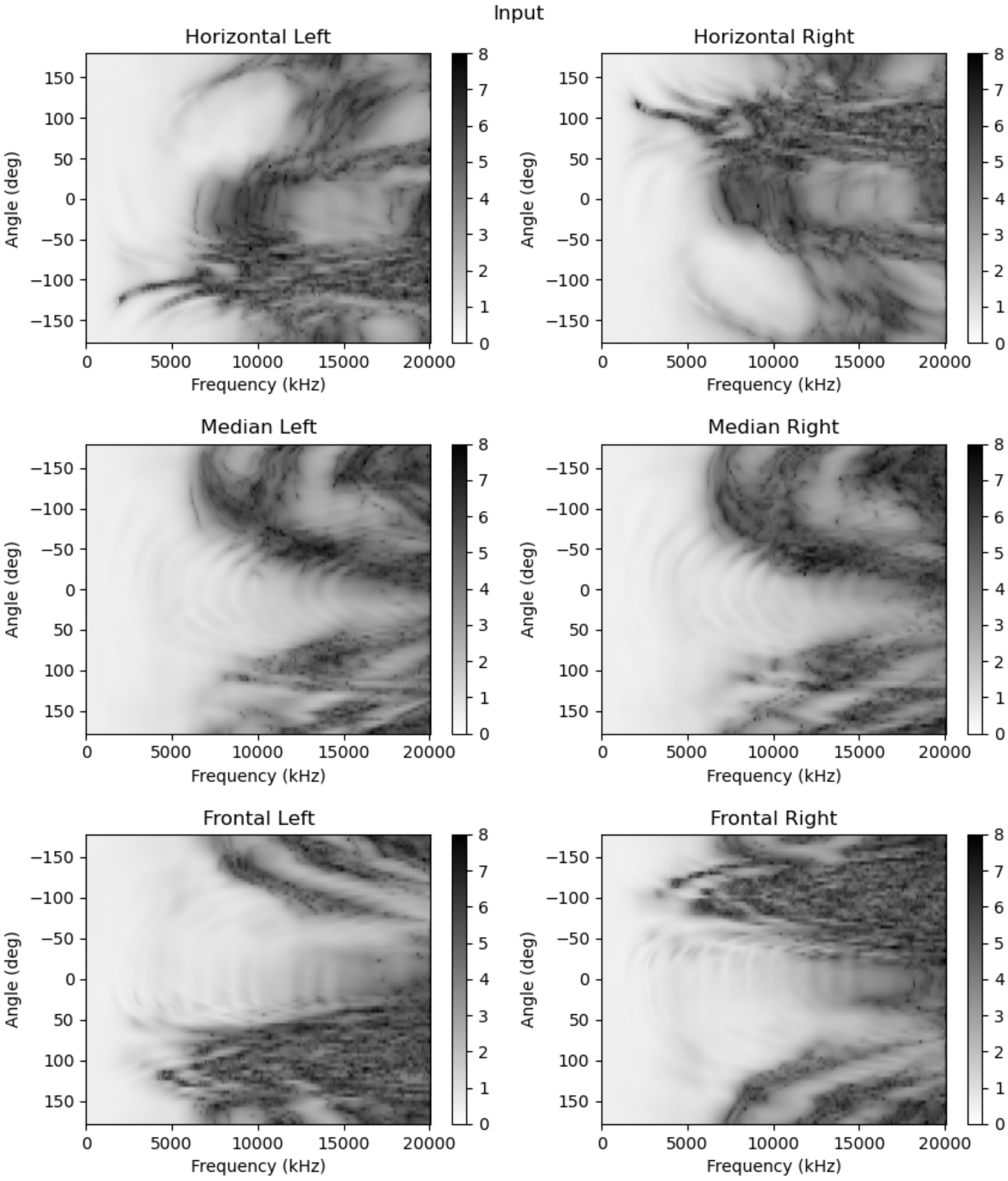


Figure A.9: Standard deviation of 100 HRTF validation input data, Subject 5001 to 5100 from the HUMMNGBIRD

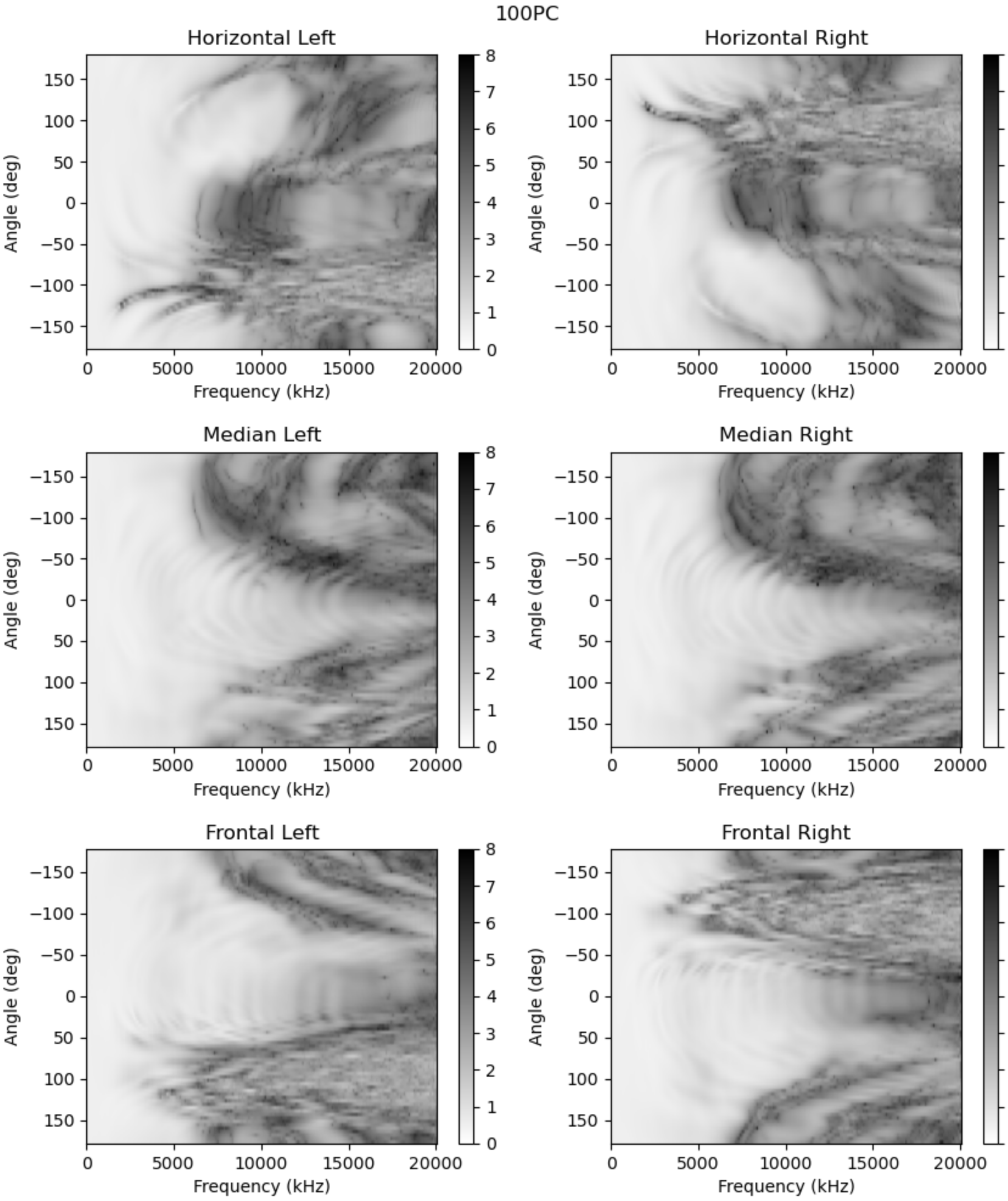


Figure A.10: Standard deviation of 100 HRTF validation output from the PCA model with 100 PCs trained with 5000 HRTF sets

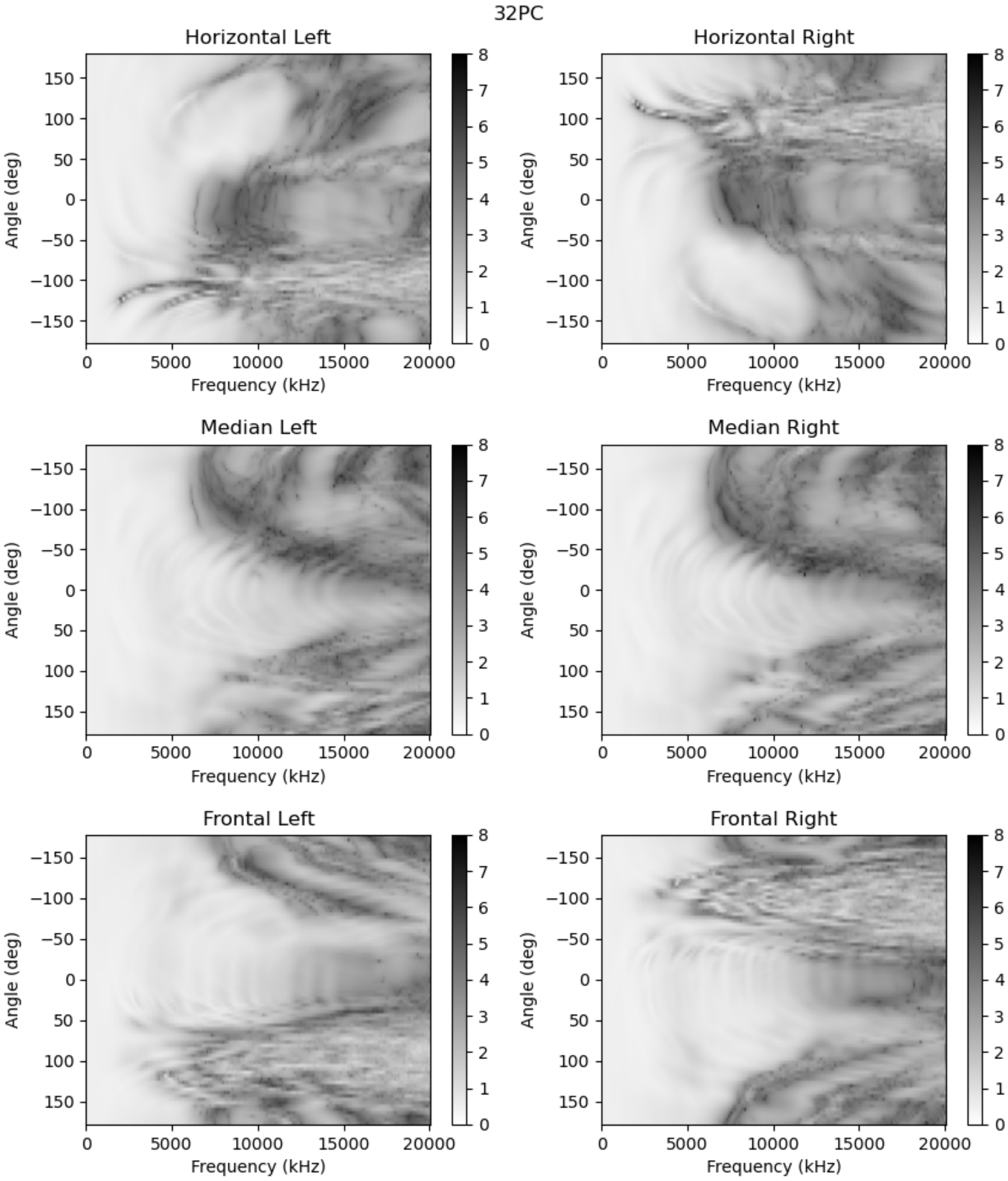


Figure A.11: Standard deviation of 100 HRTF validation output from the PCA model with 32 PCs trained with 5000 HRTF sets

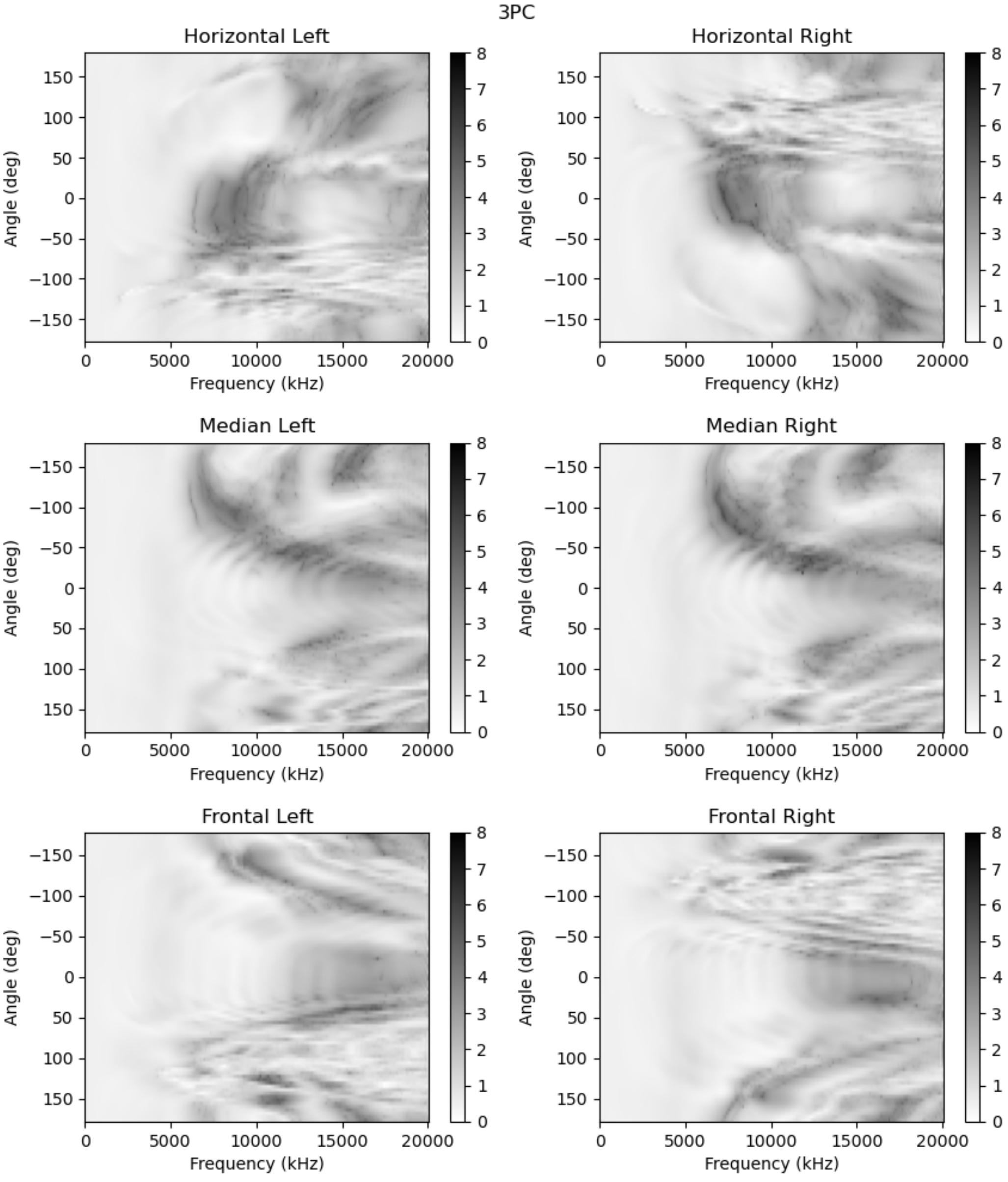


Figure A.12: Standard deviation of 100 HRTF validation output from the PCA model with 3 PCs trained with 5000 HRTF sets

