
Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Elettronica, Elettrotecnica e delle Telecomunicazioni

---

# Deep Optimization of Discrete Time Filters for Listening Experience Personalization

**Deep Optimization of Discrete Time Filters for Personal Audio Systems**

Ph.D. Dissertation of:
**Giovanni Pepe**

Advisor:
**Prof. Stefano Squartini**

Coadvisor:
**Ing. Luca Cattani**

Curriculum Supervisor:
**Prof. Giuseppe Orlando**

XX edition - new series


Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Elettronica, Elettrotecnica e delle Telecomunicazioni

---

# Deep Optimization of Discrete Time Filters for Listening Experience Personalization

**Deep Optimization of Discrete Time Filters for Personal Audio Systems**

Ph.D. Dissertation of:
**Giovanni Pepe**

Advisor:
**Prof. Stefano Squartini**

Coadvisor:
**Ing. Luca Cattani**

Curriculum Supervisor:
**Prof. Giuseppe Orlando**

XX edition - new series

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Facoltà di Ingegneria
Via Brecce Bianche – 60131 Ancona (AN), Italy

*To my family*

# Acknowledgments

I wish to thank my supervisor, Prof. Stefano Squartini, for supporting me during my years of study and research, motivating me, and teaching me what it means to do research with patience and dedication. I would also like to thank Leonardo Gabrielli, who, thanks to his experience, has helped me a lot in these three years. I also thank all the colleagues of the A3Lab group with whom I shared these three years.

A special thank goes to ASK Industries S.p.A. for supporting the research activity. In particular, I would like to thank Luca Cattani and Carlo Tripodi for giving me the resources for the research activity, and Nicolò Strozzi, Michele Ebri, Livio Ambrosini and Marco Vizzaccaro for the support they gave me.

Finally, I would like to thank my family and friends for supporting me during these three years.

*Ancona, Novembre 2021*

Giovanni Pepe

# Abstract

This thesis describes the study of Machine Learning techniques for the optimization of digital filters for Multipoint Audio Equalization and Personal Sound Zones (PSZ) in a car scenario. Multipoint Audio Equalization is a topic that aims to improve the audio quality in a loudspeaker system using digital filters. The Personal Sound Zones is a task that allows the reproduction of different sounds in several regions contained within a listening environment where multiple listeners are present.

An up-to-date state of the art on digital filter design, Multipoint Audio Equalization and PSZ techniques have been reported in this thesis. Neural network-based optimization techniques, referred to as Deep Optimization, proved to be the best performing and the most analyzed methods within the proposed approaches. The technique exploits neural networks to iteratively optimize the filter parameters using the feed-forward and backpropagation, updating the weights with an optimizer. A new Deep Optimization architecture has been analyzed, called Bias Network (BiasNet), which uses the bias terms as input and updates its weights to obtain the optimal filters.

Experiments for Multipoint Audio Equalization with FIR filters were performed within various automotive scenarios, achieving better results than the state-of-the-art techniques. Other experiments were carried out with Parametric IIR filters, achieving better performance than baseline IIR and FIR filter design methods. Furthermore, analyzing the computational cost, Parametric IIR filters require less operations and memory.

Finally, experiments were conducted to design FIR and Parametric IIR filters for PSZ, introducing regularization and penalty terms to eliminate artefacts generated by FIR filters. The results are very promising, achieving a high acoustic contrast keeping high sound quality. IIR filters achieved comparable results with a lower computational cost than FIR filters.

# Sommario

Questa tesi descrive lo studio di tecniche di Machine Learning per l'ottimizzazione di filtri digitali per l'Equalizzazione Audio Multipunto e la Personal Sound Zones (PSZ) all'interno di uno scenario automotive. L'Equalizzazione Audio Multipunto è un argomento che mira a migliorare la qualità audio in un sistema di altoparlanti utilizzando filtri digitali. La Personal Sound Zones è un task che permette la riproduzione dei suoni in diverse regioni contenute in un ambiente d'ascolto dove sono presenti più ascoltatori.

In questa tesi, è stato riportato uno stato dell'arte aggiornato sulla progettazione di filtri digitali, tecniche di Equalizzazione Audio Multipunto e di PSZ. In questa dissertazione, le tecniche di ottimizzazione basate sulle reti neurali, denominate Deep Optimization, hanno dimostrato di essere le più performanti tra i metodi proposti. L'approccio sfrutta le reti neurali per ottimizzare iterativamente i parametri dei filtri utilizzando la feed-forward e la backpropagation e aggiornando i pesi con un ottimizzatore. È stata analizzata una nuova architettura di ottimizzazione profonda, chiamata Bias Network (BiasNet), la quale utilizza i termini di bias come input e aggiorna i suoi pesi per ottenere i filtri ottimali.

Gli esperimenti per l'equalizzazione audio con filtri FIR sono stati eseguiti all'interno di vari scenari automotive, ottenendo risultati migliori rispetto alle tecniche presenti nello stato dell'arte. Altri esperimenti sono stati eseguiti con i filtri Parametrici IIR, ottenendo prestazioni migliori rispetto alle tecniche di progettazione dei filtri IIR e FIR. Infine, analizzando il costo computazionale, i filtri IIR Parametrici richiedono meno operazioni e meno memoria.

Infine, sono stati condotti esperimenti per progettare filtri FIR e IIR parametrici per PSZ, introducendo termini di regolarizzazione e penalità per eliminare gli artefatti generati dai filtri FIR. I risultati sono molto promettenti, ottenendo un alto contrasto acustico mantenendo una qualità del suono alta. I filtri IIR hanno ottenuto dei risultati comparabili con un costo computazionale inferiore rispetto ai filtri FIR.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

In recent years the use of vehicles has been continuously growing, becoming an increasingly used listening environment: from music, radio, calls to the use of voice assistants, infotainment systems are widely used inside the car; for this reason, they are improving their quality, with increasingly high performance hardware. Indeed, Digital Signal Processing (DSP) allows the control of audio systems, designing digital filters of, theoretically, unlimited orders, but actually with a limitation of filter coefficients due to the necessity to keep the cost of hardware components low in the automotive industry. Furthermore, several complex techniques are implemented within the DSP systems, i.e., the Active Noise Control and Speech Enhancement, reducing the computational resources available to improve sound quality in real-time.

A wide variety of interior configurations characterizes the automotive listening environment: speaker locations, glass surface area, seat surface materials, plastic, geometry, and small size of the cabin [1] could affect the response of an audio system aspects [2]. In particular, early reflections and standing waves are the most critical in the car cabin, causing a high degradation due to the Hass effect, the Masking and the spatial sensation [1]. Depending on the absorbing or reflecting materials, the speaker positions and the shape of the car cabin, the reflected sounds could attenuate or amplify the direct sound from the loudspeakers [3].

In the research community and, in particular, in the automotive industry, new techniques for digital filter design for Multipoint Audio Equalization and Personal Sound Zones (PSZ) are being studied and improved. These two tasks are the main topics of this thesis work, and their deployment will focus on different automotive scenarios.

The Multipoint Audio Equalization task has the goal to improve sound quality within the listening environment. This technique usually provides a frequency curve in order to modify the listening experience. The curve could be referred to the listening of particular musical genres to boost low frequencies in the presence of background noise coming from the engine noise, as the automotive equalization [4]. In some cases, the filters are designed to perform a

virtual displacement of the sound sources to give to the listener the subjective impression of hearing at a pair of virtual loudspeakers [5].

The Personal Sound Zones is the other task discussed in this thesis: in a region of space composed of multiple listeners, the goal is to reproduce a sound source in a particular zone, minimizing the interference to the listeners located in the other zones. The idea is to deliver different interface-free audio to multiple listeners in the same environment without physical isolation or headphones.

In the automotive scenario, the PSZ could be applied to listen different sound sources without the use of headphones. For example: the voice navigator assistant is directed to the driver position, while the radio is straight to the co-driver, and the audio from a video file is directed towards the rear passengers.

For conciseness, from here, the term PSZ will be referred only to the two zones used for experiments, the driver and co-driver position,

A widespread issue is to analyze the performance using objective metrics: the most common ones are based on frequency response but they do not take into account the psychoacoustic aspects. In literature, many works discuss perceptual metrics that depend strongly on the results obtained by a small group of persons or experts, so they are referred to as subjective analysis. Particularly in the case of PSZ, many analyses have been performed on the effectiveness of the contrast between two zones, concluding that much depends on the media content reproduced, the contrast, the acoustic scene, and the listeners (whether they are experts, have hearing problems, and so on).

## 1.1 Problem Statement and Motivation

Multipoint Audio Equalization and Personal Sound Zones are similar tasks: the second one is an extension of the first because, in one zone, the goal is to achieve a quiet zone, whereas, in the other zone, the aim is to improve the sound quality.

Multipoint Audio Equalization is a complex task, although it is a linear problem (see Figure 1.1): several sources and microphones involve a large number of impulse responses to equalize. The complexity increases with the number of sources and microphones.

Multipoint Audio Equalization is usually performed using filters [6]: in a scenario with $S$ speakers and $\mathcal{M}$ microphones, the filter $g_s(t)$ of the $s$-th loudspeaker filters the input signal $x(t)$. The signal on the $m$-th microphone is the combination of the $S$ filtered signals:

$$y_m(t) = \sum_{s=1}^{\mathcal{S}} h_{m,s}(t) * (g_s(t) * x(t)) \tag{1.1}$$

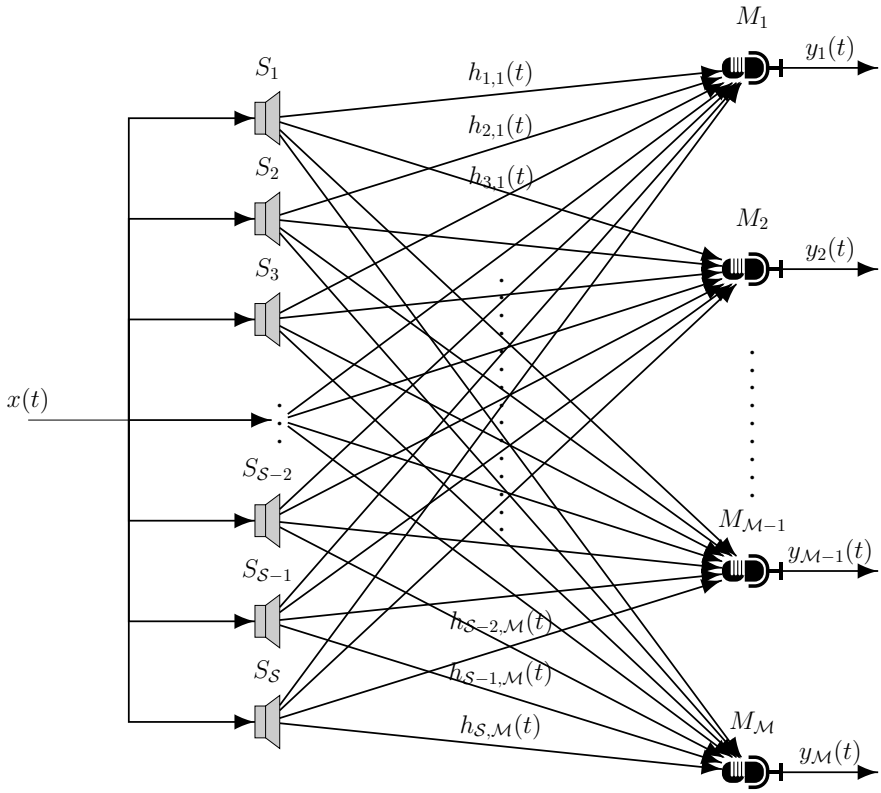



Figure 1.1: Multipoint Audio Equalization problem.

where $h_{m,s}(t)$ is the impulse response between the $s$-th speaker and the $m$-th microphone.

The frequency response at the $m$-th microphone is given by the Fourier transform of the output signal $y_m(t)$.

$$Y_m(\omega) = |\mathscr{F}(y_m(t))| \tag{1.2}$$

where $\mathscr{F}$ is the Fourier transform.

When the input signal is a Dirac impulse response ($x(t) = \delta(t)$), the Equation 1.1 could be written as:

$$\tilde{h}_m(t) = \sum_{s=1}^{\mathcal{S}} \tilde{h}_{m,s}(t) = \sum_{s=1}^{\mathcal{S}} h_{m,s}(t) * (g_s(t) * \delta(t)) = \sum_{s=1}^{\mathcal{S}} h_{m,s}(t) * (g_s(t)) \tag{1.3}$$

and the frequency responses at the $m$-th microphone is:

$$\tilde{H}_m(\omega) = |\mathscr{F}(\tilde{h}_m(t))| \tag{1.4}$$

Filtering allows to obtain the different characteristics desired by the end-user. In the research community, a desired amplitude flat band of 0 dB is usually used, although there are other very common shapes, such as having a higher amplitude of a few dB at low frequencies to accentuate the bass.

However, it is necessary to consider that filters can add artefacts, in particular Finite Impulse Response (FIR) filters which can bring reverberation, ringing, pre-ringing and most importantly, a delay. For this reason, some Multipoint Audio Equalization algorithms take these psychoacoustic aspects into account, or symmetric FIR filters are made at least to reduce artefacts [7].

Infinite Impulse Response (IIR) filters do not suffer from these problems, but they can lead to instability and non-linear phase if poorly designed [6].

The Personal Sound Zones has similar characteristics to the Multipoint Audio Equalization: digital filters need to be designed to improve sound quality in one zone and attenuate the sound pressure in the other one.

In this thesis, Deep Neural Networks (DNNs) are used to optimize FIR filters coefficients and Parametric IIR filters parameters: since these tasks are optimization problems, the neural networks must not generalize, but they should provide the output values that will serve to equalize or to separate the sounds in the two zones. The reasons why it was chosen to optimize instead of generalizing are the following: the impulse responses are measured with very different instruments; the acoustic scenes, in particular inside the car cabin, have different characteristics; few data are available to generalize the issue.

## 1.2 Thesis Outline

The thesis is organized as follows. In Chapter 2, Machine Learning techniques, Evolutionary Algorithms, Deep Neural Networks and Deep Optimization are explained, while in Chapter 3 a description of FIR and IIR filter design is presented. In Chapter 4, Multipoint Audio Equalization is discussed, describing the state-of-the-art algorithms used for the task and the new proposed methods. Finally, the experiments and the results are presented. In Chapter 5, the Personal Sound Zones is debated: the state-of-the-art algorithms are discussed, then the new proposed methods and the perceptual metrics used for the evaluation are presented. Finally, the experiments and the results are shown.

Chapter 6 shows the other contribution relying on Artificial Intelligence but not concerns the Multipoint Audio Equalization and PSZ. The Road Type Classification using Deep Learning methods is described, implementing the

method in a DSP system. In particular, roughness and wetness road classification is studied, starting from a separate investigation of the two tasks and a going to joint analysis. Then, Voice Activity Detection and Speaker Localization in a Multi-Room environment are discussed, jointly implementing a data-driven method for the two tasks. Finally, Sound Event Detection and Source Separation systems are discussed, using Deep Learning and Machine Learning approaches.

Finally, Chapter 7 concludes the thesis, also formulating some proposals for future works.

# Chapter 2

# Machine Learning Techniques and Optimization Problems

Machine Learning is a subset of Artificial Intelligence that build a mathematical model based on sample data [8]. The goal is to make predictions or decisions without being explicitly programmed to perform the task. It derives from statistics, computer science, engineering, optimization theory and other disciplines of science and mathematics [9].

Machine Learning can be classified according to the kind of learning:

- Supervised learning: given a set of input and output variables, the aim is to learn a mapping to predict the outputs for unseen data [10];
- Unsupervised learning: given a set of unlabelled inputs, the goal is to find a solution on its own with no supervised target [10]: the algorithm builds representations of the input data for decision making and predicts the unknown inputs. An example of unsupervised learning is clustering;
- Semi-supervised learning: given a small amount of labelled data and a large amount of unlabelled data, the goal is to teach itself to predict unseen labels [8];
- Reinforcement learning: the goal is to map situations to actions in order to maximize a reward. The algorithm must discover which actions yield the most reward by trying them. Actions may affect reward or a penalty [11];
- Transfer learning: the goal is to improve learning in a new task by leveraging knowledge from a related task that has already been learned [12].

The optimization problems arise throughout Machine Learning because the objective is to design a classifier or a predictor to give a correct value for any unknown input vector.

An optimization problem is the task of finding the best solution from all the feasible solutions [13]. Given the optimization variable and the bounds for the constraints to the problem, the goal is to minimize the objective function:

$$\begin{aligned} &\text{minimize} \quad f_0(x) \\ &\text{subject to} \quad f_i(x) \leq b_i, \quad i = 1, \cdots, m. \end{aligned} \tag{2.1}$$

where $x = (x_1, \cdots, x_n)$ is the optimization variable of the problem, the function $f_0$ is the objective function, $f_i$, $i = 1, \cdots, m$ are the constraint functions and $b_i$ are the limits or bound for the constraints. A vector $x^*$ is called optimal or a solution if it has the smallest objective value among all vectors that satisfy the constraints.

The optimization problems can be divided into two main classes. The first is the linear program if the objective functions are linear:

$$f_i(\alpha x + \beta y) = \alpha f_i(x) + \beta f_i(y) \tag{2.2}$$

where $\alpha$ and $\beta$ are constant values. If the optimization problem is not linear, it is called a nonlinear program.

A solution method is an algorithm that computes a solution to the problem. The effectiveness of the technique depends on factors such as the particular forms of the objective and constraint functions, how many variables and constraints are present, and their particular structure, i.e. sparsity.

An optimization problem is convex if the objective and constraint functions are convex (see Figure 2.1.a); thus, they satisfy the following inequality:

$$f_i(\alpha x + \beta y) \leq \alpha f_i(x) + \beta f_i(y) \tag{2.3}$$

Because the inequality is more restrictive than equality and is held only for specific values of $\alpha$ and $\beta$, the convex optimization can be considered a generalization of the linear program.

An optimization problem is not convex when the objective function or any of the constraints are non-convex [14]. Local optimization methods have been studied to find a local and global minimum point. Usually, straightforward problems with many variables can be intractable, so an optimal solution is determined at a local minimum. This kind of optimization problems presents feasible and flat regions, a widely varying curvature, several saddle points and multiple local minima within each region (see Figure 2.1.b).

Local optimization methods are sensitive to algorithm parameter values. The optimization may require an initial guess, which is critical and can significantly affect the objective value of the local solution.

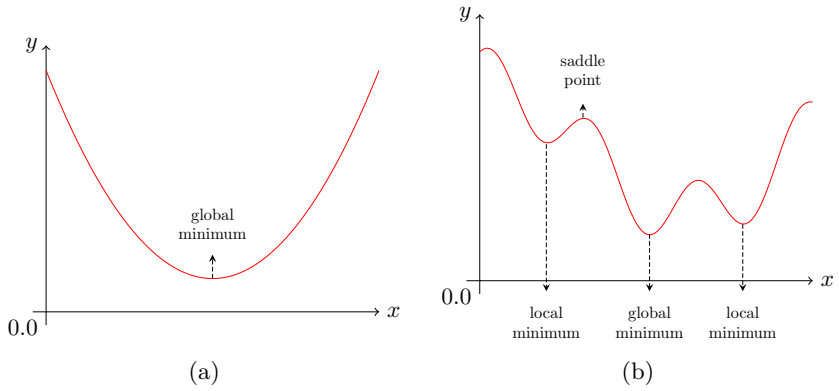



Figure 2.1: Example of convex and non-convex optimization problem: convex optimization problem (a) is composed of local minimum that is the same of the global minimum; non-convex optimization problem is composed of several local minimums and saddle points and a global minimum.

A multi-objective optimization problem is an optimization problem that includes multiple objective functions:

$$\begin{aligned} &\text{minimize} \quad (f_{0,1}(x), f_{0,2}(x), \cdots, f_{0,K}(x)) \\ &\text{subject to} \quad f_i(x) \leq b_i, \quad i = 1, \cdots, m. \end{aligned} \tag{2.4}$$

In a multi-objective optimization problem, does not exist a feasible solution that minimizes all objective functions simultaneously [15]. For this reason, the solutions improve one objective function but degrade at least one of the other objective functions. The set of Pareto optimal outcomes is called Pareto front, Pareto frontier or Pareto boundary. A Pareto optimal solution dominates any other feasible solution in the search space [16]. In Figure 2.2 is presented an example of the Pareto frontier using two generic objective functions. Diamonds and circles represent the feasible choices: the first ones are the Non-Pareto optimal choices because they are dominated by the other points (circle marks). Circles are Pareto optimal because any other points do not dominate them. Hence they lie on the frontier (red curve).

In [17] a Pareto Active Learning is performed to maximize progress on designs, identifying the set of Pareto-optimal in a multi-objective scenario. A Fully Connected Neural Network (FCNN) is used in [18] for the Drop-on-Demand Bioprinting, optimizing the voltage to print droplets. In [19] is discussed the first use of neural networks for multi-objective optimization.

Multi-Objective Evolutionary Algorithms are used to update neural networks

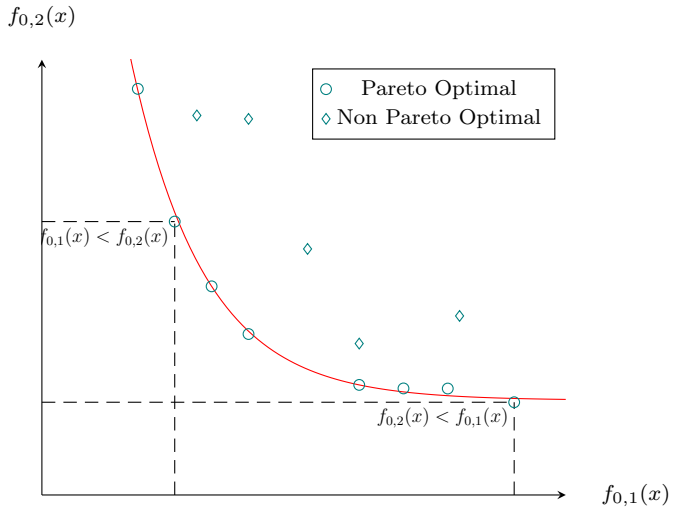



Figure 2.2: Example of Pareto frontier with two objective function. Circles and diamonds are the feasible choices; diamonds are not Pareto optimal solutions; circles are Pareto optimal solutions. The Non Pareto optimal solutions are dominated by the Pareto optimal solutions.

that dominate in several Pareto optimal solutions [20]. In [21] Multi-Objective Particle Swarm Optimization (MOPSO) is used to tune hyperparameters of several Machine Learning algorithms, as the Back-Propagation Neural Network (BPNN), Random Forest (RF), Support Vector Regression (SVR), Regression Tree (RT), $k$-Nearest Neighbor (KNN) and the Logistic Regression, to optimize the concrete mixture proportions. A new hybrid method has been described in [22], in which a Neural Evolutionary Algorithm, called Multi-Objective Neural Evolutionary Algorithm based on Decomposition and Dominance (MONEADD) is used for combinatorial optimization problems.

## 2.1 Evolutionary Algorithms

Evolutionary algorithms [23] are optimization techniques inspired by biology and natural phenomenon. According to the criteria and constraints of the problem, these methods can be distinguished between single-objective and multi-objective evolutionary algorithms [24].

Briefly, all evolutionary algorithms present the following steps: a population of individuals (agents, particles, genes etc.) is created; the values of objective functions are calculated for each individual; a fitness value is determined; the selection phase is executed, where solution candidates with bad or good fitness are selected, then the worst candidates are usually discarded; the reproduction

phase is performed, in which there is the combination of the selected individuals. Finally, the evolution is stopped when the criteria are met; elsewhere, the algorithm continues to optimize.

Evolutionary algorithms can be divided into three main classes [25]: biology-based algorithms, physics-based algorithms and geography-based algorithms. The biology-based methods present characteristics of natural evolution and biological behaviours, emulating cooperative behaviours of swarms, like the Particle Swarm Optimization (PSO). Physics-based techniques are inspired by physical phenomenons and rules. Finally, geography-based algorithms use geographics information to optimize problems; an example is the Tabu Search algorithm [26].

Many evolutionary algorithms are present in literature [23], from Genetic Algorithms [27] to new techniques such as Gravitational Search Algorithm (GSA) [28].

Evolutionary Algorithms are used for many purposes: PSO is used in [29] to analyze the multi-variable optimization problems, while in [30] an Adaptive PSO is described. Differential Evolution is presented in [31] to minimize non-linear and non-differentiable continuous space functions, analyzing it also in a multi-objective optimization problem [31].

Artificial Immune System [32] is a recent evolutionary algorithm based on antigens. Several works used the Artificial Immune system for multi-objective optimization [33, 34].

Gravitational Search Algorithm is the most recent analyzed algorithm [35]: in [25], a hybrid Gravitational Search and Pattern Search Algorithm (GSA-PS) approach is used to optimize and manage the energy in a grid network, whereas in [36], the GSA achieved better performance than the PSO in a multi-objective optimization analysis.

In this thesis, the techniques used for Multipoint Audio Equalization, both FIR and IIR, will be explained: the PSO is described in Section 2.1.1, while the GSA is explained in Section 2.1.2.

### 2.1.1 Particle Swarm Optimization

The Particle Swarm Optimization is an optimization algorithm based on the social behaviour of bird flocking and fish schooling [37]. The PSO is based on a population, or a swarm, of individual particles. Each particle crosses through the solution space to search for the global optimum. Each particle then modifies its position using the information of the distance between the current position, the local best $p_{best}$ and the global best $g_{best}$. The algorithm iteratively evaluates the fitness function at different locations creating a map of the best fitness values.

The algorithm starts with the generation of particles in a random position in the solution space. Then the fitness function is evaluated for each particle, calculating the $p_{best}$ in the current iteration and the $g_{best}$. Finally, the position $x_i^d$ and the velocity $v_i^d$ at instant $k$ is calculated:

$$v_i^d(k+1) = W \cdot v_i^d(k) + c_1 \cdot \zeta(k) \cdot (p_{best} - x_i^d) + c_2 \cdot \zeta(k) \cdot (g_{best}(k) - x_i^d(k)) \quad (2.5)$$

$$x_i^d(k+1) = x_i^d(k) + v_i^d(k+1) \quad (2.6)$$

where $W$ is the inertia weight, $\zeta(k)$ is a random value in the range [0, 1] and $c_1$ and $c_2$ are constants.

### 2.1.2 Gravitational Search Algorithm

Gravitational Search Algorithm [38] is a metaheuristic method based on the law of gravitational and mass attraction forces. The solution vectors $A$ are considered as agents attracted by each other by a force. The $i$-th agent is defined by the position $X_i = [x_i^1, \cdots, x_i^d, \cdots, x_i^D]$, where $x_i^d$ is the position of the $i$-th agent in the $d$-th dimension and $n$ is the dimension of each space. The mass of each agent $M_i$ is calculated as:

$$M_i(k) = \frac{q_i(k)}{\sum_{j=1}^{A} q_j(k)} \quad (2.7)$$

where $q_i$ is calculated by:

$$q_i(k) = \frac{fit_i(k) - worst(n)}{best(k) - worst(k)} \quad (2.8)$$

where $fit_i(k)$ is the fitness value of the $i$-th agent, $best(k)$ and the $worst(k)$ are the best and the worst fitness value of all agents at $k$-th iteration.

The force of each agent $F_i$ is calculated considering a set $A_{best}$ of heavier masses:

$$F_i^d(k) = \sum_{\substack{j \in A_{best} \\ j \neq i}} rand_j \cdot \mathcal{G}(k) \cdot \frac{M_j(k)\dot{M}_i(k)}{R_{ij}(k) + \epsilon} \cdot (x_j^d(k) - x_i^d(k)) \quad (2.9)$$

where $\epsilon$ is a small value, $R_{ij}(k)$ is the Euclidean distance between two agents, defined as $R_{ij}(k) = ||X_i(n), X_j(k))||$ and $A_{best}$ is the set of $A$ agents with the best fitness value and biggest mass. $\mathcal{G}$ is the gravitational constant, with an

initial value $\mathcal{G}_0$ and then it will be reduced with time:

$$\mathcal{G}(k) = \mathcal{G}(\mathcal{G}_o, k) \tag{2.10}$$

After, the acceleration of the agent $a_i^d$ is computed:

$$a_i^d(k) = \frac{F_i^d(k)}{M_i(k)} = \sum_{\substack{j \in A_{best} \\ j \neq i}} rand_j \cdot \mathcal{G}(k) \cdot \frac{M_j(k)}{R_{ij}(k) + \epsilon} \cdot (x_j^d(k) - x_i^d(k)) \tag{2.11}$$

Finally, the velocity and position of the agent, $v_i^d$ and $x_i^d$ respectively, are calculated:

$$v_i^d(k+1) = rand_i \cdot v_i^d(k) + a_i^d(k) \tag{2.12}$$

$$x_i^d(k+1) = x_i^d(k) + v_i^d(k+1) \tag{2.13}$$

where $rand_i$ and $rand_j$ are two uniformly distributed random numbers in the interval [0, 1].

The main differences between PSO and GSA are described in [28]: in PSO, the direction of a particle is calculated considering the local and global best position, while in GSA is based on the overall force obtained by all other agents; in GSA, the force is proportional to fitness value, and it is proportional to the distance between solutions. Finally, GSA is memory-less. Only the current position of the agents plays a role in the updating procedure.

## 2.2 Deep Optimization

The scope of this dissertation is to optimize digital filters with limited information (IRs, number of microphones and speakers), employing neural networks with a different approach than the common classification and regression problems: the neural network iteratively adjusts the filters accordingly to several objective functions, regularization and penalty terms by fitting its weights.

The neural networks can solve non-convex, non-linear or complex problems through the minimization of the cost function [39]; in fact, the training of a neural network is an optimization problem, where the loss term is minimized by the backpropagation of the error through the neural network. In this thesis, the proposed DNNs are used to solve optimization problems. This idea is not completely new, but it has already been proposed in recent years [40, 41]. In [42], the authors analyzed the MultiLayer Perceptron (MLP) for multi-objective and multi-level programming. In [39], neural networks are used to solve non-

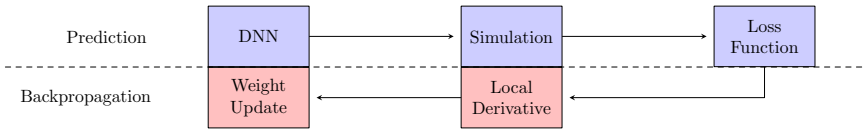



Figure 2.3: Diagram of a Deep Optimization process.

convex optimization problems, achieving good performance, possibly due to their parameter redundancy [43].

The Deep Optimization network process is divided into two steps (see Figure 2.3): the first is the prediction step, in which the neural network outputs the parameters to optimize. Then the simulation is performed and the fitness function is calculated. The second step is the backpropagation step: the partial derivatives are calculated from the local derivatives of the loss functions and the simulations. Finally, the weights and the parameters are updated using an optimizer.

## 2.3 Deep Neural Networks

Deep Neural Networks are mathematical models inspired by neuroscience [44]. The hidden layer of the network can be represented by a vector in which each element could be interpreted as a neuron, also called unit. Each neuron receives input from other units and computes its activation value.

Deep Neural Networks exploits backpropagation [45] to compute the gradients and update weights and bias using optimization algorithms like Stochastic Gradient Descent [44].

The advantage of using a neural network with respect to linear models is the non-convex optimization, resulting in a better loss function decrease.

DNNs were implemented to design filters for both Multipoint Audio Equalization and Personal Sound Zones. As described in Section 2.2, the proposed DNNs optimize parameters to solve problems with limited information. Several common neural architectures were analyzed, and a new architecture, named Bias Network (BiasNet) and described in Section 2.3.5, was implemented for the optimization problems under consideration.

### 2.3.1 MultiLayer Perceptron

The MultiLayer Perceptron is a feed-forward neural network composed of an Input layer, a cascade of hidden layers, and the Output layer [44] (see Figure 2.4). The network maps the input values to output values, achieving a function composed of non-linear activation functions.

The hidden layers are composed of computation nodes, called hidden neurons. Each neuron applies an activation function over the weighted sum of its inputs.

In Figure 2.4 is presented an example of a MLP: the input examples are fed into the Input layer, then the resulting output is propagated through the hidden layers towards the output layer. In the backpropagation, the error, calculated by the loss function, is sent back through the layers, and the network parameters are tuned with an optimization algorithm.

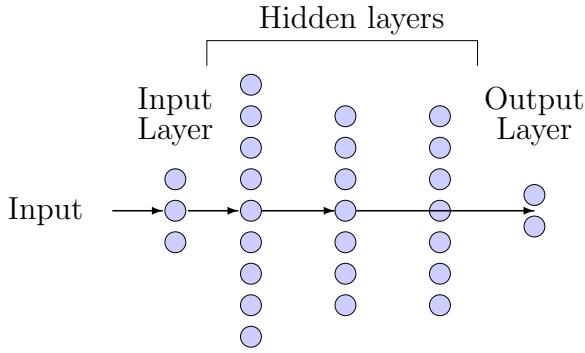



Figure 2.4: Example of MLP

### 2.3.2 Convolutional Neural Network

Convolutional Neural Network (CNN) is a neural network that is used to process data that has grid-like topology [44, 46]. This architecture has the properties to use the convolution between the input and the kernel (see Figure 2.5), achieving as output the feature map. The input is a multidimensional array (tensor) of data, and the kernel is usually a tensor adapted by the learning algorithm.

The input tensor is divided into local receptive fields, a region of the same size as the kernel. The convolution kernel processes the receptive field and slides it across the entire input. The whole input tensor is handled, repeating the convolution across its receptive field and achieving the feature map. The receptive field is convolved with the kernel, a bias term is added, and the activation function is performed. The weights of each feature map are shared.

Usually, a CNN is composed of convolutional layers, max pooling and hidden layers. Pooling layers reduce the dimension of the tensor using a mathematical rule, such as selecting the maximum value from a submatrix or averaging the submatrix [44]. These kinds of layers are useful because they introduce tolerance against shifts in the input patterns.

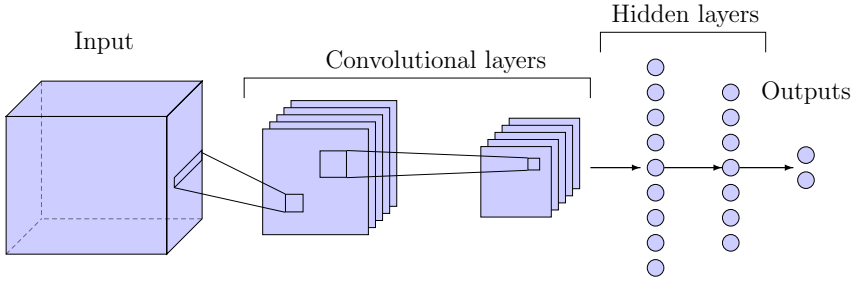



Figure 2.5: Scheme of a CNN.

The convolution process for a generic kernel is determined by:

$$y_m = f\Big(\sum_{d=1}^{D_3} W_m * u_d + b_m\Big) \tag{2.14}$$

where $y_m \in \mathbb{R}^{D_1 \times D_2}$ is the feature map, $W_m$ is the $m$-th kernel, $b_m \in \mathbb{R}^{D_1 \times D_2}$ is the bias vector and $u_d \in \mathbb{R}^{D_1 \times D_2}$ is the matrix of the input tensor $u \in \mathbb{R}^{D_1 \times D_2 \times D_3}$.

### 2.3.3 Autoencoder

Autoencoder (AE) is a neural network that is used to attempt to copy its input to its output [44]. The architecture is described in Figure 2.6. It is composed of a code in the network which forces a compressed knowledge representation of the original input. Usually, the code is a hidden layer that represents the input. The network consists of an encoder function that converts the input data into a different representation and a decoder function that converts the new representation back into the original format. Their scope is to preserve as much information as possible when the input runs through the encoder and the decoder. Indeed, the model learns the useful properties of the data [44]. Recently, AEs have been used as generative models [47] or Denoising Autoencoders [48].


Figure 2.6: Scheme of an Autoencoder.

### 2.3.4 Generative Adversarial Network

Generative Adversarial Network (GAN) [44] is a generative model that learns to map samples, obtained from a random distribution, that are as similar as possible to training examples. This architecture is composed of the Generator that learns to get an effective mapping that can imitate the real data distribution to generate novel samples related to the training set, and it does not memorize input-output pairs. The other component is the Discriminator, which is typically a binary classifier. Its inputs are either real samples coming from the training set or fake samples given from the Generator. In Figure 2.7 is presented the diagram of a GAN.

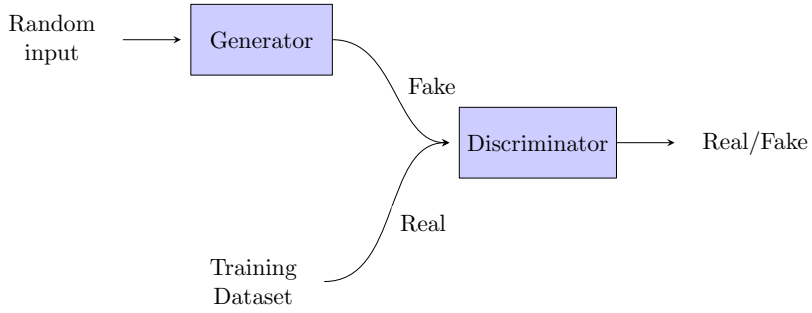



Figure 2.7: Scheme of a GAN.

The Discriminator attempts to learn correctly classify samples as real or fake. At the same time, the Generator attempts to fool the Discriminator into believing its samples are real. When the loss function converges, the Generator outputs indistinguishable samples from real data, and the Discriminator gives as output a value equal to 0.5.

### 2.3.5 Bias Network

In a Deep Optimization scenario, a neural network can be seen as a unidirectional graph of non-linear computations. With a fixed input, only the weights can change, then the convergence is only determined by the update of the network weights. If the activation functions satisfy the relation $f(0) = 0$, an input tensor with 0s will not produce any output, so the computation will be 0s. To inject an input to the network, bias terms of the input layer can be used because they are learnable parameters. Therefore, they will be updated during the optimization procedure.

Bias Network is presented in Figure 2.8. The vector of bias terms of input neurons $b^0$ can be seen as a learnable input vector. The network provides parameters that are used in the simulation process. Then, as described in

Figure 2.3, the loss function is calculated, and the partial derivatives of the simulation process are used to learn the optimal network weights and input bias terms.

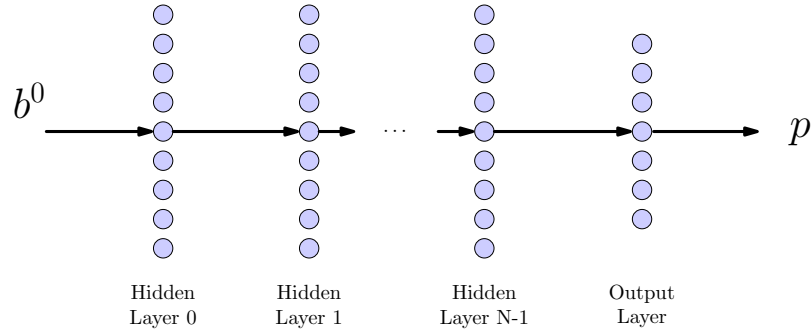



Figure 2.8: Bias Network.

The advantages of using this architecture lie in the absence of input, which avoids tweaking the input size and content, and the low number of network parameters to be learned, influencing the convergence speed.

The proposed BiasNet is composed of a cascade of fully connected layers. The neurons of the first hidden layer (Hidden layer 0 in Figure 2.8) is calculated as:

$$y^0 = f(w^0 \cdot x + b^0) \tag{2.15}$$

where $y^0$ is the vector of the neuron outputs, $w^0$ is the matrix of the weights, and $x$ is the input of the network. To change the weights in biases, $x$ must be a constant value, like 0 or 1. For simplicity, if $x = 0$, the neuron output could be:

$$y^0 = f(b^0) \tag{2.16}$$

and the partial derivative of the output with respect to the bias is:

$$\frac{\partial y^0}{\partial b^0} = \frac{\partial f(b^0)}{\partial b^0} \tag{2.17}$$

Assuming $x = 1$, the output is:

$$y^0 = f(w^0 + b^0) \tag{2.18}$$

and the partial derivatives are:

$$\frac{\partial y^0}{\partial w^0} = \frac{\partial f(w^0 + b^0)}{\partial w^0} \tag{2.19}$$

$$\frac{\partial y^0}{\partial b^0} = \frac{\partial f(w^0 + b^0)}{\partial b^0} \tag{2.20}$$

# Chapter 3

# FIR and IIR Filter Design

Digital filters are used to process the input signal [49], add digital effects or equalize either the acoustic scene or the channel communications. They are classified as FIR and IIR filters. The two groups of filters are Linear Time Invariant (LTI) systems [50].

FIR filters are systems with finite impulse responses. The difference equation is given by:

$$y(n) = \sum_{k=0}^{K-1} b_k \cdot x(n-k) \tag{3.1}$$

and the finite impulse response is:

$$h(n) = \sum_{k=0}^{K-1} b_k \cdot \delta(n-k) \tag{3.2}$$

thus, the impulse response is given by the weighted sum of the present sample and the past $K$ samples. By the feed-forward characteristic, these types of filters are non-recursive. The transfer function in Z-domain is given by:

$$H(z) = b_0 + b_1 \cdot z^{-1} + \cdots + b_{K-1} \cdot z^{-(K-1)} \tag{3.3}$$

FIR filtering operation can be used to operate on a block or sample-by-sample basis [50]. In block processing, the input signal is considered as a single block of signal samples. It is filtered by convolving it with the filter, generating the output signal as another block of samples. In the sample processing case, the signal samples are processed one at a time when they arrive at the input. In this case, the filter is used as a state machine. Each sample is used in conjunction with the current state of the filter to calculate the current output sample and update the internal state of the filter in preparation for the process of the next signal sample. The second approach is useful in real-time and adaptive filtering applications.

IIR filters are systems with an impulse response of infinite duration [51]. The

difference equation is given by:

$$y(n) = \sum_{k=0}^{K-1} b_k \cdot x(n-k) - \sum_{l=0}^{N-1} a_l \cdot y(n-l) \tag{3.4}$$

and the transfer function in Z-domain is:

$$H(z) = \frac{\sum_{k=0}^{K-1} b_k z^{-k}}{1 + \sum_{l=1}^{N-1} a_l z^{-k}} \tag{3.5}$$

IIR filters present poles in the transfer function, obtaining filters with a slightly non-linear phase [51]. Another characteristic due to the presence of the poles is stability; if the poles are outside the unit circle in Z-domain, the system is unstable.

IIR filters are designed using optimization techniques [51]. The classical approach in the time domain is the System Identification (SI) approach, in which the IIR system is designed by recursively updating the filter coefficients until the response is close to the unknown system [52]. Linear optimization techniques minimize the gradient of the error function with respect to filter coefficients. Fletcher-Powell algorithms [53], Linear Programming approach [54] and Least Squares techniques [55, 56] are examples of linear optimization techniques used for the IIR filter design [51].

Digital filters can be realized as direct form, canonical form and cascade form [50].

The direct form, called direct form I realization, is the block diagram representation of the difference equation:

$$y(n) = -a_1 \cdot y(n-1) - a_2 \cdot y(n-2) + \cdots + b_0 \cdot x(n) + b_1 \cdot x(n-1) + b_2 \cdot x(n-2) \tag{3.6}$$

In Figure 3.1 is presented a graphical example of the direct form I realization.

The canonical form, called direct form II, is achieved by the direct form I: the equation is split into two subgroups, the recursive and non-recursive terms:

$$y(n) = (b_0 \cdot x(n) + b_1 \cdot x(n-1) + b_2 \cdot x(n-2)) + (-a_1 \cdot y(n-1) - a_2 \cdot y(n-2)) \tag{3.7}$$

resulting in a realization of the cascade of two filters: one consists only of the feed-forward terms and the other of the feedback terms, the numerator and the denominator, respectively:

$$H(z) = N(z) \cdot \frac{1}{D(z)} \tag{3.8}$$

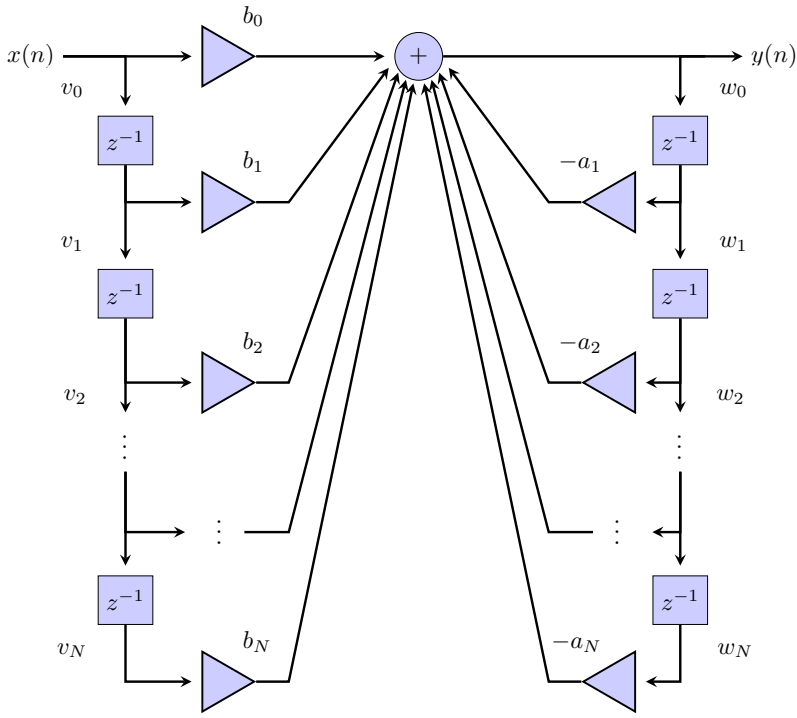



Figure 3.1: Direct form I realization of a generic IIR filter

The order of the cascade factors can be mathematically changed, thus:

$$H(z) = \frac{1}{D(z)} \cdot N(z) \tag{3.9}$$

The output signal of the filter $\frac{1}{D(z)}$ becomes the input of the second filter $N(z)$. The delays can be merged in one set, shared by both the first and second filters, leading to the canonical realization form, as shown in Figure 3.2.

