POLITECNICO DI TORINO

DEPARTMENT OF COMPUTER AND CONTROL ENGINEERING

Master degree course in Mechatronic Engineering

Master Degree Thesis

Bidimensional Feature Extraction from an Audio Signal

Reasons and development of an algorithm capable of obtaining information from the analysis of an audio signal and its practical applications

Supervisor
prof. Michele Taragna

Laureando
Andrea Giuseppe Calabrò
ID number: 250334

April 2020

† A mia nonna, che aspettava tanto questo momento e purtroppo l'ha perso per poco. Mi hai insegnato cosa sia l'amore incondizionato e che "A cucinare si comincia andando a fare la spesa".

Abstract

In the music industry there is a "hidden" engine: this produces 1.7 million hours of content every month. The objective of the proposed thesis is to extract information from this generator, which is fed by DJs all year round. To give an order of magnitude: the database that could be analysed boasts about 370 million hours of music, while the total of Spotify's one, Europe's largest music streaming app, is around 275 million. Thanks to an analytical approach, which is the result of the synthesis of knowledge and skills obtained during the years of university studies, an algorithm has been designed and implemented to analyse the DJ sets (the components of this archive) giving millions of people access to millions of data. We can think of each song as ingredients, specially selected by the artist (also called "selector"), to go to form a dish, our DJ set: the instrumentation that allows this process, the kitchen, is called mixer or console. A fundamental characteristic of each mixer is the possibility to modulate the volumes and frequency spectrum i.e., decide to increase or decrease the energy of sounds at certain frequencies, of each song. Going into detail: each console is always composed of various input channels, each with the necessary instrumentation to apply the above changes and on which one track can be loaded at a time; and only one output channel from which a new audio signal passes, composed by the union of the modified input ones. Each channel is formed by two amplifiers, which allow the total volume control; and by a series of filters, with different working intervals, whose task is to allow changes in frequency. The algorithm, starting from the output measurement i.e., the result of the work done by the DJ through the mixer, and thanks to the previous knowledge of the inputs i.e., the unmodified songs, aims to trace how the artist decided to work to get the final product. The thesis, put in canonical terms, can be defined as a problem of non-linear time varying reverse filtering, applied to a system with multiple inputs and only one output. The work, after having assessed the feasibility of the project, has been developed with an inductive method: starting from the characterization of each single filter, passing through the creation of a model for each channel, we have arrived at a recursive algorithm able to obtain an indication of the status of each filter and the volume of the channel for each instant of sampling. The obtained result stops at the possibility to analyse the signal obtained with the use of a single channel, the first continuation of the work will be to make the algorithm able to work when several channels are used simultaneously. The applications are various, and some will be presented in the thesis, as a Graphic Interface able to show how the artist is using the mixer. Another interesting continuation can be seen in the feeding of an Artificial Intelligence in order to make it generate a DJ set not only based on the mathematical calculation built on waveforms, but adding a more human touch.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The past decade has seen a huge change in the way people communicate, shop, watch TV or listen to music. When someone ask for road directions, everyone usually does what thinks that the requester could have done by itself: open Google Maps. This *"mobile internet"* era has now given way to the next: the data one. It is become clearer how the classic business model, based only on the customer paying for a service or a product, is not working good anymore. It is possible to think the same for the products on the market which do not support integration with data, that instead would lead to an always better customization (ask to the automotive market players[1]).
The objective of this thesis is to navigate in the same direction of the current: an algorithm able to extrapolate specific features from an audio signal will be shown, as some of the possible usage of these obtained information.

1.1 Object

Signal processing is the most commonly applied methodology when dealing with feature extraction and classification from audio signals, when employed in this field its primary goal is to provide specific information. The work will be relative to a non linear time-varying environment, and therefore will present the obvious issues that this entails. As will be discussed in a deeper way in the next section 1.2, the final aim of the dissertation is to trace the changes that DJs perform on songs. The system to be modelled is the Hardware through which an artist modifies the tracks, in order to mix together several tunes, generating a *DJ set* i.e., a series of songs mixed together to generate a continuous audio signal. This instrument is also called mixer or, slightly incorrectly, console, and now is going to be presented for seek of completeness. As appreciable in figure 1.1, a mixer is principally an ensemble of rotary knobs and some "linear modifiers".

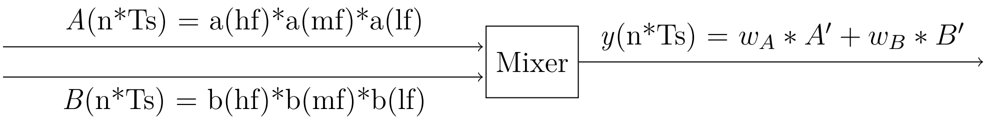
[1]The 16^{th} of January 2020 *REUTERS* reported Volkswagen Chief Executive Herbert Diess said the German carmaker needs to accelerate its transformation to avoid becoming another Nokia , which lost its dominance in the handset market to Apple. His words were: *"The car is no longer a mode of transport and carmakers are no longer only manufacturers of vehicles"*.

Leaving the graphical division in an upper and lower operating area, a crucial aspect to be defined is the one of channel. In picture it is displayed a two-channel mixer, that's why four vertical sets of objects can be counted. Each channel has the same peculiarity: it modifies the audio signal passing through it. The similarities don't end here: it is possible to load one signal at a time on each channel and the output of each is its modified input, while the total mixer output is the weighted sum of the outputs of all channels. Following the same advice that etiquette gives for cutlery, the explanation will start from those outside.
The left one is used to eventually, and hopefully not, connect a microphone to the mixer: the possible work on the signal passing through it is just about the volume. In particular the "*LEVEL*" is the first feasible modulation, it is analogical and is done before the Analog-to-Digital conversion, while the "*TRIM*" is completely digital and happens after the discretization.
The one found in the upper right position is defined as the **MASTER**, already from the name it is possible to understand its importance: its *LEVEL* knob is the one that controls the whole output volume. It is even able to modify what the artist listens through the headphones, these are connected in order to hear also what still needs to be added to the total output appreciated by the public. The reason for the use of a secondary listening channel just for the DJ, is to let him hear the unmixed songs together in order to perform the needed modification to get the desired result.
Finally, the central channels i.e., the real area of modulation. As it is graphically appreciable they are all made of the same components. Going down it is possible to see:

1. the already discussed *TRIM*, each one regards its own relative volume;
2. a triplets of filters, where each one works at a different interval (*HI* stands for High, *MID* Medium and *LOW* low frequencies) and is capable of performing an attenuation or an amplification;
3. the *COLOR*, another kind of possible frequency modulation, but instead of being restricted to a given interval, it spans on the whole frequency range;
4. the blue buttons *CUE* are the switches to activate the hearing of that given track via headphones, even if still is not "launched";
5. the vertical linear slider, called *FADER*, is a measure of the weight of each channel in the output sum i.e., a linear modulation of the influence on the total output of each channel, in terms of volume;
6. the horizontal linear slider, the *CROSS-FADER*, mostly used in genres as Hip-Hop to perform an abrupt switch between channels' weights on the output.

Usually the artist's workflow is the following: the first song, that will be identified as A, is loaded on its channel and eventually modified. Now the output y is completely given by the modified version of A i.e., *A'* = a'(hf)*a'(mf)*a'(lf) where hf,mf and lf indicate the frequency range of modification, with a strength controlled via the w_A weight. At a given instant, he or she, decides to add to the to output y a song B, loaded on the second channel, by means of suitable modulations which generates *B'* = b'(hf)*b'(mf)*b'(lf) weighted with w_B. At this point the updated output will be $y = w_A * A' + w_B * B'$.

There is not a state-of-the-art way to perform the mixing, but the constant idea is that the following track will have to became, at a certain instant, the only contribution to the output: it is possible to think to a process that aims to attenuate the A track and amplify the B track. So, after the modification, the new output will be $y = w_B * B'$, and the artist will start again with a third song, C.



The above mentioned alteration procedure is carried out entirely through the adjustment of the filters' value and of each channel's weight, and is permitted by the presented hardware.
The object of the thesis will be the study of the system's output signal.

1.2 Goals

The algorithm can be defined as a reverse filtering one: by means of the audio signal evaluation, the goal is to derive the states of the system-console i.e., how the DJ is mixing modifying the knobs' value.
Specifically there will be a series of features extracted from the signal, for every sampling instant, thanks to the process:

- each filter's working value, in percentage terms spanning from -100% to +100%,
- the offsets or weight values,
- the initial and final instant of the change

The percentage value indicates how much a filter is attenuating or amplifying the track in its frequencies' interval, and is obtained by means of an evaluation of the signal's energy at known ranges.

1.3 Relationship with other works

The result of the research lends itself to a variety of applications, principally thought in the music-tech's field. As explained in 2.2.3: the generation of a Web-app is the reason behind the study. But the signal process that will be presented has the peculiarity not only to work on the frequency axis, but in terms of time too: as stated before, there is also an indication of when the modification occurs (and is over).

1.3.1 Display how the DJ is playing (the WebApp)

The first link to be introduced is clearly the one that has given way to the whole. The extrapolation is addressed to be immediately exploited in this sense, as it is possible to understand from the feature regarding the filter's working value: this can be translated in

a graphical indication of the position of each knob in the channel.
A first beta-version of a Graphical Interface for the users has been developed, and both the database creation to link Back and Front End codes 4.1 and the GUI 4.2 will be discussed.

1.3.2 Augmented Recommendation

One of the best music's recommendation process will be presented in 2.2.2, in general the algorithms employed to extract features are very concentrated around the human voice, moreover they are always used for single songs. The one developed here focuses on the changes made by the system DJ-console on every single song, in order to obtain a DJ set. Thanks to this, a whole new range of information will be available to subdivide each artist into a specific category. It will also be possible to label songs according to how these are modified e.g., by looking at how the same tracks are used by different artists or finding different songs modulated in similar way.

1.3.3 Visual Arts during Live

The Visual Arts are a crucial part of the Live Performances: from small clubs to big festivals every venue is equipped to make the audiovisual experience the best possible. This analysis can led to an automation in the algorithms used by the people behind the adaptation of the show to what the DJ is doing: usually just the speed of the track and the overall volumes are automatically detected, while the technician has to tailor the evolution of the displayed images (also lasers, . . .) to the sound. By obtaining the exact value of the changes made, this information will only increase the possibilities of the software used in this area.

1.3.4 Real AI DJ

In the last years some "*Robot-DJ*" have made people talk about themselves, but mostly in negative ways. The algorithm behind the mixing is based on aligning the beats between two songs and does not go a lot further than mathematical calculations on musical notes. The present work permits the extraction and collection of tons of data that could be used to feed some AIs to make them learn how to properly mix. There will be the possibility to use this "*AI-DJ*" to generate suitable mixes out of playlists: Spotify has an automix option, and is a clear illustration of how much effort still needs to be exerted in this direction.

1.3.5 Time, not only frequency

The last example is also the most academic one. Usually this kind of extractions does not focus on the time information: it is just a question of frequency-dependent detection. By broadening the horizons to a two-dimensional analysis, it will be possible to obtain a more complete picture of the system studied.

1.4 Methods

Every time a new obstacle came up, a literature search was carried out, followed by the implementation of the proposed solutions. By means of a trial and error process, each idea was tested and compared with the others: the one giving better results, was therefore selected and, when needed, optimized in order to be suitable at best to fulfill the requested task.

The approach to execute the signal process is quite standard. The first performed operation is the one of pre-processing: in particular a reduction of the dimension of the vector carrying information about the audio signal is done, and also its normalization; another needed step is the one of aligning the modified track to its unmodified equivalent stored in the database.
From this audio signal, a suitable spectrum will be obtained, 2.3.2, a problem dimension's reduction will follow, thanks to a suitable analysis, and a clustering of the resulting time-series 2.3.3. The algorithm is useful first to obtain each filter characteristic, 3.2 and the system's model 3.3.3. The application of the results of the processes listed above to a real signal is presented in 3.4: this will also include a link to a GUI, exhibited in 4.

In order to be clear: this dissertation will cover the technological core of a wanna be start-up. To explain it only from a scientific point of view would have been reductive, so it is hopeful that the reader will have the right approach to the work. For example, the 2.2.3 subsection (where the reasons of the thesis are explained) will be a report of the tasks performed in order to understand what was really needed from the customers, showing how this idea is born through a proper validation process. It has to be stressed out how that part has a different linguistic style, since it can be seen as something by itself: a first smattering of the path to follow when anyone wants to create a company, starting from a product (the so-called *technology-push* ones).
Others non-technical considerations can be found all over the first section of the elaborate, particularly at 2.2 and in the Conclusions 5.

Figure 1.1: A 2 channel PIONEER DJMS3 MIXER

Chapter 2

Study and Signal Analysis

2.1 Introduction

The following chapter contains both the reasons behind the thesis and the work done, both theoretical and of implementation, which will then allow to analyze the audio signals to develop the system modelling process and the code for a recursive analysis presented in the next chapter 3.
The state of the art both from an economic and a technical point of view are discussed in 2.2: the start-up, which wants to have its core in this technology, aims to take its first steps in the electronic music market 2.2.1, also increasing the capacity of recommendation systems 2.2.2.
The evaluation of the chance of entering this market has followed a rigorous path: the process of continuous feedback research, so that future efforts can be directed in the best possible direction can be appreciated in 2.2.3. This can be in any case summarized in the words of a man far more intelligent than the one who is writing, Abraham Lincoln: "*Give me six hours to chop down a tree and I will spend the first four sharpening the axe*". In particular, a series of tests were carried out on potential customers so that they could tell their needs, in order to build the technology on those requests.
Following both tracks, economic and technical, also the 2.2.4 will cover the problems in either directions.
From the state of the art, the whole dissertation will focus almost entirely on engineering aspects. First a complete workflow, implemented by Spotify, is introduced in 2.2.5: this will explain a feature extraction and its classification.
Later a more classic literature study is presented in 2.2.5: here the two operations of feature extraction and classification, or identification, are treated separately.
The final part of the chapter,which is also the real technological core of the thesis, is presented in 2.3.3.
Starting from the needed preprocessing, 2.3.1, it moves to the generation of the right spectrum to reach the goal, 2.3.2. Given the time-frequency representation, a process of dimension reduction of the problem is used to simplify the feature extraction 2.3.3: here will be explained how a suitable time series will be divided in clusters, in order to extract the wanted information by means of a comparison of the results of non-linear processes applied to these subsets.

2.2 The reason and critical analysis

The field of work of the algorithm will be audio signal processing, and it will be applied to the products of a section of the music industry, while its results can be useful for its whole market. To understand the reasons behind the thesis, a bit of economics insight must be done. According to the Goldman Sachs' report [1]: *"the music industry is on the cusp of a new era of growth after nearly two decades of disruption. The rising popularity and sophistication of streaming platforms like Spotify and Pandora is ushering in a second digital music revolution – one that is creating value rather than destroying it like the piracy and unbundling that came before. [...] we lay out the converging trends that we expect to almost double global music revenues over the next 15 years to $104bn, spreading benefits across the ecosystem."*.

This means that even this field has been pushed up thanks to the new development, as explained at the very beginning of 1. Streaming is based on the massive spread of smartphones and the like, and has unlocked a previously unworn power: the one of data. This medium will be helpful in many ways e.g., the label's work of promoting and discovery of artists, that actually can cost as far as $10-$15 million globally per artist: a shift to paid stream will help to improve efficiency of this task.

The analysts behind the Morgan Stanley's report [2] believe that the most obvious area in which this enhancement could happen is, indeed, marketing, where the data collected by the streaming platforms on subscriber's music tastes could transform the ROI[1] of the label's expenditure. The data that the music labels could collect from streaming platforms would create a richer portrait of consumer interest and have a material impact on profitability and artist development.

After this first insight into the actual situation of the market, which shows what saved it, there is the need to move the eyes on the targeted segment and understand the secret weapon of the music industry's *"prince charming"*.

2.2.1 Electronic Music Market Share

According to a survey [4] made on 19,000 consumers, aged 16-64 and in 18 countries, Electronic Music is the third most popular genre in the World: this means that 1.5 Billion people typically listen to it. Born around the '50s of the past Century, even though the first ever electronic instrument is the Theremin[2] which was invented in 1919, it is now

[1]The Return On Investment is used in financial analysis and provides the investor with an indication of the efficiency of an investment by comparing profits related to capital invested. It therefore allows a comparison of alternative investment options based on efficiency. *ROI* can be estimated using a ratio between the net present value of benefits and the net present value of costs. [3].

$$ROI = \frac{Current\,Value\,of\,Investment - Cost\,of\,Investment}{Cost\,of\,Investment}$$

[2]It consists of two antennas positioned at square angle to each other, and between which there is a box containing the electronic part of the instrument. It is played changing the hands positions with respect to the perpendicular components: the horizontal one modulates the intensity, while the vertical

part (willingly or not) of everyone's everyday life: it is enough to think at advertising or soundtracks. It was approximately the 1980, when the very young DJ Nicky Siano, the first "resident" of the famous Studio 54, understood that two different tracks could be mixed together, in order to generate something new. The very first time Electronic Music became a worldwide phenomenon was during the "*second summer of love*", at the end of the 80s: after the death of Punk (sorry *Exploited*[3]), this genre took its place in the underground scenes, and gradually become more loved from the infamous *X-Generation*. In these 30 years in between its development has been capillary: in China and Taiwan 64% of the people that responded to a survey, made by NIELSEN-ENTERTAINMENT in 2017, stated to listen to Electronic Music, in Korea the amount reaches 74% [5]. A popular series of events, called *Ultra*, is hosted in 20 countries and the largest one[4] have been attended by more than 165k people. In 2017 *Sonar Festival*, which that year counted 124k patrons from 64 different countries, conducted an artistic-scientific experiment in collaboration with the Catalonia Institute of Space Studies and Messaging Extra-terrestrial Intelligence International. In this project 38 artists, both famous and not, sent music to GJ273b, a potentially habitable exoplanet[5] , 12.4 light years away from Earth[6]. Clearly, the induced effects of these huge events heavily influence the total value of the discussed industry, but not only: The New York Times inserted Torino, the city where this work has been developed, at the 31st place of the "*52 Places to Go in 2016*" also thanks to its electronic music festivals [6].
There is not even the necessity to be physically present at a venue to enjoy an artist's performance: in 2017 more than 100 millions viewers have seen live acts through the video streaming portal *BE-AT.TV*, and this is not the only platform providing this service (unfortunately much more than useful in these months).
The first to unlock the potential of showing a party was Blaise Bellvile (a name that should ring a few bells for fans of the genre[7]): in 2010 he founded the *Boiler Room*. Starting from a dutch taped webcam broadcasting from a dismissed boiler room, it is now a cornerstone: from 2012 to 2017 its audience raised from 10 millions to 303 millions [7], triplicating also the single average viewing time.
Others have followed its example:

- Cercle began to stream in April 2016, now has its own Festival, streams each 2 weeks and have reached 136 million YouTube views.

the pitch (so the frequency).

[3]"Punk's not dead" is an album from the UK group The Exploited.

[4]Ultra Music Festival Miami, in 2017.

[5]A Planet outside the solar system.

[6]To have more information, and status updates visit the dedicated page https://www.sonarcalling.com/en/ .

[7]The Bellville Three is also the group indicated as the inventor of Techno around 1988.

- Mixmag is a a magazine launched in 1983, from 2012 started "*The Lab*" series, streaming parties in its offices around the World. Now the print magazine represents just the 10% of the business.

The electronic music industry in 2018/2019 was valued around $7.2 billions. While the growth followed the one of total music industry, pushed by streaming services from 2013, there have been some losses due to the fall of clubs' worth. [8].
The link with streaming services has become even stronger when on February the 20^{th} the biggest website that sponsors and sells tickets for electronic music events, Resident Advisor, announced the beginning of its partnership with Spotify.

2.2.2 Spotify Recommendation model

Here a fast explanation of the ace up the streaming services' sleeve is proposed.
Every Monday, the over 100 million Spotify's users find a new customized playlist waiting for each one individually: this is the 30 songs custom mixtape *"Discover Weekly"*. According to the author of the article *"How does Spotify Know You So Well"* [9] this makes her feel *"seen"*, and she also brought to the attention other example of enthusiasts as her. The creation of playlists evolved following a path already seen several times in other fields: at the beginning of the 2000s' there was a manual curation, possible thanks to teams of music experts, that was not addressing the user taste but it was based upon the curator's one. After this *Pandora* came out with the simple (now) idea of tagging attributes, still manually, of songs.
In 2005, a research spin-off of the MIT Media-Lab came out: *The Echo Nest*. This visionary start-up (to give a timing reference: Youtube was launched the same year), acquired in 2014 by Spotify, used algorithms as Natural Language Processing [8] for texts, and others to obtain the Acoustic fingerprint [9] of each song, united with Machine Learning to extrapolate information and contextualize the user searches.
Although the awareness of the fact that the work is technical, and in fact now only aspects related to this viewpoint will be presented, it would be more honest to accompany the reading of the next lines thinking also about the transformation of the user from active to passive subject, and its possible impact on the habits.
To create the Discover Weekly playlist, the Swedish-based company mixes the three main used techniques to recommend:

- Collaborative Filtering, which examine both personal's behaviour and others'.

- Natural Language Processing.

[8] By "natural language" we mean a language that is used for everyday communication by humans; languages such as English, Hindi, or Portuguese. In contrast to artificial languages such as programming languages and mathematical notations, natural languages have evolved as they pass from generation to generation, and are hard to pin down with explicit rules. [...] At one extreme, it [NLP] could be as simple as counting word frequencies to compare different writing styles. At the other extreme, NLP involves "understanding" complete human utterances, at least to the extent of being able to give useful responses to them. [10]

[9] It can be seen as a digital summary, a fingerprint, deterministically generated from an audio signal, that can be used to identify an audio sample or quickly locate similar items in an audio database. [11]

- Audio models, to analyze raw audio tracks.

Collaborative filtering

Introduced for music by *Last.fm*, but which power was shown by *Netflix* with the users' star-based ratings, is a technique commonly used to build personalized recommendations. Spotify's data are not the user ratings, but they are more implicit: the stream counts, with some other information as if the users saved that specific track into one of their own playlists, or visited the artist's page (and Netflix too switched to this kind of approach, since it is seen as more reliable). To be simple, the idea behind this kind of algorithm is that if a lot of users listen to three tracks, then those track are probably similar. The matrix shown below is actually a $124Million \times 50Million$ one, where each row represents a user and each column one of the 50 Million songs in the app's database.

$$\begin{array}{cc} Users & \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix} \\ & Songs \end{array}$$

This has a binary entry method: 1 if the user has listened to the tune, 0 otherwise. It is possible to find also some matrices accounting for the number of streams, so integer positive numbers expressing how many times each user listened to a track instead of one and zeros. Starting from this, in order to see how similar two users selections are, in the European Startup, two algorithms are used [12]:

- minimize the *RMSE*[10] (root mean squared error) using a function of plays, context and recency as weight;
- maximize the *log-likelihood*[11] of the binary preference matrix, weighting observations as in the previous case.

[10]It is an a-dimensional quantity indicating the standard deviation of the residuals. The residual are a measure of the distance between a given point and the regression line.

$$RMSE_i = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{N}}$$

Where $\hat{x}_i$ indicates the mean value i.e., the regression value chosen for this kind of evaluation. [13]

[11]It is an estimator i.e., a function that starting from acquired data associates a value to the variable to be estimated. The estimate is the particular value returned by the estimator when applied to the particular observed data. The likelihood estimators, chosen a probability density function, aim to find the parameters that best approximate the observed data. To find the best parameters the usual chosen approach is the optimization process i.e., find the maxima or minima of a function in a given interval: the "log" part comes here in help, since working with the logarithm of the function provided by these estimator simplifies the differentiation step.

Natural Language Processing

These algorithms are often exploited with *Sentiment analysis*[12]: the idea is to crawl the Web looking for written texts to see what people is saying about songs, extrapolating also the language used by that specific fandom and which other artists or tracks are discussed in the same threads. The operation is now facilitated by the playlists created by the users: considering them as a document and the tunes inside as words, Spotify can also get information from its inside.
Following the same process presented in 2.2.2, a vector representation of songs is created and used to determine the similarities between those.

Raw Audio Models Analysis Introduction

The third, and most relevant, method is the one that works with the audio signals. It is the best way to introduce the topic about the state of the art of feature extraction for music: this is the reason it will be presented in details in 2.2.5.
By now, it is enough to know that every song in Spotify's database is processed, and some attributes are obtained from it. Then the results are merged together with the collaborative filtering ones in order to understand if some correlations are present.

2.2.3 Passion, but with a lean approach

The main reason behind this study is very simple: passion[13]. Without a deep knowledge of the "*modus-operandi*" of DJs the work would have been more a blind fly game.
As said in 1.4, this part of the thesis will be both personal and also a nice practical first steps guide for anyone interested in starting an own start-up.
The story begins the Halloween night of 2018, when a dear friend of mine, Matteo Giovannetti[14] told me about his desire to move his steps in the music-tech's industry.
At that time both were obsessive users of the services presented at the end of 2.2.1, and therefore a few ideas came to our minds.
The next step was to validate our vision, in the start-up market the mantra is to be customer oriented: they have always the last word in product development.
Following the approach established by Steve Blank[15], the lean startup, we needed to get

[12] "*Contextual mining of text which identifies and extracts subjective information in source material, and helping a business to understand the social sentiment of their brand, product or service while monitoring online conversations*"[14] .

[13] *Tutto è follia nel mondo* *Everything is madness in the World*
Ciò che non è piacer. *What is not to delight.*
Sings Violetta in Verdi's "La Traviata".

[14] Actually Co-foudner & COO of *"Clearbox AI"*, start up that won the 2019 edition of the StartCup Italia, and advisor for I3P, elected best public incubator in the Word.

[15] The person who launched the lean startup movement i.e., a methodology that wants to help in speeding up the process of product creation and business model development.

back some key metrics directly from the possible customers, but first we had to find them. In a classic, but smart, way of tackling the problem: the decision has been aiming for a niche market that strongly needed to see its problem solved, and then expand more and more the boundaries. Our target was the public of the various Mixmag, Cercle or Boiler Room streams: there we could have found our *early adopters*[16].
The golden rule, in this first contact with the possible clients, is to be honest: doing a very small Google Form, with the right amount of questions and a compilation time of 2 minutes has been hard but very important. The riskier situation, in this moment, is to generate some biased questions: knowing what we wanted to obtain, there was the possibility to lead the reader in a definite direction, and this is a huge mistake when going through such validation processes.
Thanks to social media power, it is nowadays possible to reach people from all around the Planet: Facebook groups were the perfect place where to find our picked audience. The only problem is that there is a kind of etiquette to follow: from asking the permission to join some of them to the needed level of engagement that could lead the other members of the group to help by answering to the questions.
Around the end of November we were ready to start our first research, initially the interviewed people where friends: we began with an Italian version of the above mentioned survey, not least because we still had to achieve the goals we needed to get enough feedback from socials and also to find any criticisms in the way we raised the queries. After a few time we were able to start also the International part of our primary research. In figure 2.1 it is possible to appreciate all the Countries (there is also Singapore, which is not quite visible) reached by the form.

The main differences between the Italian and International versions are, other than the obvious:

- in the Italian the responses arrived for 90% from people under 30 years old, and the 47% has declared to be a musician or a DJ;
- the International version experienced answers at 72% from people between 31 and 50 years of age, with only the 22% of under 30, while the 77% were musicians or DJs.

Given a total number of replies equal to 201, divided in 78 nationals and the rest worldwide, 186 people gave back meaningful results.
The Italians' one showed a huge amount of people, the 85%, interested in knowing how the DJ was performing a given modification (figure 2.2), while in the International's form this percentage was a little lower: the 70% (figure 2.3).

Regarding the time spent looking at video streams, we have been able to see what displayed in table 2.1, or in terms of absolute numbers in the bar charts screened in figure 2.4 and figure 2.5.

[16]The users who start using a product as soon as possible, helping in its development thanks to continuous feedback and strong engagement.

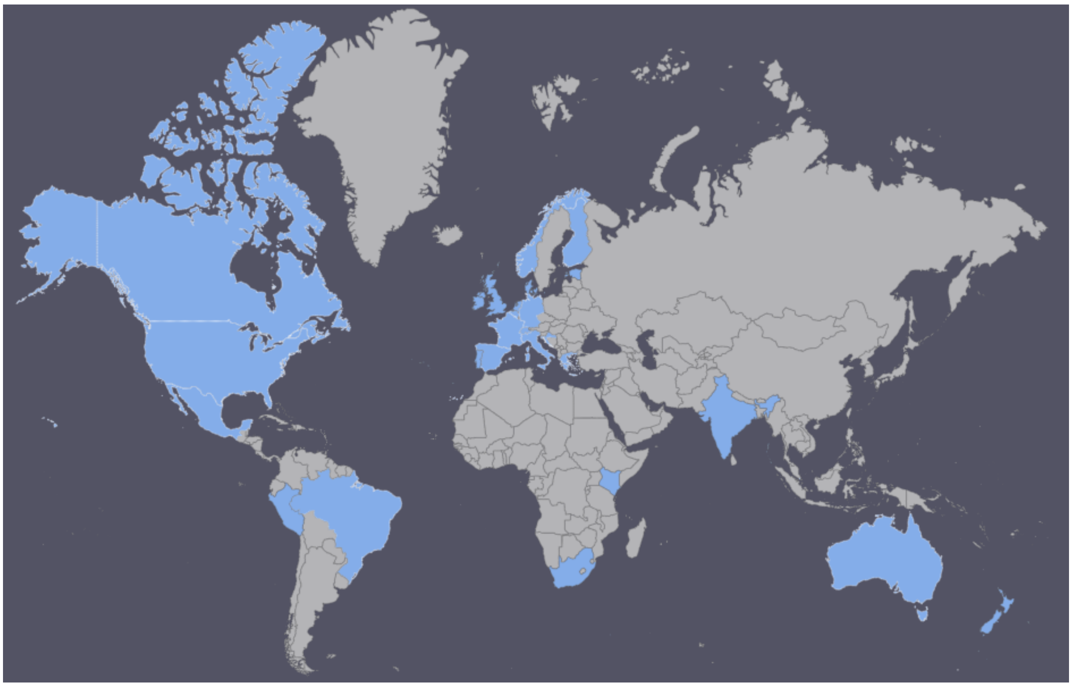


Figure 2.1: Map showing in Blue the Countries reached by the Survey.

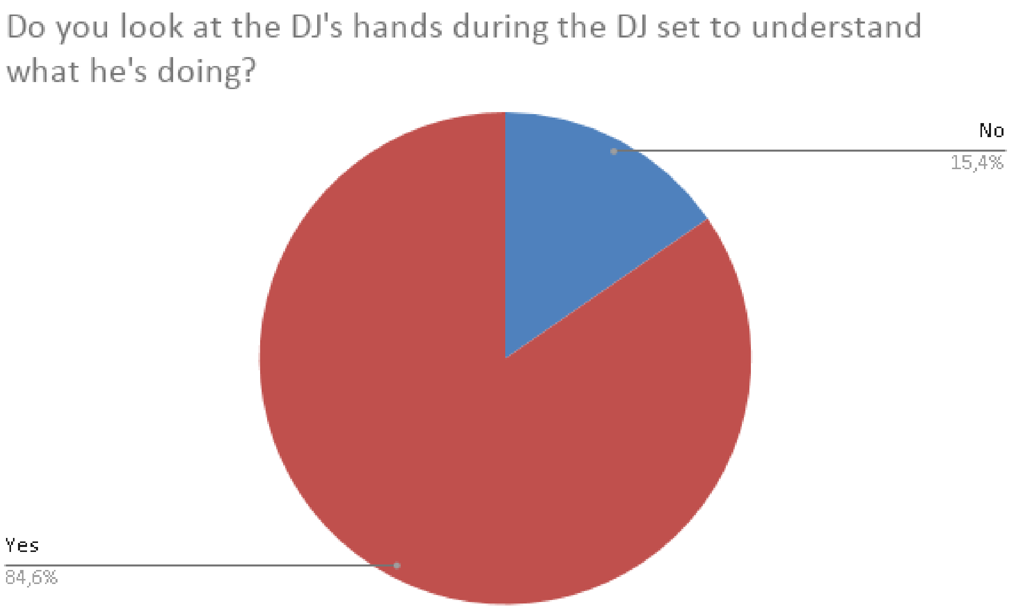


Figure 2.2: The pie chart of Italians' interest (in red) vs non interest (in blue) in looking the DJ's hands to understand what is doing. The proposed question is also shown.

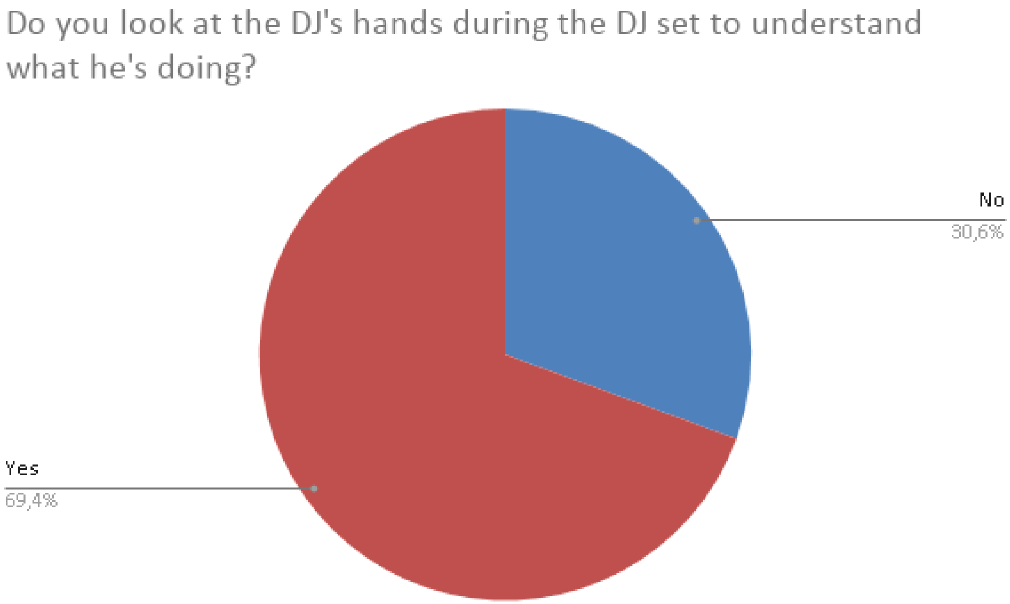


Figure 2.3: The pie chart of Internationals' interest (in red) vs non interest (in blue) in looking the DJ's hands to understand what is doing. The proposed question is also shown.

Hours/week of DJ sets watched	International Form	Italian Form
More than 3	40%	29%
From 1 to 3	34%	45%
Less than 1	15%	19%
Not watching them	11%	9%

Table 2.1: Table of the Form's second question Results

Coming from a technical formation, it is easy to decide that the collected data are enough to start building a product: **wrong**.
The form had also other questions, intended to understand a way to reach the early adopters as soon as possible: a sort of *hook* to make people start using the platform to avoid launching the real innovation without a community.
Once found this *hook*, we had to directly test also the eventual subscribers response on that: the second step of the customer discovery has been creating a *Landing Page*.
A *Landing Page* can be of various shapes, but it always aims to push the user ended on it to perform a specific task: if he or she is going to do what wanted it will result in a conversion.
The conversions are the number of positive results obtained, the conversion rate is usually calculated as any percentage value:

$$conversion\ rate = \frac{conversions}{total\ users\ on\ page} \times 100$$

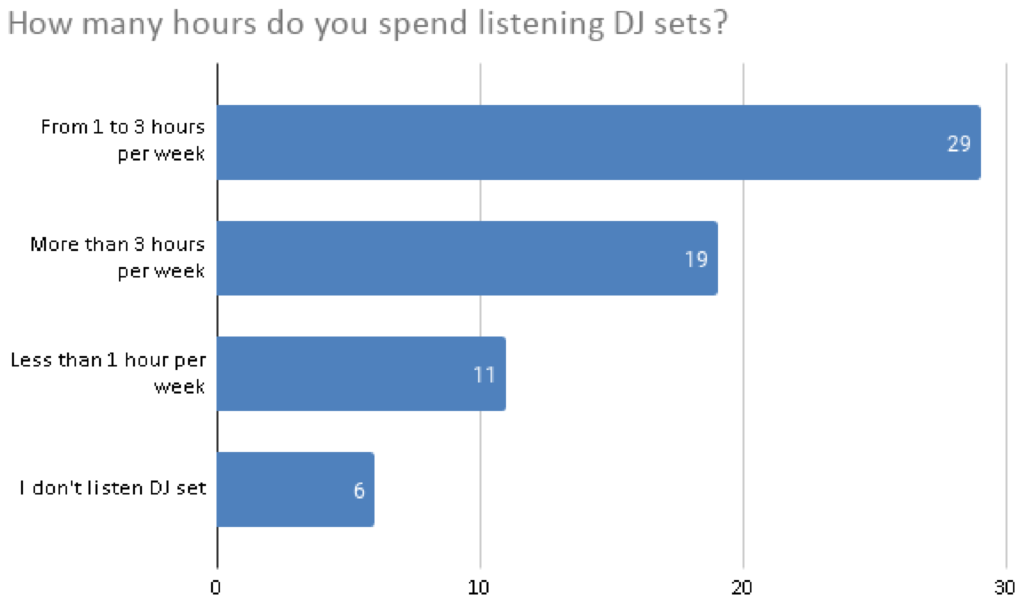


Figure 2.4: The bar chart of Italians' time spent looking at DJ sets streams per week, in terms of Hours. The proposed question is also shown.

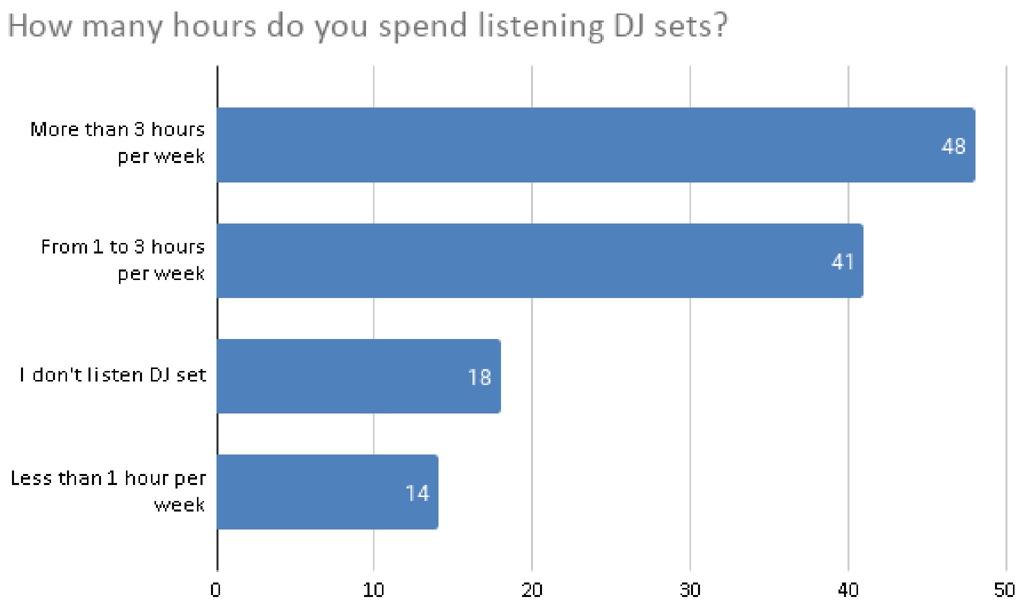


Figure 2.5: The bar chart of Internationals' time spent looking at DJ sets streams per week, in terms of Hours. The proposed question is also shown.