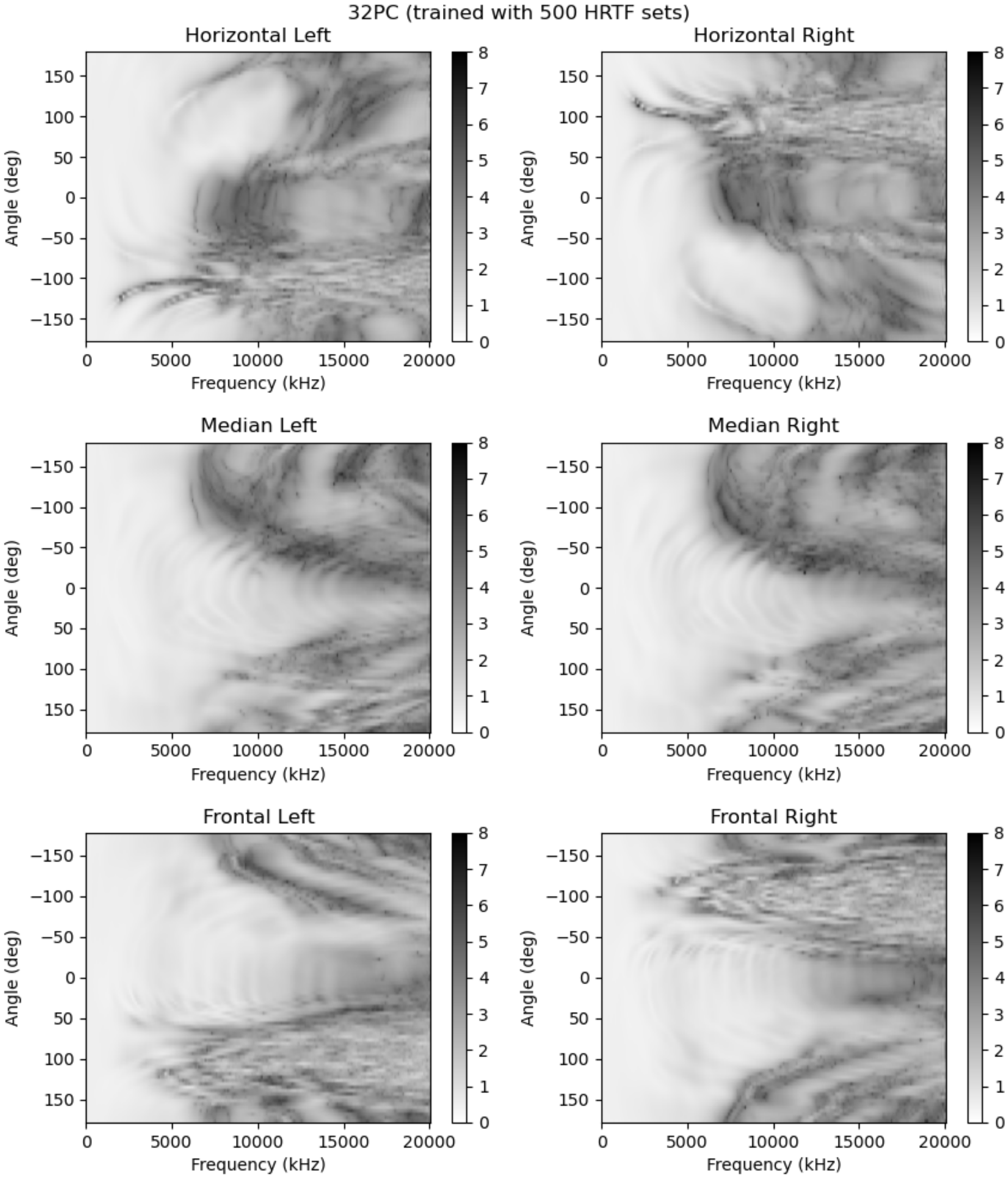


Figure A.13: Standard deviation of 100 HRTF validation output from the PCA model with 32 PCs trained with 500 HRTF sets only

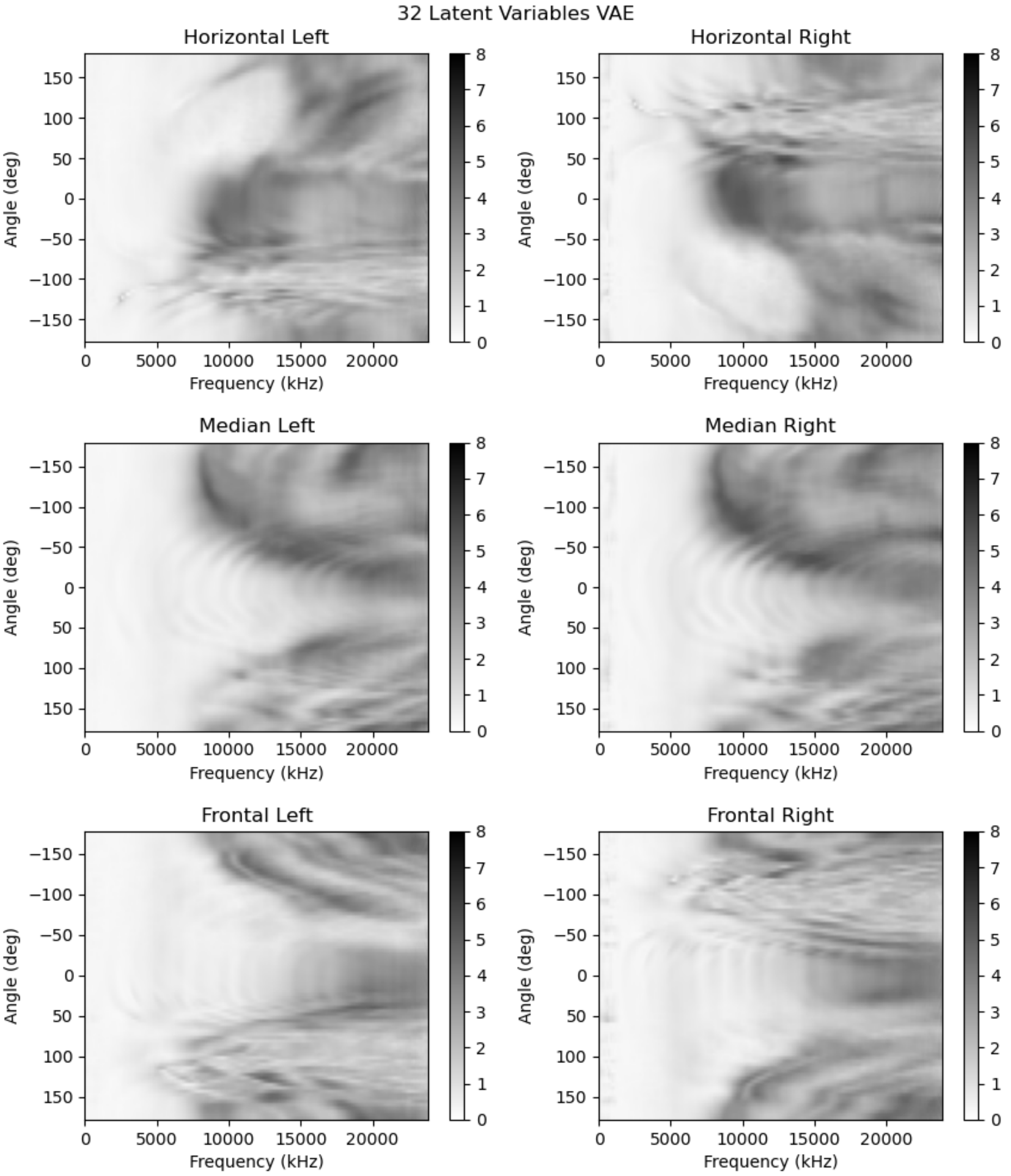


Figure A.14: Standard deviation of 100 HRTF validation output from the VAE model with 32 latent variables trained with 5000 HRTF sets

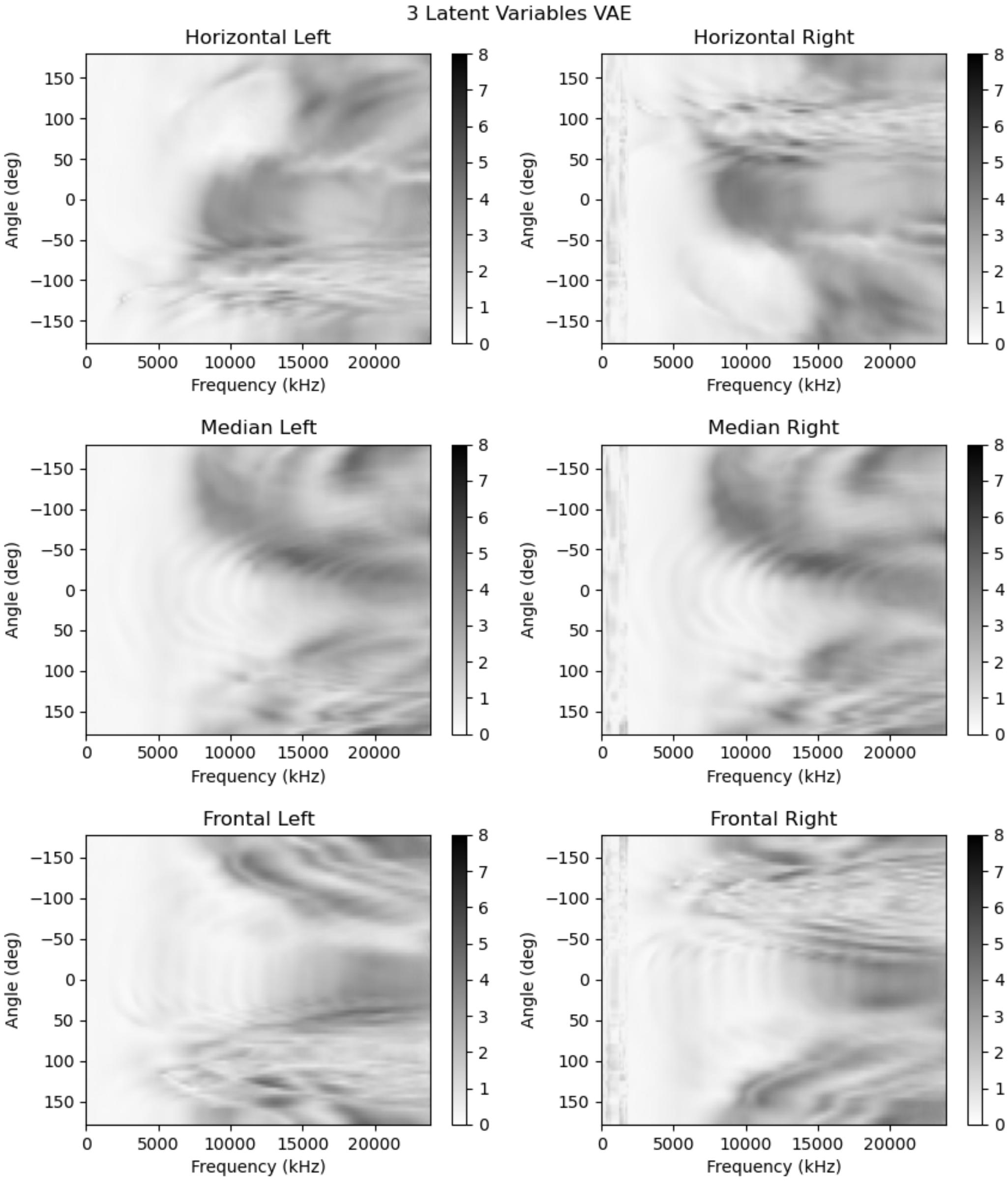


Figure A.15: Standard deviation of 100 HRTF validation output from the VAE model with 3 latent variables trained with 5000 HRTF sets

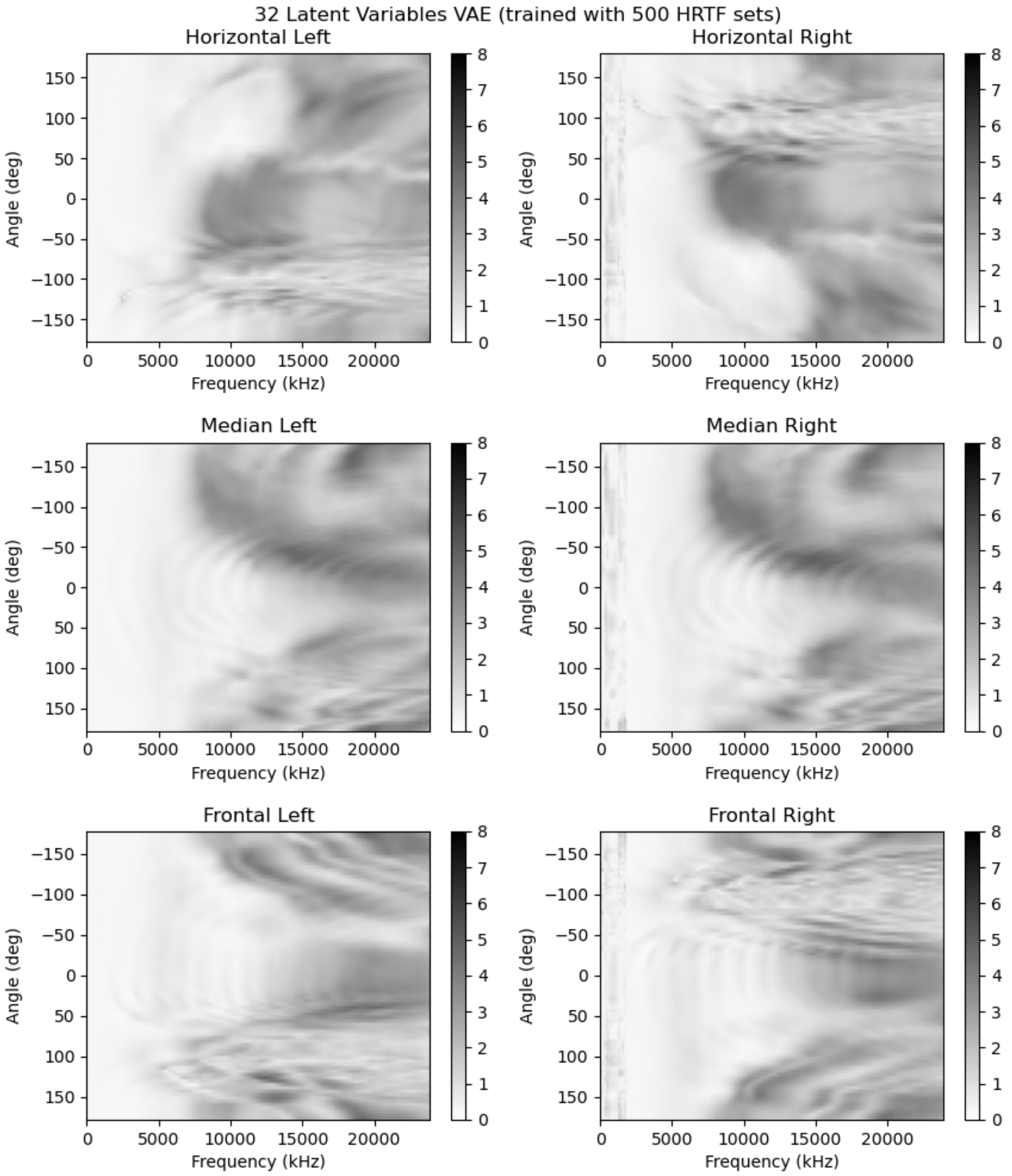


Figure A.16: Standard deviation of 100 HRTF validation output from the VAE model with 32 latent variables trained with 500 HRTF sets