A Second Order Section (SOS) is a second order transfer function:

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \tag{3.10}$$

The cascade realization form of a general transfer function assumes that the transfer function is the product of the SOS's:

$$H(z) = \prod_{i=0}^{K-1} H_i(z) = \prod_{i=0}^{N-1} \frac{b_{i,0} + b_{i,1} z^{-1} + b_{i,2} z^{-2}}{1 + a_{i,1} z^{-1} + a_{i,2} z^{-2}} \tag{3.11}$$

with real valued coefficients.

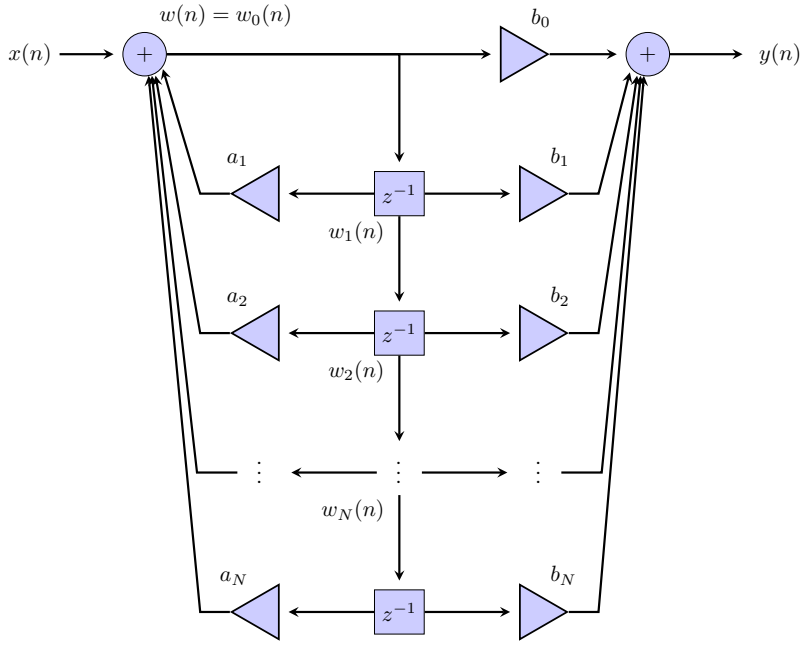



Figure 3.2: Direct form II realization of a generic IIR filter.

## 3.1 Parametric IIR Filter

The Parametric IIR filters are used to implement tunable low/high/band-pass and band-reject filters [57]. Several filters are needed to shape the spectrum, connecting them as a cascade of first and second-order filters.

In this thesis, Parametric IIR filters are used for the Multipoint Audio Equalization and PSZ task. The band-pass filters are designed, optimizing the central frequency $f_c$, the quality factor $Q$, the gain $G_0$ of each SOS, and the overall gain of the speaker $G_S$. The two typologies of filters to design are Boost filters and Cut filters: the first ones amplify the spectrum in a specific band region, while the Cut filters attenuate it [57]. In Tables 3.1 and 3.2 are described the calculus of the IIR filter coefficients from the parameters. After obtaining the coefficients, filtering is done using frequency convolution. Therefore the transfer function is transformed into frequency response.

## 3.2 State-of-the-art of FIR and IIR filters design

Digital filter design needs optimization algorithms [58]. A scheme of the main optimization techniques is shown in Figure 3.3.

Convex optimization methods have been used on the FIR filter design [59],

| IIR Coefficients | Boost ($V_{0_{s,\kappa}} \geq 1$) |
|---|---|
| $b_{s,\kappa,0}$ | $\frac{1+\frac{V_{0,s,\kappa}}{Q_{s,\kappa}} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}{1+1/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |
| $b_{s,\kappa,1}$ | $\frac{2 \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2 - 1}{1+1/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |
| $b_{s,\kappa,2}$ | $\frac{1-\frac{V_{0,s,\kappa}}{Q_{s,\kappa}} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}{1+1/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |
| $a_{s,\kappa,0}$ | 1 |
| $a_{s,\kappa,1}$ | $\frac{2 \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2 - 1}{1+1/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |
| $a_{s,\kappa,2}$ | $\frac{1-\frac{1}{Q_{s,\kappa}} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}{1+1/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |

Table 3.1: Coefficients calculated according to the Boost filter [57].

| IIR Coefficients | Cut ($0 < V_{0_{s,\kappa}} < 1$) |
|---|---|
| $b_{s,\kappa,0}$ | $\frac{1+\frac{1}{Q_{s,\kappa}} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}{1+V_{0,s,\kappa}/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |
| $b_{s,\kappa,1}$ | $\frac{2 \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2 - 1}{1+V_{0,s,\kappa}/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |
| $b_{s,\kappa,2}$ | $\frac{2 \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2 - 1}{1+V_{0,s,\kappa}/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |
| $a_{s,\kappa,0}$ | 1 |
| $a_{s,\kappa,1}$ | $\frac{2 \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2 - 1}{1+V_{0,s,\kappa}/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |
| $a_{s,\kappa,2}$ | $\frac{1-\frac{V_{0,s,\kappa}}{Q_{s,\kappa}} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}{1+V_{0,s,\kappa}/Q_{s,\kappa} \cdot tan(\pi \cdot f_{c_{s,\kappa}}/f_s) + tan(\pi \cdot f_{c_{s,\kappa}}/f_s)^2}$ |

Table 3.2: Coefficients calculated according to the Cut filter [57].

but, generally, the non-convex techniques are used due to the multiple local non-linear points. Parks-McClellan is one of the first convex optimization methods used for the FIR filter coefficients optimization [60]. Linear programming is used to achieve linear-phase FIR filters of minimum length using linear objectives related to linear constraints [61]. The peak-constrained least-squares (PCLS) approach [62] aims to minimize the error with a least-square approximation. Convex optimization methods are also used for IIR filter design. Linear Programming and Quadratic Programming [54] are used to design a recursive filter with a linear phase. Semidefinite Programming (SDP) is used to design a stable IIR filter [63].

With the convex techniques, the cost function could fall in a local minimum.. For this reason, some novel non-convex techniques have been studied, particularly in recent years. In [64], a $p$-norm minimization is used to design FIR filter coefficients. Joint Sparsity and Order Optimization is performed in [65] using as cost function the norm operation and the Alternating Direction Method of Multipliers (ADMM) algorithm [66]. The Piecewise Linear Con-

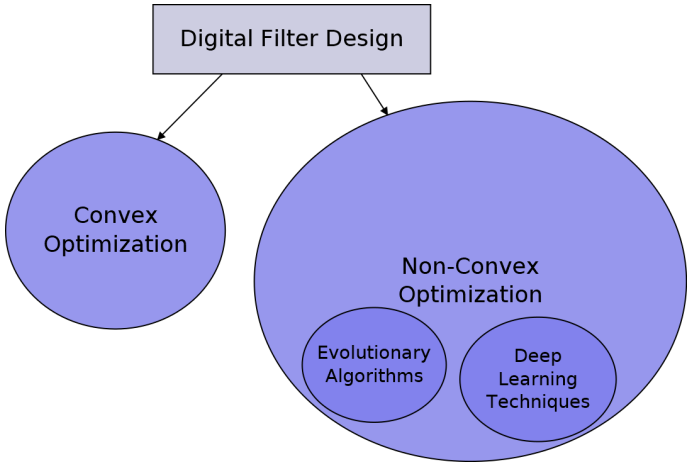



Figure 3.3: Diagram of the main optimization techniques for the design of digital filters

cave Optimization (PLCO) is used in the optimization problem to design the sparse FIR filters [67], solving non-linear and non-convex problems. IIR filter design is analyzed in [68], in which a Sequential Constrained Least-Square (SCLS) method with Steiglitz-McBride (SM) algorithm is used to achieve good performance concerning the only SM methods. Constraint Transcription and Filled Function methods are used in [69] to achieve a global minimum in the IIR filter design. An iterative second-order Cone Programming is used in [70] to design IIR filters, relaxing the non-convex problem. Finally, the Partial Fraction Decomposition Method is used for the optimal design of IIR filters [71], decomposing the transfer function into a sum of low order fractions to optimize poles and zeros, achieving better performance comparing it with the minmax-based methods.

Evolutionary algorithms are used for FIR and IIR filter design to overcome the limitations of linear and gradient based techniques. Indeed, evolutionary algorithms can be used to obtain non sub-optimal solutions from non-linear and multi-objective problems [51]. The common evolutionary algorithms used for IIR filter design are Genetic Algorithm [72], Particle Swarm Optimization [73] and Gravitational Search Algorithm [38]. Other evolutionary methods for IIR filter design are presented in [51]. The PSO is used for FIR filter design in [74], while in [75], the Flower Pollination Algorithm (FPA) is used to design FIR filter coefficients. The achieved filters approximate the desired specifications and give better performance than windowing and Parks-McClellan techniques, using several fitness functions. Opposition-based Harmony Search Algorithm is studied in [76] for the optimal design of FIR filter coefficient, achieving better performance than the Parks-McClellan method, PSO, Real-coded Genetic

Algorithm and Differential Evolution.

A Deep Learning approach is presented in [77], where a neural network is implemented to update IIR filter coefficients to achieve the desired frequency response and a stable filter. Another solution is presented in [78], where the optimization is performed assuming the IIR filter coefficients as the weights of the neural network. In [79], an adaptive FIR and IIR filter design are described using a MLP. A neural network approach for FIR filter design is presented in [80], in which the Deep Learning method is used with the Frequency Response Masking and gives the magnitude response at each frequency bin. In [81], a Radial Basis Function (RBF) is used to design FIR filter coefficients. In [82], the RBF is compared with the Back Propagation Neural Network (BPNN) and General Regression Neural Network (GRNN), optimizing low pass FIR filter coefficients and feeding the network with the cut-off frequency and a scale value, while in [83] the authors compared the neural approach with the windowing method, outperforming the performance according to the target low pass FIR filter frequency response.

# Chapter 4

# Multipoint Audio Equalization

Multipoint Audio Equalization improves the sound quality within a listening environment [84], constraining the system to obtain specific characteristics defined by the user, e.g. the frequency response, in which usually a flat band frequency response is desired within a frequency range [6]. In practice, different shapes are also used, such as amplifying low frequencies by some dB.

In Figure 4.1, the aspects and goals of an audio equalizer [85] are shown. The equalizer can be divided into a non-parametric and parametric equalizer, while the temporal decay control is an equalizer that does not invert a given frequency response but regularize the long decays in a listening room [86]. Parametric equalizers can be divided into AR or ARMA models [87, 88]. Audio equalizers can be divided into minimum-phase and mixed-phase: the first affect the minimum phase part of phase response, while the mixed-phase can also affect the excess phase [89]. Finally, some constraints are considered to design an equalizer, like the listener position, variation of room response, and psychoacoustic factors [85]. The operating frequency range is another feature to consider when designing an equalizer.

Two scenarios are considered to perform equalization [90]: the fixed case, where the equalization is performed without considering variations within the listening environment; the adaptive case, where variations are taken into account, i.e., the listener's movement, the number of people (or passengers inside the car cabin), or variations in temperature and humidity that can affect the frequency response [91]. Finally, an audio equalizer is designed according to a single position or multiple positions to be enhanced; in the latter case, several methods improve the sound quality on different points [92].

Many Multipoint Audio Equalization techniques have been described in literature [6, 84, 85]. In this Section, a brief description of the current Multipoint Audio Equalization techniques is described, focused in Section 4.2.1 the Steepest Descent method, Section 4.2.2 the Frequency Deconvolution (FD) method, and Section 4.3.1 the Direct Search Method (DSM), used for the Parametric IIR equalizer design [93].

The digital equalization problem started with [94], where a minimum phase


Figure 4.1: Diagram of Multipoint Audio Equalization aspects.

inverse filter is designed to remove the effect of the Room Impulse Response in a speech signal. A review of the main room response equalization algorithms are described in [84]. The authors classify the techniques in five classes: Homomorphic filtering, where the Room Impulse Response equalizer is achieved by the direct inversion of the minimum phase part [95], resulting in FIR filters with very long taps and a model that is very sensitive to impulse response; Linear Predictive Coding analysis [96], where the room response is modeled as a minimum-phase all-pole filter (estimating the common acoustical poles between the transfer functions) and the equalizer is a FIR filter. The limitation of this technique is that it can be only used for the minimum phase equalization; Frequency Domain Deconvolution [97, 98] is a method where the equalizer can be directly designed in the Discrete Fourier Transform domain by considering the reciprocal of the Room Response. This technique could introduce artefacts; thus, pre-processing techniques or a regularization parameter could be

used; Least Square Optimization methods are used for adaptive equalization [99, 100], but these techniques encounter some challenges such as the high sensitivity to the peaks and the notches in the room response and the non-uniform distribution of the error in the spectrum [84]; Multiple-Input/Multiple-Output Inverse Theorem (MINT) methods construct the inverse Room Impulse Response from multiple FIR filters, using multiple loudspeakers or microphones. When the number of speakers is increased, the MINT methods enhance the inversion of the room response [101].

Other equalization techniques, not properly just for the audio equalization, are described in [6]. The parametric equalizer is the most powerful and flexible [6] because it allows the correction of peaks or notches in a given frequency band using only three parameters: center frequency $f_c$, gain $G_0$ and quality factor $Q$ (or bandwidth $B_0$). A parametric equalizer can be created from first or second order shelving filters [102, 103] and second order peak or notch filters [104]. Most popular parametric equalizers are designed to achieve IIR filters, although some literature solutions implement parametric equalizers with FIR filters [105].

The graphic equalizer is a more closed-form equalizer than the parametric one [6]. It is commonly used in music production because it modifies sound effects, while parametric equalizer is used for the task of Multipoint Audio Equalization because the parameters to be modified are very practical. This technique allows controlling the gain of each filter set with a central frequency and a bandwidth a priori. Cascade [106] and parallel bank filter [107] structures have been studied, with both FIR and IIR filters implementations [108].

Pre-processing techniques are used to overcome the limitations of Multipoint Audio Equalization, such as the use of very long impulse responses of the equalizer (e.g. very long FIR filters) or the variation of the impulse response, which can cause a shift of several dB in the frequency response [84]. A solution could be to design short filters using a coarse model of the impulse response [109]. Other pre-processing techniques are based on the characteristics of the impulse responses and the human ear. Thus, the frequency response is more regular and insensitive to the position at low frequencies [84], like in [110], where the authors used the fractional octave-band smoothing to the transfer functions. Another technique is the frequency warping [111], that replaces the unit delay $z^{-1}$ with an all-pass filter [112]. Other similar solutions are the Kautz filters [113] and the discrete Kautz functions [114]. Multirate approaches [115] divide the spectrum into sub-bands. Each sub-band is down-sampled and later separately processed with filters of different lengths. Finally, the Room Impulse Response Reshaping [116] could be used to shorten the Impulse Response, achieving the desired time window and minimizing the undesired tails of the Impulse Response [117].

Machine Learning techniques have been investigated for the Single Input - Single Output (SISO) Audio Equalization task. The Nearest Neighbor pattern recognition is used to assist the user to automatically adjust the timbre according to the user preferences [118]. An End-to-End Convolutional Neural Network is investigated in [119] for music production, amplifying or attenuating audio content in a certain frequency region, while in [120], a Time Delay Neural Network is used as Inverse Room Response cascaded with the room acoustic scene. Although these techniques are very interesting, they cannot be used as audio equalizers as they have a very high computational cost.

A first attempt to exploit Machine Learning techniques for IIR filter design for Multipoint Audio Equalization in an automotive scenario is presented in [121]; in particular, GSA is used to optimize the coefficients of IIR filters arranged as a cascade of SOS's. Optimal IIR filter coefficients are based on a fitness function and on two alternative methods to avoid unstable filters. The evolutionary method is compared with the DSM, achieving superior results. Despite the good performance, some experiments have led to considerations about the design of fitness functions. In fact, in some experiments, one speaker had much higher energy than the other speakers, or denominator coefficients resulted in poles very close to the unit circle. These tests were discarded from the final evaluation and comparison.

## 4.1 Metrics

Multipoint Audio Equalization is analyzed using several metrics. For the FIR filter design, the results were provided in terms of Mean Square Error (MSE) and standard deviation $\sigma$ [122, 123]. The MSE of the magnitude response is calculated bin-by-bin for each microphone between the desired frequency response; after, the results are averaged between all microphones:

$$\overline{MSE} = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} \left( \frac{\sum_{\omega=\omega_l}^{\omega_h} \left( |\tilde{H}_m(\omega)| - |H_{des}(\omega)| \right)^2}{\omega_h - \omega_l} \right) \tag{4.1}$$

The average standard deviation $\overline{\sigma}$ is calculated as:

$$\overline{\sigma} = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} \sigma_m \tag{4.2}$$

where $\sigma_m$ is the standard deviation of $m$-th microphone:

$$\sigma_m = \sqrt{\frac{1}{\omega_h - \omega_l + 1} \sum_{\omega=\omega_l}^{\omega_h} (10 \cdot log_{10}|\tilde{H}_m(\omega)| - D)^2} \tag{4.3}$$

$$D = \frac{1}{\omega_h - \omega_l + 1} \sum_{\omega=\omega_l}^{\omega_h} (10 \cdot log_{10}|\tilde{H}_m(\omega)|) \tag{4.4}$$

Replacing H with the unfiltered frequency response gives the results without equalization [3].

In the case of Parametric IIR filter design, metrics are evaluated in the one-third octave band.

## 4.2 Multipoint Audio Equalization using FIR filter design

### 4.2.1 Steepest Descent

The Steepest Descent (SD) is an adaptive method for Multipoint Audio Equalization [124, 125] and it aims to equalize the impulse response in order to match a target impulse response:

$$h_{des} = [\underbrace{0 \quad \dots \quad 0 \quad 1 \quad 0 \quad \cdots \quad 0}_{L+\mathcal{T}-1}]^T \tag{4.5}$$

where $\mathcal{T}$ is the number of taps of the FIR filters and $L$ is the length of the impulse response.

The FIR filters are adapted to match the target impulse response:

$$\tilde{h}_m = h_{m,1} * g_1 + h_{m,2} * g_2 + \cdots + h_{m,S} * g_s = \sum_{s=1}^{\mathcal{S}} h_{m,s} * g_s \approx h_{des} \tag{4.6}$$

The optimization aim is to minimize the cost function:

$$J = ||h_{\mathcal{M}} - \tilde{h}||_2 \tag{4.7}$$

where $\tilde{h}$ is the vector containing the output impulse response $\tilde{h} = [\tilde{h}_1, \tilde{h}_2, \cdots, \tilde{h}_M]$ and $h_{\mathcal{M}}$ is the vector containing $\mathcal{M}$ times the target impulse response. The inverse system $g$ can be calculated by:

$$g = H^+ h_{\mathcal{M}} \tag{4.8}$$

where $H^+$ is the pseudo inverse of the system matrix $H$:

$$H = \begin{bmatrix} H_{1,1} & H_{1,2} & \cdots & H_{1,\mathcal{S}} \\ \vdots & \vdots & \vdots & \vdots \\ H_{\mathcal{M},1} & H_{\mathcal{M},2} & \cdots & H_{\mathcal{M},S} \end{bmatrix} \tag{4.9}$$

which its elements $H_{m,s}$ are $(L + \mathcal{T} - 1) \times \mathcal{T}$ circular matrices composed by the impulse responses $h_{m,s}$ [124]:

$$H_{m,s} = \begin{bmatrix} h_{m,s}(0) & 0 & \cdots & 0 \\ h_{m,s}(1) & h_{m,s}(0) & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ h_{m,s}(L-1) & \cdots & \vdots & \vdots \\ 0 & h_{m,s}(L-1) & \ddots & \vdots \\ 0 & \cdots & 0 & h_{m,s}(L-1) \end{bmatrix} \tag{4.10}$$

To adaptively calculate the FIR filters, the gradient of the cost function $\nabla J$ is determined:

$$\nabla J = -2H^T h_{\mathcal{M}} + 2H^T H g \tag{4.11}$$

thus the inverse system can be obtained by:

$$g(n+1) = g(n) - \mu \nabla J \tag{4.12}$$

where $\mu$ is the step-size.

### 4.2.2 Frequency Deconvolution

The Frequency Deconvolution method [97] is widely used for designing FIR filters for Multipoint Audio Equalization. This technique is based on deconvolution in the frequency domain and on the Fast Fourier Transform (FFT), achieving a matrix of causal FIR filters.

The FD is optimal in the Least Squares sense [97]. Thus the problem is expressed as a convex optimization problem where the cost function $J$ is given by:

$$J = e^H e + \beta v^H v \tag{4.13}$$

where $e^H e$ is the performance error term and measure the error between the desired and the reproduced signal at the microphones, while $\beta v^H v$ is the effort penalty term and $\beta$ is a regularization term.

The matrix of optimal filters in the frequency domain $G(k)$ is computed

according the following equation:

$$G(k) = [H^H(k)H(k) + \beta I]^{-1} H^H(k)A(k) \tag{4.14}$$

where $H(k)$ is the matrix of the transfer functions of the impulse responses and $A(k)$ is the transfer function of the desired signal. When $G(k)$ is computed, FIR filters are achieved by calculating the Inverse Fast Fourier Transform (IFFT) and performing a circular shift of $k/2$ samples, where $K$ is the FFT size.

### 4.2.3 Proposed Method

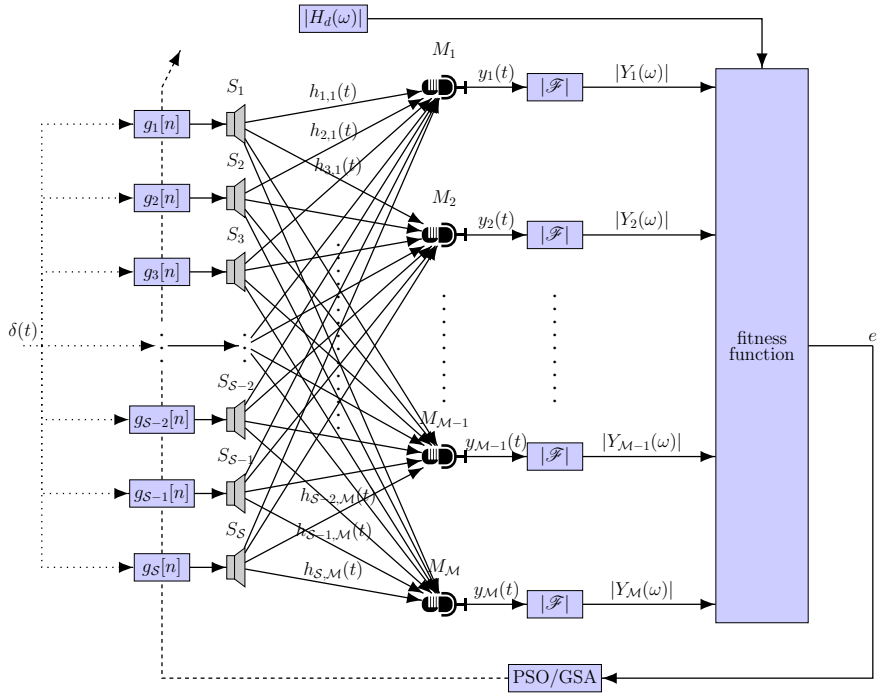



Figure 4.2: Scheme of PSO and GSA used for Multipoint Audio Equalization.

Several Machine Learning techniques were proposed for the Multipoint Audio Equalization task. The first ones implemented have been Evolutionary Algorithms, as they are widely used to design SISO case Audio Equalization [126] and IIR and FIR filters [51, 127]. Particularly, PSO and GSA have been compared for FIR filter design for Multipoint Audio Equalization [122]. The two evolutive algorithms explained in Sections 2.1.1 and 2.1.2, respectively, optimize FIR filter coefficients according to a cost function (see Figure 4.2), that is the Mean Square Error between the desired frequency response and the

achieved one by the simulation:

$$\bar{E}_{MSE} = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} \left( \sum_{\omega} \left( |\tilde{H}_m(\omega)| - |H_{des}(\omega)| \right)^2 \right) \tag{4.15}$$

Then, the $p_{best}$ for the PSO and the $best(k)$ for the GSA with their respective update procedure are chosen by the minimum MSE.

The main diagram of the Deep Optimization technique for Multipoint Audio Equalization is described in Figure 4.3: the neural network gives as output the optimized parameters needed for the simulation; the values resulting from the simulation, such as the frequency responses, will be used to calculate the loss function. The error is used to optimize network parameters through backpropagation. Section 4.3.2 describes the partial derivative from the loss functions to the output network parameters for the IIR Parametric filter design for Multipoint Audio Equalization.

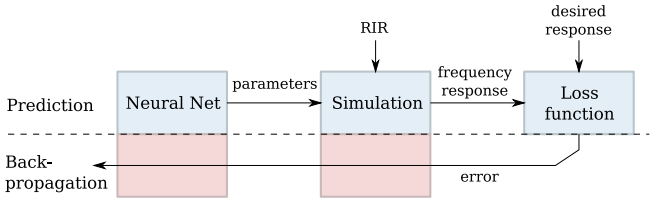



Figure 4.3: General scheme of Deep Optimization for FIR filter design for Multipoint Audio Equalization.

The first studies on Deep Optimization were carried out for the design of FIR filters for Multipoint Audio Equalization [123, 128]. The only available information is the measured impulse responses, which were used as input to the neural network. The first studied neural architecture is the GAN [128] (see Figure 4.4): the generator network is composed of convolutional layers and a stack of fully connected layers, giving the optimal FIR filter coefficients (thus a tensor of length $\mathcal{S} \times \mathcal{T}$). The discriminator compares the desired frequency response concerning the achieved one from the simulation. The input of the generator (and for the CNN and AE) consists of a 3D matrix that stacks all the measured impulse responses. It is a tensor of size $\mathcal{S} \times \mathcal{M} \times L$.

The discriminator network is designed to minimize a least squared loss, $L_D$, following [47], which is defined as:

$$\mathscr{L}_D = \frac{1}{2} \cdot \mathbb{E}_{|H_{des}(\omega)|}[(D(|H_{des}(\omega)|) - 1)^2] + \frac{1}{2} \cdot \mathbb{E}_{\tilde{h}}[(D(G(\tilde{h})))^2] \tag{4.16}$$

where $\mathbb{E}_{|H_{des}(\omega)|}[(D(|H_{des}(\omega)|) - 1)^2]$ is the expectation that discriminator clas-

sifies $|H_{des}(\omega)|$ as desired frequency response, while $\mathbb{E}_{\tilde{h}}[(D(G(\tilde{h})))]$ is the expectation that discriminator classifies the frequency responses achieved from the generator as not desired frequency responses.

The generator minimizes a different loss function, $L_G$, calculated as the sum of least squares error and the Euclidean distance [129] between the calculated frequency response $|\tilde{H}_m(\omega)|$ and the desired frequency response $|H_{des}(\omega)|$:

$$\mathscr{L}_G = \frac{1}{2} \cdot \left( \mathbb{E}_{\tilde{h}}[(D(G(\tilde{h})) - 1)^2] \right) + \lambda_G \cdot \left( \sum_{m=1}^{M} \left\| |\tilde{H}_m(\omega)| - |H_{des}(\omega)| \right\|_2 \right) \quad (4.17)$$

where $\lambda_G$ is a weight value that regulates the activity of the generator.

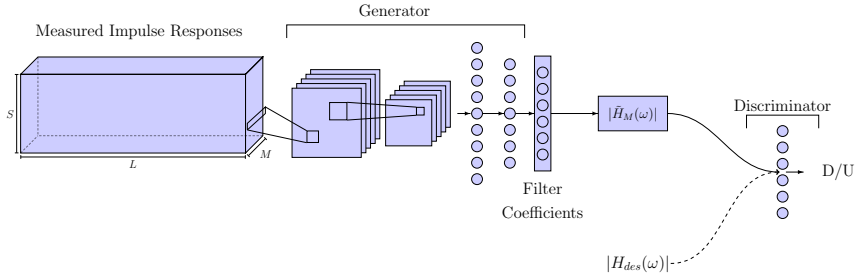



Figure 4.4: Generative Adversarial Network architecture used for FIR filters design for Multipoint Audio Equalization.

The Convolutional Neural Network is composed of a series of convolutional layers and a stack of fully connected layers (see Figure 4.5). The convolutional layers reduce the dimensionality of the input and extract features for the fully connected layers. The loss function is calculated as:

$$\mathscr{L}_C = \sum_{m=1}^{\mathcal{M}} \left\| |\tilde{H}_m(\omega)| - |H_{des}(\omega)| \right\|_2 . \quad (4.18)$$

The Multi-Layer Perceptron is a network composed of several fully connected layers. The input is a vector composed of concatenated impulse responses (see Figure 4.6), achieving a tensor of size $S \times \mathcal{M} \times L$. The loss function is the same as described in Equation 4.18.

The Autoencoder is a generative model [44] based on an encoder, a decoder and an internal representation that interconnects the two, called latent space. The encoder is composed of convolutional and fully connected layers (see Figure 4.7), similar to the CNN; the decoder performs the inverse mapping, thus is composed of fully connected and de-convolutional layers. Filter coefficients are optimized for the internal representation. Thus the latent space is composed of a vector of size $\mathcal{S} \times \mathcal{T}$.

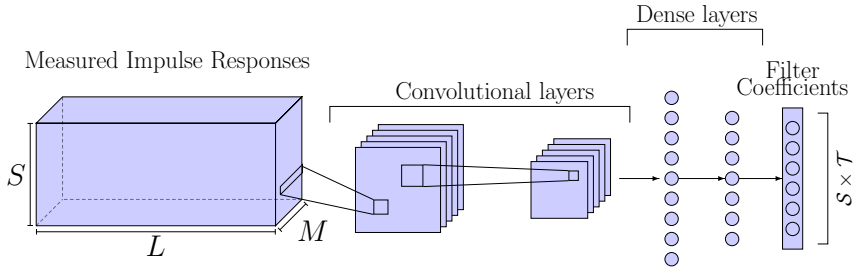



Figure 4.5: Convolutional Neural Network architecture used for FIR filters design for Multipoint Audio Equalization.

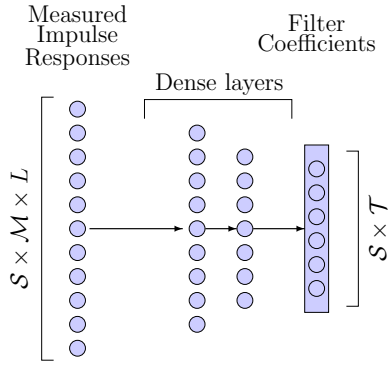



Figure 4.6: Multi-Layer Perceptron architecture used for FIR filters design for Multipoint Audio Equalization.

The loss function for the Autoencoder is given by the sum of the Euclidean distance (Equation (4.18)) and the reconstruction loss. The latter is expressed as the Euclidean distance between the input impulse response and the reconstructed one. The final loss for the Autoencoder is given by:

$$\mathscr{L}_{AE} = \alpha \cdot \sum_{m=1}^{\mathcal{M}} \left\| |\tilde{H}_m(\omega)| - |H_{des}(\omega)| \right\|_2 + (1-\alpha) \cdot \sum_{m=1}^{\mathcal{M}} \sum_{s=1}^{\mathcal{S}} \left\| \tilde{h}_{m,s}(n) - h_{m,s}(n) \right\|_2 \tag{4.19}$$

The term $\alpha$ allows to weight the two losses, set to 0.5.

### 4.2.4 Experimental Setup

Several experiments were performed for FIR filter design for Multipoint Audio Equalization. Two automotive scenarios were used; they are shown in Figure 4.8. The first is the Alfa Romeo Giulia, composed of 7 loudspeakers: four door woofers, one subwoofer in the trunk, one speaker in the center of the dashboard

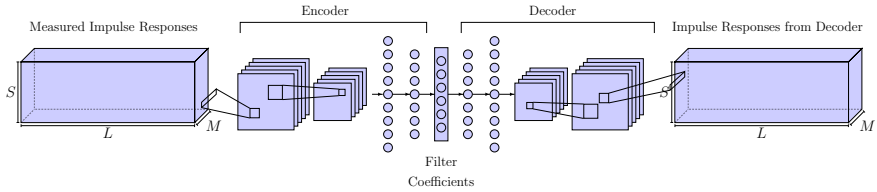
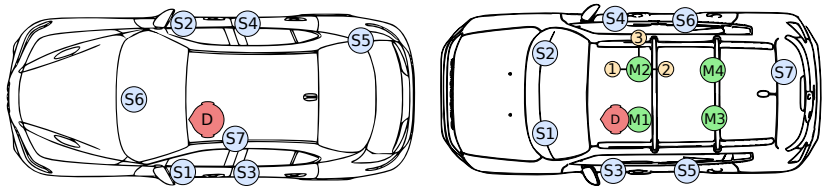



Figure 4.7: Auto-Encoder architecture used for FIR filters design for Multipoint Audio Equalization.

and one speaker in the driver's headrest. The second is the Jeep Renegade, used both for the analysis of binaural audio equalization, both for multipoint equalization. This scenario comprises 7 loudspeakers: a subwoofer in the trunk, two woofers in the front doors, two woofer-tweeters in the back doors and two tweeters




(a) (b)

Figure 4.8: Top view of the Alfa Romeo Giulia (a) and the Jeep Renegade (b) showing the placement of the $\mathcal{S}$ loudspeakers and the $\mathcal{M}$ microphones. D indicates the dummy head.

The binaural impulse responses in the Alfa Romeo Giulia were measured using the sine sweep method [130] implemented by the Aurora plugins. Sampling was 28.8 kHz with Roland Octa-Capture audio interface. Then the impulse responses were resampled to 48 kHz. A Kemar 45BA mannequin was placed on the driver's seat; the distance between its ears is 18 cm.

Regarding the impulse responses measurement inside the Jeep Renegade, several microphones were used: 4 omnidirectional microphones, one per seat (labelled M1, M2, M3 and M4 in Figure 4.8.b) needed for the optimization in the car cabin; 3 microphones in the proximity of the microphone M2 (labelled as PM1, PM2 and PM3 in Figure 4.8.b) used for the analysis of the head movements on the equalization performance, and finally a binaural mannequin mounted on the driver seat. The proximity microphones were placed at a distance of 6.5 cm forward, 6.5 cm backward and 22.5 cm lateral, respectively. The sine sweep method has been used, with a sampling rate of 48 kHz using an Audio Precision APX-586 analyzer and a Crown D-75A power amplifier to drive the loudspeakers.

The baseline methods have been implemented in *Matlab*, whereas the proposed methods have been implemented in *Python* using *Keras* with *Tensorflow* as backend. The experiments were performed using a machine with an Intel Core i7-4930K 3.40 GHz clock processor, 32 GB of RAM and Nvidia GTX-Titan GPU with 12 GB of dedicated RAM.

Preliminary experiments were conducted to determine the values for the training hyperparameters: a sufficiently high number of iterations allows the networks to converge to very low errors. The learning rate was set to $1 \cdot 10^{-3}$, $\lambda_G$ was set to 100.0. Adam optimizer is used with a decay equal to $3 \cdot 10^{-8}$. The SD has a number of iterations equal to 250,000, while the number of iterations for the neural proposed methods was 200,000.

Four convolutional layers configurations were generated randomly; they are applied to GAN, CNN and AE architecture. The first convolutional layer has kernels of size $\mathcal{M} \times 1$, the second, if present, has kernels of size $1 \times \mathcal{S}$. The fully connected layers following the convolutional layers; varied from 1 to 2. Four MLP architectures were derived from the hidden layers used in the convolutional architectures, adding the other three configurations. In Table 4.1 is reported all the neural network configurations used for the experiments.

CNN

| Configuration | Number of Kernels | Number of Units | Trainable Parameters |
|---|---|---|---|
| Conv #1 | [48, 24] | [10] | 7,481,943 |
| Conv #2 | [10, 5] | [100, 10] | 3,826,153 |
| Conv #3 | [100, 25] | [100, 100] | 12,483,433 |
| Conv #4 | [10] | [1000] | 3,825,863 |

MLP

| Configuration | Number of Units | Trainable Parameters |
|---|---|---|
| MLP #1 | [10] | 6,798,935 |
| MLP #2 | [100, 10] | 67,280,035 |
| MLP #3 | [100, 100] | 67,934,875 |
| MLP #4 | [1000] | 679,183,175 |
| MLP #5 | [100] | 67,924,775 |
| MLP #6 | [100,100,100] | 67,944,975 |
| MLP #7 | [5] | 36,003,713 |
| MLP #8 | [10,1000,1000] | 14,914,185 |

Table 4.1: The CNN and MLP configurations used in the experiments. The number of parameters are referred to filters of 1024-th order.

The PSO algorithm have been analyzed with different hyperparameters. A search of hyperparameters $W_{max}$, $W_{min}$, $c_1$ and $c_2$ has been performed in the following ranges: $0.01 < W_{max} < 10.0$, $0.0001 < W_{min} < 0.1$, $2 \cdot 10^{-6} <$

$c_1, c_2 < 2$. The inertia weight $W$ is calculated after every iteration as:

$$W = W_{max} - (W_{max} - W_{min}) \cdot n/N \tag{4.20}$$

where $N$ is the total number of iterations, and $k$ is the current iteration. The algorithm stops when more than 500 iterations expire without an improvement of $g_{best}$, with a maximum number of iterations equals 2,000.

The hyperparameters for the GSA algorithm are: $\mathcal{G}_{0_{max}}$, $\mathcal{G}_{0_{min}}$, $A$ and $k_{best}$. $A_0 = A$ because the number of agents is low. The gravitational constant $\mathcal{G}(n)$ decreases linearly starting from $\mathcal{G}_{0_{max}}$ up to $\mathcal{G}_{0_{min}}$:

$$\mathcal{G}(n) = \mathcal{G}_{0_{max}} - (\mathcal{G}_{0_{max}} - \mathcal{G}_{0_{min}}) \cdot n/N \tag{4.21}$$

The two evolutionary algorithms were implemented in *Tensorflow*. In [122] PSO and GSA were compared in time- and frequency-domain with FIR filter length of 1024 samples, then the degradation is evaluated when they optimize reduced FIR length (512, 640, 768, 896 and 1024). This dissertation presents the best results achieved in the time domain and compares them with neural approaches and baseline methods.

### 4.2.5 Results

The first experiments were performed using the Alfa Romeo Giulia scenario: the algorithms optimize FIR filters coefficients from 512-th order to 1024-th order. In Table 4.2, the results are shown: the proposed neural network methods excepts of MLP outperform evolutionary algorithms and baseline techniques. The MLP does not reach the same performance as the FD and SD. The CNN achieves slightly better results than the convolutional AE and the GAN architecture, despite being simpler in implementation and computational cost.

The best overall results have been achieved using the CNN with FIR filters of 1024-th order. Despite shorter filters designed by the convolutional methods present slight performance degradation, the $\overline{MSE}$ remains very low.

Regarding the evolutionary algorithms, the GSA achieved better performance than PSO, designing FIR filters of 768-th order (PSO gets the best performance with 640-th order). The $\overline{MSE}$ is equal to 0.13 and $\overline{\sigma}$ is 2.07.

Analyzing the results with the baseline methods, GSA achieved better performance than FD and SD, but it does not come close to the results achieved with convolutional networks (see Table 4.2).

Magnitude frequency responses at the dummy head left and right microphone are shown in Figure 4.9: the green line represents the non-equalized frequency response, whereas the blue line corresponds to the equalized one. The CNN FIR filters correct the frequency responses achieving a flat magnitude (see Figures

| | Filter Order | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 512 | | 640 | | 768 | | 896 | | 1024 | |
| Method | $\overline{MSE}$ | $\overline{\sigma}$ | $\overline{MSE}$ | $\overline{\sigma}$ | $\overline{MSE}$ | $\overline{\sigma}$ | $\overline{MSE}$ | $\overline{\sigma}$ | $\overline{MSE}$ | $\overline{\sigma}$ |
| MLP | 0.32 | 2.88 | 0.36 | 2.73 | 0.46 | 2.80 | 0.45 | 2.80 | 0.32 | 2.75 |
| GAN | $9.89{\cdot}10^{-4}$ | 0.14 | $3.88{\cdot}10^{-4}$ | 0.09 | $2.21{\cdot}10^{-4}$ | 0.06 | $1.06{\cdot}10^{-4}$ | 0.04 | $6.95{\cdot}10^{-4}$ | 0.04 |
| AE | $9.72{\cdot}10^{-4}$ | 0.14 | $3.80{\cdot}10^{-4}$ | 0.08 | $1.66{\cdot}10^{-4}$ | 0.06 | $1.07{\cdot}10^{-4}$ | 0.04 | $6.85{\cdot}10^{-5}$ | 0.03 |
| CNN | $7.90{\cdot}10^{-4}$ | 0.12 | $3.74{\cdot}10^{-4}$ | 0.08 | $1.79{\cdot}10^{-4}$ | 0.06 | $1.02{\cdot}10^{-4}$ | 0.04 | $6.31{\cdot}10^{-5}$ | 0.04 |
| PSO | 0.21 | 2.70 | 0.21 | 2.67 | 0.22 | 2.74 | 0.22 | 2.65 | 0.22 | 2.67 |
| GSA | 0.14 | 2.23 | 0.14 | 2.24 | 0.13 | 2.07 | 0.14 | 2.14 | 0.13 | 2.18 |
| FD | 0.18 | 2.52 | 0.15 | 2.34 | 0.14 | 2.23 | 0.12 | 2.07 | 0.10 | 1.93 |
| SD | 0.98 | 7.26 | 0.98 | 7.13 | 0.99 | 7.09 | 0.99 | 7.091 | 1.03 | 6.39 |

Table 4.2: Multipoint Audio Equalization results for the Alfa Romeo Giulia with binaural microphones. Please note that the $\overline{MSE}$ in absence of equalization is 2.19, with $\overline{\sigma}$ 3.52.

4.9.a and 4.9.b). No relevant peaks or notches are present in the equalized frequency responses. GSA FIR filters achieved magnitude responses close to the desired frequency responses but with some notches. The FD method achieves a rather flat spectrum, but peaks and notches are still visible. The SD presents the higher $\overline{MSE}$, while the $\overline{\sigma}$ is lower than the FD, this is because the frequency responses present fewer peaks, but the magnitude response is biased and sits below 0 dB. The same conclusions happen for the other FIR filter orders.

The performance of FD is dependent on the $\beta$ parameter since it avoids excessive gain in the inverse filter or avoids equalization at all. Some $\beta$ values have been tested using FIR filters of order 1024. In Table 4.3 the $\overline{MSE}$ and the $\overline{\sigma}$ are reported with the respective $\beta$ value and two frequency-dependent $\beta$used for the FD method. With a lower value of $\beta$, the inversion should get closer to the ideal, thus reaching a low $\overline{MSE}$. With a larger value of $\beta$, the performance decreases as expected. Some frequency-dependent configurations of $\beta$ achieved good results. The V-shaped can reduce the $\overline{MSE}$ by a tiny amount, but no significant improvement can be found by using a frequency-dependent $\beta$; thus, the choice of $\beta$ does not improve the performance.

### Jeep Renegade

The Jeep Renegade scenario was used to compare the CNN and the FD results since they are the best of the proposed methods and the two baseline techniques, respectively. To increase the complexity of the problem, the number of microphones to be optimized was increased. Binaural experiments and the four-seat equalization experiment were performed. In Table 4.4 the results of the FIR filter of 1024-th order are shown. The CNN achieved the best results

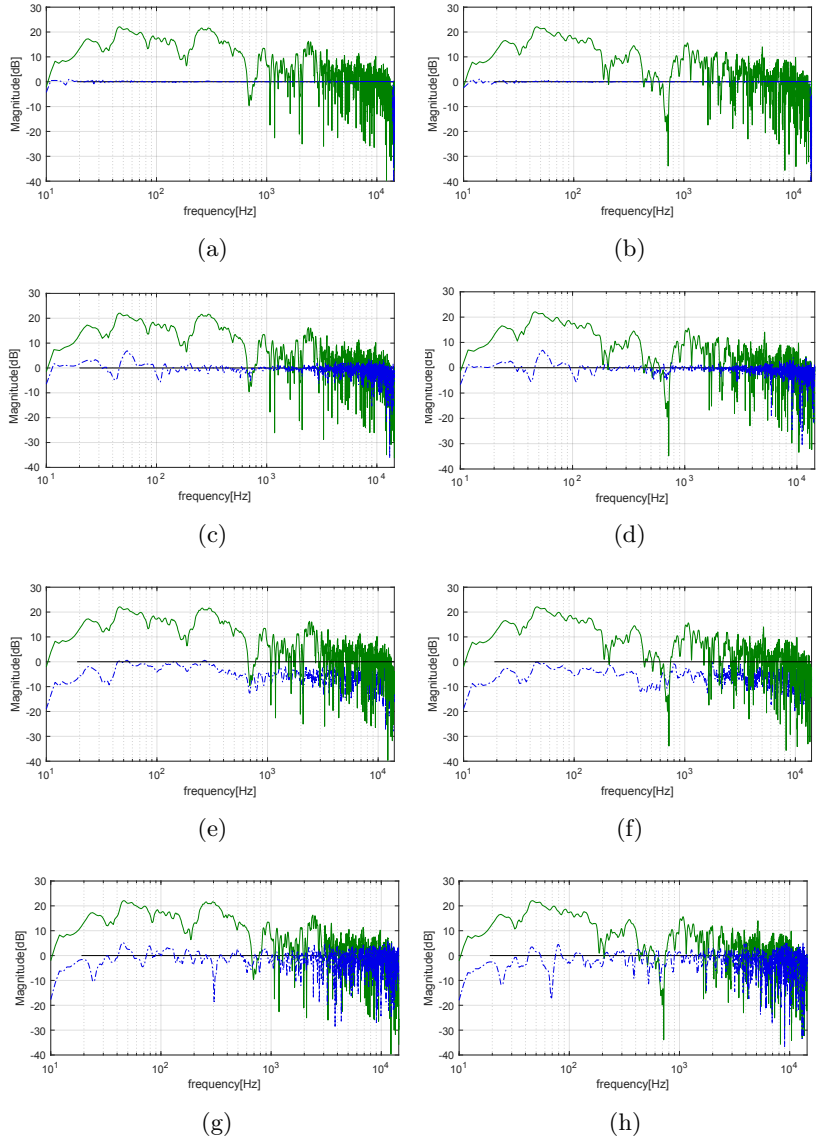



Figure 4.9: Magnitude frequency responses at the left and right microphones of the dummy head in the Alfa Romeo Giulia after applying filters obtained from the CNN (a, b), Frequency Deconvolution (c, d), Steepest Descent (e, f), PSO (e, f) and GSA (g,h) methods. The original magnitude frequency response is shown in green while the equalized frequency response is shown in blue. The target magnitude response is shown in black.

| $\beta$ | $\overline{MSE}$ | $\overline{\sigma}$ |
|---|---|---|
| $10^{-4}$ | 0.123 | 1.83 |
| $10^{-3}$ | 0.118 | 1.82 |
| $10^{-2}$ | 0.108 | 1.81 |
| $10^{-1}$ | 0.108 | 1.93 |
| 1 | 0.281 | 2.71 |
| 10 | 0.686 | 4.2 |
| 100 | 0.937 | 5.09 |
| V-shaped | **0.101** | **1.829** |
| U-shaped | 0.124 | 1.86 |

Table 4.3: Performance when parameter $\beta$ varies. The V-shaped configuration refers to a frequency-dependent $\beta$ with a minimum of $10^{-4}$ at 1 kHz and a maximum of $10^{-1}$ at DC and Nyquist, varying linearly on a dB scale. The U-shaped configuration takes a value of $10^{-4}$ in the range 100 Hz-10 kHz and 1 elsewhere.

for both the experiments, reaching a $\overline{MSE}$ equal to $6.19{\cdot}10^{-5}$ and $5.7{\cdot}10^{-4}$ for binaural and multipoint equalization, respectively. In the multipoint experiment, as expected, the results decrease, but they are still low. Regarding the FD method, a slight degradation of the performance is found for the 4-seats equalization. In conclusion, despite the performance degradation, results are still superior to the state-of-the-art method, even in the multipoint scenario.

| Setup | CNN | | | FD $\beta = 0.1$ | |
|---|---|---|---|---|---|
| | Conf | $\overline{MSE}$ | $\overline{\sigma}$ | $\overline{MSE}$ | $\overline{\sigma}$ |
| Binaural | #1 | $6.19 \cdot 10^{-5}$ | 0.035 | 0.05 | 1.21 |
| 4 seats | #1 | $5.7 \cdot 10^{-4}$ | 0.106 | 0.15 | 1.95 |

Table 4.4: Multipoint Audio Equalization results for the Jeep Renegade with binaural microphones and four microphones (one per seat). The FIR order is 1024.