In the presented case, the goal was quite classic: a click on a "**I WANT IT**" button. It is possible to appreciate the page generated for this assignment at figure 2.6.
Also in this case, the results were absolutely positive: the average value of a good conversion rate is around 10%, while we achieved 19.27% during our test in February 2019.
It was time to start the building.

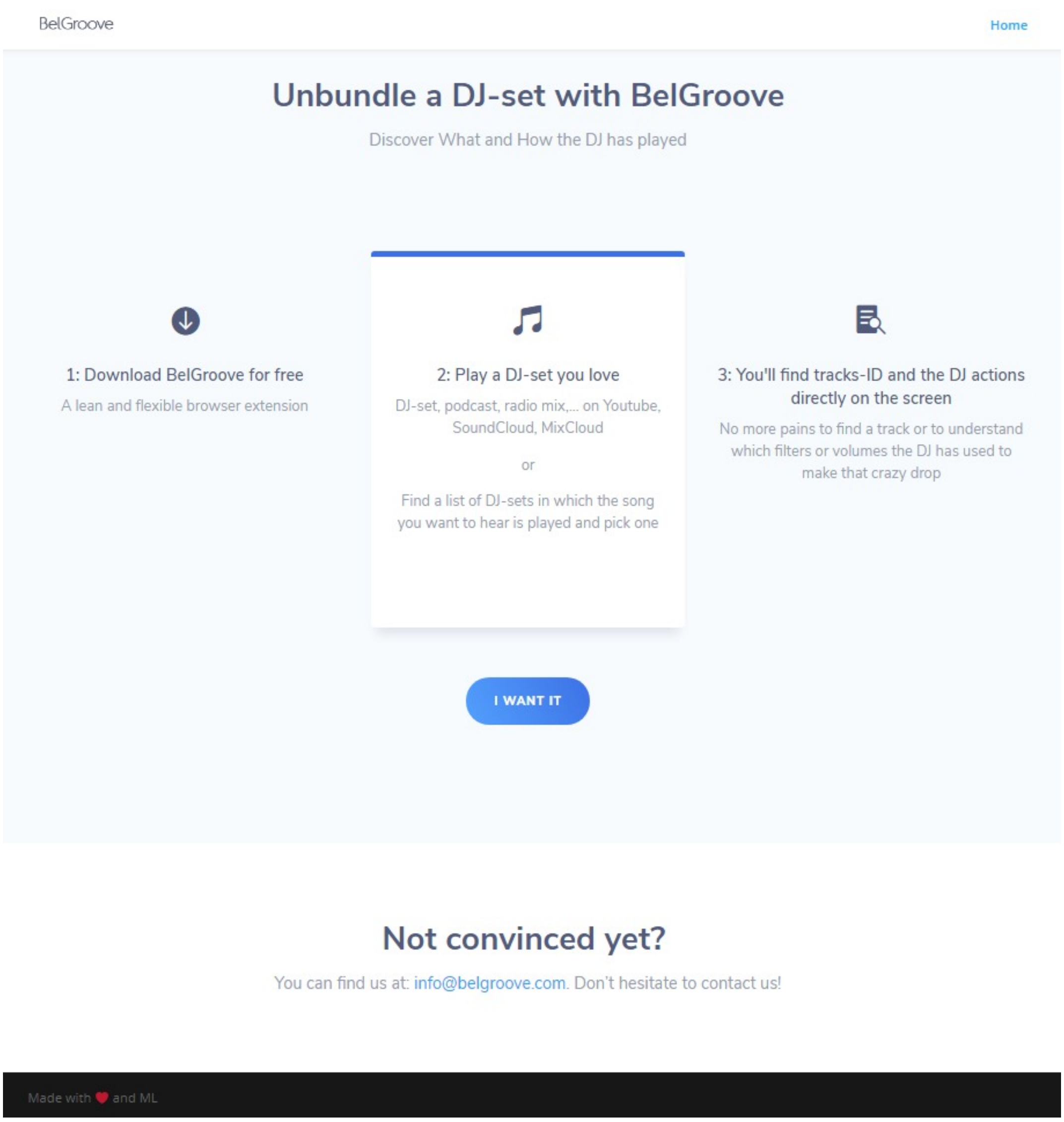


Figure 2.6: The discussed *Landing Page*. BelGroove was the chosen name for the Company.

2.2.4 CRITICAL ANALYSIS

Technical Problems

Even if the obtained model gives good results, as shown in the dedicated part 3.4.4, there is a major problem: it is grounded on some strong assumptions. It is widely known, how the process of reaching the representation of a real system is a constant trade-off between the ability of reproducing the effective behaviour as much as possible, and low computational effort (or the number of equations). The two main encountered real-problems were: a strong non-linearity and the Multiple Inputs.

Regarding the first one: the experiments performed to understand the the behaviour of individual filters have shown that, when working at given intervals, a huge distortion in the signal can take place. The problem has been individuated in something that is normally not perceived by humans, and is discussed in depth in its own subparagraph 3.2.4 . Here it is not said that no one can hear this, indeed it is even a kind of effect exploited by some artists thanks to specific instruments, but the one generated by the consoles are not strong enough to be normally appreciated. This particular aspect led to the decision not to directly tackle the problem in this thesis: the model will only work in a more limited range than its real-world counterpart. It has to be stressed out how this is more an academic issue rather than an application one: its incidence corresponds to an area where the overwhelming majority of DJs decide to perform an unique type of modification, so it can be seen as a secondary matter in terms of future development. This will be discussed with more accuracy in the Conclusions, at chapter 5.

Regarding the Multiple Inputs, the issue becomes harder to deceive. What was needed, in order to complete the work, was an hypothesis not always suitable for a real application: a sort of logical pipe for the modification. The assumption is like thinking that there is a switch between channels: if the artist is using one to perform the modification, it is not possible to modify in the meantime a track loaded on another one. Even if by now this can seem strong, it is to be kept in consideration that the system is ruled by humans, so even in the presence of an highly skilled artists, working at modern calculators' speed will help in improving a solution based on this idea. Again the future developments will be discussed in the relative section, in chapter 5.

What is also to stress is that the back-end code is written with Matlab, so a translation in another, faster, language is needed to be able to generate the virtual machines able to make everything run on behalf of a call from an user.

As Michael Seibel, CEO of Y Combinator[17], said in his lesson "*How to Plan an MVP*[18]" of the Y Combinator Startup School 2019 (the content of which is free and on YouTube): "*If you can walk away from one thing from this presentation is launch something bad, quickly*".

[17] One of the biggest startup accelerator, short programs to accelerate growth, in the World.

[18] *MVP means Minimum Viable Product, and is defined as the minimum working product.*

It is possible to say that at least the "bad" point of the product to be launched has been fulfilled.

"Puppy Shooting": the Business Problems

Since the previous paragraph has been completely focused in an economic insight, and full of positiveness, it wouldn't be fair to leave the reader without the other side of the medal. After the outstanding results obtained while testing the B2C[19] branch of the enterprise, it was time to start moving the sight in direction of someone able to begin showing some interest in the data we eventually could obtain from the first day of launch: the record companies. For this B2B[20] test, the decision was to send some presentation e-mails, with a new form attached, tailor-made to test the new group's needs. Unfortunately the excitement found a moment of sudden stop, seen the huge difficulties in getting in touch with big groups without a working product or a famous name behind.

2.2.5 Literature study, what's new, state-of-the-art

The common way to tackle the family of problems, to which the one discussed in the thesis belongs, is to analyze the signal by selecting the right domain of analysis, and then decide which feature extract from the result of this procedure: once the information has been extrapolated there is a range of approaches to perform its classification or recognition.
There are mainly three different types of possibilities to execute the process of a signal:

- Time domain, that lacks the frequency description of signal.
- Frequency domain, as the one made by means of Fourier Transforms, which can't depict the time variation of the spectral content.
- Time-Frequency domain, where the time variable is introduced in Fourier based analysis.

Since the goal of the presented work, the choice has fallen in the Time-Frequency domain, which algorithms in fact will be the only ones to be discussed. Regarding the extraction an example of implementation and a theoretical approach are going to be introduced.
The first subparagraph, 2.2.5, is going to present the Sporify's idea to perform this task, from the start to the end, while the second part is a summary of the most pertinent literature.

Raw Audio Models, state of the art

This method does not count just as a 33% of the app's extrapolation package: it is, as said in 2.2.2, the most important one. The reason is really simple: it does not discriminates on the number of listeners. The first two ways of recommendation are based upon popularity,

[19] Business to Consumer: the economical relationship with the consumers.

[20] Business to Business: the economical relationship with other enterprises.

obviously they work great but can't be suitable to new songs, especially if from unknowns. One could think that the "mainstream" artist are there for a reason, and here this won't be discussed, but the first word of Spotify's trump card is *"Discover"*.
The analysis is performed by means of *Convutional Neural Networks*, adapting these algorithms, especially generated for pictures, to work with audio signals instead of pixels. To be more clear what is presented in an article [15] written by S. Dieleman, a Google DeepMind [21] researcher at his time at Spotify (where he worked on content-based music recommendation) will be used as guide for this topic. To explain what a *Convutional Neural Networks* is, a little digression is needed. As explained in [16]: a Neural Network, in a classification problem for a vector(which will be explained in 2.3.3), instead of using immediately some clustering equations would first compute some transformations in order to reduce the dimensions of the input so, obtaining a given number of intermediate vectors thanks to some nonlinear operations, it will simplify the problem until the desired result is easily achievable. Every time a nonlinear operation is computed, there is an *hidden layer*, made of neurons all connected with the ones of the previous *layer* and independent with all the others in the same *layer*, as shown in figure 2.7. Each neuron has the capacity to like or dislike some regions of its input space based on some nonlinear activation functions (which are different and won't be discussed), that are then linearized in order to turn neurons into linear classifier: usually the aim is to obtain a binary logic as *Yes/No* or *In/Out*.
The *Convolutional* architectures are similar to the one just explained, what changes is the initial hypothesis: they know to be working with images. Instead of being 2D, as the one in picture, these architectures expand themselves in terms of volume: they are able to reduce the input image into a single vector of class scores, arranged along the depth dimension. If all the neurons of the same depth slice have all the same parameters, the reduction step of each layer is seen just as a convolution[22] between 3-D input and the neurons' parameters.

In Spotify's implementation network the input is a spectrogram[23] with 599 frames, along the time axis, and a 128 frequencies bin: the mel-spectrogram, which is a kind of

[21] Using their own words: "*a team of scientists, engineers, machine learning experts and more, working together to advance the state of the art in artificial intelligence. We use our technologies for widespread public benefit and scientific discovery, and collaborate with others on critical challenges, ensuring safety and ethics are the highest priority.*"

[22] The convolution is a mathematical process that allows to obtain a signal as a consequence of the combination of two others.

[23] A visual representation of a signal in terms of frequencies varying in time.

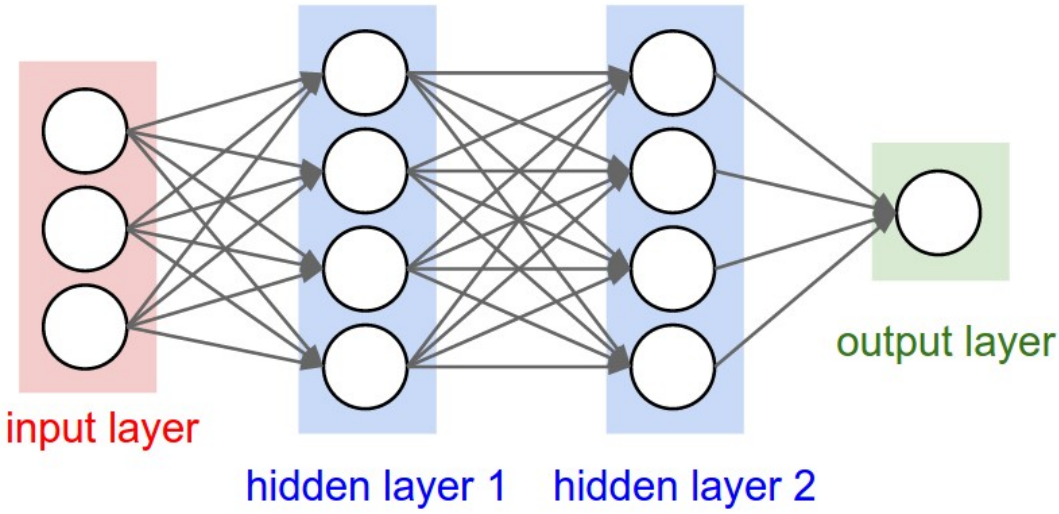


Figure 2.7: A 3-layers neural network with three inputs, two hidden layers of 4 neurons each and one output layer. Notice that there are connections between neurons across layers, but not within a layer. Image taken from the Course: Convolutional Neural Network for Visual Recognition[16].

time-frequency representation[24]. This is obtained by calculating 599 *Fast Fourier Transform*[25], each of which constitutes a frame. These successive frames are then concatenated into a matrix to form the spectrogram.
Finally, the frequency axis is changed from a linear scale to a mel scale[26] to reduce the

[24]Often it is graphically represented as a plane with time and frequency on the axis as in 2.14, 2.15 and 2.16. It is the representation of a signal both from a time and a frequency point of view.

[25]An efficient algorithm for computing the DFT[17]. The DFT, or Discrete Fourier Transform, is based on the idea that any finite sequence $x(n)$, with N number of samples, can be represented as a polynomial $X(k)$: the numbers in this sequence are the polynomial's coefficients.

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi n \frac{k}{N}}, \quad k = 0,1,2 \dots N-1.$$

With $f_k = \frac{k}{N}$ N equispaced frequencies. [18]

[26]A subjective scale for the measurement of pitch. This scale differs from both the musical scale and the frequency scale, neither of which is subjective: this is a numerical scale proportional to the perceived magnitude of subjective pitch [19]. To convert from Hz to mel [20]:

$$m = 2529 \log_{10}(1 + \frac{f}{700}).$$

dimensionality, and the magnitudes are tranformed logarithmically to find their *dB values* [27]. The first three showed convolution happens only in time-dimension, avoiding the frequency's one: differently from the images' case, here the two axis have a different meaning (space against time-vs-frequency). The convolutional layers are portrayed with red rectangles in picture 2.8.

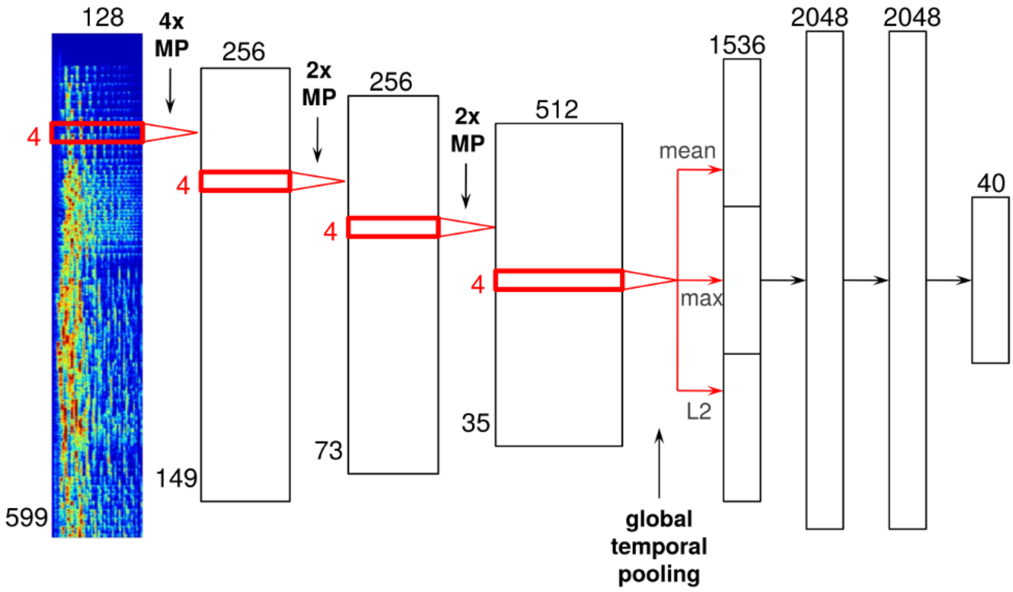


Figure 2.8: The presented convolutional neural network. The time axis, which convoluted over, is vertical.

In order to decrease the size of the representation, reducing the amount of computation needed, some *pooling layers* are created. As explained before, these layers have neurons that perform a classification based on a nonlinear function, that in the presented case will be the maximum's one.
From the *input layer* i.e., the *time-frequency* representation of the audio signal, a first *Max-Pooling* operation is fulfilled: a filter spans all the input by means of 4×4 squares, searching for the maximum of these small subsets. Usually the first filters pick up the

[27]The decibel is a logarithmic unit defining the magnitude of a physical quantity (usually power) relative to a reference level [21]
For power usually is used:

$$P_{dB} = 10 \log \frac{P_1}{P_2}$$

For amplitudes, the one used in the presented case:

$$A_{dB} = 20 \log \frac{A_1}{A_2}$$

harmonic content, as they tend to detect the human voice. The next layer, called *convolutional layer*, will be reduced to one-fourth of the input's size (with respect to the frequency axis) thanks to this operation. The successive two filters will have a so-called *pool size* i.e., the area side or the reducing factor, of 2 and operate still with the same methodology.
After the last *convolutional layer*, an additional *global temporal pooling layer* has been added, which is pooling across the whole time layer, with three different nonlinear functions(the mean, the maximum, and L2 i.e., the Euclidean distance norm): this is done since the actual time position of a certain feature is not relevant in this kind of analysis. Now the most pertinent information about a song are obtained.
The network is then trained to minimize the mean squared error [28] between the prediction obtained from raw audio and the results of the collaborative filtering method.

Others feature extraction

Spotify is just an example of a real-implementation for music related feature extraction and classification, so now a look at the whole picture of the actual state-of-the-art will be developed. The work made by E. Sejdić, I. Djurović and J.Jiang [22] provides a complete overview, therefore it will be used as a support. The basic goal of a Time-Frequency analysis, as the one presented in the thesis, is to determine the energy concentrations along the frequency axis, at a given time instant. The possible algorithms to perform this task can be classified according to their approach.
For the **first** class of time-frequency representation, a signal $x(t)$ can be expressed by means of the equation:

$$TF_x(t,w) = \int_{-\infty}^{+\infty} x(\tau)\phi^*_{t,w} d\tau \ = < x, \phi_{t,w} >$$

where $\phi_{t,w}$ is the basis function and * stands for complex conjugate[29]. In other words, the signal is being decomposed over waveforms that reveal signal's proprieties [23]. This group includes Fourier transforms and wavelets: the difference stays in the deployed basis function of finite energy $\phi_{t,w} \in L^2(\mathbb{R})$. Unlike the Fourier transform, the wavelet one introduces an analysis with sharper time resolution for high frequency components than for the low frequency ones and treats frequency in a logarithmic way [24].

Another category is represented by the **Cohen's classes**. These can produce better results, but a problem is given by interference which can lead to ambiguous representations.

[28] The Mean Squared Error is the most commonly-used measure to understand the quality of Fit of a regressor.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2,$$

where $\hat{f}(x_i)$ is the prediction that the function $\hat{f}$ gives for the ith observation [13], in this approach it is the mean value of the vector y.

[29] Given a complex number $c = a + jb$, its complex conjugate is $c* = a$ - jb. So two complex conjugates numbers have same real part but opposite imaginary.

Starting with the ambiguity function, i.e., the two-dimensional autocorrelation of waveform in time and frequency [25], defined by Cohen himself as [26]:

$$A(\theta, \tau) = \int_{-\infty}^{+\infty} x(u + \frac{\tau}{2}) x * (u - \frac{\tau}{2}) e^{j\theta u} du.$$

It can be shown that the auto-terms are always connected to the origin (0,0) of the time-frequency plane where the autocorrelation lies, and the cross-terms, which are responsible for the problem, tend to spread to other places. This observation suggests the application of some weighting functions (or kernels) $\phi(\theta, \tau)$ to suppress $A(\theta, \tau)$ for $\theta, \tau >> 0$, i.e., low-pass filters in the ambiguity domain, reducing the cross-terms interference [27]. Rewriting the Cohen's class in the time-frequency domain as:

$$TFD_x(t, w) = \frac{1}{4\pi^2} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} A(\theta, \tau) \phi(\theta, \tau) e^{-j\tau t} e^{-j\tau w} d\tau d\theta.$$

Many different kernels have been designed, leading to different distributions: usually these present one or more tunable parameters, in order to obtain better representations and be able to optimize the distributions with respect to the application.

The **Instantaneous Power Spectrum** is another solution for Time-Varying Spectral Estimation. In [28], it is defined as the combination of the derivatives of the two running energy spectra ρ^+ and ρ^-:

$$IPS(t, f) = \frac{1}{2}[\rho^-(t, f) \ + \ \rho^+(t, f)].$$

From the energy spectrum definition:

$$S_t^+(f) = \int_{-\infty}^{t} s(\tau) e^{-j2\pi f \tau} d\tau \quad and \quad S_t^-(f) = \int_{t}^{-\infty} s(\tau) e^{-j2\pi f \tau} d\tau$$

it can be shown that:

$$IPS(t, f) = \mathrm{Re}[s^*(t)(S_t^+(f) \ + \ S_t^-(f)) e^{j2\pi f t}] \ = \ \mathrm{Re}[s^* S(f) e^{j2\pi f t}]$$

where *S(f)* is the Fourier transform of signal *s(t)* and * denotes complex conjugation. This solution has demonstrated to be less sensitive to cross-terms and inability to work at the endpoint of the analysis segment [29].

The last Time-Frequency representation algorithm presented is based on the rotation of the TF plane i.e., the **Radon Transformation**. It is another idea to avoid the problems of cross-terms correlation and withe noise: in particular this method is used to be able to emphasize more slowly-varying components contributions [30]. This feature lead to its use in multi-component signals analysis, defining the Randon operator [31]:

$$\mathbb{R}[f(x, y)] = \int f(r\cos(\phi) - s\sin(\phi), r\sin(\phi) + s\cos(\phi)) ds$$

with r and s corresponding to the time and frequency axis rotated counter-clockwise of an angle ϕ. Applying the $\mathbb{R}$ operator to a time-frequency distribution, a new integration path, that can be modulated regarding the type of signal to be analyzed, is given.

After the selected way to perform the time-frequency representation, there is the necessity to analyze the obtained spectrum in order to be able to extract the needed information. The CNN used by Spotify, and explained in 2.2.5, is not the only possible path. In [32] an approach based on a **Markov Chain Monte Carlo**, which uses the energetic distribution concentrated around curves in the time-frequency representation of the signal is provided. The MCMC algorithm is a solution to both the problem of dimensions for Monte Carlo sampling and its assumption that each state is independent from the others: the Markov Chain approach assumes that the system must evolve, between the current state and its temporally subsequent, randomly based exclusively on the present state and not on the whole history of evolution [33]. The Markov Chain can be represented by means of a weighted Graph Algorithm where the vertices are the states and each edge represents the possible transition, with relative probability as weight [34].
The assumption is that the desired characterization of a signal can be achieved just by estimating the local maxima of the square modulus of its time-frequency representation (the values of which are the energy of the signal at that frequency in that instant). In other words, it is searching for the ridges of a surface $M(b, \psi) \in D$ with b time instant, ψ the frequency variable (this depends on the chosen way to obtain the representation) and D domain. Since again, the interest is to understand the energy distribution, a ridge set R is defined as the set of local maxima in ψ of the M function, assumed smooth enough to think at a slow varying function in its domain.
The idea of the algorithm, called *Crazy Climber*, is to generate a large number of particles (i.e., the climbers) randomly seeded on the domain D at the initial step. Each climber evolves according to a Markov chain, and these are encouraged to "climb the hills" by means of some penalization functions. Since the implementation will evolve on a discrete time-frequency plane, assuming a finite time interval with B elements $[b_0, b_1, \ldots, b_{B-1}]$, and for the frequency variable a finite set of K values $[\psi_0, \psi_1, \ldots, \psi_{K-1}]$ the analysis is reduced to a $B \times K$ positive semi-definite matrix. With N climbers, positioned casually on the sampled grid, and able to move left or right on the B vector and up and down along the K direction. The move on the vertical direction takes place just if the delta:

$$\Delta M = M(j', k') - M(j', k)$$

is positive, so if it is moving in the direction of a local maximum. The vertical move is made with probability $P_v = \exp\left[\Delta M/T(t)\right]$.
At each time t, two occupation measures are considered (one weighted and one not) in order to find the local maxima.

What's new

All of the above methods are widely used for feature extraction applications, but a common aspect is the search for indications just about instantaneous frequencies. It is not an issue characterizing only the audio signal field e.g., the frequency information can be used to detect mechanical failures [35]. What is missing is the mining on the other axis: the time one.
With the help of some already existing algorithms, as the one for the generation of the

Time-Frequency signal's representation, a way to extrapolate knowledge both about the evaluation of the change in the signal's energy at given frequency and when occurs has been developed and implemented.
What needs to be stressed out is that the proposed goal imposes this bidimensional extraction, while Spotify just analyze some random intervals in a song and the algorithms to detect failures are there to communicate the problem as soon as it occurs.
Another aspect to keep in consideration, is that the usual scanning regarding music are strongly focused on the Human Voice, while the presented algorithm does not have this bias: as explained, it is not working on the song itself but on the changes made to the latter.

2.3 Signal Analysis

In this segment the core of the signal process will be explained. Since this algorithm has been repeated, with a few small variations, in each analysis (from the one for making the filter mask to the code for extracting the information searched for the GUI), a separate section is the best way to avoid unnecessary repetition and give it the space it deserves.
Later in the work, the small edits will be explained situation for situation.

2.3.1 Preprocessing

A necessary operation to perform when handling a signal is Preprocessing. The worst case is when there is a great presence of noise, which is defined in the Modern Dictionary of Electronics[36], as *any unwanted electrical disturbance or spurious signal that modifies the transmitting, indicating, or recording of desired data.* In the presented case, there won't be the necessity to deal with noise cancellation: since the recording of the signal is done directly on the Mixer Output, the only noises are the purely electronic ones, which are so small that lead to very high (tending to infinity) SNRs[30].
In the paragraph will be presented the algorithms necessary to make the signal ready to be analyzed.

Acquisition and Normalization

Signal acquisition is the first and necessary step. It has been done using the Matlab function *"audioread"*: this takes as input a character vector i.e., the name of the file in the folder, and as output both an m-by-n matrix, whit m number of audio samples read and n number of audio channels in the file, and the sampling frequency in hertz.
The sampling theorem specifies the minimum-sampling rate at which a continuous-time signal needs to be uniformly sampled so that the original signal can be completely recovered or reconstructed by these samples alone. This is usually referred to as Shannon's sampling theorem in the literature.

[30]The Signal to Noise Ratio, or SNR, is a dimensionless values defined as the ratio between Signal Power and Noise Power: $\frac{P_s}{P_N}$. It is used as quality factor for transmission systems and, more widely, in information theory.

This principle affirms that: if a continuous time signal contains no frequency components higher than W Hz, then it can be completely determined by uniform samples taken at a rate f_s samples per second where

$$f_s \geq 2 \times W$$

or, in term of the sampling period T:

$$T \leq \frac{1}{2 \times W}$$

The minimum sampling rate allowed by the sampling theorem ($f_s = 2W$) is called the *Nyquist rate*[37].
Having in mind that : *the human ear has receptors that can detect sound frequencies ranging from 16 vibrations to 20.000 vibrations per second*[38], it is possible to assume $W = 20kHz$.
The sampling frequency (f_s) for the recordings has been chosen equal to 44.100 Hz: it is clearly possible to work at higher rates, but the decision was to stay (as far as possible) with the worst case scenario and to always take into consideration the attempt to make the process as fast as possible.
Since the recording is performed directly on the mixer's output, the number of audio channels in the file i.e., n, is equal to 2: this is because *RCA stereo* or *MIDI* cables have to be linked to the consoles' output and, to be simple, these are couples of wires, that's why the recorded signals holds two channels. Given that the goal is to run the algorithm in real-time, the taken path was to half the number of data to work with: the double-channel is useful to have a better auditory experience for the users[31], but it does not add a substantial improvement for the performed analysis.
The n_1 and n_2 signals are merged together subtracting each of the two to their sum divided for the length of the corresponding value on the row i.e., the average. Then the rows of the resulting matrix are summed.
The last performed operation is the so called peak normalization[32], so adjust every value with respect to the highest signal level present in the record. This last part is very important in order to avoid some problems regarding the recording power: it is not possible to assume that everyone would use the same settings in terms of volumes, and moreover the differences are also evident with the same song downloaded from two different platforms. The solution is then to work just with normalized values, making each audio signal as loud as possible (it is always reached the limit of 0 dBFS) and closer (in terms of volume) the two audio files to be compared.
Both the modified track and its reference, stored in the database, are pre-processed in this way.

[31] The double-channel means that the record is done in "Stereo" mode: it has a Left and a Right channel, which are used to address the signal to the desired amplifiers in sound systems.

[32] This method was preferred to the RMS normalization, since this last one presents a few limitations since it can be needed to turn down some volumes to let them stay under the digital audio limit of 0 dB relative to Full Scale.

Alignment

The alignment of two time series can be simplified with a two-steps algorithm:

1. Comparison and feature extrapolation via a measurement function that provides information on the two data streams.

2. Delay or Shift of one of the two signals.

The compared signals are the one that is gone through the mixer i.e., the one which is modified by the system, and the unmodified one or reference.
Before finding the best solution, different approaches have been tried.

The Dynamic Time Warping (**DTW**) algorithm's objective consists in identifying an optimal alignment between sequences by warping the time axis iteratively.
In order to align two time series $X := (x_1, x_2, \ldots, x_N)$ and $Y := (y_1, y_2, \ldots, y_M)$ of length N and M respectively, an N-by-M *cost matrix* $C \in \mathbb{R}^{N \times M}$ is computed. Each (n^{th}, m^{th}) element of the matrix C, corresponds to the distance between each pair of elements of the sequences X and Y. The employed distance by the Matlab command "*dtw*" is the Euclidean one

$$[C]_{n,m} = \sqrt{(x_n - y_m)^2}$$

The goal of DTW is to find the optimal warping alignment path between X and Y, having minimum overall cost.
The optimal warping path W^*, a set of matrix elements that define the optimal relationship between X and Y, must satisfy the following conditions:

- Boundary condition, which enforces that the first and last elements of X and Y are aligned to each other (to cover the entire signal).

- Monotonicity condition, that forces the points in the warping path to be monotonically spaced in time (the path can't turn on itself).

- Step size condition, to avoid omissions in elements and replications in the alignment of X and Y (no skip in samples and no repetition of features).

The path can be computed backward or forward.[39]
This elegant and complicated solution was discarded due to the too high computational costs that requires.

The Edit distance on real signals (**EDR**) Matlab command still works thorough Euclidean distance (as default). Given a real value ϵ, specified in the command inputs, the mth sample of X and nth sample of Y are declared to *match* if

$$d_{mn}(X, Y) = \sqrt{(x_m - y_n)^2} < \epsilon$$

If not matching, the algorithm try to make them match in three different ways, which are not only removing samples from a signal to shift it, but also substituting m^{th} with n^{th} value in the first signal (or vice-versa).[33]

[33]These and eventual further information can be found at the Mathworks site for the edr command.

This path needed not only a too long time to give results, but introduced also the risk of a modification in the signals.

The adopted solution was based on a simpler and most effective idea: the cross correlation. To be more specific it is a **normalized running correlation** which delays one or the other signal i.e., the modified and reference tracks, to make the two coincide as much as possible. The first step of the algorithm is to evaluate the cross correlation between two signals, with the *xcorr* Matlab command. When performing this operation between two time-series, a so called *correlation signal* $\hat{R}$ is produced. By default *xcorr*, when dealing with two signals of different length[34], computes raw correlation without normalization:

$$\hat{R}_{xy}(m) = \sum_{n=0}^{N-m-1} x_{n+m} y_n^*, \quad m \geq 0.$$

The function's output is the vector:

$$c(m) = \hat{R}_{xy}(m - N), \quad m = 1,2,\ldots,2N-1$$

and where y^* is the complex conjugate of the y signal. To work in a more effective way, the following step is the one of normalizing the c vector with respect to its maximum value. The *xcorr* command also provides the lag values. Searching the lag corresponding to the maximum value of the normalized cross-correlation vector, the indication of the delay between the two signals can be found. In figure 2.9 it is possible to see the c normalized cross correlation vector , with the relative lags. While in figure 2.10 the same image is displayed, only enlarged around the maximum value of the c vector.

Based on the chosen lag value sign, which is the N such that x and y are aligned, the algorithm decides whether to delay one or the other signal:

$$\begin{cases} x, & if\, N \geq 0 \\ y, & if\, N < 0 \end{cases}$$

[34] Due to the fact that a track doesn't have to be played from beginning to end, but can also be used just for some seconds in between its boundaries.

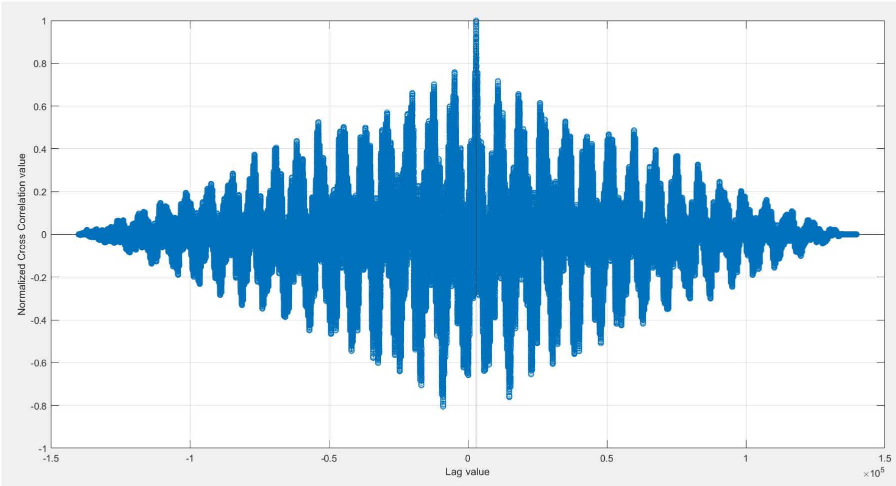


Figure 2.9: The complete c vector is shown, a horizontal line highlights the chosen value couples

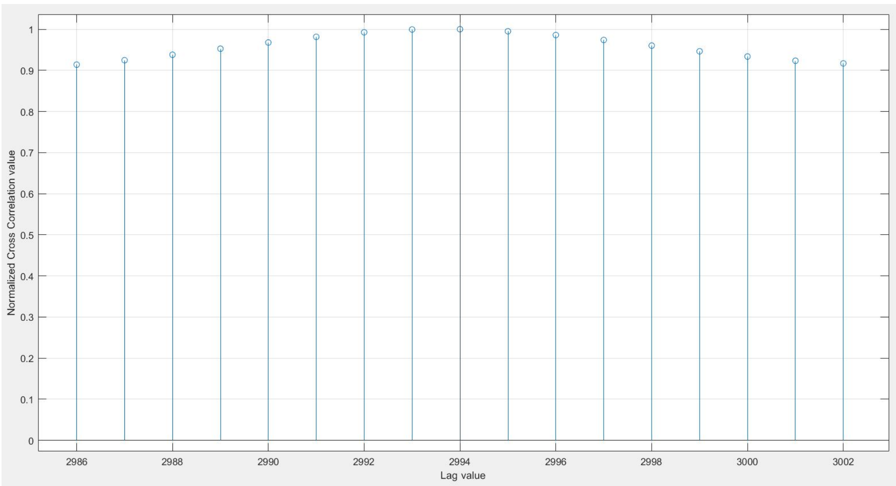


Figure 2.10: The zoom of c vector around its maximum is shown, a horizontal line highlights the chosen value couples

As shown in this presented case, the lag is at the 2994^{th} sample: a delay of the x time-series of that value is needed to align it to y time-series. In figure 2.11 the x and y vectors are appreciable superimposed before the alignment, while in the following figure 2.12 one can esteem the result of the function.

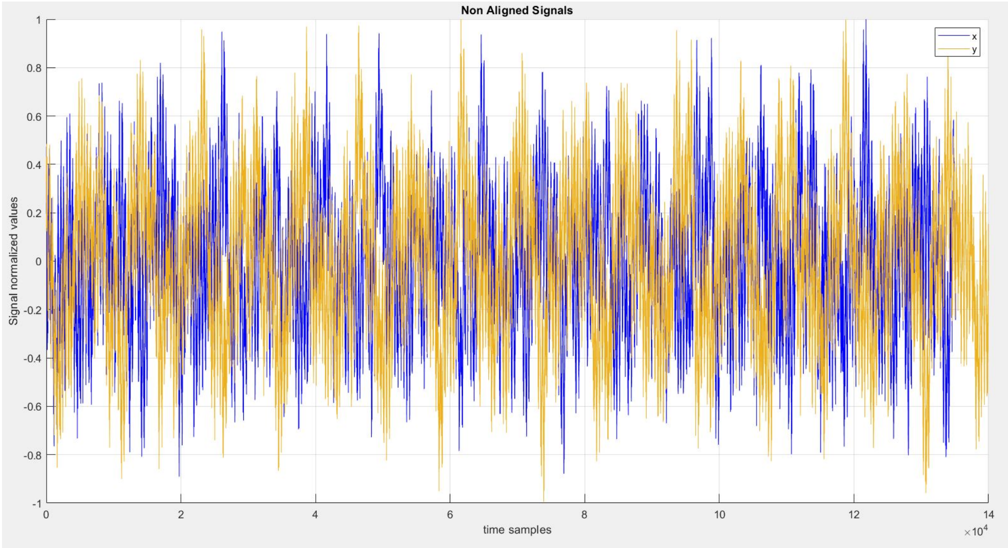


Figure 2.11: The non aligned signals.

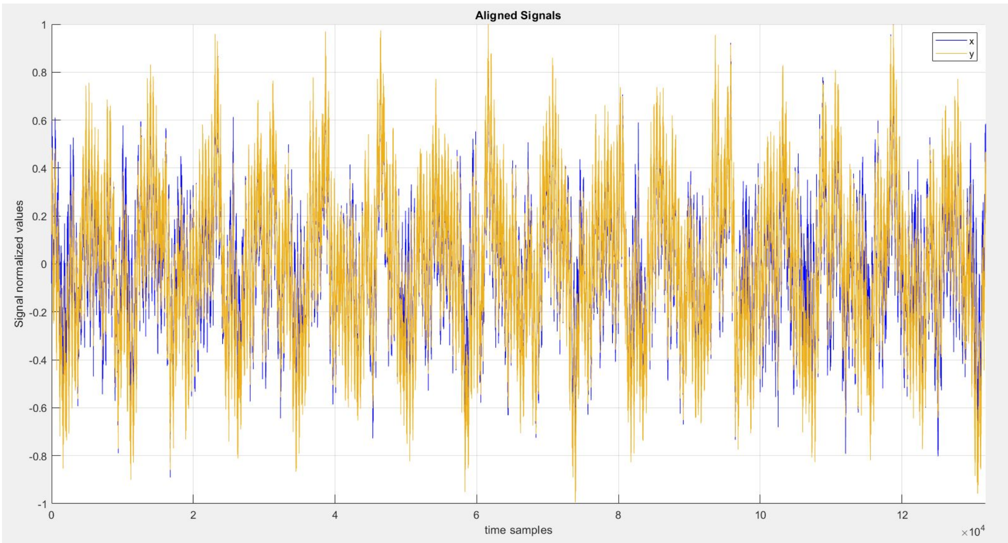


Figure 2.12: The aligned signals. The evident differences in heights are due to modifications in the x (blue) signal.