A.3 Random Generated HRTF sets from the PCA and VAE models

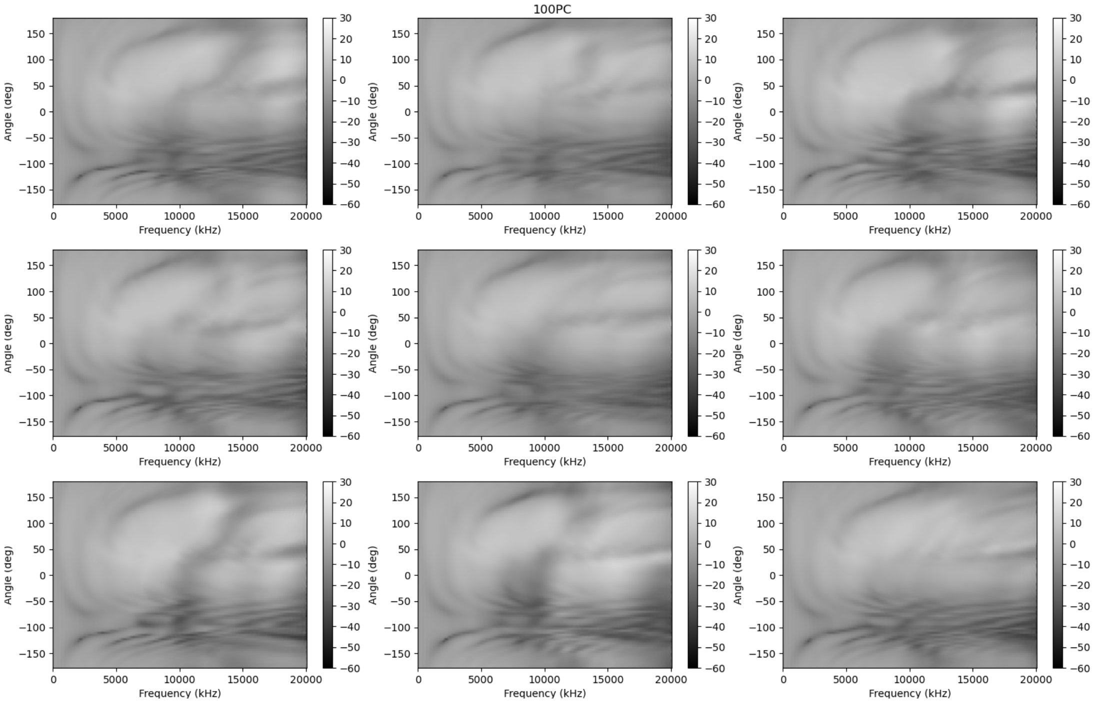


Figure A.17: 9 random generated HRTF sets from the PCA model with 100 PCs trained with 5000 HRTF sets, left channel of the HRTFs on the horizontal plane

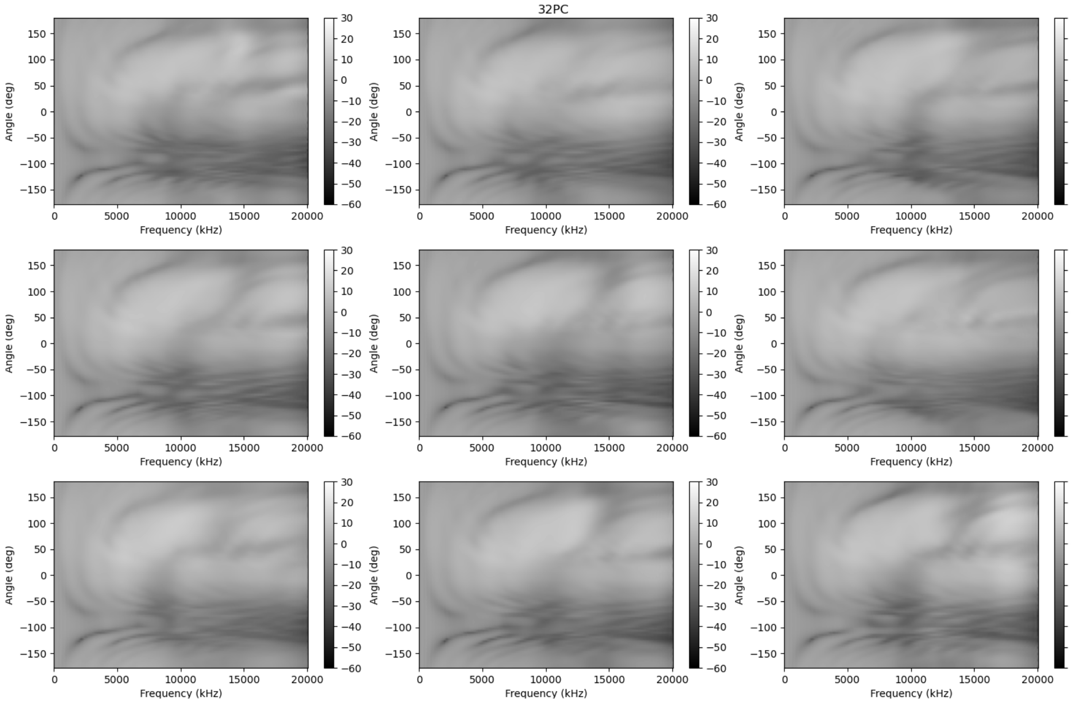


Figure A.18: 9 random generated HRTF sets from the PCA model with 32 PCs trained with 5000 HRTF sets, left channel of the HRTFs on the horizontal plane

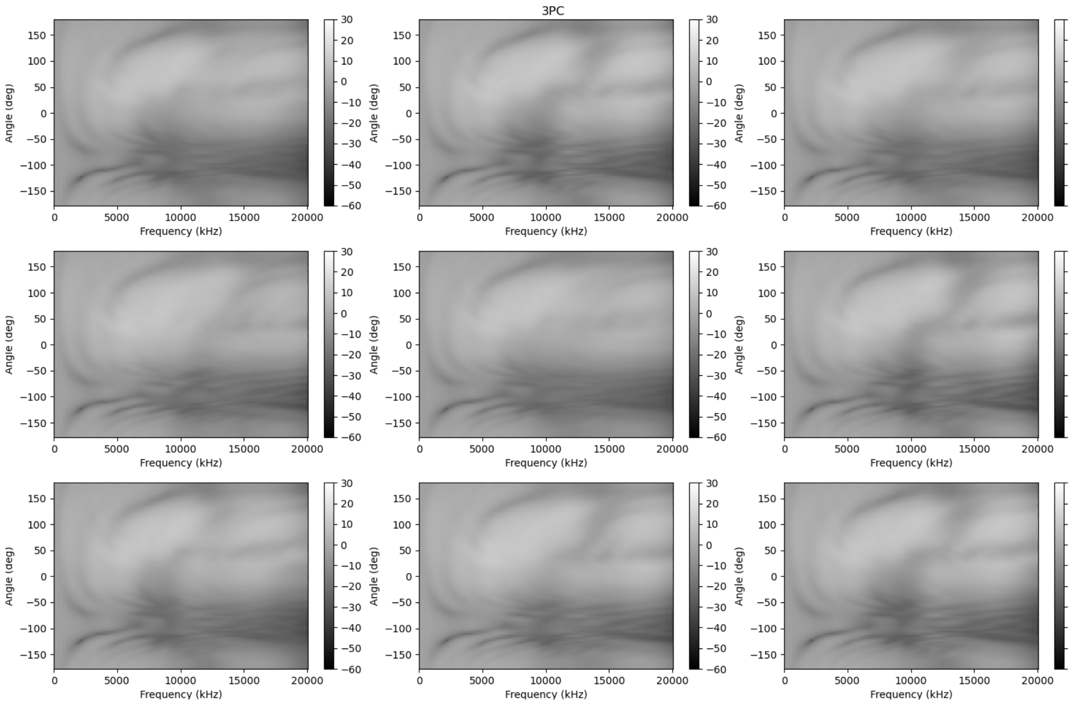


Figure A.19: 9 random generated HRTF sets from the PCA model with 3 PCs trained with 5000 HRTF sets, left channel of the HRTFs on the horizontal plane

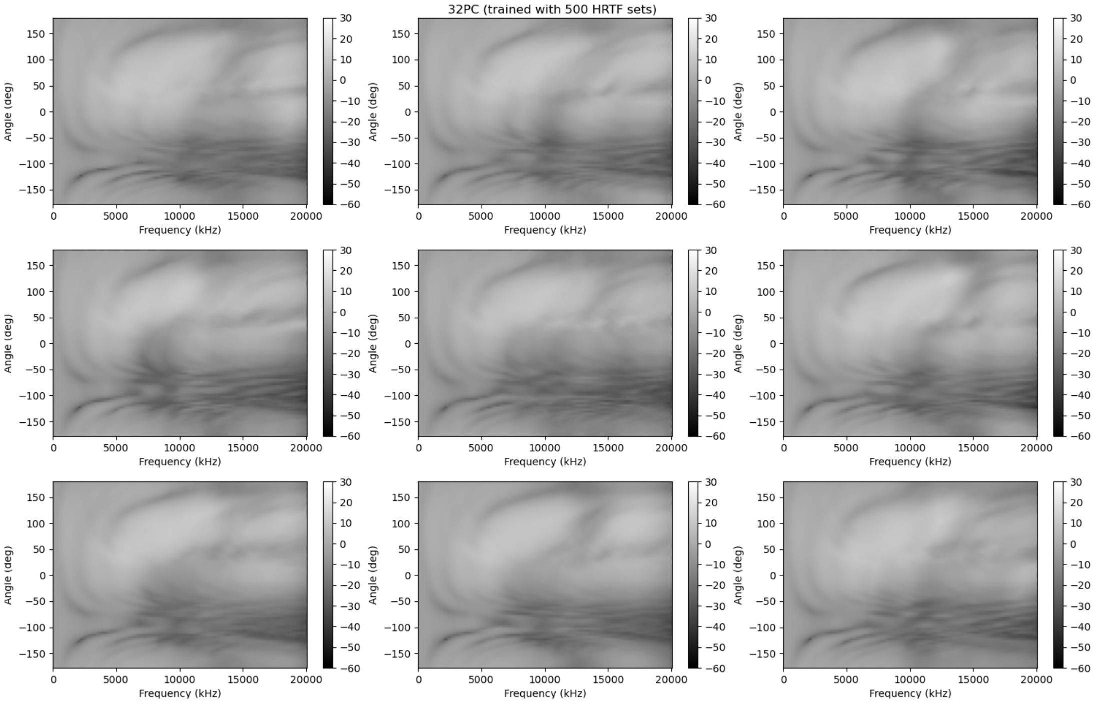


Figure A.20: 9 random generated HRTF sets from the PCA model with 32 PCs trained with 500 HRTF sets only, left channel of the HRTFs on the horizontal plane

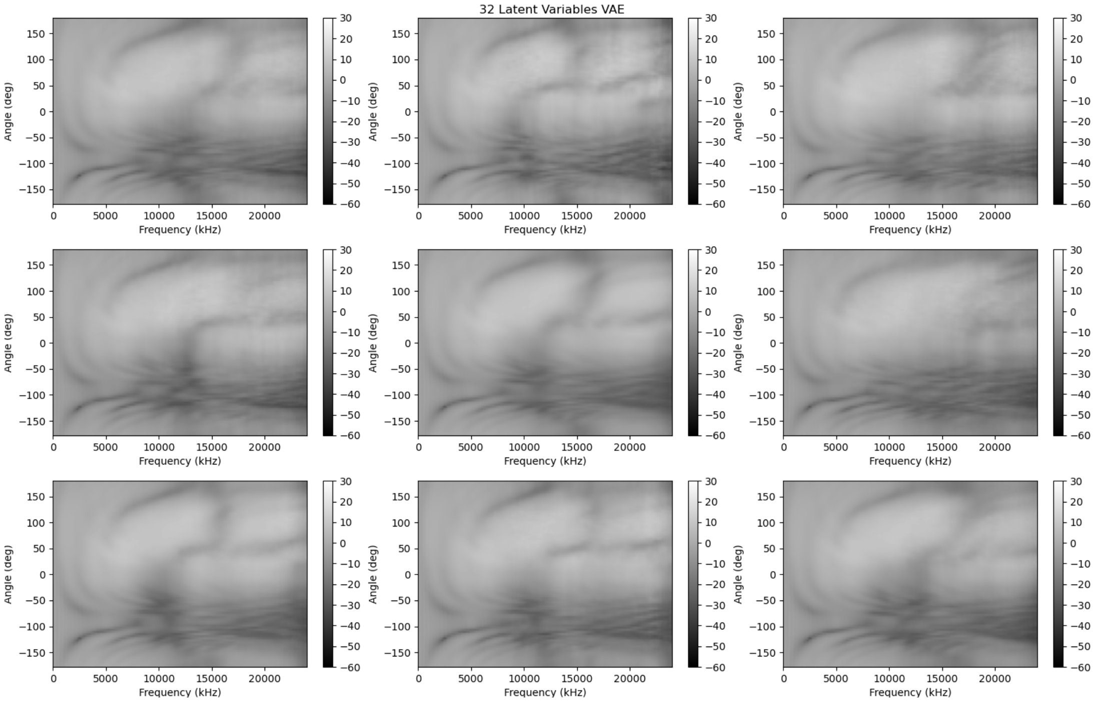


Figure A.21: 9 random generated HRTF sets from the PCA model with 32 PCs trained with 5000 HRTF sets, left channel of the HRTFs on the horizontal plane

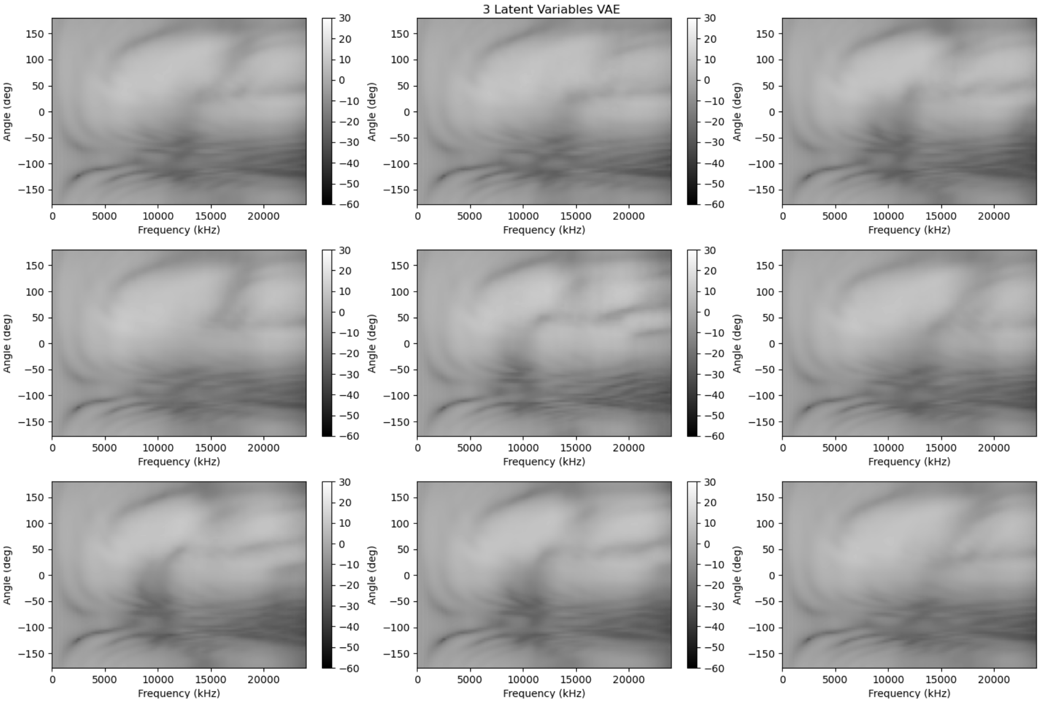


Figure A.22: 9 random generated HRTF sets from the PCA model with 3 PCs trained with 5000 HRTF sets, left channel of the HRTFs on the horizontal plane