To assess the validity of the proposed approach in response the small and large head movements, the frequency responses of three additional points were analyzed. The listening points are labelled as PM1 and PM2 (small head movement) and PM3 (large head movement). Comparing the results with the microphone M2, for reference, the error tends to rise for high frequencies, for

which the wavelength is short or the same order of magnitude as the distance between microphone M2; however, at low frequencies, the response is almost flat, as shown in Figure 4.10.

| Mic. | CNN | | FD | |
|---|---|---|---|---|
| | $\overline{MSE}$ | $\overline{\sigma}$ | $\overline{MSE}$ | $\overline{\sigma}$ |
| M2 | $5.07 \cdot 10^{-4}$ | 0.10 | 0.14 | 1.82 |
| PM1 | 0.61 | 2.88 | 1.2 | 2.9 |
| PM2 | 0.50 | 3.31 | 0.57 | 3.07 |
| PM3 | 0.80 | 3.09 | 0.84 | 3.12 |

Table 4.5: Multipoint Audio Equalization results for microphone M2 and microphones PM1, PM2 and PM3. The evaluation is achieved by the experiments performed using the Jeep Renegade with four microphones (see Table 4.4).

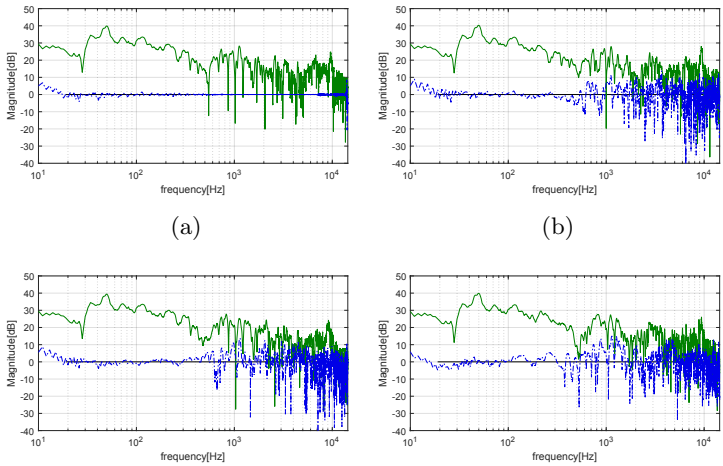



Figure 4.10: Frequency response at microphone M2 (a); microphones PM1 and PM2 (b,c), corresponding to small forward and backward head movements; microphones PM3 (d), corresponding to a large lateral head movement.

### Sensitivity to the Input

Since Deep Learning techniques as optimizing algorithms are uncommon in the signal processing literature, some experiments were performed using differ-

ent inputs to improve the understanding of the input to the neural networks, specifically, to assess the role of the input in driving the optimization process.

The input matrix is filled with: (a) random values changing at each iteration; (b) random values constant for all the training; (c) all ones; (d) all zeros. The same matrix size of the previous experiments is used for the experiments, leaving the input layers and the number of trainable parameters unchanged. The CNNs optimized FIR filters of order 1024 in the Alfa Romeo Giulia scenario. Results are shown in Table 4.6. In case (a), the performance is similar to the FD method but worse than the proposed method. Case (b) and (c) achieved results closer to the neural approaches, but they have not achieved the best results for the test. Finally, with null matrix, the coefficients are zero, making this method unsuitable to the optimization process.

It seems that the overall method can gain some advantage from the use of the measured impulse responses as input features; however, the network is able to design FIR filters even with non-informative input content, gaining information about the problem setup from the loss, where the impulse responses are employed to calculate the distance. These conclusions were helpful to implement a new kind of neural architecture, defined in Section 4.3.2 and used for the design of parametric IIR filters.

| Input | $\overline{MSE}$ | $\overline{\sigma}$ | Conf. |
|---|---|---|---|
| Impulse Responses | $\mathbf{6.31 \cdot 10^{-5}}$ | **0.034** | Conv #1 |
| Random Iteration | 0.14 | 2.152 | Conv #1 |
| Random Fixed | $1.35 \cdot 10^{-4}$ | 0.052 | Conv #1 |
| All 1s | $1.17 \cdot 10^{-4}$ | 0.049 | Conv #1 |
| All 0s | ill-conditioned | | |

Table 4.6: Effect of the input type on the results of the CNN (filter order 1024). The Table reports the best results.

#### Over-Determined Case

Further experiments were performed with the number of filters equal to or smaller than the number of microphones (over-determined case $\mathcal{M} < \mathcal{S}$), selecting a subset of the available impulse responses, thus simulating the presence of a lower number of speakers.

The results are reported in Table 4.7. The CNN achieved better performance than the FD method, meaning that non-convex optimization techniques can improve the optimal solution in the least-squares sense. The performance degradation from the $1 \times 1$ to the $2 \times 1$ case is extremely low, suggesting that the two impulse responses are quite similar.

| Car | Setup | CNN | | FD | |
|---|---|---|---|---|---|
| | | $\overline{MSE}$ | $\overline{\sigma}$ | $\overline{MSE}$ | $\overline{\sigma}$ |
| Giulia | $1 \times 1$ | 0.52 | 8.57 | 0.62 | 9.84 |
| | $2 \times 1$ | 0.57 | 7.81 | 0.64 | 9.19 |
| Renegade | $1 \times 1$ | 0.03 | 1.34 | 0.12 | 2.01 |
| | $4 \times 1$ | 0.22 | 2.76 | 0.44 | 3.62 |

Table 4.7: Results in the single-channel and over-determined audio equalization cases. Setup is $\mathcal{M} \times \mathcal{S}$.

### FIR Filter Remarks

The proposed filter design technique requires a complete training of the network for the design procedure. However, despite a large number of iterations, the loss exponentially decays. In the Alfa Romeo Giulia, 1024-th order FIR filters CNN experiments, the MSE decays below $1 \cdot 10^{-4}$ after 4200 iterations. Thus, it is possible to set the desired error threshold and stop the training as soon as it is reached.

The FD method provides symmetrical filters, thus linear phase frequency responses, while the SD does not. The proposed method does not constraint the phase, however, the phase response of FIR filters is almost linear. In Figure 4.11 is presented the comparison between a linear phase response and the averaged overall filters generated in the 1024-th order CNN case from Table 4.2.

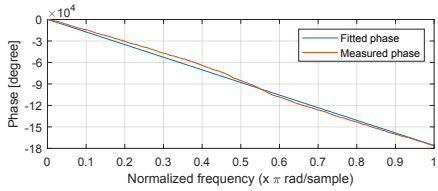



Figure 4.11: Phase response of one of the filters achieved with the CNN method (FIR order 1024) and a linear fitting. Frequency is normalized according to Nyquist.

A downside of the proposed method is as follows: energy along the FIR coefficients is spreading, as shown in Figure 4.12, where a FIR filter with the GAN is shown. The lack of damping of the filters impulse response can produce smearing of input transients.

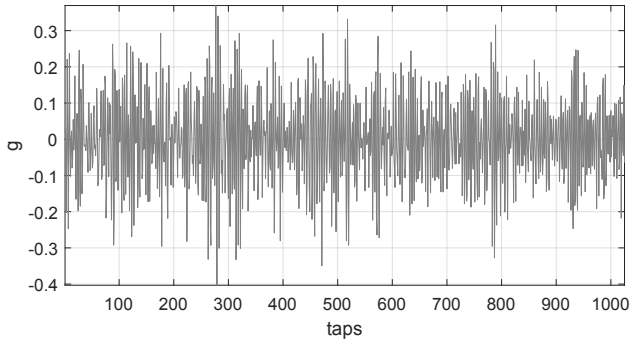



Figure 4.12: Sample of FIR filter obtained with the best GAN configuration.

## 4.3 Multipoint Audio Equalization using IIR filter design

### 4.3.1 Direct Search Method for IIR Parametric Equalizer

The Direct Search Method is a free-derivative, commonly used method for optimization problems [131]. The main features of this algorithm are its ease of implementation, which makes it widely used, and it can work with no constraints [132]. The DSM is used in [93] to optimize an IIR Parametric equalizer.

The algorithm is described as follows [133]: a generic parameter vector $c$ is varied by a small quantity:

$$\hat{c}_i = c_i \cdot (1 + \Gamma) \tag{4.22}$$

where $\Gamma$ is a random variation in the range $-\gamma \leq \Gamma \leq \gamma$. If the new parameters achieve a better cost function, they are kept, otherwise, they are rejected and another random variation $\Gamma$ is performed. The process continues until the demands have been met.

### 4.3.2 Proposed method

Continuing the studies on Deep Optimization networks, the problem of Multipoint Audio Equalization with parametric IIR filters was also tackled, arriving at the solution proposed in Figure 4.13: the BiasNet gives the parameters of the IIR Parametric filters, which will be used to calculate the coefficients of the filters and thus for the frequency convolution. Finally, the loss function calculation is performed, and the update of parameters using an optimizer (like the Adam) through the backpropagation is executed.

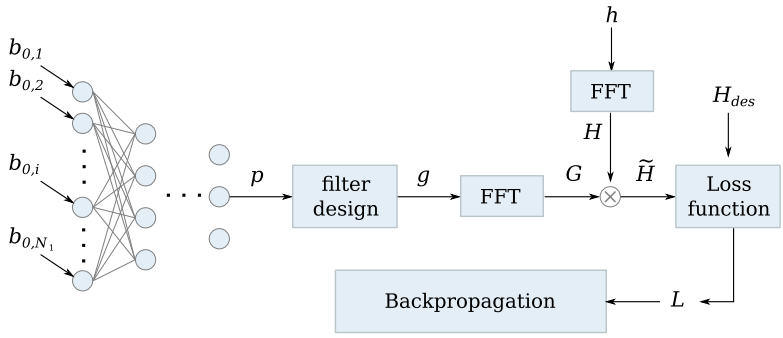



Figure 4.13: BiasNet architecture for Parametric IIR filter design for Multipoint Audio Equalization.

The BiasNet output parameters are normalized between the range $[-1, 1]$. The denormalization is performed to achieve the real parameter value; for example, the central frequency response is calculated as:

$$f_c = \frac{f_{c,max} - f_{c,min}}{2} \cdot p_{f_c,\kappa} + \frac{f_{c,max} + f_{c,min}}{2} \tag{4.23}$$

where $f_{c,max}$ and $f_{c,min}$ are the maximum and minimum allowed values for the center frequency. These values can be devised, e.g., according to one-third octave bands or any other subdivision of the audio range. This subdivision constraints the number of SOS's to the number of frequency bands, avoiding the overlap of the filters operative bandwidth, which, in turn, may result in excessive gains for some bands. Furthermore, mapping the range $[-1, 1]$ to a narrow portion of the spectrum reduces the prediction error of the $f_c$. The other parameters are denormalized mapping $[-1, 1]$ to their full range, which can be defined according to the application. Gains are designed by the network on a dB scale and are then converted into linear values when computing the IIR biquad equations of Table 3.1 and 3.2. From the rest of the dissertation, gains in dB scale will be denoted as $V_{0,dB}, V_{s,dB}$ to avoid confusion with their linear counterparts $V_0, V_s$.

The loss used in this work is a combination of a spectral loss $\mathscr{L}_1$, and a multichannel energy regularization term $\mathscr{L}_2$:

$$\mathscr{L} = \gamma_1 \cdot \mathscr{L}_1 + \gamma_2 \cdot \mathscr{L}_2 \tag{4.24}$$

with $\gamma_1 = 1$ and $\gamma_2 = log_2(\mathcal{S}) + log_2(\mathcal{M})$ being weights.

The loss function $\mathscr{L}_1$ is given by the Euclidean distance between the simu-

lated magnitude response and the desired magnitude response $|H_{des}(k)|$:

$$\mathscr{L}_1 = \sum_{m=1}^{\mathcal{M}} \sqrt{\sum_k (|\tilde{H}_m(k)| - |H_{des}(k)|)^2} \tag{4.25}$$

where: $\tilde{H}_m(k)$ is the Discrete Fourier Transform of the equalized impulse response at the m-th microphone $\tilde{h}_m(n)$.

The regularization term $\mathscr{L}_2$ is required when $\mathcal{S} > 1$ to keep the original energy balance between a reference speaker and each other speaker, avoiding unwanted change in spatial perception. The term is defined as:

$$\mathscr{L}_2 = \sum_{m=1}^{\mathcal{M}} \sqrt{\sum_{s=1}^{\mathcal{S}} (\hat{r}_{s,m} - r_{s,m})^2} \tag{4.26}$$

where $r_{s,m}$ and $\tilde{r}_{s,m}$ are the ratios between the energy of a reference speaker and the $s$-th speaker before and after equalization, respectively.

### Backpropagation of Multipoint Audio Equalization problem

In this Section, the mathematical expressions of the derivability of the loss function when the BiasNet is used for the Parametric IIR filter design for Multipoint Audio Equalization are described.

Backpropagation is performed calculating the partial derivative of the loss function with respect to the control parameters $\partial\mathscr{L}/\partial f_c$, $\partial\mathscr{L}/\partial Q$, $\partial\mathscr{L}/\partial V_{0,dB}$ and $\partial\mathscr{L}/\partial V_{s,dB}$. The partial derivatives are calculated as the product of cascaded local ones.

The first operations consist in calculating the partial derivatives with respect to a generic equalized impulse response:

$$\frac{\partial\mathscr{L}}{\partial\tilde{h}_{s,m}} = \frac{\partial\mathscr{L}}{\partial\mathscr{L}_1} \cdot \frac{\partial\mathscr{L}}{\partial\tilde{h}_{s,m}} + \frac{\partial\mathscr{L}L}{\partial\mathscr{L}_2} \cdot \frac{\partial\mathscr{L}_2}{\partial\tilde{h}_{s,m}} \tag{4.27}$$

where $\frac{\partial\mathscr{L}}{\partial\mathscr{L}_1} = \gamma_1$ and $\frac{\partial\mathscr{L}}{\partial\mathscr{L}_1} = \gamma_2$.

The partial derivative $\partial\mathscr{L}/\partial\tilde{h}_{s,m}$ is given by the product of:

$$\frac{\partial\mathscr{L}}{\partial\tilde{h}_{s,m}} = \frac{\partial\mathscr{L}}{\partial\tilde{h}_m} \cdot \frac{\partial\tilde{h}_m}{\partial\tilde{h}_{s,m}} \tag{4.28}$$

At the listening point $\tilde{h}_{s,m} = \sum_{s=1}^{\mathcal{S}} \tilde{h}_{s,m}(n)$, thus the local derivative $\frac{\partial\tilde{h}_m}{\partial\tilde{h}_{s,m}} = 1$ and $\frac{\partial\mathscr{L}}{\partial\tilde{h}_{s,m}} = \frac{\partial\mathscr{L}}{\partial\tilde{h}_m}$ . The filtered room response is computed in frequency

| IIR Partial Derivative | Boost case |
|---|---|
| $\frac{\partial b_{s,\kappa,0}}{\partial f_{c_{s,\kappa}}}$ | $\frac{\pi\cdot(\frac{1}{Q_{s,\kappa}}-\frac{10^{G_{s,\kappa}/20}}{Q_{s,\kappa}})(tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})-1)\cdot\frac{1}{cos^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})}}{f_s\cdot(\frac{1}{Q_{s,\kappa}}tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})+tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)^2}$ |
| $\frac{\partial b_{s,\kappa,1}}{\partial f_{c_{s,\kappa}}}$ | $\frac{\pi\cdot\frac{1}{cos^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})}\cdot\left(2\cdot\frac{1}{Q_{s,\kappa}}\cdot tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+\frac{1}{Q_{s,\kappa}}+6\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})\right)}{f_s\cdot(\frac{1}{Q_{s,\kappa}}tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})+tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)^2}$ |
| $\frac{\partial b_{s,\kappa,2}}{\partial f_{c_{s,\kappa}}}$ | $\frac{\pi\cdot(\frac{1}{Q_{s,\kappa}}+\frac{10^{G_{s,\kappa}/20}}{Q_{s,\kappa}})\cdot\frac{1}{cos^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})}\cdot\left(tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})-1\right)}{f_s\cdot(\frac{1}{Q_{s,\kappa}}tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})+tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)^2}$ |
| $\frac{\partial a_{s,\kappa,0}}{\partial f_{c_{s,\kappa}}}$ | 0 |
| $\frac{\partial a_{s,\kappa,1}}{\partial f_{c_{s,\kappa}}}$ | $\frac{\pi\cdot\frac{1}{cos^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})}\cdot\left(2\cdot\frac{1}{Q_{s,\kappa}}\cdot tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+\frac{1}{Q_{s,\kappa}}+6\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})\right)}{f_s\cdot(\frac{1}{Q_{s,\kappa}}tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})+tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)^2}$ |
| $\frac{\partial a_{s,\kappa,2}}{\partial f_{c_{s,\kappa}}}$ | $\frac{2\cdot\pi\frac{1}{Q_{s,\kappa}}\cdot\frac{1}{cos^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})}\cdot\left(tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})-1\right)}{f_s\cdot(\frac{1}{Q_{s,\kappa}}tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})+tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)^2}$ |
| | Cut Case |
| $\frac{\partial b_{s,\kappa,0}}{\partial f_{c_{s,\kappa}}}$ | $\frac{Q_{s,\kappa}\cdot\pi\cdot(10^{G_{s,\kappa}/20}-1)(tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})-1)\cdot(tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)\cdot\frac{1}{cos^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})}}{f_s\cdot(Q_{s,\kappa}tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+Q_{s,\kappa}+10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s}))^2}$ |
| $\frac{\partial b_{s,\kappa,1}}{\partial f_{c_{s,\kappa}}}$ | $\frac{2\cdot Q_{s,\kappa}\cdot\pi\cdot(6\cdot Q_{s,\kappa}\cdot sin(2\cdot\pi\cdot\frac{f_{c_{s,\kappa}}}{f_s})-10^{G_{s,\kappa}/20}\cdot(cos(2\cdot\pi\cdot\frac{f_{c_{s,\kappa}}}{f_s})-3)}{f_s\cdot(2\cdot Q_{s,\kappa}+10^{G_{s,\kappa}}\cdot sin(2\cdot\pi\cdot\frac{f_{c_{s,\kappa}}}{f_s}))^2}$ |
| $\frac{\partial b_{s,\kappa,2}}{\partial f_{c_{s,\kappa}}}$ | $-\frac{4Q_{s,\kappa}\pi(10^{G_{s,\kappa}+1})\cdot cos(2\pi\frac{f_{c_{s,\kappa}}}{f_s})}{f_s\cdot(2Q_{s,\kappa}+10^{G_{s,\kappa}}cos(2\pi\frac{f_{c_{s,\kappa}}}{f_s}))^2}$ |
| $\frac{\partial a_{s,\kappa,0}}{\partial f_{c_{s,\kappa}}}$ | 0 |
| $\frac{\partial a_{s,\kappa,1}}{\partial f_{c_{s,\kappa}}}$ | $\frac{2\cdot Q_{s,\kappa}\cdot\pi\cdot(6\cdot Q_{s,\kappa}\cdot sin(2\cdot\pi\cdot\frac{f_{c_{s,\kappa}}}{f_s})-10^{G_{s,\kappa}/20}\cdot(cos(2\cdot\pi\cdot\frac{f_{c_{s,\kappa}}}{f_s})-3)}{f_s\cdot(2\cdot Q_{s,\kappa}+10^{G_{s,\kappa}}\cdot sin(2\cdot\pi\cdot\frac{f_{c_{s,\kappa}}}{f_s}))^2}$ |
| $\frac{\partial a_{s,\kappa,2}}{\partial f_{c_{s,\kappa}}}$ | $\frac{2\cdot\pi\cdot\frac{10^{G_{s,\kappa}/20}}{Q_{s,\kappa}}(tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})-1)\cdot\frac{1}{cos^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})}}{f_s\cdot(\frac{10^{G_{s,\kappa}/20}}{Q_{s,\kappa}}tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})+tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)^2}$ |

Table 4.8: Local derivative of coefficients with respect to the central frequency, for a generic SOS.

| IIR Partial Derivative | Boost case |
|---|---|
| $\frac{\partial b_{s,\kappa,0}}{\partial G_{s,\kappa}}$ | $\frac{\frac{1}{Q_{s,\kappa}} ln(10) \cdot 10^{G_{s,\kappa}/20} \cdot tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})}{\frac{1}{Q_{s,\kappa}} \cdot 20 tan(\pi \frac{f_{c_{s,\kappa}}}{f_s}) + 20 \cdot tan^2(\pi \frac{f_{c_{s,\kappa}}}{f_s})}$ |
| $\frac{\partial b_{s,\kappa,1}}{\partial G_{s,\kappa}}$ | 0 |
| $\frac{\partial b_{s,\kappa,2}}{\partial G_{s,\kappa}}$ | $-\frac{\frac{1}{Q_{s,\kappa}} ln(10) \cdot 10^{G_{s,\kappa}/20} \cdot tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})}{\frac{1}{Q_{s,\kappa}} \cdot 20 tan(\pi \frac{f_{c_{s,\kappa}}}{f_s}) + 20 \cdot tan^2(\pi \frac{f_{c_{s,\kappa}}}{f_s})}$ |
| $\frac{\partial a_{s,\kappa,0}}{\partial G_{s,\kappa}}$ | 0 |
| $\frac{\partial a_{s,\kappa,1}}{\partial G_{s,\kappa}}$ | 0 |
| $\frac{\partial a_{s,\kappa,2}}{\partial G_{s,\kappa}}$ | 0 |
| | **Cut Case** |
| $\frac{\partial b_{s,\kappa,0}}{\partial G_{s,\kappa}}$ | $-\frac{log(10) \cdot 10^{G_{s,\kappa}/20} \cdot tan(\pi \frac{f_{c_{s,\kappa}}}{f_s}) \cdot \left( \frac{tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}} + tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})^2 + 1 \right)}{Q_{s,\kappa} \cdot 20 \cdot \left( \frac{10^{G_{s,\kappa}/20} \cdot tan(\pi f_{c_{s,\kappa}})}{Q_{s,\kappa}} + tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})^2 + 1 \right)^2}$ |
| $\frac{\partial b_{s,\kappa,1}}{\partial G_{s,\kappa}}$ | 0 |
| $\frac{\partial b_{s,\kappa,2}}{\partial G_{s,\kappa}}$ | $-\frac{log(10) \cdot 10^{G_{s,\kappa}/20} \cdot tan(\pi \frac{f_{c_{s,\kappa}}}{f_s}) \cdot \left( -\frac{tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}} + tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})^2 + 1 \right)}{Q_{s,\kappa} \cdot 20 \cdot \left( \frac{10^{G_{s,\kappa}/20} \cdot tan(\pi f_{c_{s,\kappa}})}{Q_{s,\kappa}} + tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})^2 + 1 \right)^2}$ |
| $\frac{\partial a_{s,\kappa,0}}{\partial G_{s,\kappa}}$ | 0 |
| $\frac{\partial a_{s,\kappa,1}}{\partial G_{s,\kappa}}$ | 0 |
| $\frac{\partial a_{s,\kappa,2}}{\partial G_{s,\kappa}}$ | $-\frac{log(10) \cdot 10^{G_{s,\kappa}} \cdot tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa} \cdot 20 \left( \frac{10^{G_{s,\kappa}} \cdot tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}} + tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})^2 + 1 \right)}$ $-\frac{log(10) \cdot 10^{G_{s,\kappa}/20} \cdot}{Q_{s,\kappa} \cdot 20 \left( \frac{10^{G_{s,\kappa}} \cdot tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}} + tan(\pi \frac{f_{c_{s,\kappa}}}{f_s})^2 + 1 \right)^2}$ |

Table 4.9: Local derivative of coefficients with respect to the gain, for a generic SOS.

| IIR Partial Derivative | Boost case |
|---|---|
| $\frac{\partial b_{s,\kappa,0}}{\partial Q_{s,\kappa}}$ | $\frac{(10^{G_{s,\kappa}/20}-1)\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})\cdot(tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)}{(Q_{s,\kappa}\cdot tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+Q_{s,\kappa}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s}))^2}$ |
| $\frac{\partial b_{s,\kappa,1}}{\partial Q_{s,\kappa}}$ | $\frac{(2\cdot tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})-1)\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{(Q_{s,\kappa}\cdot tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+Q_{s,\kappa}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s}))^2}$ |
| $\frac{\partial b_{s,\kappa,2}}{\partial Q_{s,\kappa}}$ | $\frac{(10^{G_{s,\kappa}/20}+1)\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})\cdot(tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)}{(Q_{s,\kappa}\cdot tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+Q_{s,\kappa}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s}))^2}$ |
| $\frac{\partial a_{s,\kappa,0}}{\partial Q_{s,\kappa}}$ | 0 |
| $\frac{\partial a_{s,\kappa,1}}{\partial Q_{s,\kappa}}$ | $\frac{(2\cdot tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})-1)\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{(Q_{s,\kappa}\cdot tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+Q_{s,\kappa}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s}))^2}$ |
| $\frac{\partial a_{s,\kappa,2}}{\partial Q_{s,\kappa}}$ | $\frac{(tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+1)\cdot 2\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{(Q_{s,\kappa}\cdot tan^2(\pi\frac{f_{c_{s,\kappa}}}{f_s})+Q_{s,\kappa}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s}))^2}$ |
| | Cut Case |
| $\frac{\partial b_{s,\kappa,0}}{\partial Q_{s,\kappa}}$ | $\frac{10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})\left(\frac{tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})^2+1\right)}{Q_{s,\kappa}^2\cdot\left(\frac{10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}}+1\right)^2}-\frac{tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}^2\left(\frac{10^{G_{s,\kappa}/20}}{Q_{s,\kappa}}+tan(\pi\frac{f_{c_{S,\kappa}}}{f_s})^2+1\right)}$ |
| $\frac{\partial b_{s,\kappa,1}}{\partial Q_{s,\kappa}}$ | $-\frac{log(10)\cdot 10^{g_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})\cdot(2\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})^2+1)}{Q_{s,\kappa}\cdot 20\cdot\left(\frac{10^{G_{s,\kappa}\cdot tan(pi\frac{f_{c_{s,\kappa}}}{f_s})}}{Q_{s,\kappa}}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})^2+1\right)^2}$ |
| $\frac{\partial b_{s,\kappa,2}}{\partial Q_{s,\kappa}}$ | $\frac{10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})\left(\frac{tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})^2+1\right)}{Q_{s,\kappa}^2\cdot\left(\frac{10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}}+1\right)^2}+\frac{tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}^2\left(\frac{10^{G_{s,\kappa}/20}}{Q_{s,\kappa}}+tan(\pi\frac{f_{c_{S,\kappa}}}{f_s})^2+1\right.}$ |
| $\frac{\partial a_{s,\kappa,0}}{\partial Q_{s,\kappa}}$ | 0 |
| $\frac{\partial a_{s,\kappa,1}}{\partial Q_{s,\kappa}}$ | $-\frac{log(10)\cdot 10^{g_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})\cdot(2\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})^2+1)}{Q_{s,\kappa}\cdot 20\cdot\left(\frac{10^{G_{s,\kappa}\cdot tan(pi\frac{f_{c_{s,\kappa}}}{f_s})}}{Q_{s,\kappa}}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})^2+1\right)^2}$ |
| $\frac{\partial a_{s,\kappa,2}}{\partial Q_{s,\kappa}}$ | $\frac{10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}^2\cdot\left(\frac{10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})^2+1\right)}+\frac{10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})\left(-\frac{10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}}-tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})^2+1\right)}{Q_{s,\kappa}^2\cdot\left(\frac{10^{G_{s,\kappa}/20}\cdot tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})}{Q_{s,\kappa}}+tan(\pi\frac{f_{c_{s,\kappa}}}{f_s})^2+1\right)^2}$ |

Table 4.10: Local derivative of coefficients with respect to the quality factor, for a generic SOS.

domain, thus, the partial derivative is given by:

$$\frac{\partial \mathscr{L}_1}{\partial \tilde{h}_m(n)} = \sum_{k=0}^{N-1} \left[ \frac{\partial L_1}{\partial |\tilde{H}_m(k)|} \cdot \frac{\partial |\tilde{H}_m(k)|}{\partial Re[\tilde{H}_m(k)]} \cdot \frac{\partial Re[\tilde{H}_m(k)]}{\partial \tilde{h}_m(n)} + \frac{\partial L_1}{\partial |\tilde{H}_m(k)|} \cdot \frac{\partial |\tilde{H}_m(k)|}{\partial Im[\tilde{H}_m(k)]} \cdot \frac{\partial Im[\tilde{H}_m(k)]}{\partial \tilde{h}_m(n)} \right] \tag{4.29}$$

then, using the Wirtinger calculus, the partial derivative is:

$$\begin{aligned} \frac{\partial \mathscr{L}_1}{\partial \tilde{h}_m(n)} = &\sum_{k=0}^{N-1} \left[ \frac{|\tilde{H}_m(k)| - |H_{des,m}(k)|}{\sqrt{\sum_k (|\tilde{H}_m(k)| - |\tilde{H}_{des,m}(k)|)^2}} \cdot \frac{Re[\tilde{H}_m(k)]}{|\tilde{H}_m(k)|} \cdot cos(\frac{2\pi}{N} kn) \right] \\ &- \sum_{k=0}^{N-1} \left[ \frac{|\tilde{H}_m(k)| - |\tilde{H}_{des,m}(k)|}{\sqrt{\sum_k (|\tilde{H}_m(k)| - |\tilde{H}_{des,m}(k)|)^2}} \cdot \frac{Im[\tilde{H}_m(k)]}{|\tilde{H}_m(k)|} \cdot sin(\frac{2\pi}{N} kn) \right] \end{aligned} \tag{4.30}$$

To calculate the local derivative of $\tilde{h}_{s,m}$ with respect to a generic control parameter $p$, the Wirtinger calculus are used to determine the local derivative:

$$\frac{\partial \tilde{h}_{s,m}}{\partial \tilde{H}_{s,m}} = \sum_{n=0}^{N-1} cos(2\pi \frac{kn}{N}) + sin(2\pi \frac{kn}{N}) \tag{4.31}$$

The partial derivative of the magnitude response with respect to the channel gain $\frac{\partial \tilde{H}_{s,m}}{\partial V_{s,dB}}$ is:

$$\frac{\partial \tilde{H}_{s,m}}{\partial V_{s,dB}} = H_{s,m}(k) \cdot \frac{log(10) \cdot 10^{V_{s,dB}/20}}{20} \prod_{j=1}^{K} \frac{B_{s,j}(k)}{A_{s,j}(k)} \tag{4.32}$$

Regarding the other parameters, the partial derivative of $\frac{\partial \tilde{H}_{s,m}(k)}{\partial B_{s,\kappa}(k)}$ and $\frac{\partial \tilde{H}_{s,m}(k)}{\partial A_{s,\kappa}(k)}$ are calculated:

$$\frac{\partial \tilde{H}_{s,m}(k)}{\partial B_{s,\kappa}(k)} = H_{s,m}(k) \cdot V_s \cdot \frac{1}{A_{s,\kappa}(k)} \prod_{j=1, j\neq\kappa}^{K} \frac{B_{s,j}(k)}{A_{s,j}(k)} \tag{4.33}$$

$$\frac{\partial \tilde{H}_{s,m}(k)}{\partial A_{s,\kappa}(k)} = -H_{s,m}(k) \cdot V_s \cdot \frac{B_{s,\kappa}(k)}{A^2_{s,\kappa}(k)} \prod_{j=1, j\neq\kappa}^{K} \frac{B_{s,j}(k)}{A_{s,j}(k)} \tag{4.34}$$

The partial derivative of $\tilde{h}_{s,m}$ with respect to a generic parameter $p_{s,k}$ is calculated exploiting the conversion from time to frequency and from frequency

to time using the Wirtinger calculus:

$$\begin{aligned}\frac{\partial \tilde{h}_{s,m}}{\partial p_{s,\kappa}} =& \left( \sum_{n=0}^{N-1} cos(2\pi \frac{kn}{N}) + \sum_{n=0}^{N-1} sin(2\pi \frac{kn}{N}) \right) \cdot \left[ \frac{\partial b_{0_{s,\kappa}}}{\partial p_{s,\kappa}} \sum_{k=0}^{N-1} \frac{\partial \tilde{H}_{s,m}(k)}{Re(B_{s,m}(k))} \right. \\ &+ \frac{\partial b_{1_{s,\kappa}}}{\partial p_{s,\kappa}} \Big( \sum_{k=0}^{N-1} \frac{\partial \tilde{H}_{s,m}(k)}{Re(B_{s,m}(k))} cos(2\pi \frac{k}{N}) - \sum_{k=0}^{N-1} \frac{\partial \tilde{H}_{s,m}(k)}{Im(B_{s,m}(k))} sin(2\pi \frac{k}{N}) \Big) \\ &+ \frac{\partial b_{2_{s,\kappa}}}{\partial p_{s,\kappa}} \Big( \sum_{k=0}^{N-1} \frac{\partial \tilde{H}_{s,m}(k)}{Re(B_{s,m}(k))} cos(2\pi \frac{2k}{N}) - \sum_{k=0}^{N-1} \frac{\partial \tilde{H}_{s,m}(k)}{Im(B_{s,m}(k))} sin(2\pi \frac{2k}{N}) \Big) \\ &+ \frac{\partial a_{1_{s,\kappa}}}{\partial p_{s,\kappa}} \Big( \sum_{k=0}^{N-1} \frac{\partial \tilde{H}_{s,m}(k)}{Re(A_{s,m}(k))} cos(2\pi \frac{k}{N}) - \sum_{k=0}^{N-1} \frac{\partial \tilde{H}_{s,m}(k)}{Im(A_{s,m}(k))} sin(2\pi \frac{k}{N}) \Big) \\ &+ \left. \frac{\partial a_{2_{s,\kappa}}}{\partial p_{s,\kappa}} \Big( \sum_{k=0}^{N-1} \frac{\partial \tilde{H}_{s,m}(k)}{Re(A_{s,m}(k))} cos(2\pi \frac{2k}{N}) - \sum_{k=0}^{N-1} \frac{\partial \tilde{H}_{s,m}(k)}{Im(A_{s,m}(k))} sin(2\pi \frac{2k}{N}) \Big) \right] \end{aligned} \tag{4.35}$$

where, through the Wirtinger calculus, the partial derivatives of frequency responses of numerator and denominator of the SOS's with respect to the their coefficients are $\frac{\partial B(k)}{\partial b_{0,\kappa}} = \frac{\partial A(k)}{\partial a_{0,\kappa}} = 1$ and $\frac{\partial B(k)}{\partial b_{1,\kappa}} = \frac{\partial B(k)}{\partial b_{2,\kappa}} = \frac{\partial A(k)}{\partial a_{1,\kappa}} = \frac{\partial A(k)}{\partial a_{2,\kappa}} = cos(2\pi \frac{k}{N}) - sin(2\pi \frac{k}{N})$.

The local derivative of the IIR filter coefficients with respect to the parameters $f_c$, $V_{0,dB}$, $Q$ can be calculated through the partial derivatives of the equations presented in Tables 3.1 and 3.2, according to the Boost or Cut case. The local derivatives of the IIR coefficients with respect th the central frequency are presented in Table 4.8, whereas in Tables 4.9 and 4.10, the local derivatives of IIR filter coefficients with respect to the gain and quality factor are shown, respectively.

Finally, the local derivative of the denormalization step is executed:

$$\frac{\partial q}{\partial p} = \frac{q_{max} - q_{min}}{2} \tag{4.36}$$

where $p$ is the normalized parameter, $q$ is any of the denormalized equalizer parameters, and the terms $q_{max}$ and $q_{min}$ denote its range.

The local derivative with respect to the regularization term is given by the cascade of three local derivatives:

$$\frac{\partial \mathscr{L}_2}{\partial \tilde{h}_{s,n}(n)} = \frac{\partial \mathscr{L}_2}{\partial \hat{r}_{s,m}} \cdot \frac{\partial \hat{r}_{s,m}}{\partial \hat{\epsilon}_{s,m}} \cdot \frac{\partial \hat{\epsilon}_{s,m}}{\partial \tilde{h}_{s,m}(n)} \tag{4.37}$$

where the local derivatives are given by:

$$\frac{\partial \mathscr{L}_2}{\partial \hat{r}_{s,m}} = \frac{\hat{r}_{s,m} - r_{s,m}}{\sqrt{\sum_{s=1}^{S} (\hat{r}_{s,m} - r_{s,m})^2}} \tag{4.38}$$

$$\frac{\partial \hat{r}_{s,m}}{\partial \hat{\epsilon}_{s,m}} = -\frac{\hat{\epsilon}_{1,m}}{\hat{\epsilon}^2_{s,m}} \tag{4.39}$$

$$\frac{\partial \hat{\epsilon}_{s,m}}{\partial \tilde{h}_{s,m}(n)} = 2 \cdot \tilde{h}_{s,m}(n) \tag{4.40}$$

### Wirtinger's calculus

Wirtinger's calculus are a means of computing gradients of real valued cost functions defined on complex domains [134]. A differentiable function $f : \mathbb{R} \to \mathbb{C}$, its real derivative at $a \in \mathbb{R}$ is defined as $\frac{\partial f}{\partial x}(x)$ [135].

For a differentiable $f : \mathbb{R}^M \to \mathbb{C}$, its real gradient at $a \in \mathbb{R}$ is denoted as:

$$\frac{\partial f}{\partial x}(a) = \Big(\frac{\partial f}{\partial x_1}(a), \frac{\partial f}{\partial x_2}(a), \cdots, \frac{\partial f}{\partial x_M}(a)\Big) \tag{4.41}$$

The spectrum of a generic vector $\mathbf{c} \in \mathbb{C}^M$, denoted as $\mathbf{C} \in \mathbb{C}^N$, is defined as:

$$C_n = \mathscr{F}(c)_n = \sum_{m \in [0, M-1]} c_m e^{-j\frac{2\pi mn}{N}} \tag{4.42}$$

The Inverse Discrete Fourier Transform (IDFT) is given by:

$$c_m = \mathscr{F}^{-1}(C)_m = \frac{1}{N} \sum_{n \in [0, N-1]} C_n e^{j\frac{2pimn}{N}} \tag{4.43}$$

The components of the vector $c \in \mathbb{C}$ can be decomposed as $a + jb$ with $(a, b) \in \mathbb{R}^2$ its real and imaginary parts. Similarly, any function $f \in \mathbb{C} \to \mathbb{C}$ can be considered as a function of $\mathbb{R}^2 \to \mathbb{C}$ with $f(c) = f(a, b)$. The derivative of $f$ at $c$ with respect to the real part of its input is denoted by $\frac{\partial f}{\partial x}(c)$ and with respect to the imaginary part is denoted as $\frac{\partial f}{\partial y}(c)$.

The Wirtinger calculus can be used when $f$ is differentiable with respect to both the real and imaginary part [135].

If $f : \mathbb{C} \to \mathbb{C}$ is differentiable in the real sense, its Wirtinger derivative at $c \in \mathbb{C}$ is defined as:

$$\frac{\partial f}{\partial z}(c) = \frac{1}{2}\Big(\frac{\partial f}{\partial x}(c) - j\frac{\partial f}{\partial y}(c)\Big) \tag{4.44}$$

while its conjugate Wirtinger derivative is defined as:

$$\frac{\partial f}{\partial z^*}(c) = \frac{1}{2}\Big(\frac{\partial f}{\partial x}(c) + j\frac{\partial f}{\partial y}(c)\Big) \tag{4.45}$$

Thus, calculating the two derivatives is equivalent to manipulating the two real partial derivatives.

Wirtinger's calculus satisfy some properties, such as the linearity property. Defining two functions $f$ and $g$ and $(\alpha, \beta) \in \mathbb{C}^2$, linearity can be expressed as:

$$\frac{\partial(\alpha \cdot f + \beta \cdot g)}{\partial z} = \alpha \frac{\partial f}{\partial z} + \beta \frac{\partial g}{\partial z} \tag{4.46}$$

$$\frac{\partial(\alpha \cdot f + \beta \cdot g)}{\partial z^*} = \alpha \frac{\partial f}{\partial z^*} + \beta \frac{\partial g}{\partial z^*} \tag{4.47}$$

Another property is the function composition: defining two function $f$ and $g$ differentiable in the real sense, the Wirtinger chain rule gives:

$$\frac{\partial f * g}{\partial z} = \left(\frac{\partial f}{\partial z} * g\right) \cdot \frac{\partial g}{\partial z} + \left(\frac{\partial f}{\partial z^*} * g\right) \cdot \frac{\partial g^*}{\partial z} \tag{4.48}$$

$$\frac{\partial f * g}{\partial z} = \left(\frac{\partial f}{\partial z^*} * g\right) \cdot \frac{\partial g}{\partial z} + \left(\frac{\partial f}{\partial z^*} * g\right) \cdot \frac{\partial g^*}{\partial z^*} \tag{4.49}$$

The last property is the complex conjugate. Indeed, if $f$ denotes a function differentiable in the real sense, the following property holds:

$$\left(\frac{\partial f}{\partial z}\right)^* = \frac{\partial f^*}{\partial z *} \tag{4.50}$$

instead, if $f$ is real-valued:

$$\left(\frac{\partial f}{\partial z}\right)^* = \frac{\partial f}{\partial z^*} \tag{4.51}$$

Finally, if $f$ is $\mathbb{C}$-differentiable, both its complex and Wirtinger gradients are equal:

$$f \quad \text{is} \quad \mathbb{C}\text{-differentiable} \Leftrightarrow \begin{cases} f & \text{is differentiable in the real sense} \\ \frac{\partial f}{\partial z^*} = 0 \end{cases} \tag{4.52}$$

These properties are useful to convert the Wirtinger gradients of real-valued function between time and frequency domains. Defining a function $\mathcal{E}$:

$$\begin{aligned} \mathcal{E} : \mathbb{C}^M &\to \mathbb{R} \\ z &\to \mathcal{E}(z) \end{aligned} \tag{4.53}$$

The gradient of the Discrete Fourier Transform (DFT) of $\mathbf{c}$, $\mathbf{C} \in \mathbb{C}^N$, is defined as:

$$\begin{aligned} \tilde{\mathcal{E}} : \mathbb{C}^M &\to \mathbb{R} \\ z &\to \mathcal{E}(\mathscr{F}^{-1}(z)) \end{aligned} \tag{4.54}$$

According to the chain rule of Wirtinger calculus, for $n \in [0, N-1]$, the local derivative of $\tilde{\mathcal{E}}$ with respect to $z_n$ is given by:

$$\frac{\partial \tilde{\mathcal{E}}}{\partial z_n}(\mathbf{C}) = \sum_{m \in [1,M]} \left( \frac{\partial \mathcal{E}}{\partial z_m}(\mathscr{F}^{-1}(\mathbf{C})) \right) \cdot \frac{\partial \mathscr{F}_m^{-1}}{\partial z_n}(\mathbf{C}) + \left( \frac{\partial \mathcal{E}}{\partial z_m^*}(\mathscr{F}^{-1}(\mathbf{C})) \right) \cdot \frac{\partial (\mathscr{F}_m^{-1})^*}{\partial z_n}(\mathbf{C}) \tag{4.55}$$

$\mathscr{F}^{-1}$ is $\mathbb{C}$-differentiable, thus the Wirtinger$^*$ gradient is null. Then, according to Equation 4.50, the Wirtinger$^*$ gradient of $(\mathscr{F}^{-1})^*$ is also null. The equation is:

$$\frac{\partial \tilde{\mathcal{E}}}{\partial z_n}(\mathbf{C}) = \sum_{m \in [1,M]} \frac{\partial \mathcal{E}}{\partial z_m}(\mathbf{c}) \cdot \frac{\partial \mathscr{F}_m^{-1}}{\partial z_n}(\mathbf{C}) \tag{4.56}$$

and it can be derived as:

$$\frac{\partial \mathscr{F}_m^{-1}}{\partial z_n}(\mathbf{C}) = \frac{1}{N} \cdot \frac{\partial}{\partial z_n} \Big( \sum_k z_k e^{j\frac{2\pi mk}{N}} \Big) \Big|_{z=\mathbf{C}} = \frac{1}{N} e^{j\frac{2\pi mk}{N}} \tag{4.57}$$

thus, the equation is:

$$\frac{\partial \tilde{\mathcal{E}}}{\partial z_n}(\mathbf{C}) = \frac{1}{N} \sum_{m \in [1,M]} \frac{\partial \mathcal{E}}{\partial z_m}(\mathbf{c}) e^{j\frac{2\pi mn}{N}} = \frac{1}{N} \left[ \sum_{m \in [1,M]} \Big( \frac{\partial \mathcal{E}}{\partial z_m}(\mathbf{c}) \Big)^* e^{j\frac{2\pi mn}{N}} \right]^* \tag{4.58}$$

The exèression recognizes the DFT, thus, the Wirtinger gradient of $\tilde{\mathcal{E}}$ is:

$$\frac{\partial \tilde{\mathcal{E}}}{\partial z}(\mathbf{C}) = \frac{1}{N} \mathscr{F} \Big( \Big[ \frac{\partial \mathcal{E}}{\partial z}(\mathbf{c}) \Big]^* \Big)^* \tag{4.59}$$

Since $\mathcal{E}$ and $\tilde{\mathcal{E}}$ are real-valued functions, the previous expression can be formulated as:

$$\frac{\partial \tilde{\mathcal{E}}}{\partial z^*}(\mathbf{C}) = \frac{1}{N} \mathscr{F} \Big( \frac{\partial \mathcal{E}}{\partial z^*}(\mathbf{c}) \Big) \tag{4.60}$$

The expression can be reversed to give the Wirtinger gradient from frequency to time domain:

$$\frac{\partial \mathcal{E}}{\partial z^*}(\mathbf{c}) = N \mathscr{F} \Big( \frac{\partial \tilde{\mathcal{E}}}{\partial z^*}(\mathbf{C}) \Big) \tag{4.61}$$

### 4.3.3 Experimental Setup

Two scenarios were used for the Parametric IIR filters experiments: A room composed of eight speakers and two microphones, and the Alfa Romeo Giulia, presented in Section 4.2.4.