In the table 2.2 are presented the results of a random test on the function: the error is evaluated by looking at the position (in terms of sample number) of the same peak in the two signals, and then calculating the simple mathematical difference between them. Regarding the percentage value it is simply obtained doing:

$$percentage\ error = \frac{|pk_{s1} - pk_{s2}|}{signal\ length} * 100$$

Trial	Absolute Error	Percentage Error
1	4	0.003%
2	1	0.0008%
3	2	0.0015%
4	1	0.0009%
5	1	0.0008%
6	1	0.0007%
7	1	0.0008%
8	4	0.0031%
9	1	0.0008%
10	1	0.0007%

Table 2.2: Table to test on the Signal Shift function

It is clear that this solution matches very good results with the necessary speed of execution, since the average time needed to perform it is around 0.13 seconds.

2.3.2 Time Frequency Representation: Gammatone Filterbank

To obtain the Time-Frequency representation of the signal, this is decomposed through a bank of *Gammatone filters*[35] equally spaced on the ERB scale[36].

The motivation that led to this choice, is that the result must always be customer oriented: a feature used to model the human auditory system seemed to be the perfect way to show to people what they already hear, without being able to quantify it. As stated in [40]: since frequency responses along the basilar membrane vary exponentially, logarithmic frequency representations might prove most efficient. One such representation could be derived using a *Gammatone Filterbank*.

Proceeding in chronological order with respect to the definition given above: first the definition of a *Gammatone Filter* will be given and later the *Filterbank* idea will be shown. A *Gammatone filter* can be formally represented as follows in the form of impulse response in the time domain [41]:

$$g(t) = at^{\nu-1}e^{-2\pi bt}cos(2\pi f_c t + \phi)$$

where f_c is the central frequency of the filter, and ϕ is the phase (usually set to zero). The constant a is the amplitude and ν the filter order. There is also b as decay factor, which is related to f_c. To simplify the operation of a filter as much as possible, it can be seen as composed of a series of coefficients whose objective is to modify the signal. Passing

[35] *Gammatone filter* banks were designed to model the human auditory system.

[36] The ERB is a psychoacoustic measure of the width of the auditory filter at each point along the cochlea[43]. The cochela is where the sensory organ of hearing is contained.

an impulse through a filter, this pass across a fixed frame of coefficients. Multiplying each couple *filter coefficient* and *sample*, a number of values equal to the number of coefficients are obtained, through a sum of the values of these products the output sample are extracted. The sum of the products will reproduce the shape of the coefficients curve [42]. The bandwidth of the filter is proportional to the ERB representation of human auditory filter [44]. An estimation of the size of an ERB in the human auditory system is $ERB(f_c) = 24.7 + 0.108 f_c$ [45].
The scale is defined as the number of ERBs below each frequency with $ERBs(f_c) = 21.4 \log_{10}(0.00437 f_c + 1)$. This distribution led to a greater grouping of filters at low frequencies compared to high frequencies, as can be seen in the example at figure 2.13 .

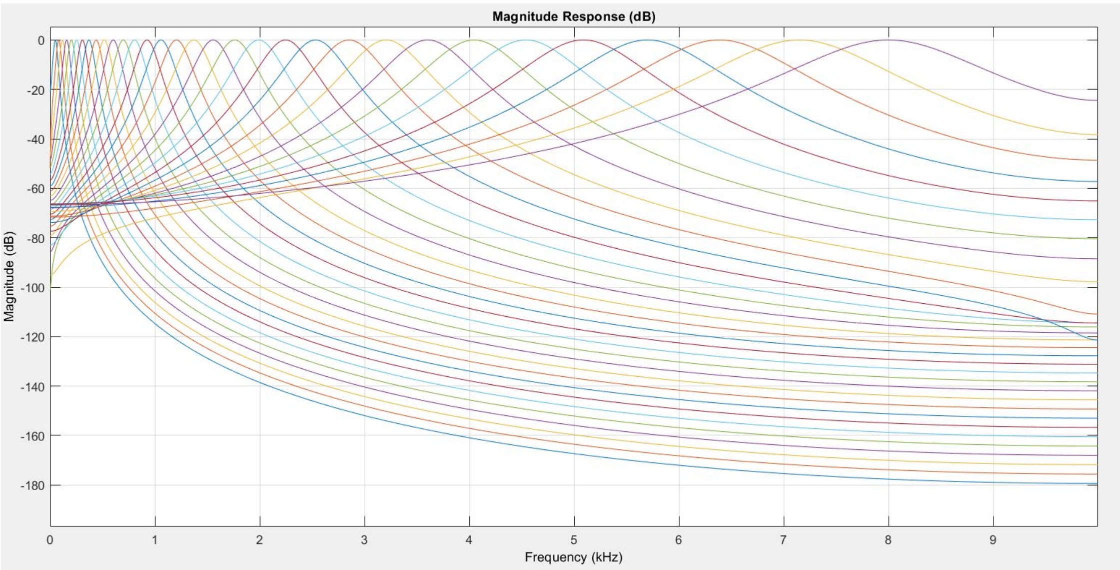


Figure 2.13: In this example the magnitude frequency response of 32 Gammatone filters in a frequency range spanning from 50 Hz to 8000 Hz is shown.

To analyze the signal, a so called *Gammatone FilterBank* was created. A filterbank is a system that divides the input signal $x(n)$ into a set of analysis signals $x_1(n), x_2(n), \ldots,$ each of which corresponds to a different region in the spectrum of $x(n)$[37]. In the presented study the regions in the spectrum given by the analysis signals collectively span the entire audible range of human hearing, from approximately 20 Hz to 20 kHz. So a filterbank $h_m[n]$, $m \in [0, M-1]$ indicate the impulse responses of M gammatone filters. The $x(n)$ signal is split in M frequency subbands, in order to analyze it.
Moving to a more practical environment, during the analysis different filtrbanks are used: it depends whether a more specific or wide evaluation must be performed. The creation of these filter banks is done via a simple Matlab command, and its most important features are:

[37]Definition adapted from the one of the Center for Computer Research in Music and Acoustics of the Stanford University.

1. the Sample Rate, that is set equal to the sampling frequency, which is 44100 Hz as written in 2.3.1;
2. the number of Gammatone filters used in the Bank i.e., 256;
3. the Frequency Range indicates the frequency boundaries of the filterbank, which spans, as said, along the whole human auditory interval.

Once the filterbank has been created, it must be used to analyse the signal. The signal is sampled, along the time domain, with windows of length equal to 50ms and with 30ms of overlap.
With the divided signal, in overlapped windows, the square value of each window for each signal is taken, in order to obtain later the RMS value.[38]. In other words the signal energy[39] in each window for each channel i.e., the working range of a Gammatone Filter, is being calculated.
Once the energy value of each band is obtained, this is converted in dB and stored in the *D_sign* vector i.e., the vector that will be used to perform the difference between original and modified signal. In fact vectors relative to the dB value of the two signals in the same frequency band will be compared, with a subtraction, to generate the input of the algorithm explained after in 2.3.3.
Some real examples of the obtained spectrum are shown in figure 2.14, 2.15 and 2.16.
On the left one will always find the reference signal, while on the right hand the modified signals' colour maps used to display the time-frequency representations are visible.
The x axis is the time axis, and it is in seconds.
The y axis is the frequency axis, it is in kilo Hertz: the frequencies span from 20 to 20.000 Hz.
The z axis is expressed in terms of colours: there is a legend on the right of each graph, going from dark blue to yellow the dB values of that particular band in the given window rises from -200 to 0 dB.
For this example a FilterBanks with the following features has been created:

(a) Sample Rate equal to the sampling frequency (always 44.100 Hz);

(b) 64 Gammatone filters,spanning from 20 to 20.000 Hz;

[38]The Root Mean Square value is the square root of the mean squared:

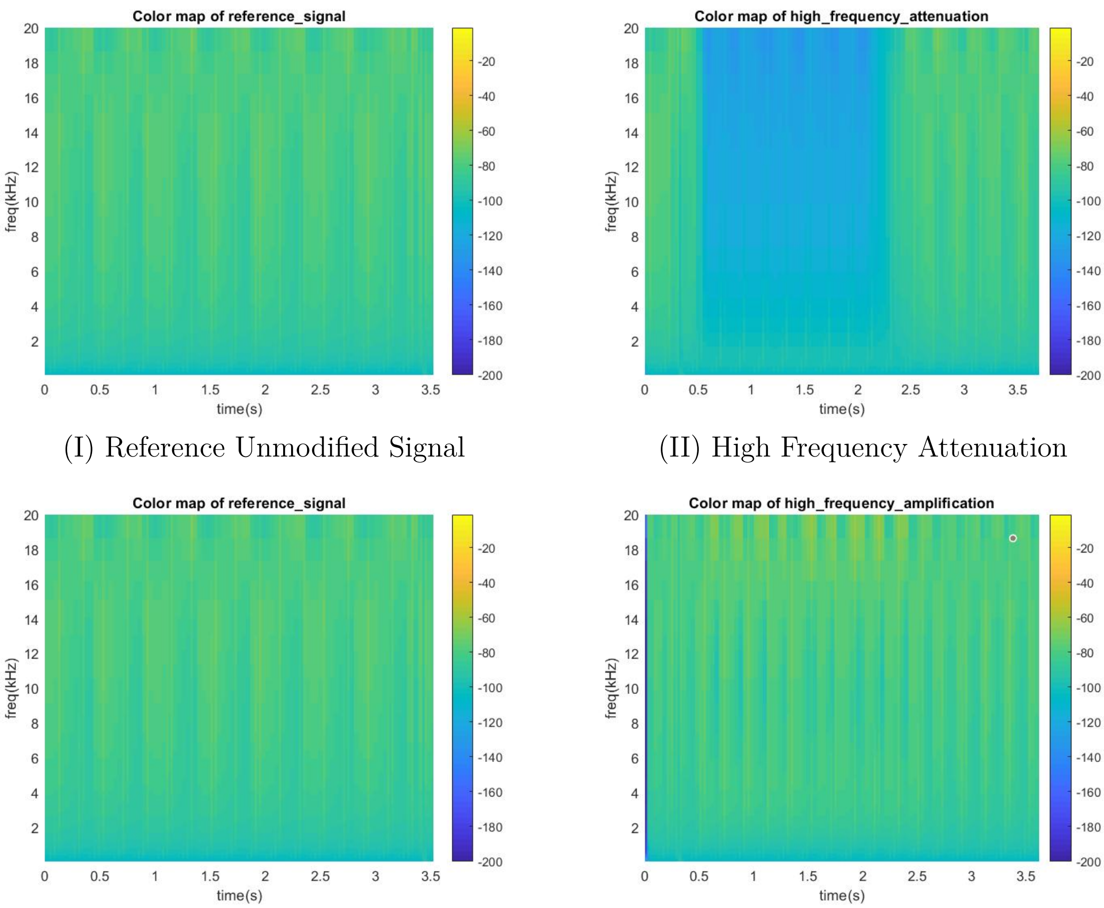
$$x_{RMS} = \sqrt{\frac{1}{n}(x_1^2 + x_2^2 + \cdots + x_n^2)}$$

[39]For discrete-time signals, the definitions for energy and power are similar to those for continuous-time signals by replacing integrals by summations[46]:

$$\epsilon_x = \sum_{n_{in}}^{n_{fin}} |x[n]|^2$$

(c) a time sampling of 50 ms, with 30 ms overlap, to create the windows.

The displayed modulations happen just for some time windows, then the signal is brought back to its original value.
In order to have the most visual impact possible, all the modification have been done by taking the filters at the edge of their working range: so at +/- 100%.
Already, thanks to these 3D matrices, a few features about the system's components can be derived.
The first, more glaring, indication is given by the strongly different change in colours (so in dB value), when an attenuation or an amplification is performed: this leads to the conclusion, that will later be discussed in 3.2, that the filters have a non linear behaviour.
The second, more obvious, is the possibility to see how each filter have its specific working band.



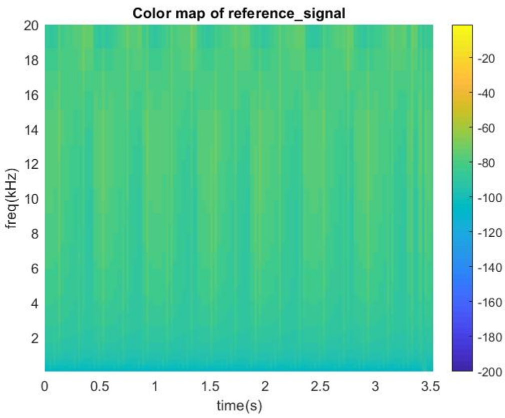
(I) Reference Unmodified Signal

(II) High Frequency Attenuation

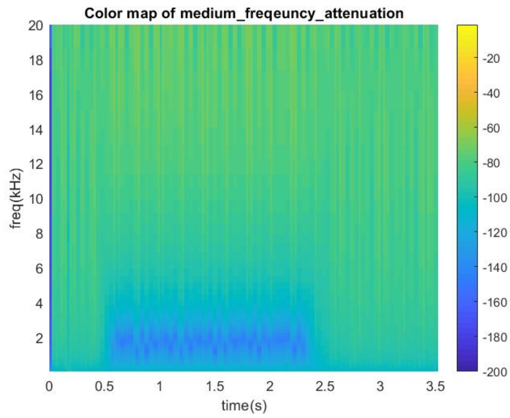
(III) Reference Unmodified Signal

(IV) High Frequency Amplification

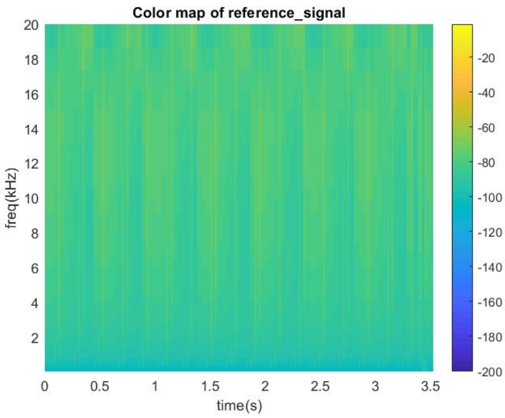
Figure 2.14: The High Frequency modifications Colour maps



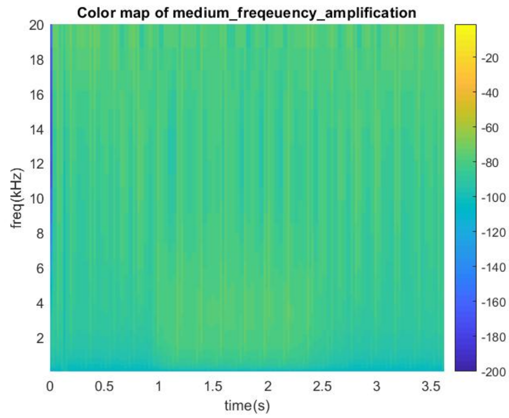
(I) Reference Unmodified Signal



(II) Medium Frequency Attenuation



(III) Reference Unmodified Signal



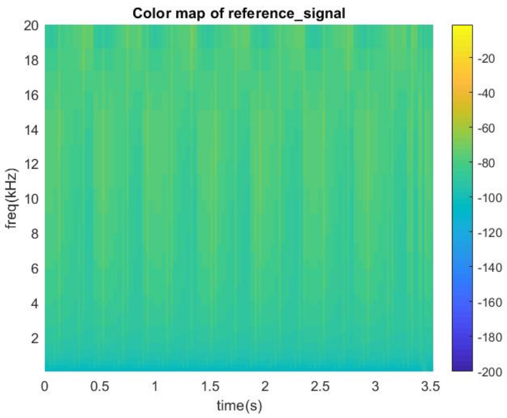
(IV) Medium Frequency Amplification

Figure 2.15: The Medium Frequency modification Colour maps

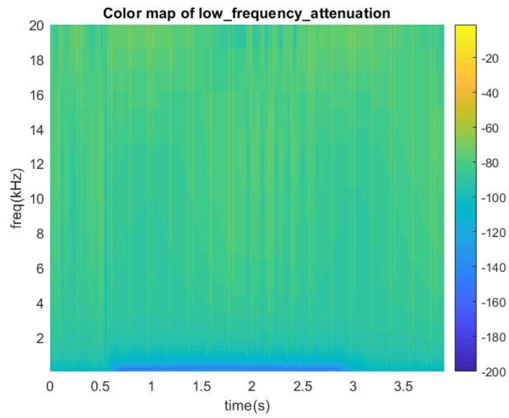
2.3.3 Clustering and Extraction

This is the function through which the double evaluation in terms of instant and amount of the change is performed. Its input is always the difference in dB between the modified and the original signals at a given frequency range: the operation of selecting the right value, explained in 3.3, permits the reduction of input's dimensions that is characteristic of the workflows similar as the one introduced in 2.2.5. Conversely the outputs, relating to the measurements in terms of energy per frequency band, are different, based on the reason of the conduced analysis: it is always evaluated the average value of the modification, sometimes also the minima and maxima in order to generate a *"route"* of the change, this differences will be highlighted later in 3.2, 3.3 and 3.4. The time indication is always a vector with two values: beginning and end of the action made by the DJ through the console.

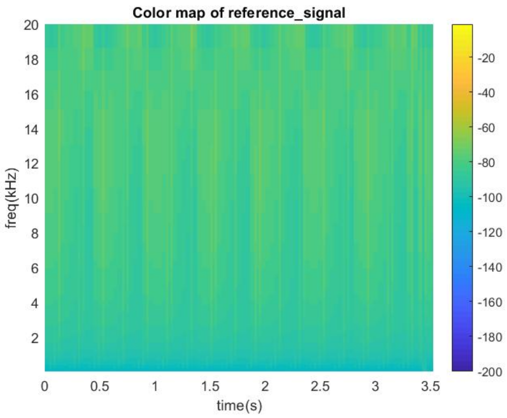
The idea is to divide each data-set i.e., the function's input, in three different subsets:



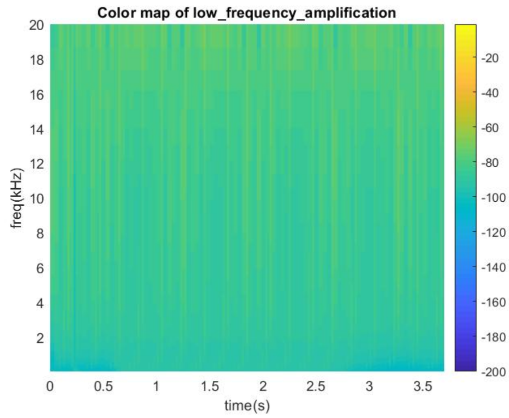
(I) Reference Unmodified Signal



(II) Low Frequency Attenuation



(III) Reference Unmodified Signal



(IV) Low Frequency Amplification

Figure 2.16: The Low Frequency modifications Colour maps

the *pre-modification* , the *modification* and the *post-modification* ones. Once that the time-series is partitioned, it is possible to find the average, maximum and minimum values of both the pre and post modification data subsets: these will be the outputs of the function i.e., the extracted features in terms of modulation's strength. Additionally the moments when this operation is performed will also be identified.

Clustering

Before going in the depth of the algorithm, a briefly overview of some pattern recognition concepts is needed. Thanks to this technique, it is possible to classify samples according to some propriety.

A *classification* problem occurs when an object needs to be assigned into a predefined group or class based on a number of observed attributes related to that object [47].

While *clustering* refers to the division of data into groups of similar objects. Each group, or cluster, consists of objects that are similar to one another and dissimilar to objects in

other groups [48].
It is easy to see how these two definitions seems to refer to the same concept. So what is the difference between the two above problems? To provide an answer, some Machine Learning's (also referred as ML) concepts are useful.
In the ML field it is possible to identify two main types of tasks: supervised and unsupervised.
The main distinction between these two is that the supervised learning is using prior knowledge of what the output values for the samples should be: the objective is find a function that, given a set of input and desired output data, best describes the relationship between these two datasets. Unsupervised learning, on the other hand, does not have labeled outputs, so its goal is to infer the natural structure present within a set of data points.
Either way the aim is to find specific relationships or structure in the input data that allow to effectively produce correct output data. In the scheme at 2.17, it is possible to appreciate a graphical explanation of what is being discussed.

Supervised learning is used in the field of classification or regression (i.e., when the output is continuous). It is very important to stress how, along this path, the correct output is determined from the training set so, even if the chosen model can rely on an assumed "ground truth", the result will not be always correct in real world applications. Another important aspect to keep in consideration, is the need to find the right model's complexity: it must be able to describe in the best possible way the sought relationship, without loss of generalization. Following the high complexity course, it will only learn how to reproduce the training set, without understanding the current trend that led to this output, the opposite situation is not being able to find any pattern. The model's variance should also be able to scale with the size and complexity of the training data e.g., having to interfere two points in a plane a straight line will be enough, while with three of them maybe a parabola will be a more suitable choice.

Unsupervised learning is mostly used for clustering, when the goal is to learn the structure without using explicitly provided labels. This kind of approach automatically identify structures in data. [49]

After this brief introduction, it is time move on to the thought and implemented algorithm. Its peculiarity is the fact that it does not require any training set: this is mostly due to the need to be as flexible as possible (the artists while performing the modification, as stated before, do not have any type of rule to follow) and also to the lack of time and resources. What has been implemented is a *K-Clustering*[40], where $K = 3$. The data set is divided by means of a specific value obtained from an evaluation of the single sample with respect to the whole data-set. Two binary vectors are then created, which are called *varup* and *vardown*: looking at the evolution of the index resulting from the previous evaluation a 1 or a 0 is assigned at both vectors for each point of the input. The successive step is an attempt of making more robust the segmentation. Since only a partition in three of the

[40] A procedure which follows a certain criteria in order to cluster the data set in a given number i.e., K, of subsets.

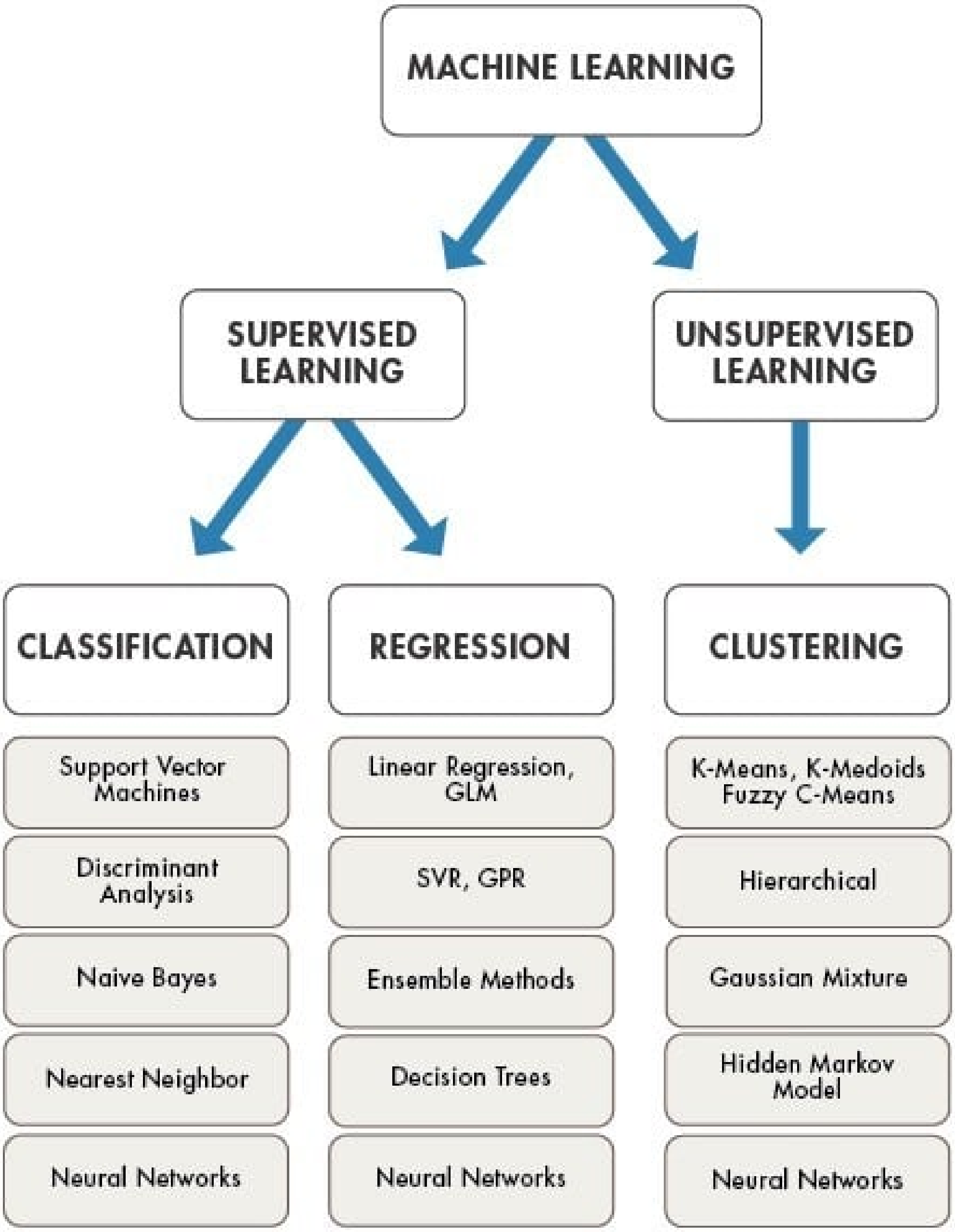


Figure 2.17: The presented block scheme clarifies some aspects of the discussed subdivision

input stream is requested, the longest series of ones or zeros is identified: this will indicate the section of the modification made to the input signal. Now that the *K-clusters* are obtained, these will be made more robust and the wanted values will be than extracted. In algorithm [1] it is possible to appreciate how this clustering mechanism proceeds.

To have a clearer view, now the input is presented: it is the difference between the dB

Algorithm 1: K-Clustering

```
input  : A vector x_k of size n
output: A partition of the vector

n = length of the input vector;
for i ← 1 to n do
    compute index(i);
varup and vardown are introduced;
for i ← 1 to n − 1 do
    if index(i + 1) ≥ index(i) then
        varup(i) = 1;
        vardown(i) = 0
    else
        varup(i) = 0 ;
        vardown(i) = 1

/* Try to make robuster the found intervals */
if varup has a series of ones, with a 0 in between then
    substitute that 0 with a 1
if vardown has a series of ones, with a 0 in between then
    substitute that 0 with a 1
/* Find the longest interval in the data stream */
if the difference between a binary value and the following is not 0 then
    Start a new interval
else
    Continue to append to the actual interval
Find the longest interval in both varup and vardown
if longest interval of varup > longest interval vardown then
    the modification is an amplification discriminating interval = longest interval of
     varup
else
    the modification is an attenuation & discriminating interval = longest interval
     of vardown
I = [discriminating interval(start) discriminating interval(end)]
/* The binary vectors are used to identify the instants of start and
   end of the modification */
Divide the input vector x_k in three subsets:
pre modification = x_k(1 : I(1));
modification = x_k(I(1) : I(2));
post modification = x_k(I(2) : end);
```

evaluations of the modified track and the reference one.
As said in 2.3.2, from the audio signal is derived a discrete 3-D matrix, where the entries are the dB values (after the conversion from energy) of the signal corresponding to each *(sampling instant, frequency value)* couple.
By choosing a *frequency value*, it is possible to obtain a vector from the above matrix: it will indicate the dB amount of a signal, sampling instant for sampling instant at that selected frequency.
The criteria which led to the choice of the right *frequency value* will be discussed later in 3.3.3. The three vectors: modified signal, reference signal and their difference at the given frequency are showed in the following pictures, 2.18 and 2.19: these present two different modification, performed by the same filter i.e., the one at high frequency of the first channel. Graphically it is easy to understand the basic idea of clustering the *difference vector*: the three sub sets are clear at first sight. One key aspect to keep in mind is the apparent plateau that characterizes the post modification interval: this is not always obtained, as introduced in the critical analysis 2.2.4 and explained in its subsection 3.2.4. To refresh the idea: when over bounding some limits in the modification's strength, there an unwanted effect appears and the signal can become compromised, making this kind of analysis impractical.

The input is a vector of values, obtained thanks to a dimension reduction from the 3-D matrix representing the signal's frequency spectrum.

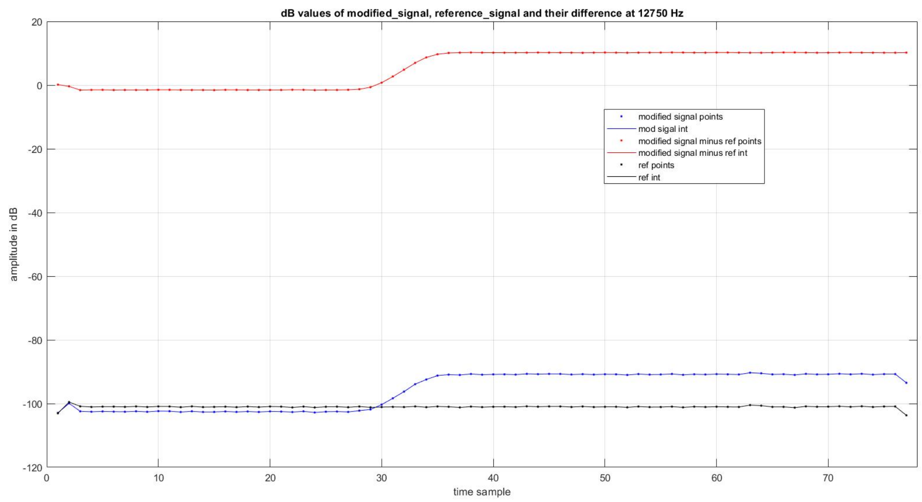


Figure 2.18: Amplification of 99% at high frequencies. The selected frequency value at which the evaluation is performed is 14.250 Hz. The black signal represents the reference signal, the blue the modified one and the red their difference i.e., the input of the function.

The figures show the points and also an interpolation between each value and the successive only to facilitate the reader's visual evaluation: the algorithm works with the amount represented by each single dot. The first approach has been to implement some robust regression algorithms presented in [50] and test them, in order to find if one could

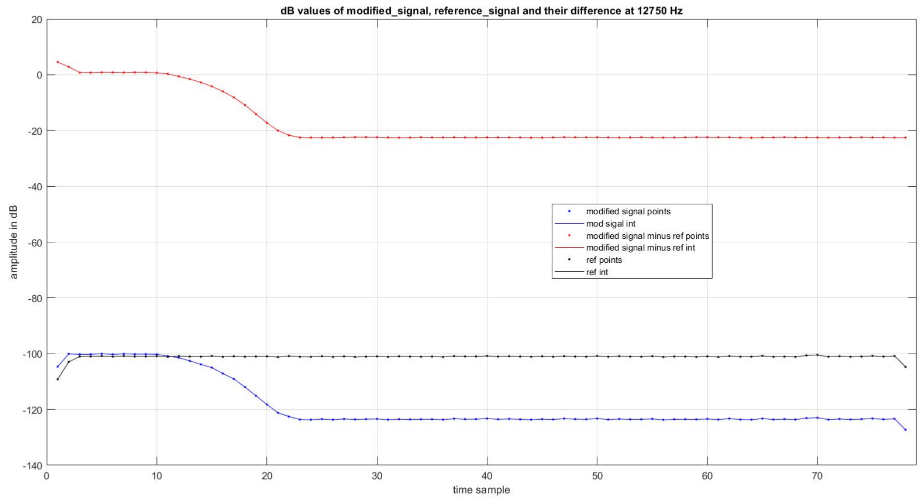


Figure 2.19: Attenuation of 64% at high frequencies. The selected frequency value at which the evaluation is performed is 14.250 Hz. The black signal represents the reference signal, the blue the modified one and the red their difference i.e., the function's input.

be suitable to fulfill the task. Considering

$$y = X\,\beta\, +\, \epsilon,$$

where y, the measured output, and ϵ, the unknown noise, are vectors of length n, X is an $n\,\times\,p$ matrix, representing the relationship between input and output, and β is the input vector of length p. A robust estimate for β, $\hat{\beta}$,is the one that minimizes :

$$\sum_{1=1}^{n} \rho(\frac{y_i - x_i\beta}{\sigma}),$$

with σ known parameter, ρ robust cost function i.e., a way to approximate the output stream, and x_i i-th row of X. Since it is a minimization problem, this means that, with $\psi = \dot{\rho}$, the searched $\hat{\beta}$ is the one which satisfies:

$$\sum_{1=1}^{n} x_{ij}\psi(\frac{y_i - x_i\beta}{\sigma}) = 0$$

The idea was to create a sort of "running regression": updating the value of β point-by-point, when a certain threshold in the difference $\beta_n - \beta_{n-1}$ would be crossed, then this meant that the value corresponding to the n_{th} sampling instant belongs to a new group. In order to obtain a suitable limit value, a good way would have been using the scheme presented in [51] of an adaptive threshold for outlier detection could have been utilized: dividing the data set in training and validation set, a classical problem could have been set up.
As an observant reader may have already noticed from the title of the subsubsection: this

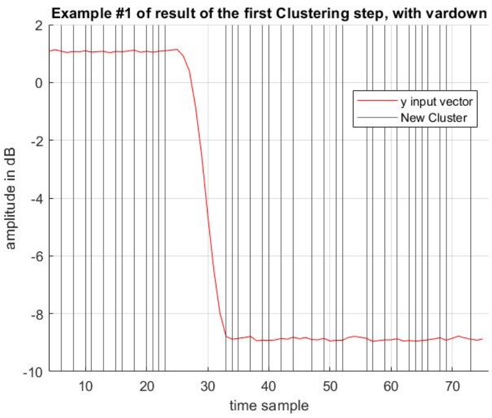
was not the route taken.
The implementation derives from a read about the optimal detection of changepoints [52]. The changepoint analysis can be defined as the identification of points within a data set where statistical properties change. Given a dataset $y = [y_1, \ldots, y_n]$, it will have a number of changepoints $m \leq n$ at position $\tau = [\tau_1, \ldots, \tau_m]$, splitting the y vector in $m+1$ segments. Adapting the Kullback-Leibler divergence [53] i.e., a measurement of information lost when using a given model to approximate a dataset [54], an index *ind* has been obtained:

$$ind(i) = \frac{1}{n}\sum_{i=1}^{n}(y_i * \log_{10}(\frac{y_i}{n * \sum_{i=1}^{n} y_i})).$$

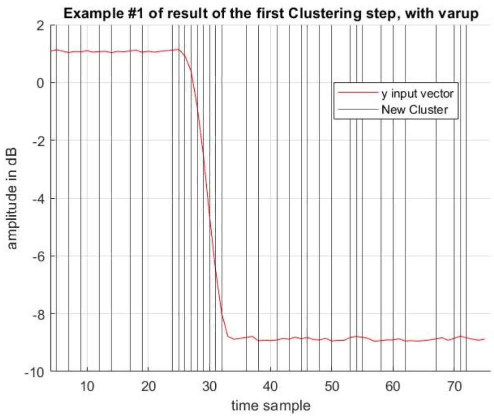
Where n is the input vector's length. If someone is familiar with quantum mechanics, the equation holds something that just who arrives from information theory should have been able to understand, thanks to "information lost": it is an entropy-based value [55]. Since each point of the dataset has an identical independent probability distribution, it was possible to take Boltzmann entropy's idea, so that $Entropy = k * \ln(Disorder)$ [56], and adapt it to the application. What *ind(i)* tells is how much entropy the i^{th} value of the input vector adds to the stream. If a person decides to walk following an imaginary path, until the mental route will be straight in front of him, he's gonna have to spend a little energy to follow it trying to stay on track, but when the next step has to be done in another direction, there some problems may arise: in order to reach the goal this could fall, thus increasing his entropy (and the one of the Universe).
The *ind(i)* value is an indication of how strong the turn of the "next step" is, with respect to the trail followed up to that point. If the y_i is in a friendly position for the imaginary walker i.e., if this point is aligned with the ones before, *ind(i)* will be small.
Comparing the index value of a point with the previous one, two vectors, called *varup* and *vardown*, are updated : when $ind(i) \geq ind(i-1)$, than *varup(i) = 1* and *vardown(i) = 0*; if $ind(i) < ind(i-1)$, than *varup(i) = 0* and *vardown(i) = 1*. The step of converting some integer values in a binary vector has been performed to simplify the analysis. A new subset is initialized when a change in the entry is experienced e.g., if *vardown* has a series of zeros and then a one comes: there a new cluster will start (it has to be stressed out that the opposite variation will happen in *varup*). This double-vector has been introduced so that it was possible to recognize either an amplification was taking place, so a positive difference between modified and reference signal i.e., an increasing variation; or an attenuation, resulting in a variation generating a downward direction input stream.

The result of this first clustering step on the input is appreciable in 2.20: both a 43% signal's attenuation at high frequencies and an amplification of a signal of 65% at medium frequencies are displayed.

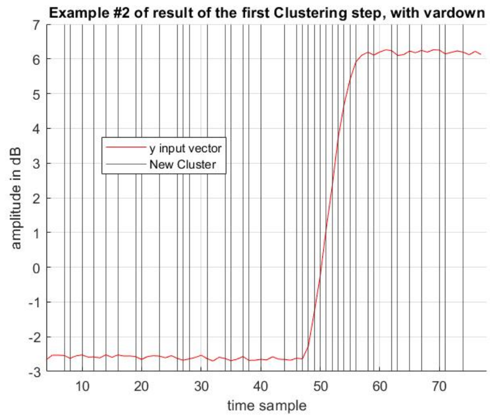
After this subdivision, the section corresponding to the modification interval is clear: when the system performs an attenuation, there *vardown* will have a long stream of ones in its correspondence, while if an amplification is actuated the ones stream is appreciable in *varup*. Before imposing a maximum number of clusters, a small robustness enhancing step must be performed: the signal modulation is not always steep enough to generate a continuous cluster corresponding to its duration, so a little trick which checks whether a long series of ones is interrupted by a single zero has been introduced.
The request of a $K = 3$ value has been enforced just by searching the "*longest-run*" of ones i.e., the subset corresponding to the system's state change: creating the vector of the



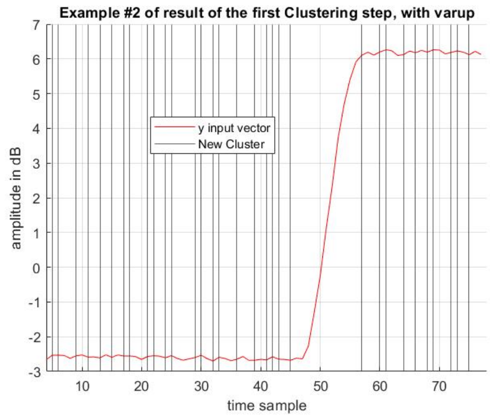
(I) Result of the *vardown*-means clustering



(II) Result of the *varup*-means clustering



(III) Result of the *vardown*-means clustering



(IV) Result of the *varup*-means clustering

Figure 2.20: Here the first clustering is displayed, both by means of *vardown* and *varup*. Each black vertical line indicates the boundary between two successive subsets.

absolute value of the differences between two following entries of the binary vector, it was possible to look at the longest stream of zeros between two ones (if the difference is zero, then there is a constant growth or descend).
The results of the overall clustering process, for the above presented example, are shown in pictures 2.21 and 2.22.

Now that the input has been divided as requested, the algorithm can proceed to extrapolate the wanted values.