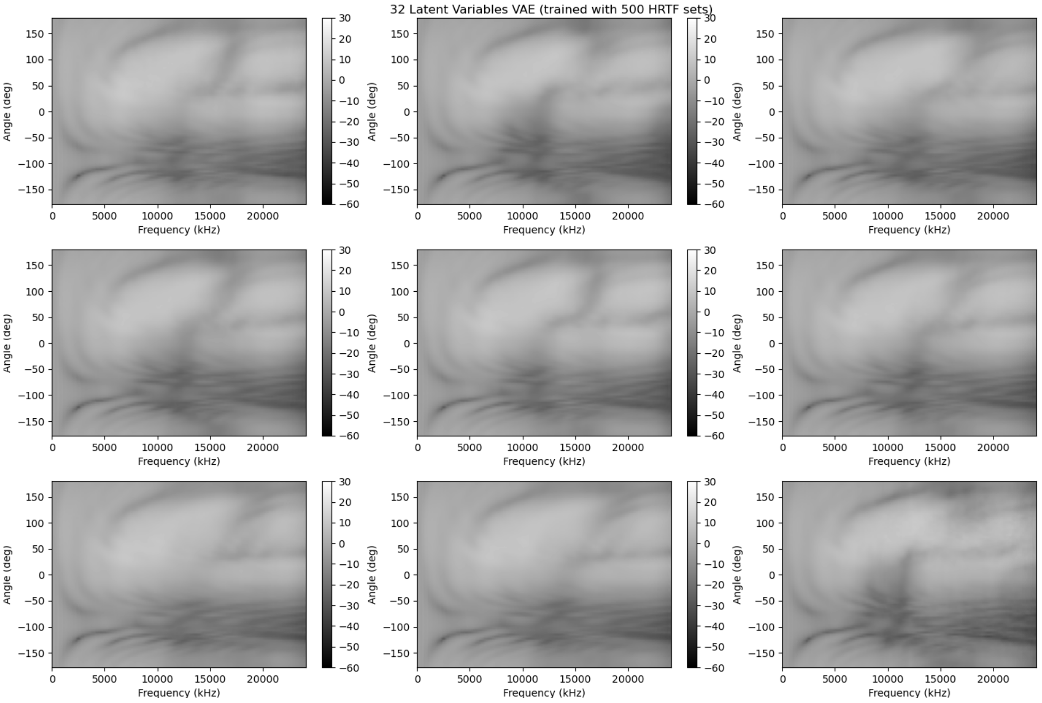


Figure A.23: 9 random generated HRTF sets from the PCA model with 32 PCs trained with 500 HRTF sets, left channel of the HRTFs on the horizontal plane

List of Acronyms

ML	Machine Learning
HRTF	Head Related Transfer Function
HRIR	Head Related Impulse Response
TRIR	Torso Related Impulse Response
PRIR	Pinna-Related Transfer Functions
3DMMs	Three-dimensional Morphable Models
HUMMNGBIRD	HUman Morphable Model-based Numerically Generated Binaural Impulse Response Database
PCA	Principle Component Analysis
PCs	Principle Components
VAE	Variational Auto-Encoder
VR	Virtual Reality
AR	Augmented Reality
XR	Extended Reality
ITD	Interaural Time Difference
ILD	Interaural Level Difference
NN	Neural Networks
CNN	Convolution neural network
AE	Auto-encoder
VAE	Variational Auto-encoder
CAE	Convolution Auto-encoder
ResNet	Residual Network
GAN	Generative Adversarial Networks
BEM	Boundary Element Method
3D	Three Dimensional
HMD	Head-Mounted Devices
FDTD	Finite-Difference Time-Domain
UWVF	Ultra-Weak Variational Formulation
MRI	Magnetic Resonance Imaging
CT	Computed Tomography
SOFA	Spatially Oriented Format for Acoustics
SH	Spherical Harmonic
SN3D	Schmidt Semi-Normalisation
TA	Time Alignment
SD	Spectral Distortion

NLP	Neutral Language Processing
SVM	Support vector machine
KL	Kullback-Leibler
BCE	Binary Cross Entropy
MSE	Mean Squared Error
SUpDEq	Spatial Upsampling by Directional Equalization
RAM	Random-access memory
PSD	Perceptual Spectral Difference
IQR	Interquartile Range

References

[1] B. Xie, *Head-related transfer function and virtual auditory display*, Second edi. J.Ross Publishing, 2013, p. 501, ISBN: 1604270705. [Online]. Available: https://books.google.co.uk/books/about/Head_Related_Transfer_Function_and_Virtu.html?id=LEaNmQEACAAJ{\&}redir_esc=y.

[2] D. M. D. M. Howard and J. Angus, *Acoustics and Psychoacoustics*, 4th. Focal Press, 2009, ISBN: 9788578110796. DOI: 10.1017/CBO9781107415324.004.

[3] P. Paukner, M. Rothbucher, and K. Diepold, "Sound Localization Performance Comparison of Different HRTF-Individualization Methods," Ph.D. dissertation, 2014. [Online]. Available: https://mediatum.ub.tum.de/doc/1207048/1207048.pdfhttp://mediatum.ub.tum.de/doc/1207048/1207048.pdf.

[4] D. Poirier-Quinot and B. F. Katz, "Impact of HRTF individualization on player performance in a VR shooter game I," in *Spatial Reproduction*, 2018, pp. 1–9, ISBN: 0780309405. DOI: https://doi.org/10.17743/aesconf.2018.978-1-942220-20-6. [Online]. Available: http://www.aes.org/publications/conferences/?confNum=ID-183.

[5] D. Poirier-Quinot and B. F. Katz, "Impact of HRTF individualization on player performance in a VR shooter game II," in *Audio for Virtual and Augmented Reality*, 2018, p. 8.

[6] M. Geronazzo, S. Spagnol, and F. Avanzini, "Do we need individual head-related transfer functions for vertical localization? the case study of a spectral notch distance metric," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 26, no. 7, pp. 1243–1256, 2018, ISSN: 23299290. DOI: 10.1109/TASLP.2018.2821846.

[7] G. D. Romigh and B. D. Simpson, "Do you hear where i hear?: Isolating the individualized sound localization cues," *Frontiers in Neuroscience*, vol. 8, no. OCT, pp. 1–8, 2014, ISSN: 1662453X. DOI: 10.3389/fnins.2014.00370.

[8] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 2242–2251, 2017, ISSN: 15505499. DOI: 10.1109/ICCV.2017.244.

[9] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 5967–5976, 2017. DOI: 10.1109/CVPR.2017.632.

[10] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, and H. Li, "High-resolution image inpainting using multi-scale neural patch synthesis," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 4076–4084, 2017. DOI: 10.1109/CVPR.2017.434.

[11] K. Nazeri, E. Ng, T. Joseph, F. Z. Qureshi, and M. Ebrahimi, "EdgeConnect: Generative Image Inpainting with Adversarial Edge Learning," 2019. [Online]. Available: http://arxiv.org/abs/1901.00212.

[12] Z. Yan, X. Li, M. Li, W. Zuo, and S. Shan, "Shift-net: Image inpainting via deep feature rearrangement," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11218 LNCS, pp. 3–19, 2018, ISSN: 16113349. DOI: 10.1007/978-3-030-01264-9{_}1.

[13] G. Liu, F. A. Reda, K. J. Shih, T. C. Wang, A. Tao, and B. Catanzaro, "Image Inpainting for Irregular Holes Using Partial Convolutions," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11215 LNCS, pp. 89–105, 2018, ISSN: 16113349. DOI: 10.1007/978-3-030-01252-6{_}6.

[14] J. Antic, *Jantic/deoldify: A deep learning based project for colorizing and restoring old images (and video!)* https://github.com/jantic/DeOldify, (Accessed on 10/29/2019), 2019.

[15] X.-J. Mao, C. Shen, and Y.-B. Yang, "Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections," in *30th Conference on Neural Information Processing Systems (NIPS 2016)*, vol. 9, Barcelona, Sep. 2016, p. 9. DOI: 10.1590/0102-311X00135812. [Online]. Available: http://www.scielo.br/scielo.php?script=sci_arttext{\&}pid=S0102-311X2013000900024{\&}lng=pt{\&}nrm=iso{\&}tlng=en.

[16] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual Dense Network for Image Restoration," vol. 13, no. 9, pp. 1–14, 2018. [Online]. Available: http://arxiv.org/abs/1812.10477.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December, pp. 770–778, 2016, ISSN: 10636919. DOI: 10.1109/CVPR.2016.90. arXiv: 1512.03385.

[18] I Goodfellow, J Pouget-Abadie, M. M. A. i. n. ..., and U. 2014, "Generative adversarial nets," *Papers.Nips.Cc*, pp. 1–9, 2014, ISSN: 10495258. DOI: 10.1017/CBO9781139058452. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.

[19] Acoustics Research Institute, *ARI HRTF Database*, (Accessed on 7/11/2019), 2014.

[20] R. Bomhardt, M. De La, F. Klein, and J. Fels, "A high-resolution head-related transfer function and three-dimensional ear model database A high-resolution head-related transfer function dataset and 3D ear model database," *Proc. Mtgs. Acoust. The Journal of the Acoustical Society of America*, vol. 29, no. 140, 2016. DOI: 10.1121/1.4970409. [Online]. Available: https://doi.org/10.1121/2.0000467http://asa.scitation.org/toc/pma/29/1.

[21] K. Watanabe, Y. Iwaya, Y. Suzuki, S. Takane, and S. Sato, "Dataset of head-related transfer functions measured with a circular loudspeaker array," *Acoustical Science and Technology*, vol. 35, no. 3, pp. 159–165, 2014, ISSN: 1346-3969. DOI: 10.1250/ast.35.159. [Online]. Available: https://www.jstage.jst.go.jp/article/ast/35/3/35_E1368/_pdf/-char/enhttps://www.jstage.jst.go.jp/article/ast/35/3/35_E1368/_article.

[22] *SADIE — Spatial Audio For Domestic Interactive Entertainment.* [Online]. Available: https://www.york.ac.uk/sadie-project/past_news_01.html#GoogleVR.

[23] C. Armstrong, A. Chadwick, L. Thresh, D. Murphy, and G. Kearney, "Simultaneous HRTF Measurement of Multiple Source Configurations Utilizing Semi-Permanent Structural Mounts," 2017. [Online]. Available: http://www.aes.org/tmpFiles/elib/20180319/19311.pdf.

[24] V. Algazi, R. Duda, D. Thompson, and C Avendano, "The CIPIC HRTF database," in *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No.01TH8575)*, 2001, pp. 99–102, ISBN: 0-7803-7126-7. DOI: 10.1109/ASPAA.2001.969552. [Online]. Available: http://interface.cipic.ucdavis.edu/pubs/WASSAP_2001_143.pdfhttp://ieeexplore.ieee.org/document/969552/.

[25] O. Warusfel, *Listen HRTF Database*, 2003. [Online]. Available: http://recherche.ircam.fr/equipes/salles/listen/index.htmlhttp://recherche.ircam.fr/equipes/salles/listen/index.html\%5Cnhttp://scholar.google.com/scholar?hl=en{\&}btnG=Search{\&}q=intitle:Listen+HRTF+Database#0.

[26] B. Bernschütz, "A Spherical Far Field HRIR/HRTF Compilation of the Neumann KU 100," *Fortschritte der Akustik – AIA-DAGA 2013*, pp. 592–595, 2013. [Online]. Available: http://www.audiogroup.web.fh-koeln.de/FILES/AIA-DAGA2013_HRIRs.pdf.

[27] C. Armstrong, L. Thresh, D. Murphy, and G. Kearney, "A perceptual evaluation of individual and non-individual HRTFs: A case study of the SADIE II database," *Applied Sciences (Switzerland)*, vol. 8, no. 11, 2018, ISSN: 20763417. DOI: 10.3390/app8112029.

[28] G. W. Lee and H. K. Kim, "Personalized hrtf modeling based on deep neural network using anthropometric measurements and images of the ear," *Applied Sciences*, vol. 8, no. 11, 2018, ISSN: 2076-3417. DOI: 10.3390/app8112180. [Online]. Available: https://www.mdpi.com/2076-3417/8/11/2180.

[29] B. F. Katz, "Boundary element method calculation of individual head-related transfer function. i. rigid model calculation," *The Journal of the Acoustical Society of America*, vol. 110, no. 5, pp. 2440–2448, 2001.

[30] K. Young, G. Kearney, and A. I. Tew, "Loudspeaker Positions with Sufficient Natural Channel Separation for Binaural Reproduction," in *Audio Engineering Society International Conference on Spatial Reproduction - Aesthetics and Science*, 2018. [Online]. Available: `http://www.aes.org/e-lib/browse.cfm?elib=19649`.

[31] K. Young, A. I. Tew, and G. Kearney, "Boundary element method modelling of KEMAR for binaural rendering: Mesh production and validation," *Interactive Audio Systems Symposium*, pp. 1–8, 2016. [Online]. Available: `http://www.york.ac.uk/sadie-project//IASS2016/IASS_Papers/IASS_2016_paper_4.pdf`.

[32] C. C. Berger, M. Gonzalez-Franco, A. Tajadura-Jiménez, D. Florencio, and Z. Zhang, "Generic HRTFs may be good enough in virtual reality. Improving source localization through cross-modal plasticity," *Frontiers in Neuroscience*, vol. 12, no. FEB, 2018, ISSN: 1662453X. DOI: `10.3389/fnins.2018.00021`.

[33] O. S. Rummukainen, T. Robotham, and E. A. Habets, "Head-related transfer functions for dynamic listeners in virtual reality," *Applied Sciences (Switzerland)*, vol. 11, no. 14, 2021, ISSN: 20763417. DOI: `10.3390/app11146646`.

[34] B. F. G. Katz, "Boundary element method calculation of individual head-related transfer function. I. Rigid model calculation," *The Journal of the Acoustical Society of America*, vol. 110, no. 5, pp. 2449–2455, 2001, ISSN: 0001-4966. DOI: `10.1121/1.1412441`. [Online]. Available: `http://asa.scitation.org/doi/10.1121/1.1412441`.

[35] B. F. G. Katz, "Boundary element method calculation of individual head-related transfer function. II. Impedance effects and comparisons to real measurements," *The Journal of the Acoustical Society of America*, vol. 110, no. 5, pp. 2449–2455, 2001, ISSN: 0001-4966. DOI: `10.1121/1.1412441`. [Online]. Available: `http://asa.scitation.org/doi/10.1121/1.1412441`.

[36] R. S. Woodworth, *Experimental psychology.* Oxford, England: Holt, 1938, p. 889.