The room has a dimension of $4.0 \times 5.5 \times 3.0$ m. The speakers are circularly placed around two seats, as shown in Figure 4.14. The left seat is fitted with two

omnidirectional microphones, simulating the listener's ears. The loudspeakers are mid-woofers with a frequency range between 100 Hz and 15 kHz. The impulse responses were measured using the exponential sweep method with a sampling frequency of 48 kHz, using a RME Madiface Audio Interface and a Dante-equipped amplification system.

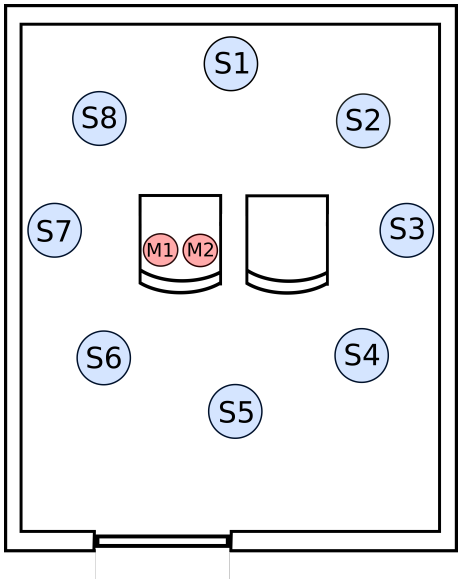



Figure 4.14: Top view of the room showing the placement of the speakers and microphones.

With respect to the car cabin scenario, the room exhibits different characteristics. The room employs one type of loudspeaker arranged in a rectangular pattern. These cover slightly more than two decades of the audio range, and the maximum excursion between minimum and maximum in the unequalized frequency responses never exceed 15 dB. As explained in Section 4.2.4, the car cabin is fitted with loudspeakers of different sizes and bandwidths, arranged irregularly. Furthermore, the car material absorption coefficient varies largely, resulting in a large excursion of the frequency response (greater than 20 dB) and a wider range to equalize, covering almost three decades. For this reason, the car cabin is a more challenging scenario.

The first experiments were conducted in the room scenario to gather more insights on the described equalization techniques, compare the proposed methods, and find suitable hyperparameters. The random search was performed in the Multiple Input - Multiple Output (MIMO) case, and then the best architecture was used for the other experiments, the SISO and Multiple Input - Single Output (MISO) cases for the room scenario and the MIMO case in the

Alfa Romeo Giulia case.

For the room scenario, the desired frequency response is a 0 dB flat band between 100 Hz and 14 kHz, whereas, in the car cabin scenario, the range is 20 Hz and 14 kHz.

Both optimization and evaluation were conducted using a one-third octave band averaging for the frequency responses. This choice is motivated by the human ear resolution, which is not very sensitive to narrow dips and notches.

The number of SOS's is equal to the number of one-third octave bands within the speaker's operating frequency range. Thus, in the room scenario, the number of SOS's for each speaker was 22, whereas for the car cabin case goes from 21 to 29. Therefore, the number of parameters to optimize is 536 and 539, respectively.

The ranges of the other parameters are: $Q_{min}$=0.05, $Q_{max}$=5.0, $V_{0,min,dB}$=-10 dB, $V_{0,max,dB}$=10 dB, $V_{s,min,dB}$=-20 dB, $V_{s,max,dB}$=20 dB. Regarding the FD, the $\beta_{FD}$ is set to $1 \cdot 10^{-4}$, whereas for the DSM, $\gamma$ is set to 0.01.

The performance is evaluated using the $\overline{MSE}$ and $\overline{\sigma}$ in the one-third octave band and within the desired frequency range.

Before performing the optimization, pre-processing is performed, calculating the delay of each speaker and then the offset gain to be added in the optimization in order to normalize the output frequency responses to 0 dB.

The BiasNet was compared with two other convolutional architectures: the CNN used for the FIR filter design for Multipoint Audio Equalization (see Section 4.2.3). The other neural architecture is called Convolutional Feedback Network (CFN) because this model is a CNN with variable input. The idea stems that the network adapts its input in addition to its weight, like the observations drawn in describing the BiasNet. At the first iteration, the Room Impulse Responses (RIRs) feed the network, however, at each successive iteration, the network is fed with the equalized RIRs. Therefore, this network establishes input-output feedback since the equalized response generated at the current iteration is fed as input at the next iteration.

As activation function, the sine activation is used [136] since it avoids local minima during network optimization. Moreover, it behaves well for backpropagation as its derivatives do not vanish.

30 BiasNet configurations were tested, varying the number of layers from 1 to 10 and the number of neurons from 16 to 4096. For the CNN and CFN, 1 or 2 convolutional layers were tested: the number of kernels equals 25 on the one convolutional layer case, 48 and 24 or 100 and 10 on the case of 2 layers. The kernel has size equals $\mathcal{M} \times 1$ and $1 \times \mathcal{S}$ for the two convolutional layers. The number of hidden layers varied from 1 to 4, with the number of neurons ranging from 32 to 1024. The Adam optimizer was used, with a learning rate equal to $1 \cdot 10^{-4}$. The number of iterations was set to 10,000.

FD and DSM were implemented in *Matlab*, while the neural architectures were implemented in *Python*, using *Tensorflow* 2.0.0. The experiments were performed using a machine with an Intel i7 processor, 32 GB of RAM and an Nvidia Titan GPU with 12 GB of dedicated RAM.

### 4.3.4 Results

#### Room Scenario

| Architecture | $\overline{MSE}$ | Layers | No. Learnable Parameters |
|---|---|---|---|
| BiasNet #26 | $1.18 \cdot 10^{-5}$ | (1024, 512, 256, 128) | 758,784 |
| BiasNet #5 | $1.19 \cdot 10^{-5}$ | (256) | 137,728 |
| BiasNet #9 | $1.29 \cdot 10^{-5}$ | (128) | 68,864 |
| CFN | $1.38 \cdot 10^{-5}$ | best CFN† | 4,673,390 |
| BiasNet #7 | $1.46 \cdot 10^{-5}$ | (256, 256, 256) | 268,800 |
| CNN | $1.66 \cdot 10^{-5}$ | same as [123] | 2,369,390 |
| BiasNet #25 | $1.68 \cdot 10^{-4}$ | (16,32,64,128,256,512, 256,128,64,32,16) | 357,792 |
| BiasNet #8 | $4.24 \cdot 10^{-4}$ | (64) | 34,432 |
| BiasNet #10 | $6.36 \cdot 10^{-4}$ | (32) | 17,216 |
| BiasNet #4 | $9.56 \cdot 10^{-1}$ | out only (with bias) | 536 |
| No EQ | 0.377 | - | - |

Table 4.11: Preliminary test comparing several neural networks in the MIMO configuration for the room scenario. The number of neurons for each hidden layer is shown in round brackets in the Layers column. †: the CFN was the best among all the tested CFN and is composed of 2 convolutional layers of 100 and 10 kernels, respectively, and 3 dense layers of 64 neurons each.

The first experiments were performed analyzing the three neural architectures, the BiasNet, the CNN and the CFN, for the Multipoint Audio Equalization task in the room scenario in the MIMO case. The best network hyperparameters will be tested for SISO and MISO cases. The results are presented in Table 4.11: many architectures proved similar performance ($\overline{MSE} = 10^{-5}$), however, the difference in the number of trainable parameters is extremely large. Comparing the CNN and the CFN, the best CFN achieves slightly better performance at the cost of an order of magnitude more parameters. The best BiasNet achieved the best performance than the other algorithms with a significantly lower number of parameters (17,764 trainable parameters) but achieved similar performance as configuration #26 that is 40 times larger to train. However, the performance decreases in other configurations: in configuration #9, the results are similar to #26, but the number of neurons is half

to its only layer. However, the performance falls when decreasing the number of neurons of the only layer, as in configuration #8 and #10. The worst performance with the BiasNet configuration is achieved with the configuration #4, which has only the output layer, with a learnable bias, resulting in an unacceptable equalization performance.

| Method | $MSE$ | $\sigma$ |
|---|---|---|
| BiasNet | $\mathbf{1.32 \cdot 10^{-5}}$ | $\mathbf{1.58 \cdot 10^{-2}}$ |
| $FD_{1024}$ | $9.74 \cdot 10^{-3}$ | $4.36 \cdot 10^{-1}$ |
| $FD_{8192}$ | $2.08 \cdot 10^{-4}$ | $5.54 \cdot 10^{-2}$ |
| DSM | $3.43 \cdot 10^{-2}$ | $9.92 \cdot 10^{-1}$ |
| No EQ | $3.80 \cdot 10^{-1}$ | 1.69 |

Table 4.12: Results for SISO equalization, room scenario.

Another factor to analyze is the computational performance, in particular, the time achieved by the network to converge. The optimization time depends on two factors: the number of iterations to reach a target goal and the time required by each iteration.

For the SISO experiments in the room scenario, the best BiasNet and the baseline techniques are compared. The results are presented in Table 4.12: the neural approach achieves better results than the baseline techniques by one or more orders of magnitude. The DSM is unable to improve the performance respect the non-equalized case. The FD approach achieved excellent performance, however, the BiasNet is superior.

| Method | Right Mic | | L+R Mic | |
|---|---|---|---|---|
| | $MSE$ | $\sigma$ | $\overline{MSE}$ | $\overline{\sigma}$ |
| BiasNet | $8.95 \cdot 10^{-6}$ | $1.26 \cdot 10^{-2}$ | $\mathbf{5.53 \cdot 10^{-2}}$ | $\mathbf{8.04 \cdot 10^{-1}}$ |
| DSM | $1.87 \cdot 10^{-2}$ | $6.98 \cdot 10^{-1}$ | $8.76 \cdot 10^{-2}$ | 1.42 |
| $FD_{1024}$ | $5.26 \cdot 10^{-2}$ | $3.55 \cdot 10^{-1}$ | $1.63 \cdot 10^{-1}$ | $9.05 \cdot 10^{-1}$ |
| $FD_{8192}$ | $\mathbf{4.88 \cdot 10^{-6}}$ | $\mathbf{9.58 \cdot 10^{-3}}$ | $1.30 \cdot 10^{-1}$ | $9.52 \cdot 10^{-1}$ |
| No EQ | $3.92 \cdot 10^{-1}$ | 1.91 | $3.77 \cdot 10^{-1}$ | 1.99 |

Table 4.13: Results for MISO equalization, room scenario evaluated at the listening point used during optimization (Right Mic) and both microphones (L+R Mic).

MISO experiments results are presented in Table 4.13, reporting the results on the right microphone and the average of the $\overline{MSE}$ calculated at both listening points. The performance is increased than the SISO case, improving

the equalization performance when the number of loudspeakers is increased. However, considering only one listening position for optimization provides a solution that does not work well for the other listening position. Indeed, the performance of the best methods decreases by at least four orders of magnitude when evaluating the performance at both microphones.

The $FD_{1024}$ method is inferior to the IIR filters provided by the DSM, whereas the $FD_{8192}$ slightly improves over the proposed method, but it suffers analyzing both the microphones.

Finally, the best BiasNet architecture is compared with the baseline techniques in the MIMO scenario, achieving better performance than the DSM and the FD method (see Table 4.14).

| Method | $\overline{MSE}$ | $\overline{\sigma}$ |
|---|---|---|
| BiasNet | $\mathbf{1.18 \cdot 10^{-5}}$ | $\mathbf{1.40 \cdot 10^{-2}}$ |
| DSM | $2.54 \cdot 10^{-2}$ | $7.18 \cdot 10^{-1}$ |
| $FD_{1024}$ | $4.57 \cdot 10^{-2}$ | $4.36 \cdot 10^{-1}$ |
| $FD_{8192}$ | $1.38 \cdot 10^{-5}$ | $1.57 \cdot 10^{-2}$ |
| No EQ | $3.77 \cdot 10^{-1}$ | 1.99 |

Table 4.14: Results for MIMO equalization, room scenario.

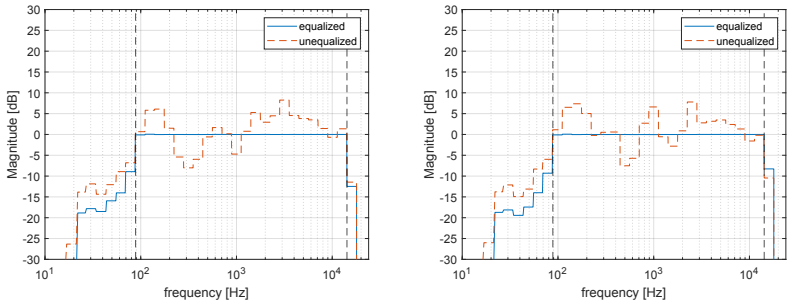
Figure 4.15 presents the one-third octave band magnitude frequency responses. The unequalized response is shown in red line, whereas the equalized magnitude response is shown in blue. The difference between the two spectrum is evident: the unequalized frequency response exhibits an excursion of more than 10 dB, the equalized response is flat in the frequency range covered by the loudspeakers (vertical dashed lines).

As shown in Figure 4.16, the energy is preserved before and after the equalization process.

### Car Cabin Scenario

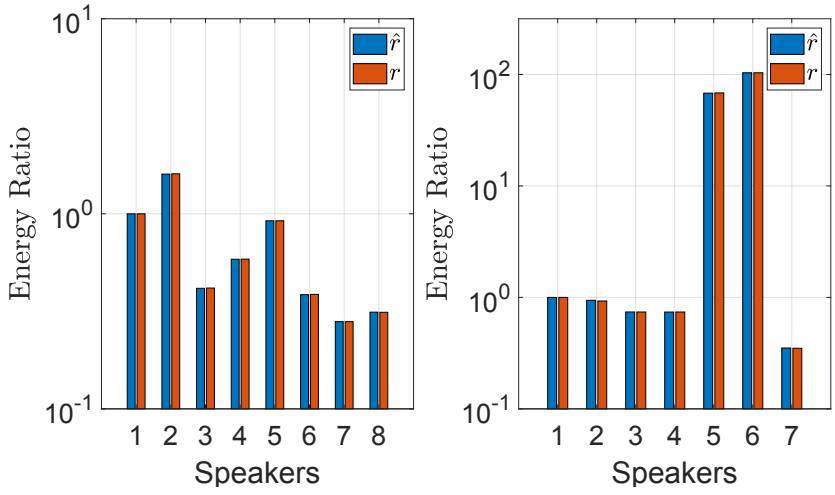
Regarding the car cabin scenario, the results are reported in Table 4.15. Since the car provides a more challenging scenario, the $\overline{MSE}$ is higher than the results presented for the room scene. The DSM fails to provide a good equalization performance. The FD method does not match the performance achieved by the proposed method.

In Figure 4.17 one-third octave band frequency responses are presented: at high frequencies, the amplitude responses are flat, both for the overlapping of the loudspeakers frequency responses and because the network optimizes in this frequency range, only two speakers out of the seven installed in the car. At




(a) Left microphone (b) Right microphone

Figure 4.15: One-third octave band magnitude response of (a) left and (b) right microphone in the room scenario. Red line is the unequalized frequency response, the blue line is the equalized one and the black dotted lines refer to the minimum and maximum frequency to be equalized.




(a) (b)

Figure 4.16: Bar graph of energy ratio after ($\hat{r}$) and before $r$ the optimization for the room scenario (a) and the car scenario (b). Speakers 5 and 6 in (b) are woofer and subwoofer, therefore have larger energy.

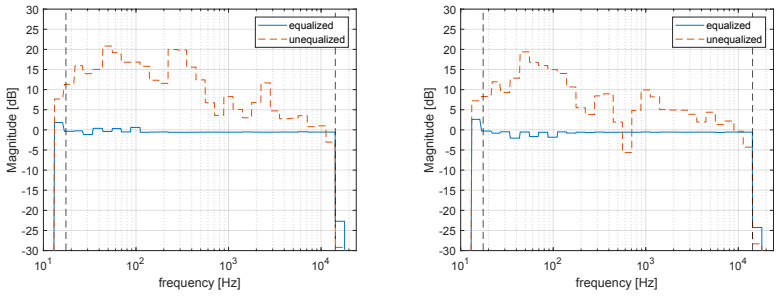
low frequencies, the network has not been able to optimize as well as at high frequencies. Indeed it presents a maximum deviation of 5 dB around 40 Hz.

Despite the more challenging scenario (speakers not arranged in a regular pattern, the different frequency ranges, irregular internal cabin volume), the equalization is superior to the FD method, which is usually considered the optimal method for room equalization, and the energy of the loudspeakers

| Method | $\overline{MSE}$ | $\overline{\sigma}$ |
|---|---|---|
| BiasNet | $\mathbf{5.74 \cdot 10^{-3}}$ | $\mathbf{1.83 \cdot 10^{-1}}$ |
| DSM | 3.62 | 2.76 |
| $FD_{1024}$ | $4.22 \cdot 10^{-2}$ | $8.15 \cdot 10^{-1}$ |
| $FD_{8192}$ | $1.84 \cdot 10^{-2}$ | $5.02 \cdot 10^{-1}$ |
| No Eq | 13.47 | 3.16 |

Table 4.15: Results for MIMO equalization, car cabin scenario.

signals is preserved (see Figure 4.16.b).




(a) Left microphone (b) Right microphone

Figure 4.17: One-third-octave band magnitude response of the measured signal at the reference microphones: (a) left and (b) right microphone in the car cabin scenario. The vertical black dotted lines denotes the frequency range to be equalized.

### Parametric IIR Filters Remarks

From the results in Table 4.14 and 4.15 is evident that the proposed method is comparable or slightly better than the $FD_{8192}$ in terms of performance. The computational cost between the FIR and IIR methods differs: regarding the FD method, each equalizer has 16,833 floating-point operations per sample, whereas the IIR equalizers are composed of 22 SOS's in the room scenario and 29 SOS's in the car cabin scenario, with 198 and 261 operations per sample, respectively, which means a reduction of the computational cost of almost two orders of magnitude. The cost rises when the number of loudspeakers increases. In the MIMO room scenario, the proposed method requires 1,594 operations against 131,064 operations for the $FD_{8192}$.

With the $FD_{1024}$, the performance is lower but acceptable, yielding an improvement of approximately one order of magnitude with respect to the non

equalized case. However, the number of operations per filter is still significantly larger than the proposed method, with a number of operations per sample equal to 2047.

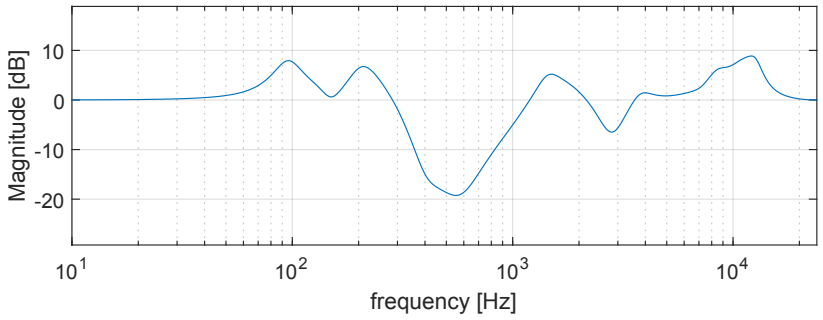



Figure 4.18: Magnitude response of an IIR filter optimized in MIMO scenario

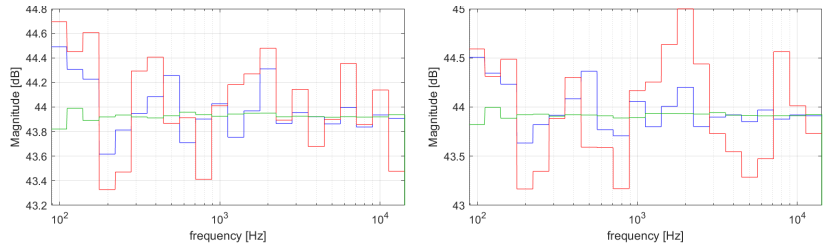
The resulting IIR filters exhibit a smooth amplitude response, it does not have very subtle peaks because the optimized Q parameter of the SOS's is not high. In Figure 4.18, the magnitude response of an IIR filter of a speaker is shown when the BiasNet optimized the Parametric IIR filters in the MIMO room scenario.

The IIR filters designed for the room experiments on the MIMO case are taken to the real room and applied for equalization. The IIR filters were loaded on a Simulink patch to preprocess the signal and fed the loudspeakers. The hardware setup is the same described in Section 4.3.3. The frequency responses were measured by reproducing white noise and comparing them with the simulated magnitude responses achieved in Section 4.3.4.

In Figure 4.19 is presented the magnitude responses: the red line is the measured one and the green line is the ideal one (obtaining by filtering a discrete-time impulse sequence). The observed deviation is at most 2 dB, but is inherent to the use of white noise as the input signal. Indeed, by computing the magnitude response of the room using white noise in the simulated environment, random deviations from the flat band (blue line) are achieved.

## 4.4 Final Remarks

In this chapter, FIR and IIR filter design for Multipoint Audio Equalization using Machine Learning techniques is presented. Binaural and Multipoint Audio Equalization experiments are shown. Deep Optimization networks are proposed to optimize FIR filter coefficients and Parametric IIR filter parameters. Regarding the state-of-the-art techniques, the proposed methods have the advantage of being non-convex; therefore, they do not stop at a local minimum




(a) Left microphone (b) Right microphone

Figure 4.19: One-third octave band magnitude response of the measured signal at the reference microphones: (a) left and (b) right microphone in the room scenario. The green line is the equalized magnitude response depicted in Figure 4.15. The blue line is simulated using white noise as input, while the red line is measured in the real scenario using white noise. Please note: the magnitude range is only 2 dB to emphasize the small differences.

and, thus, achieve better performance. The disadvantage of this technique is that it cannot be implemented for a real-time application as it has a very high computational cost, but, as shown by the experiments, the network converges very fast, thus a threshold could be set under the algorithm stops.

Several experiments were performed for the FIR filter design for Multipoint Audio Equalization. Baseline techniques were compared with evolutionary algorithms and Deep Neural Networks. The last ones achieved better results for both SISO, MISO and MIMO setup.

Several architectures and input sizes were analyzed: the CNN achieved better performance than the other deep neural architectures.

Regarding the IIR filter design, better performance is achieved than the baseline technique and FIR filter baseline method, analyzing the performance in terms of frequency response and computational cost. A novel architecture is described, the BiasNet, designed explicitly for Deep Optimization. This architecture, compared with other neural network models, achieved better performance with a lower computational cost.

The designed IIR filters were tested in a real scenario, showing almost no difference from the simulated frequency responses.

# Chapter 5

# Personal Sound Zones

Personal Sound Zones is the reproduction of sounds in certain regions, contained within an environment and where multiple listeners are present [137]. In recent years, this topic has been increasingly studied as it is fascinating from both an academic and an industrial perspective, particularly in the automotive industry [138]. Other scenarios where PSZ has been implemented are: user-computer experience [139, 140], parasol [141], in an aircraft [142], in a mobile device [143, 144] or in the car cabin [145].

The problem is solved by defining two zones: the bright zone, which is the zone of interest where the acoustic energy must be the highest; the dark zone, where the energy must be as low as possible [109].

Most of the methods used for PSZ are based on improving Acoustic Contrast $AC$, which is the ratio between the average sound power in the bright zone and the average sound power in the dark zone for each frequency bin: considering $\mathcal{M}_B$ control points to define the bright zone and $\mathcal{M}_D$ control point for the dark zone, the $AC$ is defined by [145]:

$$AC = \frac{\mathcal{M}_D}{\mathcal{M}_B} \frac{\tilde{H}_B^H \tilde{H}_B}{\tilde{H}_D^H \tilde{H}_D} = \frac{M_D}{M_B} \frac{G^H H_B^H H_B G}{G^H H_D^H H_D G} \tag{5.1}$$

where $G$ is the vector containing the frequency response of filters, $H_B$ is the matrix containing the frequency responses of the impulse responses in the bright zone, whereas $H_D$ is the matrix regarding the dark zone. $\tilde{H}_B$ and $\tilde{H}_D$ are the complex pressures in the bright and dark zone, respectively and $^H$ is the Hermitian.

Another constraint used to balance the filter energies is the Array Effort (E), which is given by [146]:

$$E = \frac{G^H G}{G_{ref}^H G_{ref}} \tag{5.2}$$

where $G^H G$ is the Array Effort required by the optimized source array and $G_{ref}^H G_{ref}$ is the Array Effort required when the array sources are driven in-phase to produce the same average sound pressure level in the bright zone as

the personal audio optimized array.

The main Acoustic Contrast-based methods are the Acoustic Contrast Control (ACC), explained in detail in Section 5.2, and the Pressure Matching (PM), discussed in Section 5.3. The Acoustic Energy Difference Maximization overcomes the limitations of the ACC [147], using as cost function the energy differences between the two zones. The Brightness Control [148] used constructive interference to produce sound zones, maximizing the sound pressure level in the bright zone. The Planarity Control Optimization is a plane-wave based method [149].

The techniques that increase contrast can reduce listening quality. The ACC does not control phase [150], so usually high acoustic contrast is always achieved, sacrificing sound quality, while the PM gives good sound quality but not high acoustic contrast [145].

Subjective tests were carried out to evaluate PSZ techniques and to get minimum acceptable acoustic contrast levels to reduce the disturbance in the dark zone [150, 151, 152]. In [151] an acoustic contrast of 11 dB was required to achieve an acceptable PSZ. In [150, 152] are showing that there is a significant difference between experienced and unexperienced listeners and between speech and music signals.

## 5.1 Metrics

The Personal Sound Zones is evaluated in terms of Acoustic Contrast and audio equalization in the bright zone. The dark zone is evaluated qualitatively and quantitatively. Qualitatively, the Acoustic Contrast is calculated as in Equation 5.1 but in one-third octave bands domain:

$$AC = 10 \cdot log_{10}\left(\frac{\mathcal{M}_D}{\mathcal{M}_B}\frac{\tilde{H}^H_{B,1/3}\tilde{H}_{B,1/3}}{\tilde{H}^H_{D,1/3}\tilde{H}_{D,1/3}}\right) \tag{5.3}$$

whereas quantitatively, three analyses are performed. From Equation 5.3, the first metric is the maximum Acoustic Contrast ($AC_{max}$), the second one is the average Acoustic Contrast in the band of interest set by the user ($\overline{AC}_{ib}$) and the third is the average Acoustic Contrast on the overall frequency range ($\overline{AC}_{fb}$). In Figure 5.1, an example of Acoustic Contrast is shown. From the graph, the $AC_{max}$ is determined, then the $\overline{AC}_{ib}$ and $\overline{AC}_{fb}$ are calculated in the one-third octave bands of interest and over the entire range, respectively.

For the cases analyzed in the experiments, the two frequency ranges correspond to 100-2000 Hz (9-th and 22-nd one-third octave band, with central frequency 99.2 Hz and 2.0 kHz, respectively) and 50-11000 Hz (7-th and 30-th one-third octave band, with central frequency 62.5 Hz and 12.7 kHz, respec-

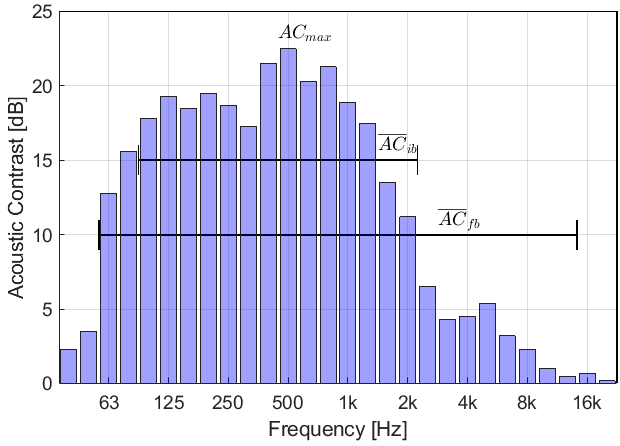



Figure 5.1: Example of Acoustic Contrast graph.

tively). The $\overline{AC}_{ib}$ is calculated between the 9-th and 22-nd one-third octave band because in this frequency range the characteristics of the frequency responses of the microphones are similar and because both the fundamental frequency and the most important harmonic components of the voice are present [153].

In the bright zone, the frequency and perceptual metrics [154, 155, 156] are evaluated. The frequency metrics are defined as in Section 4.3.2, while the perceptual metrics are explained below.

The first metric is the average Mean Square Error between reference audio and the filtered and recorded one from the microphones placed within the bright zone ($MSE_t$):

$$MSE_t = \frac{1}{\mathcal{M}_B N_a} \sum_{m=1}^{\mathcal{M}_B} \sum_{n=1}^{N_a} (x(n) - y_m(n))^2 \tag{5.4}$$

Where $N_a$ is the length of the recorded file.

Because the $MSE_t$ fails to give a good conclusion from a perceptual standpoint, other metrics are used.

Perceptual Evaluation of Speech Quality (PESQ) is a standard methodology for automated assessment of the speech quality [154]. It is used for objective voice quality testing for telecommunication companies and compressed speech files.

In Figure 5.2 is presented the scheme of PESQ method [154]. The level aligning of both signals starts the model to a standard listening level, then the signals are filtered using the FFT with an input filter to model a stan-

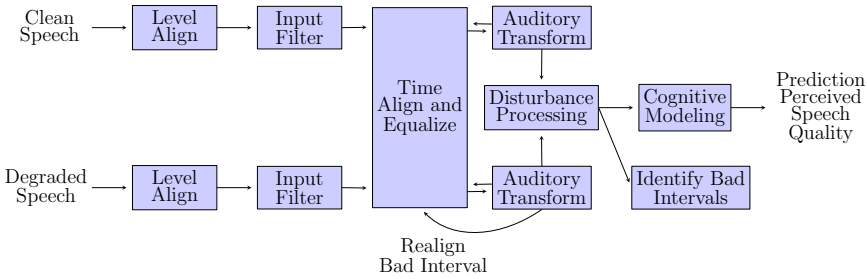



Figure 5.2: Structure of PESQ method [154].

dard telephone handset. The signals are aligned in time and then processed through an auditory transform, achieving two distortion parameters extracted from the difference between the transform of the signals. Then the parameters are aggregated in frequency and time and mapped to predict subjective Mean Opinion Score (MOS).

The Short-Time Objective Intelligibility measure (STOI) calculates the intelligibility [155] in a noisy environment. The scheme is presented in Figure 5.3. The reference and the degraded speech are time-frequency decomposed to obtain a simplified internal representation resembling the transform properties of the auditory system. The signals are segmented into 50% overlapping, windowed and zero-padded. The silent region, which does not contribute to speech intelligibility, is removed, finding the frame with maximum energy of the clean speech signal. The signals are then reconstructed, excluding all frames where the clean speech energy is lower than 40 dB with respect to their maximum clean speech energy frame. Then a one-third octave band analysis is performed, with a total of 15 one-third octave bands (the lowest center frequency is set to 150 Hz and the highest is 4.3 kHz).

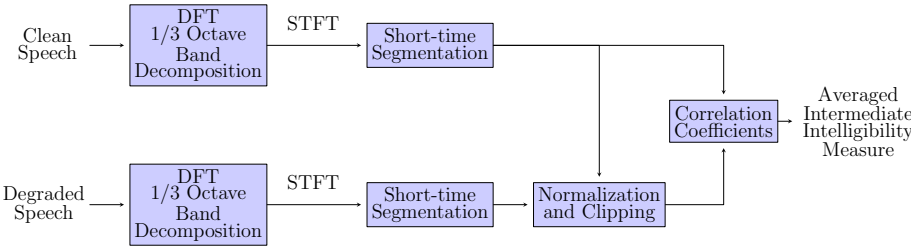



Figure 5.3: Structure of STOI method [155].

Last but not least, normalization and clipping are performed, and the intermediate intelligibility measure is calculated. Finally, the average of the intermediate intelligibility measure overall bands and frames is calculated.

The last perceptual metric used for the evaluation is the Virtual Speech

Quality Objective Listener (ViSQOL). It aims to be an objective, full-reference metric [156].

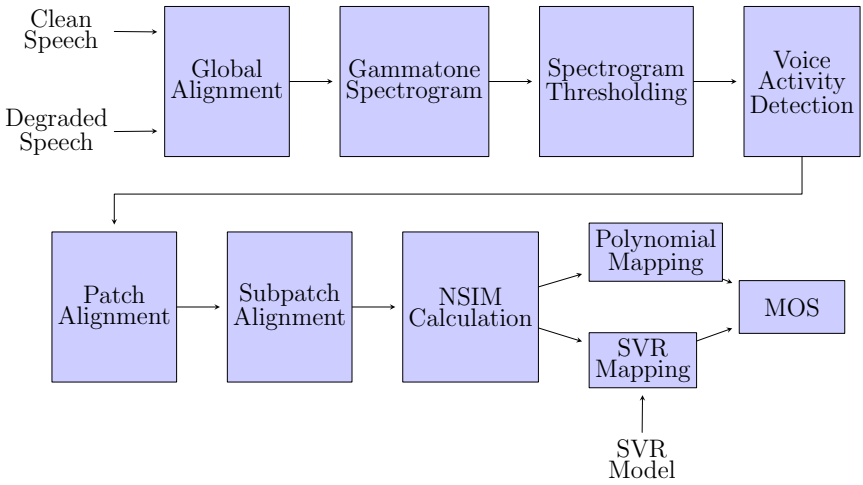



Figure 5.4: Structure of ViSQOL method [156].

The method starts with the alignment of referenced and noisy signals [157], then the gammatones are calculated (in the first version, Short-Time Fourier Transform STFT is used) using a gammatone filter. Silence gammatones are removed with a silence threshold. The resulting frames will need a simple Voice Activity Detection algorithm to detect speech. Two alignment steps have been performed, and the Neurogram Similarity Index measure (NSIM) is calculated. Support Vector Regression is trained to calculate the MOS score.

In Table 5.1 the perceptual metrics with their features and values are resumed.

| Metric | Feature | Value |
|---|---|---|
| $MSE_t$ | signals in time domain | Average of the Errors for each Sample |
| PESQ | Auditory Transform | MOS |
| STOI | STFT | Intelligibility |
| ViSQOL | Gammatone | MOS |

Table 5.1: Brief description of perceptual metrics used for the evaluations.

## 5.2 Acoustic Contrast Control

Acoustic Contrast Control maximizes the $AC$ [158]. minimizing $G^H H_D^H H_D G$ and helding constant $G^H H_B^H H_B G$. The method aims to minimize sound pressure in the dark zone and maximize it in the bright zone.

The maximization problem can be formulated as:

$$\underset{G}{argmax}\, AC \quad s.t. \quad G^H H_B^H H_B G = |H_{B,des}| \tag{5.5}$$

where $|H_{B,des}|$ is the sound pressure level on the bright zone.

In [158] are explained the direct and indirect formulation, using the Lagrange multipliers.

Regarding the Direct Formulation [159], the Lagrangian is given by:

$$\mathcal{L} = \tilde{H}_B^H \tilde{H}_B - \lambda_1(\tilde{H}_D^H \tilde{H}_D - |H_{D,des}|) \tag{5.6}$$

where $\lambda_1$ is the unknown Lagrange multipliers, which is real and positive and must satisfy the constraints. The complex differential of the real scalar $\partial\mathcal{L}/\partial G$ is defined as:

$$\frac{\partial \mathcal{L}}{\partial G} = \frac{\partial \mathcal{L}}{\partial G_R} + j\frac{\partial \mathcal{L}}{\partial G_i} = 2(H_B^H H_B G - \lambda_1 H_D^H H_D G) \tag{5.7}$$

The Lagrangian is maximized with respect to the real and imaginary components of $G$; if the vector of complex differentials is null, we have that:

$$\lambda_1 G = [H_D^H H_D]^{-1}[H_B^H H_B]G \tag{5.8}$$

Thus, $G$ is the eigenvector of the matrix $[H_D^H H_D]^{-1}[H_B^H H_B]$, and it must be associated to the largest eigenvalue $\lambda_1$. Some problems are encountered in this formulation [158]: the matrix $[H_D^H H_D]^{-1}[H_B^H H_B]$ depends on the geometry of the physical arrangement; if the number of microphones in the dark zone is less than the number of speakers, the matrix $[H_D^H H_D]$ is singular. To overcome these problems, a regularization parameter to each diagonal element of $[H_D^H H_D]$ is added.

In the Indirect formulation, we want to minimize $\tilde{H}_D^H \tilde{H}_D$, helding the constraints that $\tilde{H}_B^H \tilde{H}_B$ is equal to $|H_{B,des}|$. The Lagrangian in this case is:

$$\mathcal{L} = \tilde{H}_D^H \tilde{H}_D + \lambda_1(\tilde{H}_B^H \tilde{H}_B - |H_{B,des}|) \tag{5.9}$$

The complex differentials is:

$$\frac{\partial \mathcal{L}}{\partial G} = 2(H_D^H H_D G + \lambda_1 H_B^H H_B G) \tag{5.10}$$

The vector of complex differentials is equal to zero if:

$$\lambda_1 G = -[H_B^H H_B]^{-1}[H_D^H H_D]G \tag{5.11}$$

The solution is proportional to the eigenvector of the matrix $[H_B^H H_B]^{-1}[H_D^H H_D]$, associated with the smallest eigenvalue of this matrix.

## 5.3 Pressure Matching

The Least Square Method [145], called also Pressure Matching [160], minimizes the error between the target pressure $H_{B,des}$ and the sound pressure produced by the speakers in the bright zone and minimizes the squared pressures in the dark zone.

$$\min_G ||H_B G - H_{B,des}||^2 \quad s.t. \quad ||H_D G||^2 \leq |H_{D,des}| \tag{5.12}$$

The problem can be written as a Lagrangian cost function:

$$\mathcal{L} = ||H_B G||^2 + \lambda_1(||H_D G||^2 - |H_{D,des}|) \tag{5.13}$$

where $\lambda_1$ is real and positive.

The solution is given setting to zero the derivative $\partial\mathcal{L}/\partial G$:

$$[H_B^H H_B + \lambda_1 H_D^H H_D]G = H_B^H H_{B,des} \tag{5.14}$$

Some optimization algorithms could be used [161] to choose an appropriate value of $\lambda_1$. Another formulation is to set $\lambda_1 = 1$, which leads to a solution identical to ACC:

$$G = [H_B^H H_B + H_D^H H_D]^{-1} H_B^H H_{B,des} \tag{5.15}$$

this because the target pressures in the bright zone is an ACC solution $H_{B,des} = H_B G$ and because the same constraints are met.

The advantage of PM approach is [160]: it gives an explicit solution and does not require solving an eigenvector problem, it is suitable when several constraints are imposed on each sound zone, using one of the convex optimization algorithms as in [161] and finally, this algorithm can impose a constrain in the phase of the target pressure. The condition to achieve a solution is that the number of control points is greater than the number of sources [145].

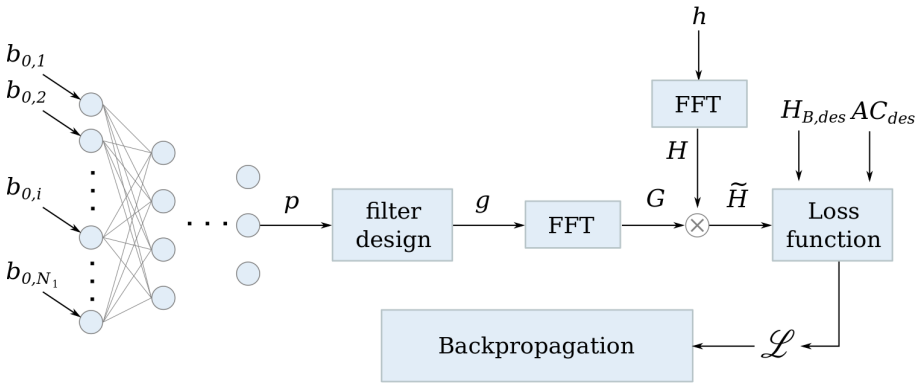



Figure 5.5: Scheme of FIR filter design for PSZ.

## 5.4 Proposed Method

In literature, to the best of our knowledge, for this task, no work has been found where Machine Learning techniques are used for filter design to get the bright and dark zones.

PSZ is an optimization problem similar to the Multipoint Audio Equalization task. Thus the Deep Optimization network is similar as described in Section 4.2.3 and 4.3.2: the first work is presented in [162], which a CNN is implemented to optimize FIR filter coefficients, achieving a double task, the desired spectrum in the bright zone and low energy in the dark zone. The input comprises a 3D matrix of the measured impulse responses, followed by the convolutional and fully connected layers. Finally, the output layer is composed of a fully connected layer of length $\mathcal{S} \times \mathcal{T}$, giving the optimized FIR filter coefficients.

Further studies have been performed, implementing the BiasNet to optimize FIR filter coefficients and parameters for Parametric IIR filters. Compared to the Multipoint Audio Equalization task, the only differences are the loss functions.

### 5.4.1 FIR Filter Design for Personal Sound Zones

For the FIR filter design using Deep Learning techniques, improvements have been made over the previous task. The neural network optimizes the coefficients to design the dark zone, equalize the bright zone, and achieve filters that present compact impulse responses. In Figure 5.5 is shown a scheme of the proposed method for FIR filter design for PSZ.

To achieve filters with compactness in the impulse responses, the network outputs are multiplied with a window function, which is a gaussian function

$w_g(\tau)$ calculated as:

$$w_g(\tau) = \sqrt{e^{\frac{(\tau-\tau_d)^2}{\sigma_f}}} \tag{5.16}$$

where $\tau_d$ is the delay, $\tau$ is the tap and $\sigma_f$ is the variance: the higher the variance, the larger the bell will be. In Figure 5.6 is presented an example of gaussian function when $\sigma_f$ is increased and when the maximum value is achieved at $\tau_d$ in the x-axis.

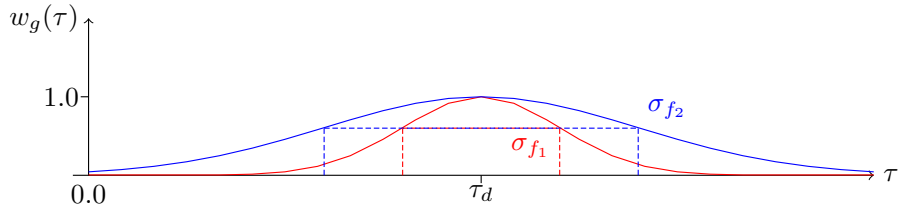



Figure 5.6: Gaussian function when the $\sigma_f$ is increased: the red line is the function when the standard deviation $\sigma_{f_1}$ is used, blue line when $\sigma_{f_2}$ is used.

The motivation behind this choice is that the neural network gives output values leading to non-compact impulse response; the other motivation is that the filters are constrained to be set to a delay provided by the user or the algorithm.

Once the network outputs are multiplied by the gaussian function, the simulation is started, and the loss functions are performed.

The first loss function is similar to Equation 4.25, with the difference that in this task, the cost function considers the microphone placed on the bright zone.

The second loss function regards the dark zone: to achieve a desired acoustic contrast $AC_{des}$, the Euclidean distance between the calculated Acoustic Contrast $\widetilde{AC}$ and the desired one is calculated. A weight function ensures to calculate the Acoustic Contrast on the desired one-third octave bands to optimize at high frequencies, as it is a punctual technique.

If the contrast in a one-third octave band exceeds the desired Acoustic Contrast weighted with the weighting function, then the error in that band will be zero:

$$C_i = \begin{cases} AC_{des,i} \cdot w_{AC,i} - \widetilde{AC}_i & \text{if} \quad \widetilde{AC}_i < AC_{des,i} \cdot w_{AC,i} \\ 0 \quad \text{if} \quad \widetilde{AC}_i \geq AC_{des,i} \cdot w_{AC,i} \end{cases} \tag{5.17}$$

Finally, the Euclidean distance is calculated:

$$\mathscr{L}_2 = \sqrt{\sum_{i=\omega_{l,1/3}}^{\omega_{h,1/3}} C_i^2} \tag{5.18}$$

The third loss function is calculated as the Equation 4.26 to solve the spatiality through the speakers, with the difference that in this task, the Euclidean distance of energy ratios is calculated using the microphones within the bright zone.

A penalty term is added to the total loss function: speakers present an operating frequency range, over which if the signal is amplified, the speaker could be damaged. With Parametric IIR filters, this problem is not present because the SOS's are placed within the frequency range.

A frequency mask is used to overcome this limitation: in Figure 5.7 is presented as an example. The magnitude frequency of the FIR filter at the bound frequency range is multiplied by the masking function $H_w$, defined as two linear functions with a decrease of a $\gamma$ set by the user.