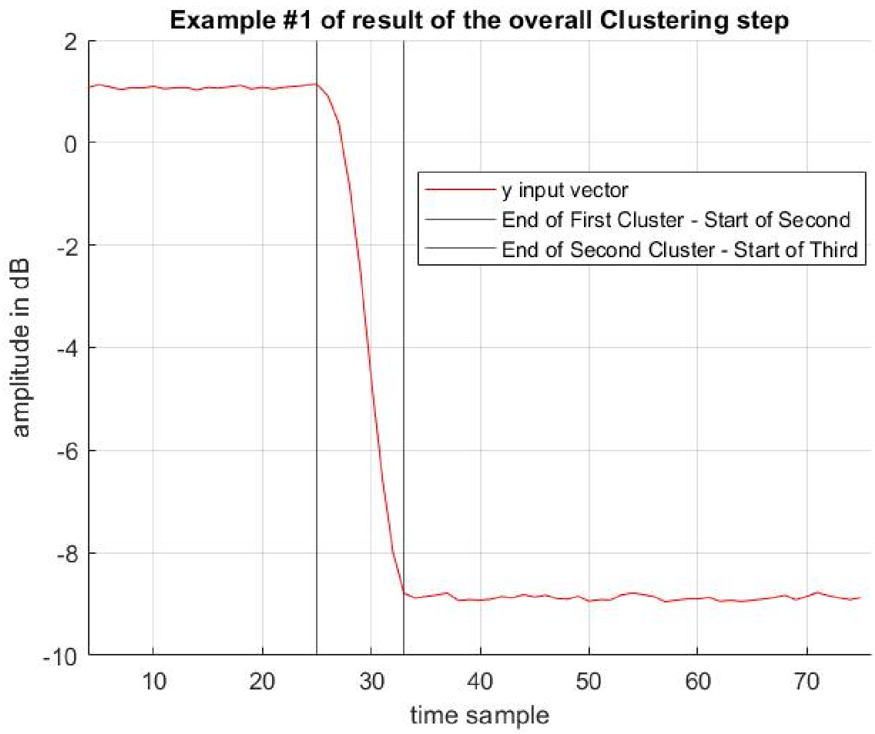


Figure 2.21: The final result of the applied 3-Mean Clustering process for an attenuation of 43% at high frequencies. The black vertical lines indicate the boundary between two subsets.

Value extrapolation and extraction

This part refers to the algorithm 2.

The easier obtainable information is the one regarding the time indication of when the system starts changing its value and when it stops: going to find the position in the input vector relative to the beginning and end of our *longest-run*, it will be possible to discover what has been sought. So a two-values vector I, indicating the relative instant of start and finish of the modification inside the sampled interval is generated $I = [longest - run\_start \quad longest - run\_finish]$.
The extraction of the modification's value is more complicated. Depending on the use that will be made of this algorithm, only the average values of the two *pre-modification* and *post-modification* subsets can be used, otherwise the maximum and minimum values of each one will also be necessary. Incidentally, these are the non-linear operations.
A further robustness move is first performed: if the boundaries of these two clusters corresponds to their minimum or maximum values then that limit point will not be considered as part of the subset. This step is needed to avoid that, if the *K-Means* algorithm has not perfectly identified the start or finish of the modification, this mistake will influence each function's output.

Algorithm 2: Value extrapolation

input : The three vectors obtained before: pre modification, modification, post modification

output: The needed extracted values

Obtain average, maximum and minimum values of each segment.

Calculate the variance of the pre modification and post modification intervals.

```
/* These will be useful later for a robustness check */
/* In order to be sure to have the most flat intervals possible, its
   boundaries are checked */
```

if *pre modification(start) is minimum(pre modification) || pre modification(start) is maximum(pre modification)* **then**
- pre modification = pre modification(start+1:end)

if *pre modification(end) is minimum(pre modification) || pre modification(end) is maximum(pre modification)* **then**
- pre modification = pre modification(start:end-1)

if *post modification(start) is minimum(post modification) || post modification(start) is maximum(post modification)* **then**
- post modification = post modification(start+1:end)

if *post modification(end) is minimum(post modification) || post modification(end) is maximum(post modification)* **then**
- post modification = post modification(start:end-1)

Check the variances of both pre modification and post modification subsets. **if** *variance pre modification > variance threshold* **then**
- Update its mean value by means of a weighted average algorithm

if *variance post modification > variance threshold* **then**
- Update its mean value by means of a weighted average algorithm

The sampling instant can also happen during a modification. This would lead to empty intervals. Here is accounted for this eventuality.

if *pre modification is empty* **then**
- Its average, maximum and minimum values are all the same, and equals to the single value corresponding to the sampling interval

if *post modification is empty* **then**
- Its average, maximum and minimum values are all the same, and equals to the single value corresponding to the sampling interval

Finally the desired values:

average difference = post modification mean - pre modification mean.

maximum difference = post modification maximum - pre modification maximum.

minimum difference = post modification minimum - pre modification minimum.

modification interval = [pre modification(*end*) post modification(*start*)].

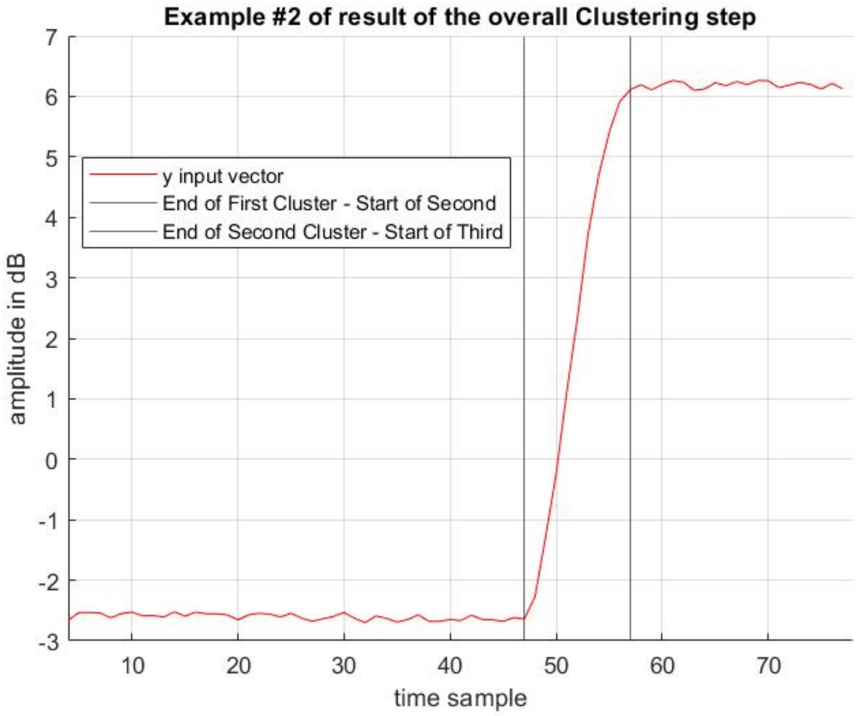


Figure 2.22: The final result of the applied 3-Mean Clustering process for an amplification of 65% at medium frequencies. The black vertical lines indicate the boundary between two subsets.

Now the average value and the variance of the *pre modification* and *post modification* ranges are calculated as:

$$interval\ average = \frac{1}{interval\ length} * \sum_{1}^{interval\ length} interval(i),$$

$$interval\ variance = \frac{1}{interval\ length - 1} * \sum_{1}^{interval\ length} |interval(i)|^2.$$

At the same time, the maximum and minimum values of these spans are also evaluated. The last check, in order to obtain the most consistent values possible, concerns the variance of each interval: if this amount is above 0.5, then it is likely that the subset contains an outlier[41]. In order to avoid that a single measurement could adversely affect the outcome, when necessary, a weighting function is introduced: understanding how far from the average is each point, it is possible to associate to it a penalization. First the residual i.e., the difference between a value and its estimate, is obtained as

$$res(i) = interval(i) - interval\ average,$$

[41] An abnormal value compared to the others components.

from this amount, the corresponding weight is evaluated as

$$w(i) = \frac{1}{1 + (\frac{res(i)}{C})^2}.$$

The reward decreases according to the power of two with the increase of the residual amount, and with a normalization with respect to a constant $C = 1.385$ obtained empirically. Now each cluster's point is multiplied for its weight, and the average value is evaluated as before, with this newly weighted vector.
To account also for the chance that the sampling may occur during the alteration interval or just at its boundaries, a check to create single-point intervals corresponding to the start or finish of the whole set is added.
The differences between average, maximum and minimum amounts of the *pre-modification* and *post-modification* set are computed and taken as output.

Chapter 3

Analysis Implementation

3.1 Introduction

After that the signal analysis has been described in chapter 2, here some practical applications of the algorithm are going to be presented.
Since the final goal of the thesis is to solve the raised issue by the the users themselves, 2.2, the mixer system had to be modelled. With a process that develops from the detail to the general: first the single knob was studied, and then it was possible to move towards the complete representation.
In particular thanks to the previous knowledge of the console and the testing phase, in which for each step the extracted features were different, a grey-box model of the hardware has been reached.
The filter behaviours, discussed in 3.2, have been obtained collecting the results of all the non linear operations (maximum, minimum and average), carried out at precise frequency intervals, and than extrapolating a "railway" characteristic.
The filter masks, which then allows to get a representation of the system, have been achieved by the single assessment of the average delta of the energy-per-frequency-band but, this time, over the whole frequencies spectrum. As will be shown in 3.3: various models have been created to cover the widest range of possibilities.
In both cases it was not possible to reach the completeness of the representation: as it will be read, there were some problems highlighted by the test phase.
All this work was necessary to obtain the recursive algorithm (found in 3.4) that analyzes the audio signal coming out of the mixer and with the previous knowledge of the tracks that are played, is able to trace back the system's states i.e., each knobs' position. The filter characteristic is used to understand the working percentage of each knob, then translated into position, and to select the right set of models that, through an optimization process, will update the reference signal so that the subsequent variation is evaluated with respect to the previous state and not the "zero" one. Finally, the temporal indications of each modulation are also collected: all this data are stored in a database that serves as a link to a GUI, as will be further explained in 4.

3.2 Filter Behaviour

With an inductive method, the work was developed starting from the study of a single component of the mixer: a single filter.
As stated, the DJ-console "object" performs the modification in a time-varying mode. What this section will show is that this task is also non-linear.
The followed testing-procedure was to feed the system with mathematically generated inputs, perform a single known change and collect the output. Having a series of modified signals, it was possible to trace the filter's behaviour.

3.2.1 Creation of the inputs for the testing and test phase

The inputs for the tests have been obtained as a sum between a plateau of constant amplitude along the whole human hearing's range, and some randomly generated sines with different amplitudes, added in order to simulate more real frequencies' distributions.
The plateau is formed generating ten thousand sin functions logarithmically spaced along the whole human auditory arc, that is from 20 to 20 kHz. Since the goal is to work with real signals, this element will only act as a white noise with small amplitude, thus any of the sines composing it is of the form:

$$y_i = 0.001 * sin(2\pi f_i t).$$

The difference between the built tracks lies in a vector of sixty-two sines, casually generated, these have been then both randomly weighted as follow: $w_i * y_i$, both were left to their original shape. The sines and their eventual weights are listed in appendix A.
Each test has been performed by carrying out a known modification to the input signal through the console, and collecting the result of this operation i.e., the output signal. The possibility to know the amount of the applied change is given by some software, sold together with the hardware components, which receive the information from rotary encoders mounted at each knob of the mixer. These programs are equipped with a GUI, which allows users to choose the tracks to play, to see the status of the system i.e., the value of each component, and has other features not inherent to the dissertation.

3.2.2 Signal Analysis for the Filter Behaviour

As already explained in 2.3.3, a function divides the time-series, obtained reducing the signal spectrum's dimensions, in three subsets: before, during and after the transition. Once these ranges are obtained, some data about them are collected. In the presented case, the extracted information were all those that the algorithm could provide: maxima, minima, mean values of *pre modification* and *post modification* and modification's start and finish.
The length of the modulation have been collected for a further study on how each mixer changes with respect to the the current state of the filter (the non-linear characteristic implies dependency between current system state and evolution).
Regarding the *pre modification* and *post modification* readings, these was needed to obtain a kind-of railway filter characteristic i.e., an interval in which the value obtained from the recursive algorithm, explained in 3.4, could be collocated.

By simply performing a difference between *post modification* and *pre modification* amounts, these *deltas* were found:

$$\Delta\, average = average_{(post\_modification)} \;-\; average_{(pre\_modification)},$$

$$\Delta\, maximum = maximum_{(post\_modification)} \;-\; maximum_{(pre\_modification)},$$

$$\Delta\, minimum = minimum_{(post\_modification)} \;-\; minimum_{(pre\_modification)}.$$

The tests were performed with both the weighed and the normal input.
The above mentioned *deltas* are shown in APPENDIX B. The acquisition of the indications regarding attenuation is not complete, but stops around -65%: this is due to a problem that will be explained in 3.2.4.

3.2.3 Filter Characteristic

With the tests' results, an algorithm, which permits to obtain the characteristics, was created: assuming that the interval between a collected measurement and the next one is small enough, the characteristic has been extracted by means of a linear interpolation between two successive data. This procedure was carried out for the three vectors corresponding to $\Delta average$, $\Delta maximum$, $\Delta minimum$. The obtained characteristic will not be represented by a single sequence, but by an interval that has as limits the maximum and minimum value between all those obtained by the interpolation of the Δ, relative to each filter working point i.e., each percentage value. As numerically appreciable, the modulation limits and dynamics of the filters change when preforming an amplification or attenuation: in particular these components presents a logarithmic behaviour, with horizontal asymptote set to $+12dB$.
A last common aspect between all the obtained characteristics is the number of values associated to the 0% state: around a $+/-5\%$ detection, this reading is more likely to be due a "human error" i.e., the artist wanted to cancel that modification but in the haste didn't reach 0%, or any small mistake due to noise, approximations and similar. To avoid a tilting indication when the filter is lying in a "resting state": the idea of a wider range for this position has been implemented.

HF filter characteristic

The specific tables for this filter's attenuation tests are in B.1 and B.2.
The lower limit has been placed at a state equal to -64% and which is indicated by the couple of values $-23.2358\,dB$ and $-23.3586\,dB$.
The amplification's tests readings are found in B.3 and B.3, here no unwanted strong nonlinearity is detected, therefore its maximum value is the one corresponding to $+100\%$ and is given by the couple $11.7212\,dB$, $11.7188\,dB$.
The frequency at which the data were collected is equal to *14250 Hz*, this decision will be explained in 3.3.
In picture 3.1 this filter's characteristic, in its boundaries, is shown; while the numerical values are listed in appendix C.1.

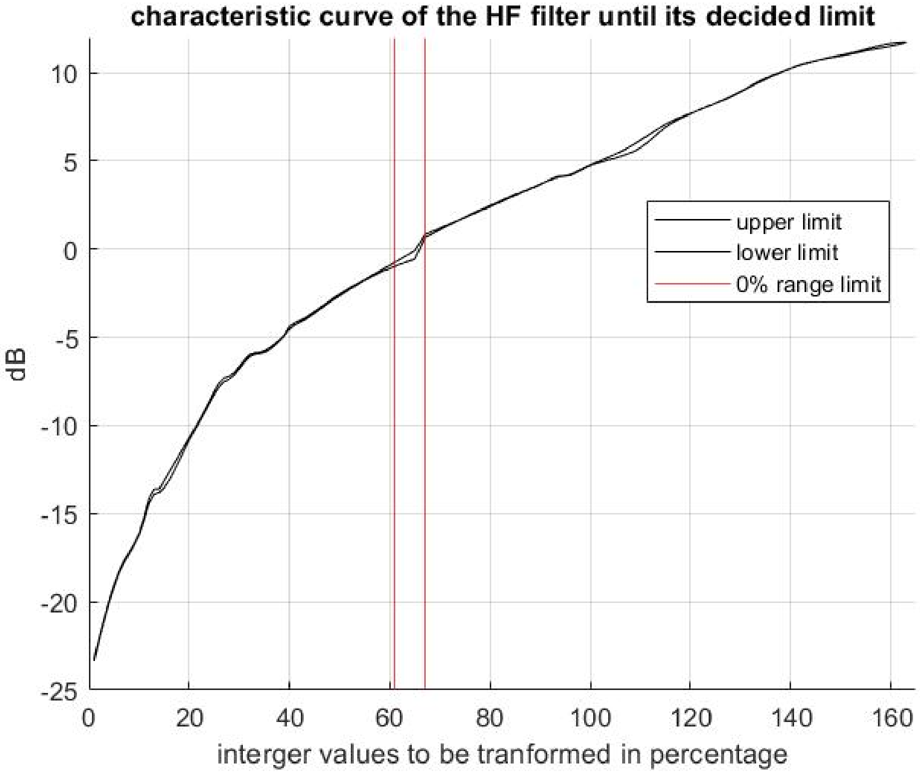


Figure 3.1: The high frequency filter's "railway characteristic" is displayed in black, while in red the 0% interval is shown. The analysis stops at -64%. Form 1 to 61, each value correspond to a % value from -64% to -4%, from 67 to 163, each value is a % from 4% to 100%.

MF filter characteristic

The tables related to this filter's attenuation tests are in B.5 and B.6.
The lower limit has been imposed to a state equal to -63% and which is indicated by the couple of values $-22.5263\,dB$ and $-22.5566\,dB$.
The amplification's test readings are listed in B.7 and B.8, its maximum value corresponds to $+100\%$ and is given by the couple $11.9337\,dB$, $11.8716\,dB$.
The frequency at which the data were collected is equal to *1500 Hz*, again the reason will be explained in 3.3.
In picture 3.2 this filter's characteristic, in its boundaries, is shown; while the numerical values are listed in appendix C.2

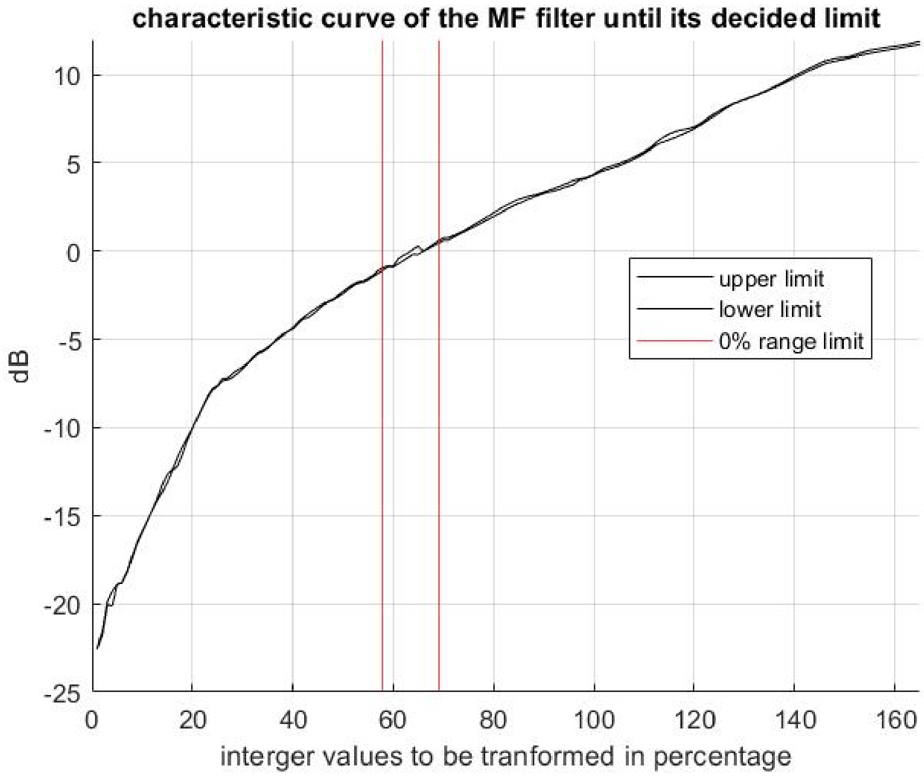


Figure 3.2: The high frequency filter's "railway characteristic" is displayed in black, while in red the 0% interval is shown. The analysis stops at -63%. Form 1 to 59, each value correspond to a % value from -63% to -5%, from 69 to 162, each value is a % from 4% to 100%.

LF filter characteristic

The tables specific to this filter's attenuation tests are in B.9 and B.10.
The lower limit has been imposed to a state equal to -62% and which is indicated by the couple of values $-19.0703\,dB$ and $-20.2243\,dB$.
The amplification's test readings B.11 and B.12, the maximum value corresponds to $+100\%$ and is given by the couple $11.5034\,dB$, $11.4165\,dB$.
The frequency at which the data were collected is equal to *100 Hz*, this decision will be explained in 3.3.
In picture 3.3 this filter's characteristic, in its boundaries, is shown; while the numerical values are listed in appendix C.3

As seen in the results, the behaviour of the different filters is very similar: they have been built in order to modify in the same way from a gain point of view, the small differences can be mostly addressed both to intrinsic fabrications uncertainties and the electronics components position in the circuit (so different disturbances).

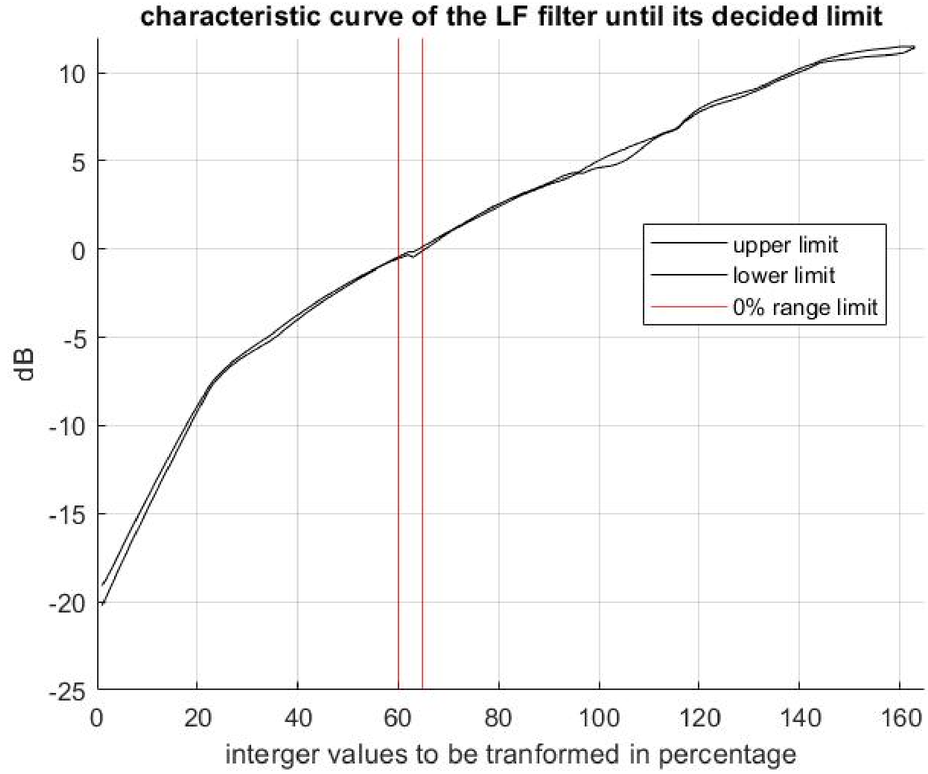


Figure 3.3: The high frequency filter's "railway characteristic" is displayed in black, while in red the 0% interval is shown. The analysis stops at -62%. Form 1 to 59, each value correspond to a % value from -62% to -2%, from 65 to 161, each value is a % from 4% to 100%.

An important consideration, especially for the following paragraph, is to note how the characteristic can be divided into three zones:

- a central one, which can be assumed linear,
- the area corresponding to the maximum value of the amplification i.e., the saturation, which is softly nonlinear,
- the extremely nonlinear part matching the filter's attenuation boundary.

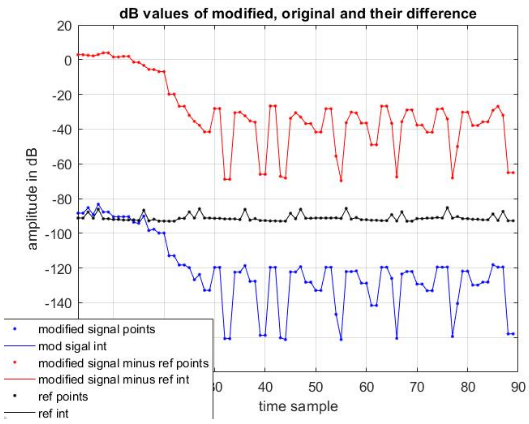
3.2.4 Distortion Issue

"*HIC SUNT DRACONES*": with this phrase the ancients pointed to unexplored territory in their maps, as visible is 3.4. Now the dragon of the thesis will be discussed.
From the testing phase, it was possible to appreciate the presence of a strong non-linear interval: given the filter's logarithmic behaviour this, while attenuating, tries to reach minus infinite (it is possible to think as it tending to an open loop gain). It could occur that, when performing an attenuation above some given values, a strong oscillation in the

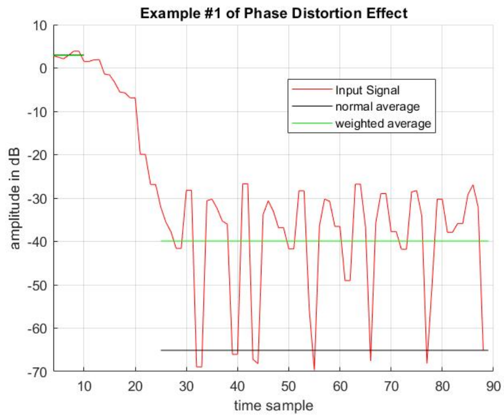
Figure 3.4: An example of a Medieval Map. Source *Wikipedia: Here be dragons.*

resulting time-series is appreciated. As shown in figure the effect is not always the same, it depends on several factors. This difference can be seen in 3.5, even if the changes are both performed around the mixer's lower working limit, and both result in a wave development of the modulated signal, and therefore of the time series, the strength of the consequence is significantly different.

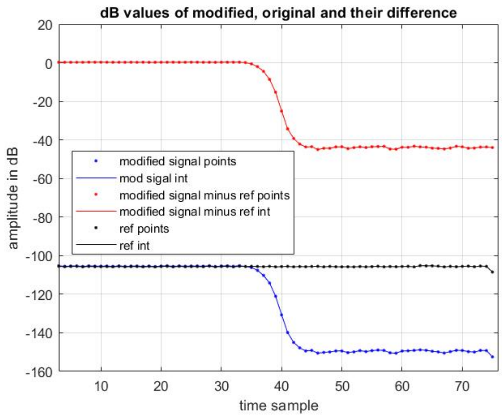
This unintended impact has been observed, and has proven to have no appreciable decay in the time intervals in which the work is carried out. The culprit has been identified in the signal's distortion: in linear systems the output spectrum can only have tones at the same frequencies as the input spectrum, when dealing with nonlinear filters instead some output contributions coming from "external" frequencies may arise [57]. Being a distortion, it modifies the waveform of the signal complicating the simple operation of subtraction between the values of the altered track and the reference one. As anyone can appreciate from the previous paragraph, the non linearity is stronger at each filter's working boundaries. There are usually three major root causes for this problem: amplitude, frequency and phase distortion.
The **amplitude distortion** is known to most because of saturation i.e., when the output voltage would like to go beyond the limits imposed by the supply voltage , but this is not the case. The possible problem can be the intermodulation, which occurs when signals composed by more than one frequency pass through a nonlinear device [58].
The **frequency distortion** describes the condition in which different frequencies within the bandwidth of the amplifier are changed by different amount i.e., when the bandwidth gain is no longer flat. Under such conditions, the signal components are amplified by different amounts and the wave becomes distorted [59].
The **phase distortion** was first encountered on long submarine circuits, so in communication systems, and is a problem that occurs when there is a delay between input and output [60], that is the reason why it was discovered in long transmission systems.
Usually the total appreciated distortion of a signal is due to these three effects simultaneously, and the weight of a single one on the output in different conditions has not been studied in the presented work. This drawback affects the amplitude of the modulation [62] i.e., the gain that the process aims to reconstruct.
As stated during the *Audio Engineering Society* convention of 2000 [61]: "*It turns out that, within very generous tolerances, humans are insensitive [...]. Under carefully contrived*



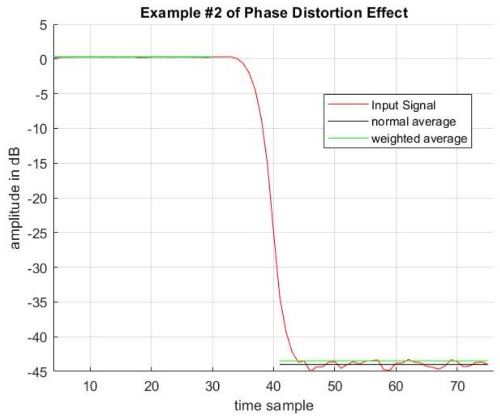
(I)95% Attenuation at Medium Frequency



(II)Attenuation Zoom and averages



(III)100% Attenuation at High Frequency



(IV)Attenuation Zoom and averages

Figure 3.5: The left column displays the reference track in black, the one modified in blue and their difference in red.
The right column focuses just on the difference, still in red, and shows in green the weighted average and in black the normal one.
The first row regards an attenuation of 95% at Medium Frequency, while the second an attenuation of 100% at High Frequency.

circumstances, special signals auditioned in anechoic[1] *conditions, or through headphones, people have heard slight differences. [...]. When auditioned in real rooms, these differences disappear*".
As seen in figure 3.5, the adopted idea of increasing the robustness of the non-linear operation that allows feature extraction i.e., weighting the average, gives good results when applied to distorted signals [63]. Wanting to make this simple: one can think that a filter,

[1] *The anechoic chambers are built to totally absorb the sound waves.*

when set to work at its lower boundary, tends to reach the value of less infinite which, however, as well known, does not exist in real applications. This clash between will and possibility is the cause of the oscillation. From these considerations, plus a knowledge of the statistical use of the mixer by DJs, further studies to model the system when it is in such states, as to generate this condition, has not been carried out for the moment: it is possible to assume that an artist would rather cancel out a certain frequency range altogether than, for example, just for the 80%.

3.3 System Identification and Models

Here the step of deriving a model of the whole console will be discussed.
Starting from the single filter characteristic, the idea was to see each channel as a series of knobs, and for each sampling interval just the one with the biggest influence is considered in order to update the modelled system state.
A first information extrapolated thanks to this analysis, was the identification of each filter working frequencies, this has led to the opportunity to get each characteristic, as explained in 3.2.
Each obtained model will be then used in 3.4 to update the reference, allowing a recursive process.

3.3.1 Filter Mask Tests

As written in 3.2.1, some suitable signals have been created in order to be used as inputs for the system: once a known modification was brought, the output has been collected in order to trace back the console's work. The difference with the previous tests is not in the methodology, but in the object: here the interest has been distributed throughout the complete spectrum to see how much each filter works at each frequency.

3.3.2 Signal analysis for the Filter Mask

Still referring to 2.3.3, after the reduction to a time-series and the clustering in intervals, the non-linear operation to extract the needed information to perform the comparison between *pre modification* and *post modification* subsets is just the average of each one. With an input as the time-frequency representations already shown in 2.14, 2.15 and 2.16 and which key features have been described in 2.3.2, the operation has been collecting just the

$$\Delta\, average = average_{(post\_modification)} - average_{(pre\_modification)},$$

for each of the 256 frequency-band used to evaluate the spectrum. Each test a vector of values was therefore generated, this have *256* entries, each representing the level of average change detected at each frequency range i.e., $avg\_difference = [\Delta average_1, \ldots, \Delta average_{256}]$ where $\Delta\, average_1$ indicates the first obtained difference, at the first sampled frequency of *20 Hz* and $\Delta\, average_{256}$ is the average modification obtained at the 256^{th} frequency *20 kHz*.
Once the masks were obtained, it was possible to identify just the three "winning" climbers, one at each major frequency range (high, medium and low), used for the other evaluations

as the one in 3.2 or 3.4. The "victorious" were the ones in the middle of the maximum modification plateau found for each filter i.e., its strongest working range.

3.3.3 Filter Masks

Some notions common to all models concern: their extension and the parameters from which they are composed.
The results of the testing phase have been used to feed an algorithm able to obtain the behaviour of each component in the complete frequency range. As already discussed in 3.2, there is the need to account for some nonlinearities: this has made it necessary to divide the models not only in terms of frequencies, but also according to the operation they perform (attenuation or amplification). A key aspect to inspect is that, as will be further explained in the following pages: some outliers have sometimes appeared, which is completely expectable, but their presence has been much more concentrated at the boundaries of the models i.e., at either very low and very high and low frequency values. To solve the problem, special considerations have been taken into account for each filter, but in general all these are based on concentrating the generation of the models in the areas without faulty measurements and then correcting them with a posteriori knowledge of how the result should be.
The second shared aspect is that each mask has been normalized with respect to its maximum value: this step is thought to generate more elastic models. These normalized values, when applied in 3.4 are multiplied for the detected highest amount of attenuation or amplification performed by that component, by means of an analysis at the right frequency's interval, to that signal and later applied to update the system's state.

3.3.4 Models Evaluation

From the results of the tests the particular descriptions of the system, which will then be interpolated to obtain the general one, are generated.
The distinctive shape of the tests' results suggested the adoption of three different methods for modelling:

1. *Spline* Model;

2. *Step* Model;

3. "*Point-by-Point*" Model.

The *Spline* Model is founded on cubic smoothing splines. It is a dummy variable[2] model, subject to one or more continuity restrictions [64]. The idea is to fit a dots function i.e., the test's output vector $avg\_difference$, with a third degree polynomial:

$$spline(x_i) = a_i(x - \Delta\, average_i)^3 + b_i(x - \Delta\, average_i)^2 + c_i(x - \Delta\, average_i) + d_i.$$

[2] Dummy variables are variables that take the values of only 0 or 1. In quantitative analysis, a dummy variable is a numeric stand-in for a qualitative fact or a logical proposition [65].

To meet the continuity requirement the above function, its first and second derivatives, $spline^{'}(x)$ and $spline^{''}(x)$, must be continuous on the interval going from $20\,Hz$ to $20\,kHz$ [66]. The mathematical steps to obtain the a_i, b_i, c_i, d_i parameters can be found in [67]. The adopted version generates a tunable, by means of a smoothing parameter p, function by means of an optimization problem which aims to reduce the Mean Squared Error between the model and the test's results keeping into account the roughness, in particular $spline(x)$ must minimize:

$$p\sum_{i=1}^{n}|\Delta\, average_i - spline(x_i)|^2 \quad + \quad (1-p)\int |spline^{''}(x)|dx.$$

Where $n = 256$, and the selected parameter was empirically chosen as $p = 0.06$ guaranteeing a very high smoothness [68].
The *Step* Model is based on another *K-means* clustering, with $K = 13$ and is performed on the result of the previous approximation i.e., *spline(x)*. Taking the start and finish of every subset, a simple linear interpolation between each partitioning point is performed (both in the same cluster and between end of the previous and start of successive one). This has also undergone an *a posteriori* work on it: some of its intervals have been imposed to represent either a total modification i.e., a unitary value, or a non-modification i.e., a zero, following some trends common to all test results. These ranges change according to the operating frequencies of the analyzed component, and also the maximum change amount.
The *Point-by-Point* Model is just the linear interpolation between every two successive values of the vector representing each test's results i.e., $\Delta\, average_i$ and $\Delta\, average_{i+1}$.
In order to evaluate more directly the models: in the pictures that will represent some examples of these, also the test results will be normalized with respect to the maximum detected modification value. Each image will be accompanied by tables containing all the amounts, in the appropriate appendices.
As will be appreciable the biggest problems have emerged at the frequencies of less work for each filter, this can be due to the algorithm used for the analysis: when the changes are almost minimal, chances are it will try to find something that isn't there, this is the reason why around the rest position of each component there is a wide palette of values leading back to this state, 3.2. Others recurrent outliers were found at the at the limits of the human auditory spectrum: these problems led to the creation of models that can't cover the full range from *20 Hz* to *20 kHz*.
For those who want to skip this (boring) part, where the most important characteristics obtained from each of the submitted tests will be listed, at 3.3.5 a summary will be presented.
It will be very important to keep in mind how the parameters of the models will be treated as if they were absolute values, so that the discussion will be easier.

High Frequency model

Starting from the first filter founded in the channel, 1.1, here some notions about its models will be presented.
The modelled frequencies are between the 12^{th} and 248^{th} index of the frequency vector i.e., from *73.2102 Hz* to *19077 Hz*.
As can be seen from both the tables and the graphics, after the *19 kHz* limit the high frequency filter starts to have a decreasing impact. Most tracks do not develop at these levels, but it cannot be said with certainty whether this component is actually a bandpass (like any high frequency filter, in the end) with a particularly low limit or whether it is due to the fact that the inputs are less and less energetic in that range.
At low frequencies, the the great presence of outliers in the vector of the test result is due to the problem, already mentioned, of very negligible variations effects on the algorithm.

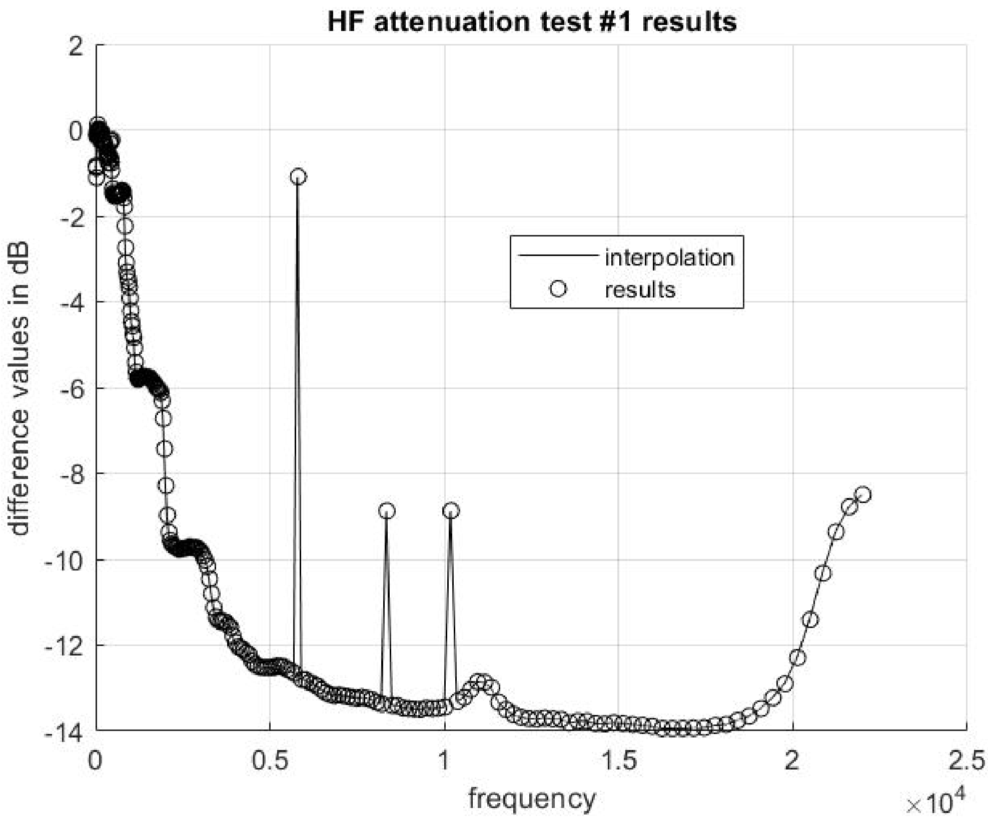


Figure 3.6: The results of a first test of the attenuation at High Frequencies. The performed modification was at the -50% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

The fist test regards an attenuation, at high frequencies, equal to the -50% of level of the filter. In the figure 3.6 the results of the operation are shown, from this is clear the presence of some outliers also in the middle of the derived mask: this circumstance, however recurrent (as often happens), is at the basis of the need to generate several different models. These can be graphically appreciated in 3.7: the limits of all the models are highlighted with the magenta vertical lines. The blue line, representing the *Spline*, is a middle ground as far as the influence of these flawed values on the final result is concerned: clearly the model obtained from the exact test results interpolation is the one that suffers more this situation, while in the *Step* there is no evidence of it.