[37] N. Aaronson and W. Hartmann, "Testing, correcting, and extending the woodworth model for interaural time difference," *The Journal of the Acoustical Society of America*, vol. 135, Jan. 2014. DOI: `10.1121/1.4861243`.

[38] J. Blauert, *Spatial Hearing: The Psychophysics of Human Sound Localization.* The MIT Press, Oct. 1996, ISBN: 9780262268684. DOI: `10.7551/mitpress/6391.001.0001`. [Online]. Available: `https://doi.org/10.7551/mitpress/6391.001.0001`.

[39] G. B. Henning, "Detectability of interaural delay in high frequency complex waveforms," *The Journal of the Acoustical Society of America*, vol. 55, no. 1, pp. 84–90, 1974. DOI: `10.1121/1.1928135`. eprint: `https://doi.org/10.1121/1.1928135`. [Online]. Available: `https://doi.org/10.1121/1.1928135`.

[40] A. Kulkarni, S. K. Isabelle, and H. S. Colburn, "Sensitivity of human subjects to head-related transfer-function phase spectra," *The Journal of the Acoustical Society of America*, vol. 105, no. 5, pp. 2821–2840, May 1999, ISSN: 0001-4966. DOI: 10.1121/1.426898. eprint: https://pubs.aip.org/asa/jasa/article-pdf/105/5/2821/8085335/2821_1_online.pdf. [Online]. Available: https://doi.org/10.1121/1.426898.

[41] M. Otani, T. Hirahara, and D. Morikawa, "Origin of frequency dependence of interaural time difference," *Acoustical Science and Technology*, vol. 42, no. 4, pp. 181–192, 2021. DOI: 10.1250/ast.42.181.

[42] V. Benichoux, M. Rébillat, and R. Brette, "On the variation of interaural time differences with frequency," *The Journal of the Acoustical Society of America*, vol. 139, no. 4, pp. 1810–1821, Apr. 2016, ISSN: 0001-4966. DOI: 10.1121/1.4944638. eprint: https://pubs.aip.org/asa/jasa/article-pdf/139/4/1810/15315932/1810_1_online.pdf. [Online]. Available: https://doi.org/10.1121/1.4944638.

[43] F. L. Wightman and D. J. Kistler, "Resolution of front back ambiguity in spatial hearing by listener and source movement," *The Journal of the Acoustical Society of America*, vol. 105, no. 5, pp. 2841–2853, 1999. DOI: 10.1121/1.426899. eprint: https://doi.org/10.1121/1.426899. [Online]. Available: https://doi.org/10.1121/1.426899.

[44] H. Wallach, "On sound localization," *The Journal of the Acoustical Society of America*, vol. 10, no. 4, pp. 270–274, 1939. DOI: 10.1121/1.1915985. eprint: https://doi.org/10.1121/1.1915985. [Online]. Available: https://doi.org/10.1121/1.1915985.

[45] d. r. begault, a. s. lee, e. m. wenzel, and m. r. anderson, "Direct comparison of the impact of head tracking, reverberation, and individualized head-related transfer functions on the spatial perception of a virtual speech source," *journal of the audio engineering society*, 2000.

[46] G. D. Romigh, D. S. Brungart, and B. D. Simpson, "Free-field localization performance with a head-tracked virtual auditory display," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 5, pp. 943–954, 2015. DOI: 10.1109/JSTSP.2015.2421874.

[47] S. Perrett and W. Noble, "The effect of head rotations on vertical plane sound localization," *The Journal of the Acoustical Society of America*, vol. 102, no. 4, pp. 2325–2332, 1997. DOI: 10.1121/1.419642. eprint: https://doi.org/10.1121/1.419642. [Online]. Available: https://doi.org/10.1121/1.419642.

[48] D. Rao and B. Xie, "Head rotation and sound image localization in the median plane," *Chinese Science Bulletin*, vol. 50, no. 5, pp. 412–416, 2005, ISSN: 1861-9541. DOI: 10.1007/BF02897454. [Online]. Available: https://doi.org/10.1007/BF02897454.

[49] D. W. Batteau and H. E. Huxley, "The role of the pinna in human localization," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 168, no. 1011, pp. 158–180, 1967. DOI: 10.1098/rspb.1967.0058. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rspb.1967.0058. [Online]. Available: https://royalsocietypublishing.org/doi/abs/10.1098/rspb.1967.0058.

[50] E. A. LopezâPoveda and R. Meddis, "A physical model of sound diffraction and reflections in the human concha," *The Journal of the Acoustical Society of America*, vol. 100, no. 5, pp. 3248–3259, 1996. DOI: 10.1121/1.417208. eprint: https://doi.org/10.1121/1.417208. [Online]. Available: https://doi.org/10.1121/1.417208.

[51] A. Andreopoulou and A. Roginska, "Evaluating HRTF Similarity through Subjective Assessments : Factors that can Affect Judgment," *Joined 40th International Computer Music Conference (ICMC) & 11th Sound and Music Computing Conference*, no. September, pp. 1375–1381, 2014. [Online]. Available: http://www.bili-project.org/wp-content/uploads/2015/03/HRTF-similarity-3.pdf.

[52] L. S. R. Simon, N. Zacharov, and B. F. G. Katz, "Perceptual attributes for the comparison of head-related transfer functions," *JASA*, vol. 140, no. 6, pp. 3623–3632, 2016, ISSN: 0001-4966. DOI: 10.1121/1.4972301. [Online]. Available: http://dx.doi.org/10.1121/1.4966115http://asa.scitation.org/toc/jas/140/5http://dx.doi.org/10.1121/1.4971424.

[53] P. M. Hofman, J. G. A. V. Riswick, and A. J. V. Opstal, "Relearning sound localization with new ears," *Nature America*, vol. 1, no. 5, pp. 417–421, 1998.

[54] B. Zhou, D. M. Green, and J. C. Middlebrooks, "Characterization of external ear impulse responses using Golay codes," *J Acoust Soc Am*, vol. 92, no. 2 Pt 1, pp. 1169–1171, 1992, ISSN: 0001-4966. DOI: 10.1121/1.404045. [Online]. Available: http://dx.doi.org/10.1121/1.404045http://asa.scitation.org/toc/jas/92/2.

[55] D. N. Zotkin, R. Duraiswami, and L. S. Davis, "Rendering localized spatial audio in a virtual auditory space," *IEEE Transactions on Multimedia*, vol. 6, no. 4, pp. 553–564, 2004, ISSN: 15209210. DOI: 10.1109/TMM.2004.827516.

[56] D. Zotkin, J. Hwang, R. Duraiswaini, and L. S. Davis, "HRTF Personalization Using Anthropometric Measurements," *IEEE Work. Appl. Signal Process. to Audio Acoust.*, pp. 157–160, 2003. [Online]. Available: http://www.umiacs.umd.edu/users/dz/pbpslist/waspaa03_dz_final_v2.pdf.

[57] S.-N. Yao, "Rendering ambisonics over headphones," Ph.D. dissertation, University of Birmingham, 2015, pp. 168–216.

[58] P. Stitt and B. F. G. Katz, "Sensitivity analysis of pinna morphology on head-related transfer functions simulated via a parametric pinna model," *The Journal of the Acoustical Society of America*, vol. 149, no. 4, pp. 2559–2572, 2021, ISSN: 0001-4966. DOI: 10.1121/10.0004128. [Online]. Available: https://asa.scitation.org/doi/10.1121/10.0004128.

[59] M. Zhang, Z. Ge, T. Liu, X. Wu, and T. Qu, “Modeling of Individual HRTFs based on Spatial Principal Component Analysis,” *arXiv*, vol. 28, pp. 785–797, 2019, ISSN: 23318422.

[60] M. Zhang, Z. Ge, T. Liu, X. Wu, and T. Qu, “Modeling of Individual HRTFs Based on Spatial Principal Component Analysis,” *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 28, no. 1, pp. 785–797, 2020, ISSN: 23299304. DOI: 10.1109/TASLP.2020.2967539. arXiv: 1910.09484.

[61] R. Miccini and S. Spagnol, “A hybrid approach to structural modeling of individualized HRTFs,” *Proceedings - 2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops, VRW 2021*, pp. 80–85, 2021. DOI: 10.1109/VRW52623.2021.00022.

[62] Y. Luo, D. N. Zotkin, and R. Duraiswami, “Statistical analysis of head related transfer function (HRTF) data,” in *ICA 2013 Montreal*, vol. 19, 2013, p. 50 011. DOI: 10.1121/1.4799872. [Online]. Available: http://dx.doi.org/10.1121/1.4799872http://asa.scitation.org/toc/pma/19/1http://link.aip.org/link/?PMA/19/050011/1.

[63] T.-Y. Chen, T.-H. Kuo, and T.-S. Chi, “Autoencoding HRTFS for DNN Based HRTF Personalization Using Anthropometric Features,” in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, pp. 271–275, ISBN: 978-1-4799-8131-1. DOI: 10.1109/ICASSP.2019.8683814. [Online]. Available: https://ieeexplore.ieee.org/document/8683814/.

[64] G. W. Lee and H. K. Kim, “Personalized HRTF modeling based on deep neural network using anthropometric measurements and images of the ear,” *Applied Sciences (Switzerland)*, vol. 8, no. 11, 2018, ISSN: 20763417. DOI: 10.3390/app8112180.

[65] H. Hu, L. Zhou, H. Ma, and Z. Wu, “HRTF personalization based on artificial neural network in individual virtual auditory space,” *Applied Acoustics*, vol. 69, no. 2, pp. 163–172, 2008, ISSN: 0003682X. DOI: 10.1016/j.apacoust.2007.05.007. arXiv: 9809069v1 [gr-qc]. [Online]. Available: http://ac.els-cdn.com/S0003682X07000965/1-s2.0-S0003682X07000965-main.pdf?_tid=a6eb40e8-5a6f-11e7-8a26-00000aacb35d{\&}acdnat=1498482305_7efb1863344a21bdcb9ab75e78259b4a.

[66] E. A. Torres-Gallegos, F. Ordu?a-Bustamante, and F. Ar?mbula-Cos?o, “Personalization of head-related transfer functions (HRTF) based on automatic photo-anthropometry and inference from a database,” *Applied Acoustics*, vol. 97, pp. 84–95, 2015, ISSN: 1872910X. DOI: 10.1016/j.apacoust.2015.04.009.

[67] V. R. Algazi, R. O. Duda, R. Duraiswami, N. A. Gumerov, and Z. Tang, “Approximating the head-related transfer function using simple geometric models of the head and torso,” *The Journal of the Acoustical Society of America*, vol. 112, no. 5, pp. 2053–2064, 2002. DOI: 10.1121/1.1508780. eprint: https://doi.org/10.1121/1.1508780. [Online]. Available: https://doi.org/10.1121/1.1508780.

[68] F. Brinkmann, R. Roden, A. Lindau, and S. Weinzierl, "Audibility and interpolation of head-above-torso orientation in binaural technology," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 5, pp. 931–942, 2015. DOI: 10.1109/JSTSP.2015.2414905.

[69] W.-S. Gan, S. Peksi, J. He, R. Ranjan, N. D. Hai, and N. K. Chaudhary, "Personalized HRTF measurement and 3D Audio Rendering for AR/VR Headsets," in *Personalized HRTF measurement and 3D Audio Rendering for AR/VR Headsets*, 2017. [Online]. Available: http://www.aes.org/tmpFiles/elib/20170528/18706.pdf.

[70] V. R. Algazi, R. O. Duda, R. Duraiswami, N. A. Gumerov, and Z. Tang, "Approximating the head-related transfer function using simple geometric models of the head and torso," *The Journal of the Acoustical Society of America*, vol. 112, no. 5, pp. 2053–2064, 2002. DOI: 10.1121/1.1508780. eprint: https://doi.org/10.1121/1.1508780. [Online]. Available: https://doi.org/10.1121/1.1508780.

[71] r. o. duda, v. r. algazi, and d. m. thompson, "The use of head-and-torso models for improved spatial sound synthesis," *journal of the audio engineering society*, 2002.

[72] N. A. Gumerov, R. Duraiswami, and Z. Tang, "Numerical study of the influence of the torso on the hrtf," in *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, 2002, pp. II–1965–II–1968. DOI: 10.1109/ICASSP.2002.5745015.

[73] M. Guldenschuh, A. Sontacchi, F. Zotter, and R. HÃ¶ldrich, "Hrtf modeling in due consideration variable torso reflections," *The Journal of the Acoustical Society of America*, vol. 123, no. 5, pp. 3080–3080, 2008. DOI: 10.1121/1.2932888. eprint: https://doi.org/10.1121/1.2932888. [Online]. Available: https://doi.org/10.1121/1.2932888.

[74] A. Sontacchi and F. Zotter, "Hrtf modelling in due consideration variable torso reflections," Jun. 2008.

[75] A. Vanne, D. Satongar, J. Vandyke, J. Merimaa, M. Johnson, and T. Huttunen, *Spatial audio reproduction based on head-to-torso orientation*, 2022.

[76] M. Lovedee-Turner and D. Murphy, "Application of Machine Learning for the Spatial Analysis of Binaural Room Impulse Responses," *Applied Sciences*, vol. 8, no. 1, p. 105, 2018, ISSN: 2076-3417. DOI: 10.3390/app8010105. [Online]. Available: http://www.mdpi.com/2076-3417/8/1/105.

[77] Y. Ma *et al.*, "EasySVM: A visual analysis approach for open-box support vector machines," *Computational Visual Media*, vol. 3, no. 2, pp. 161–175, 2017, ISSN: 20960662. DOI: 10.1007/s41095-017-0077-5.

[78] J. Ding, J. Wang, C. Zheng, R. Peng, and X. Li, "Analysis of Binaural Features for Supervised Localization in Reverberant Environments," in *Proc. of Audio Engineering Society Convention 141*, 2016, pp. 1–9. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=18446.

[79] K. Diepold, M Durkovic, and F. Sagstetter, "HRTF Measurements with Recorded Reference Signal," in *Proc. of Audio Engineering Society Conference 129*, 2010, pp. 1–8, ISBN: 9781617821943. [Online]. Available: http://www.aes.org/tmpFiles/elib/20170514/15692.pdfhttp://www.aes.org/e-lib/browse.cfm?conv=129{\&}papernum=8270.

[80] C. Rascon and I. Meza, "Localization of sound sources in robotics : A review," *Robotics and Autonomous Systems*, vol. 96, pp. 184–210, 2017, ISSN: 0921-8890. DOI: 10.1016/j.robot.2017.07.011. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2017.07.011.

[81] D. N. Zotkin, R. Duraiswami, E. Grassi, and N. A. Gumerov, "Fast head-related transfer function measurement via reciprocity.," *The Journal of the Acoustical Society of America*, vol. 120, no. 4, pp. 2202–2215, 2006, ISSN: 00014966. DOI: 10.1121/1.2207578. [Online]. Available: http://www.umiacs.umd.edu/~dz/pbpslist/jasa2006dz.pdf.

[82] J. Meyer, M. Smirnov, A. Khajeh-saeed, J. Meyer, and S. T. Prepelit, "Finite-difference time-domain simulations : Verification on head-related transfer functions of a rigid sphere model Finite-difference time-domain simulations : Verification on head-related transfer functions of a rigid sphere model," vol. 062401, no. March, 2022. DOI: 10.1121/10.0011736.