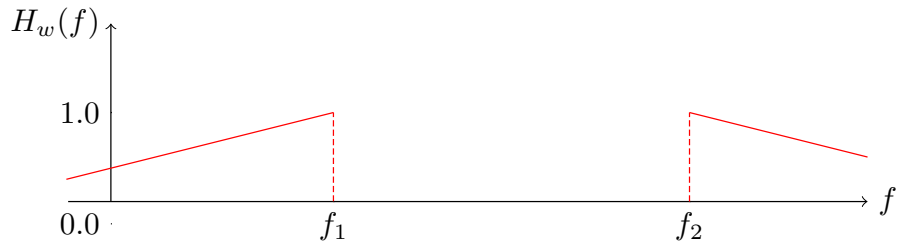



Figure 5.7: Example of masking curve used for a generic speaker. $f_1$ and $f_2$ are the operative frequency range.

The error occurs when the magnitude response exceeds the mask function:

$$C_\omega = \begin{cases} |G(\omega)| - H_w(\omega) & \text{if} \quad |G(\omega)| \geq H_w(\omega) \\ 0 \quad \text{if} \quad |G(\omega)| < H_w(\omega) \end{cases} \tag{5.19}$$

The loss function is the Euclidean distance of the errors occurred:

$$\mathscr{L}_4 = \sum_{s=1}^{S} \sqrt{\sum_{\omega=0}^{\omega=\omega_1} C_\omega^2 + \sum_{\omega=\omega_2}^{\omega=\omega_s/2} C_\omega^2} \tag{5.20}$$

A regularization term is added to the loss function to achieve a FIR filter with a compact impulse response. First, a weight function is multiplied by the square values of the filter. The weight function, shown in Figure 5.8, is

calculated as:

$$w_{L_5}(\tau) = 1 - w_g(\tau) \tag{5.21}$$

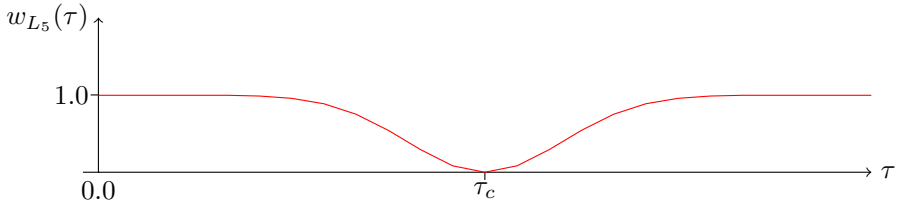



Figure 5.8: Example of weight function used to calculate the compactness of the impulse response of the FIR filter.

The sub-loss function will be given by the product of weight function and the square of the time impulse response of the filter, normalized with its energy:

$$\mathscr{L}_5 = \sum_{s=1}^{\mathcal{S}} \sqrt{\sum_{\tau=1}^{\mathcal{T}} (w_{s,\mathscr{L}_5}(\tau) \cdot g_s^2(\tau)/\epsilon_s)^2} \tag{5.22}$$

values far from the peak value will be emphasized, whereas those near the peak value will have less importance in calculating the loss function.

Finally, the last sub-loss function regards the compactness of the output impulse response. In order to reduce the compactness of the output impulse response, the loss function is calculated as in Equation 5.22, with the difference that we use the output impulse response and mask it on all samples:

$$\mathscr{L}_6 = \sum_{m=1}^{\mathcal{M}_B} \sqrt{\sum_{n=1}^{N} (w_{m,\mathscr{L}_6}(n) \cdot \tilde{h}_m^2(n)/\epsilon_m)^2} \tag{5.23}$$

The total loss function is given by the weighted sum of the six sub-loss functions:

$$\mathscr{L} = \gamma_1\mathscr{L}_1 + \gamma_2\mathscr{L}_2 + \gamma_3\mathscr{L}_3 + \gamma_4\mathscr{L}_4 + \gamma_5\mathscr{L}_5 + \gamma_6\mathscr{L}_6 \tag{5.24}$$

### 5.4.2 IIR Filter Design for Personal Sound Zones

In this thesis, the Deep Optimization method has been used to design Parametric IIR filters for the Personal Sound Zones. In literature, to the best of our knowledge, no works present PSZ with Parametric IIR filter design A depth study is conducted, analyzing the PSZ with 1 SOS for each one-third octave band, then passing to 5 SOS's per each band.

The network design is similar to the BiasNet used for the Multipoint Audio Equalization task with Parametric IIR filters. The differences consist only of the cost function. Similar to the previous Section, the loss function is the weighted combination of four sub-loss functions. The first is the Euclidean distance of the desired frequency response and the measured one, summed between the microphones used within the bright zone. The second is the sub-loss function used to define the dark zone, calculated as the previous Section. Last but not least, the ratio energy as defined in Equation 4.26. Finally, the last sub-loss function defines the output impulse response, as in the previous Section.

The total loss function is the weighted sum of four loss functions, defined as:

$$\mathscr{L} = \gamma_1 \mathscr{L}_1 + \gamma_2 \mathscr{L}_2 + \gamma_3 \mathscr{L}_3 + \gamma_6 \mathscr{L}_6 \tag{5.25}$$

Compared to the previous work, where IIR filters were used to equalize within the acoustic scene, in this task, the filters must both equalize in the bright zone and attenuate as much as possible in the dark zone.

For this reason, the number of SOS's for each band has been increased to improve the Acoustic Contrast. The performance, as will be described in Section 5.6, will also be compared in computational and perceptual terms. The SOS's have been added only in the frequency range of interest, then up to 2 kHz, to not have a large number of SOS's per speaker and because it is challenging to equalize and attenuate at high frequencies.

## 5.5 Experimental Setup

The experiments were performed using the Jeep Renegade scenario. Concerning the Multipoint Audio Equalization task, loudspeakers have been added to the car-manufacturer speakers to increase the Acoustic Contrast (see Figure 5.9): 2 speaker arrays were placed above the car dashboard, one in front of the driver seat, the other in front of the passenger position. In total, 16 full-range loudspeakers were added. The operative frequency range is 250-11000 Hz.

The impulse responses were measured using the exponential sine sweep method [130] with a sampling frequency rate of 48 kHz, using as audio interface an RME Madiface and a Dante-equipped amplification system. Two mannequins have been used, one for each seat. Optimization and evaluation have been performed with two several sets of binaural impulse responses. Indeed, the second set has been measured by varying the position of the mannequins on the seats.

The BiasNet optimizes FIR filters of 8192-th order, while for the Parametric IIR filter design, the number of SOS's to optimize goes to 1 for each one-third octave band to 5 SOS's. The ranges of the parameters are: $Q_{min}=$

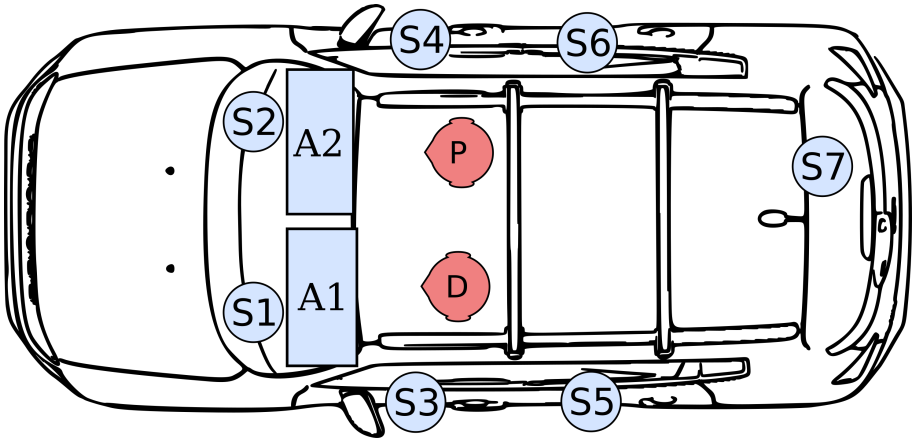



Figure 5.9: Jeep Renegade schematic with loudspeakers and microphones positions: A1 and A2 corresponds to the full-range speaker arrays. D and P stands for the binaural microphones on the driver and passenger seat, respectively.

0.05, $Q_{max}$= 10.0, $V_{0,min,dB}$= -20 dB, $V_{0,max.dB}$= 20 dB, $V_{s,min,dB}$= -40 dB, $V_{s,max.dB}$= 40 dB. Finally, to compare the performance between the Parametric IIR filters and FIR filters using the same number of coefficients, the FIR filter design of 512-th order is performed. In Table 5.2, the number of parameters to optimize is presented. Regarding the Parametric IIR filters, the BiasNet optimizes from 1,139 parameters to 4,115, whereas when the FIR filter coefficients are optimized, a total number of 188,416 parameters are designed.

| SOS's | nr. maximum parameters for each speaker | | | | | nr. maximum coefficients for each speaker | | | | | nr. parameters to optimize |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | W | T | WT | S | F | W | T | WT | S | F | |
| 1 | 31 | 37 | 76 | 19 | 52 | 60 | 36 | 150 | 36 | 102 | 1139 |
| 2 | 61 | 52 | 130 | 37 | 85 | 120 | 102 | 258 | 72 | 168 | 1883 |
| 3 | 91 | 67 | 184 | 55 | 118 | 180 | 132 | 366 | 108 | 234 | 2627 |
| 4 | 121 | 82 | 238 | 73 | 141 | 240 | 162 | 474 | 144 | 300 | 3371 |
| 5 | 151 | 97 | 292 | 91 | 174 | 300 | 192 | 582 | 180 | 366 | 4115 |
| $FIR_{8192}$ | 8192 | | | | | 8192 | | | | | 188416 |
| $FIR_{512}$ | 512 | | | | | 512 | | | | | 11776 |

Table 5.2: Number of maximum parameters and coefficients for each speaker, when IIR and FIR filters are used. The last column is the number of parameters (neural network outputs) to optimize. $FIR_{8192}$ stands for FIR filters of 8192-th order, whereas $FIR_{512}$ is the 512-th order.

The optimization with the neural approach was performed with *Python* and *Tensorflow*, while the ACC and PM have been run with *Matlab*.

Evaluation of proposed and baseline methods were performed in *Matlab*. Perceptual metrics, explained in Section 5.1, have been calculated in *Python* and *C++*.

The $AC_{des}$ was set to -50 dB, whereas the $|H_{B,des}|$ was equal to 0 dB. After a series of preliminary analyses, the weights of the loss functions, regarding the FIR filter design, were set to $\gamma_1 = 1265$, $\gamma_2 = 2$, $\gamma_3 = 1 \cdot 10^{-5}$, $\gamma_4 = 5000$, $\gamma_5 = 92$, $\gamma_6 = 2$. Instead, regarding the Parametric IIR filter design, the weights are $\gamma_1 = 142$, $\gamma_2 = 4$. $\gamma_3 = 1 \cdot 10^{-5}$ and $\gamma_6 = 1897$. The learning rate was set to $1 \cdot 10^{-4}$, the number of iterations is equal to 10,000, $\sigma_f$ is set to 48,000 and, finally, Adam algorithm is used as optimizer.

The weighting function is defined as 1 in linear scale on the one-third octave band of interest, while, in the out of range, it decreases by 6 dB per band.

To record the speech signals, 50 audio files from the LibriSpeech dataset [163] were selected. Perceptual metrics were analyzed for each microphone within the bright zone and audio file and then averaged.

## 5.6 Results

Experiments were performed when the bright zone was defined on the driver seat and the dark zone on the passenger seat, then when the bright zone was defined on the passenger and the dark zone on the driver spot. In Table 5.3 are presented the results when FIR and IIR filters are used. The ACC was used as a reference and not as a comparison because, as shown below with the perceptual analyses and as explained in Section 5, it achieved excellent acoustic contrast performance but poor perceptual results. In Table 5.3 the ACC results are reported in italic.

The ACC achieved the best results when the bright zone was defined on the driver seat (see Table 5.3.a), obtaining an $\overline{AC}_{ib}$ in the band of interest of 14.51 dB and an $AC_{max}$ of 21.17 dB. In contrast, the BiasNet obtained an $\overline{AC}_{ib}$ of 13.57 dB and $AC_{max}$ 23.22 dB. In Figure 5.10 is presented the one-third Acoustic Contrast. Finally, the PM was the worst performing technique in terms of AC, with an $\overline{AC}_{ib}$ of 12.07 dB and $AC_{max}$ of 17.89 dB.

When the bright zone is defined on the passenger seat, the BiasNet achieved results lower of almost 1 dB with respect to the ACC. Indeed, as shown in Table 5.3.b, the $\overline{AC}_{ib}$ is equal to 16.60 dB, while the BiasNet presents an $\overline{AC}_{ib}$ that is equal to 15.90 dB. The $AC_{max}$ of the ACC is 24.45 dB, while the BiasNet presents 22.26 dB of $AC_{max}$. The PM achieves the worst performance, with the $\overline{AC}_{ib}$ and $AC_{max}$ equal to 14.40 dB and 21.49 dB, respectively.

The IIR filters are compared with the FIR filters designed by the BiasNet: when the bright zone was defined on the driver seat, the IIR filters achieved the best performance, both for audio equalization and PSZ. Regarding the AC,

| Method | | Bright zone | Dark zone | | |
|---|---|---|---|---|---|
| | | $\overline{MSE}_B$ | $\overline{AC}_{fb}$ [dB] | $\overline{AC}_{ib}$ [dB] | $AC_{max}$ [dB] |
| IIR SOS per 1/3 octave band | 1 | $1.91 \cdot 10^{-1}$ | 8.53 | 12.68 | 19.71 |
| | 2 | $9.65 \cdot 10^{-2}$ | 8.37 | 13.06 | 23.86 |
| | 3 | $\mathbf{5.40 \cdot 10^{-2}}$ | 8.97 | 12.93 | 23.78 |
| | 4 | $5.48 \cdot 10^{-2}$ | 9.68 | **13.85** | 23.18 |
| | 5 | $5.80 \cdot 10^{-2}$ | 9.39 | 13.62 | **25.11** |
| FIR | PM | $1.04 \cdot 10^{-1}$ | **9.45** | 12.07 | 17.89 |
| | $\text{BiasNet}_{8192}$ | $2.25 \cdot 10^{-1}$ | 9.15 | 13.56 | 23.22 |
| | $\text{BiasNet}_{512}$ | $2.65 \cdot 10^{-1}$ | 8.82 | 12.00 | 15.40 |
| | ACC | *$2.89 \cdot 10^{-1}$* | *11.16* | *14.51* | *21.17* |

(a)

| Method | | Bright zone | Dark zone | | |
|---|---|---|---|---|---|
| | | $\overline{MSE}_B$ | $\overline{AC}_{fb}$ [dB] | $\overline{AC}_{ib}$ [dB] | $AC_{max}$ [dB] |
| IIR SOS per 1/3 octave band | 1 | $\mathbf{1.63 \cdot 10^{-1}}$ | 10.89 | 15.45 | **22.85** |
| | 2 | $1.73 \cdot 10^{-1}$ | 11.07 | 15.38 | 21.68 |
| | 3 | $2.19 \cdot 10^{-1}$ | 11.60 | **16.59** | 21.84 |
| | 4 | $1.66 \cdot 10^{-1}$ | 11.19 | 15.84 | 21.97 |
| | 5 | $1.72 \cdot 10^{-1}$ | 11.16 | 15.47 | 21.01 |
| FIR | PM | $1.97 \cdot 10^{-1}$ | 11.33 | 14.40 | 21.49 |
| | $\text{BiasNet}_{8192}$ | $3.13 \cdot 10^{-1}$ | **11.72** | 15.90 | 22.26 |
| | $\text{BiasNet}_{512}$ | $2.31 \cdot 10^{-1}$ | 10.39 | 14.54 | 20.41 |
| | ACC | *$2.56 \cdot 10^{-1}$* | *12.67* | *16.60* | *24.45* |

(b)

Table 5.3: Results for IIR filter design for PSZ and comparison with FIR filters of 8192-th order and 512-th order: (a) when the bright zone is defined on the driver seat and the dark zone on the passenger seat; (b) when the bright zone is defined on the passenger seat and the dark zone on the driver seat. Please note that the ACC results are used as reference and they were highlighted in italic. The best results with the other techniques have been highlighted in bold.

the best result was obtained by optimizing 4 SOS's per band of interest. The $\overline{AC}_{ib}$ is equal to 14.39 dB and the $AC_{max}$ equals 21.88 dB. Despite the best $\overline{AC}_{ib}$, the best $AC_{max}$ is achieved with 5 SOS's, equals 25.11 dB. Regarding the audio equalization in the bright zone, the best result is achieved when the BiasNet optimized 3 SOS's, with a $\overline{MSE}_B$ equal to $5.40 \cdot 10^{-2}$, even with the design of 4 SOS per band, the same results are obtained because the $\overline{MSE}_B$ is equal to $5.40 \cdot 10^{-2}$. Regarding the 5 SOS's per band, the $\overline{MSE}_B$ is equal to $5.80 \cdot 10^{-2}$.

Comparing 4 SOS's and FIR filters of 8192-th order using the BiasNet, the

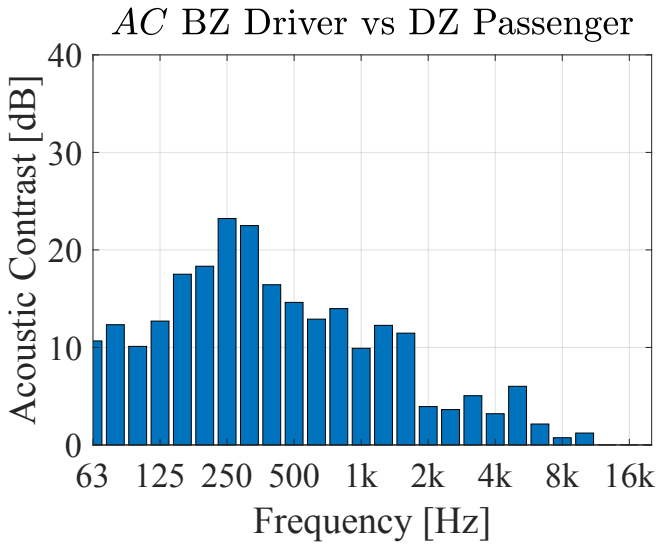



Figure 5.10: One-third octave band AC of BiasNet for FIR filter design

first achieved the best performance: the $\overline{AC}_{ib}$ is equal to 13.85 dB, while the FIR filters achieved 13.57 dB. Even with audio equalization in the bright zone, the same trend occurs. In fact, the $\overline{MSE}_B$ is 1 order of magnitude lower.

Regarding the experiments with the bright zone defined on the passenger seat, the same performance is achieved. In this case, 3 SOS's per band achieved the best $\overline{AC}_{ib}$, which is equal to 16.59 dB, instead, 4 SOS's per band achieved an $\overline{AC}_{ib}$ equal to 15.84 dB.

In Figures 5.11, 5.12, 5.13 and 5.14 are presented the results of the perceptual metrics. The PM achieved the best performance on overall perceptual

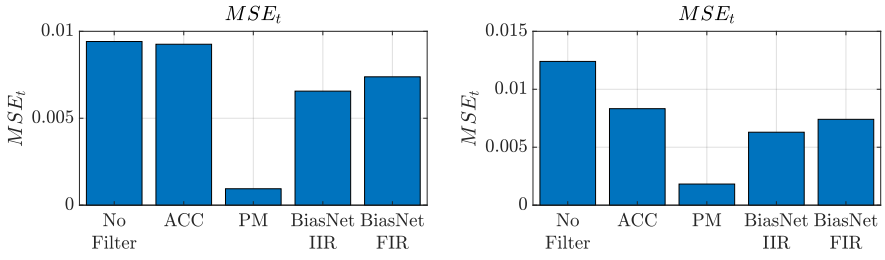
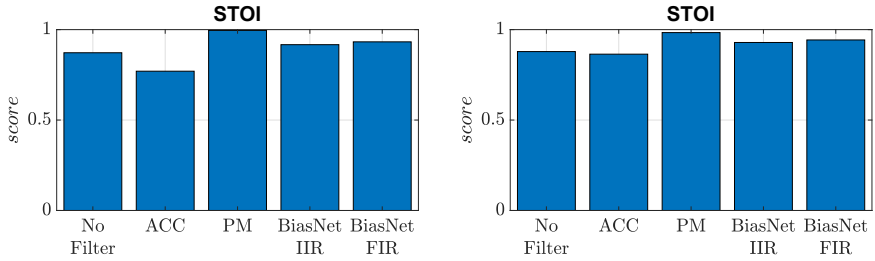



Figure 5.11: $MSE_t$ performance comparison when no filtering is applied (No Filter), using the ACC, PM, the best IIR filter design wuth the BiasNet (4 SOS's per band) and FIR filter design with the BiasNet.



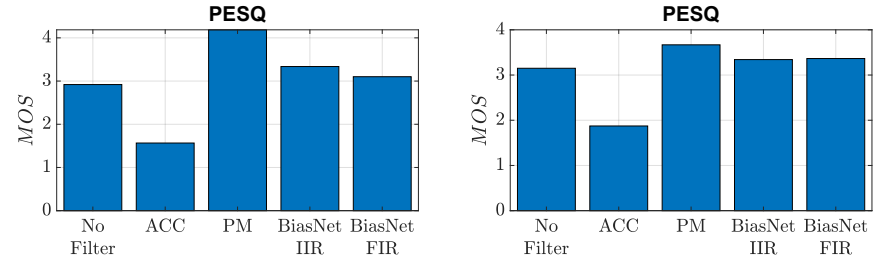

(a) (b)

Figure 5.12: STOI performance comparison when no filtering is applied (No Filter), using the ACC, PM, the best IIR filter design wuth the BiasNet (4 SOS's per band) and FIR filter design with the BiasNet.

tests, whereas the ACC obtained the worst. The BiasNet got an intermediate performance, but the results are better than no filter is used. Indeed, the PM achieved a STOI score equals 0.99 and 0.98; the MOS of the ViSQOL test is 4.17 when the bright zone is defined on the driver seat and 3.80 on the passenger seat. Finally, the PESQ MOS is 4.18 and 3.66 for driver and passenger positions, respectively. FIR filters optimized using the BiasNet achieved a STOI score of 0.96 for both positions, while ViSQOL and PESQ scores are 3.72 and 3.48 for the first metric and 3.56 for both the position for the second one.

Comparing the results between the FIR and IIR filter design, the second one achieved slightly better performance. Only the STOI score is better with the FIR filters.

Finally, some remarks about PSZ are made. Compared to the Multipoint Audio Equalization task, there have been improvements in the FIR filter design. Figure 5.15 shows an example of a compact FIR filter achieved by the optimization. Indeed, it presents a peak and few pre-ringing, thus at listening,




(a) (b)

Figure 5.13: PESQ performance comparison when no filtering is applied (No Filter), using the ACC, PM, the best IIR filter design with the BiasNet (4 SOS's per band) and FIR filter design with the BiasNet.

there are not many artefacts found in sound reproduction. Another important feature is the concentration of energy towards the peak. In this way, filters can be cut, resulting in filters with many fewer taps.

Other remarks are related to optimizing Parametric IIR filters. Despite being the first work, good results have been obtained, even outperforming the neural approach for designing FIR filters. In addition, comparing the computational cost, there is a significant cost reduction in the data processing. In Table 5.2 the number of coefficients used for each speaker are presented. The Woofer-Tweeter presents the high number of coefficients, with a total of 582 coefficients in the 5 SOS's per band experiments, but resulting in a lower computational cost with respect to the FIR filter design, in which the BiasNet optimized FIR filters of 8192 taps, and achieving the same Acoustic Contrast performance.

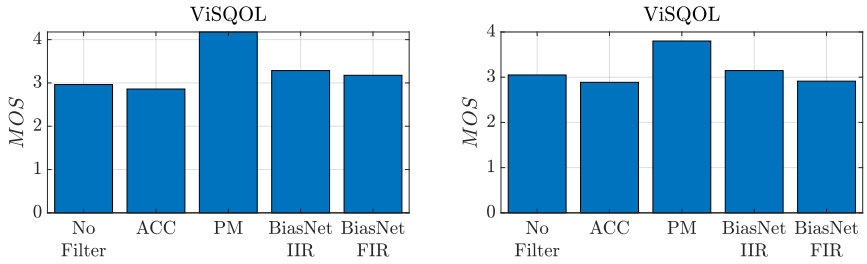



Figure 5.14: ViSQOL performance comparison when no filtering is applied (No Filter), using the ACC, PM, the best IIR filter design wuth the BiasNet (4 SOS's per band) and FIR filter design with the BiasNet.

Comparing the results of Parametric IIR filters and FIR filters of 512-th order in Table5.3, the IIR filters achieved the best performance with a lower number of coefficients concerning the FIR filter. Indeed, as shown in Table 5.2, using Parametric IIR filter with 1 SOS per band, the total number of operations is 225. The $\overline{AC}_{ib}$ is 12.68 dB and 15.45 dB for the bright zone defined on the driver and passenger seat, respectively. Using FIR filters of 512 coefficients, which the number of instructions is 513, the $\overline{AC}_{ib}$ is 12.00 dB and 14.54 dB, respectively, as shown in Table 5.3. Therefore, the results are lower by more than 1 dB with respect to the IIR filters.

Very high $Q$ values are obtained in some SOS's, resulting in a frequency response with very thin peaks or notches and, thus, obtaining impulse responses that could present oscillations. In Figure 5.16 is shown the magnitude response of an IIR filter and its impulse response: the filter presents many notches, with an amplitude response that attenuates down to -50 dB (see 5.16.a), while the impulse response of the filter has several oscillations that continue for hundreds of samples (see 5.16.b). From 50 cascaded SOS's that compose this filter, 5 of

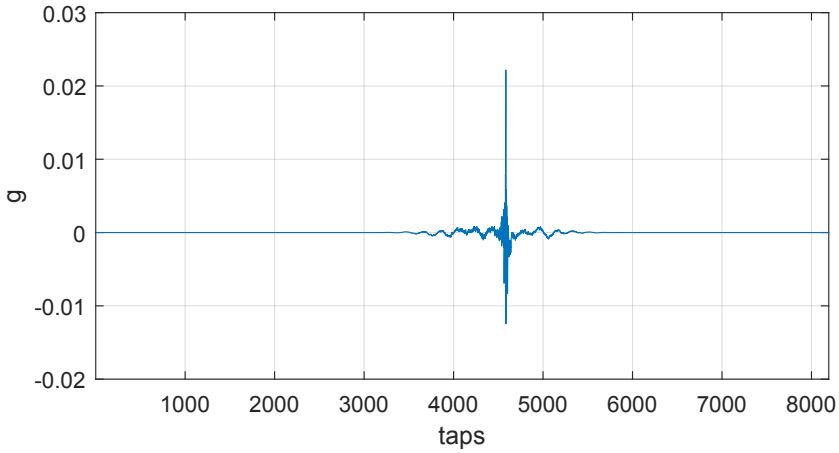



Figure 5.15: Time impulse response of the FIR filter of a full-range speaker

them have a $Q$ greater than 9.

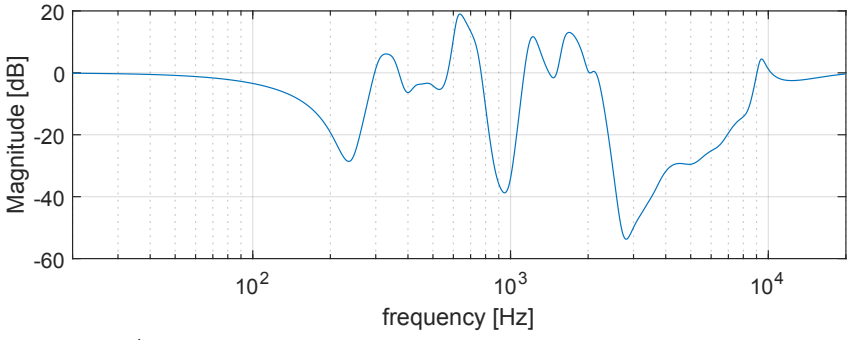
## 5.7 Final Remarks

In this chapter, Deep Optimization techniques are proposed to design IIR and FIR filters for PSZ. The DNNs are used to optimize digital filters using loss functions and regularization and penalty terms. The proposed approach is compared with two baseline techniques, the ACC and the PM. The first method usually achieves high AC performance but obtains filters that introduce artifacts. Indeed, the evaluated perceptual metrics using the filters optimized with this technique achieved low results. For this reason, the ACC results have been used as a reference.

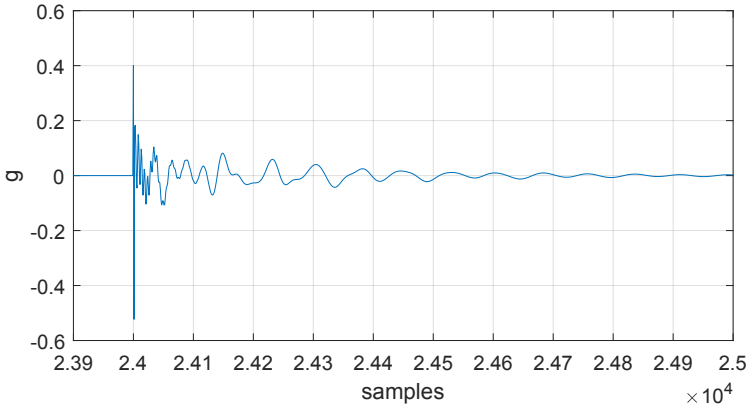
Experiments were performed in a vehicle scenario, adding two speaker arrays to the car-manufacturer loudspeakers. The evaluation was performed using binaural impulse responses located differently from the ones used for the optimization in order to obtain an Acoustic Contrast optimized in the whole area.

The results are promising, even if the Acoustic Contrast is a less than ACC of almost 1 dB, but the baseline technique achieved poor audio equalization and perception results. Comparing the results with the PM, the neural approach achieved the best performance. Instead, the $\overline{MSE}$ and the perceptual metrics are slightly worse.

Significant results concern the optimization of Parametric IIR filters, not present in the literature for this task. Despite being the first work, promising




(a) Magnitude response of the IIR filter.




(b) Impulse response in time domain of the IIR filter

Figure 5.16: Magnitude response of an IIR filter (a) and its impulse response in time domain (b).

results are obtained, even outperforming the neural approach for designing FIR filters. The audio equalization performance of the proposed method is better than the PM technique, and the perceptual results are similar to the FIR filter neural approach. In addition, comparing the computational cost, there is a significant cost reduction in the data processing.

# Chapter 6

# Other Contributions

## 6.1 Road Type Classification Using Deep Learning Models

Nowadays, cars are the main topic of interest in many aspects. The next generation of cars will become increasingly automated, therefore in addition to improving comfort in the cabin, there is an increasing interest in safety research, i.e. in new, cheaper and more reliable sensors [164].

Road conditions play an important role in intelligent cars safety systems and autonomous vehicles [165], keeping an autonomous safe distance based on the road conditions. Furthermore, this topic could allow novel scenarios in the car cabin, as intelligent speech enhancement, audio equalization, active noise control [166].

As reported in [167], the weather is the primary factor of car accident numbers, which doubles when the asphalt is wet [168]. Regarding cabin comfort, tyre-road noise is a factor of vehicle noise emissions that could affect driver's concentration and could cause annoyance on passengers [169].

Acoustic sensors have been explored for road wetness and roughness classification [166, 170], suggesting that combining them with Machine Learning techniques is possible to replace expensive optical sensors, radar and lidar systems with inexpensive ones, integrating them with the infotainment system for automatic equalization and speech enhancement [171].

Microphones have been used in combination with the Support Vector Machine (SVM) for road roughness [172] classification, extracting acoustic features as the MFCC. In [173] road wetness classification is employed using SVM and one-third octave band filters, while in [170] a Bi-Directional Long Short Term Memory (BLSTM) neural network [174] is used in combination with the Auditory Spectral Features (ASF), achieving better performance than the SVM.

### 6.1.1 Contribution

Motivated by the works explained below, the main objective of this research is to study the Deep Learning techniques for road roughness and wetness detection. The first work was to use a CNN network for the road roughness classification [166], extracting ASF to feed the network. The database was built driving the car in some areas of Ancona and on the highway, recording the sound with a multi-channel microphone arrangement. Better results were achieved with Dual-Channel CNN and Siamese Neural Network (SNN), using cross-validation and the cross-set between summer and winter tyres [171]. For road wetness detection, CNN was used, comparing it with the BLSTM [164], achieving results slightly inferior to those of the recurrent network but with much faster processing (training and testing) times.

The microphone positioned near the driving plate was exposed to the weather and damaged, while the microphone placed inside the trunk obtained the best results because it was able to isolate well the external noises and those inside the car cabin. Finally, the last work compares two CNN architectures on roughness and wetness detection, implementing them on a DSP system [175] and analyzing the results between GPU and DSP and the processing times of the latter. In Figure 6.1, the flux diagram is presented: Features are extracted from the signal recorded by a microphone, then features are normalized, and finally, the network is processed to predict what type of asphalt the car is driving on.


Figure 6.1: General scheme of roughness and wetness detection.

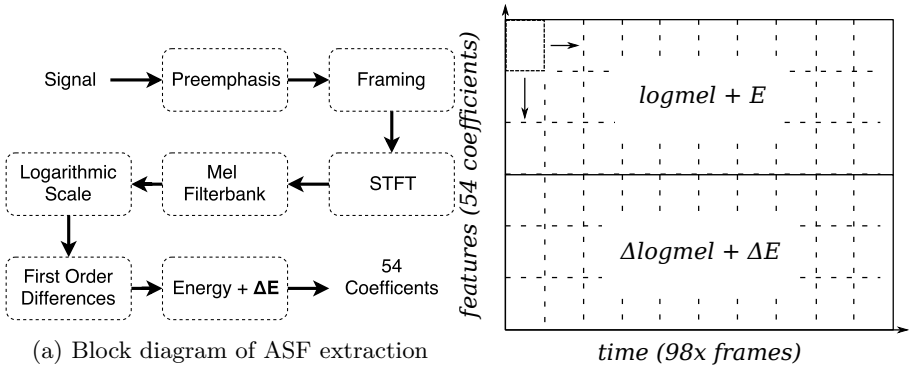
### 6.1.2 Auditory Spectral Features

ASF are achieved by calculating the STFT to the audio samples, using a frame size of 30 ms and a frame step of 10 ms. Each STFT contains the power spectrogram, which must calculate the Mel frequency scale using a filter-bank with 26 triangular filters. To match the human perception of loudness, Mel spectrograms $M_{30}(n,m)$ are transformed to a logarithmic scale:

$$M_{log}^{30}(n,m) = log(M_{30}^{30}(n,m) + 1.0) \tag{6.1}$$

ASF contain the first order differences of each LogMel spectrogram:

$$D_{30}(n,m) = M_{log}^{30}(n,m) - M_{log}^{30}(n-1,m) \tag{6.2}$$




(a) Block diagram of ASF extraction

(b) Representation of an ASF chunk.

Figure 6.2: Auditory Spectral Feature extraction process and representation. In Figure 6.2a is presented the block diagram; in Figure 6.2b the representation of ASF of a chunk is shown.

Finally, the frame energy and its derivative are added to the feature vector. In total, the feature vector is composed of 54 coefficients: 26 LogMel coefficients, 26 first order differences LogMel coefficients, the energy and its derivative. An audio chunk of 1 s is used. Thus 98 feature vectors are calculated, resulting in a 2D audio image of dimension $54 \times 98$.

In Figure 6.2 the block diagram extraction and the representation of the ASF are presented. Feature extraction (see Figure 6.2a) was processed using the toolkit openSMILE v2.3.0 [176], except in a part of [175], where the DSP system was used for the ASF extraction and network process.

### 6.1.3 Preliminary Analysis

In this project, a multichannel microphone arrangement is used to exploit microphone diversity and improve the classification or to conduct the evaluation at once. Many positions have been analyzed to place the microphones both inside and outside the car [164, 166].

A Mercedes A Class from 2014 was used to build the dataset and to place the microphones. A multichannel front end, HEAD Acoustics Squadriga II, has been used as an acquisition device to record 8 channels at different sample rates and store GPS and CAN bus signals. The audio signals were sampled at 44100 Hz at 24-bits.

External microphones are *PCB Piezotronics* model 103A24, which are IP55 microphones. These transducers have been protected with a melamine resin foam for sound absorption to reduce the effect of wind. The internal microphones are *PCB Piezotronics* model 378C20.

The first analysis was performed by placing two microphones close to the rear

wheels, one in front of the front left wheel, one inside the engine compartment and two inside the cockpit, close to the driver's head and close to the right passenger head. In Figure 6.3 microphone positions are shown. The front-right microphone has been excluded from recordings after preliminary evaluations because it records a large amount of engine noise with respect to the other microphones. The rear-right microphone had the lowest noise coming from the engine. Finally, the engine compartment microphone has been used to record the engine conditions for future works.

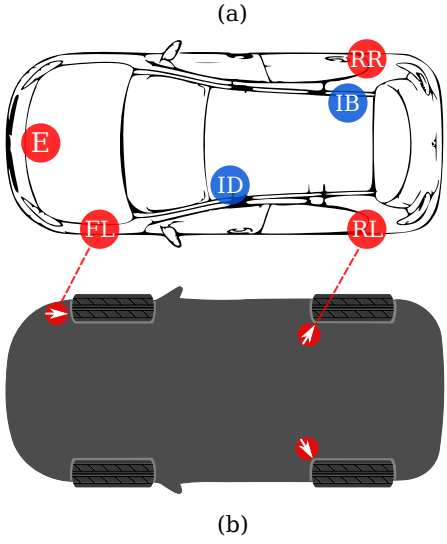



Figure 6.3: Position of microphones for road roughness detection [166]. Top view (a) and bottom view (b). The microphones are placed in the engine compartment (E), close to the front-left, rear-left and rear-right (FL, RL and RR, respectively), inside the car: close to the driver (ID) and in the back seat (IB). The arrows in (b) show how the capsule was positioned to minimize the wind effect. The rear microphones are protected in the wheelhouse.

The magnitude response of 1 s of two audios recorded on smooth and rough asphalt are shown in Figure 6.4. The major differences between the two audios are found at 400 Hz.

For road wetness detection, other microphone positions have been studied [164]. External microphones had attenuation problems caused by the wet foam: in Figure 6.5, the difference between the frequency response when recording on the wet road using a dry and a wet foam is shown.

For this reason, in [164] the microphones are not close to the tires. In Figure 6.6 microphone placements are shown: as above, two microphones inside the car cabin are placed, one close to the driver position and the other to the rear

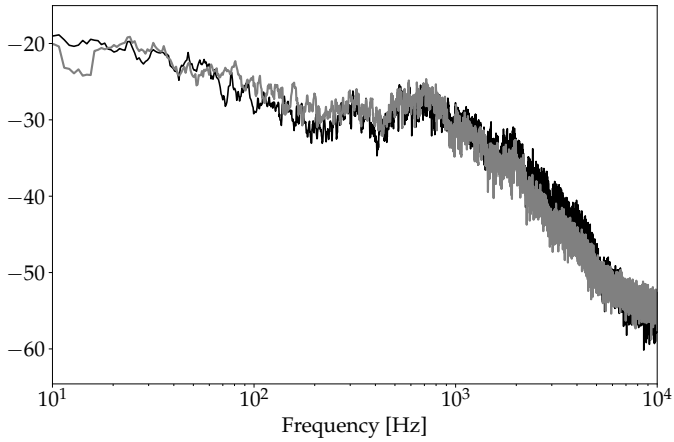



Figure 6.4: DFT from two 1 s of smooth (black line) and rough (gray line) road.

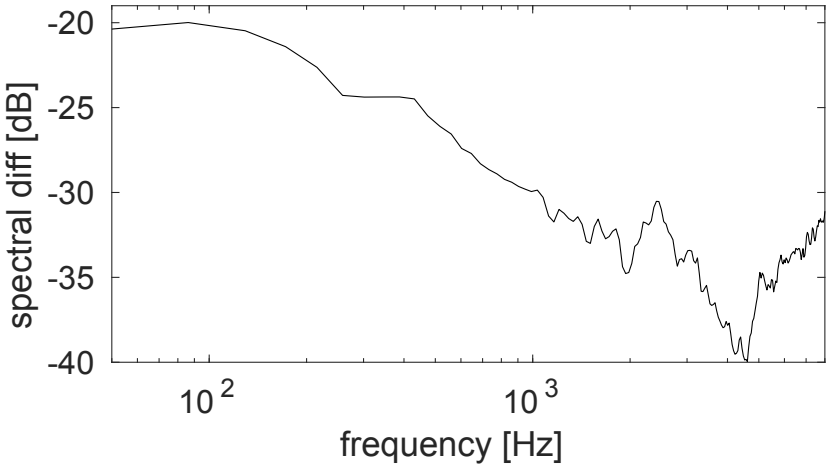



Figure 6.5: Spectral difference between smooth and rough pavement frequency response.

passenger position. One microphone is positioned in the bottom of the trunk, below the spare wheel. Externally, one microphone was positioned below the trunk hatch, near to the driving plate.

The microphone placed inside the trunk isolates noise from the exterior and the cabin remarkably well. In Figure 6.7, the measured noise attenuation is shown: a white noise source has been placed in cabin front row, recording the signals at the ID and T microphone, achieving a mean octave band attenuation in the range 40 Hz - 10 kHz of 21 dB with respect to ID microphone.

Since the microphone placed inside the trunk is protected from weather and

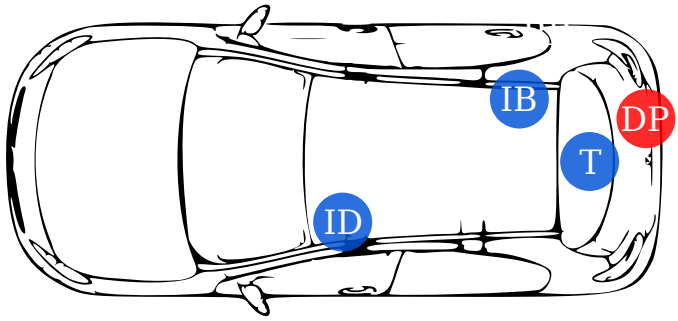



Figure 6.6: Position of microphones for road wetness detection [164]. The microphones are placed inside the car: close to the driver (ID) and back seat (IB) and below the trunk (T). Outside the car, one microphone is located below the trunk hatch, near the driving plate (DP).

wind and also manages to isolate the noise coming from the cabin, it could be the most promising for road wetness and, from a commercial point of view, it could manage to attenuate the sound inside the passenger compartment, e.g. music and speech.

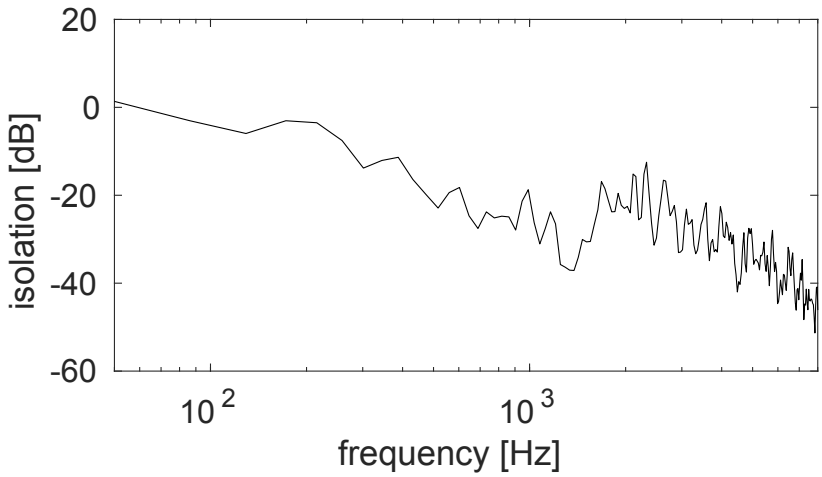
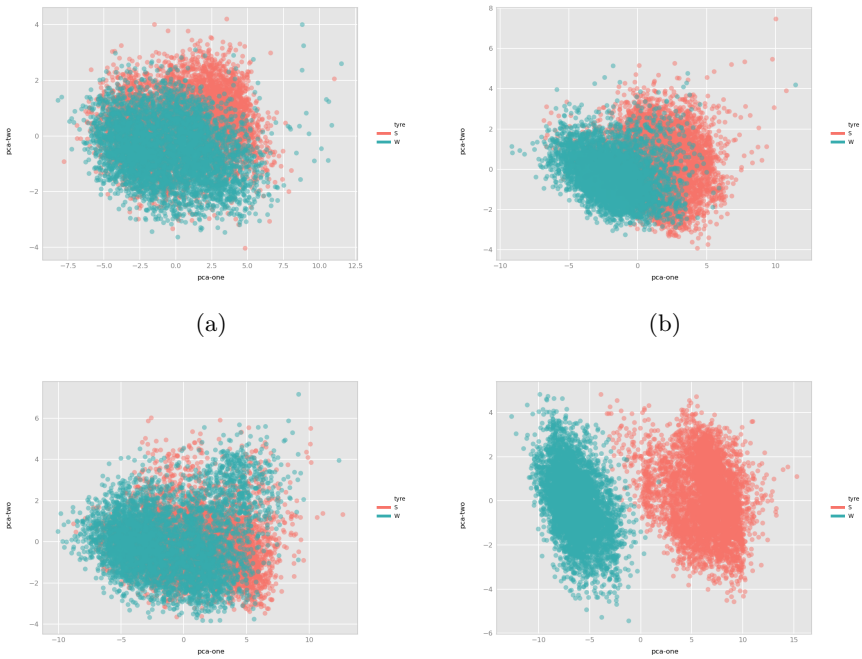



Figure 6.7: Measured noise attenuation between the driver and the trunk microphone in the frequency range (50 Hz and 8 kHz) achieved by the difference of the two log magnitude spectra.

As shown in Section 6.1.5, DP microphone achieved low performance be-

cause of the windshield wiper noise that interferes with the classification in wet conditions. The wet recordings were extremely different from the recordings of the dry sessions in terms of levels and spectral profile due to the foam soaking. The position of DP microphone is shielded from wind, but it is not completely repaired from the rain, thus the microphone foam protection is soaked.

In Figure 6.8 the 2D Principal Component Analysis (PCA) is shown. Red and blue dots correspond to summer and winter tyres. PCA in two different groups clusters the driving plate recordings from the wet sessions (see Figure 6.8b), instead of from the dry recordings sessions shows a strong overlap (see Figure 6.8a). The back seat position signals from the same recording sessions are totally overlapped (see Figure 6.8c and Figure 6.8d)). The same overlapping is presented on the other internal microphones, suggesting an issue with the DP microphone cross-domain, related to the recording conditions, probably the microphone from soaking.