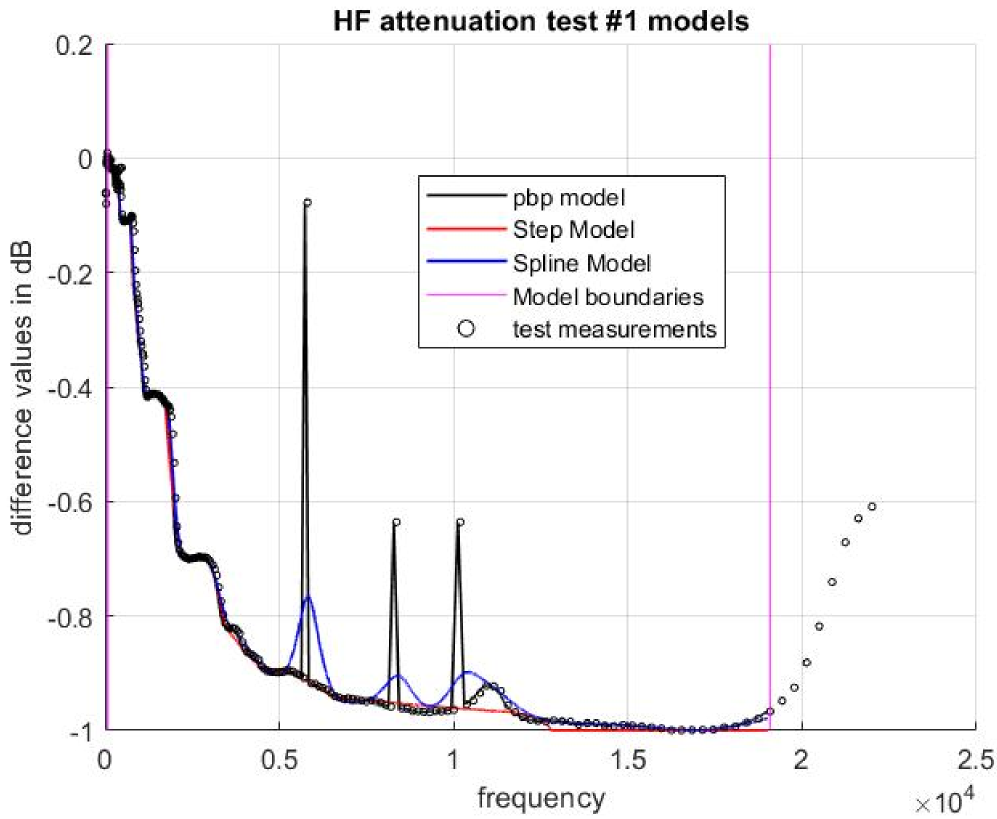


Figure 3.7: The models resulting from the first presented test of the attenuation at High Frequencies (at the -50% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point Model* , the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

The main features to extract are now presented. Being these models normalized with respect to their biggest value, when not specified, the amount relative to the maximum, in absolute terms, will be the unit.
The system's representation obtained directly from the test has its top value at *16539.9 Hz*, where the *Spline* indication is of *-0.9986* and the *Step* model has also a *-1* amount.
The *Spline*'s maximum, this time, does not reach the greatest change worth, therefore its detected biggest value equals to a *-0.9994* at *17141.3 Hz*, where the *Step* model is still

represented by *-1* and the *Point-by-Point* gives a *-0.9993*.
The *Step* , as said, is the only one presenting some *a posteriori* adjustments, so it will need a bit longer explanation: its value is imposed to be equal to *0* from *73.21 Hz*, where *Point-by-Point* model = *0.0110* and the *Spline* is still not evaluated (but at *78.6 Hz* has a value of *0.0018*), until *112.85 Hz* where *Point-by-Point* model = *0.0009* and *Spline* = *-0.0042*. These low frequency evaluation show the presence of positive values, albeit very slightly, where they should be negative. The *Step* model has a greatest modification range imposed from *12877.6 Hz* to *19088.9 Hz*, here *Spline* respectively descend from *-0.9837* to *-0.9827* and the *Point-by-Point* from *-0.9790* to *-0.9660*: these results are an indication of the mitigation of the effect experienced around the highest limit of the human hearing frequency interval.

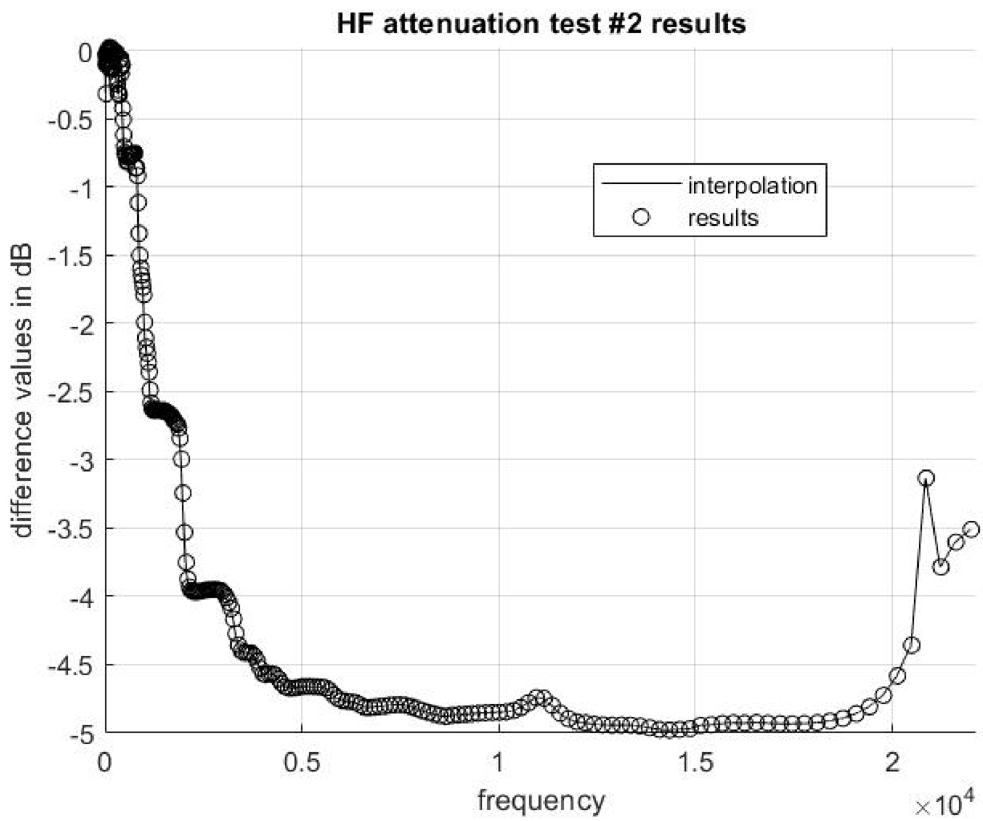


Figure 3.8: The results of a second test of the attenuation at High Frequencies. The performed modification was at the -26% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

The second example of an attenuation at high frequencies test, this time equal to a filter's level of -26%, is displayed in figure 3.8: this time no glaring outliers have been observed, the only one relevant is above *20 kHz* where the models don't work anymore. The models are graphically appreciable in 3.9.

The *Point-by-Point* maximum is at *14335.5 Hz*, where the *Step* model presents the same value (*-1*) and the *Spline* is equal to *-0.9974*.
The *Spline* maximum is *-0.9975* and at *14594.4 Hz*, here the *Point-by-Point* corresponds to *-0.9992* while the *Step* still *-1*.

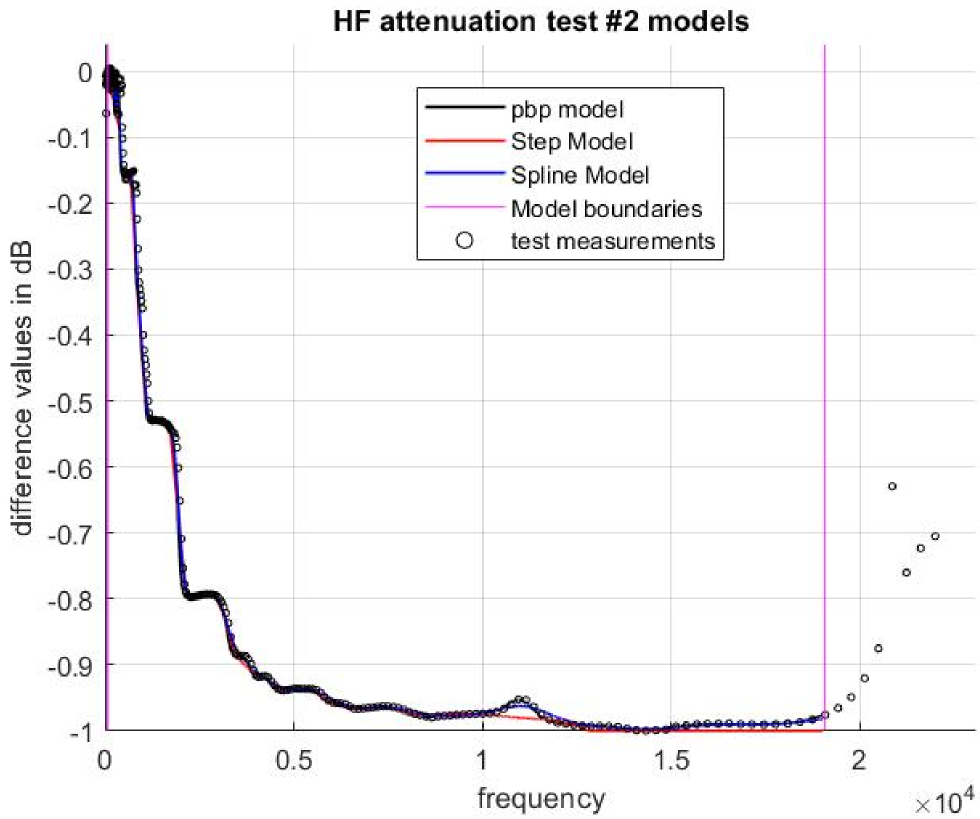


Figure 3.9: The models resulting from the second presented test of the attenuation at High Frequencies (at the -26% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* , the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

Again, for the *Step* , there is an initial string of zeros from the first modelled frequency until *112.85 Hz*, where the *Spline* descends, going in the opposite direction of what one would expect, from *-0.0086* to *-0.0042* and the *Point-by-Point* from *-0.0116* to *-0.0084*.
The maximum plateau is always from *12874.6 Hz*, where the *Spline* = *-0.9915* and the *Point-by-Point* = *-0.9926*, to *19077.9 Hz* where *Spline* = *-0.9825* and *Point-by-Point* = *-0.9760*.
The behavior already mentioned is again revealed.

Now some amplifications will be treated.
The first test regarding a boost in the energy value at high frequencies has been done by

leading to +89% the filter, as showed in 3.10.
The models are graphically represented in 3.11.
The *Point-by-Point* model has its highest value at a frequency of *15676.7 Hz*, while the *Step* is in its maximum plateau and the *Spline* indicates *0.9996*.
The *Step* has always the same imposed values at the already described frequencies: from *73.21 Hz*, where *Step-by-Step = 0.0044* and the *Spline* is still not defined, to *112.85 Hz* to which corresponds a *Point-by-Point* of *0.0050 Spline* of *0.0035*. The maximum interval spans from *12874.6 Hz* to *19077.9* and again a small decrease of the other two parametrizations is appreciated: the *Spline* goes from *0.9938* to *0.9933*, while the *Point-by-Point* from *0.9973* to *0.9925*.

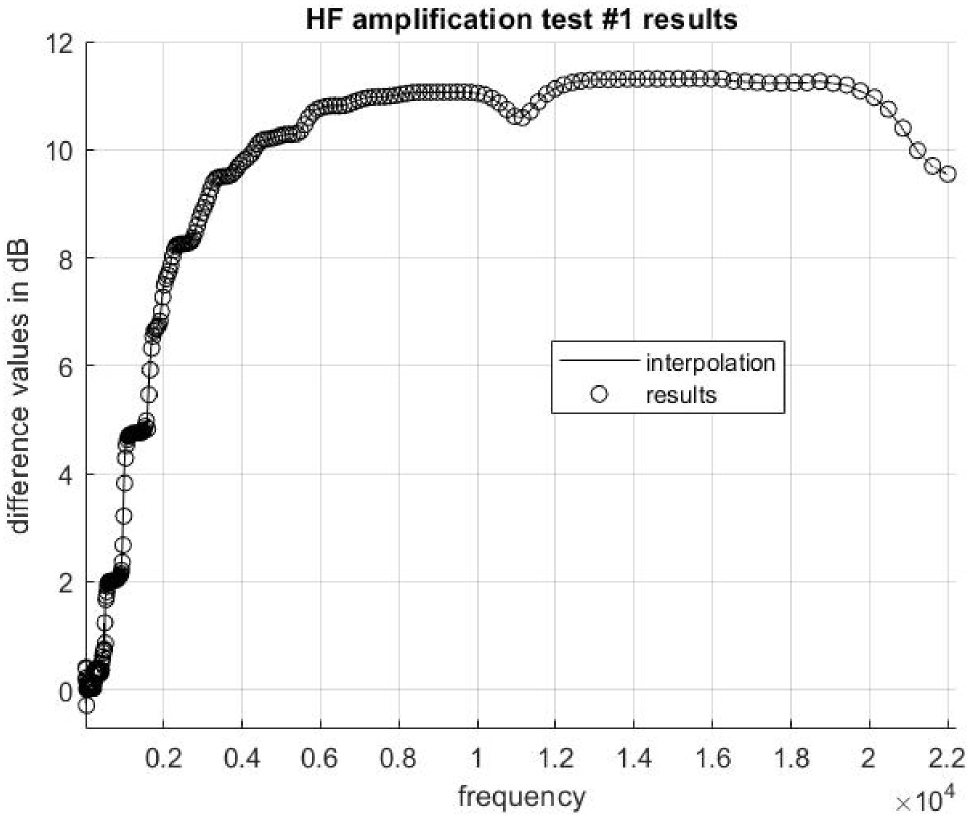


Figure 3.10: The results of a first test of the amplification at High Frequencies. The performed modification was at the +89%. working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

In this case, the *Spline* model is the most interesting one: it presents a triplets of values above the expected maximum i.e., greater than the unit. At *14081.2 Hz* it would multiply the detected maximum amplification for *1.0001*, the same for *14335.5* and *14594.4 Hz*, while the *Step Model* is in its maximum plateau and the *Point-by-Point* swings from *0.9992* to *0.9995*. Again these are all amount which can be rounded to the unit, nevertheless these effects are present and assessed.

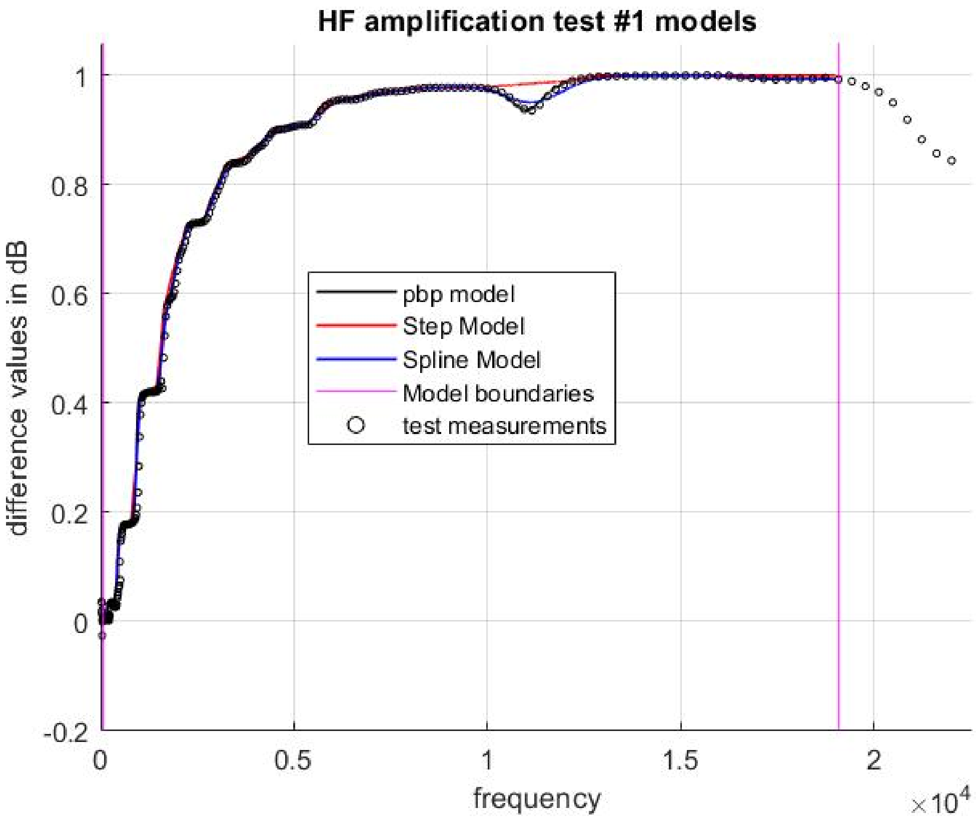


Figure 3.11: The models resulting from the first presented test of the amplification at High Frequencies (at the +89% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* model, the red line the *Step* model, the blue line the *Spline* model while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

The last presented experiment at high frequencies is an amplification of +37%, showed in 3.12. Again some outliers can be detected, improving the graphic evaluation of the various models in 3.13.
In this situation the *Point-by-Point* and *Spline* models have their maximum at *14335.6 Hz*, where the second reaches a value of *0.9996*.
The *Step* model has still its ones i.e., the maximum plateau, from *12874.6 Hz*, where *Point-by-Point = 0.9948* and *Spline = 0.9930*, to *19077.9*, including therefore also the frequency at which the other two models are at their maximum, at which *Point-by-Point* descends at *0.9897* and *Spline* to *0.9879*. The non-modifying zeros interval is again from *73.21* to *112.85 Hz*: here the *Spline* descends from *0.0052* to *0.0050* and *Point-by-Point* from *0.0051* to *0.0039*.

In general, the differences between models are negligible. However, some common trends are visible: the uncertainty in low-frequency measurements and descending behaviour above a certain value at high frequencies are unexpected and always present.
A constant is the frequency's area where the maximum values of the models can be found: this consideration led to declare the "*winner climber*".
In the last presented algorithm, the one at 3.4, the high frequency behaviour will be detected by scanning frequencies from *14000* to *14500* Hz.
The *Step Model* aims to be a first attempt to generate a working representation of the system, which also manages not to be affected by the negative aspects obtained in some tests.

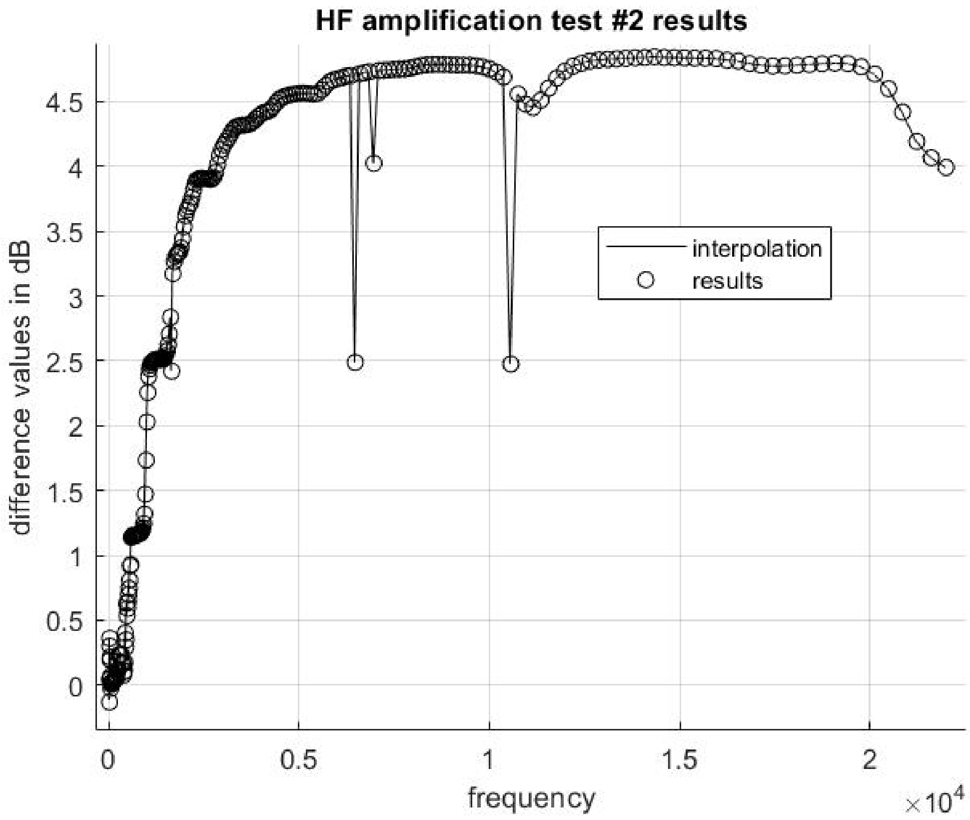


Figure 3.12: The results of a second test of the amplification at High Frequencies. The performed modification was at the +37%. working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

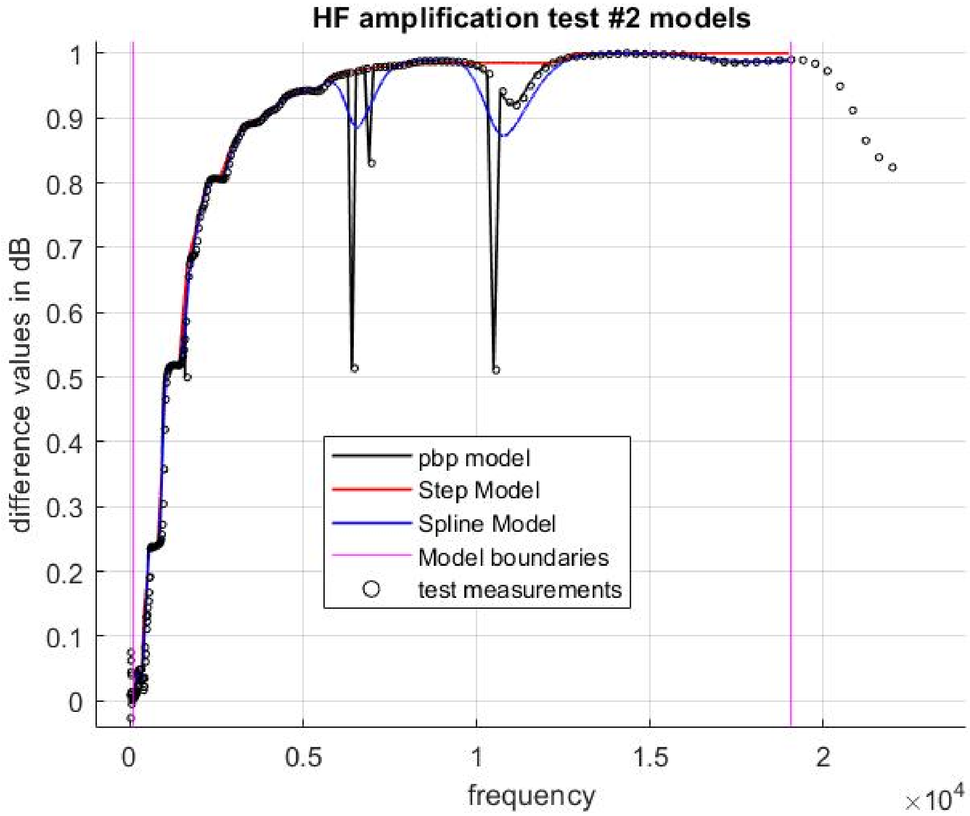


Figure 3.13: The models resulting from the second presented test of the amplification at High Frequencies (at the +37% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* model, the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

Medium Frequency model

Moving to the Medium Frequency *knob*, again two attenuations and two amplifications tests will be presented both in graphical, by means of images, and numerical form.
The modelled frequencies are between the one at 7^{th} and at 241^{th} position of the frequency vector, which correspond to *47.74* to *16838 Hz*. This is due to the continuous presence of faulty measurements around the two, upper and lower, limits of the human audibility range.
In addition to the imposition on the *Step* model, this time there will also be a boundary condition for the *Spline*: this must have a value of zero in correspondence of the last modelled frequency.

The first test, 3.14, represent an attenuation nearby the lowest level to avoid the distortion problem, as described in 3.2.4: +60%. Here, besides some fluctuating values in correspondence of the very low frequencies, there are no obvious unexpected behaviours.

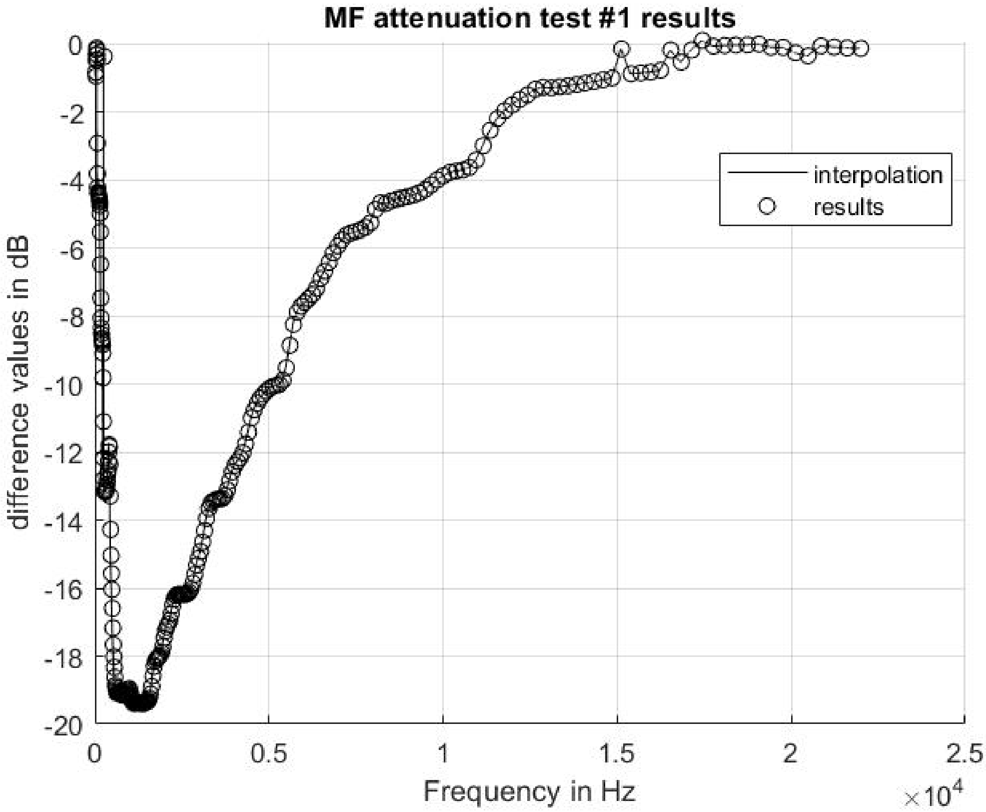


Figure 3.14: The results of a first test of the attenuation at Medium Frequencies. The performed modification was at the -60% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

At *1381.5 Hz* both the *Point-by-Point* and *Spline* maximums are found, while for the first the value is taken for granted, the second one exceeds the unit: from *1325.7* to *1410.1 Hz* it swings from *-1.0001* to *-1.0005*, this last amount is also its maximum. The last model, in these intervals, is always worth *-1*, while the *Point-by-Point* is always around *-0.9992*. The *Step* model will have a value equal to 0 at *47.74 Hz*, where *Spline* = *+0.0021*, while it should have a negative value, and *Point-by-Point* = *-0.0063*. The maximum plateau is now between *671.5 Hz*, at which frequency corresponds a *Spline* equal to *-0.9853* and a *Point-by-Point* of *-0.9838*, and *1795.9 Hz*, where the *Spline Model* is *-0.9321* and the *Point-by-Point -0.9301*.
The overall values can be graphically seen at 3.15.

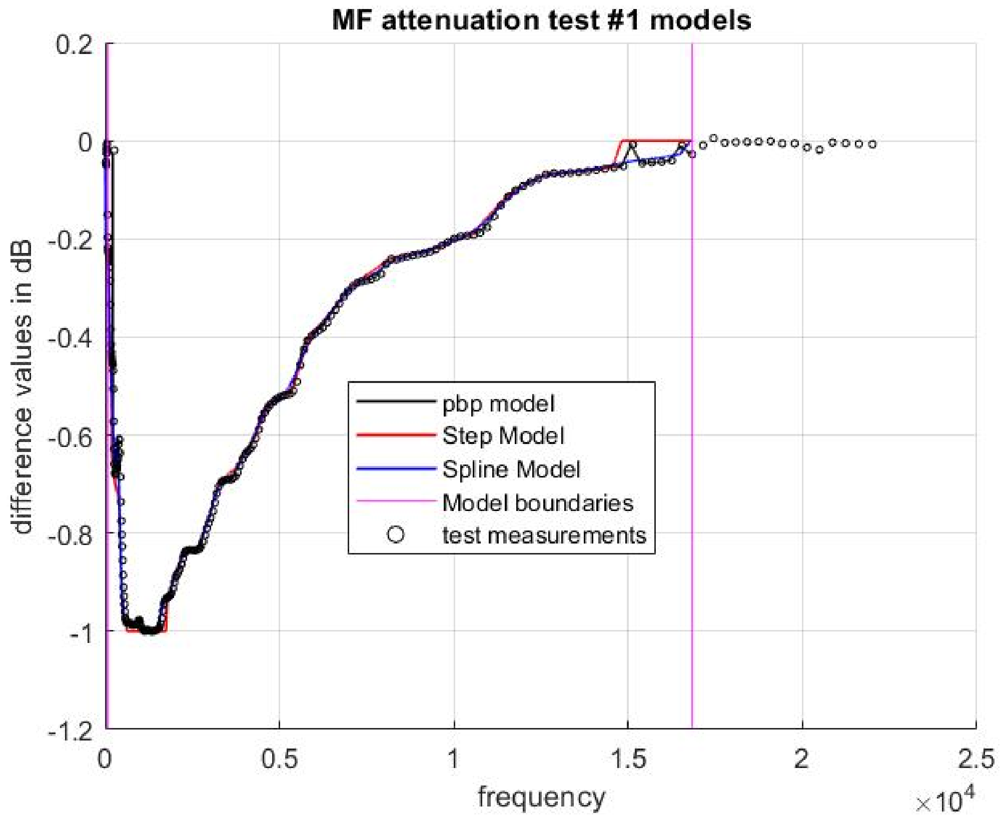


Figure 3.15: The models resulting from the first presented test of the attenuation at Medium Frequencies (at the -60% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* model, the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

The second test i.e., an attenuation corresponding to the -22% working value of the filter, is much more interesting as far as the presence of outliers is concerned, which make the differences between the models more and more evident. The tests and models can be seen in 3.16 and 3.17.
This time the faulty measurements strongly affects also the maximum working range. The *Point-by-Point* maximum is in the ones interval of the *Step* model, at *845.1 Hz*, but at this same frequency the *Spline* is equal just to *-0.798* and this is due to a series of three strong outliers: from *771.9* to *807.8 Hz* the test results spans from *-0.1380* to *-0.276* while they should be around the maximum value. Since the *Spline* approximates the vector containing the result of the experiments, this occurrence particularly penalizes it. As if that weren't enough, a fourth outlier happens shortly after, lowering again the values of this approximation: the *Spline* reaches a maximum of *-0.9760* only at *1561 Hz*, long after the test's one but still, for short amount, in the maximum modification region of the *Step*

Model.
The *Step* model has still a first value imposed to 0, while *Spline* is *0.0140*, again of opposite sign than it should be, and *Point-by-Point* = *-0.0298*. Also the ones plateau is still from *671.5* to *1795.9 Hz*, and here the *Point-by-Point* model follows a parable between *-0.9689* and *-0.9524*, while the *Spline* from *-0.9705* and *-0.9524*.

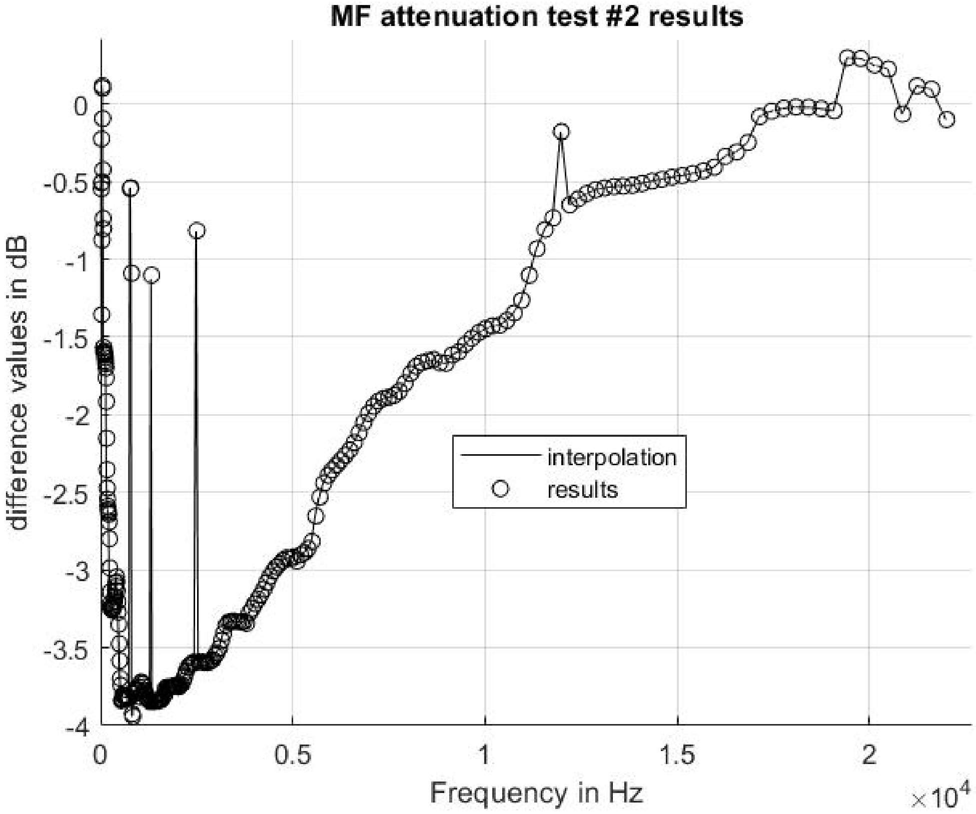


Figure 3.16: The results of a second test of the attenuation at Medium Frequencies. The performed modification was at the -22% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

A key aspect regards its final zeros vector: now it starts later than before i.e., at *15676.7 Hz* where *Spline* = *-0.1050* and *Point-by-Point* is *-0.1090*, and finishes at *16837.9 Hz* where the Spline is constrained to be 0 and the *Point-by-Point* model identifies a value of *-0.0627*. This shift should not come as a surprise, as at 3.2 has been already demonstrated how these are non-linear components, therefore their operating range may vary according to the set percentage value.
A final consideration can be spent on values beyond the model's limits: easily appreciable is a concentration of outliers around *20 kHz*, with also positive amounts while everything should be less than 0.

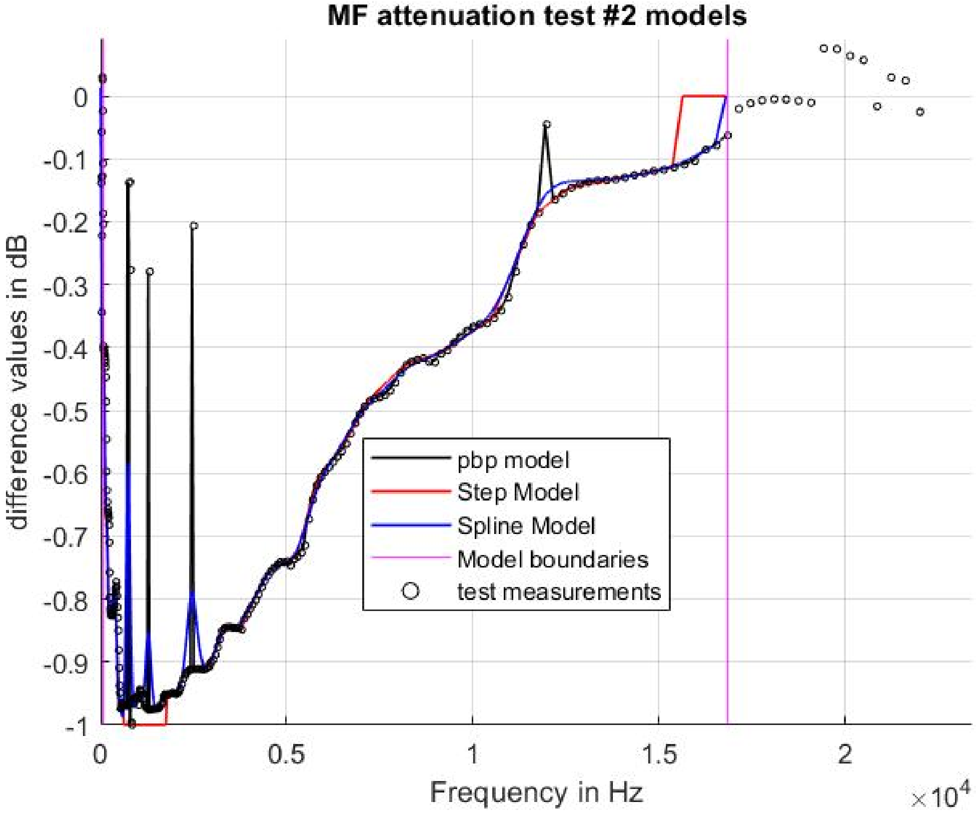


Figure 3.17: The models resulting from the second presented test of the attenuation at Medium Frequencies (at the -22% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point Model*, the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

The first amplification test is useful because, finally, it shows (at 3.19) a positive aspect of the *Spline* model compared to the *Step*.
The test results are graphically appreciable at 3.18: it is immediately apparent that above 10kHz there is a highly unexpected trend. First there is a sort of ripple in the data, followed by a surge at the high frequencies, when there should be an attenuation of the filter effect. A last drawback of the presented experiments are the two clear outliers, one of which in what should be the maximum working range and the second even more influential, being immediately after a downward curve.
The most important numerical values are again the maximum of the *Point-by-Point* model, founded at *1561 Hz*, where the *Step* is in the maximum plateau and the *Spline = 0.9901*, and this last one maximum at *1271.9 Hz*, where it is equal to *0.9964* while the *Step* is still in its ones range and the *Point-by-Point* model reaches *0.9942*.
Regarding the *Step* model: it has still a first imposed value of 0 at *47.74 Hz*, while the

Spline is *0.0683* and *Point-by-Point 0.0602*. Its maximum modification interval is still between *671.5* and *1795.9 Hz*, where the *Spline* rises from *0.9933* to its maximum, to then descend until *0.9760* and the *Point-by-Point* model follows a similar path but starting from *0.9894* and arriving to *0.9851*. The high frequency zeros stream, now starts from *16247 Hz*, in correspondence of which the *Spline* value is *0.0629* and the *Point-by-Point* = *0.0663*, to *16837.9 Hz* where the *Spline* has the imposed 0 value and the *Point-by-Point* is at *0.0465*. Again the starting point of this last series changes with respect to the ones already presented.