[83] T. Xiao and Q. Huo Liu, "Finite difference computation of head-related transfer function for human hearing," *The Journal of the Acoustical Society of America*, vol. 113, no. 5, pp. 2434–2441, 2003. DOI: 10.1121/1.1561495. eprint: https://doi.org/10.1121/1.1561495. [Online]. Available: https://doi.org/10.1121/1.1561495.

[84] t. huttunen, a. kärkkäinen, l. kärkkäinen, o. kirkeby, and e. t. seppälä, "Some effects of the torso on head-related transfer functions," *journal of the audio engineering society*, 2007.

[85] T. HUTTUNEN, E. T. SEPPÄLÄ, O. KIRKEBY, A. KÄRKKÄINEN, and L. KÄRKKÄINEN, "Simulation of the transfer function for a head-and-torso model over the entire audible frequency range," *Journal of Computational Acoustics*, vol. 15, no. 04, pp. 429–448, 2007. DOI: 10.1142/S0218396X07003469. eprint: https://doi.org/10.1142/S0218396X07003469. [Online]. Available: https://doi.org/10.1142/S0218396X07003469.

[86] H. Ziegelwanger, P. Majdak, and W. Kreuzer, "Numerical calculation of listener-specific head-related transfer functions and sound localization: Microphone model and mesh discretization," *Journal of the Acoustical Society of America*, vol. 138, no. 1, pp. 208–222, 2015, ISSN: 0001-4966. DOI: 10.1121/1.4922518. [Online]. Available: http://dx.doi.org/10.1121/1.4922518http://asa.scitation.org/toc/jas/138/1.

[87] H. Ziegelwanger, W. Kreuzer, and P. Majdak, "Mesh2hrtf: An open-source software package for the numerical calculation of head-related transfer functions," in *The 22nd International Congress on Sound and Vibration*, Jul. 2015, pp. 1–8. DOI: 10.13140/RG.2.1.1707.1128.

[88] M. D. Burkhard and R. M. Sachs, "Anthropometric manikin for acoustic research," *The Journal of the Acoustical Society of America*, vol. 58, no. 1, pp. 214–222, 1975. DOI: 10.1121/1.380648. eprint: https://doi.org/10.1121/1.380648. [Online]. Available: https://doi.org/10.1121/1.380648.

[89] T. Tew, C. Hetherington, and J. Thorpe, "Morphoacoustic perturbation analysis: Principles and validation," Apr. 2012.

[90] C. T. Jin *et al.*, "Creating the sydney york morphological and acoustic recordings of ears database," *IEEE Transactions on Multimedia*, vol. 16, no. 1, pp. 37–46, 2014, ISSN: 15209210. DOI: 10.1109/TMM.2013.2282134.

[91] S. Ghorbal, X. Bonjour, and R. Séguier, "Computed Hrirs and Ears Database for Acoustic Research," in *148th Audio Engineering Society International Convention*, 2020, pp. 1–7.

[92] C. Guezenoc and R. Seguier, "A Wide Dataset of Ear Shapes and Pinna-Related Transfer Functions Generated by Random Ear Drawings," 2020. arXiv: 2003.06182. [Online]. Available: http://arxiv.org/abs/2003.06182.

[93] K. Pollack, P. Majdak, and H. Furtades, "A Landmark-Based Parametric Pinna Model For The Calculation of Head-Related Transfer Functions," in *Forum Acusticum*, Lyon, France, Dec. 2020, pp. 1357–1360. DOI: 10.48465/fa.2020.0280. [Online]. Available: https://hal.science/hal-03235345.

[94] K. Pollack and P. Majdak, "Evaluation of a parametric pinna model for the calculation of head-related transfer functions," in *2021 Immersive and 3D Audio: from Architecture to Automotive (I3DA)*, 2021, pp. 1–5. DOI: 10.1109/I3DA48870.2021.9610885.

[95] P. Majdak *et al.*, "Spatially oriented format for acoustics: A data exchange format representing head-related transfer functions," *134th Audio Engineering Society Convention 2013*, pp. 262–272, 2013. [Online]. Available: http://www.aes.org/tmpFiles/elib/20170514/16781.pdfhttp://www.scopus.com/inward/record.url?eid=2-s2.0-84883350105{\&}partnerID=tZOtx3y1.

[96] Audio Engineering Society Inc., *AES standard for file exchange - Spatial acoustic data file format*, 2015. DOI: 10.1186/BF03356041. [Online]. Available: http://www.aes.org/standards.

[97] A. Andreopoulou and A. Roginska, "Towards the Creation of a Standardized HRTF Repository," 2011. [Online]. Available: http://www.aes.org/tmpFiles/elib/20180221/16096.pdf.

[98] *General information on SOFA*, 2013. [Online]. Available: https://www.sofaconventions.org/mediawiki/index.php/General_information_on_SOFA.

[99] *SOFA - Spatially Oriented Format for Acoustics*, 2015. [Online]. Available: https://github.com/sofacoustics/API_MO.

[100] F. Brinkmann, M. Dinakaran, R. Pelzer, P. Grosche, D. Voss, and S. Weinzierl, "A cross-evaluated database of measured and simulated HRTFs including 3D head meshes, anthropometric features, and headphone impulse responses," *J. Audio Eng. Soc.*, vol. 67, no. 1, pp. 1–16, 2019.

[101] H. Gamper, "Head-related transfer function interpolation in azimuth, elevation, and distance," *The Journal of the Acoustical Society of America*, vol. 134, no. 6, EL547–EL553, 2013, ISSN: 0001-4966. DOI: 10.1121/1.4828983. [Online]. Available: http://asa.scitation.org/doi/10.1121/1.4828983.

[102] F. Grijalva, L. C. Martini, D. Florencio, and S. Goldenstein, "Interpolation of Head-Related Transfer Functions Using Manifold Learning," *IEEE Signal Processing Letters*, vol. 24, no. 2, pp. 221–225, 2017, ISSN: 10709908. DOI: 10.1109/LSP.2017.2648794.

[103] K. Hartung, J. Braasch, and S. J. Sterbing, "Comparison of different methods for the interpolation of head-related transfer functions," in *AES 16th International Conference: Spatial Sound Reproduction*, 1999, pp. 319–329. [Online]. Available: http://www.aes.org/tmpFiles/elib/20170805/8026.pdfhttp://www.aes.org/e-lib/browse.cfm?elib=8026.

[104] R. L. Martin and K. McAnally, "Interpolation of Head-Related Transfer Functions," *Air Operations Division Defence Science and Technology Organisation*, 2007.

[105] M. J. Evans, J. A. S. Angus, and A. I. Tew, "Analyzing head-related transfer function measurements using surface spherical harmonics A model of head-related transfer functions based on principal components analysis and minimum-phase reconstruction Sensitivity of human subjects to head-related transf," *The Journal of the Acoustical Society of America The Journal of the Acoustical Society of America The Journal of the Acoustical Society of America*, vol. 104, no. 43, pp. 2400–1637, 1998. DOI: 10.1121/1.3336399. [Online]. Available: https://doi.org/10.1121/1.423749http://asa.scitation.org/toc/jas/104/4.

[106] F. Zotter and M. Frank, *Ambisonics* (Springer Topics in Signal Processing), 1st ed. Cham: Springer International Publishing, 2019, vol. 19, pp. XIV, 210, ISBN: 978-3-030-17206-0. DOI: 10.1007/978-3-030-17207-7. [Online]. Available: https://www.springer.com/gp/book/9783030172060http://link.springer.com/10.1007/978-3-030-17207-7.

[107] A. R. Institute, *ARI HRTF Database*, 2014. [Online]. Available: https://www.kfs.oeaw.ac.at/index.php?option=com_content{\&}view=article{\&}id=608{\&}Itemid=606{\&}lang=en#AnthropometricData.

[108] F. Brinkmann, M. Dinakaran, R. Pelzer, P. Grosche, D. Voss, and S. Weinzierl, "A cross-evaluated database of measured and simulated hrtfs including 3d head meshes, anthropometric features, and headphone impulse responses," *J. Audio Eng. Soc*, vol. 67, no. 9, pp. 705–718, 2019. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=20546.

[109] H. S. Braren and J. Fels, "Towards child-appropriate virtual acoustic environments: A database of high-resolution hrtf measurements and 3d-scans of children," *International Journal of Environmental Research and Public Health*, vol. 19, no. 1, 2022, ISSN: 16604601. DOI: 10.3390/ijerph19010324.

[110] M. Chapman *et al.*, "A standard for interchange of ambisonic signal sets including a file standard with metadata," in *AMBISONICS SYMPOSIUM 2009*, Jan. 2009, pp. 25–27.

[111] J. Daniel, "Représentation de champs acoustiques, application à la transmission et à la reproduction de scènes sonores complexes dans un contexte multimédia," Ph.D. dissertation, University of Paris VI, Jan. 2000.

[112] S. Bertet, D. Jérôme, and M. Sébastien, "3D Sound Field Recording with Higher Order Ambisonics – Objective Measurements and Validation of Spherical Microphone," *AES 120th Convention*, pp. 1–24, 2006. [Online]. Available: http://rndnet.starkey.com/sites/hrtjournal/pages/DSPJournalSeminar.aspx.

[113] M. Zaunschirm, C. Schörkhuber, and R. Höldrich, "Binaural rendering of Ambisonic signals by head-related impulse response time alignment and a diffuseness constraint," *The Journal of the Acoustical Society of America*, vol. 143, no. 6, pp. 3616–3627, 2018, ISSN: 0001-4966. DOI: 10.1121/1.5040489.

[114] T. Mckenzie, D. T. Murphy, and G. Kearney, "An Evaluation of Pre-Processing Techniques for Virtual Loudspeaker Binaural Ambisonic Rendering," in *EAA Spatial Audio Signal Processing symposium*, 2019, pp. 149–154. DOI: 10.25836/sasp.2019.09.

[115] E. J. Selfridge, Rod; Reiss, Joshua D;Avital, "HRTF Individualization: A Survey," in *Audio Engineering Society*, 2018, pp. 1–13. DOI: 10.1111/j.1365-2966.2010.17474.x. arXiv: 1003.2417.

[116] K. Mcmullen, A. Roginska, and G. Wakefield, "Subjective Selection of Head-Related Transfer Functions (HRTFs) based on Spectral Coloration and Interaural Time Differences (ITD) Cues," in *133rd AES Convention*, 2012, pp. 1–9, ISBN: 9781622766031. [Online]. Available: http://www.kylamcmullen.com/Kyla_McMullen/Research_files/paper133_kmedit.pdf.

[117] Y. Iwaya, "Individualization of head-related transfer functions with tournament-style listening test: Listening with other's ears," *Acoustical Science and Technology*, vol. 27, no. 6, pp. 340–343, 2006, ISSN: 1346-3969. DOI: 10.1250/ast.27.340. [Online]. Available: http://www.asj.gr.jp/2006/data/ast2706.html..

[118] A. Andreopoulou and B. F. G. Katz, "Subjective HRTF evaluations for obtaining global similarity metrics of assessors and assessees," *Journal on Multimodal User Interfaces*, vol. 10, no. 3, pp. 259–271, 2016, ISSN: 17838738. DOI: 10.1007/s12193-016-0214-y.

[119] A. Andreopoulou and B. F. Katz, "Investigation on Subjective HRTF Rating Repeatability," in *140th Audio Engineering Society Convention*, 2016, pp. 1–10. [Online]. Available: `http://www.aes.org/tmpFiles/elib/20170514/18295.pdfhttp://www.aes.org/e-lib/browse.cfm?elib=18295`.

[120] D. Poirier-quinot and B. F. G. Katz, "The Anaglyph binaural audio engine," in *AES 144th Convention Paper*, Milan, 2018, pp. 2–5.

[121] D. Zotkin, J. Hwang, R. Duraiswaini, and L. S. Davis, "HRTF Personalization Using Anthropometric Measurements," *IEEE Work. Appl. Signal Process. to Audio Acoust.*, pp. 157–160, 2003. [Online]. Available: `http://www.umiacs.umd.edu/users/dz/pbpslist/waspaa03{\_}dz{\_}final{\_}v2.pdf`.

[122] K. Iida, Y. Ishii, and S. Nishioka, "Personalization of head-related transfer functions in the median plane based on the anthropometry of the listener's pinnae," *The Journal of the Acoustical Society of America*, vol. 136, no. 1, pp. 317–33, 2014, ISSN: 1520-8524. DOI: `10.1121/1.4880856`. [Online]. Available: `http://dx.doi.org/10.1121/1.4880856http://asa.scitation.org/toc/jas/136/1http://dx.doi.org/10.1121/1.4880856\%5Cnhttp://www.ncbi.nlm.nih.gov/pubmed/24993216`.

[123] L. Li and Q. Huang, "HRTF personalization modeling based on RBF neural network," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3707–3710, 2013, ISSN: 15206149. DOI: `10.1109/ICASSP.2013.6638350`. [Online]. Available: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6638350`.

[124] S.-N. Yao, T. Collins, and C. Liang, "Head-Related Transfer Function Selection Using Neural Networks," *Archives of Acoustics*, vol. 42, no. 3, pp. 365–373, 2017, ISSN: 2300-262X. DOI: `10.1515/aoa-2017-0038`. [Online]. Available: `http://www.degruyter.com/view/j/aoa.2017.42.issue-3/aoa-2017-0038/aoa-2017-0038.xml`.

[125] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 770–778, 2016, ISSN: 10636919. DOI: `10.1109/CVPR.2016.90`.

[126] F. Shahid, N. Javeri, K. Jain, and S. Badhwar, "AI DevOps for large-scale HRTF prediction and evaluation: an end to end pipeline," in *Audio for Virtual and Augmented Reality*, 2018, p. 8.

[127] S. Kaneko, T. Suenaga, and S. Sekine, "DeepEarNet: individualizing spatial audio with photography, ear shape modeling, and neural networks," in *Audio for Virtual and Augmented Reality*, 2016. [Online]. Available: `http://www.aes.org/tmpFiles/elib/20171210/18509.pdf`.

[128] B. F. G. Katz, "Boundary element method calculation of individual head-related transfer function. ii. impedance effects and comparisons to real measurements," *The Journal of the Acoustical Society of America*, vol. 110, no. 5, pp. 2449–2455, 2001. DOI: `10.1121/1.1412441`. eprint: `https://doi.org/10.1121/1.1412441`. [Online]. Available: `https://doi.org/10.1121/1.1412441`.

[129] L. Bonacina, A. Canalini, F. Antonacci, M. Marcon, A. Sarti, and S. Tubaro, "A low-cost solution to 3D pinna modeling for HRTF prediction," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2016-May, pp. 301–305, 2016, ISSN: 15206149. DOI: 10.1109/ICASSP.2016.7471685.

[130] G. W. Lee, J. M. Moon, C. J. Chun, and H. K. Kim, "On the Use of Bottleneck Features of CNN Auto-encoder for Personalized HRTFs," in *AES 144th Convention Paper*, Milan, 2018, pp. 1–13. DOI: 10.13140/2.1.1598.6882.

[131] T. Huttunen, A. Karjalainen, and A. Vanne, "Personal HRTFs in a VR environment," in *AES 144th Convention Paper*, Milan, 2018, pp. 2–5.

[132] C. Jenny, P. Majdak, and C. Reuter, "SOFA Native Spatializer Plugin for Unity – Exchangeable HRTFs in Virtual Reality," in *AES 144th Convention Paper*, Milan, 2018, pp. 2–5.

[133] E. Mauskopf, *Google developers blog: Open sourcing resonance audio*, https://developers.googleblog.com/2018/03/resonance-audio-goes-open-source.html, (Accessed on 10/29/2018), Mar. 2018.