(a) (b)

(c) (d)

Figure 6.8: 2D PCA discriminating the summer and winter tires, described by red and color dots, using ASF. (a) and (b) represent the dry and wet roads, respectively, using the back seat microphone. (c) and (d) represent the dry and wet roads, respectively, using the driver plate microphone. The PCA axis ranges have no physical meaning, thus they are not reported.

### 6.1.4 Road Roughness Classification

The first work that has been conducted concerns the study of Deep Learning techniques for road surface classification [166]. Convolutional Neural Network has been used because of its low computational cost and low memory usage with respect to the recurrent neural network, useful when implementing in a DSP system. The dataset is not balanced and it is composed of 41% of rough road samples and 59% of smooth road samples, with a total of 50 minutes of recording signals.

A 5-fold cross-validation procedure has been performed, combining it with a random search of the CNN hyperparameters. The metrics have been determined by averaging them across all cross-set. Several optimizers were used, but the best was the Adam optimizer [166].

In Table 6.1 are reported the best configurations tested, while in Table 6.2 the best five results are presented. Best performance is achieved with configuration 10, with an F-score equal to 86% and an Accuracy of 87.1%.

| Conf. | Filters | Kernel Size | Strides | Max Pooling | FCL Size |
|---|---|---|---|---|---|
| 1 | [20, 20] | $[3 \times 5], [1 \times 2]$ | $[3 \times 3], [1 \times 2]$ | y, y | [200, 100] |
| 2 | [15, 20] | $[3 \times 5], [1 \times 2]$ | $[3 \times 3], [1 \times 2]$ | y, y | [200, 100] |
| 3 | [30, 20] | $[3 \times 5], [1 \times 2]$ | $[3 \times 3], [1 \times 2]$ | y, y | [200, 100] |
| 4 | [15, 20, 30] | $[3 \times 5], [2 \times 2], [1 \times 4]$ | $[3 \times 5], [2 \times 2], [1 \times 4]$ | n, y, n | [300, 100] |
| 5 | [20, 20, 30] | $[1 \times 7], [9 \times 1], [3 \times 7]$ | $[1 \times 7], [1 \times 9], [2 \times 2]$ | n, n, n | [300, 100] |
| 6 | [20, 20, 30] | $[1 \times 7], [9 \times 1], [3 \times 7]$ | $[1 \times 7], [1 \times 9], [2 \times 2]$ | n, n, n | [600, 200] |
| 7 | [20, 20, 30] | $[1 \times 7], [9 \times 1], [3 \times 7]$ | $[1 \times 7], [1 \times 9], [2 \times 2]$ | n, n, n | [600, 100] |
| 8 | [54, 54, 30] | $[1 \times 7], [9 \times 1], [3 \times 7]$ | $[1 \times 7], [1 \times 9], [2 \times 2]$ | n, n, n | [200, 100] |
| 9 | [15, 20, 30] | $[3 \times 3], [2 \times 2], [1 \times 4]$ | $[3 \times 1], [2 \times 2], [1 \times 4]$ | n, y, n | [200, 100] |
| 10 | [20, 20, 30] | $[3 \times 3], [2 \times 2], [1 \times 4]$ | $[3 \times 1], [2 \times 2], [1 \times 4]$ | n, y, n | [200, 100] |
| 11 | [20, 20, 30] | $[3 \times 3], [2 \times 2], [1 \times 4]$ | $[3 \times 1], [2 \times 2], [1 \times 4]$ | n, y, n | [300, 100] |
| 12 | [30, 20, 30] | $[3 \times 3], [2 \times 2], [1 \times 4]$ | $[3 \times 1], [2 \times 2], [1 \times 4]$ | n, y, n | [200, 100] |

Table 6.1: Best tested configurations for CNNs. The kernel size and the stride are expressed as $[features \times time]$. FCL stands for Fully Connected Layers. In Max Pooling column, the term "y" and "n" represent the presence or not of Max Pooling Layers.

| Configuration | Accuracy (%) | F-measure (%) | Recall (%) | Precision (%) |
|---|---|---|---|---|
| 10 | 87.10 | 86.00 | 93.08 | 79.92 |
| 5 | 86.11 | 85.19 | 93.83 | 78.00 |
| 2 | 86.18 | 85.19 | 93.38 | 78.31 |
| 1 | 85.88 | 84.87 | 93.41 | 77.76 |
| 7 | 85.28 | 83.85 | 89.04 | 79.24 |

Table 6.2: Top 5 configurations sorted by performance obtained in cross-validation analysis with unbalanced training classes. The configuration numbers are the same reported in Table 6.1.

Further studies have been conducted on convolutional architectures to learn

a more robust representation when different tyre types are mounts. A Siamese approach and a Dual-Channel Convolutional Neural Network are compared with the Single-Channel CNN.

The use of SNN is due to the increase in the robustness of spectral changes, which are independent of the road surface conditions. Thus, if the method is robust to variations in the tyre model, the neural network does not need a large training corpus to be employed in a real-world case. The SNN calculates the degree of similarity between two inputs: it is provided of two inputs and is expected to evaluate a distance $d$ in the range [0, 1]. The reference input must belong to a known class and a threshold must be applied to the output. The SNN discriminates against whether the inputs belong to the same class or not. The SNN is composed of two identical networks, taking one input each, as shown in Figure 6.9.

The training needs to minimize the distance for the positive or same class examples and maximize the negative examples. The contrastive loss is used, calculated as:

$$\mathscr{L} = (1 - Y) \cdot \frac{1}{2} d^2 + Y \frac{1}{2} \cdot max(0, m - d)^2 \tag{6.3}$$

where $m$ is called margin and it is positive and allows only negative examples to loss calculation when the distance is less the radius defined by $m$.

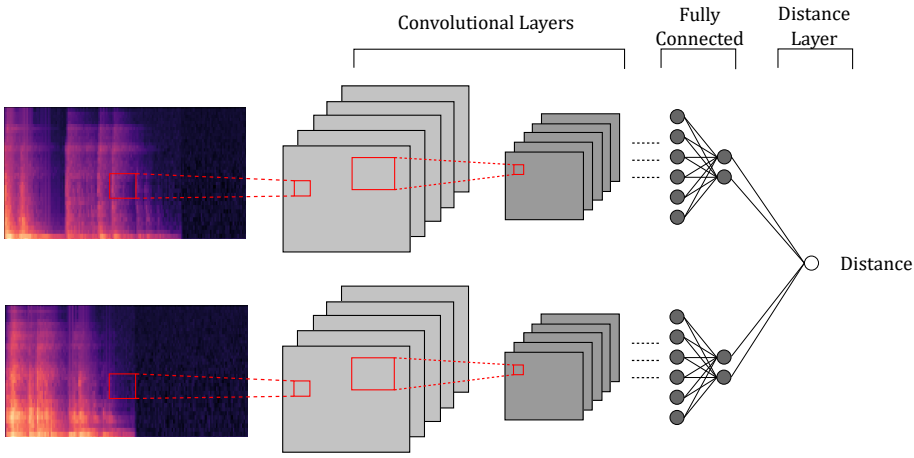



Figure 6.9: Siamese Neural Network scheme.

The temporal correlation between successive input frames is exploited using a sequence of $L$ chunks and calculating the distance between the current chunk and each of the previous $L$ chunks. The distances are then averaged according it to the following expression $d_i = 1/l \cdot \sum_{l=1}^{L} d_{i,l}$, where $i$ is the current chunk. Finally, the mean distance is processed to get a binary classification, using a

low-pass filter to the mean distance to a more stable and consistent value across frames and then applying a threshold. The diagram is shown in Figure 6.10.

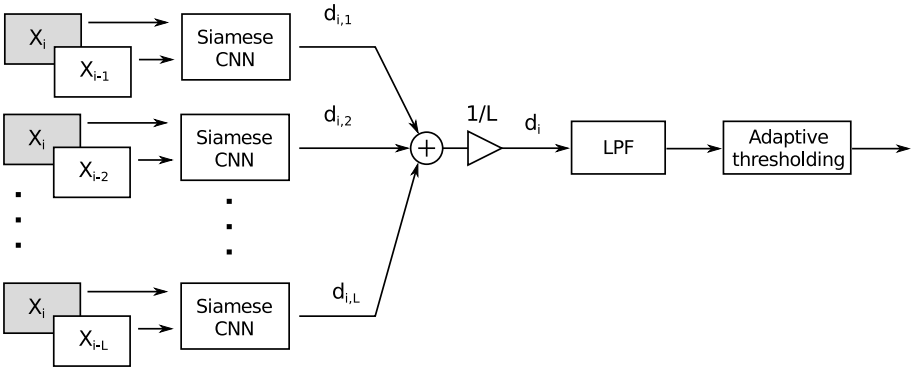



Figure 6.10: Diagram od the proposed algorithm. For each frame, the SNN run $L$ times, one for each input pair $(x_i, x_{i-l})$ with $1 < l < L$.

A new dataset was recorded for the experiments, resulting in 95 minutes of recording, with 53% of smooth asphalt and 47% of rough roads, using winter and summer tyres.

The experiments were performed by training and testing the networks with the same tyre types or training on a tyre type and testing on the other kind. A 5-fold cross-validation procedure has been performed using 64% of the dataset for training, 16% to validation and 20% to test, thus 3640, 864 and 1080 ASF spectrograms for training, validation and test set.

The metrics have been calculated for each training/testing set combination and then averaged to achieve the un-weighted average metrics. The F1-score has been used for comparison, averaged between each test fold. Early Stopping is used to reduce machine run-time, with a maximum of 1000 epochs and 100 patience epochs.

The best results of Single-Channel CNN is reported in Table 6.3. The performance decrease when the testing set is recorded with a tyre type unseen during the training. Using some tyres during training reduces the performance.

In Table 6.3, the results with Dual-Channel CNN are presented. Best results than Single-Channel CNN are achieved, except in the Summer-Winter case, where the performance is low, with $F1$-score equal to 72.35% respect to 76.17% of Single-Channel CNN.

The best results presented in Table 6.3 are achieved with the SNN in all training/test cases. In particular, the mixed training/testing cases achieved the same performance with Summer-Summer and Winter-Winter ones.

SNN seems to be the most robust neural architecture to unseen tyres in road roughness classification, achieving the best results in the overall train-

ing/testing set. In Table 6.3, a brief resume is shown: the SNN achieved an averaged F1-score of 95.58%, while the Single-Channel and Dual-Channel CNN achieved an $F1$-score equal to 81.6% and 85.24%, respectively. The Dual-Channel CNN achieved better results than Single-Channel CNN, due to the vulnerability of the Single-Channel CNN to temporary noise sources and temporary changes of the acoustic paths: the first could interfere with the classification task (car horns, construction sites etc.), and the second is present on walls, sidewalks, and other reflective surfaces that may affect the spectral characteristics of the road noise.

| Train/Test | CNN (%) | Dual CNN (%) | Siamese CNN (%) |
|---|---|---|---|
| W/W | 85,65 | 89,57 | 98,14 |
| S/S | 80,65 | 94,58 | 94,69 |
| W/S | 83,93 | 84,46 | 95,08 |
| S/W | 76,17 | 72,35 | 94,40 |
| Average | 81.6 | 85.24 | 95.58 |

Table 6.3: Comparison of best F1 Scores (%) and their average obtained with single channel CNN, dual channel CNN and Siamese CNN with different Train/Test tyre combinations (S = summer, W = winter).

Summarizing, if the double of the computational cost is feasible, the SNN outperforms the Dual-Channel CNN.

### 6.1.5 Road Wetness Classification

The road wetness detection was studied towards the end of the work on road roughness detection. In addition to the new microphone placement study, a new dataset was created for this task, recording on different tyre types (summer and winter) and in different weather conditions, in which the asphalt could be wet or dry.

The recordings amount to 146 minutes of audio, which 37% on wet roads. A 5-fold cross-validation procedure has been performed, disposing 64% of the dataset to train, 16% to validation and 20% to test. The metrics were calculated for each combination of training/testing and then averaged to achieve the unweighted average metrics.

The training was performed using an early stopping method, with a patience of 10 epochs and a number of epochs equal to 1000. The Adam optimizer and binary cross-entropy loss function are used.

Convolutional Neural Network is compared with the BLSTM. CNN was deployed in a random search, while the used BLSTM architectures are described

in [170].

In Table 6.4, the best results are presented: the CNN model can perform very well when training and testing sets are recorded with the same type of tyres, reaching up to 99% F1-score. Results are slightly lower with the Summer/Winter and Winter/Summer datasets due to different datasets. The best performing microphone is in the trunk, confirming the hypothesis that the distance from the engine and the protection from wind and rain may help. Its performance is shortly followed by that of the back seat microphone. Driver microphone achieved low performance, probably because of the windshield wiper noise that interferes with the classification in wet conditions. As described above, the DP microphone achieved the worst performance due to the differences in wet recordings; thus, the CNN is unable to classify frames from the domain unseen during training correctly. In the Summer/Summer and Winter/Winter case, the issue is alleviated because the F1-scores are 98.84% and 92%, respectively, going from no foam soaking issue to frequent soaking foam.

| | | | CNN | | | |
|---|---|---|---|---|---|---|
| Train/Test | Mic | F1-score | CNN Layer Sz | Kernel Shape | Strides Shape | Dense Layer Sz |
| W/W | DP | 98.94% | 20,20 | [[10,6], [10,2]] | [[3,3], [4,2]] | 1000,900 |
| W/W | IP | **99.15%** | 20,20 | [[3,6], [8,3]] | [[4,2], [3,5]] | 100, 600 |
| W/W | ID | 67.25% | 30, 20 | [[5,10], [1,4]] | [[4,4], [6,2]] | 900, 800 |
| W/W | T | 96.00% | 30, 20 | [[4,8], [4,5]] | [[5,1], [1,5]] | 400, 300 |
| W/S | DP | 1.00% | 30,20 | [[7, 10], [4, 5]] | [[4,1], [4,3]] | 700, 900 |
| W/S | IP | 94.67% | 20,20 | [[3,8], [6,1]] | [[2,5], [7,3]] | 300, 800 |
| W/S | ID | 74.00% | 30,20 | [[5,10], [1,4]] | [[4,4], [6,2]] | 900, 800 |
| W/S | T | **95.00%** | 30, 25 | [[1,1], [9,2]] | [[1,8], [8, 1]] | 800, 600 |
| S/W | DP | 9.00% | 20,20 | [[9, 7], [7, 10]] | [[1,2], [6,1]] | 100, 800 |
| S/W | IP | 96.76% | 20, 25 | [[10, 8], [7, 3]] | [[3,3], [7,5]] | 900,300 |
| S/W | ID | 62.00% | 25, 30 | [[9, 2], [6,6]] | [[3,3], [4, 3]] | 100, 100 |
| S/W | T | **97.33%** | 20, 25 | [[10, 8], [7, 3]] | [[3,3], [7,5]] | 900,300 |
| S/S | DP | 92.00% | 20, 25 | [[6, 7], [5, 1]] | [[1,4], [7,2]] | 800, 400 |
| S/S | IP | 98.40% | 30,30 | [[4,9], [2,3]] | [[4,1], [2,6]] | 1000,600 |
| S/S | ID | 96.24% | 20, 25 | [[4,5], [3,5]] | [[4,1], [3,5]] | 900, 700 |
| S/S | T | **99.38%** | 20, 20 | [[3,1], [1,7]] | [[2,3], [3,3]] | 800, 400 |

Table 6.4: Best performing CNN models from the tests. Training and testing have been conducted on summer (S) and winter (W) tires, with driver plate (DP), back seat passenger (IP), driver (ID) and trunk (T) microphones.

The BLSTM architecture achieved higher results in all training/testing combinations, excepts in Winter/Summer case. The best performing microphones are the T microphone and the one in the passenger back seat position. The driver plate microphone in the cross-domain combinations achieved the same issue highlighted above for the CNN case. Indeed, the $F1$-sore is zero due to the zero true positive occurrences, whereas the accuracy is slightly larger than

50%.

| | | BLSTM | |
|---|---|---|---|
| Train/Test | Mic | F1-score (%) | LSTM shape |
| W/W | DP | 95.71% | 156, 256, 156 |
| W/W | IP | 97.96% | 216, 316, 216 |
| W/W | ID | 70.8% | 54, 54, 54 |
| W/W | T | **99.80%** | 216, 316, 216 |
| W/S | DP | 0.0% (*) | 216, 216, 216 |
| W/S | IP | **93.37%** | 216, 316, 216 |
| W/S | ID | 74.4% | 54, 54, 54 |
| W/S | T | 93.30% | 216, 216, 216 |
| S/W | DP | 0.0% (*) | 54, 54, 54 |
| S/W | IP | 85.58% | 216, 316, 216 |
| S/W | ID | 57% | 54, 54, 54 |
| S/W | T | **99.70%** | 54, 54, 54 |
| S/S | DP | 89.88% | 54, 30, 54 |
| S/S | IP | 97.6% | 216, 216, 216 |
| S/S | ID | 96.34% | 54, 54, 54 |
| S/S | T | **99.75%** | 156, 256, 156 |

Table 6.5: Best performing BLSTM models from the tests. Training and testing have been conducted on summer (S) and winter (W) tires, with driver plate (DP), back seat passenger (IP), driver (ID) and trunk (T) microphones. (*) Please note that the F1-score is due to zero true positive occurrences. In those cases the Accuracy is 56.1% (W/S) and 62% (S/W), respectively.

From Table 6.4 and 6.5 seems that different combinations succeed in different training/testing conditions.

Taking the T microphone as the best performing microphone and averaging the results of the four training/testing combinations for each tested network, the best performance is achieved with the BLSTM network (see Table 6.6), with an improvement of the classification results of 2% with respect to the best CNN network.

| **CNN** | | | | | **BLSTM** | |
|---|---|---|---|---|---|---|
| CNN layer size | kernel shape | strides shape | Dense layer size | F1-score [%] | LSTM shape | F1-score [%] |
| 30, 25 | [[8, 1], [4, 10]] | [[4, 2], [1, 1]] | 800, 500 | **95.89** | 156, 256, 156 | **97.96** |
| 30, 20 | [[7, 10], [4, 5]] | [[4, 1], [4, 3] | 700, 900 | 95.33 | 4, 30, 54 | 97.40 |
| 30, 25 | [[7, 3], [6, 4]] | [[9, 2], [1, 7]] | 500 300 | 94.79 | 54, 54, 54 | 96.71 |
| 20, 20 | [[10, 6], [10, 2]] | [[3, 3], [4, 2]] | 1000, 900 | 93.93 | 216, 216, 216 | 96.52 |
| 20, 25 | [[7, 1], [5, 6]] | [[3, 2], [9, 9]] | 900, 100 | 93.90 | 216, 316, 216 | 95.47 |

Table 6.6: Best performing CNN and BLSTM combinations for the trunk microphone. The F1-score is averaged over the 4 summer-winter train-test combinations.

The computational cost is considered for real-time implementation. The computational workload of the CNN and BLSTM networks are extremely different on the machine used for the training and testing case. The CNN worked on an Nvidia GeForce GTX 970 GPU. It requires on average 3 seconds per epoch, while the same task requires 490 seconds per epoch for the BLSTM model. Similarly, the time to evaluate one ASF frame during testing is 11 ms for the CNN and 100 ms for the BLSTM. Since the ASF has a context of 1 s, both networks can achieve a Real-Time Factor (RTF) lower than 1.

### 6.1.6 Road Type Classification with a Real-Time Implementation

As described in Sections 6.1.4 and 6.1.5, the CNN architecture could be adopted for scalability in an embedded processor for practical implementation of the system. In this work, CNN was tested in a joint classification task to reduce the computational cost (memory required to store network weights and data and computational burden). Experiments were performed according to Figure 6.11.

The dataset is the same used for wetness detection. However, no cross-validation was performed to analyze the feasibility of the approach in a real scenario. The dataset is split into 64% for the train set, 16% for the validation and 20% for the test set. In total, 5675 s are used for training, 1418 s for validating and 1773 s for testing. The number of dry and wet samples were balanced for the training, leaving the rough/smooth samples unbalanced.

The first step was to compare two CNN architectures, the joint-CNN and the TL-CNN. The joint-CNN, described in Figure 6.12, is a CNN trained for joint classification of wetness and roughness, combining two binary outputs for joint classification. The TL-CNN is an architecture created following a Transfer Learning approach from two specialized CNNs trained separately: the best networks from individual evaluations (one for wetness classification, the other for roughness classification) are merged, adding one dense layer that is trained to optimize performance with the joint classification problem, while the CNNs are not re-trained (see Figure 6.13).

In the second step, shown in Figure 6.11.b, feature extraction is implemented in C++ on the embedded processor to assess the performance variation implied by a different implementation of the extraction process. The features are transferred from the processor to a computer and used to run a second batch of experiments using the best architecture of the first step and assess the performance variation.

Finally, the best configuration is used to compare the results when the test is performed on GPU and the embedded system (see Figure 6.11.c).

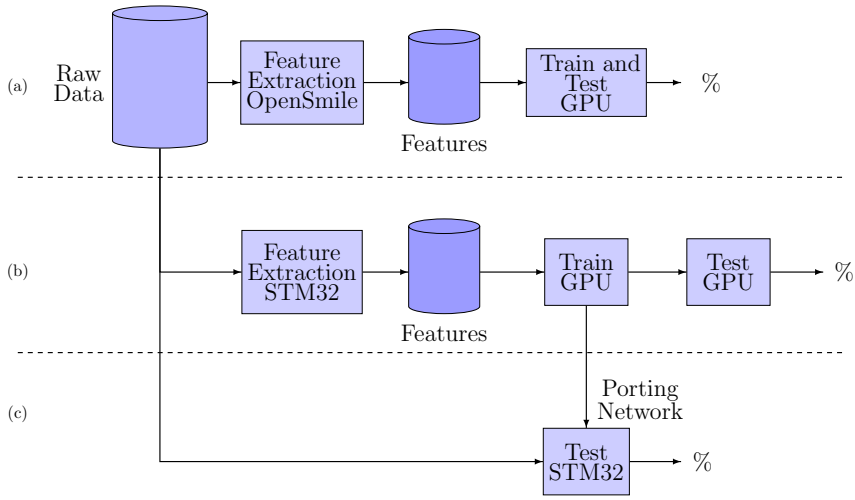



Figure 6.11: Experiments overview: (a) feature extraction using OpenSmile [177], train and test of networks using GPU; (b) feature extraction using STM32 board, train and test of networks using GPU; (c) importing of network trained by GPU on board and test of networks using STMicroelectronics (STM) board.

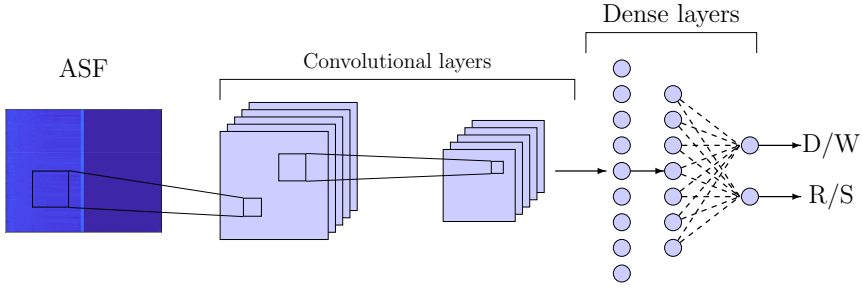



Figure 6.12: Joint-CNN for roughness and wetness classification.

The experiments using the GPU were performed using a random search. All configurations have a max pooling layer of dimension $2 \times 2$ and strides $1 \times 1$. In Table 6.7 the best results using the joint-CNN for wetness and roughness classification are presented, while in Table 6.8 and 6.9 the results are presented separately for the two tasks, whereas in Table 6.10 $F1$-score with merged networks are presented.

Best results are achieved with the joint-CNN; however, the TL-CNN shortly follows. Both approaches improve the results due to the wetness task, which achieves high performance but fails to provide remarkable performance on the road roughness task.

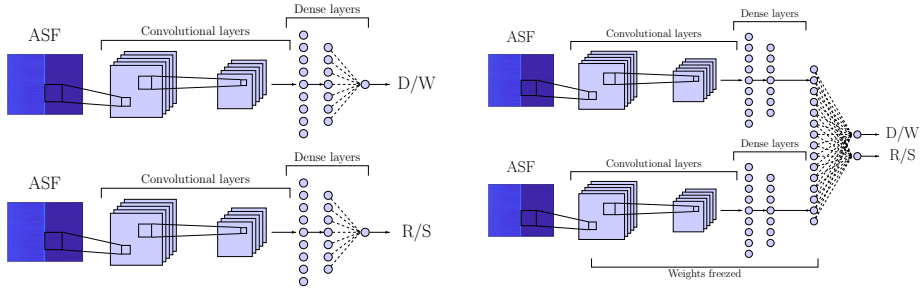



Figure 6.13: Left figure represents the networks that separately perform wetness and roughness detection, right figure represents the Transfer Learning approach adding one dense layer.

| Mic | CNN Layer Size | Kernel Shape | Strides Shape | Dense Layer Size | $F1_{macro}$ (%) |
|---|---|---|---|---|---|
| T | 20, 25 | [[1, 2], [1, 7]] | [[2, 2], [3, 6]] | 900, 300 | **94.10** |
| IB | 25, 20 | [[6, 7], [2, 8]] | [[2, 2], [6, 3]] | 900, 400 | **91.56** |

Table 6.7: Best performing joint-CNN for T and IB microphone for wetness and roughness classification. The F1-score is the average-macro.

The joint-CNN was used in the second step. The experiments were performed extracting features on STM32H743ZI board that has a 32-bit ARM® processor with a frequency up to 480 MHz, 2 MB of Flash Memory and 1 MB of RAM.

| CNN Layer Size | Kernel Shape | Strides Shape | Dense Layer Size | D/W F1-Score (%) | R/S F1-Score (%) | $F1_{macro}$ (%) |
|---|---|---|---|---|---|---|
| 25, 30 | [[5, 3], [5, 2]] | [[4, 5],[5, 2]] | 200, 800 | 98.31 | 82.70 | **88.10** |
| 20, 20 | [[8, 3], [6, 6]] | [[3, 1], [7, 10]] | 900, 900 | 97.73 | 81.28 | 87.67 |
| 30, 20 | [[9, 6], [2, 2]] | [[6, 2], [2, 9]] | 200, 1000 | 90.59 | 85.59 | 87.62 |
| 20, 25 | [[1, 2], [2, 3]] | [[4, 3], [2, 5]] | 200, 500 | 97.29 | 77.05 | 85.42 |
| 30, 25 | [[2, 8], [5, 3]] | [[4, 2], [2, 8]] | 700, 400 | 97.42 | 80.24 | 84.87 |
| 30, 30 | [[7, 5], [1, 2]] | [[6, 1], [4, 8]] | 600, 700 | 97.45 | 75.42 | 81.74 |
| 20, 30 | [[4, 4], [4, 5]] | [[4, 2], [6, ]] | 500, 200 | 95.62 | 77.77 | 81.68 |
| 25, 30 | [[8, 3], [5, 3]] | [[3, 2], [9, 5]] | 200, 200 | 90.66 | 64.82 | 80.49 |
| 30, 20 | [[3, 2], [8, 2]] | [[2, 1], [3, 10]] | 300, 1000 | 82.08 | 80.55 | 80.32 |
| 25, 20 | [[4, 7], [6, 1]] | [[4, 3], [5, 4]] | 500, 800 | 89.77 | 67.12 | 79.95 |

Table 6.8: Results obtained with the separated networks using T microphone.

To compare the experiments using the neural network on the STM32 board and GPU, audio data have been transferred using UART communication to the board for the feature extraction. The extracted features are transferred to the PC to training neural networks on GPUs. First experiments were performed testing networks on GPU using the features extracted on the board. Comparing the results in Table 6.11 and 6.7, $F1_{macro}$ is 3.89% and 6.35% lower for T and IB respectively. This is caused by the differences in the feature extraction algorithms. This problem can be alleviated by performing a random search with the features extracted by the board.

| CNN Layer Sz | Kernel Shape | Strides Shape | Dense Layer Sz | D/W | R/S | $F1_{macro}$ (%) |
|---|---|---|---|---|---|---|
| | | | | $F_1$ (%) | $F_1$ (%) | |
| 20, 20 | [[2, 6], [2, 6]] | [[3, 1], [10, 6]] | 500, 200 | 94.56 | 74.22 | **81.16** |
| 25, 20 | [[4, 7], [6, 1]] | [[4, 3], [5, 4]] | 500, 800 | 81.40 | 67.23 | 81.11 |
| 30, 20 | [[7, 3], [6, 4]] | [[9, 2], [1, 7]] | 500, 300 | 89.64 | 77.98 | 80.81 |
| 20, 25 | [[2, 8], [1, 8]] | [[5, 2], [8, 3]] | 800, 600 | 92.21 | 75.92 | 79.49 |
| 30, 20 | [[9, 6], [2, 2]] | [[6, 2], [2, 9]] | 200, 1000 | 89.03 | 72.11 | 78.96 |
| 30, 20 | [[5, 8], [4, 10]] | [[4, 2], [7, 3]] | 600, 700 | 95.37 | 61.08 | 78.42 |
| 20, 25 | [[8, 2], [1, 6]] | [[3, 3], [3, 2]] | 100, 100 | 95.17 | 61.08 | 78.39 |
| 20, 30 | [[5, 10], [7, 8]] | [[3, 2], [8, 3]] | 500, 1000 | 92.90 | 48.84 | 77.63 |
| 30, 20 | [[7, 7], [2, 2]] | [[5, 4], [5, 1]] | 500, 800 | 93.41 | 60.07 | 77.47 |
| 20, 30 | [[10, 10], [9, 6]] | [[3, 2], [7, 4]] | 600, 300 | 96.02 | 59.65 | 77.30 |

Table 6.9: Results obtained with the separated networks using IB microphone.

The trained network was also deployed on the STM32 board using the STM32CubeMX tool. A factor of ×4 compression was employed. The results achieved on GPUs and the board are presented in Table 6.11. Considering the performance degradation of the feature extraction on the board, the embedded processor and the GPU can achieve similar results, with a $F1_{macro}$ bearing as little as 0.27% degradation on the IB microphone.

| T microphone | | IB microphone | |
|---|---|---|---|
| Dense Layer Sz | $F_{1_{macro}}$ (%) | Dense Layer Sz | $F_{1_{macro}}$ (%) |
| 20 | 93.40 | 20 | 90.15 |
| 40 | **94.01** | 40 | 90.03 |
| 60 | 92.71 | 60 | 90.18 |
| 80 | 93.69 | 80 | 90.30 |
| 100 | 93.67 | 100 | **90.38** |
| 120 | 93.65 | 120 | 90.29 |
| 140 | 93.70 | 140 | 90.34 |
| 160 | 93.75 | 160 | 90.36 |
| 180 | 93.73 | 180 | 90.23 |
| 200 | 93.72 | 200 | 90.20 |

Table 6.10: Results obtained with the merged networks training the new layers. The best performance is obtained using CNN composed by 2 layers of 20 and 30 kernels respectively, dimensions of kernel are [[4, 4], [4, 5]], strides equal to [[4, 2], [6, 1]] and two dense layers of 500 and 200 units respectively for microphone T and 2 layers of 20 kernels each, with dimensions [[2, 6], [2, 6]] and strides [[3, 1], [10, 6]] and two dense layers of 500 and 200 units respectively for microphone IB.

The performance has been evaluated using Multiply-and-Accumulate Complexity (MACC), RTF, and RAM size regarding the computational complexity. The MACC index indicates the complexity of a model, including multiply-and-

accumulate instructions and an estimate of the activation functions computational cost. Feature extraction comes in 1 ms for Mel spectrograms in logarithmic scale and Energy processing for each frame and 1 ms for the first order derivative. The network processes input data in 178 ms (best network for T microphone) and 235 ms (best network for IB microphone). In both cases, the RTF is lower than 1, 27.7% and 33.4%, respectively.

| Mic | GPU | | STM32 | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $F_{1_{macro}}$ (%) | Memory Size (MB) | $F_{1_{macro}}$ (%) | RTF (%) | Compression Factor | Memory Size (MB) | RAM (kB) | Complexity (MACC) |
| T | 90.21 | 19.09 | 90.22 | 27.7 | x4 | 1.57 | 200.86 | 2099885 |
| IB | 85.21 | 16.21 | 84.94 | 33.4 | x4 | 1.35 | 251.08 | 3510810 |

Table 6.11: Results obtained with the same architecture used in Table 6.7 but trained with features extracted from ST board.

### 6.1.7 Final Remarks

In this work, wetness and roughness classification is presented, with a study on real-time implementation. Deep Learning approaches were studied, analyzing performance both in terms of classification results and computational cost. Different CNN architectures have been discussed for roughness classification, achieving the best results with the Siamese Neural Network. However, the weights stored in memory and the computational cost are almost twice as high as for a single-channel CNN. The same consideration were discussed for the road wetness classification, where the CNN is compared with the BLSTM. The recurrent network achieved better results, but the training and testing time was considerably longer than CNN.

CNN was the best choice to use the neural approach within an embedded processor. After analyzing two joint CNN architectures, the best model was used to evaluate the performance degradation implied by the deployment to a DSP system. The CNN with joint classification achieved better results than the two specialized CNN and transfer learning approaches. Regarding the deployment, the performance was evaluated, showing comparable results with GPUs.

The extraction of the features and the processing are computationally feasible, not exceeding 33.4% of the available time.

In conclusion, the road conditions classification by Deep Learning on an embedded processor is feasible with lightweight architectures such as CNN.

## 6.2 Joint VAD and SLOC with Acoustic Data Augmentation

In the research community, the task of detecting human speech and the speaker position, referred to as Voice Activity Detection (VAD) and Speaker LOCalization (SLOC), respectively, deserve much attention, finding applications in audio surveillance, human hearing modeling, speech enhancement, human and robot interaction and so forth [178, 179].

In literature, speaker detection and its localization are usually treated as two separate problems. Classical VADs are analyzed on specific signal characteristics [180] or rely on statistical models of the speech and noise signals [181]. SLOCs have been evaluated by classical techniques such as Cross Spectrum Phase (CSP) [182] and Steered-Response Power Phase Transform (SRP-PHAT) [183].

Recently, Deep Learning techniques are investigated for VAD and SLOC tasks. Regarding the VAD, numerous DNN architectures have been investigated, like Recurrent Neural networks (RNN) [184], DBN, MLP, BLSTM [185], and CNN [186].

A MLP is used in [186] for speaker localization in a binaural context. Another MLP is used in [187] using the Time Difference of Arrival (TDOA) as a feature and measured with eight microphones. A CNN is used in [188]. Multiple speaker localization was analyzed in [189], whereas, in [190], a CNN is exploited for predicting the speaker localization in Cartesian coordinates in a multi-room environment.

Few works present a solution for both problems at the same time. In [191], VAD and SLOC algorithms were used in a cooperative but distinct way, while in [192], DNNs were used jointly for VAD and SLOCs. An ensemble of several VAD and SLOC algorithms in a multi-room environment were studied in [193], with the integration of DNN and Gaussian Mixture Model (GMM), leading to a higher overall accuracy.

Generally, SLOC algorithms are evaluated within the condition of a perfectly detected speaker activity, called Oracle VAD. However, for a real-world scenario, the application is not appropriate because VADs commit errors, affecting the accuracy of the localization algorithms. For this reason, VAD and SLOC could be considered as unique problems.

This work is a progression of the work presented in [192], where the neural architecture is composed of a Neural SLOC cascaded to the Neural VAD. In this work, some architectures were analyzed: several neural VAD models were investigated, then a novel Neural SLOC is presented, maximizing the accuracy and the reliability of a VAD, in this way, the minimum amount of wrongly detected speech by the VAD is passed to the SLOC.

The evaluation of the proposed method is performed using a multi-room environment because it could replicate a real scenario. Crosstalk between multiple speakers in the same rooms and in different rooms could be present. Thus, a speaker detection and localization model must be robust against utterances pronounced in a room different from the one under observation. Background noise from other rooms could affect VAD and SLOC evaluations. Room reverberations annoy the signals in several manners. Finally, noise and speech are present inside and outside the understudy room.

The proposed method is compared with the ensemble techniques described in [193], where the authors used the same multi-room scenario used to analyze the Neural VAD and SLOC framework.

### 6.2.1 Proposed Method

The proposed method is presented in Figure 6.14. The speaker's detection and localization are performed using two different algorithms disposed of in a cascade configuration. The VAD algorithm predicts speech activity by elaborating audio features extracted from audio signals captured in the room under observation. The localization is performed by the SLOC algorithm over speech frames correctly detected by the VAD algorithm.

Feature extraction is performed to obtain LogMel and GCC-PHAT Pattern features which feed the proposed neural networks, depending on the model configuration. A post-processing technique is employed only for localization predictions. Four data-driven models for VAD are investigated, while for SLOC, two neural networks with Oracle VAD have been studied. Localization is performed in terms of speaker coordinates. The height of the speaker from the ground is not taken into account. Hence, considering the 2-D top view of a room, the speaker Cartesian coordinates will be referred to as $\chi$ and $\psi$, being normalized to the range [0, 1] by dividing for the wall length.

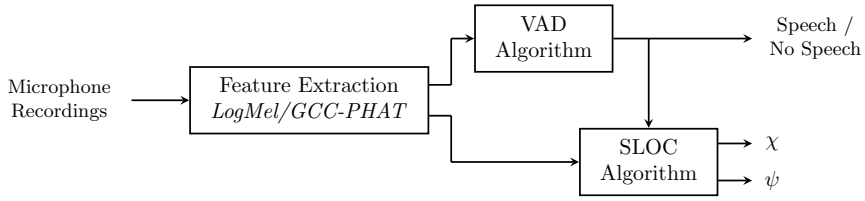



Figure 6.14: Conceptual scheme of the proposed method. Audio features are extracted from the recorded signals, which are used by VAD and SLOC algorithm depending on their specific configuration. After that, the SLOC algorithm performs localization over speech frames detected by the VAD algorithm.

### Features Extraction

For the proposed method, two different features are used: LogMel and GCC-PHAT. LogMel have been explained in Section 6.1.2 for the Auditory Spectral Feature extraction for the Road Roughness and Wetness classification.

GCC-PHAT are used to estimate the delay between two audio signals recorded by a microphone pair in the presence of the same sound event [182]. The motivation is due to the sound propagation, in which the sound wave reaches the two microphones in different time instants, allowing to estimate the Direction of Arrival (DOA) of the audio event. GCC-PHAT Patterns computation relies on the frequency domain cross-correlation between the two microphones audio signals, from which the Fourier inverse transform is then applied.

Since microphones pairs distance 50 cm, only the 51 values of the inverse transform are selected. Frame size and hop size of 60 ms and 50ms respectively are used in the feature extraction stage. Finally, features are normalized in the range [0, 1].

### Voice Activity Detection

Four neural models for the VAD task are discussed and compared.

The first one is the Joint-V VAD model, proposed in [192], also referred to as *Joint* VAD-SLOC, with the term Joint that stands for the employment of both detection and localization features, -$V$ stands for the use of its detection output.

In [192] was concluded that this architecture improves performance in terms of VAD accuracy. The model is presented in Figure 6.15, it is composed of a CNN fed by LogMel and GCC-PHAT, and it is trained using three outputs dedicated to both speech detection and speaker localization. Two branches of convolutional layers process the two feature sets, then a concatenation of the branch-dependent feature map is performed. The branches share the same number of hyperparameters. Finally, a set of hidden layers is applied. The model ends with the three outputs: the first one estimates the speech presence, the other two correspond to the speaker coordinates inside the room in a 2-D plane. Due to the $[-1, 1]$ range, *hard tanh* is employed as activation function of the localization outputs, while *sigmoid* activation function is used for the speech detection output.

A temporal context extends the amount of data processed by the network frame-by-frame, processing previous and future frames together with the actual frame, for a total of $C$ frame, where $C$ is the context. The *stride* is set equal to 1. A 2-D matrix is obtained for each microphone for the current frame: the rows are the features, and the columns are the frames with the context. Then, the different microphone features are stacked, leading to a 3-D tensor. The

model training is performed on speech and non-speech data. Speech detection is performed only by the speech detection output.

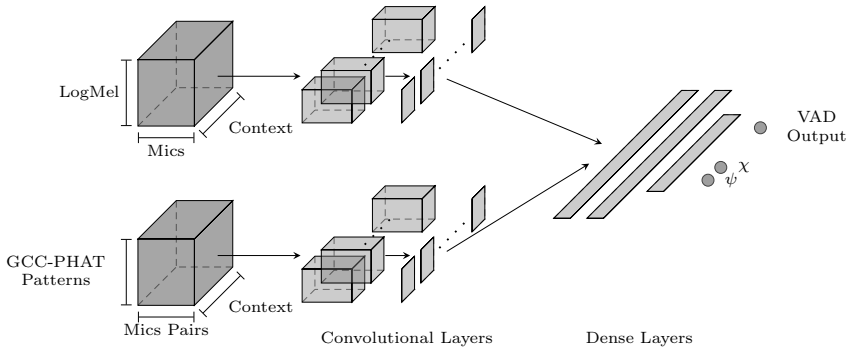



Figure 6.15: Architecture of the Joint-V VAD model.

The VAD output gives a value in the range [0, 1], whereas the localization task is treated as a regression problem. Hence the two localization outputs are mapped in the continuous [-1, 1] range: when speech is present, the speaker coordinate is given in the range [0, 1], while in speaker inactivity, the labels are set to -1.

The second architecture is called Joint-S VAD. This neural network shares the same neural architecture with the Joint-V VAD, using detection and localization features, and three outputs characterize it. However, the speaker activity is determined employing the localization outputs instead of the detection one., Speech detection is then performed through a particular threshold, an oblique line in the 2-D plane of the room. The purpose of this implementation is to compare the Joint-V VAD and to show that SLOC outputs can be accurately trained, even if their training is more sensible to employed data compared to its VAD output.

The third architecture is the Alt Joint VAD, which shares many aspects with the Joint-V VAD, but the output is composed of only the speaker detection, as shown in Figure 6.16.

Finally, the last studied architecture is the Neural VAD. The neural network process only LogMel and no SLOC outputs are present at the end of the network (see Figure 6.17). This model shows the importance of localization features for the detection task.

### Speaker Localization

Regarding the speaker localization task, two CNN architectures were analyzed. Both networks are trained on speech data by means of the oracle VAD, and their outputs are the room coordinates in the range $[0, 1]$. ReLU is used as an

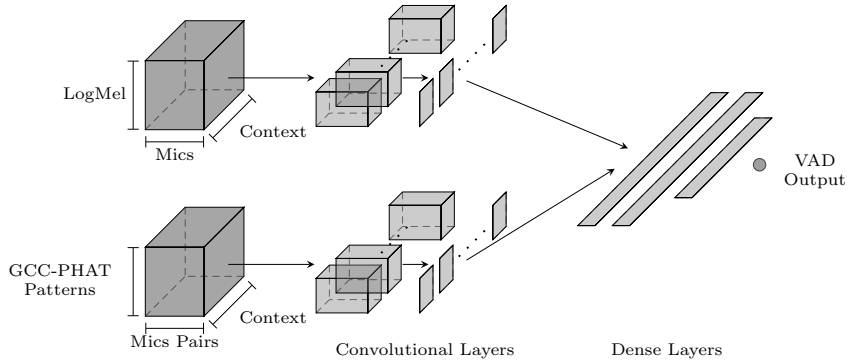



Figure 6.16: The Alt Joint VAD model. Its architecture shares many aspects with the Joint-V VAD shown in Figure 6.15, however the $\chi$ and $\psi$ outputs are absent.

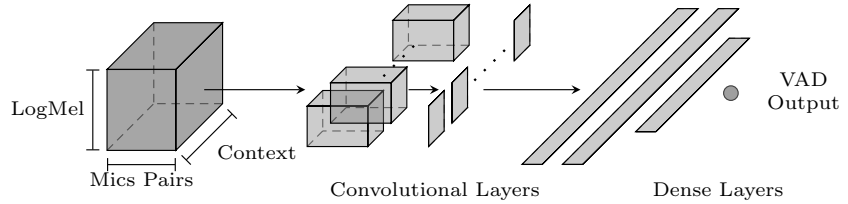



Figure 6.17: The Neural VAD model [192].

activation function.

The first model, defined as Single-Channel SLOC ($\text{SLOC}_{\text{SC}}$), where the GCC-Patterns features are organized in a 3-D tensor (see Figure 6.18). In Figure 6.19, the second neural architecture, called Multi-Channel SLOC ($\text{SLOC}_{\text{MC}}$) is presented: the input features organization and elaboration differ from the previous architecture. A standalone input is created for each pair of microphones, realizing a set of 2-D matrices, where rows and columns of each matrix are the temporal context and the features. The CNN is then characterized by several inputs equal to the considered microphone pairs.

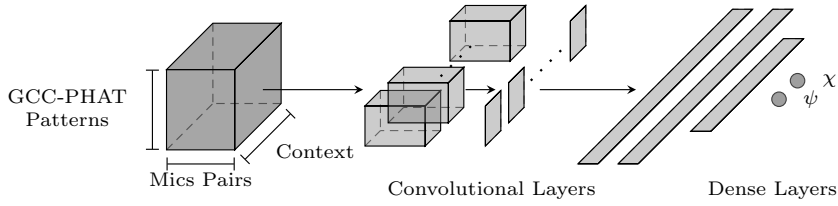



Figure 6.18: Single-Channel SLOC architecture.

Finally, the SLOC output is further processed by using a smoothing technique. A moving average filter of window size equal to 5 is applied to each predicted coordinate.

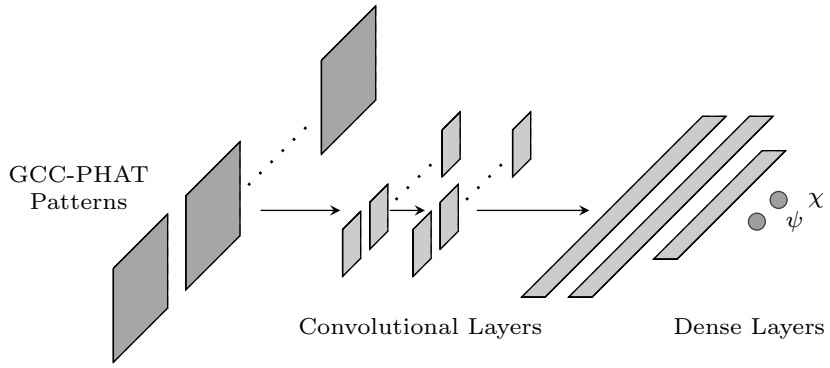



Figure 6.19: Multi-Channel SLOC architecture.