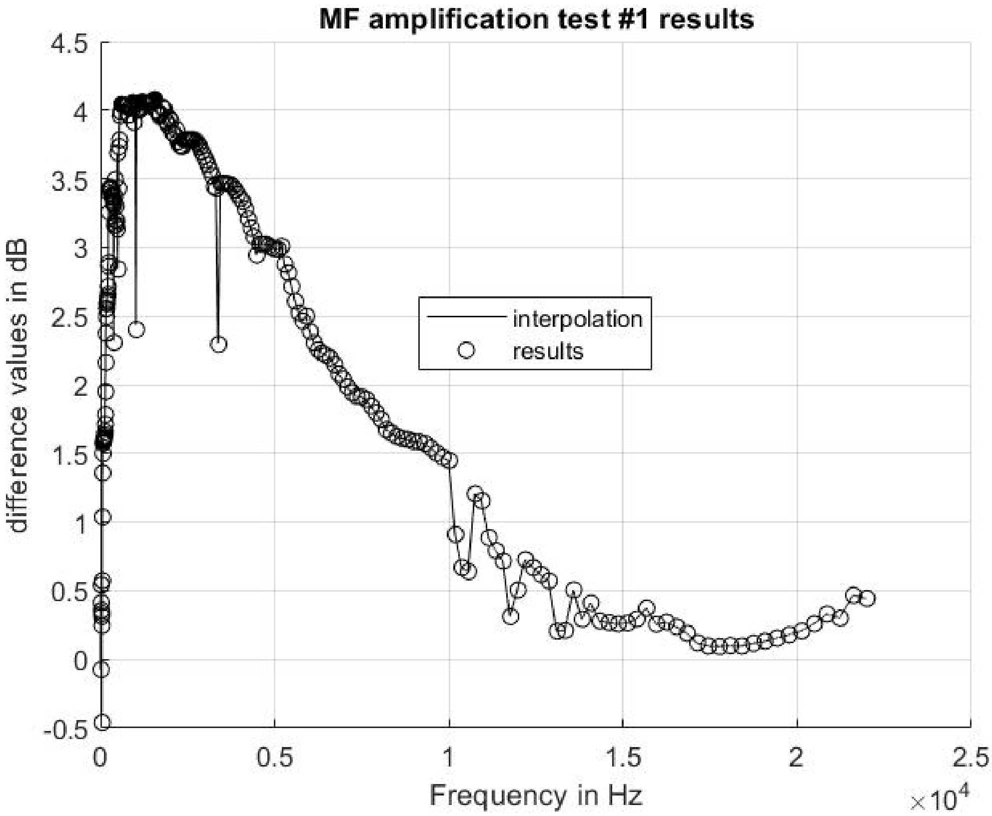


Figure 3.18: The results of a first test of the amplification at Medium Frequencies. The performed modification was at the +30% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

What needs to be highlighted is how the *Spline* model can manage to describe an expected pattern of operation from *10 kHz* to the models' upper limit, while the *Step* is highly influenced by these kind of "jumps" problem in the results, as is obviously also the *Point-by-Point*. It can therefore be seen that models that are more robust than certain types of errors are weaker on others and vice versa.

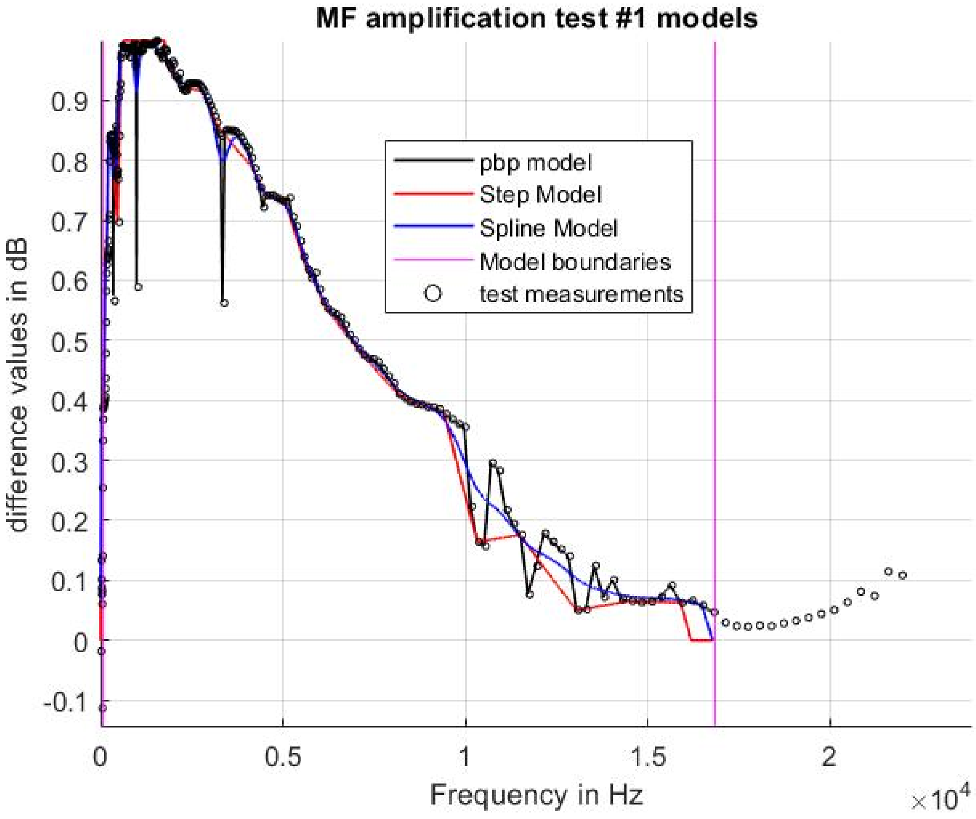


Figure 3.19: The models resulting from the first presented test of the amplification at High Frequencies (at the +30% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* model, the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

A final clarification regards the unmodelled part, after *16247 Hz*: this change of gradient seems to be simply a sharper trend than the problems already found at the same frequencies in the attenuation.

The last medium frequencies test is a +75% amplification, 3.20. With this, some characteristics similar to those obtained from the previous one are shown again i.e., the final ascending interval and the *Step* model weakness. While from an outliers point of view, this time everything seems to be smooth at the greatest work frequencies.
Always following the usual procedure, the *Point-by-Point* maximum is situated at *1051.8 Hz*, where the *Spline* = *0.9917* and the *Step* is in its maximum plateau.
The *Spline* model has its maximum, equal to *0.9922*, at *1097.8 Hz*, where *Point-by-Point* = *0.9921* and *Step* is unitary.

The *Step* model still starts with a null value, while *Spline* and *Point-by-Point* have respectively a value of *0.0123* and *0.0120*. It also presents the usual maximum interval,

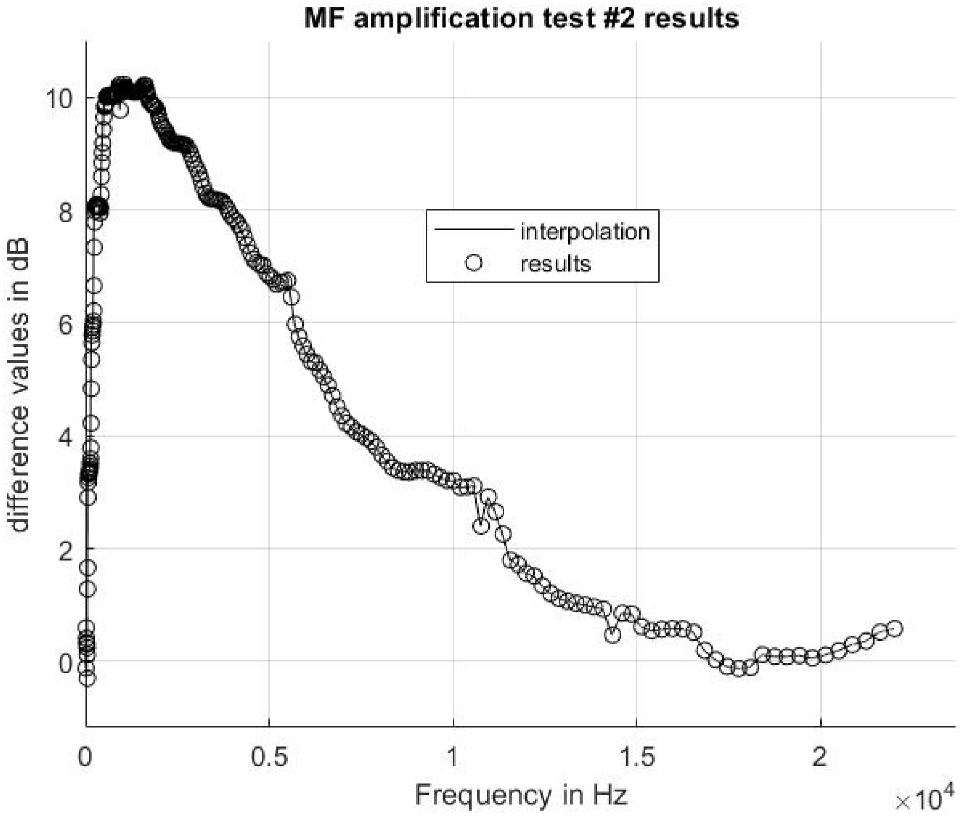


Figure 3.20: The results of a second test of the amplification at Medium Frequencies. The performed modification was at the +75% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

with the same frequencies as before, where the *Spline* moves from *0.9799* to *0.9688* and the *Point-by-Point* from *0.9796* to *0.9653*.
The most interesting aspect is to notice how now, since the modification made is much greater than the previous amplification, the final range of zeros has moved towards the value obtained from the first attenuation test, which was at -60% (so again a large amount). This last subset of the models spans from *14594.4 Hz*, at which the *Spline* indicates a *0.07345* and the *Point-by-Point* a *0.0840*, to the last frequency limit of these models i.e., *16837.9 Hz*, where *Point-by-Point = 0.0195* and the *Spline* is imposed to be zero.
As already stated, it is possible to appreciate how the *Spline* seems to fit better the last decade of this filter's mask, managing to minimize the problems due to its shape.

While before the *Step* appeared to be the best model, the last two showed sample experiments have proven that this is not true, justifying the presence of several descriptions of the system. This last evaluation can be done graphically at 3.21.
With a similar consideration to the one made for the previously exposed filter, the medium

frequency's "*winning climber*" covers an area of 200 Hz around 1500 Hz.

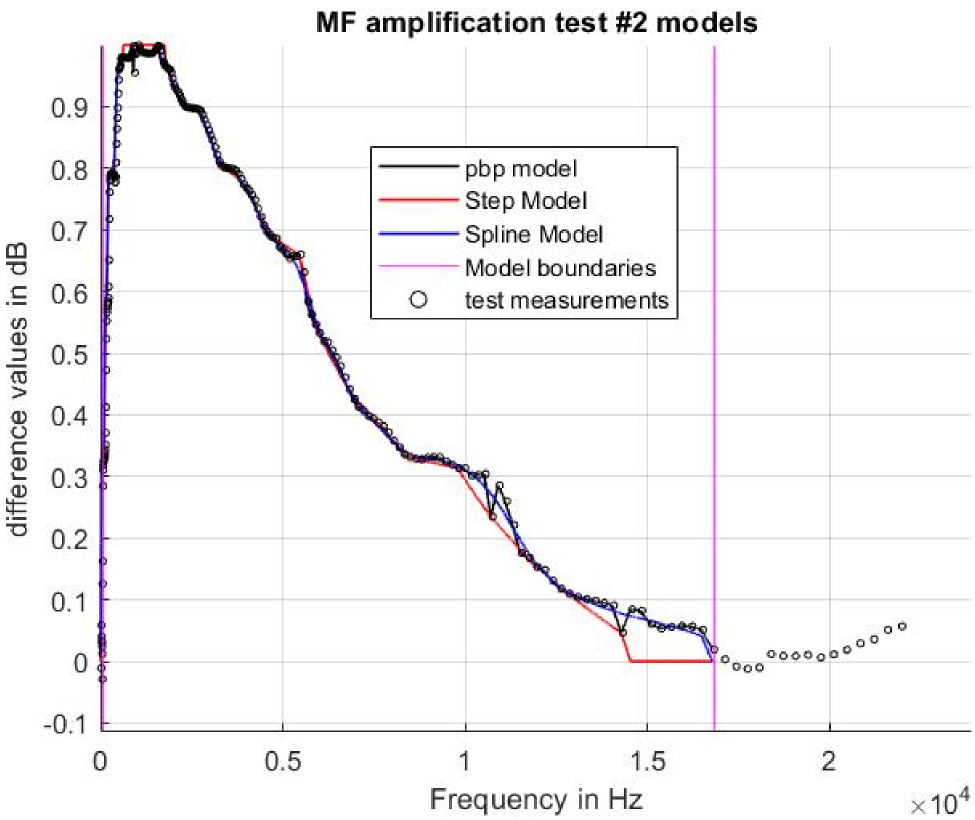


Figure 3.21: The models resulting from the 9 second presented test of the amplification at High Frequencies (at the +75% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* model, the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

Low Frequency model

Here the last frequencies interval will be studied.
Unfortunately the ERB scale does not allow a simple graphical evaluation at first impact, it will be necessary to pay a moment of attention to the positions of the axes to avoid thinking you are back to the high frequencies section.
The models ground idea does not change: an interval between the 9^{th} and 216^{th} value of the frequency vector, or from *57.66* to *10758* Hz, will be the range represented by these models.
While the *Spline* model presents again a constraint at its higher limit i.e., to end with a zero, the *Step* model can now be seen as the mirror of the one for high frequencies: it starts with the maximum modification plateau and finishes with the zeros stream.
Again four sample test will be discussed: two attenuations and two amplifications. The issues remain the same: the change of direction at high frequencies and the low frequency initial outliers. This time, the problem of wrong evaluation is clearly the opposite of the high frequency examples: the very small variations are experienced at frequencies above *12 kHz*.

The first presented example regards an attenuation at the limits of the elbow of maximum non-linearity of the filter characteristic: -58%. This case highlights a strong uncertainty in the results, as graphically appreciable in 3.22: at high frequencies one can see a clear divergence from what should be the expected value i.e., zero.
At 3.23 the models are compared with the test results, normalized, of course.
The *Point-by-Point* maximum can be found at *89.6 Hz*. The *Spline* maximum does not reach the unit, it is instead *-0.9987* at *84.04 Hz*.
Regarding the *Step Model*: it starts with the greatest modification plateau from *57.66 Hz*, where *Spline* = *-0.9795* and *Point-by-Point* = *-0.9760*, and finished at *106.88 Hz* where the other two models, after having reached their highest value, descends respectively to *-0.9790* and *-0.9895*. While from *8356.6* to *10758.1 Hz* the *Step* model presents the zeros vector: in this interval the *Point-by-Point* ascends from *-0.0019* to *-0.0942*, instead of moving downwards, while the *Spline* is constrained to get to zero at the model's limit, starting from *-0.0031*.

The second presented low frequency attenuation test was done by moving the relative *Knob* until the -32% position, the graphical representation of the results vector is shown in 3.24.
The relative models, instead, are pictured at 3.25.
This second example, even if the behaviour at high frequencies seems to be stable for a wider interval, will be useful to highlight other interesting cases.
Both the *Point-by-Point* and *Spline* have their maximum in the maximum amplification plateau of the *Step Model*, in particular at *84.04 Hz*, where the *Spline* = *-0.9991*. This correspondence is quite common when changes do not reach high levels, as already explained the *Spline* is a cubic interpolation of the test results: if the output vector is smooth then it can be more easily followed.

The values to be highlighted concern all the *Step* model: the initial maximum modification plateau ends now earlier than the previous case i.e., at *101.01 Hz* where *Spline* = *-0.992* and *Point-by-Point* = *-0.9958*. As the discussion in 3.2 has shown, the components have nonlinear behaviour, resulting also in masks related to the modification level.

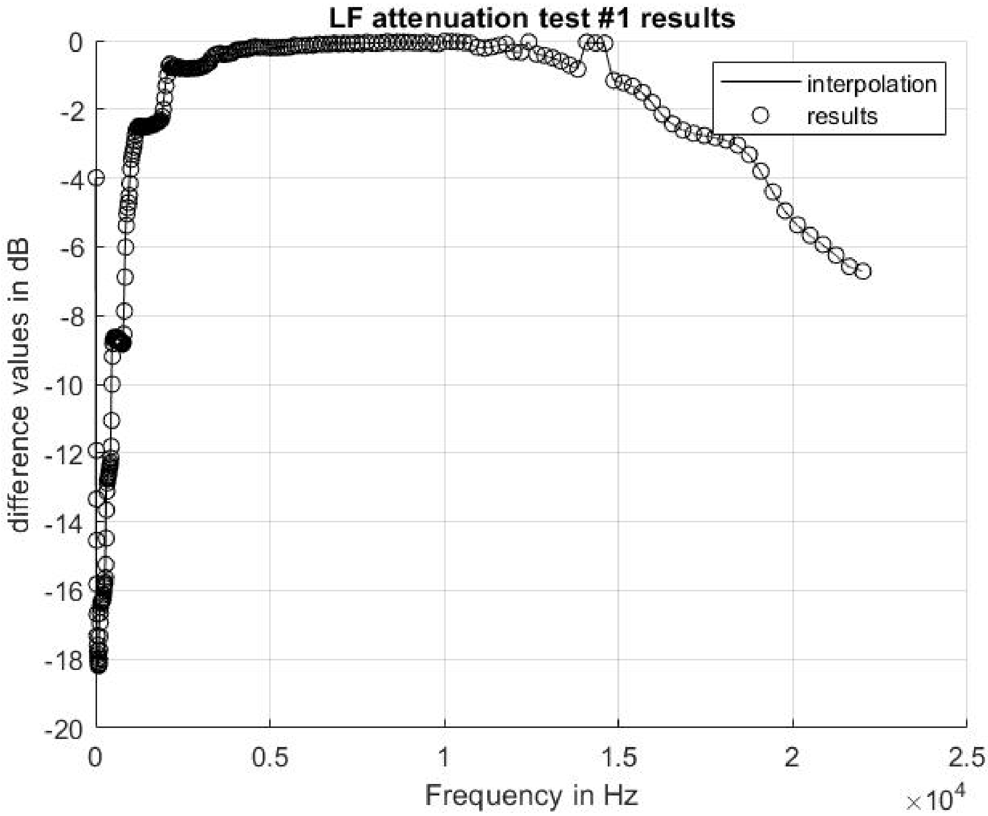


Figure 3.22: The results of a first test of the attenuation at Low Frequencies. The performed modification was at the -58% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

This is also appreciable from a shifted start of this zeros plateau: now is at *9313.8 Hz*, where the *Point-by-Point* model has a value of *-0.0039* and the *Spline* presents a positive normalized amount of *0.0012*, again against what it should be. The end of this final interval is imposed, so at the model's higher limit the *Point-by-Point* is equal to *-0.0010* and the *Spline* must meet its only constraint i.e., is equal to zero.

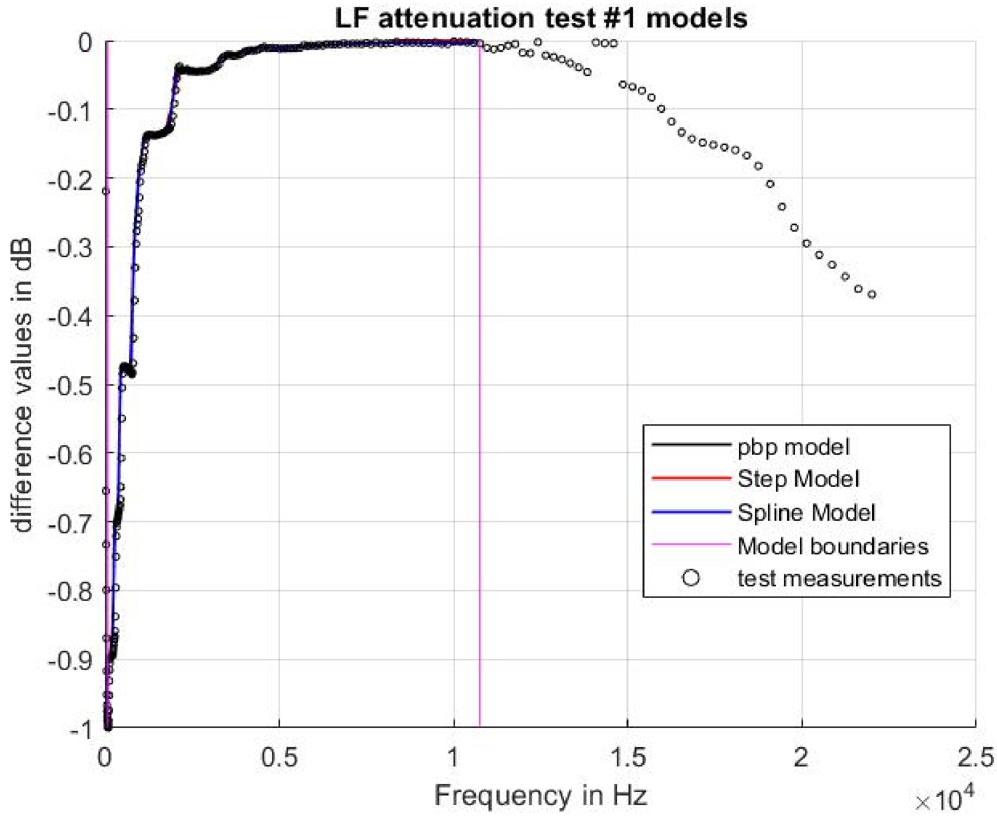


Figure 3.23: The models resulting from the first presented test of the attenuation at Low Frequencies (at the -58% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* model, the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

The first amplification is a +70% one. Except for the high-frequency soaring, it seems to have no particular case history, as graphically displayed in 3.26. Also the models here are broadly similar, as in fact the *Spline* and *Point-by-Point* maximums are both at *84.04 Hz*, where the first one reaches *0.9996*. All the numerical values can be graphically appreciated by means of 3.27.
The *Step* model always begins with the maximum plateau, starts at the first modelled frequency *57.66 Hz*, at which *Spline = 0.9923* and *Point-by-Point = 0.9913*, until *101.02 Hz*, where the *Spline* registers a *0.9934* and *Point-by-Point = 0.9968*.
The *Point-by-Point* model represents the unexpected rise of values in correspondence of the *Step* high frequency zeros interval: from *10010.58* to *10758.1 Hz* it goes from *0.0061* to *0.0263*. The *Spline* model, instead, descends from *0.0083* to its imposed null value.

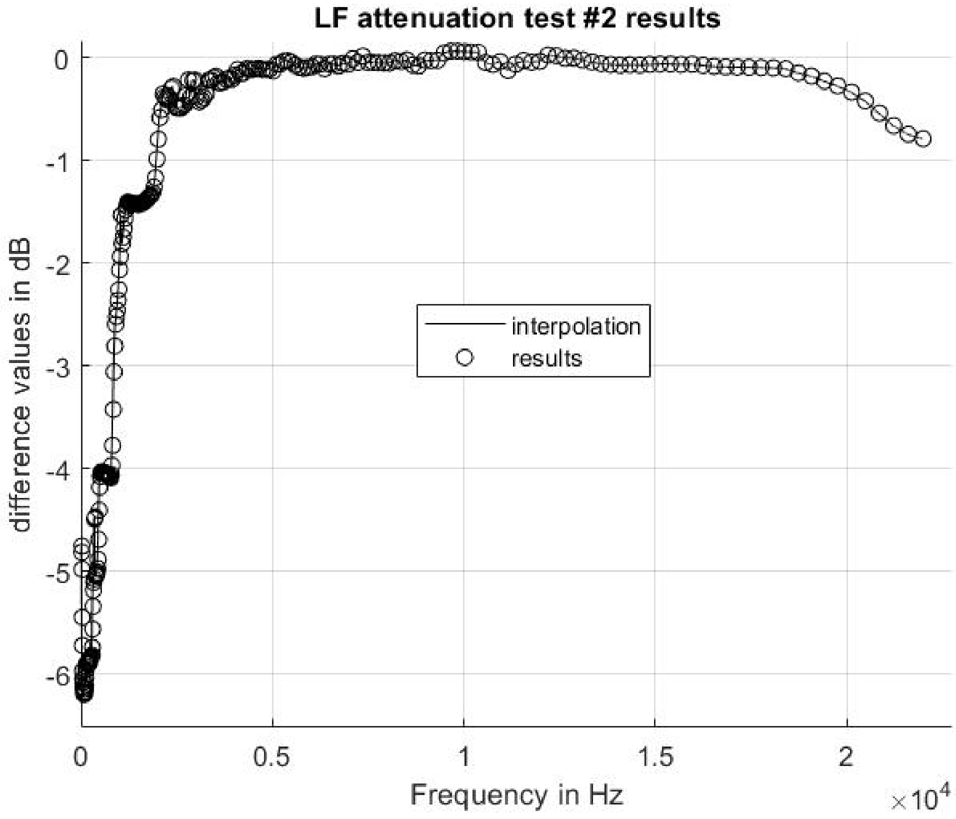


Figure 3.24: The results of a second test of the attenuation at Low Frequencies. The performed modification was at the -32% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

The last presented case is a low frequency amplification performed with the *knob* moved to +19%.
This test will be useful both to point out again how the masks are affected by the value of the change and to show another situation where the *Step Model* failed to fully represent the real operation. The well-known change of slope can be evaluated graphically at 3.28. As usual, the models are represented at 3.29.
The *Step* model initial ones stream spans from *57.66* to *78.57 Hz*, showing again the non-linearity of the components, where both *Spline* and *Point-by-Point* rise respectively from *0.9653* to *0.9737* and from *0.9656* to *0.9716*: the monotonicity of these two other models is explained by the fact that their maximum is outside this range, which is designed to always contain them.

The *Spline* has its maximum at *112.85 Hz*, where *Step* = *0.9707* and *Point-by-Point* = *0.9875*.

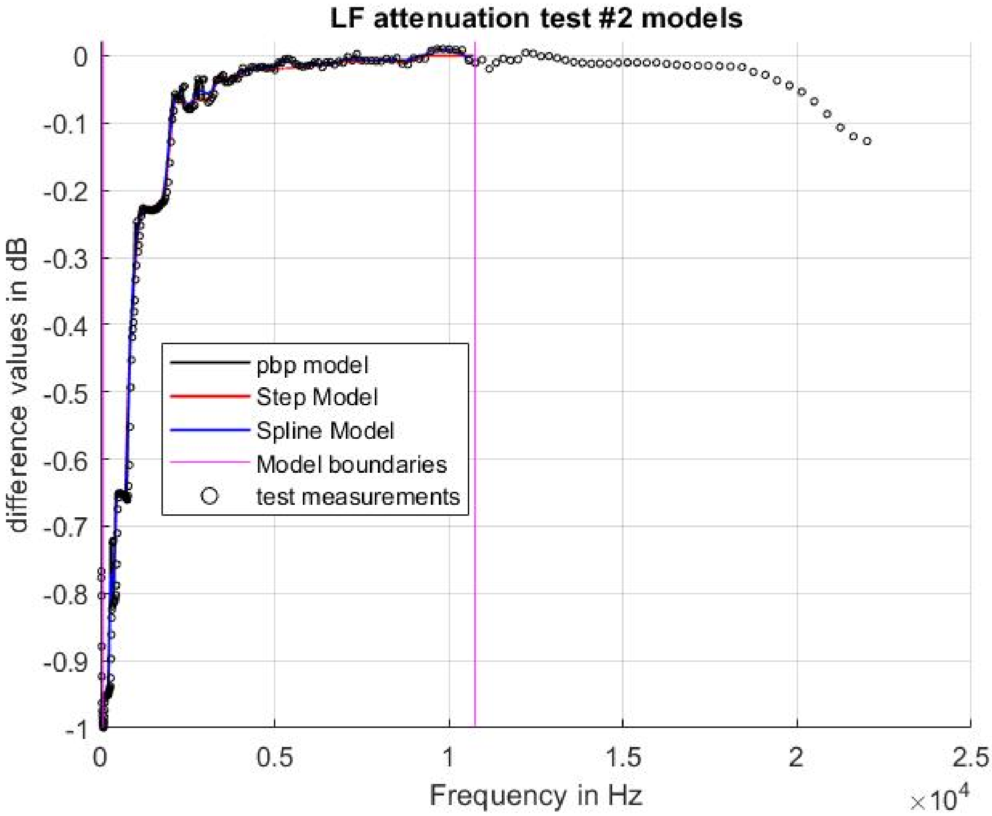


Figure 3.25: The models resulting from the second presented test of the attenuation at Low Frequencies (at the -32% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* model, the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

The *Point-by-Point* model reaches the unitary value at *125.1 Hz*, there the *Step* registers a *0.9679* and the *Spline* a *0.9825*.
The final non-modification interval of the *Step* model starts at *10377.71 Hz*, where *Spline = 0.0087* and *Point-by-Point = 0.0062*, and finishes at *10758.09 Hz*: here the *Spline* has a forced value of zero, but the *Point-by-Point* again shows a rise to *0.0199*, while it should be decreasing.

Again, the need for more models for filter masks has been confirmed.
The low frequency "*winning climber*" will span from 100 to 130 Hz.

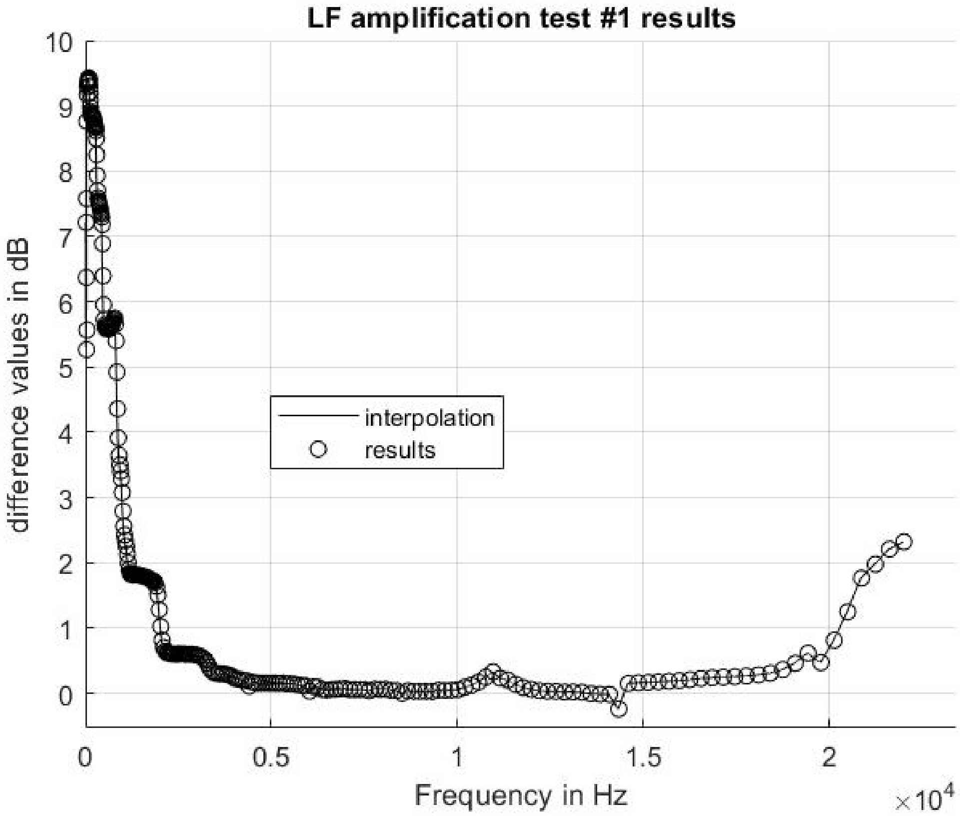


Figure 3.26: The results of a first test of the amplification at Low Frequencies. The performed modification was at the +70% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

3.3.5 Filter Models Conclusions

The presented tests was just a small part of the whole work used to create the masks: the total number of experiments was, in fact, 98. Such an high amount of data was needed because each model resulting from a given modification, has been linearly interpolated with the one due to the recorded successive one: in order to use a linear operation, the differences between two successive operations had to be small enough.
In any case, the few examples have been selected because they cover the whole range of concerns encountered in this phase and have clarified the aspects that now are going to be summarized.
The main problems, as far as unexpected behaviour is concerned, have always occurred at the limits of the human auditory range, which also correspond to the extremes where filters have less influence. This issue has made it mandatory to define limit frequencies for all models of each component, both at low and high frequencies.
Clearly the tests were limited to areas where the filter characteristic could be obtained, as explained is 3.2. Unfortunately, the signal analysis showed some vulnerabilities when

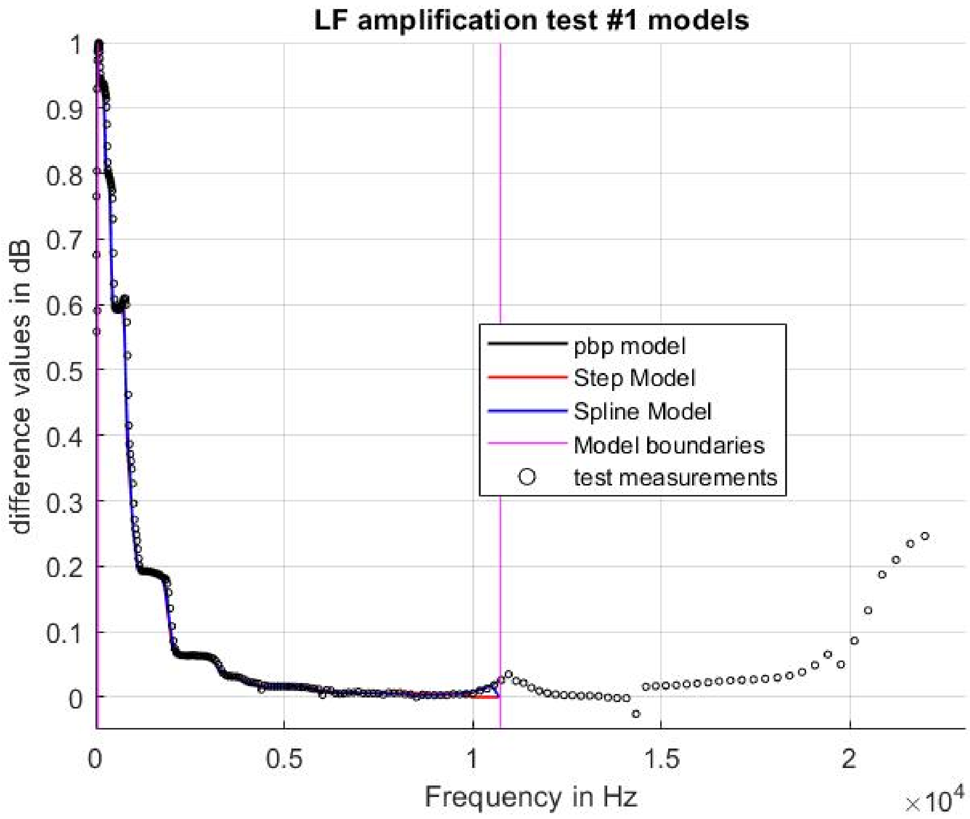


Figure 3.27: The models resulting from the first presente9 test of the amplification at Low Frequencies (at the +70% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* model, the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

changes were made at too low values, so the decision was made to narrow the field also in terms of modelled operations. The most important features, for each frequencies interval, are:

- The high frequency evaluation of new signals will be performed from 14000 to 14500 Hz i.e., the zone where this filter has shown the highest modification values.
- The medium frequency evaluation of new signals will be performed from 1490 to 1690 Hz.
- The low frequency evaluation of new signals will be performed from 100 to 130 Hz.

As expected, there is about a decade between the filters' maximum working intervals.
The following table 3.3.5 will show the operating ranges of the models, relative to each filter's operation.

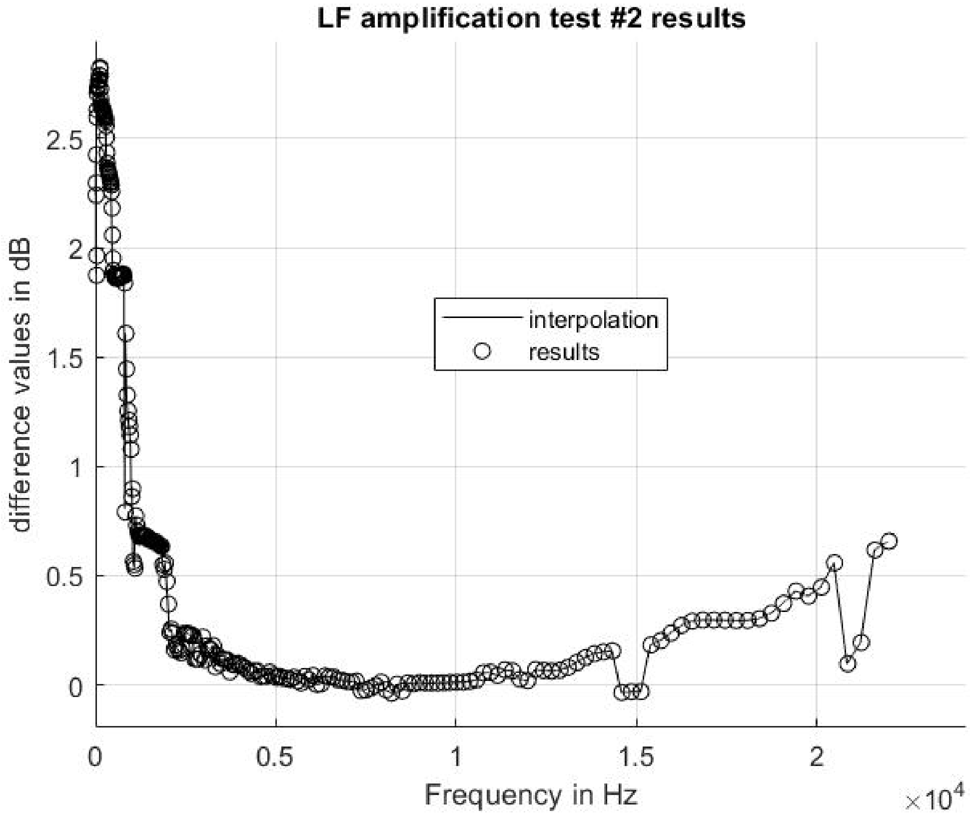


Figure 3.28: The results of a second test of the amplification at Low Frequencies. The performed modification was at the +19% working point of the filter. The empty dots indicates the result of the test, the continuous line the linear interpolation of these values. The frequencies are disposed on the ERB scale 2.3.2.

Performed Modification	Frequency Limits [Hz]	Modification Limits
High frequency Attenuation	73.21 to 19077.89 Hz	-10% to -64%
High frequency Amplification	73.21 to 19077.89 Hz	10% to 100%
Medium frequency Attenuation	47.74 to 16837.9 Hz	-10% to -60%
Medium frequency Amplification	47.74 to 16837.9 Hz	10% to 100%
Low frequency Attenuation	57.66 to 10758.09 Hz	-16% to -58%
Low frequency Amplification	57.66 to 10758.09 Hz	10% to 100%

It has also been demonstrated how each model has its strengths and weaknesses: the *Step* model can't be seen as reliable when the results of the analysis present a kind of ripple; while the *Spline* model is too influenced by the occurence of outliers. This makes it clear how the generation of multiple representations has been useful.
The optimization process, in order to apply the right model, will be presented in 3.4.2.