[134] *Steam audio*, https://valvesoftware.github.io/steam-audio/, (Accessed on 10/29/2018).

[135] *Features*, https://developer.oculus.com/documentation/audiosdk/latest/concepts/audiosdk-features/, (Accessed on 10/29/2018).

[136] M. Geronazzo, J. Kleimola, E. Sikstroöm, A. d. Götzen, S. Serafin, and F. Avanzini, "HOBA-VR: HRTF On Demand for Binaural Audio in immersive virtual reality environments," in *AES 144th Convention Paper*, Milan, 2018, pp. 2–5.

[137] t. rudzki, d. murphy, and g. kearney, "Xr-based hrtf measurements," *journal of the audio engineering society*, 2022.

[138] J. C. Middlebrooks and D. M. Green, "Observations on a principal components analysis of head-related transfer functions," *The Journal of the Acoustical Society of America*, vol. 92, pp. 597–599, July 1992 2014.

[139] P. Mokhtari *et al.*, "Further observations on a principal components analysis of head-related transfer functions," *Scientific Reports*, vol. 9, pp. 1–7, 1 2019, ISSN: 20452322. DOI: 10.1038/s41598-019-43967-0. [Online]. Available: http://dx.doi.org/10.1038/s41598-019-43967-0.

[140] D. J. Kistler and F. L. Wightman, "A model of head-related transfer functions based on principal components analysis and minimum-phase reconstruction," *Journal of the Acoustical Society of America*, vol. 91, pp. 1637–1647, 3 1992, ISSN: NA. DOI: 10.1121/1.402444.

[141] S. Qing and M. Hua, "Individualized HRTF matching and compression for virtual," pp. 1–8, 1984.

[142] K. J. Fink and L. Ray, "Tuning principal component weights to individualize HRTFs," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 389–392, ISBN: 9781467300469. DOI: 10.1109/ICASSP.2012.6287898. [Online]. Available: http://www.mirlab.org/conference_papers/International_Conference/ICASSP2012/pdfs/0000389.pdf.

[143] K. J. Fink and L. Ray, "Individualization of head related transfer functions using principal component analysis," *Applied Acoustics*, vol. 87, pp. 162–173, 2015, ISSN: 1872910X. DOI: 10.1016/j.apacoust.2014.07.005. [Online]. Available: http://ac.els-cdn.com/S0003682X14001753.

[144] L. Picinali and B. F. G. Katz, "System-to-user and user-to-system adaptations in binaural audio," in *Sonic Interactions in Virtual Environments*, M. Geronazzo and S. Serafin, Eds. Cham: Springer International Publishing, 2023, pp. 115–143, ISBN: 978-3-031-04021-4. DOI: 10.1007/978-3-031-04021-4_4. [Online]. Available: https://doi.org/10.1007/978-3-031-04021-4_4.

[145] A. Roginska, G. H. Wakefield, and T. S. Santoro, "Stimulus-dependent HRTF preference," 2010. [Online]. Available: http://www.aes.org/tmpFiles/elib/20180107/15690.pdf.

[146] D. Silver *et al.*, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017, ISSN: 14764687. DOI: 10.1038/nature24270. [Online]. Available: http://dx.doi.org/10.1038/nature24270.

[147] V. Mnih, D. Silver, M. Riedmiller, A. Graves, I. Antonoglou, and D. Wierstra, "Playing Atari with Deep Reinforcement Learning Volodymyr," pp. 1–9, 2013, ISSN: 0028-0836. DOI: 10.1038/nature14236.

[148] B. Tan, N. Xu, and B. Kong, "Autonomous Driving in Reality with Reinforcement Learning and Image Translation," 2018. [Online]. Available: http://arxiv.org/abs/1801.05299.

[149] X. Pan, Y. You, Z. Wang, and C. Lu, "Virtual to Real Reinforcement Learning for Autonomous Driving," 2017. [Online]. Available: http://arxiv.org/abs/1704.03952.

[150] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3389–3396, 2017. [Online]. Available: http://arxiv.org/abs/1610.00633.

[151] P. Kormushev, S. Calinon, and D. G. Caldwell, "Robot motor skill coordination with EM-based reinforcement learning," *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, pp. 3232–3237, 2010, ISSN: 2153-0858. DOI: 10.1109/IROS.2010.5649089.

[152] A. Umam, *Highordercomp.jpg (705×585)*, https://ardianumam.files.wordpress.com/2017/09/highordercomp.jpg, (Accessed on 10/29/2018), Sep. 2017.

[153] S. Sayad, *Logistic regression*, https://www.saedsayad.com/logistic_regression.htm, (Accessed on 10/29/2018).

[154] R. Jain, *Simple tutorial on svm and parameter tuning in python and r — hackerearth blog*, https://www.hackerearth.com/blog/machine-learning/simple-tutorial-svm-parameter-tuning-python-r/, (Accessed on 10/29/2018), Feb. 2017.

[155] *Understanding k-means clustering with examples — edureka*, https://www.edureka.co/blog/k-means-clustering/, (Accessed on 01/02/2023).

[156] V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan, "Finding a "kneedle" in a haystack: Detecting knee points in system behavior," in *2011 31st International Conference on Distributed Computing Systems Workshops*, 2011, pp. 166–171. DOI: 10.1109/ICDCSW.2011.20.

[157] *Knee/elbow point detection — kaggle*, https://www.kaggle.com/code/kevinarvai/knee-elbow-point-detection, (Accessed on 01/02/2023).

[158] D. Bokde, S. Girase, and D. Mukhopadhyay, "Matrix Factorization model in Collaborative Filtering algorithms: A survey," *Procedia Computer Science*, vol. 49, no. 1, pp. 136–146, 2015, ISSN: 18770509. DOI: 10.1016/j.procs.2015.04.237.

[159] J. B. Schafer, D Frankowski, J Herlocker, and S Sen, "Collaborative Filtering Recommender Systems," *The adaptive Web: methods and strategies of Web personalization*, no. 9, pp. 291–324, 2007, ISSN: 20407459. DOI: 10.1504/IJEB.2004.004560. [Online]. Available: v:\%5CResearch\%5CResources\%5Cpapers\%5CRM_6609\%5CRM_6609.pdf\%5Cn10.1007/978-3-540-72079-9_9.

[160] D. Soydaner, "A comparison of optimization algorithms for deep learning," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 34, no. 13, p. 2052013, 2020. DOI: 10.1142/s0218001420520138. [Online]. Available: https://doi.org/10.1142%2Fs0218001420520138.

[161] A. Lydia and S. Francis, "A survey of optimization techniques for deep learning networks," pp. 2454–9150, May 2019. DOI: 10.35291/2454-9150.2019.0100.

[162] Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Y. Ng, "On optimization methods for deep learning," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ser. ICML'11, Omnipress, 2011, 265–272, ISBN: 9781450306195.

[163] A. Mustapha, L. Mohamed, and K. Ali, "Comparative study of optimization techniques in deep learning: Application in the ophthalmology field," *Journal of Physics: Conference Series*, vol. 1743, no. 1, p. 012002, 2021. DOI: 10.1088/1742-6596/1743/1/012002. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/1743/1/012002.

[164] R. S. Sexton, R. E. Dorsey, and J. D. Johnson, “Optimization of neural networks: A comparative analysis of the genetic algorithm and simulated annealing,” *European Journal of Operational Research*, vol. 114, no. 3, pp. 589–601, 1999, ISSN: 0377-2217. DOI: https://doi.org/10.1016/S0377-2217(98)00114-3. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221798001143.

[165] J. B. Ahire, *The artificial neural networks handbook: Part 1 - data science central*, https://www.datasciencecentral.com/profiles/blogs/the-artificial-neural-networks-handbook-part-1, (Accessed on 10/29/2018), Aug. 2018.

[166] S. Jadon, *Introduction to different activation functions for deep learning*, https://medium.com/@shrutijadon10104776/survey-on-activation-functions-for-deep-learning-9689331ba092, (Accessed on 10/29/2018), Mar. 2016.

[167] O. Russakovsky *et al.*, “ImageNet Large Scale Visual Recognition Challenge,” *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015. DOI: 10.1007/s11263-015-0816-y.

[168] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[169] D. Silver *et al.*, “A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play,” *Science*, vol. 1144, no. December, pp. 1140–1144, 2018.

[170] L. Hertel, E. Barth, T. Kaster, and T. Martinetz, “Deep convolutional neural networks as generic feature extractors,” in *Proceedings of the International Joint Conference on Neural Networks*, vol. 2015-Septe, 2015, ISBN: 9781479919604. DOI: 10.1109/IJCNN.2015.7280683. [Online]. Available: http://www.inb.uni-luebeck.de/publications/pdfs/HeBaKaMa15.pdf.

[171] Y. M. Costa, L. S. Oliveira, and C. N. Silla, “An evaluation of Convolutional Neural Networks for music classification using spectrograms,” *Applied Soft Computing*, vol. 52, pp. 28–38, 2017, ISSN: 15684946. DOI: 10.1016/j.asoc.2016.12.024. [Online]. Available: http://www.inf.ufpr.br/lesoliveira/download/ASOC2017.pdfhttp://linkinghub.elsevier.com/retrieve/pii/S1568494616306421.

[172] H. Kon and H. Koike, “Deep neural networks for cross-modal estimations of acoustic reverberation characteristics from two-dimensional images,” in *AES 144th Convention Paper*, Milan, 2018, pp. 1–13. DOI: 10.13140/2.1.1598.6882.

[173] E. Thuillie, H. Gamper, and I. J. Tashev, “Spatial Audio Feature Discovery With Convolutional Neural Networks,” in *IEEE ICASSP 2018*, 2018, p. 5. [Online]. Available: https://www.microsoft.com/en-us/research/uploads/prod/2018/04/Spatial_audio_feature_discovery_with_convolutional_neural_networks_ICASSP_2018.pdf.

[174] D. Britz, *Recurrent neural networks tutorial, part 1 – introduction to rnns – wildml*, http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/, (Accessed on 10/29/2018), Sep. 2015.

[175] G. Hoffman, *Introduction to lstms with tensorflow - o'reilly media*, https://www.oreilly.com/ideas, (Accessed on 10/29/2018), Jan. 2018.

[176] A. v. d. Oord *et al.*, "WaveNet: A Generative Model for Raw Audio," pp. 1–15, 2016, ISSN: 0899-7667. DOI: 10.1109/ICASSP.2009.4960364. [Online]. Available: http://arxiv.org/abs/1609.03499.

[177] A. Vaswani *et al.*, "Attention Is All You Need," *IEEE Industry Applications Magazine*, vol. 8, no. 1, pp. 8–15, 2017, ISSN: 1077-2618. DOI: arXiv.1706.03762. arXiv: 1706.03762. [Online]. Available: http://arxiv.org/abs/1706.03762.

[178] J. Wei *et al.*, *Emergent abilities of large language models*, 2022. arXiv: 2206.07682 [cs.CL].

[179] T. B. Brown *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[180] OpenAI, *Gpt-4 technical report*, 2023. arXiv: 2303.08774 [cs.CL].

[181] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, *Deep reinforcement learning from human preferences*, 2023. arXiv: 1706.03741 [stat.ML].

[182] X. Lu, Y. Tsao, S. Matsuda, and C. Hori, "Speech enhancement based on deep denoising autoencoder," in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2013, pp. 436–440.

[183] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 4, pp. 1802–1812, Dec. 2013, ISSN: 00189545. DOI: 10.1109/TVT.2013.2287343. [Online]. Available: http://arxiv.org/abs/1312.6114.

[184] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019. DOI: 10.1561/2200000056. [Online]. Available: https://doi.org/10.1561%2F2200000056.

[185] I. Higgins *et al.*, "Beta-VAE: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=Sy2fzU9gl.

[186] O. Rybkin, K. Daniilidis, and S. Levine, *Simple and effective vae training with calibrated decoders*, 2020. DOI: 10.48550/ARXIV.2006.13202. [Online]. Available: https://arxiv.org/abs/2006.13202.

[187] H. Fu, C. Li, X. Liu, J. Gao, A. Celikyilmaz, and L. Carin, *Cyclical annealing schedule: A simple approach to mitigating kl vanishing*, 2019. DOI: `10.48550/ARXIV.1903.10145`. [Online]. Available: `https://arxiv.org/abs/1903.10145`.

[188] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008. [Online]. Available: `http://jmlr.org/papers/v9/vandermaaten08a.html`.

[189] J. He, W.-S. Gan, and E.-l. Tan, "Can One " Hear " the Shape of a Person : Anthropometry Estimation via Head- Related Transfer Functions," in *2016 AES International Conference on Headphone Technology*, 2016, pp. 1–14, ISBN: 9781942220091. [Online]. Available: `http://www.aes.org/tmpFiles/elib/20180313/18347.pdfhttp://www.aes.org/e-lib/browse.cfm?elib=18347`.

[190] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2014. DOI: `10.48550/ARXIV.1409.1556`. [Online]. Available: `https://arxiv.org/abs/1409.1556`.

[191] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015. arXiv: `1505.04597`. [Online]. Available: `http://arxiv.org/abs/1505.04597`.

[192] I. Goodfellow, "NIPS 2016 Tutorial: Generative Adversarial Networks," in *NIPS 2016*, Oct. 2016, ISBN: 0018-8158. DOI: `10.1007/s10750-011-0734-0`. [Online]. Available: `http://arxiv.org/abs/1701.00160`.

[193] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep Image Prior," *Biorheology*, vol. 12, no. 3-4, pp. 219–24, Nov. 2017, ISSN: 0006-355X. DOI: `10.1023/A:1009804020976`. [Online]. Available: `http://arxiv.org/abs/1711.10925http://www.ncbi.nlm.nih.gov/pubmed/1124`.

[194] F. Farnia and A. Ozdaglar, "Do GANs always have Nash equilibria?" In *Proceedings of the 37th International Conference on Machine Learning*, H. D. III and A. Singh, Eds., ser. Proceedings of Machine Learning Research, vol. 119, PMLR, 2020, pp. 3029–3039. [Online]. Available: `https://proceedings.mlr.press/v119/farnia20a.html`.

[195] S. Agrawal, *A dozen times artificial intelligence startled the world*, `https://medium.com/archieai/a-dozen-times-artificial-intelligence-startled-the-world-eae5005153db`, (Accessed on 10/29/2018), Jul. 2017.

[196] K. McMullen and Y. Wan, "A machine learning tutorial for spatial auditory display using head-related transfer functions," *The Journal of the Acoustical Society of America*, vol. 151, no. 2, pp. 1277–1293, 2022, ISSN: 0001-4966. DOI: `10.1121/10.0007486`.

[197] X.-Y. Zeng, S.-G. Wang, and L.-P. Gao, "A hybrid algorithm for selecting head-related transfer function based on similarity of anthropometric structures," *Journal of Sound and Vibration*, vol. 329, no. 19, pp. 4093–4106, 2010, ISSN: 0022460X. DOI: `10.1016/j.jsv.2010.03.031`.

[198] C. Hoene, I. C. P. Mejía, and A. Cacerovschi, “MySofa: Design Your Personal HRTF,” in *142nd Convention Audio Engineering Society*, 2017, pp. 2–6. [Online]. Available: `http://www.aes.org/tmpFiles/elib/20170619/18640.pdf`.

[199] *360 reality audio — so immersive. so real. — sony uk*, `https://www.sony.co.uk/electronics/360-reality-audio`, (Accessed on 01/03/2023).

[200] *Aural id - genelec.com*, `https://www.genelec.com/aural-id`, (Accessed on 01/03/2023).