### 6.2.2 Baseline method

The baseline method is proposed in [193]: it consists of an ensemble of multiple VAD and SLOC algorithms, as shown in Figure 6.20. Two algorithms are considered for VAD, the Sohn's method and the Switching Kalman Filter (SKF). Four SLOC algorithms are taken into account, where three are derived from the Cross Spectrum Phase method, 2D-CSP, multi-channel CSP and Template CSP, and the last SLOC algorithm is the Steered Response Power (SRP-PHAT). Finally, three integration algorithms are analyzed for jointly processing VAD and SLOC predictions: Minimum Cost Criterion, SVM and a neural network-based classifier. The ensemble optimization is based on a three stages selection procedure.

The baseline model is more complex than the proposed method for many aspects. A manual tuning is required for each method of the VAD, SLOC and integration algorithms. The single-room prediction requires the analysis of all the other rooms. The proposed method does not require an extensive hand-tuning of each algorithm and processes each room independently from the others. Moreover, the proposed method avoids a third integration stage.

### 6.2.3 Experimental Setup

For the VAD and SLOC tasks, DIRHA project was used for the speaker detection and localization experiments [194]. The project regards speech detection, localization and recognition in a domestic environment. 40 omnidirectional microphones are installed in the walls and the ceilings of the apartment, as shown

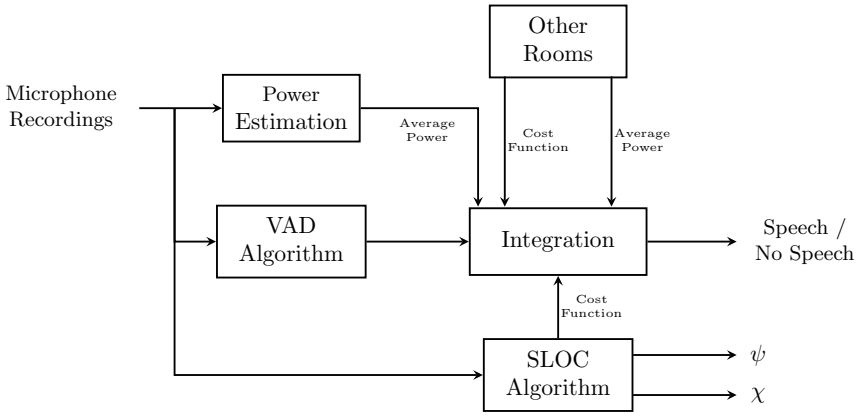



Figure 6.20: Conceptual scheme of the baseline method.

in Figure 6.21, in a 5 rooms apartment. Adjacent microphones are spaced by 50 cm, while walls installations are about 200 cm from the ground. The ceiling installations are present only in the kitchen and living room.

The DIRHA dataset is split into Real and Simulated subsets: the first one is composed of real recordings, with moving speakers, while the Simulated subset is achieved by convolving measured RIRs with speech data, overlapping speech events (this characteristic is not present in the Real subset).

Simulated subset is used for the experiments in this work since it comprises a great amount of speech data. The proposed methods are tested in the kitchen and livingroom since a ceiling installation is present, and most speech events are expected to occur in these two rooms in a real scenario. In total, 17 speaker positions are available for the kitchen and livingroom, with a number of microphones equal to 13 and 15, respectively.

Two distinct versions of the Simulated DIRHA dataset are used in this work. The EVALITA dataset, used in [192], and the HSCMA dataset. The first contains 70 scenes of Italian spoken utterances. The HSCMA dataset, used in [193] is composed of 80 samples of one minute length, equally divided in Italian, Greek, German and Portuguese languages. The Simulated HSCMA dataset is divided into the HSCMA-Dev and HSCMA-Test subsets, each composed of 40 scenes. The first is employed for training, and the second is for testing the performance. The training and validation sets were divided into 90% and 10% of the HSCMA-Dev subset, respectively. The HSCMA-test has been used to evaluate the proposed models with and without data augmentation better to evaluate the regularizing effect of the data augmentation strategy. When data augmentation is not considered, the models are trained using the HSCMA-Dev subset. When data augmentation is applied, the whole EVALITA data (excluding the duplicate files) and the DIRHA-DLS (DIRHA-LibriSpeech) subset

have been added to the HSCMA-Dev.

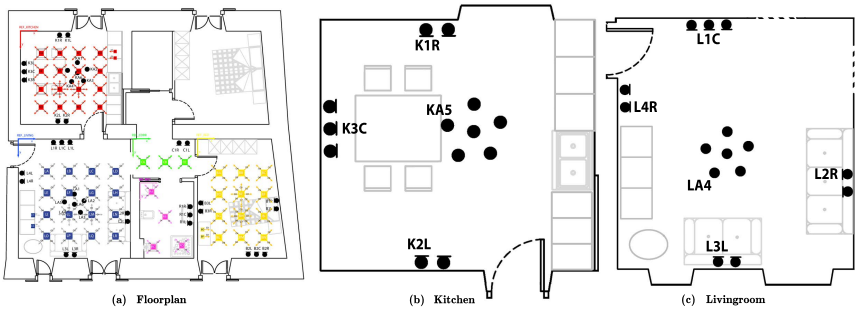



Figure 6.21: The map of the apartment used for the DIRHA project (a). Figures (b) and (c) show the considered rooms, where the thick black dots are the installed microphones.

The DIRHA-DLS dataset was created replicating the acoustic scenes of the kitchen and living room. The original RIRs were recorded within the DIRHA project, thus are not publicly available. For this reason, a new set of RIRs must be generated. A Room Impulse Response generator [195] is employed, which relies on the Image Source Model theory [196]. For each room, 17 positions, with 4 different orientations were used, as shown in Figure 6.22a and 6.22b, and 13 and 15 microphones, respectively, were used to record impulse response, with a total of 884 and 1020 RIRs.

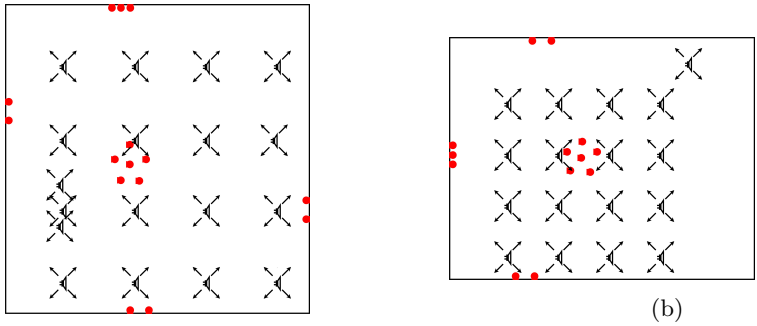
Speech data employed for DLS is randomly selected for the LibriSpeech dataset [163], with a total of 500 utterances. The desired Signal to Noise Ratio (SNR) is achieved by adding artificial noise created with the Maximum Length Sequence (MLS) technique [197]: first, the full-length utterance power $\sigma_S$ is estimated. After that, the noise power $\sigma_N$ is calculated as:

$$\sigma_N = \sigma_S / SNR_d \tag{6.4}$$

where $SNR_d$ is the desired signal to noise ratio, the RIRs are generated replicating the rooms dimensions and the reverberation time. They are generated at 48 kHz, and then the noise is added to get a $SNR_d$ equal to 40 dB, finally, the audio files are downsampled at 16kHz. In Figure 6.23, the block diagram is shown .

Three metrics were used to evaluate the VAD performance, the False Alarm rate (FA), the deletion rate (Del) and the overall Speech Activity Detection (SAD), defined as:

$$\text{Del} = \frac{N_{del}}{N_{sp}}, \quad \text{FA} = \frac{N_{fa}}{N_{nsp}}, \quad \text{SAD} = \frac{N_{fa} + \beta N_{del}}{N_{nsp} + \beta N_{sp}}, \tag{6.5}$$



(a) (b)

Figure 6.22: The living room (a) and kitchen (b) design through the data augmentation process.

where $N_{del}$, $N_{fa}$, $N_{sp}$ and $N_{nsp}$ are the total number of deletions (false negative), false alarms (false positive), speech and non-speech frames, respectively. The term $\beta = N_{nsp}/N_{sp}$ balances the different amount of data between speech and non-speech in the test set.

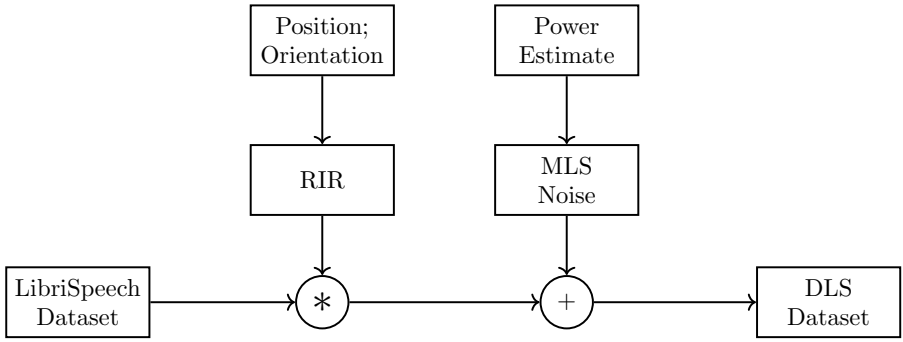



Figure 6.23: Block diagram of the algorithm used for the realization of the DLS dataset.

Root Mean Square Error (RMSE) and $P_{cor}$ measure the localization accuracy. RMSE is defined as:

$$\text{RMSE} = \frac{\sum_{i=0}^{N_{TOT}} \sqrt{(\chi_i - \chi_{\text{ref},i})^2 + (\psi_i - \psi_{\text{ref},i})^2}}{N_{TOT}}, \tag{6.6}$$

where $\chi_i$ and $\psi_i$ are the $i$-th network outputs, $\chi_{\text{ref,i}}$ and $\psi_{\text{ref},i}$ are the $i$-th reference speaker coordinates, and $N_{TOT}$ is the total number of frames. The latter is defined as $P_{cor} = N_{FINE}/N_{TOT}$, where $N_{FINE}$ is the number of frames localized with RMS inferior than 500 mm.

Two machines were used to exploit simulations: the first one is an HP note-

| | | Joint-V VAD<br>Joint-S VAD<br>Alt Joint VAD<br>Neural VAD | $SLOC_{SC}$<br>$SLOC_{MC}$ |
|---|---|---|---|
| Convolutional Layers | Number of Layers | 1, 2 | 1, 2 |
| | Number of Kernels | 64, 128 | 64, 128, 256 |
| | Kernel Size | 3, 4, 5 | 3, 4, 5 |
| | Kernel Strides | 1, 2, 3, 4, 5 | 1, 2, 3, 4, 5 |
| Hidden Layers | Number | 1, 2, 3, 4 | 1, 2, 3, 4, 5, 6, 7 |
| | Neurons | 256, 512, 1024, 2048 | 512, 1024, 2048 |

Table 6.12: Hyper-parameters of the DNN models, investigated through random search in the first optimization stage.

book model 15-p257nl equipped with a 4-core Intel i7 2.4 GHz, 16 GB of RAM and an Nvidia GeForce 840M graphic card; the second one is equipped with a 6-core Intel i7, 32 GB of RAM and a GeForce GTX970 graphic card. A total of 19 and 20 microphone pairs is selected for the kitchen and living room, respectively, for the GCC-PHAT Patterns, and 13 and 15 microphones for the LogMel feature extraction.

The DNN optimization strategy relies on two stages: first, the neural network is investigated through a random search technique, then the most performing model has trained again by using an augmented dataset. Adam optimizer is used, the number of epochs equals 500 and the batch size is 200. Neural networks weights are initialized with a gaussian distribution with $\mu = 0$ and $\sigma = 0.1$. Convolutional kernel were regularized with $L1$ and $L2$ regularizer, set both to $1 \cdot 10^{-4}$. Early Stopping is applied after 5 epochs. The context is set to 15 frames. Dropout equal to 0.5 to hidden layers. The investigated hyperparameters are reported in Table 6.12.

The best results of the VAD task are presented in Table 6.13: the best performing architecture is the Joint-V VAD, which achieves the lowest SAD of 8.3% over the HSCMA-Test subset. When data augmentation is used (symbol $^{\dagger}$ in Table 6.13), all four models improve performance, with the Joint-V VAD that achieved a SAD equal to 3.7%, better than Alt Joint VAD and Joint-S VAD, confirming the strategy to use two distinct VAD and SLOC algorithms.

Regarding the SLOC performance, neural networks were analyzed with the Oracle VAD and with the Joint-V VAD$^{\dagger}$. In Table 6.15 is presented the performance of SLOC algorithms with the Oracle VAD. The best results are achieved with the $SLOC_{MC}$, with a RMS equals 747 mm, while the $SLOC_{SC}$ achieved

| | | Kitchen | Living Room | Average |
|---|---|---|---|---|
| Joint-V VAD | SAD (%) | 7.6 | 9.0 | 8.3 |
| | Del (%) | 9.3 | 16.3 | 12.8 |
| | FA (%) | 5.9 | 1.7 | 3.8 |
| Joint-V VAD† | SAD (%) | 4.7 | 2.7 | **3.7** |
| | Del (%) | 7.4 | 3.5 | 5.4 |
| | FA (%) | 2.0 | 1.9 | 1.9 |
| Joint-S VAD | SAD (%) | 9.9 | 11.3 | 10.6 |
| | Del (%) | 16.9 | 21.5 | 19.2 |
| | FA (%) | 3.0 | 10.5 | 6.7 |
| Joint-S VAD† | SAD (%) | 7.2 | 8.6 | 7.9 |
| | Del (%) | 13.7 | 16.9 | 15.3 |
| | FA (%) | 0.7 | 0.3 | 0.5 |
| Alt Joint VAD | SAD (%) | 8.2 | 8.9 | 8.6 |
| | Del (%) | 13.9 | 15.9 | 15.0 |
| | FA (%) | 2.5 | 1.9 | 2.2 |
| Alt Joint VAD† | SAD (%) | 6.1 | 3.7 | 4.9 |
| | Del (%) | 11.4 | 6.7 | 9.0 |
| | FA (%) | 0.7 | 0.7 | 0.7 |
| Neural VAD | SAD (%) | 8.4 | 11.3 | 9.9 |
| | Del (%) | 8.8 | 16.5 | 12.6 |
| | FA (%) | 8.1 | 6.2 | 7.1 |
| Neural VAD† | SAD (%) | 4.6 | 3.9 | 4.3 |
| | Del (%) | 5.9 | 5.4 | 5.7 |
| | FA (%) | 3.2 | 2.7 | 2.9 |

Table 6.13: Achieved results for the three proposed data-driven algorithms on the HSCMA-*Test* set. For each model the first main line corresponds to the first optimization stage, where neural networks hyperparameters are investigated. The second line shows the result when data augmentation is applied, denoted with †.

almost the same result (751 mm). Using the augmented set, the performance is improved by achieving better localization accuracy: $\text{SLOC}_{\text{MC}}{}^{\dagger}$ achieved the best performance, with a RMS of 431 mm , while the $\text{SLOC}_{\text{SC}}{}^{\dagger}$ achieved 472 mm of RMS.

In the presence of the Joint-V VAD†, the $\text{SLOC}_{\text{MC}}{}^{\dagger}$ and the $\text{SLOC}_{\text{SC}}{}^{\dagger}$ achieve respectively 372 mm and 425 mm of RMS, as shown in Table 6.15. The $P_{cor}$ of $\text{SLOC}_{\text{MC}}{}^{\dagger}$ is 94.1%, concluding that this architecture is capable of better exploiting data recorded from multiple microphones, providing a better capability of generalizing compared to the $\text{SLOC}_{\text{SC}}$.

In Table 6.16 the best overall performance in terms of VAD and SLOC achieved in [193] are reported. The four baseline SLOCs are tested in the presence of an Oracle VAD. The two more accurate techniques are then separately

| Oracle VAD | | Kitchen | Living Room | Average |
|---|---|---|---|---|
| $SLOC_{SC}$ | RMS (mm) | 757 | 745 | 751 |
| | $P_{cor}$ (%) | 62.8 | 63.2 | 63.0 |
| $SLOC_{SC}^{\dagger}$ | RMS (mm) | 508 | 436 | 472 |
| | $P_{cor}$ (%) | 85.8 | 90.8 | 88.3 |
| $SLOC_{MC}$ | RMS (mm) | 788 | 707 | 747 |
| | $P_{cor}$ (%) | 57.5 | 66.7 | 62.1 |
| $SLOC_{MC}^{\dagger}$ | RMS (mm) | 447 | 415 | **431** |
| | $P_{cor}$ (%) | 90.4 | 94.0 | 92.2 |

Table 6.14: Results for the two proposed SLOC when tested in the presence of an Oracle VAD detecting speech over the HSCMA-*Test* subset. The $^{\dagger}$ denotes the application of data augmentation.

| Joint-V VAD$^{\dagger}$ | | Kitchen | Living Room | Average |
|---|---|---|---|---|
| $SLOC_{SC}$ | RMS (mm) | 724 | 600 | 662 |
| | $P_{cor}$ (%) | 66.5 | 69.3 | 67.9 |
| $SLOC_{SC}^{\dagger}$ | RMS (mm) | 451 | 399 | 425 |
| | $P_{cor}$ (%) | 87.8 | 91.3 | 90.0 |
| $SLOC_{MC}$ | RMS (mm) | 745 | 563 | 654 |
| | $P_{cor}$ (%) | 61.1 | 74.6 | 67.8 |
| $SLOC_{MC}^{\dagger}$ | RMS (mm) | 367 | 377 | **372** |
| | $P_{cor}$ (%) | 93.0 | 95.3 | 94.1 |

Table 6.15: Performance of the two VADs when tested over true positive frames detected by the Joint-V VAD$^{\dagger}$.

coupled with the Sohn's and SKF algorithms. Then, the less performance of the previously selected SLOCs is rejected. Finally, three proposed integration algorithms are applied to the remaining SLOC coupled with the two VADs. As a result, the best combination is Sohn's VAD and the Template method as SLOC when the SVM performs the integration. Sohn's method with the SVM integration is referred to as $VAD_{B}$ (where $_{B}$ stands for Baseline), while the Template method is referred to as $SLOC_{B}$. In Table 6.16 the average results are presented because the kitchen and livingroom results are not available separately in [193].

In Table 6.17, the most performing configuration in terms of SLOC accuracy for the baseline method is reported. The configuration shares the same SLOC method ($SLOC_{B}$), while the integration is the MLP with the SKF as VAD algorithm ($VAD_{B\_SKF}$).

Finally in Table 6.18 is reported the performance of $SLOC_{B}$ when the Oracle VAD is used.

The overall performances of the proposed approach with the baseline model

| | | Average |
|---|---|---|
| $VAD_B$ | SAD (%) | 6.7 |
| | DeL (%) | 6.1 |
| | FA (%) | 6.1 |
| $SLOC_B$ | RMS (mm) | 961 |
| | $P_{cor}$ (%) | 59.2 |

Table 6.16: The best overall performance in terms of VAD and SLOC for the baseline method.

| | | Average |
|---|---|---|
| $VAD_{B\_SKF}$ | SAD (%) | 17.4 |
| | DeL (%) | 25.8 |
| | FA (%) | 4.1 |
| $SLOC_B$ | RMS (mm) | 768 |
| | $P_{cor}$ (%) | 66.7 |

Table 6.17: Results of the baseline method when VAD and SLOC algorithms are selected in order to achieve the most accurate SLOC predictions.

| Oracle VAD | | Average |
|---|---|---|
| $SLOC_B$ | RMS (mm) | 1094 |
| | $P_{cor}$ (%) | 56.4 |

Table 6.18: Best performance of the baseline SLOC in the presence of an Oracle VAD.

are discussed. As the best configuration of the baseline method is taken, the $VAD_B$ and $SLOC_B$. In Table 6.19 a comparison between the two approaches for speaker localization is presented. $\Delta$ presents the subtraction of the result achieved by the baseline model from the result related to the most performing proposed algorithm. $SLOC_{MC}{}^{\dagger}$ and $SLOC_B$ are analyzed over speech detected using the Oracle VAD. The data-driven model is more robust against the multi-room environment, outperforming the classical localization algorithm of more than 35%.

| Oracle VAD | | Average |
|---|---|---|
| $\Delta$ | RMS (mm) | -663 |
| | $P_{cor}$ (%) | +35.8 |

Table 6.19: Difference of the most performing SLOC proposed by the authors ($SLOC_{MC}{}^{\dagger}$) with the $SLOC_B$ in the presence of an Oracle VAD.

Analyzing the results with the VAD algorithms, the overall performance $\Delta$

is better using data-driven models. In Table 6.20 a reduction of SAD of 3.0% is calculated, and a decreasing of 4.2% and 0.7% of FA and Del, respectively, is observed when the Joint-V VAD$^{\dagger}$ is employed. Using the SLOC$_{\mathrm{MC}}{}^{\dagger}$, a higher accuracy on $P_{cor}$ of 34.9% and a reduction of RMS of 589 mm with respect to SLOC$_{\mathrm{B}}$.

| | | Average |
|---|---|---|
| $\Delta$ | SAD (%) | -3.0 |
| | DeL (%) | -0.7 |
| | FA (%) | -4.2 |
| | RMS (mm) | -589 |
| | $P_{cor}$ (%) | +34.9 |

Table 6.20: Differences between the proposed data-driven approach and the baseline model of [193].

### 6.2.4 Final Remarks

A novel data-driven framework for detecting and localizing a speaker in a multi-room environment is studied. In literature, these two tasks have been studied as two separated problems; however, their mutual dependency must be addressed in a real-world scenario.

In this work, the architecture consists of a SLOC cascaded to VAD. The framework is compared with the only other framework present in literature for the detection and localization of a speaker in a multi-room environment.

Four CNN-based VAD algorithms are compared, where the most performing one is able to process audio features usually employed for VAD and SLOC, respectively. Two different SLOC architectures are then proposed to exploit data recorded by multiple microphone installations properly. Moreover, data augmentation is exploited, adding two subsets to the original DIRHA dataset: the first one is another version of the dataset; the second one is the result of the technique used for the RIR generation and convolution with speech data.

The Joint-V VAD model has been trained with data augmentation technique, a SAD reduction of 3.0% is observed compared to the baseline work. The same discussion for the SLOC architecture, achieving a $P_{cor}$ of 34.9% higher and an RMS of 589 mm lower than baseline techniques. The effectiveness of data augmentation is clearly observed for VAD and SLOC.

Future works will target the employment of new features for VAD and SLOC, principally aiming at a joint model performing detection and localization. Other neural network architectures could be studied, like recurrent neural networks, thus transferring learning techniques to adapt models developed for certain rooms to other rooms, even related to different residential

environments.

## 6.3 Sound Event Detection and Separation for the DCASE Challenge

Sound Event Detection (SED) is the task of recognizing the set of active sound events in a given audio recording. This technique could be used in a variety of applications: acoustic monitoring, human-computer interaction, meeting room transcription [198].

In literature, most of the baseline techniques for SED use supervised techniques with DNNs [199]: a neural classifier is trained using a strongly-labelled dataset of possibly co-occurring audio events. Since acquiring and creating strongly labelled data is a costly procedure, weakly labelled and unlabelled data are used to decrease the reliance on strongly annotated data via Semi-Supervised Learning.

CNN [199] and CRNN [200] are the widest architectures used for this kind of issue, in combination with the Multiple Instance Learning pooling method [201] that works with weak labels and a consistency loss for exploiting unlabelled data [202].

Source Separation (SS) is the task to extract from an acoustic mixture its underlying acoustic components [198]. DNNs based methods have significantly improved the separation of arbitrary sounds [203]. Source separation has a variety of applications such as hearing aid devices, Automatic Speech Recognition [204], diarization and video editing.

SED and SS have been used for the Detection and Classification of Acoustic Scenes and Events (DCASE) 2020 Task 4 challenge. In particular, several methods were employed for improving SED systems, while the SS is used to improve performance for the baseline SED technique proposed for the DCASE challenge.

### 6.3.1 The DCASE 2020 Task 4 Challenge Dataset

The DCASE 2020 Task 4 challenge allows to tackle Sound Event Detection (SED) in domestic environments facing real-world issues such as weakly annotated data, unlabelled data and only a very small corpus of strongly annotated, synthetic data. The goal of the challenge is to develop a SED system being able to tag onset and offset different sound event classes: Speech, Dog, Cat, Alarm Bell/Ringing, Dishes, Frying, Blender, Running water, Vacuum cleaner, Electric shaver/toothbrush.

The dataset is unbalanced and diverse. The DESED dataset [205] is composed of weakly labelled and unlabelled real soundscapes and isolated synthetic

events with strong labels. The SINS [206] and TUT Acoustic scenes 2017 [207] datasets offer background noise. The FUSS source separation dataset [208], aimed for the SS task, offers isolated events but not annotations.

### 6.3.2 Sound Event Detection

This Section discusses the proposed SED for the DCASE 2020 Task 4 Sound Event Detection and Source Separation. The CRNN based architecture is kept, as well as the mean teacher training scheme with the same network and optimization hyperparameters as the baseline.

The main contributions are in the training procedure, feature preprocessing and prediction post-processing and smoothing. The training dataset is composed of real and synthetic data. Only a portion of the real recording is provided with weak annotations. Test and Development sets include only data from real-world recordings.

Domain Adversarial Training (DAT) [209] enforces the model to learn features that are invariant to the change of domains in order to better generalize by learning from the synthetic data of domain examples. The domain adaptation process is embedded into the training procedure, adding a branch with a gradient reversal layer to the original architecture. This branch is only used at training time and then dropped at test time, so there is no computational overhead at run time.

Both the network and the domain classifier are jointly optimized during the training step. The gradient reversal layer serves the original architecture to work adversarially to the added domain classifier by extracting features that are domain-invariant, maximizing the loss of the domain classification task. In this way, the DAT enforces learning of features invariant between the synthetic examples domain and real-world recordings domain, reducing the chance of overfitting the strong-labelled synthetic examples.

Conv-TasNet separator network is used as the adversarial branch. The separator network outputs a probability in the whole input example by using mean pooling. Indeed, the network must classify whether the input example belongs to synthetic examples or weak/unlabelled examples. The fully convolutional architecture of the Conv-Tasnet helped the gradient propagation from the discriminator to the main network. Finally, the adversarial branch was placed in parallel to the RNN block after the CNN layers in the CRNN architecture, as shown in Figure 6.24.

The CRNN and the adversarial branches are then updated in two several steps adversarially: the loss for the CNN training $\mathscr{L}_{main}$ is comprised of only strong labelled loss, weak labelled loss and consistency loss between teacher

and student. Thus for the CRNN, the update rule for its parameters $\theta_C$ is:

$$\theta_c \leftarrow \theta_c - \alpha \left( \lambda \frac{\partial \mathscr{L}_{main}}{\partial \theta_c} - (1-\lambda) \frac{\partial \mathscr{L}_{adv}}{\partial \theta_c} \right), \tag{6.7}$$

where $\mathcal{L}_{adv}$ is the binary cross-entropy loss for the adversarial network, $\lambda$ is a hyper-parameter that controls the relative magnitude of the two losses, and $\alpha$ is the learning rate. Differently, for the adversarial network with parameters $\theta_a$, the update rule is:

$$\theta_a \leftarrow \theta_a - \alpha(1-\lambda) \frac{\partial \mathscr{L}_{adv}}{\partial \theta_a}. \tag{6.8}$$

In this work, the gradient reversal layer is used as a two step optimizing procedure like the one used in GANs because this approach gives better results than the gradient reversal layer approach, leading to more stable gradients during training.

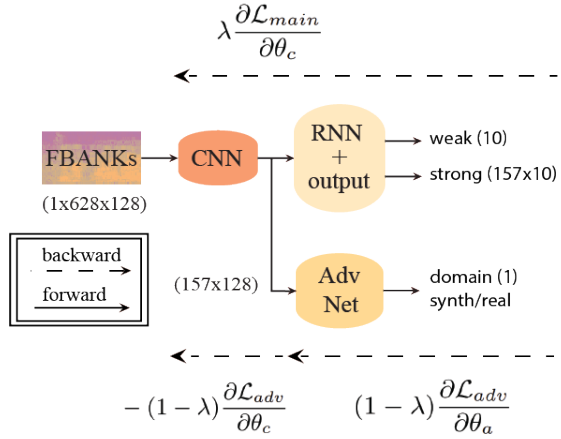



Figure 6.24: Domain adversarial training scheme.

Online augmentation strategy is employed because of the limited amount of acoustic diversity of DESED synthetic examples. Each synthetic training example is composed of randomly sampling from one to five random foregrounds and one background file from SINS. Reverberation is applied using FUSS RIRs, then a random time-domain augmentation chain is applied with different effects to each source, with a maximum of two random cascaded effects: additive noise bursts, additive sine bursts, time-varying comb filters, compression, pitch shifting, low-pass and high-pass filtering.

The foreground and background are mixed, with the foreground sampled between -35 dB and 0 dB, while the backgrounds constrained to be at max 5 dB over the foreground. Gaussian noise with SNR between -10 dB and 10 dB is

added, employing SpecAugment [210], ensuring a virtually amount of different strongly labelled data.

For weak and unlabelled data, an additional background from SINS is added to the original mixture with 50% probability and employ the feature domain augmentations.

As a trainable dynamic compression strategy, Per-Channel Energy Normalization (PCEN) [211] is used. This technique enhances transient audio events while transforming many soundscape noise patterns into additive white Gaussian noise, improving the robustness of audio classification algorithms in the presence of background noise with minimal computational overhead. PCEN is defined as:

$$\mathrm{PCEN}(t,f) = \left(\frac{E(t,f)}{(\epsilon + M(t,f))^{\alpha}} + \delta\right)^{r} - \delta^{r}, \tag{6.9}$$

where $t$ and $f$ denote time and Mel frequency band index, $\alpha$, $\epsilon$, $r$ and $\delta$ are positive constants and $E(t,f)$ denotes filter bank energy used as feature representation. $M(t,f)$ is a smoothed version of $E(t,f)$, which is computed using a first-order IIR filter as $M(t,f) = (1-s)\cdot M(t-1,f) + sE(t,f)$, with $s$ the smoothing coefficient.

The PCEN operation can have a negative impact on the stationary sounds, as vacuum cleaner or blender events. Therefore, several PCEN transformations in parallel (Parallel PCEN, PPCEN) are proposed in order to specialize each layer to a certain group of sounds. The output of each layer is given as feature channels to the CRNN model and jointly optimize the parameters of such PPCEN front-end layers using backpropagation, optimizing parameter $\alpha$, $\delta$, $r$, predetermining the two smoothing coefficients $s_1$=0.014 and $s_2$=0.25 and learning a combination of the smoother outputs.

In Figure 6.25 the output of the proposed 2-layers PPCEN front-end is shown when it is fed with speech and vacuum cleaner example. The first PCEN layer also captures more slow-varying events. The background noise and the vacuum cleaner harmonics can be distinguished and are enhanced with respect to the original log-Mel features. The second PCEN layer focuses only on events with faster onset, such as speech.

Finally, Hidden Markov Model (HMM) is used for the final prediction with two states for each class. The silence self-loop transition probability was tied to be the same for all HMM. It is tuned for every class and silenced on the development set using 50% split by using Random Forest and with the objective of maximizing the event based $F_1$ macro-average score of the trained SED model. Once the optimal parameters for the HMMs transition probabilities are found, the inference is performed by running Viterbi decoding on the CRNN, achieving the probabilities for each class. The HMMs emission probabilities were fixed in the pre-trained SED classifier, tuning the transition probabilities.

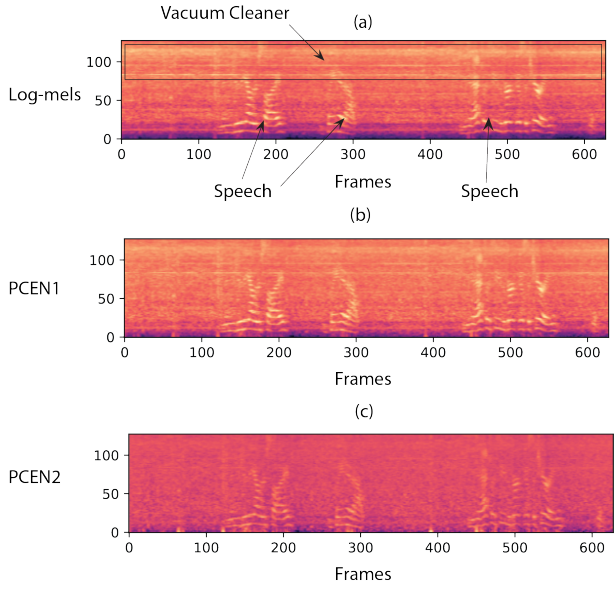



Figure 6.25: Output of the PPCEN layer: (a) original mixture LogMels, (b) first PCEN layer, (c) second PCEN layer. The two parallel layers capture different spectro-temporal dynamics.

#### Results

In Table 6.21 the results are reported on the development and evaluation sets. Four SED systems were compared: two single systems and two ensemble systems.

Regarding the single systems, the PPCEN with HMM (PPCEN+HMM) and the DAT with HMM (DAT+HMM) were employed, both improving performance over the baseline systems. DAT+HMM achieved the highest score for the development but not for the test. The PPCEN+HMM system generalizes slightly better the evaluation set.

The ensemble systems derive from a combination of PPCEN and DAT systems. Only the HMM transition probabilities differ. The second ensemble system (DAT+PPCEN+HMM 2) achieves a higher score on development but has work performance on the test, showing that HMM transition probabilities tuning can have a substantial impact on the final system performance and can be prone to overfitting.

### 6.3.3 Source Separation System

The combined separation and SED used as the proposed method are composed of the released pre-trained SED baseline system together with the proposed

| Method | Event macro $F_1$ score | | PSDS |
|---|---|---|---|
| | dev | eval | dev |
| Baseline | 34.8 | 34.9 | 0.61 |
| PPCEN+HMM | 43.69 | 42.6 | 0.63 |
| DAT+HMM | 45.20 | 42.0 | 0.68 |
| Ensemble DAT+PPCEN+HMM | 46.17 | 44.4 | 0.69 |
| Ensemble DAT+PPCEN+HMM 2 | 47.44 | 43.2 | 0.69 |

Table 6.21: Performance on development and evaluation sets.

separation system.

The baseline SED system derives from [212] and is trained on a synthetic dataset comprised of FUSS and synthetic examples from DESED. The baseline model is optimized with an End-to-End (E2E) waveform that denoises the background noise from the mixtures. The network architecture is based on TDCNN++ [203] and it performs the analysis, masking and synthesis: a DNN is used to estimate a mask in the STFT magnitude spectra domain in a transformed domain for each source.

The denoising process can introduce a mismatch because the baseline SED model is trained on noisy mixtures. For this reason, the proposed method is the Task-Aware separation training, which solves the domain mismatch problem that is present when the denoising is performed on a system trained on noisy mixtures. This method allows to train a separation system using a pre-trained SED back-end with the SED objective, thus avoiding the domain mismatch problem. A significant advantage over the joint training is that potentially a robust back-end, pre-trained and a significant amount of data, for which oracle targets for separation are not available, can be directly used.

A DNN mask-based separation is used on Mel-spectrograms. The separated features are then fed to the pre-trained SED system after applying logarithm and scaling. The predictions of the SED and its internal activations are used to train the mask-estimation DNN network. The back-end SED model is not updated, but the gradients are back-propagated through it in order to update the mask-estimation network.

Permutation Invariant Training (PIT) [213] and Mean Teacher are used to train the mask-estimation DNN. PIT is used to avoid overfitting of the weakly and synthetic examples very quickly.

The PIT loss function can be calculated as:

$$\mathscr{L}_{PIT} = \min_{\sigma \in \mathcal{F}_{\mathcal{J}}} \mathscr{L}(\hat{f}_{\sigma}, f) \tag{6.10}$$

where $f = [f_j(t)]_{j=1,\cdots,J}^{t=1\cdots,N}$ and $\hat{f} = [\hat{f}_j(t)]_{j=1,\cdots,J}^{t=1,\cdots,N}$ are the matrices of true and estimated targets, where $J$ and $N$ are respectively the maximum number of sources and the length of the estimated and true targets; $\hat{f}_{\sigma}$ is a permutation

of $f$ by $\sigma \in \mathcal{F}_{\mathcal{J}}$, with $\mathcal{F}_{\mathcal{J}}$ defined as the set of permutation of $[1, \cdots, J]$. The procedure consists in computing the loss $\mathscr{L}$ for all possible permutations of the targets and finding the permutation $\sigma$ for which the loss is minimized. Several different losses depending on what the labels are available for the current example are used to train the mask-estimation DNN.

For the strongly labelled examples, thus the DESED synthetic data, where the foreground features $f$ are available, the loss is the PIT Mean-Squared Error loss $\mathcal{L}_{MSE}$, finding the optimal permutation $\sigma_{opt}$ for the estimated separated features $\hat{f}$:

$$\mathscr{L}_{MSE} = \min_{\sigma \in \mathcal{F}_{\mathcal{J}}} MSE(\hat{f}_{\sigma}, f) \tag{6.11}$$

The estimated foregrounds features are then re-ordered according to $\sigma_{opt}$ and fed to the SED model for computing the Deep Feature Loss (DFL) $\mathscr{L}_{DFL}$, computed between each SED internal activations obtained with re-ordered estimated foregrounds $SED(\hat{f}_{\sigma_{opt}})$ and those obtained with oracle foregrounds $SED(f)$:

$$\mathscr{L}_{DFL} = \sum_{m=1}^{M} ||SED(\hat{f}_{\sigma_{opt}})^m - SED(f)^m|| \tag{6.12}$$

where the sum is calculated over all M layers of the SED back-end and $SED(f)^m$ denotes the activation of the m-th layer when the SED model is fed the feature matrix $f$. The total loss for strongly labelled examples are the sum of the two terms:

$$\mathscr{L}_{strong} = \mathscr{L}_{DFL} + \mathscr{L}_{MSE} \tag{6.13}$$

For weakly labelled examples, no oracle foregrounds are available, thus the separation model is trained in order to minimize the PIT binary cross entropy (BCE) between weak predictions of the SED model when it is fed the estimated foregrounds features $\hat{w}_{\sigma} = SED(\hat{f}_{\sigma})_{weak}$ and the weak labels:

$$\mathscr{L}_{weak} = \min_{\sigma \in \mathcal{W}_{\mathcal{J}}} BCE(\hat{w}_{\sigma}, w_{weak}) \tag{6.14}$$

The Mean-Teacher consistency is used for the unlabelled data. The Mean Teacher Semi-Supervised loss is used for the mask-estimation network and enforce SED weak and strong predictions consistency between the valued obtained with a student separation model $S(f; \theta_t)$ using permutation invariant MSE loss $\mathcal{L}_{teach}$ between the separated features of the two models:

$$\mathscr{L}_{teach} = \min_{\sigma \in \mathcal{F}_{\mathcal{J}}} MSE(SED(S(f_{\sigma})), SED(T(f))) \tag{6.15}$$

The total loss $\mathscr{L}_{tot}$ used to train the mask-estimation DNN is the sum of strong, weak and mean-teacher losses.

#### Results

In Table 6.22 the performance of the proposed separation system trained with the Task-Aware separation objective (Proposed) is presented, comparing the results with the SED challenge baseline back-end system without pre-processing (SED-only Baseline), the combined separation and the sound event detection baseline with (SEP+SED Baseline) and without averaging of predictions between noisy and denoised mixtures (SEP+SED Baseline no avg.).

The combined separation and SED Baseline system fail to improve the SED back-end performance when no ensembling is performed, whereas ensembling produces a moderate performance improvement. The proposed method offers more than 2% improvement over the plain SED-only baseline with no ensembling and a significantly smaller separation model.

| Method | Event macro $_1$-score | Parameters |
|---|---|---|
| SED-only Baseline | 34.8 | 1 M |
| SEP+SED Baseline | 35.6 | 10 M |
| SEP+SED Baseline no avg | 33.4 | 10 M |
| Proposed | 37.0 | 4M |

Table 6.22: Performance of combined separation and SED systems on DCASE 2020 Task 4 development set.

### 6.3.4 Final Remarks

Sound Event Detection and Source Separation system have been investigated in this work, tackling DCASE Challenge Task 4.

Regarding the SED, the baseline SED system is investigated, adding a PCEN front-end feature pre-processing, Domain Adversarial Training and online data augmentation and mixing, achieving an improvement of performance with a minimal computational overhead at inference time. The HMM smoothing is also investigated, improving the results of the system by refining network predictions.

A novel training scheme combined Source Separation and Sound Event Detection is presented for the Source Separation purpose. The Source separation system is trained in an End-to-End fashion with a pre-trained SED system that is not updated. A combination of permutation-invariant training objectives is used, booth signal-based and SED-based, called Task-Aware separation, as the separation system is optimized directly with the back-end task objective.

Comparing the proposed approach with the combined Source Separation and Sound Event Detection DCASE 2020 Task 4 baseline methods, the proposed one achieves better performance with fewer parameters.

In future works, the End-to-End approach will be further investigated also for Automatic Speech Recognition. Further studies will also be investigated for the SED task.

# Chapter 7

# Conclusions and Future Works

## 7.1 Conclusions

In this dissertation, Machine Learning and Deep Optimization techniques for digital filters design for Multipoint Audio Equalization and Personal Sound Zones have been analyzed.

Multipoint Audio Equalization aims to improve sound quality in a listening environment composed of different sound sources and listening points. The Personal Sound Zones methods aim to separate the sound sources within two areas that are located within a listening environment.

Many experiments have been performed using various automotive scenarios. The car cabin is composed of an irregular and small volume, several materials on all surfaces, and large obstacles inside the car cabin, adding non-linearities to the acoustic scene. Machine Learning methods and Deep Neural Networks have been widely investigated to solve the two optimization problems, achieving better results than the state-of-the-art methods.

In Chapter 1, there is a brief introduction on the automotive listening environment, Multipoint Audio Equalization and Personal Sound Zones. From there, the problem statement and motivations have been described.

In Chapter 2, the Machine Learning techniques and multi-objective optimization problems have been explained. Then, the evolutionary algorithms and neural networks used for experiments have been illustrated.

In Chapter 3, FIR, IIR, Parametric IIR filters and a review of the up-to-date digital filter design techniques have been described, including evolutionary algorithms and deep neural networks.

Multipoint Audio Equalization and Personal Sound Zones have been introduced in the two successive chapters.

In Chapter 4, the Multipoint Audio Equalization techniques presented in literature have been discussed. Several DNNs have been compared, describing the experiments and results obtained between the baseline techniques and the proposed method. The first studies on DNNs have been addressed in the design of FIR filters for Multipoint Audio Equalization, analyzing over-determined

systems and different automotive scenarios. Neural techniques performed better than baseline and evolutionary algorithms. Moreover, among the neural architectures used for optimization, the CNN obtained the best results. Despite the good performance in the frequency domain, the achieved filters are not compact, leading to many artefacts during the sound reproduction. These problems have been addressed in the Personal Sound Zones task.

With the analysis on the design of Parametric IIR filters for Multipoint Audio Equalization, there was a significant advancement in the study of neural networks for parameters optimization. A novel architecture, the BiasNet, has been compared with a baseline technique for parametric IIR filters design and another one used for the FIR filters design, resulting in better performance. Comparing the results with the CNN, the BiasNet achieved better results with a lower computational cost.

The Personal Sound Zones task has been presented in Chapter 5. The most important techniques shown in literature have been discussed. Next, the experiments in a car scenario have been introduced, using FIR and Parametric IIR filters. Regarding the design of the FIR filters for PSZ, the problems encountered in the Multipoint Audio Equalization have been solved, adding regularization terms to obtain compact filters. The results are promising with the proposed approach, achieving comparable results with respect to the baseline methods.

FIR filters have been compared with Parametric IIR filters, increasing the number of SOS's per one-third octave band in the latter. Using 4 SOS's per one-third octave band achieved the best performance, obtaining comparable results with respect to FIR filters, meaning that Parametric IIR filters could be preferable because of the computational cost reduction with respect to the FIR filters.

## 7.2 Future Works

Future works will be directed on studying new Deep Optimization techniques for the design of digital filters for Multipoint Audio Equalization and PSZ. A study of new neural architectures and a further analysis of the BiasNet will be conducted to obtain optimized parameters with fewer iterations and resources.

The tests carried out have been used for a pre-tuning stage in a static scenario. The filters obtained can be inserted into DSP systems. In the future, the goal will be to exploit Deep Optimization techniques for adaptive filtering to optimize parameters in real-time. Nowadays, it is challenging to achieve this solution because DSP systems have limited computational resources.

The two tasks have been considered as multi-objective optimization problems. Thus, future works must be conducted to study new cost functions that

allow optimized parameters with fewer loss functions.

Further studies will be conducted on evaluation metrics. Other frequency domain and perceptual metrics will be investigated. In addition, subjective tests within the acoustic scene will be analyzed.

# List of Publications

- Giovanni Pepe, Leonardo Gabrielli, Livio Ambrosini, Stefano Squartini and Luca Cattani, "Detecting road surface wetness using microphones and convolutional neural networks", in *Audio Engineering Society Convention 146*, Dublin, Ireland, 2019.
- Paolo Vecchiotti, Giovanni Pepe, Emanuele Principi, Stefano Squartini, "Detection of activity and position of speakers by using deep neural networks and acoustic data augmentation", in *Expert Systems with Applications*, vol. 134, 2019, pages 53-65, ISSN 0957-4174.
- Giovanni Pepe, Leonardo Gabrielli, Squartini Stefano, Luca Cattani, "Designing Audio Equalization Filters by Deep Neural Networks", in *Applied Sciences*, vol. 10, num. 7, 2020.
- Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, Luca Cattani and Carlo Tripodi, "Generative Adversarial Networks for Audio Equalization: an evaluation study", in *Audio Engineering Society Convention 148*, Wien, Austria, 2020.
- Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, Luca Cattani, "Evolutionary Tuning of Filters Coefficients for Binaural Audio Equalization", in *Applied Acoustics*, vol. 163, 2020, ISSN 0003-682X.
- Samuele Cornell, Giovanni Pepe, Emanuele Principi, Manuel Pariente, Michel Olvera, Leonardo Gabrielli, Stefano Squartini, "The UNIVPM-INRIA Systems for the DCASE 2020 Task 4", in *DCASE 2020 Challenge*, 2020.
- Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, Luca Cattani and Carlo Tripodi, "Deep Learning for Individual Listening Zone", in *2020 IEEE 22nd International Workshop on Multimedia Signal Processing*, Tampere, Finland, 2020.
- Samuele Cornell, Michel Olivera, Manuel Parente, Giovanni Pepe, Emanuele Principi, Leonardo Gabrielli, Stefano Squartini, "Domain-Adversarial Training and Trainable Parallel Front-end for the DCASE 202 Task 4 Sound Event Detection Challenge", in *DCASE 2020 5th Workshop on Detection and Classification of Acoustic Scenes and Events*, 2020.