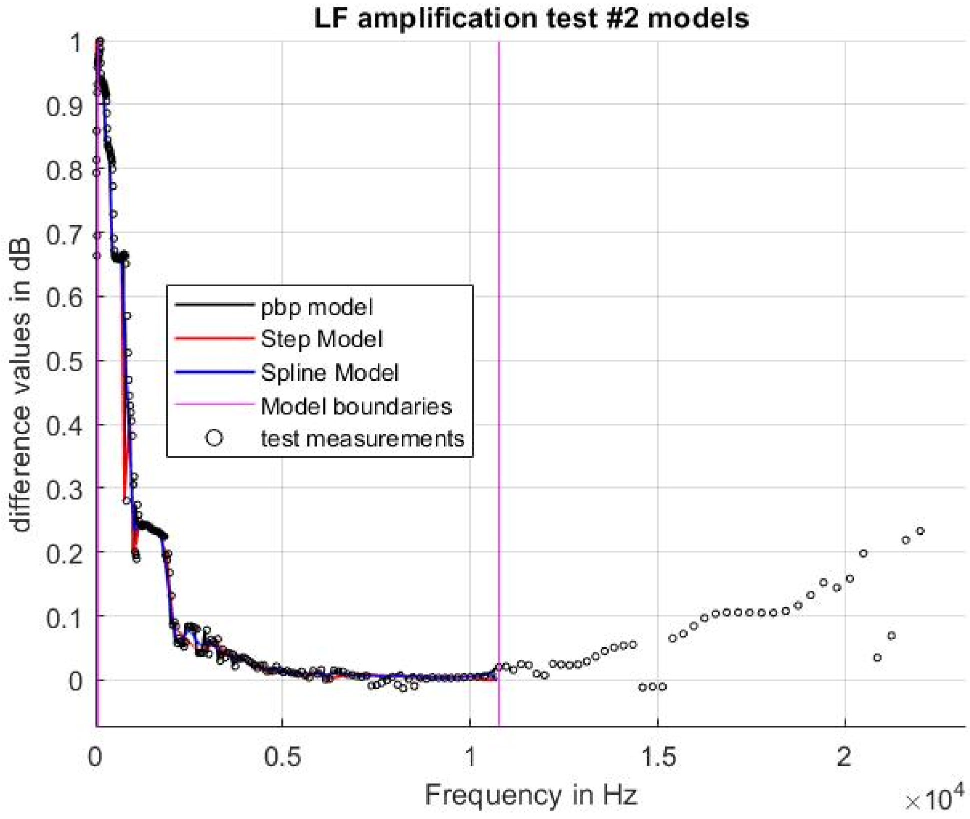


Figure 3.29: The models resulting from the second presented test of the amplification at Low Frequencies (at the +19% working point of the filter). The empty dots indicates the result of the test, the black line the linear interpolation of these values i.e., the *Point-by-Point* model, the red line the *Step Model*, the blue line the *Spline Model* while the two magenta vertical lines the models' boundaries. Everything is normalized, as explained in 3.3.4. The frequencies are disposed on the ERB scale 2.3.2.

3.4 Evaluation

The last implemented algorithm is a recursive analysis of a signal stream: by means of given length sampling, information about the modification of the system's states can be obtained.
The workflow develops as follows: first the "climbers" are created. Then the audio files related to the modified signal and its reference are acquired, pre-processed and finally aligned, as explained in 2.3.1. The time-frequency representations are hence obtained, and the climbers are ready to cross it: the imposed high level sampling time is one second i.e., each second the algorithm explained at 2.3.2 is called and its output is collected to be analysed.
First the results of the three "*winner climbers*" are computed. Then the complete spectrum is scanned, in order to evaluate the eventual offset level i.e., the constant difference in the

volumes between reference and modified signals.
These two operations allow to extract the required features: position of each *Knob* in terms of % and time indications of when the eventual modification has started and/or finished. From these it is also possible to select the right model to be applied in order to modify the reference signal: this recursive process is clearly essential, since it allows to have a constant update of the values with which to compare the modified track.

3.4.1 Signal Analysis for the Recursive Evaluation

Here the only feature in terms of energy variation to be extracted is the difference between the average value of the *pre modification* and *post modification* interval of the time series resulting in the comparison between reference and modified tunes. The time indication of the change is always registered.
The analysis starts at the three specific frequencies that correspond to the maximum working area of each filter: 100-130 Hz for the low frequency, 1490-1690 Hz for the medium frequency and 14000-14500 for the high frequency. The assessments are carried out in parallel, and the results of all three are then compared with each other: the particle which will give the greatest modification value, in absolute terms, will be selected as the "most climbing" one. This indication will led to the collection only of the most meaningful data (in terms of modulation time and amount) and a variable used to indicate to the code which *knob* is the one responsible for the variation, will be switched to its correct setting.
It is important to underline that the algorithm indicated in 2.3.3 is not affected by the constant offset between track and reference: this in fact works on the relative confront of the difference between modified and reference signal before and after the modulation, a constant bias does not influence the measurement.
Once this operation is finished, thanks to the indication on when the modification will start, an evaluation over the entire frequency spectrum is performed to determine the total difference between the volumes of the reference track, saved in a database, and the modified one i.e., the offset. This value is obtained searching for the common energy differences between all the frequency intervals before the start of the change, and can even not be developed on the whole spectrum, speeding up the analysis: a good trade-off has been identified in an examination from 137.79 to 10566 Hz.
In picture 3.30 an example of offset extrapolation for an amplification at low frequencies of +82% is shown.
As it is possible to see, the offset value swings inside a very restricted interval i.e., a 0.14 dB one: from this first investigation its amount is set equal to -7.76 dB.
Here just the time indication of the 2.3.3 algorithm has been used, the next paragraph will present how the evaluation on the other axis is employed.

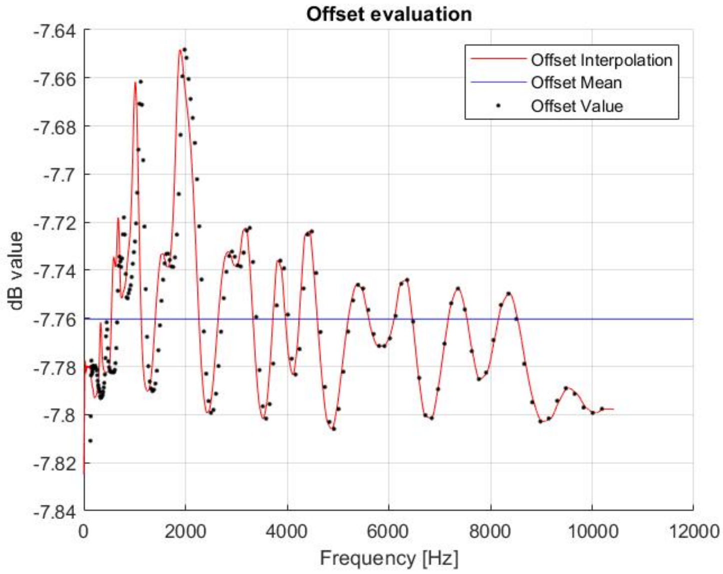


Figure 3.30: The offset evaluation from 137.79 to 10566 Hz. Each black dot represents the value obtained by each climber at its own frequency, the red line is their interpolation while the blue line is the average i.e., the value used to correct the modified signal with respect to the one in the database.

3.4.2 Modification Evaluation and Model Application

The model application starts from the analysis presented in 3.4.1: as said both an indication on which is the most active frequency interval and the value of the maximum modification in dB at the selected interval are obtained.
First the measured change amount is associated to a percentage value thanks to the characteristics obtained in 3.2: as said, to each position of the filter is connected a pair of values, indicating the trails of the possible delivered modulation for each *knob*. Starting from the bottom of the relative characteristic, the feature in terms of energy is linked to both the elements of the couple representing each % of the component's level through the euclidean distance calculation: the pair that minimizes this parameter is the one indicating the filter's state.
Once that this index has been obtained, it is used to browse inside the libraries containing the models, and permits the extraction of the right triplets of mask's representation. As presented in 3.3.3 there are three normalized models, stored in some look up tables with their respective absolute values, for each state of the system: these are multiplied by the evaluated modification amount generating a more reliable representation of the mixing operation.
Now the optimization comes into action: for each discrete frequency obtained through the ERB representation, the error between model and real value (achieved from the analysis of the modified audio signal) is calculated. With this computation the optimal model for each frequency interval is detected and the final optimal representation will be given by the linear interpolation of these optimal values. This approach is very useful to eliminate

the problems related to single models discussed in 3.3.4.

3.4.3 Recursiveness

The idea is quite simple: the results of the previous sample computation, are used for the next one. As said the update of the reference track, by means of an optimal model of the mixer, is used to be the new reference.

3.4.4 Results

The results will be presented both in terms of time necessary to perform a complete analysis and differences between the computed models and the real signals.
Regarding the time needed for the overall computation:

- the creation of all climbers always takes about one fifth to one third of a second, and it is an operation which only needs to be done once, at the beginning of the analysis.
- The preprocessing lasts between 1.2 and 2 seconds, it is not so much the length of the signal that influences it, but the possible time lag between reference and modified track.
- For each sample, the time needed to reduce the size of the problem and derive the three time-series related to the respective "*winning climbers*" takes about 0.004 second.
- The Offset evaluation's duration is about 0.004 s.
- For each single sample, the modification evaluation and model application requires 0.35 seconds.

A complete process, related to a signal sampled 16 times, takes something more than 10 seconds, including the creation of the frequency particles.
To evaluate the optimal model, this is compared to the real modified signal in an interval which goes from after the end of the modification to the end of the sampling, in any case before a new change occurs. The deviations between these two signals are collected instant by instant, in this sort of modified steady-state region: the average and maximum value of the gap, relative to each frequency particle discretizing the spectrum, are extracted.
Now a test for each possible category of system's state will be displayed and, in addition, the most significant data will be highlighted.
In particular, the values of the minimum and maximum residue at the limits of each maximum working range[3](where the evaluation concerning the position of the *knob* is carried out) will be listed for each filter, moreover, an indication will be added on the overall maximum values of both the mean and maximum error: all this always along with the positioning on the frequency axis.
All the presented examples will have some characteristics ascribable to the problems outlined in 3.3.4.

[3]For high frequencies is 100-130 Hz, for medium frequencies 1490-1690 Hz and for low frequencies 14000-14500 Hz.

High Frequencies results

The first shown example is an attenuation performed by moving the high frequency filter at its -50% position. The algorithm indicates a -51%, while regarding the model: the residuals are shown in 3.31 and the relative key elements in table 3.4.4.

Key Feature	Frequency Value [Hz]	Percentage Amount
Average Error	100 Hz	0.7224%
Maximum Error	100 Hz	0.8215%
Average Error	130 Hz	0.8925%
Maximum Error	130 Hz	0.9918%
Maximum Value of Average Error	1869.6 Hz	2.1229%
Maximum Value of Maximum Error	1833.6 Hz	2.399%

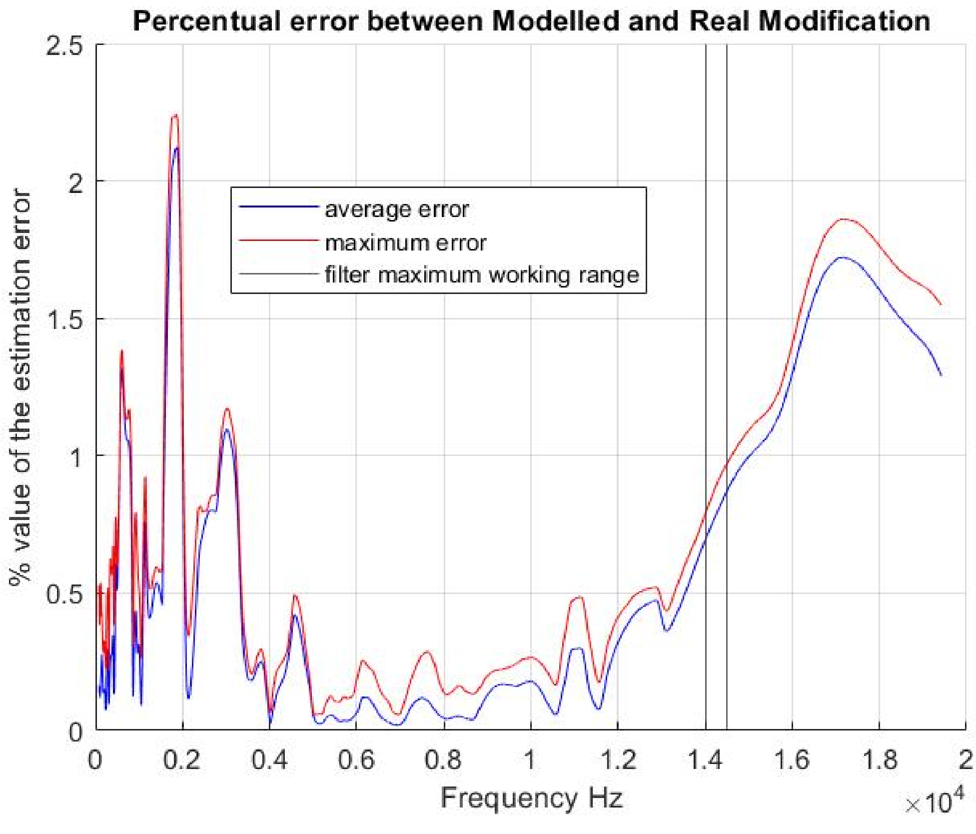


Figure 3.31: Estimation Error Evaluation between the optimal model and the real modified signal for an attenuation at high frequencies of -50%. In red the maximum value for each frequency is shown, in blue the average error while the black vertical lines indicates where the evaluation for the filter's state is performed i.e., its maximum working range.

The second shown example is an high frequency amplification equivalent to a +100% position. The algorithm indicates a +100%. The optimal model residuals, with respect to the real modified signal, are shown in 3.32 and the relative key elements in table 3.4.4.

Key Feature	Frequency Value [Hz]	Percentage Amount
Average Error	100 Hz	0.7646%
Maximum Error	100 Hz	1.1551%
Average Error	130 Hz	0.9844%
Maximum Error	130 Hz	1.4683%
Maximum Value of Average Error	19422 Hz	2.3959%
Maximum Value of Maximum Error	19422 Hz	2.8098%

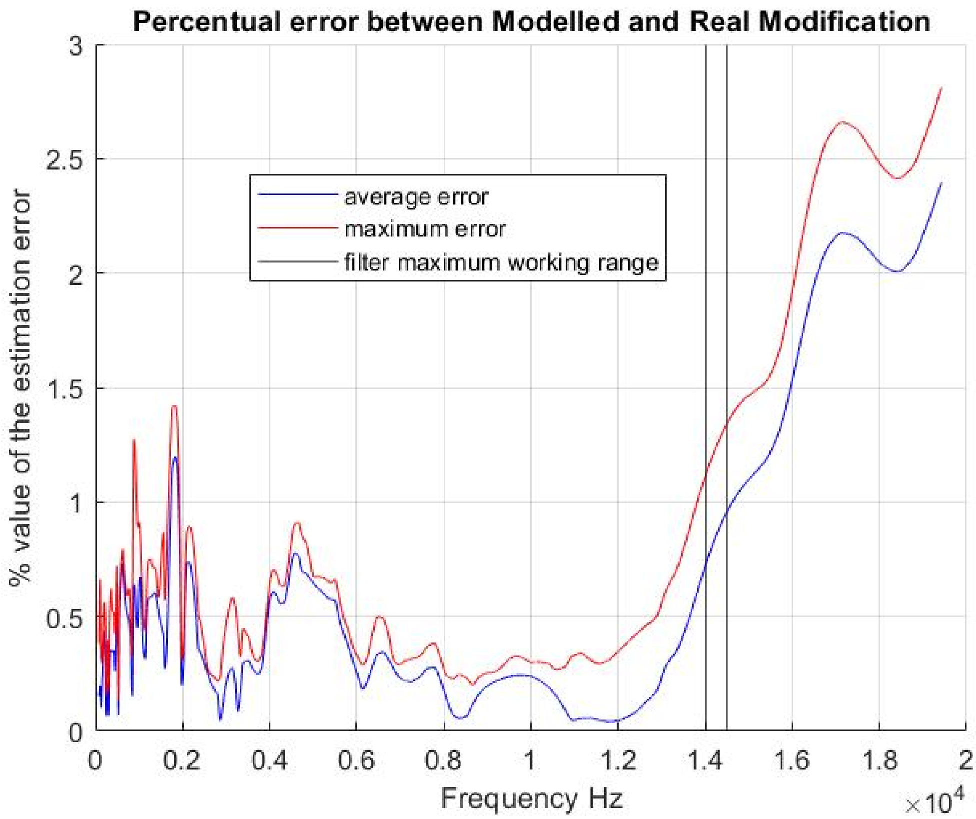


Figure 3.32: Estimation Error Evaluation between the optimal model and the real modified signal for an amplification at high frequencies of +100%. In red the maximum value for each frequency is shown, in blue the average error while the black vertical lines indicates where the evaluation for the filter's state is performed i.e., its maximum working range.

Both cases show two upsurges in the residuals: one at very low frequencies and one at very high frequencies. In both examples the maximum error does not exceed 3%: but, while in the attenuation this value was found at the medium frequencies, in the amplification instead it is in correspondence of the model's limits.

Medium Frequencies results

The presented residual example for medium frequency regards a -44% attenuation, the algorithm has detected -44% so it is working correctly. The picture 3.33 exhibit the complete range of values, while the table 3.4.4 evidences the most important features.

Key Feature	Frequency Value [Hz]	Percentage Amount
Average Error	1490 Hz	0.0506%
Maximum Error	1490 Hz	0.2232%
Average Error	1690 Hz	0.3048%
Maximum Error	1690 Hz	0.4334%
Maximum Value of Average Error	63.66 Hz	3.3278%
Maximum Value of Maximum Error	58.66 Hz	8.23%

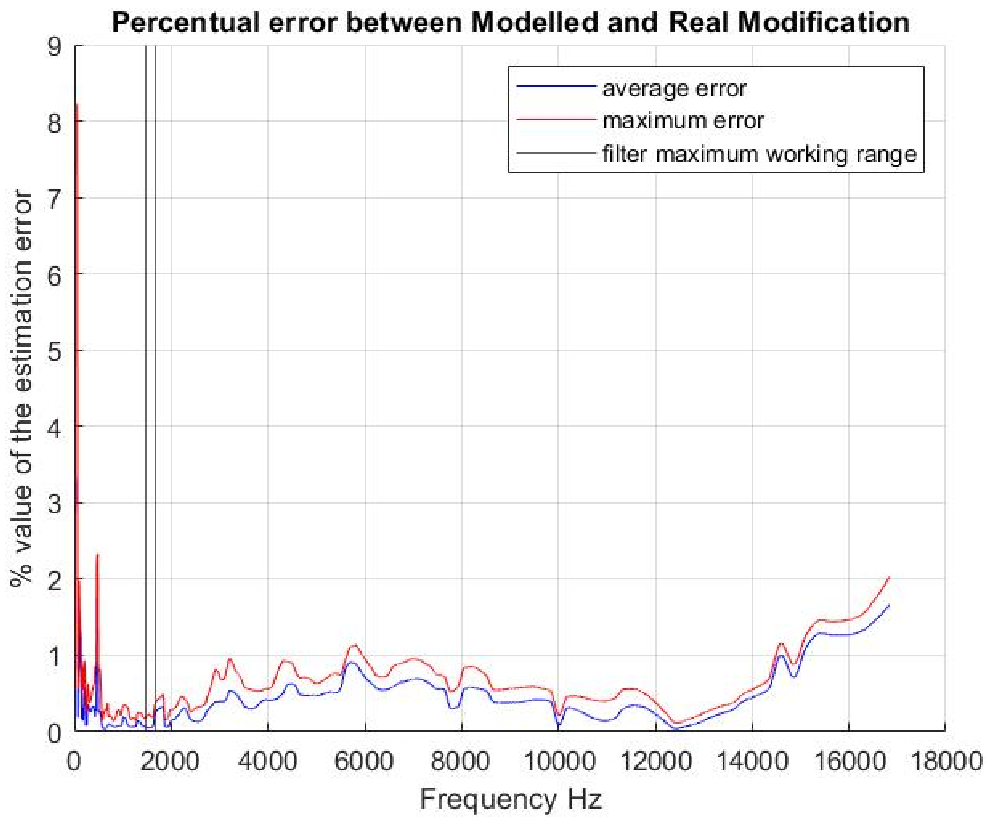


Figure 3.33: Estimation Error Evaluation between the optimal model and the real modified signal for an attenuation at medium frequencies of -44%. In red the maximum value for each frequency is shown, in blue the average error while the black vertical lines indicates where the evaluation for the filter's state is performed i.e., its maximum working range.

The second medium frequencies example is an amplification performed by moving the respective filter at its +76% position. The algorithm indicates a +76%, while regarding the model: the residuals are shown in 3.34 and the relative key elements in table 3.4.4.

Key Feature	Frequency Value [Hz]	Percentage Amount
Average Error	1490 Hz	0.0684%
Maximum Error	1490 Hz	0.3294%
Average Error	1690 Hz	0.0624%
Maximum Error	1690 Hz	0.2604%
Maximum Value of Average Error	151.66 Hz	4.3473%
Maximum Value of Maximum Error	151.66 Hz	4.6472%

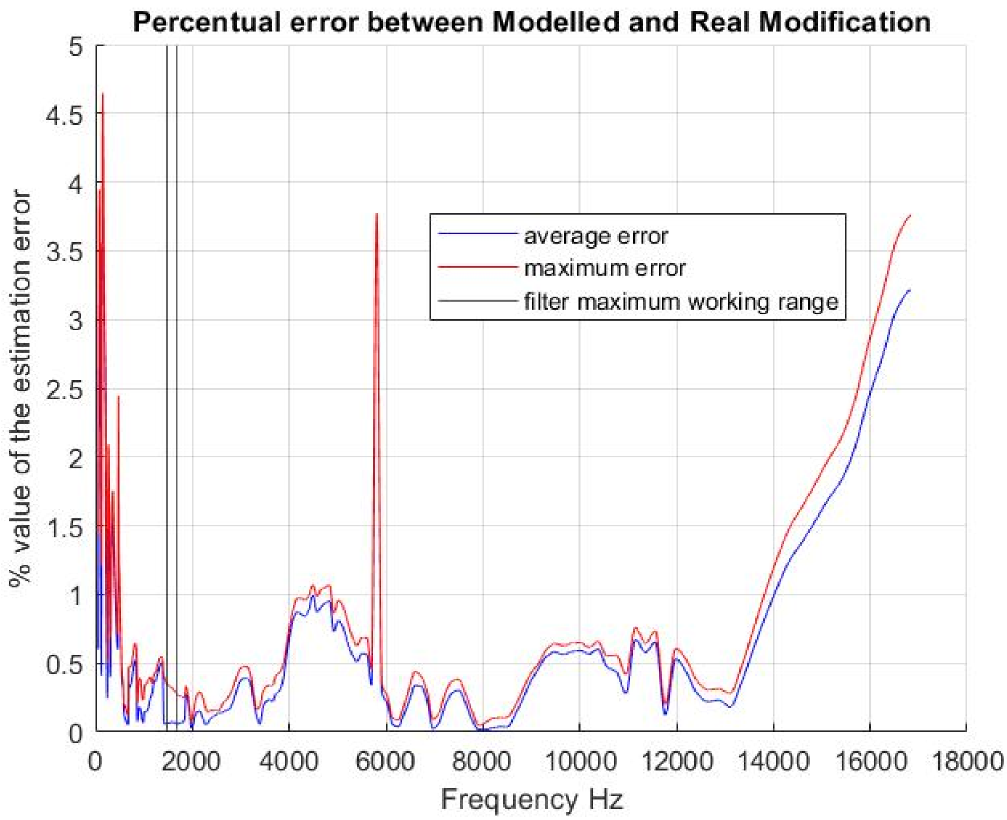


Figure 3.34: Estimation Error Evaluation between the optimal model and the real modified signal for an amplification at medium frequencies of +76%. In red the maximum value for each frequency is shown, in blue the average error while the black vertical lines indicates where the evaluation for the filter's state is performed i.e., its maximum working range.

In both cases there is a strong discrepancy between the ability to almost perfectly replicate the real variation and the high error rates at low frequencies. Again the highest gap values are concentrated at the models' boundaries. In the amplification example the presence of an outlier can also be appreciated, at which average and maximum values overlap.

Low Frequencies results

The last pair of examples consists primarily of an attenuation of -43%, performed by means of the low frequency filter: the algorithm, again, is able to evaluate the right amount. The residuals are shown in 3.35 and the relative key elements in table 3.4.4.

Key Feature	Frequency Value [Hz]	Percentage Amount
Average Error	100 Hz	1.0528%
Maximum Error	100 Hz	3.6864%
Average Error	130 Hz	1.4463%
Maximum Error	130 Hz	6.2568%
Maximum Value of Average Error	202.21 Hz	1.4466%
Maximum Value of Maximum Error	209.21 Hz	6.4267%

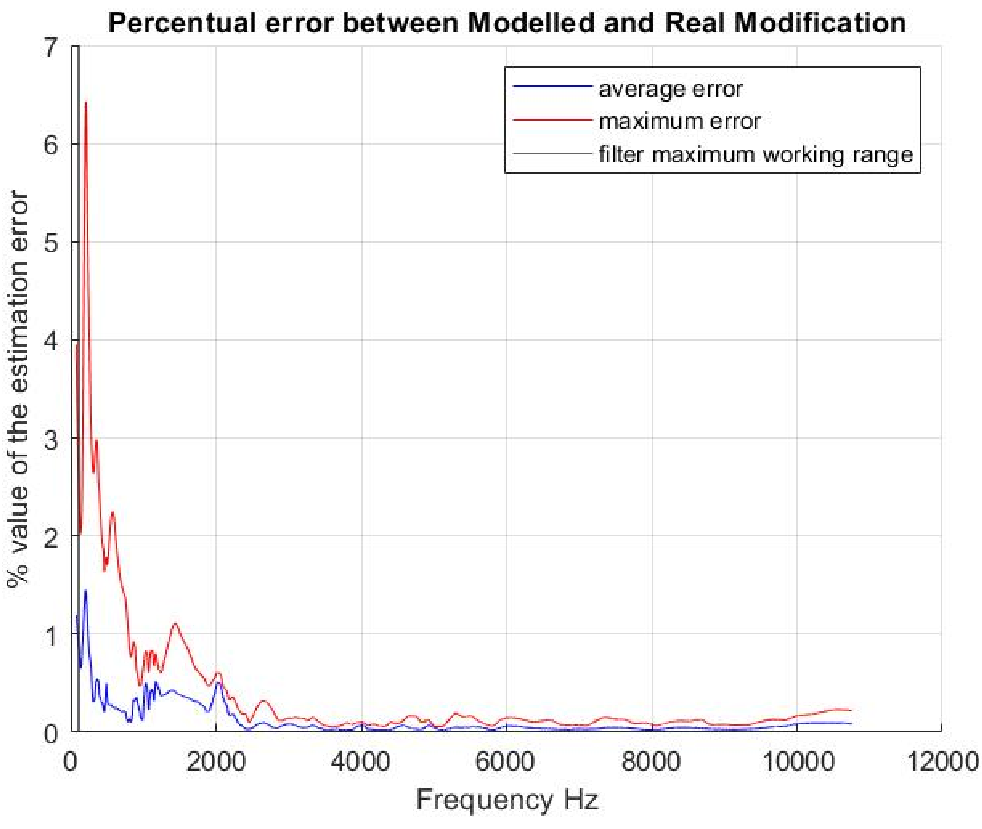


Figure 3.35: Estimation Error Evaluation between the optimal model and the real modified signal for an attenuation at low frequencies of -43%. In red the maximum value for each frequency is shown, in blue the average error while the black vertical lines indicates where the evaluation for the filter's state is performed i.e., its maximum working range.

The overall last example is a +49% amplification at low frequencies, that the algorithm detects as a +47% one: this shows that even the part of the project that seems to have been completed still needs to be improved. Again, and for the last time, the picture 3.36 shows the overall data, while the table 3.4.4 highlights the most important information.

Key Feature	Frequency Value [Hz]	Percentage Amount
Average Error	100 Hz	0.6063%
Maximum Error	100 Hz	0.7407%
Average Error	130 Hz	0.8685%
Maximum Error	130 Hz	1.0043%
Maximum Value of Average Error	202.21 Hz	0.8690%
Maximum Value of Maximum Error	209.21 Hz	1.0159%

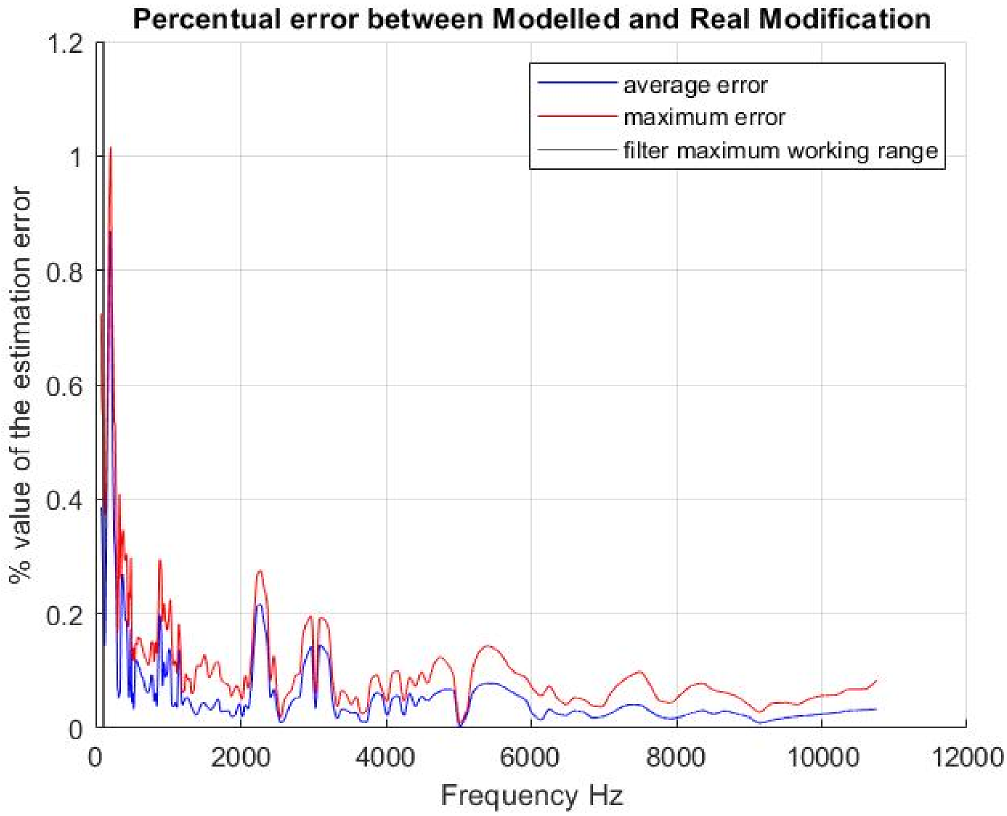


Figure 3.36: Estimation Error Evaluation between the optimal model and the real modified signal for an amplification at low frequencies of +47%. In red the maximum value for each frequency is shown, in blue the average error while the black vertical lines indicates where the evaluation for the filter's state is performed i.e., its maximum working range.

For this last pair, it is useful to note that the maximum errors are very close to the zone of greatest variation and always at the same frequencies. This model stops being evaluated before high frequency problems can occur, so no consideration can be made in this direction.

Chapter 4

The Web Application

As anticipated in 2.2, the whole work is designed to be the backend code of a graphical user interface. Now both the GUI and the link between frontend and backend will be discussed.

4.1 Database structure

The standard way to perform the operation of creating and updating database is SQL.
SQL stands for Structured Query Language, and is a used to communicate with database, which is where the data are stored.
The adopted release of this code is SQLite which is a lighter version of SQL: with this it is possible to create database without a server. Usually a server is needed to manage database, in particular for the handling of users, clients, their communications and the related security levels. In the presented scenario the users will be the customers, and the client the Web Application: every time a user wants to perform a task, the client will make a call to the server, the future cloud environment where the virtual machines to compute will be built, and it will show to the user the received response. With MySQL there would be the need to set up a server, the client, users, password and permission: this would have been useless, since the environment in which both backend and frontend are running, in this moment, is unique.
With SQLite there are no such things, it just ends up generating single files: the reason why this type of files (.db) are better than a text one lies in the distribution of information in memory. If just some data are required, there is no need to open and scan the entire file, but a so called *query* i.e., an interrogation of the file, can be done in order to directly obtain the wanted values. Each generated file is a database containing tables, while the tables include data. In a single file more tables can be created. For this app two database have been set up: one for the filter masks 4.2.4, obtained thanks to user tests, and one for exchanging information needed for the page showing the mixer states 4.2.3. Each database has inside more different tables making the flow of data faster, easier and more straightforward.
Another important aspect is that SQLite can be found by default both on Matlab and Python.
Quite simply: the inputs of the Matlab codes come from requests perpetrated through the

GUI by users, when the computation is concluded, then the backend results are shipped to the frontend. The intermediary, in which the codes search for the data needed to fulfill their tasks, is a database created and updated via SQLite.
For example: the Matlab codes for the Filter Masks and, obviously, for the recursive evaluation of a signal to obtain the *knobs* position, have both a built connection with a given table in their respective database. Each time a computation is completed, these virtual warehouses are updated with the new values. When the frontend code will come back to check this archive, it will find the new amounts and update accordingly.

4.2 GUI by means of Python

The interface has been built by means of Python, with the Tkinter library. The GUI consists of 4 pages, a simple drop-down menu to save, exit and access a tutorial. The last common aspect is a "hidden button", placed at the end of each screen, which allows you to return to the home page.

4.2.1 Home Page

The initial menu just permits to browse through the pages composing the interface, and has the additional possibility to close the app. The screen can be seen at 4.1

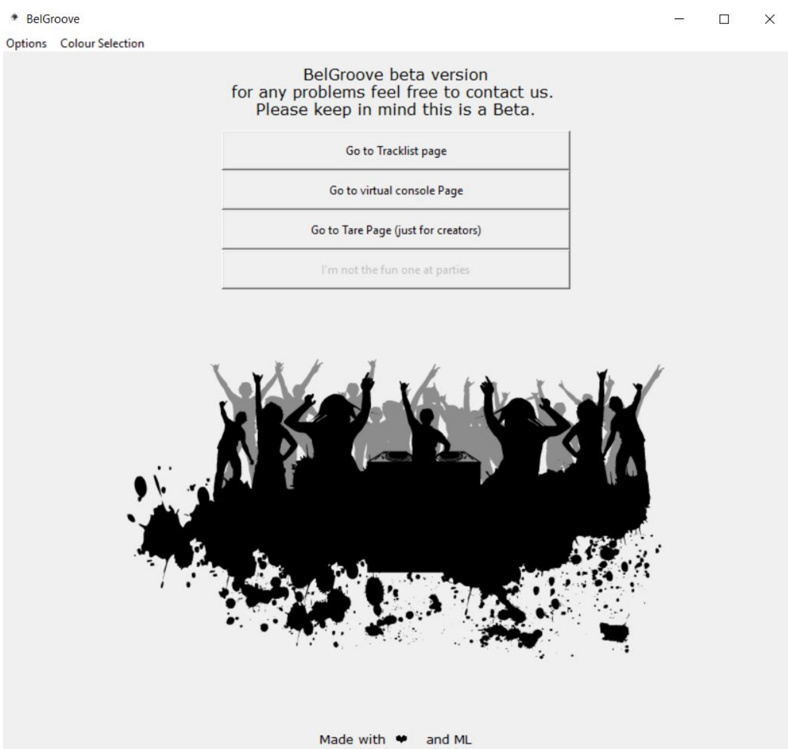


Figure 4.1: The Home Page of the GUI.

4.2.2 Traklist Page

This page is linked to a start-up side not introduced in the thesis work, therefore it won't be presented.

4.2.3 Knobs Page

This is the first version of what was emerged to be a possible solution to a problem highlighted by the forms discussed in 2.2.3. The goal was to represent as much as possible a mixer, shown in 1.1.
This presents two channels of filters, an additional *knob* which pictures the offset i.e., the *Main*, and which value is indicated in *dB* instead of percentage since this is also the choice made by the manufacturer in their Software linked to the Hardware. Each *knob* is represented by means of a circle. The filters' values are updated by reading a table, as explained in 4.1, and according to this the position of a thicker arc of circumference, located within the *knob* representation, is also updated. On the same page there is also a box that gives an indication of the time that has passed.

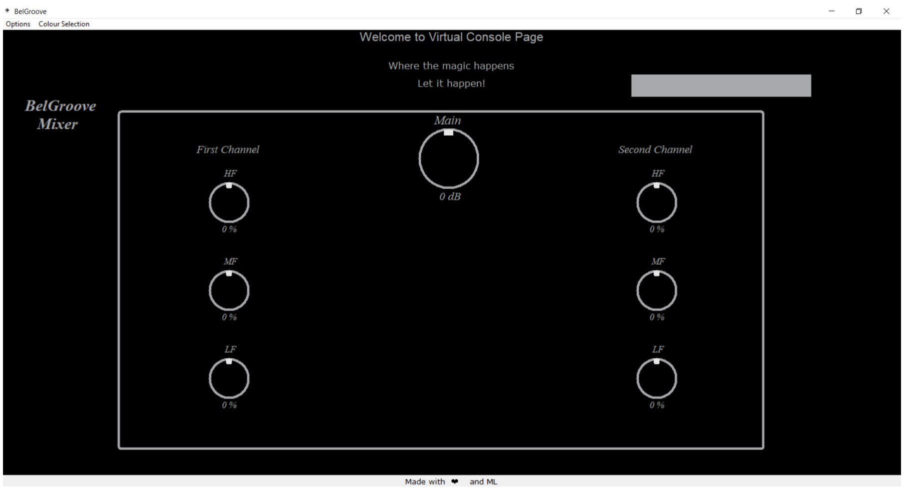


Figure 4.2: The default Knobs Page.

In figure 4.2 the default page, where all the values are set to 0%, is shown, while in 4.3 an example referencing to a modification of -43% at high frequencies of the signal on the first channel and with an offset of 1.8 dB can be appreciated.

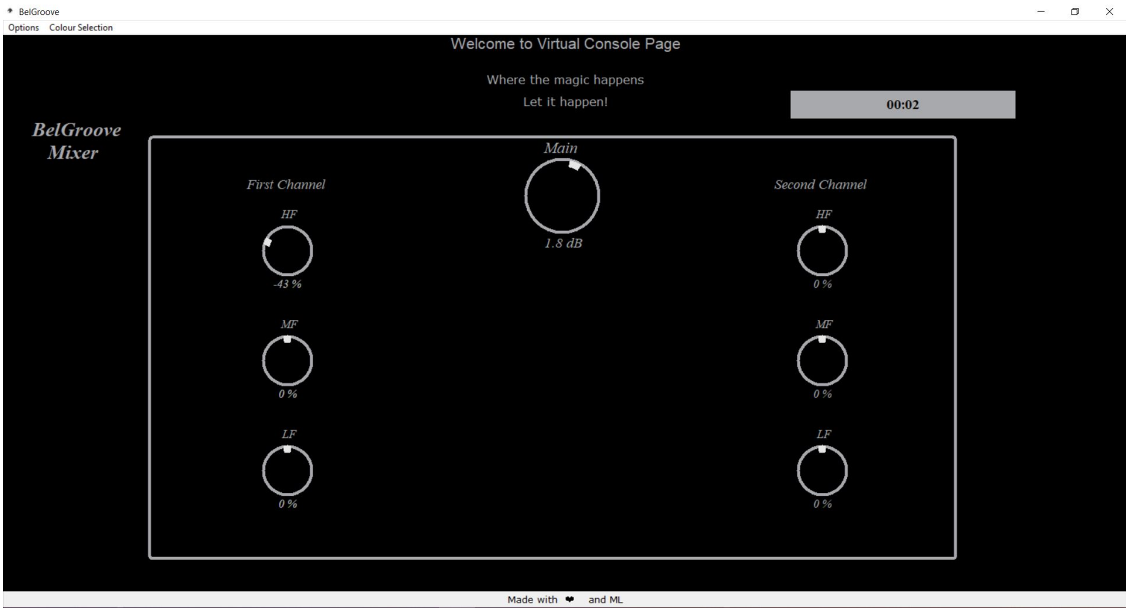


Figure 4.3: The Updated Knob Page. On the top right corner it is possible to see the update time indication, while the high frequency knob of the first channel has an updated value of -43%. The biggest central *knob* i.e., the *Main*, marks 1.8 dB.