[201] *Listen with personalised spatial audio for airpods and beats – apple support (uk)*, `https://support.apple.com/en-gb/HT213318`, (Accessed on 01/03/2023).

[202] D Yao *et al.*, “An individualization approach for head-related transfer function in arbitrary directions based on deep learning,” *JASA Express lett. (JASA-EL)*, vol. 2, no. 6, p. 064 401, 2022.

[203] X Zhang, Q Zhang, W Zhang, and Z Liu, “Individualized hrtf estimation with a small set of measurements using collaborative regression forest,” in *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, 2016, pp. 6210–6214.

[204] J Gamper and P. Kennedy, “Deep learning for head-related transfer function interpolation,” in *Proc. IEEE Workshop on Applications of Signal Process. to Audio and Acoust. (WASPAA)*, 2019, pp. 161–165.

[205] Y Ito, T Nakamura, S Koyama, and H Saruwatari, “Head-related transfer function interpolation from spatially sparse measurements using autoencoder with source position conditioning,” in *Proc. Int. Workshop on Acoust. Signal Enhancement (IWAENC)*, 2022, pp. 1–5.

[206] G Kestler, S Yadegari, and D Nahamoo, “Head related impulse response interpolation and extrapolation using deep belief networks,” in *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, 2019, pp. 266–270.

[207] P Siripornpitak, I Engel, I Squires, S. Cooper, and L Picinali, “Spatial upsampling of hrtf sets using generative adversarial networks: A pilot study,” *Front. in Signal Process.*, vol. 4, 2021.

[208] S Spagnol, F Avanzini, and F Bettin, “A machine learning approach to hrtf personalization exploiting anthropometric features,” *Acta Acustica*, vol. 4, pp. 1–17, 2020.

[209] K. Yamamoto and T. Igarashi, “Fully perceptual-based 3D spatial sound individualization with an adaptive variational autoencoder,” *ACM Transactions on Graphics*, vol. 36, no. 6, pp. 1–13, 2017, ISSN: 07300301. DOI: `10.1145/3130800.3130838`. [Online]. Available: `http://dl.acm.org/citation.cfm?doid=3130800.3130838`.

[210] D. Schönstein and B. F. Katz, “Variability in perceptual evaluation of HRTFs,” *AES: Journal of the Audio Engineering Society*, vol. 60, no. 10, pp. 783–793, 2012, ISSN: 15494950. [Online]. Available: `http://www.aes.org/tmpFiles/elib/20180216/16552.pdf`.

[211] A. Andreopoulou and B. F. Katz, "Investigation on Subjective HRTF Rating Repeatability," Tech. Rep., 2016. [Online]. Available: http://www.aes.org/e-lib.

[212] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7585, pp. 484–489, 2016, ISSN: 0028-0836. DOI: 10.1038/nature16961. [Online]. Available: http://dx.doi.org/10.1038/nature16961.

[213] A. Andreopoulou, D. R. Begault, and B. F. Katz, "Inter-Laboratory Round Robin HRTF Measurement Comparison," *IEEE Journal on Selected Topics in Signal Processing*, vol. 9, no. 5, pp. 895–906, 2015, ISSN: 19324553. DOI: 10.1109/JSTSP.2015.2400417.

[214] Audio Engineering Society Inc., "AES standard for file exchange - Spatial acoustic data file format," AES69–2015, 2015, ISSN: 19443277. DOI: 10.1186/BF03356041. [Online]. Available: http://www.aes.org/standards.

[215] G. Kearney and T. Doyle, "A HRTF database for virtual loudspeaker rendering," in *Proc. AES 139th Convention*, 2015, pp. 1–10. [Online]. Available: http://www.aes.org/tmpFiles/elib/20170805/17980.pdf.

[216] *Acoustics Research Institute, ARI HRTF Database.* https://www.kfs.oeaw.ac.at/index.php?view =article&id=608&lang=en, 2014, Accessed 19th July 2018.

[217] W. O. Brimijoin and M. A. Akeroyd, "The moving minimum audible angle is smaller during self motion than during source motion," *Frontiers in Neuroscience*, vol. 8, no. SEP, pp. 1–8, 2014, ISSN: 1662453X. DOI: 10.3389/fnins.2014.00273.

[218] G. Wersényi, "Localization in a HRTF-based Minimum-Audible-Angle Listening test for GUIB applications," *Electronic Journal Technical Acoustics*, 2007. [Online]. Available: http://www.ejta.org.

[219] A. Andreopoulou and B. F. Katz, "Comparing the effect of HRTF processing techniques on perceptual quality ratings," in *AES 144th Convention Paper*, Milan, 2018, pp. 1–6.

[220] *SOFA: Spatially Oriented Format for Acoustics.* https://github.com/sofacoustics/API 2015, Accessed 19th July 2018.

[221] A. McKeag and D. S. McGrath, "Sound field format to binaural decoder with head tracking," *Audio Engineering Society 6th Australian Reagional Convention*, pp. 1–9, 1996. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=7477.

[222] M. Noisternig, A. Sontacchi, T. Musil, and R. Holdrich, "A 3d ambisonic based binaural sound reproduction system," in *Audio Engineering Society Conference: 24th International Conference: Multichannel Audio, The New Reality*, 2003. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=12314.

[223] C. Porschmann, J. M. Arend, and F. Brinkmann, "Directional equalization of sparse head-related transfer function sets for spatial upsampling," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 27, no. 6, pp. 1060–1071, 2019, ISSN: 23299290. DOI: 10.1109/TASLP.2019.2908057.

[224] G. Kearney and T. Doyle, "Height perception in ambisonic based binaural decoding," *139th Audio Engineering Society International Convention, AES 2015*, pp. 1–10, 2015.

[225] R. Sutton, *The bitter lesson*, http://www.incompleteideas.net/IncIdeas/BitterLesson.html, (Accessed on 10/29/2019), Mar. 2019.

[226] L. Sorber, M. V. Barel, and L. D. Lathauwer, "Unconstrained optimization of real functions in complex variables," *SIAM J. OPTIM*, vol. 22, 2012.

[227] T. Kim and T. Adalı, "Approximation by fully complex multilayer perceptrons," *Neural Computation*, vol. 15, no. 7, pp. 1641–1666, 2003.

[228] C. Trabelsi *et al.*, "Deep complex networks," in *ICLR2018-conf*, arxiv:1705.09792, 2018. [Online]. Available: https://iclr.cc/Conferences/2018/Schedule?showEvent=2.

[229] A. M. Sarrof, "Complex neural networks for audio," Ph.D. dissertation, Dartmouth College, 2018.

[230] J. Pauwels and L. Picinali, *On the relevance of the differences between hrtf measurement setups for machine learning*, 2022. arXiv: 2212.04283 [eess.AS].

[231] C. Szegedy *et al.*, "Going deeper with convolutions," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June, pp. 1–9, 2015, ISSN: 10636919. DOI: 10.1109/CVPR.2015.7298594.

[232] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," *Nature*, vol. 521, no. 7553, p. 800, 2016, ISSN: 0028-0836. DOI: 10.1038/nmeth.3707. [Online]. Available: http://goodfeli.github.io/dlbook/\%0Ahttp://dx.doi.org/10.1038/nature14539.

[233] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness," no. c, pp. 1–22, Nov. 2018. [Online]. Available: http://arxiv.org/abs/1811.12231.

[234] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *NIPS*, 2017.

[235] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[236] D. Masters and C. Luschi, "Revisiting Small Batch Training for Deep Neural Networks," pp. 1–18, 2018, ISSN: 1873-2976. DOI: 10.1016/j.biortech.2007.04.007. [Online]. Available: http://arxiv.org/abs/1804.07612.

[237] D. R. Wilson and T. R. Martinez, “The general inefficiency of batch training for gradient descent learning,” *Neural Networks*, vol. 16, no. 10, pp. 1429–1451, 2003, ISSN: 08936080. DOI: 10.1016/S0893-6080(03)00138-2.

[238] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2015.

[239] A. Krogh and J. A. Hertz, “A simple weight decay can improve generalization,” in *Advances in Neural Information Processing Systems 4*, J. E. Moody, S. J. Hanson, and R. P. Lippmann, Eds., Morgan-Kaufmann, 1992, pp. 950–957. [Online]. Available: http://papers.nips.cc/paper/563-a-simple-weight-decay-can-improve-generalization.pdf.

[240] L. N. Smith, “A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay,” *ArXiv*, vol. abs/1803.09820, 2018.

[241] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” in *ICLR*, 2019.

[242] G. E. Hinton, A. Krizhevsky, and I. Sutskever, *System and method for addressing overfitting in a Neural Network*, 2016. [Online]. Available: https://patents.google.com/patent/US9406017B2/en.

[243] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html.

[244] P. Baldi and P. Sadowski, “Understanding dropout,” *Advances in Neural Information Processing Systems*, pp. 1–9, 2013, ISSN: 10495258. DOI: 10.17744/mehc.25.2.xhyreggxdcd0q4ny.

[245] *Other regularization methods - practical aspects of deep learning — coursera*, https://www.coursera.org/lecture/deep-neural-network/other-regularization-methods-Pa53F, (Accessed on 01/01/2020).

[246] C. Armstrong, T. McKenzie, D. Murphy, and G. Kearney, “A perceptual spectral difference model for binaural signals,” *145th Audio Engineering Society International Convention, AES 2018*, pp. 1–5, 2018.

[247] R. Baumgartner, P. Majdak, and B. Laback, “Modeling sound-source localization in sagittal planes for human listeners,” *The Journal of the Acoustical Society of America*, vol. 136, no. 2, pp. 791–802, 2014, ISSN: 0001-4966. DOI: 10.1121/1.4887447. [Online]. Available: https://www.kfs.oeaw.ac.at/research/Baumgartner_et_al_2014.pdf.

[248] T. May, S. Van De Par, and A. Kohlrausch, “A probabilistic model for robust localization based on a binaural auditory front-end,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, no. 1, pp. 1–13, 2011, ISSN: 15587916. DOI: 10.1109/TASL.2010.2042128.

[249] P. Søndergaard and P. Majdak, “The auditory modeling toolbox,” in *The Technology of Binaural Listening*, J. Blauert, Ed., Berlin, Heidelberg: Springer, 2013, pp. 33–56.

[250] Engel, Isaac, Goodman, Dan F. M., and Picinali, Lorenzo, "Assessing hrtf preprocessing methods for ambisonics rendering through perceptual models," *Acta Acust.*, vol. 6, p. 4, 2022. DOI: 10.1051/aacus/2021055. [Online]. Available: https://doi.org/10.1051/aacus/2021055.

[251] C. C. Aggarwal, *Neural Networks and Deep Learning: A Textbook*. Springer International Publishing, 2018, ISBN: 9783319944630. [Online]. Available: https://books.google.co.uk/books?id=achqDwAAQBAJ.

[252] R. M. Zur, Y. Jiang, L. L. Pesce, and K. Drukker, "Noise injection for training artificial neural networks: A comparison with weight decay and early stopping," *Medical Physics*, vol. 36, no. 10, pp. 4810–4818, Sep. 2009, ISSN: 00942405. DOI: 10.1118/1.3213517. [Online]. Available: http://doi.wiley.com/10.1118/1.3213517.

[253] A. S. Rakin, Z. He, and D. Fan, "Parametric Noise Injection: Trainable Randomness to Improve Deep Neural Network Robustness against Adversarial Attack," pp. 1–15, Nov. 2018. [Online]. Available: http://arxiv.org/abs/1811.09310.

[254] L. Gatys, A. Ecker, and M. Bethge, "A Neural Algorithm of Artistic Style," *Journal of Vision*, vol. 16, no. 12, p. 326, 2016, ISSN: 1534-7362. DOI: 10.1167/16.12.326.

[255] M. Zhang and Y. Zheng, "Hair-gans: Recovering 3d hair structure from a single image," *arXiv preprint arXiv:1811.06229*, 2018.

[256] A. Brock, J. Donahue, and K. Simonyan, "Large Scale GAN Training for High Fidelity Natural Image Synthesis," *Veterinary Immunology and Immunopathology*, vol. 166, no. 1-2, pp. 33–42, Sep. 2018, ISSN: 18732534. DOI: 10.1016/j.vetimm.2015.04.007. [Online]. Available: http://arxiv.org/abs/1809.11096.

[257] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016, ISSN: 10495258.

[258] S. Ploumpis *et al.*, "Towards a complete 3D morphable model of the human head," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 4142–4160, 2021.

[259] G. Hu *et al.*, "Face recognition using a unified 3d morphable model," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham: Springer International Publishing, 2016, pp. 73–89, ISBN: 978-3-319-46484-8.

[260] V. Blanz and T. Vetter, "Face recognition based on fitting a 3d morphable model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, pp. 1063–1074, 2003. DOI: 10.1109/TPAMI.2003.1227983.

[261] O. Aldrian and W. A. Smith, "Inverse rendering of faces with a 3d morphable model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 5, pp. 1080–1093, 2013. DOI: 10.1109/TPAMI.2012.206.

[262] F. C. Staal, A. J. Ponniah, F. Angullia, C. Ruff, M. J. Koudstaal, and D. Dunaway, "Describing crouzon and pfeiffer syndrome based on principal component analysis," *Journal of Cranio-Maxillofacial Surgery*, vol. 43, no. 4, pp. 528–536, 2015, ISSN: 1010-5182. DOI: `https://doi.org/10.1016/j.jcms.2015.02.005`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S101051821500027X`.

[263] H. Dai, N. Pears, W. Smith, and C. Duncan, "A 3D Morphable Model of Craniofacial Shape and Texture Variation," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 3104–3112, 2017, ISSN: 15505499. DOI: `10.1109/ICCV.2017.335`.

[264] J. Booth, A. Roussos, S. Zafeiriou, A. Ponniahy, and D. Dunaway, "A 3D morphable model learnt from 10,000 faces," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December, pp. 5543–5552, 2016, ISSN: 10636919. DOI: `10.1109/CVPR.2016.598`.

[265] M. Geronazzo, S. Spagnol, and F. Avanzini, "Mixed structural modelling of head-related transfer functions for customized binaural audio delivery," in *2013 18th International Conference on Digital Signal Processing (DSP)*, 2013, pp. 1–8. DOI: `10.1109/ICDSP.2013.6622764`.

[266] H. Møller, C. B. Jensen, D. Hammershøi, and M. F. Sørensen, "Design criteria for headphones," *Journal of the Audio Engineering Society*, vol. 43, no. 4, pp. 218–232, 1995.

[267] V. Larcher, J.-M. Jot, and G. Vandernoot, "Equalization methods in binaural technology," in *Audio Engineering Society Convention 105*, Audio Engineering Society, 1998, p. 4858.

[268] W. Kreuzer, P. Majdak, and Z. Chen, "Fast multipole boundary element method to calculate head-related transfer functions for a wide frequency range," *The Journal of the Acoustical Society of America*, vol. 126, pp. 1280–1290, 3 Sep. 2009, ISSN: 0001-4966. DOI: `10.1121/1.3177264`.

[269] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, pp. 125–141, May 2008. DOI: `10.1007/s11263-007-0075-7`.

[270] L. Zhou, H. Liu, J. Bae, J. He, D. Samaras, and P. Prasanna, *Self pre-training with masked autoencoders for medical image analysis*, 2022. DOI: `10.48550/ARXIV.2203.05573`. [Online]. Available: `https://arxiv.org/abs/2203.05573`.

[271] M. Kohlbrenner, "Pre-training cnns using convolutional autoencoders," 2017.