- Samuele Cornell, Michel Olvera, Manuel Pariente, Giovanni Pepe, Emanuele Principi, Leonardo Gabrielli, Stefano Squartini, "Task-Aware Separation for the DCASE 2020 Task 4 Sound Event Detection and Separation Challenge", in *DCASE 2020 5th Workshop on Detection and Classification of Acoustic Scenes and Events*, 2020.

- Giovanni Pepe, Leonardo Gabrielli, Emanuele Principi, Stefano Squartini and Luca Cattani, "Road Type Classification Using Acoustic Signals: Deep Learning Models and Real-Time Implementation", in *Progresses in Artificial Intelligence and Neural Systems*, Springer, 2021.

- Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, Luca Cattani and Carlo Tripodi, "Gravitational Search Algorithm for IIR Filter-Based Audio Equalization", in *2020 28th European Signal Processing Conference (EUSIPCO)*, Amsterdam, Netherland, pages 496-500, 2021.

- Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, Carlo Tripodi, Nicolò Strozzi and Alessandro Costalunga, "Metodo e sistema per l'equalizzazione audio in uno o più punti di ricezione all'interno di un ambiente di ascolto", Patent filled to the Italian patent office on 10/05/2021 with number 102021000011891 (Pending).

- Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, Carlo Tripodi, Nicolò Strozzi, "Deep Optimization of Parametric IIR Filters for Audio Equalization", IEEE/ACM Transactions on Audio, Speech, and Language Processing (in Review)

- G. Pepe, L. Gabrielli, S. Squartini, C. Tripodi and N. Strozzi, "Deep Optimization of FIR Filters for Individual Listening Zone", 2022 IEEE International Conference on Acoustics, Speech and Signal Processing, 2022, Singapore (in Review)

# Bibliography


[1] Andrea Azzali, Alberto Bellini, Angelo Farina, and Emanuele Ugolotti, "Design and implementation of psychoacoustics equalizer for infotainment," *DSP Implementation Day, Politecnico di Milano*, vol. 23, 2002.

[2] Omar Al-Jarrah and Adnan Shaout, "Automotive volume control using fuzzy logic," *Journal of Intelligent & Fuzzy Systems*, vol. 18, no. 4, pp. 329–343, 2007.

[3] Stefania Cecchi, L. Palestini, P. Peretti, F. Piazza, and A. Carini, "Multipoint equalization of digital car audio systems," in *2009 Proceedings of 6th International Symposium on Image and Signal Processing and Analysis*, September 2009, pp. 650 – 655.

[4] S. Cecchi, L. Palestini, E. Moretti, and F. Piazza, "A new approach to digital audio equalization," in *2007 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2007, pp. 62–65.

[5] Philip A. Nelson, Felipe Orduña-Bustamante, David Engler, and Hareo Hamada, "Experiments on a system for the synthesis of virtual acoustic sources," *Journal of the Audio Engineering Society*, vol. 44, no. 11, pp. 990–1007, November 1996.

[6] Vesa Välimäki and Joshua D. Reiss, "All about audio equalization: Solutions and frontiers," *Applied Sciences*, vol. 6, no. 5, 2016.

[7] Alberto Bellini, Gianfranco Cibelli, and Angelo Farina, "AQT - a new objective measurement of the acoustical quality of sound reproduction in small compartments," *Journal of the Audio Engineering Society*, May 2001.

[8] Xian-Da Zhang, *Machine Learning*, pp. 223–440, Springer Singapore, Singapore, 2020.

[9] Zoubin Ghahramani, *Unsupervised Learning*, pp. 72–112, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[10] Pádraig Cunningham, Matthieu Cord, and Sarah Jane Delany, *Supervised Learning*, pp. 21–49, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[11] Richard S. Sutton and Andrew G Barto, *Reinforcement Learning: An introduction*, MIT press, 2018.

[12] L. Torrey and J. Shavlik, *Transfer Learning*, pp. 242–264, IGI Global, 2010.

[13] Stephen Boyd and Lieven Vandenberghe, *Convex optimization*, Cambridge university press, 2004.

[14] Prateek Jain and Purushottam Kar, "Non-convex optimization for machine learning," *Foundations and Trends®in Machine Learning*, vol. 10, no. 3-4, pp. 142–363, 2017.

[15] Kalyanmoy Deb, *Multi-objective Optimisation Using Evolutionary Algorithms: An Introduction*, pp. 3–34, Springer London, London, 2011.

[16] Kalyanmoy Deb and Himanshu Gupta, "Searching for robust pareto-optimal solutions in multi-objective optimization," in *Evolutionary Multi-Criterion Optimization*, Carlos A. Coello Coello, Arturo Hernández Aguirre, and Eckart Zitzler, Eds., Berlin, Heidelberg, 2005, pp. 150–164, Springer Berlin Heidelberg.

[17] Marcela Zuluaga, Guillaume Sergent, Andreas Krause, and Markus Püschel, "Active learning for multi-objective optimization," in *Proceedings of the 30th International Conference on Machine Learning*, Sanjoy Dasgupta and David McAllester, Eds., Atlanta, Georgia, USA, 17–19 June 2013, vol. 28 of *Proceedings of Machine Learning Research*, pp. 462–470, PMLR.

[18] Jia Shi, Jinchun Song, Bin Song, and Wen F. Lu, "Multi-objective optimization design through machine learning for drop-on-demand bioprinting," *Engineering*, vol. 5, no. 3, pp. 586–593, 2019.

[19] G. P. Liu and V. Kadirkamanathan, "Learning with multi-objective criteria," in *1995 Fourth International Conference on Artificial Neural Networks*, 1995, pp. 53–58.

[20] Yaochu Jin and Bernhard Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 3, pp. 397–415, 2008.

[21] Junfei Zhang, Yimiao Huang, Yuhang Wang, and Guowei Ma, "Multi-objective optimization of concrete mixture proportions using machine learning and metaheuristic algorithms," *Construction and Building Materials*, vol. 253, pp. 119208, 2020.

[22] Yinan Shao, Jerry Chun-Wei Lin, Gautam Srivastava, Dongdong Guo, Hongchun Zhang, Hu Yi, and Alireza Jolfaei, "Multi-objective neural evolutionary algorithm for combinatorial optimization problems," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–11, 2021.

[23] Thomas Weise, *Global Optimization Algorithm: Theory and Application*, Institue of Applied Optimization, Hefei University, Hefei, China, 2009.

[24] Coello Coello Carlos Artemio, "A comprehensive survey of evolutionary-based multiobjective optimization techniques," *Knwoledge and Information Systems*, 1999.

[25] Ning Li, Zhanguo Su, Houssem Jerbi, Rabeh Abbassi, Mohsen Latifi, and Noritoshi Furukawa, "Energy management and optimized operation of renewable sources and electric vehicles based on microgrid using hybrid gravitational search and pattern search algorithm," *Sustainable Cities and Society*, vol. 75, pp. 103279, 2021.

[26] Eric Taillard, "Tabu search," in *Metaheuristics*, pp. 51–76. Springer, 2016.

[27] David E. Goldberg and John Henry Holland, "Genetic algorithms and machine learning," 1988.

[28] Esmat Rashedi, Hossein Nezamabadi-pour, and Saeid Saryazdi, "GSA: A gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009, Special Section on High Order Fuzzy Sets.

[29] Feng Wang, Heng Zhang, and Aimin Zhou, "A particle swarm optimization algorithm for mixed-variable optimization problems," *Swarm and Evolutionary Computation*, vol. 60, pp. 100808, 2021.

[30] Zhi-Hui Zhan, Jun Zhang, Yun Li, and Henry Shu-Hung Chung, "Adaptive particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 6, pp. 1362–1381, 2009.

[31] Ruhul Sarker and Hussein A. Abbass, "Differential evolution for solving multiobjective optimization problems," *Asia-Pacific Journal of Operational Research*, vol. 21, no. 02, pp. 225–240, 2004.

[32] Steven A. Hofmeyr and Stephanie Forrest, "Architecture for an artificial immune system," *Evolutionary Computation*, vol. 8, no. 4, pp. 443–473, 2000.

[33] Carlos A. Coello Coello and Nareli Cruz Cortés, "Solving multiobjective optimization problems using an artificial immune system," *Genetic programming and evolvable machines*, vol. 6, no. 2, pp. 163–190, 2005.

[34] Jianyong Chen, Qiuzhen Lin, and Zhen Ji, "A hybrid immune multiobjective optimization algorithm," *European Journal of Operational Research*, vol. 204, no. 2, pp. 294–302, 2010.

[35] Yirui Wang, Shangce Gao, Yang Yu, Zonghui Cai, and Ziqian Wang, "A gravitational search algorithm with hierarchy and distributed framework," *Knowledge-Based Systems*, vol. 218, pp. 106877, 2021.

[36] Hamid Reza Hassanzadeh and Modjtaba Rouhani, "A multi-objective gravitational search algorithm," in *2010 2nd International Conference on Computational Intelligence, Communication Systems and Networks*, 2010, pp. 7–12.

[37] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, 1995, vol. 4, pp. 1942–1948 vol.4.

[38] Suman Kumar Saha, Rajib Kar, Durbadal Mandal, and S.P. Ghoshal, "Gravitation search algorithm: Application to the optimal iir filter design," *Journal of King Saud University - Engineering Sciences*, vol. 26, no. 1, pp. 69–81, 2014.

[39] David Lopez-Paz and Levent Sagun, "Easing non-convex optimization with neural networks," 2018.

[40] A. Cochocki and Rolf Unbehauen, *Neural Networks for Optimization and Signal Processing*, John Wiley & Sons, Inc., USA, 1st edition, 1993.

[41] Gabriel Villarrubia, Juan F. De Paz, Pablo Chamoso, and Fernando De la Prieta, "Artificial neural networks used in optimization problems," *Neurocomputing*, vol. 272, pp. 10–16, 2018.

[42] Hsu-Shih Shih, Ue-Pyng Wen, S. Lee, Kuen-Ming Lan, and Han-Chyi Hsiao, "A neural network approach to multiobjective and multilevel programming problems," *Computers & Mathematics with Applications*, vol. 48, no. 1, pp. 95–108, 2004.

[43] Misha Denil, Babak Shakibi, Laurent Dinh, Marc'Aurelio Ranzato, and Nando de Freitas, "Predicting parameters in deep learning," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, Red Hook, NY, USA, 2013, NIPS'13, p. 2148–2156, Curran Associates Inc.

[44] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016, `http://www.deeplearningbook.org`.

[45] Rumelhart David, Hinton Geoffrey, and Williams Ronald, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[46] Yann LeCun, Koray Kavukcuoglu, and Clement Farabet, "Convolutional networks and applications in vision," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, 2010, pp. 253–256.

[47] Santiago Pascual, Antonio Bonafonte, and Joan Serrà, "Segan: Speech enhancement generative adversarial network," in *Proc. Interspeech 2017*, 2017, pp. 3642–3646.

[48] Guillaume Alain and Yoshua Bengio, "What regularized auto-encoders learn from the data-generating distribution," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3563–3593, 2014.

[49] V. Verfaille, M. Holters, and U. Zölzer, *DAFX: Digital Audio Effects*, chapter 1, pp. 1–46, John Wiley & Sons, Ltd, 2011.

[50] Sophocles J. Orfanidis, *Introduction to signal processing*, Prentice-Hall, Inc., 1995.

[51] N. Agrawal, A. Kumar, Varun Bajaj, and G. K. Singh, "Design of digital IIR filter: A research survey," *Applied Acoustics*, vol. 172, pp. 107669, 2021.

[52] Oscar Montiel, Oscar Castillo, Patricia Melin, and Roberto Sepulveda, "The evolutionary learning rule for system identification," *Applied Soft Computing*, vol. 3, no. 4, pp. 343–352, 2003, Soft Computing for Control of Non-Linear Dynamical Systems.

[53] A. Deczky, "Synthesis of recursive digital filters using the minimum p-error criterion," *IEEE Transactions on Audio and Electroacoustics*, vol. 20, no. 4, pp. 257–263, 1972.

[54] A. Chottera and G. Jullien, "A linear programming approach to recursive digital filter design with linear phase," *IEEE Transactions on Circuits and Systems*, vol. 29, no. 3, pp. 139–149, 1982.

[55] Y.-C. Lim, J.-H. Lee, C. K. Chen, and R.-H. Yang, "A weighted least squares algorithm for quasi-equiripple FIR and IIR digital filter design," *IEEE Transactions on Signal Processing*, vol. 40, no. 3, pp. 551–558, 1992.

[56] F. Argenti and E. Del Re, "Design of iir eigenfilters in the frequency domain," in *Proceedings of 13th International Conference on Digital Signal Processing*, 1997, vol. 2, pp. 629–632 vol.2.

[57] P. Dutilleux, M. Holters, S. Disch, and U. Zölzer, *DAFX: Digital Audio Effects*, chapter 2, pp. 47–81, John Wiley & Sons, Ltd, 2011.

[58] Amrik Singh and Narwant Singh Grewal, "Review on FIR filter designing by implementations of different optimization algorithms," *Int J Adv Inf Sci Technol*, vol. 31, no. 31, pp. 171–175, 2014.

[59] Choon Ki Ahn, Peng Shi, and Sung Hyun You, "A new approach on design of a digital phase-locked loop," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 600–604, 2016.

[60] J. McClellan, T. Parks, and L. Rabiner, "A computer program for designing optimum FIR linear phase digital filters," *IEEE Transactions on Audio and Electroacoustics*, vol. 21, no. 6, pp. 506–526, 1973.

[61] K. Steiglitz, T. W. Parks, and J. F. Kaiser, "Meteor: a constraint-based FIR filter design program," *IEEE Transactions on Signal Processing*, vol. 40, no. 8, pp. 1901–1909, 1992.

[62] J.W. Adams and J.L. Sullivan, "Peak-constrained least-squares optimization," *IEEE Transactions on Signal Processing*, vol. 46, no. 2, pp. 306–321, 1998.

[63] W.-S. Lu, "Design of stable minimax IIR digital filters using semidefinite programming," in *2000 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2000, vol. 1, pp. 355–358 vol.1.

[64] Dennis Wei, "Non-convex optimization for the design of sparse FIR filters," in *2009 IEEE/SP 15th Workshop on Statistical Signal Processing*, 2009, pp. 117–120.

[65] Ryo Matsuoka, Seisuke Kyochi, Shunsuke Ono, and Masahiro Okuda, "Joint sparsity and order optimization based on ADMM with non-uniform group hard thresholding," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 5, pp. 1602–1613, 2018.

[66] Daniel Gabay and Bertrand Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976.

[67] Xiangming Xi and Yunjiang Lou, "Sparse FIR filter design with $k$-max sparsity and peak error constraints," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 4, pp. 1497–1501, 2021.

[68] Xiaoping Lai and Zhiping Lin, "Minimax design of IIR digital filters using a sequential constrained least-squares method," *IEEE Transactions on Signal Processing*, vol. 58, no. 7, pp. 3901–3906, 2010.

[69] Bingo Wing-Kuen Ling, CZ Wu, Kok Lay Teo, and Volker Rehbock, "Global optimal design of IIR filters via constraint transcription and filled function methods," *Circuits, Systems, and Signal Processing*, vol. 32, no. 3, pp. 1313–1334, 2013.

[70] Aimin Jiang and Hon Keung Kwan, "Minimax design of IIR digital filters using iterative SOCP," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 6, pp. 1326–1337, 2010.

[71] He Qi, Zhi Guo Feng, Ka Fai Cedric Yiu, and Sven Nordholm, "Optimal design of IIR filters via the partial fraction decomposition method," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 8, pp. 1461–1465, 2019.

[72] K. F. Man, K. S. Tang, and S. Kwong, "Genetic algorithms: concepts and applications [in engineering design]," *IEEE Transactions on Industrial Electronics*, vol. 43, no. 5, pp. 519–534, 1996.

[73] D. J. Krusienski and W. K. Jenkins, "Adaptive filtering via particle swarm optimization," in *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, 2003, vol. 1, pp. 571–575 Vol.1.

[74] Durbadal Mandal, Rajib Kar, Sakti Prasad Ghoshal, et al., "Digital FIR filter design using fitness based hybrid adaptive differential evolution with particle swarm optimization," *Natural Computing*, vol. 13, no. 1, pp. 55–64, 2014.

[75] Luis M. San-José-Revuelta and Juan Ignacio Arribas, "A new approach for the design of digital frequency selective fir filters using an FPA-based algorithm," *Expert Systems with Applications*, vol. 106, pp. 92–106, 2018.

[76] S. K. Saha, R. Dutta, R. Choudhury, R. Kar, D. Mandal, and S. P. Ghoshal, "Efficient and accurate optimal linear phase FIR filter design using opposition-based harmony search algorithm," *The Scientific World Journal*, vol. 2013, 2013.

[77] Amir A. Bature and Sunusi S. Adamu, "Design of digital recursive filter using artificial neural network," 2012.

[78] Boris Kuznetsov, Julian D. Parker, and Fabián Esqueda, "Differentiable IIR filters for machine learning applications," in *Proc. Int. Conf. Digital Audio Effects (eDAFx-20)*, 2020, pp. 297–303.

[79] P. Campolucci, A. Uncini, and F. Piazza, "Fast adaptive IIR-MLP neural networks for signal processing applications," in *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, 1996, vol. 6, pp. 3529–3532 vol. 6.

[80] Xiao-Hua Wang and Yi-Gang He, "A neural network approach to FIR filter design using frequency-response masking technique," *Signal Processing*, vol. 88, no. 12, pp. 2917–2926, 2008.

[81] Harpreet Kaur and Balwinder Dhaliwal, "Design of low pass FIR filter using artificial neural network," *International Journal of Information and Electronics Engineering*, vol. 3, no. 2, pp. 204, 2013.

[82] M. A. Singh and V. B. V. Thakare, "Artificial neural network use for design low pass FIR filter a comparison," *International Journal of Electronics and Electrical Engineering IJEEE*, vol. 3, no. 3, pp. 216–219, 2015.

[83] M. Kumari, M. Kumar, R. Saxena, and A. Wal, "Performance analysis of FIR low pass filter using artificial neural network," *Int. J. Eng. Trends Technol*, vol. 50, pp. 58–62, 2017.

[84] Stefania Cecchi, Alberto Carini, and Sascha Spors, "Room response equalization—a review," *Applied Sciences*, vol. 8, no. 1, 2018.

[85] M. Karjalainen, T. Paatero, J. N. Mourjopoulos, and P. D. Hatziantoniou, "About room response equalization and dereverberation," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005.*, 2005, pp. 183–186.

[86] Aki Mäkivirta, Poju Antsalo, Matti Karjalainen, and Vesa Välimäki, "Modal equalization of loudspeaker - room responses at low frequencies," *Journal of the Audio Engineering Society*, vol. 51, no. 5, pp. 324–343, May 2003.

[87] Matti Karjalainen, Esa Piirilä, Antti Järvinen, and Jyri Huopaniemi, "Comparison of loudspeaker equalization methods based on DSP techniques," *Journal of the Audio Engineering Society*, vol. 47, no. 1/2, pp. 14–31, January/February 1999.

[88] Matti Karjalainen, Paulo A. A. Esquef, Poju Antsalo, Aki Mäkivirta, and Vesa Välimäki, "Frequency-zooming ARMA modeling of resonant

and reverberant systems," *Journal of the Audio Engineering Society*, vol. 50, no. 12, pp. 1012–1029, December 2002.

[89] Panagiotis D. Hatziantoniou and John N. Mourjopoulos, "Errors in real-time room acoustics dereverberation," *Journal of the Audio Engineering Society*, vol. 52, no. 9, pp. 883–899, September 2004.

[90] Stefania Cecchi, Lorenzo Palestini, Paolo Peretti, Francesco Piazza, Ferruccio Bettarelli, and Romolo Toppi, "Automotive audio equalization," June 2009.

[91] David M. Howard and Jamie A. S. Angus, "Chapter 1 - introduction to sound," in *Acoustics and Psychoacoustics (Fourth Edition)*, David M. Howard and Jamie A. S. Angus, Eds., pp. 1–72. Focal Press, Boston, fourth edition edition, 2010.

[92] Panagiotis D. Hatziantoniou and John N. Mourjopoulos, "Generalized fractional-octave smoothing of audio and acoustic responses," *Journal of the Audio Engineering Society*, vol. 48, no. 4, pp. 259–280, April 2000.

[93] Germán Ramos and José J. López, "Filter design method for loudspeaker equalization based on iir parametric filters," *Journal of the Audio Engineering Society*, vol. 54, no. 12, pp. 1162–1178, December 2006.

[94] Stephen T. Neely and J. B. Allen, "Invertibility of a room impulse response," *The Journal of the Acoustical Society of America*, vol. 66, no. 1, pp. 165–169, 1979.

[95] J. Mourjopoulos, P. Clarkson, and J. Hammond, "A comparative study of least-squares and homomorphic techniques for the inversion of mixed phase signals," in *ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1982, vol. 7, pp. 1858–1861.

[96] Y. Haneda, S. Makino, and Y. Kaneda, "Common acoustical pole and zero modeling of room transfer functions," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 320–328, 1994.

[97] O. Kirkeby, P.A. Nelson, H. Hamada, and F. Orduna-Bustamante, "Fast deconvolution of multichannel systems using regularization," *IEEE Transactions on Speech and Audio Processing*, vol. 6, no. 2, pp. 189–194, 1998.

[98] Barry D. Kulp, "Digital equalization using fourier transform techniques," *Audio Engineering Society Convention 85, Los Angeles, CA, USA*, November 1988.

[99] Peter M. Clarkson, John Mourjopoulos, and J. K. Hammond, "Spectral, phase, and transient equalization for audio systems," *Journal of the Audio Engineering Society*, vol. 33, no. 3, pp. 127–132, March 1985.

[100] Stephen J. Elliott and Philip A. Nelson, "Multiple-point equalization in a room using adaptive digital filters," *Journal of the Audio Engineering Society*, vol. 37, no. 11, pp. 899–907, November 1989.

[101] W. Putnam, D. Rocchesso, and J. Smith, "A numerical investigation of the invertibility of room transfer functions," in *Proceedings of 1995 Workshop on Applications of Signal Processing to Audio and Accoustics*, 1995, pp. 249–252.

[102] A. G. Constantinides, "Spectral transformations for digital filters," *Proceedings of the Institution of Electrical Engineers*, vol. 117, pp. 1585–1590(5), August 1970.

[103] F. Keiler and U. Zolzer, "Parametric second- and fourth-order shelving filters for audio applications," in *IEEE 6th Workshop on Multimedia Signal Processing, 2004.*, 2004, pp. 231–234.

[104] Robert Bristow-Johnson, "The equivalence of various methods of computing biquad coefficients for audio parametric equalizers," *Audio Engineering Society Convention 97, San Francisco, CA, USA*, November 1994.

[105] Yong Lian and Yong Ching Lim, "Linear-phase digital audio tone control using multiplication-free fir filter," *Journal of the Audio Engineering Society*, vol. 41, no. 10, pp. 791–794, October 1993.

[106] Fred Harris and Eric Brooking, "A versatile parametric filter using an imbedded all-pass subfilter to independently adjust bandwidth, center freqeuncy, and boost or cut," October 1993.

[107] Jussi Rämö, Vesa Välimäki, and Balázs Bank, "High-precision parallel graphic equalizer," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 12, pp. 1894–1904, 2014.

[108] Jørgen Arendt Jensen, "A new principle for an all digital preamplifier and equalizer," *Journal of the Audio Engineering Society*, vol. 35, no. 12, pp. 994–1003, December 1987.

[109] Joung-Woo Choi and Yang-Hann Kim, "Generation of an acoustically bright zone with an illuminated region using multiple sources," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1695–1700, 2002.

[110] P. Hatziantoniou, D. Tsoukalas, J. Mourjopoulos, and S. Salamouris, "Time-frequency mapping based on non-uniform smoothed spectral representations," in *Proceedings of the Acoustics, Speech, and Signal Processing, 1999. on 1999 IEEE International Conference - Volume 03*, USA, 1999, ICASSP '99, p. 1425–1428, IEEE Computer Society.

[111] Matti Karjalainen, "Auditory interpretation and application of warped linear prediction," in *Proceedings of Consistent & Reliable Acoustic Cues for Sound Analysis, Aalborg, Denmark*, September 2001.

[112] A. Oppenheim, D. Johnson, and K. Steiglitz, "Computation of spectra with unequal resolution using the fast fourier transform," *Proceedings of the IEEE*, vol. 59, no. 2, pp. 299–301, 1971.

[113] W. Kautz, "Transient synthesis in the time domain," *Transactions of the IRE Professional Group on Circuit Theory*, vol. CT-1, no. 3, pp. 29–39, 1954.

[114] Paul W. Broome, "Discrete orthonormal sequences," *J. ACM*, vol. 12, no. 2, pp. 151–168, April 1965.

[115] Ronald P. Genereux, "Adaptive loudspeaker systems: Correcting for the acoustic environment," *Journal of the Audio Engineering Society*, May 1990.

[116] M. Kallinger and A. Mertins, "Room impulse response shortening by channel shortening concepts," in *Conference Record of the Thirty-Ninth Asilomar Conference on Signals, Systems and Computers*, 2005, pp. 898–902.

[117] Richard K. Martin, Ming Ding, Brian L. Evans, and C. Richard Johnson, "Efficient channel shortening equalizer design," *EURASIP Journal on Advances in Signal Processing*, 2003.

[118] Dale Reed, "A perceptual assistant to do sound equalization," in *Proceedings of the 5th International Conference on Intelligent User Interfaces*, New York, NY, USA, 2000, IUI '00, p. 212–218, Association for Computing Machinery.

[119] Marco A. Martinez Ramirez and Joshua D. Reiss, "End-to-end equalization with convolutional neural networks," in *Proceedings of the 21st International Conference on Digital Audio Effects (DAFx-18),Aveiro, Portugal*, 2018, Retrieved from: http://dafx2018.web.ua.pt.

[120] Po-Rong Chang, C. G. Lin, and Bao-Fuh Yeh, "Inverse filtering of a loudspeaker and room acoustics using time-delay neural networks," *The*

*Journal of the Acoustical Society of America*, vol. 95, no. 6, pp. 3400–3408, 1994.

[121] Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, Luca Cattani, and Carlo Tripodi, "Gravitational search algorithm for IIR filter-based audio equalization," in *2020 28th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 496–500.

[122] Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, and Luca Cattani, "Evolutionary tuning of filters coefficients for binaural audio equalization," *Applied Acoustics*, vol. 163, pp. 107204, 2020.

[123] Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, and Luca Cattani, "Designing audio equalization filters by deep neural networks," *Applied Sciences*, vol. 10, no. 7, 2020.

[124] Wancheng Zhang, Andy W. H. Khong, and Patrick A. Naylor, "Adaptive inverse filtering of room acoustics," in *2008 42nd Asilomar Conference on Signals, Systems and Computers*, 2008, pp. 788–792.

[125] Ajay Dagar, Satyavolu Sai Nitish, and Rajesh Hegde, "Joint adaptive impulse response estimation and inverse filtering for enhancing in-car audio," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 416–420.

[126] Shaymah Yasear and Angela Amphawan, "Channel impulse response equalization scheme based on particle swarm optimization algorithm in mode division multiplexing," *EPJ Web Conf.*, vol. 162, pp. 01023, 2017.

[127] Ali A. Al-Shaikhi, Adil H. Khan, Ali T. Al-Awami, and Azzedine Zerguine, "A hybrid particle swarm optimization technique for adaptive equalization," *Arabian Journal for Science and Engineering*, June 2018.

[128] Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, Luca Cattani, and Carlo Tripodi, "Generative adversarial networks for audio equalization: an evaluation study," May 2020.

[129] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros, "Context encoders: Feature learning by inpainting," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 2536–2544.

[130] A. Farina, "Advancements in impulse response measurements by sine sweeps," in *Audio Engineering Society Convention 122*, May 2007.

[131] Robert Hooke and T. A. Jeeves, "Direct search solution of numerical and statistical problems," *J. ACM*, vol. 8, no. 2, pp. 212–229, April 1961.

[132] Robert Michael Lewis, Virginia Torczon, and Michael W. Trosset, "Direct search methods: then and now," *Journal of Computational and Applied Mathematics*, vol. 124, no. 1, pp. 191–207, 2000, Numerical Analysis 2000. Vol. IV: Optimization and Nonlinear Equations.

[133] Herwig Behrends, Adrian Von Dem Knesebeck, Werner Bradinal, Peter Neumann, and Udo Zölzer, "Automatic equalization using parametric IIR filters," *Journal of the Audio Engineering Society*, vol. 59, no. 3, pp. 102–109, March 2011.

[134] W. Wirtinger, "Zur formalen theorie der funktionen von mehr komplexen veränderlichen," *Mathematische Annalen*, vol. 97, pp. 357–375, 1927.

[135] Hugo Caracalla and Axel Roebel, "Gradient conversion between Time and Frequency Domains Using Wirtinger Calculus," in *DAFx 2017*, Edinburgh, United Kingdom, September 2017.

[136] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein, "Implicit neural representations with periodic activation functions," 2020, vol. 33.

[137] Terence Betlehem, Wen Zhang, Mark A. Poletti, and Thushara D. Abhayapala, "Personal sound zones: Delivering interface-free audio to multiple listeners," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 81–91, 2015.

[138] Jordan Cheer and Stephen Elliott, "Design and implementation of a personal audio system in a car cabin," *Proceedings of Meetings on Acoustics*, vol. 19, no. 1, pp. 3251, 2013.

[139] Ji-Ho Chang, Chan-Hui Lee, Jin-Young Park, and Yang-Hann Kim, "A realization of sound focused personal audio system using acoustic contrast control," *The Journal of the Acoustical Society of America*, vol. 125, no. 4, pp. 2091–2097, 2009.

[140] Jin-Young Park, Ji-Ho hang, and Yang-Hann Kim, "Generation of independent bright zones for a two-channel private audio system," *Journal of the Audio Engineering Society*, vol. 58, no. 5, pp. 382–393, May 2010.

[141] Jung-Min Lee, T Lee, Jin-Young Park, and Yang-Hann Kim, "Generation of a private listening zone; acoustic parasol," in *Proceedings of 20-th International Congress on Acoustic, ICA 2010, Sidney, Australia*, August 2010.

[142] Stephen J. Elliott and Matthew Jones, "An active headrest for personal audio," *The Journal of the Acoustical Society of America*, vol. 119, no. 5, pp. 2702–2709, 2006.

[143] Stephen J. Elliott, Jordan Cheer, Harry Murfet, and Keith R. Holland, "Minimally radiating sources for personal audio," *The Journal of the Acoustical Society of America*, vol. 128, no. 4, pp. 1721–1728, 2010.

[144] Jordan Cheer, Stephen J. Elliott, Youngtae Kim, and Jung-Woo Choi, "Practical implementation of personal audio in a mobile device," *Journal of the Audio Engineering Society*, vol. 61, no. 5, pp. 290–300, May 2013.

[145] Xiangning Liao, Jordan Cheer, Stephen j. Elliott, and Sifa Zheng, "Design of a loudspeaker array for personal audio in a car cabin," *Journal of the Audio Engineering Society*, vol. 65, no. 3, pp. 226–238, March 2017.

[146] Michele Ebri, Nicolo Strozzi, Filippo Maria Fazi, Angelo Farina, and Luca Cattani, "Individual listening zone with frequency-dependent trim of measured impulse responses," *Journal of the Audio Engineering Society*, October 2020.

[147] Mincheol Shin, Sung Q. Lee, Filippo M. Fazi, Philip A. Nelson, Daesung Kim, Semyung Wang, Kang Ho Park, and Jeongil Seo, "Maximization of acoustic energy difference between two spaces," *The Journal of the Acoustical Society of America*, vol. 128, no. 1, pp. 121–131, 2010.

[148] Philip Coleman, Philip J. B. Jackson, Marek Olik, Martin Møller, Martin Olsen, and Jan Abildgaard Pedersen, "Acoustic contrast, planarity and robustness of sound zone methods using a circular loudspeaker array," *The Journal of the Acoustical Society of America*, vol. 135, no. 4, pp. 1929–1940, 2014.

[149] Philip Coleman, Philip Jackson, Marek Olik, and Jan Abildgaard Pedersen, "Optimizing the planarity of sound zones," *Audio Engineering Society Convention 52, New York, NY, USA*, September 2013.

[150] Khan Baykaner, Philip Coleman, Russell Mason, Philip J. B. Jackson, Jon Francombe, Marek Olik, and Søren Bech, "The relationship between target quality and interference in sound zone," *Journal of the Audio Engineering Society*, vol. 63, no. 1/2, pp. 78–89, January 2015.

[151] W. F. Druyvesteyn and John Garas, "Personal sound," *Journal of the Audio Engineering Society*, vol. 45, no. 9, pp. 685–701, September 1997.

[152] Jon Francombe, Russell Mason, Martin Dewhirst, and Søren Bech, "Determining the threshold of acceptability for an interfering audio programme," *Journal of the Audio Engineering Society*, April 2012.

[153] Adrian P. Simpson, "Phonetic differences between male and female speech," *Language and Linguistics Compass*, vol. 3, no. 2, pp. 621–640, 2009.

[154] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, 2001, vol. 2, pp. 749–752 vol.2.

[155] Cees H. Taal, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen, "An algorithm for intelligibility prediction of time–frequency weighted noisy speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2125–2136, 2011.

[156] Michael Chinen, Felicia S. C. Lim, Jan Skoglund, Nikita Gureev, Feargus O'Gorman, and Andrew Hines, "ViSQOL v3: An open source production ready objective speech and audio metric," in *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, 2020, pp. 1–6.

[157] Hines A., Skiglund J., Kikaram A. C., and N. Harte, "ViSQOL: an objective speech quality model," *EURASIP Journal of Audio Speech Music Processing, 2015*, , no. 1, pp. 13, 2015.

[158] S. J. Elliott, J. Cheer, J. Choi, and Y. Kim, "Robustness and regularization of personal audio systems," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 7, pp. 2123–2133, 2012.

[159] Joung-Woo Choi and Yang-Hann Kim, "Generation of an acoustically bright zone with an illuminated region using multiple sources," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1695–1700, 2002.

[160] T. Betlehem, W. Zhang, M. A. Poletti, and T. D. Abhayapala, "Personal sound zones: Delivering interface-free audio to multiple listeners," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 81–91, 2015.

[161] T. Betlehem and P. D. Teal, "A constrained optimization approach for multi-zone surround sound," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 437–440.

[162] Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, Luca Cattani, and Carlo Tripodi, "Deep learning for individual listening zone," in *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, 2020, pp. 1–6.

[163] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 5206–5210.

[164] Giovanni Pepe, Leonardo Gabrielli, Livio Ambrosini, Stefano Squartini, and Luca Cattani, "Detecting road surface wetness using microphones and convolutional neural networks," *Audio Engineering Society Convention 146, Dublin, Ireland*, March 2019.

[165] Xinyu Zhang, Hongbo Gao, Mu Guo, Guopeng Li, Yuchao Liu, and Deyi Li, "A study on key technologies of unmanned driving," *CAAI Transactions on Intelligence Technology*, vol. 1, no. 1, pp. 4–13, 2016.

[166] Livio Ambrosini, Leonardo Gabrielli, Fabio Vesperini, Stefano Squartini, and Luca Cattani, "Deep neural networks for road surface roughness classification from acoustic signals," *Audio Engineering Society Convention 144, Milan, Italy*, May 2018.

[167] Rajesh Singh, Pinaki Mondal, Nitin Sharma, Abhishek Kumar, U. D. Bhangale, and Dinesh Tyagi, "Effect of rainfall and wet road condition on road crashes : A critical analysis," in *SIAT 2011*. January 2011, The Automotive Research Association of India.

[168] Pinaki Mondal, Nitin Sharma, Abhishek Kumar, Prashant Vijay, U. D. Bhangale, and Dinesh Tyagi, "Are road accidents affected by rainfall? a case study from a large indian metropolitan city," *Current Journal of Applied Science and Technology*, pp. 16–26, 2011.

[169] Junoh Ahmad Kadri, Wan Muhamad Wan Zuki Azman, Mohd Nopiah Zulkifli, Mohd Jailani Mohd Nor, Ahmad Kamal Ariffin, and Mohammad Hosseini Fouladi, "A study on the effects of tyre to vehicle acoustical comfort in passenger car cabin," in *2011 3rd International Conference on Computer Research and Development*, 2011, vol. 4, pp. 342–345.

[170] Irman Abdić, Lex Fridman, Daniel E. Brown, William Angell, Bryan Reimer, Erik Marchi, and Björn Schuller, "Detecting road surface wetness from audio: A deep learning approach," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, 2016, pp. 3458–3463.

[171] Leonardo Gabrielli, Livio Ambrosini, Fabio Vesperini, Valeria Bruschi, Stefano Squartini, and Luca Cattani, "Processing acoustic data with siamese neural networks for enhanced road roughness classification," in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019, pp. 1–7.

[172] Dağhan Doğan, "Road-types classification using audio signal processing and SVM method," in *2017 25th Signal Processing and Communications Applications Conference (SIU)*, 2017, pp. 1–4.

[173] J. Alonso, J.M. López, I. Pavón, M. Recuero, C. Asensio, G. Arcas, and A. Bravo, "On-board wet road surface identification using tyre/road noise and support vector machines," *Applied Acoustics*, vol. 76, pp. 407–415, 2014.

[174] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[175] Giovanni Pepe, Leonardo Gabrielli, Emanuele Principi, Stefano Squartini, and Luca Cattani, *Road Type Classification Using Acoustic Signals: Deep Learning Models and Real-Time Implementation*, pp. 33–43, Springer Singapore, Singapore, 2021.

[176] Florian Eyben, Felix Weninger, Florian Gross, and Björn Schuller, "Recent developments in opensmile, the munich open-source multimedia feature extractor," in *Proceedings of the 21st ACM International Conference on Multimedia*, New York, NY, USA, 2013, MM '13, p. 835–838, Association for Computing Machinery.

[177] Florian Eyben, Felix Weninger, Florian Gross, and Björn Schuller, "Recent developments in openSMILE, the munich open-source multimedia feature extractor," in *Proc. of ACM Multimedia 2013*, Barcelona, Spain, 2013, pp. 835–838, ACM.

[178] Paolo Vecchiotti, Giovanni Pepe, Emanuele Principi, and Stefano Squartini, "Detection of activity and position of speakers by using deep neural networks and acoustic data augmentation," *Expert Systems with Applications*, vol. 134, pp. 53–65, 2019.

[179] Diego Augusto Silva, José Augusto Stuchi, Ricardo P Velloso Violato, and Luís Gustavo D Cuozzo, *Exploring convolutional neural networks for voice activity detection*, pp. 37–47, Springer International Publishing, 2017.

[180] Robert E. Yantorno, Kasturi Rangan Krishnamachari, Jereme M. Lovekin, Daniel S. Benincasa, and Stanley J. Wenndt, "The Spectral Autocorrelation Peak Valley Ratio (SAPVR) - Usable Speech Measure Employed as a Co-channel Detection System," in *Proceedings of International Workshop on Intelligent Signal Processing (WISP)*, 2001.

[181] A. Lee, K. Nakamura, R. Nisimura, H. Saruwatari, and K. Shikano, "Noise robust real world spoken dialogue system using gmm based rejection of unintended inputs," in *Proceedings of 8th International Conference on Spoken Language Processing (ICSLP)*, 2004, pp. 173–176.

[182] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 4, pp. 320–327, August 1976.

[183] Jose A. Belloch, Alberto Gonzalez, Antonio M. Vidal, and Maximo Cobos, "On the performance of multi-gpu-based expert systems for acoustic localization involving massive microphone arrays," *Expert Systems with Applications*, vol. 42, no. 13, pp. 5607 – 5620, 2015.

[184] T. Hughes and K. Mierle, "Recurrent neural networks for voice activity detection," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2013, pp. 7378–7382.

[185] Fabio Vesperini, Paolo Vecchiotti, Emanuele Principi, Stefano Squartini, and Francesco Piazza, "Deep neural networks for multi-room voice activity detection: Advancements and comparative evaluation," in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 3391–3398.

[186] N. Ma, T. May, and G. J. Brown, "Exploiting deep neural networks and head movements for robust binaural localization of multiple sources in reverberant environments," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 12, pp. 2444–2453, December 2017.

[187] Marko Kovandžić, Vlastimir Nikolić, Abdulathim Al-Noori, Ivan Ćirić, and Miloš Simonović, "Near field acoustic localization under unfavorable conditions using feedforward neural network for processing time difference of arrival," *Expert Systems with Applications*, vol. 71, pp. 138 – 146, 2017.

[188] E. L. Ferguson, S. B. Williams, and C. T. Jin, "Sound source localization in a multipath environment using convolutional neural networks," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2018, pp. 2386–2390.

[189] Weipeng He, Petr Motlicek, and Jean-Marc Odobez, "Deep neural networks for multiple speaker detection and localization," in *Proceedings of International Conference on Robotics and Automation (ICRA)*, 2018, pp. 74–79.

[190] Fabio Vesperini, Paolo Vecchiotti, Emanuele Principi, Stefano Squartini, and Francesco Piazza, "Localizing speakers in multiple rooms by using deep neural networks," *Computer Speech & Language*, vol. 49, pp. 83–106, 2018.

[191] S. Chakrabarty and E. A. P. Habets, "Broadband DOA estimation using convolutional neural networks trained with noise signals," in *Proceedings of Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, October 2017, pp. 136–140.

[192] Paolo Vecchiotti, Emanuele Principi, Stefano Squartini, and Francesco Piazza, "Deep neural networks for joint voice activity detection and speaker localization," in *Proceedings of 26th European Signal Processing Conference (EUSIPCO)*, September 2018, pp. 1567–1571.

[193] Yuuki Tachioka, Tomohiro Narita, Shinji Watanabe, and Jonathan Le Roux, "Ensemble integration of calibrated speaker localization and statistical speech detection in domestic environments," in *Proceedings of Hands-free Speech Communication and Microphone Arrays (HSCMA)*, 2014, pp. 162–166.

[194] Luca Cristoforetti, Mirco Ravanelli, Maurizio Omologo, Alessandro Sosi, Alberto Abad, Martin Hagmüller, and Petros Maragos, "The DIRHA simulated corpus," in *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC'14)*, May 2014, pp. 2629–2634.

[195] Sunit Sivasankaran, Emmanuel Vincent, and Douglas R. Campbell, "Room impulse response generator," 2017.

[196] Jont B. Allen and David A. Berkley, "Image method for efficiently simulating small-room acoustics," *The Journal of the Acoustical Society of America*, vol. 65, no. 4, pp. 943–950, 1979.

[197] Solomon W. Golomb and Guang Gong, *Signal design for good correlation: for wireless communication, cryptography, and radar*, Cambridge University Press, 2005.

[198] Samuele Cornell, Michel Olvera, Manuel Pariente, Giovanni Pepe, Emanuele Principi, Leonardo Gabrielli, and Stefano Squartini, "Task-Aware Separation for the DCASE 2020 Task 4 Sound Event Detection

and Separation Challenge," in *DCASE 2020 - 5th Workshop on Detection and Classification of Acoustic Scenes and Events*, Virtual, Japan, November 2020.

[199] Karol J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6.

[200] Emre Çakır, Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, 2017.

[201] Yun Wang, Juncheng Li, and Florian Metze, "A comparison of five multiple instance learning pooling functions for sound event detection with weak labeling," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 31–35.

[202] Antti Tarvainen and Harri Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. 2017, vol. 30, Curran Associates, Inc.

[203] Ilya Kavalerov, Scott Wisdom, Hakan Erdogan, Brian Patton, Kevin Wilson, Jonathan Le Roux, and John R. Hershey, "Universal sound separation," in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, pp. 175–179.

[204] Thilo von Neumann, Keisuke Kinoshita, Lukas Drude, Christoph Boeddeker, Marc Delcroix, Tomohiro Nakatani, and Reinhold Haeb-Umbach, "End-to-end training of time domain audio separation and recognition," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7004–7008.

[205] Romain Serizel, Nicolas Turpault, Ankit Shah, and Justin Salamon, "Sound event detection in synthetic domestic environments," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 86–90.

[206] Gert Dekkers, Steven Lauwereins, Bart Thoen, Mulu Weldegebreal Adhana, Henk Brouckxon, Bertold Van den Bergh, Toon van Waterschoot, Bart Vanrumste, Marian Verhelst, and Peter Karsmakers, "The SINS database for detection of daily activities in a home environment using an

acoustic sensor network," *Detection and Classification of Acoustic Scenes and Events 2017*, 2017.

[207] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen, "TUT database for acoustic scene classification and sound event detection," in *24th European Signal Processing Conference 2016 (EUSIPCO 2016)*, Budapest, Hungary, 2016.

[208] Scott Wisdom, Hakan Erdogan, Daniel P. W. Ellis, Romain Serizel, Nicolas Turpault, Eduardo Fonseca, Justin Salamon, Prem Seetharaman, and John R. Hershey, "What's all the fuss about free universal sound separation data?," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 186–190.

[209] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky, *Domain-Adversarial Training of Neural Networks*, pp. 189–209, Springer International Publishing, Cham, 2017.

[210] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[211] Yuxuan Wang, Pascal Getreuer, Thad Hughes, Richard F. Lyon, and Rif A. Saurous, "Trainable frontend for robust and far-field keyword spotting," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5670–5674.

[212] Lionel Delphin-Poulat and Cyril Plapous, "Mean teacher with data augmentation for dcase 2019 task 4," *DCASE 2019 Tech Report*, 2019.

[213] Dong Yu, Morten Kolbæk, Zheng-Hua Tan, and Jesper Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 241–245.