4.2.4 Tare Page

This page is born after the amount of work needed to generate the masks. In order to have a wide library of mixer's typologies, the users could became active by performing some tests allowing to model his/her mixer. Just by uploading the audio reference and its modified equivalent, there will be only the further need to specify which was the used filter, if it was an attenuation or an amplification and the value of such modification. At this point the relative backend code , 3.3.3, will perform its computation and the frontend will be able to show an image of the mask obtained.

The starting display 4.4 allows the user to send, to the future cloud, both the reference track and its modified counterpart.
After this operation, there will be the need to add to the shipping some information that allow to call into question the code related to the study of the mask for the filter that is used in the test. As shown in 4.5 it will be enough to select among some options and indicate the percentage value of the change by means of a popup.
Once the computation will be complete, the screen will display the filter mask 4.6.

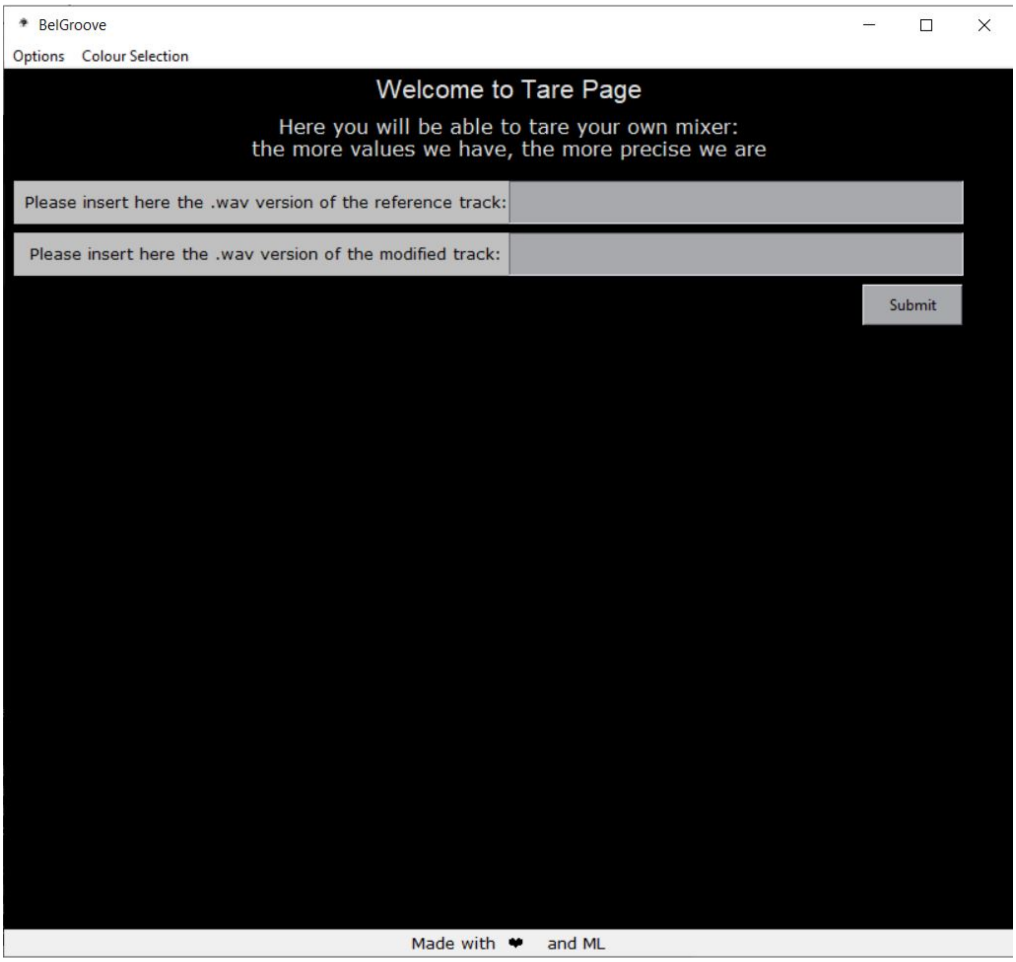


Figure 4.4: The Knobs Page.

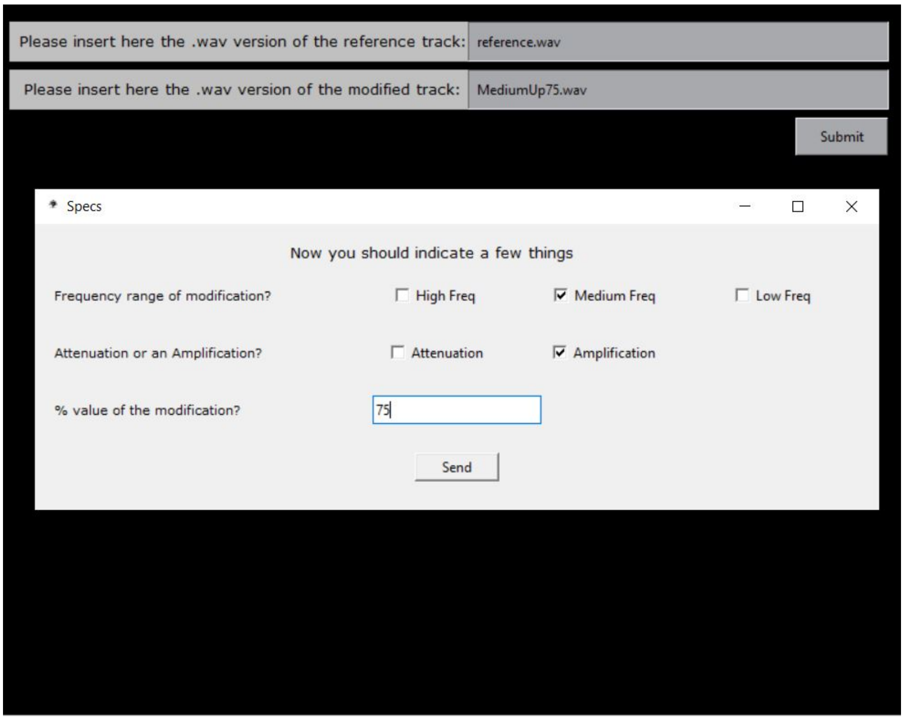


Figure 4.5: The Knobs Page popup.

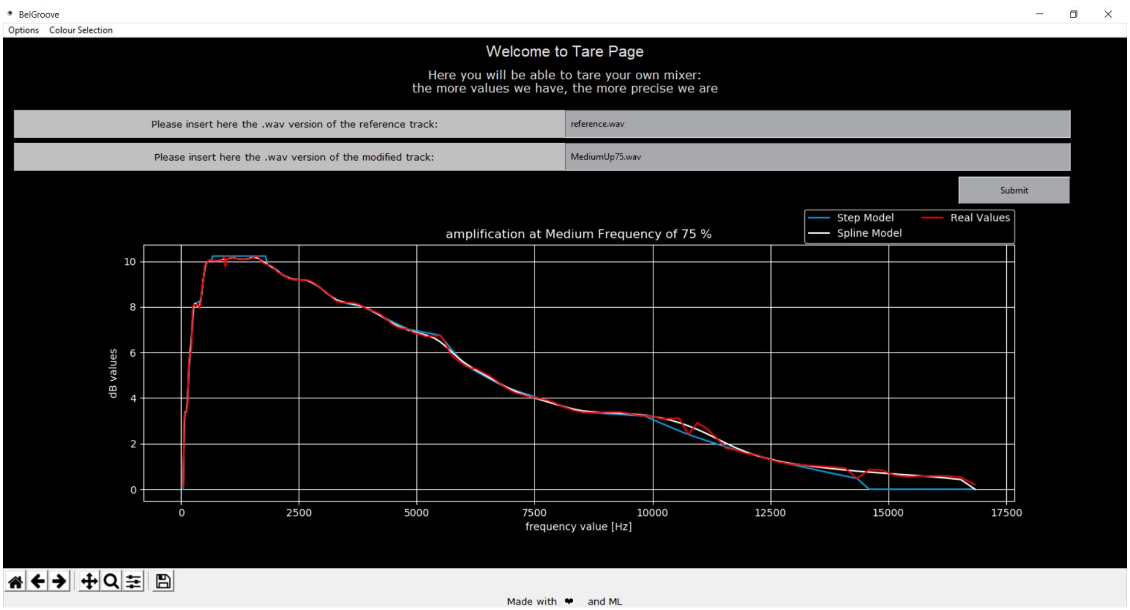


Figure 4.6: The Knobs Page.

Chapter 5

Conclusions

As can be seen, the thesis covers all the steps of the so-called proof of concept: starting from this, demonstrated its effectiveness, it is possible to move to the next level of fundraising, while the *hook* will perform its engagement task.
The chapter 4 is essential, since without a beta version of the graphics, any pitch would certainly fail.

5.1 Future Development

Before singing the own praises, a little time will be spent on the future development from a technical point of view: the troubleshooting.
The possibility of having a perfect model, also taking into account the necessary speed of an application used by outsiders, who rightly must not have the slightest idea of the work behind it, is definitely incompatible with reality. It is well established how answering any factual problems always paves the way for other unresolved problems, as any cause and effect process, and the issues have not been hidden during the dissertation.
The difficulty to obtain a proper filter's characteristic, when performing a too strong attenuation, will be tackled individually for each frequency range: it is absurd, in fact, to think that a low frequency attenuation behaves in the same way as an high frequency one.
A model, based on the weighted average already implemented, can be developed to cancel the wave effect presented in 3.2.4: an evaluation if e.g., one of the *deltas* deviates too far from the other two can be carried out in order to discriminate which values to keep and which to discard.
After all, infinite is just a way to say that the real limit must still be found.
The masks (3.3.3), used to generate the model of the mixer to be applied, can be improved through a number of more tests.
The thesis shows the method to be adopted to characterize and describe the influence of each component on the output: therefore a similar path can be followed for the two *COLOR* elements. The evaluation, will then necessarily need two additional *climbers*, at suitable frequencies, to detect the strength of the inter-range modulation: as said in 1, these *knobs* have a wider influence, which is not restricted to a frequency interval.
Another problem was the MIMO one. Since the algorithm actually sub-samples at a rate

of 50 ms, it is possible to move towards more and more frequent data collection so that for every moment only one component is affecting the output of the system. Once that the right model, regarding the modification, is applied to update the reference track, as explained in 3.4, the new values can be used to perform an inverse of the process shown in 2.3.2: from the time-frequency representation a waveform can be obtained, and this will be helpful to understand which is the track, and therefore the channel, which is providing the modulation.
A translation of the code from Matlab to a faster and a more suitable language for real time applications is hereby not only useful, but also an indispensable condition for the building of virtual machines: in fact the back-end code will have to run on cloud, and exchange information with the front-end in a way similar to the one presented in 4.1.
In general, all future developments should lead to the creation, in a short time, of a first prototype to be brought to market.

5.2 Achievements

Leaving aside the level of results related to the single channel's evaluation, this section will focus on more academic aspects. Here the mask of the sloth will have to be dropped to see if it is actually possible to talk about AI or not.
The followed process is composed of two principal steps: reduction of the problem's dimension and non linear operations to extract the right information. The first stage has been possible thanks to a work of identification of the maximum operating range of each component: this was done with some knowledge of the mixer structure, therefore it is not possible to talk about a black box modelling. Subsequently, the actual extraction was carried out by comparing the results of non-linear operations (maximum, minimum and average). The procedure seems to be a classic Convolutional Neural Network one, but on a so-called grey box i.e., when theoretical structures are combined with data to obtain a representation. The core algorithm itself presents a proper clustering operation, with some a posteriori work on it in order to obtain better subsets i.e., where the non linear operations will act: this adjustment is thus completely made in the pursuit of robustness. It should be kept in mind that the division is done by searching for the longest cluster, which is found by the formula applied to fulfill the task, so no previous knowledge of the output intervenes in the time series subdivision. This kind of operations are acknowledged to be the first step of artificial intelligence.
What is clear is that the black box path would have been a useless addition of difficulty, a "save for the photographers" as the football fans would say: this does not detract from the fact that both the process and the most innovative part of the work can be traced back to artificial intelligence.
Precisely with regard to the extraction itself, this allows to expand the limits of the analysis. Since features both regarding when the system modifies the signal, and itself, and how strongly the DJ is performing this operation are obtained: it can be seen how it is expanded on both the time and the energy axis of the 3-D time-frequency representation of the signal.
This is certainly an interesting result, both from a real application and a more academic point of view.

Appendix A

The components of the vector of sines with their weights

Index (i)	Sine Function	Weight
1	$y_1 = sin(2\pi 22t)$	$w_1 = 1.8$
2	$y_2 = sin(2\pi 37t)$	$w_2 = 0.7$
3	$y_3 = sin(2\pi 50t)$	$w_3 = 0.05$
4	$y_4 = sin(2\pi 68.3t)$	$w_4 = 0.7$
5	$y_5 = sin(2\pi 82.5t)$	$w_5 = 0.33$
6	$y_6 = sin(2\pi 93t)$	$w_6 = 0.6$
7	$y_7 = sin(2\pi 100.1t)$	$w_7 = 0.01$
8	$y_8 = sin(2\pi 148.02t)$	$w_8 = 0.2$
9	$y_9 = sin(2\pi 163t)$	$w_9 = 0.02$
10	$y_{10} = sin(2\pi 199.12t)$	$w_{10} = 0.02$
11	$y_{11} = sin(2\pi 205t)$	$w_{11} = 0.01$
12	$y_{12} = sin(2\pi 267.15t)$	$w_{12} = 0.03$
13	$y_{13} = sin(2\pi 302.3t)$	$w_{13} = 0.03$
14	$y_{14} = sin(2\pi 347.1t)$	$w_{14} = 0.01$
15	$y_{15} = sin(2\pi 383t)$	$w_{15} = 0.05$
16	$y_{16} = sin(2\pi 407.1t)$	$w_{16} = 1.4$
17	$y_{17} = sin(2\pi 429t)$	$w_{17} = 0.05$
18	$y_{18} = sin(2\pi 469t)$	$w_{18} = 0.3$
19	$y_{19} = sin(2\pi 484.84t)$	$w_{19} = 0.3$
20	$y_{20} = sin(2\pi 529t)$	$w_{20} = 0.02$
21	$y_{21} = sin(2\pi 555t)$	$w_{21} = 0.02$
22	$y_{22} = sin(2\pi 597t)$	$w_{22} = 0.02$
23	$y_{23} = sin(2\pi 623t)$	$w_{23} = 0.02$
24	$y_{24} = sin(2\pi 667t)$	$w_{24} = 0.02$
25	$y_{25} = sin(2\pi 703.24t)$	$w_{25} = 0.3$
26	$y_{26} = sin(2\pi 755.2t)$	$w_{26} = 0.02$
27	$y_{27} = sin(2\pi 792t)$	$w_{27} = 0.08$
28	$y_{28} = sin(2\pi 900t)$	$w_{28} = 0.01$

29	$y_{29} = sin(2\pi 1123t)$	$w_{29} = 0.2$
30	$y_{30} = sin(2\pi 1863t)$	$w_{30} = 0.01$
31	$y_{31} = sin(2\pi 2231t)$	$w_{31} = 0.02$
32	$y_{32} = sin(2\pi 2765t)$	$w_{32} = 0.4$
33	$y_{33} = sin(2\pi 3011t)$	$w_{33} = 0.05$
34	$y_{34} = sin(2\pi 3437t)$	$w_{34} = 0.01$
35	$y_{35} = sin(2\pi 4783t)$	$w_{35} = 0.03$
36	$y_{36} = sin(2\pi 5137t)$	$w_{36} = 0.36$
37	$y_{37} = sin(2\pi 6177t)$	$w_{37} = 0.02$
38	$y_{38} = sin(2\pi 6873t)$	$w_{38} = 0.1$
39	$y_{39} = sin(2\pi 7333t)$	$w_{39} = 0.07$
40	$y_{40} = sin(2\pi 8073.1t)$	$w_{40} = 0.01$
41	$y_{41} = sin(2\pi 8623.1t)$	$w_{41} = 0.1$
42	$y_{42} = sin(2\pi 9329.91t)$	$w_{42} = 0.03$
43	$y_{43} = sin(2\pi 9999.98t)$	$w_{43} = 0.5$
44	$y_{44} = sin(2\pi 10733t)$	$w_{44} = 0.07$
45	$y_{45} = sin(2\pi 11333.01t)$	$w_{45} = 0.06$
46	$y_{46} = sin(2\pi 11743.03t)$	$w_{46} = 0.1$
47	$y_{47} = sin(2\pi 12379.04t)$	$w_{47} = 0.09$
48	$y_{48} = sin(2\pi 12784.3t)$	$w_{48} = 0.005$
49	$y_{49} = sin(2\pi 13432.09t)$	$w_{49} = 0.042$
50	$y_{50} = sin(2\pi 13872.02t)$	$w_{50} = 0.23$
51	$y_{51} = sin(2\pi 14289.1t)$	$w_{51} = 0.01$
52	$y_{52} = sin(2\pi 14877.1t)$	$w_{52} = 0.02$
53	$y_{53} = sin(2\pi 15092.3t)$	$w_{53} = 0.03$
54	$y_{54} = sin(2\pi 15783.3t)$	$w_{54} = 0.04$
55	$y_{55} = sin(2\pi 16023.22t)$	$w_{55} = 0.05$
56	$y_{56} = sin(2\pi 16783.3t)$	$w_{56} = 0.01$
57	$y_{57} = sin(2\pi 17420.23t)$	$w_{57} = 0.02$
58	$y_{58} = sin(2\pi 17992.3t)$	$w_{58} = 0.03$
59	$y_{59} = sin(2\pi 16532.2t)$	$w_{59} = 0.04$
60	$y_{60} = sin(2\pi 18829.4t)$	$w_{60} = 0.005$
61	$y_{61} = sin(2\pi 19734.4t)$	$w_{61} = 0.021$
62	$y_{62} = sin(2\pi 20202.03t)$	$w_{62} = 0.01$

Table A.1: The Sines, with the associated Weights.

Appendix B

Results of each test for filters' characteristic

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
-10%	-1,7389	-1,7463	-1,7589	-0,2496
-15%	-2,6073	-2,5475	-2,6445	-0,3188
-22%	-3,952	-3,9141	-3,9551	-0,6387
-25%	-4,49	-4,3842	-4,5306	-0,2804
-26%	-4,87	-4,8623	-4,881	-0,6824
-31%	-5,9035	-5,8556	-5,8784	-0,7315
-32%	-5,9258	-5,8846	-5,947	-0,7233
-37%	-7,3166	-7,2531	-7,3626	-0,7402
-38%	-7,4733	-7,4231	-7,5169	-0,8093
-43%	-9,7707	-9,6908	-9,7764	-0,89
-44%	-10,2347	-10,1976	-10,2339	-1,0198
-50%	-13,4366	-13,1075	-13,5089	-1,2854
-51%	-13,7555	-13,5975	-13,7981	-0,9884
-52%	-13,7721	-13,5973	-13,8851	-0,8637
-55%	-16,1406	-16,1125	-16,1963	-1,464
-59%	-18,3212	-18,2652	-18,3715	-1,6657
-64%	-23,2833	-23,2358	-23,3586	-1,4595

Table B.1: Table of the first test to characterize the High Frequency Filter Attenuation.

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
-10%				
-1,7355	-1,755	-1,7176	-0,1583	
-15%	-2,6042	-2,5994	-2,5892	-0,2757
-22%	-3,9568	-3,9369	-4,0101	-0,2815
-25%	-4,4869	-4,4782	-4,4614	-0,3863
-26%	-4,8745	-4,8977	-4,8636	-0,327
-31%	-5,9094	-5,9157	-5,9266	-0,4149
-32%	-5,9166	-5,8823	-5,9729	-0,3236
-37%	-7,2027	-7,223	-7,1954	-0,2893
-38%	-7,3221	-7,3231	-7,31174	-0,7994
-43%	-9,7798	-9,7768	-9,7938	-0,7462
-44%	-10,2348	-10,2693	-10,2898	-0,5109
-50%	-13,4694	-13,453	-13,4564	-1,1283
-51%	-13,7585	-13,7181	-13,7796	-0,7984
-52%	-13,7729	-13,8066	-13,7463	-0,8417
-55%	-16,1538	-16,1057	-16,1406	-0,9021
-59%	-18,3212	-18,3602	-18,2931	-0,6103
-64%	-23,2669	-23,2447	-23,266	-0,7764

Table B.2: Table of the second test to characterize the High Frequency Filter Attenuation.

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
10%	1,5227	1,5507	1,546	0,1668
19%	2,7201	2,6567	2,717	0,1735
27%	3,6712	3,6681	3,6468	0,3033
31%	4,1232	4,101	4,1414	0,2112
32%	4,1537	4,1389	4,1389	0,3008
37%	4,7765	4,7752	4,7416	0,4051
44%	5,5217	5,6564	wrong entry	0,4146
46%	5,9493	5,9834	5,5784	0,2652
53%	7,1746	7,22355	7,0946	0,5974
59%	7,9156	7,9157	7,9264	0,5302
65%	8,6488	8,6509	8,6206	0,3941
71%	9,4955	9,5061	9,5497	0,3421
80%	10,5431	10,5287	10,5213	0,58
89%	11,0652	11,1176	11,1066	0,5269
99%	11,6703	11,7109	11,6295	0,5785
100%	11,7209	11,7212	11,7188	0,8404

Table B.3: Table of the first test to characterize the High Frequency Filter Amplification.

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
10%	1,5227	1,5507	1,546	0,1668
19%	2,7201	2,6567	2,717	0,1735
27%	3,6712	3,6712	3,6468	0,3033
31%	4,1232	4,101	4,1414	0,2112
32%	4,1485	4,1087	4,1716	0,2929
37%	4,7765	4,7752	4,7416	0,4051
44%	5,5217	5,6564	wrong entry	0,4146
46%	5,9493	5,9834	5,5784	0,2652
53%	7,2124	7,2122	7,2088	0,328
59%	7,8885	7,9084	7,9226	0,415
65%	8,6488	8,6509	8,6206	0,3941
71%	9,4955	9,5061	9,5497	0,3421
80%	10,5431	10,5287	10,5213	0,58
89%	11,0652	11,1176	11,1066	0,5269
99%	11,6679	11,6655	11,6599	0,5785
100%	11,9191	11,6986	12	0,5036

Table B.4: Table of the second test to characterize the High Frequency Filter Amplification.

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
-10%	-1,7604	-1,6906	-1,7182	-0,1383
-15%	-2,6040	-2,5071	-2,5494	-0,2026
-16%	-2,7553	-2,7467	-2,7834	-0,2374
-22%	-3,8209	-3,7841	-3,85	-0,3536
-24%	-4,3904	-4,4199	-4,354	-0,2945
-26%	-4,7563	-4,6933	-4,7871	-0,3196
-30%	-5,6721	-5,6914	-5,6249	-0,2727
-33%	-6,3865	-6,3817	-6,34888	-0,2787
-37%	-7,3076	-7,3038	-7,3209	-0,5624
-39%	-7,6467	-7,6341	-7,6707	-0,6391
-44%	-10,0452	-10,0287	-10,0506	-0,1423
-48%	-12,3758	-12,418	-12,3342	-0,8722
-55%	-16,5572	-16,4617	-16,5543	-0,6881
-59%	-18,8736	-18,8691	-18,8767	-0,9976
-60%	-19,3792	-19,2776	-20,1199	-1,9658
-63%	-22,4898	-22,4896	-22,5661	-1,7322
-65%	-23,9636	-23,3711	-24,0846	-0,9618

Table B.5: Table of the first test to characterize the Medium Frequency Filter Attenuation

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
-16%	-2,7729	-2,7736	-2,7995	-0,3196
-22%	-3,8425	-3,8089	-3,836	-0,1534
-26%	-4,7496	-4,7252	-4,7553	-0,3127
-31%	-5,8381	-5,7734	-5,7774	-0,3775
-37%	-7,2945	-7,2546	-7,2115	-0,3626
-38%	-7,2967	-7,2324	-7,3544	-0,8012
-39%	-7,6340	-7,6374	-7,6744	-0,8191
-48%	-12,3957	-12,3811	-12,4305	-1,1284
-52%	-14,6994	-14,7	-14,679	-0,5678
-59%	-18,8895	-18,8636	-18,8597	-0,7325
-60%	-19,3384	-19,3649	-19,3335	-1,1394
-63%	-22,5586	-22,5394	-22,5236	-0,872

Table B.6: Table of the second test to characterize the Medium Frequency Filter Attenuation.

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
4%	0,6574	0,6107	0,7454	0,0936
19%	2,8248	3,0253	2,895	0,2354
29%	3,782	3,8838	2,6717	0,2113
31%	4,1022	4,0766	4,1298	0,21
37%	4,8366	4,8667	4,7344	0,2322
45%	5,9636	6,0551	5,937	0,293
49%	6,5987	6,752	6,4097	0,4896
57%	7,6667	7,7813	7,6496	0,2732
63%	8,5837	8,5981	8,62	0,2292
75%	10,1115	10,2291	10,1024	0,4742
79%	10,6663	10,7754	10,6224	0,4729
83%	10,9624	10,9853	10,8795	0,4389
96%	11,6866	11,7522	11,6162	0,4509
100%	11,9255	11,8716	11,9337	0,6266

Table B.7: Table of the first test to characterize the Medium Frequency Filter Amplification.

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
9%	0,6614	0,5825	0,7838	0,0925
26%	3,5619	3,4904	3,6554	0,2537
30%	4,0192	4,0417	4,0838	0,1834
35%	4,5645	4,6815	4,5534	0,2227
44%	5,715	5,8075	5,7081	0,3114
46%	6,1191	6,2938	6,0944	0,367
53%	7,044	7,056	6,9469	0,4701
60%	8,241	8,2674	8,2137	0,3341
67%	9,0197	9,05759	9,0265	0,3923
77%	10,4354	10,5208	10,3597	0,3881
82%	10,903	10,9644	9,7217	0,6474
88%	11,2438	11,3773	11,2103	0,3273
99%	11,9068	11,9409	11,8033	0,4562
100%	11,925	11,9291	11,9131	0,9911

Table B.8: Table of the second test to characterize the Medium Frequency Filter Attenuation.

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
-6%	-1,0616	-1,0349	-1,0631	-0,1186
-19%	-3,2983	-3,1401	-3,3444	-0,3623
-26%	-4,7381	-4,5662	-4,8840	-0,3728
-39%	-7,6091	-7,5023	-7,6122	-0,5274
-41%	-8,5251	-8,4045	-8,6596	-0,7630
-52%	-14,6217	-14,5121	-14,7648	-0,6736
-61%	-19,8911	-19,0703	-20,1941	-1,9899
-67%	-25,3591	-23,1633	-25,9488	-1,9868

Table B.9: Table of the first test to characterize the Low Frequency Filter Attenuation

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
-6%	-1,0602	-1,0642	-1,0540	-0,1116
-19%	-3,3008	-3,2712	-3,3404	-0,1959
-26%	-4,6780	-4,6268	-4,7390	-0,4747
-39%	-7,6318	-7,5854	-7,6810	-0,3704
-41%	-8,5711	-8,4309	-8,6708	-0,4162
-52%	-14,5334	-14,0968	-14,6776	-0,5336
-61%	-19,9469	-19,2362	-20,2243	-0,7843
-67%	-25,6056	-25,5439	-25,6529	-0,9946

Table B.10: Table of the second test to characterize the Low Frequency Filter Attenuation.

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
8%	1,1000	1,0878	1,1072	0,1223
16%	2,3546	2,4068	2,3882	0,1804
26%	3,5646	3,6167	3,5538	0,1577
32%	4,1765	4,1811	4,2144	0,2309
33%	4,3106	4,3396	4,3414	0,3029
36%	4,7258	4,7681	4,6673	0,3930
40%	5,0479	5,0241	5,0421	0,2396
50%	6,5675	6,5645	6,5988	0,2680
53%	6,9238	6,8932	6,9387	0,2535
53%	7,0156	6,9900	7,0222	0,4176
70%	9,2475	9,1789	9,3022	0,3137
81%	10,5694	10,5697	10,5940	0,3395
88%	10,8747	10,8511	10,8863	0,5701
98%	11,4120	11,3754	11,4824	0,2444
99%	12,7418	11,5069	14,1318	0,5254
100%	11,4646	11,4841	11,4165	0,5249

Table B.11: Table of the first test to characterize the Low Frequency Filter Amplification.

Percentage value of Modification	Avg Diff dB	Max Diff dB	Min Diff dB	Slope
8%	1,0937	1,1302	1,0863	0,0864
16%	2,3155	2,3209	2,2609	0,2287
26%	3,6325	3,6249	3,6019	0,2963
33%	4,3426	4,3297	4,3155	0,2950
33%	4,2320	4,3565	4,3301	0,3931
36%	4,5516	1,7379	4,6883	0,2982
40%	4,7602	0,2473	5,0644	0,2428
50%	6,5732	6,5803	6,5447	0,3900
53%	6,9417	6,9619	6,9097	0,4012
53%	6,9899	7,0217	6,9797	0,4988
67%	8,8457	8,8122	8,8369	0,5169
70%	9,2124	9,1948	9,2827	0,6655
80%	10,4805	10,5102	10,4209	0,6860
81%	10,5985	10,6900	10,6062	0,5322
88%	10,8225	9,7731	10,8595	1,0494
98%	6,2274	5,0617	11,1324	0,6426
100%	11,5034	11,4415	11,4912	1,2946

Table B.12: Table of the second test to characterize the Low Frequency Filter Attenuation.

Appendix C

Filter Characteristic

C.1 HF characteristic

Percentage value of Modification	First Interval Value dB	Second Interval Value dB
-64%	-23.2358000000000	-23.3586000000000
-63%	-22.0310191417099	-22.1517079534297
-62%	-20.9195407584630	-21.0377571936223
-61%	-19.9133928043612	-20.0285424571001
-60%	-19.0246032335063	-19.1358584803853
-59%	-18.2652000000000	-18.3715000000000
-58%	-17.6645267559187	-17.7605093829328
-57%	-17.1747137625812	-17.2571891745340
-56%	-16.6919688879531	-16.7672243788682
-55%	-16.1057000000000	-16.1963000000000
-54%	-15.1659985457954	-15.3420612214432
-53%	-14.1021826062310	-14.4027841111849
-52%	-13.5973000000000	-13.8851000000000
-51%	-13.5975000000000	-13.7981000000000
-50%	-13.1075000000000	-13.5089000000000
-49%	-12.6222126641598	-13.0893701746980
-48%	-12.1406850132158	-12.5633617635255
-47%	-11.6601884842481	-11.9952648201722
-46%	-11.1779945143363	-11.3992265962870
-45%	-10.6913745405603	-10.8242253142816
-44%	-10.1976000000000	-10.2898000000000
-43%	-9.69080000000000	-9.79380000000000
-42%	-9.18070986493718	-9.28320371064178
-41%	-8.62858231212796	-8.75279810283095
-40%	-8.05511790272013	-8.24304063969923
-39%	-7.59238232887487	-7.81178878437834
-38%	-7.31174000000000	-7.51690000000000

-37%	-7.19540000000000	-7.36260000000000
-36%	-6.97426788071127	-7.11095665305975
-35%	-6.64695305284312	-6.77601269727415
-34%	-6.29069506152363	-6.42533041495867
-33%	-6.00055568717454	-6.12647208842880
-32%	-5.88230000000000	-5.97290000000000
-31%	-5.85560000000000	-5.92660000000000
-30%	-5.75904554575986	-5.85735053163579
-29%	-5.59095987271974	-5.69993159490736
-28%	-5.38281142679969	-5.46988739236105
-27%	-5.14326865391976	-5.19366212654315
-26%	-4.86230000000000	-4.89770000000000
-25%	-4.38420000000000	-4.53060000000000
-24%	-4.18582724789309	-4.30773876786197
-23%	-4.05484828676705	-4.14975402310994
-22%	-3.91410000000000	-4.01010000000000
-21%	-3.73394171493970	-3.82977306455913
-20%	-3.53977059794129	-3.63173954074050
-19%	-3.33698029089398	-3.42209162580180
-18%	-3.13096443568699	-3.20692151700071
-17%	-2.92711667420952	-3.01231694054491
-16%	-2.73083064835078	-2.82739022843402
-15%	-2.54750000000000	-2.64450000000000
-14%	-2.37544577803250	-2.46374039109313
-13%	-2.20935542885799	-2.28490186855120
-12%	-2.04917219066723	-2.10788315046270
-11%	-1.88228291744900	-1.93258295491614
-10%	-1.71760000000000	-1.75890000000000
-9%	-1.55749009045922	-1.60349752390505
-8%	-1.40183042214301	-1.46637511155688
-7%	-1.24654175991909	-1.33487600114626
-6%	-1.07831494840960	-1.20894343086395
-5%	-0.911198967973023	-1.08852063890070
-4%	-0.745092536697840	-0.973550863447281
0%	-0.579894372672551	-0.863977342694447
0%	-0.415503193985650	-0.759743314832959
0%	-0.251817718725635	-0.660792018053576
0%	-0.0887366649809982	-0.567066690547058
0%	0.363546904508416	0.0368360007128157
4%	0.832716653768439	0.650282048087006
5%	0.951223956781650	0.799701209453633
6%	1.07020321943267	0.947525714092482
7%	1.18964744840075	1.09374670373505
8%	1.30954965036516	1.23835532011283
9%	1.42990283200515	1.38134270495731
10%	1.55070000000000	1.52270000000000

11%	1.68265745780653	1.66241834697239
12%	1.81769881474115	1.79359832177129
13%	1.95111529117447	1.91568548890626
14%	2.08289810747706	2.03818866911312
15%	2.21303848401953	2.16110086907113
16%	2.34152764117246	2.28441509545957
17%	2.46835679930646	2.40812435495769
18%	2.59381213525621	2.53222165424474
19%	2.72010000000000	2.65670000000000
20%	2.84436240377258	2.78296017442782
21%	2.96648948418831	2.91162257692886
22%	3.08678105952954	3.04150963997682
23%	3.20553694807860	3.17144379604540
24%	3.32305696811783	3.29793580971494
25%	3.43964093792958	3.41224354266858
26%	3.55558867579619	3.52820575524637
27%	3.67120000000000	3.64680000000000
28%	3.79874023507492	3.79117639585566
29%	3.95421805584606	3.91675598044471
30%	4.08666302094227	4.02623257481141
31%	4.14140000000000	4.10100000000000
32%	4.15370000000000	4.13890000000000
33%	4.22765860586421	4.18457494131405
34%	4.34867105894495	4.30003282394215
35%	4.49498420909358	4.45292323591323
36%	4.64484490616147	4.61089576525620
37%	4.77650000000000	4.74160000000000
38%	4.89918268303099	4.84120898093181
39%	5.01959713752949	4.93006212813558
40%	5.13896490686595	5.01205859858540
41%	5.25980753441080	5.09109754925535
42%	5.38464656353447	5.17107813711952
43%	5.51600353760739	5.25589951915198
44%	5.65640000000000	5.34946085232683
45%	5.81413812068596	5.45566129361814
46%	5.98340000000000	5.57840000000000
47%	6.15909795809859	5.73900785871747
48%	6.34478373763655	5.94537730346038
49%	6.53475791127339	6.18124981086998
50%	6.72332105166865	6.43036685758750
51%	6.90477373148182	6.67646992025418
52%	7.07341652337243	6.90330047551127
53%	7.22355000000000	7.09460000000000
54%	7.35482834507933	7.25544256392821
55%	7.47371646885489	7.40362798059030
56%	7.58473103378071	7.54198634125740

57%	7.69238870231082	7.67334773720064
58%	7.80120613689924	7.79542652121887
59%	7.92640000000000	7.91560000000000
60%	8.04701960691034	8.03438278028301
61%	8.15963245814629	8.15337249933905
62%	8.27610078556308	8.26876773839054
63%	8.39688394103185	8.37895463232578
64%	8.52143290749859	8.49472232463470
65%	8.65090000000000	8.62060000000000
66%	8.78806608497937	8.76351409325306
67%	8.93313084148461	8.92198190491505
68%	9.08791891820337	9.07408884060679
69%	9.25324061633538	9.22007024956548
70%	9.40986248252846	9.36181067828641
71%	9.54970000000000	9.49550000000000
72%	9.67673166994611	9.62522522377288
73%	9.80002981361280	9.75628137616645
74%	9.91915064429227	9.88572950959160
75%	10.0336503752767	10.0078059425223
76%	10.1430852198583	10.1255753929058
77%	10.2538532613999	10.2377936922936
78%	10.3615925489860	10.3432166722376
79%	10.4586741620285	10.4365625681079
80%	10.5431000000000	10.5213000000000
81%	10.6163373083943	10.5997696086826
82%	10.6822037703334	10.6732929478837
83%	10.7505782660262	10.7422269417321
84%	10.8152826523417	10.7979343785051
85%	10.8771450090988	10.8508536365671
86%	10.9372366292702	10.9025122718329
87%	10.9966288058290	10.9544378402171
88%	11.0563928317480	11.0081578976346
89%	11.1176000000000	11.0652000000000
90%	11.1833386821225	11.1252623858949
91%	11.2543170860026	11.1867047869510
92%	11.3280362591377	11.2490469986582
93%	11.4019972490251	11.3061162958898
94%	11.4737011031622	11.3518741750218
95%	11.5406488690462	11.3989261201521
96%	11.6003415941747	11.4487434032478
97%	11.6502803260449	11.5027972962762
98%	11.6879661121542	11.5625590712046
99%	11.7109000000000	11.6295000000000
100%	11.7212000000000	11.7188000000000

Table C.1: The High Frequency Filter characteristic's values.

C.2 MF characteristic

Percentage value of Modification	First Interval Value dB	Second Interval Value dB
-63%	-22.5236000000000	-22.5586000000000
-62%	-21.5561824717667	-21.7848969536000
-61%	-19.8756199540217	-20.0606835984645
-60%	-19.2776000000000	-20.1199000000000
-59%	-18.8691000000000	-18.8767000000000
-58%	-18.7518586091846	-18.7995720848639
-57%	-18.1417370140368	-18.2071386316322
-56%	-17.2565840449882	-17.3468013874711
-55%	-16.4617000000000	-16.5572000000000
-54%	-15.8539322794726	-15.9233559688241
-53%	-15.2757663050215	-15.3042365442612
-52%	-14.6790000000000	-14.7000000000000
-51%	-14.0093778460812	-14.1390648425789
-50%	-13.2487330854055	-13.6452391671250
-49%	-12.6339967820271	-13.1061689081086
-48%	-12.3811000000000	-12.4305000000000
-47%	-11.6894002170653	-12.1812624685513
-46%	-11.0920932402210	-11.5255784495505
-45%	-10.5648950573354	-10.7179368834521
-44%	-10.0287000000000	-10.0506000000000
-43%	-9.44218399011077	-9.47487379216415
-42%	-8.81045602005936	-8.86066783732850
-41%	-8.22540801337291	-8.29670498641076
-40%	-7.79898667112430	-7.87170809032868
-39%	-7.63400000000000	-7.67440000000000
-38%	-7.23240000000000	-7.35440000000000
-37%	-7.21150000000000	-7.29450000000000
-36%	-6.92328323041739	-7.15066493555937
-35%	-6.72703132286040	-6.92242570149910
-34%	-6.55735875387321	-6.65322361668928
-33%	-6.34888000000000	-6.38650000000000
-32%	-6.04319041764771	-6.09987515675473
-31%	-5.77340000000000	-5.83810000000000
-30%	-5.62490000000000	-5.69140000000000
-29%	-5.43774479203109	-5.50446471740140
-28%	-5.22138796454573	-5.22711289285468
-27%	-4.92446584459934	-5.00592454725093
-26%	-4.69330000000000	-4.78710000000000
-25%	-4.56195777627420	-4.57920250473263
-24%	-4.35400000000000	-4.41990000000000
-23%	-4.02035921329322	-4.11431939227752
-22%	-3.78410000000000	-3.85000000000000

-21%	-3.56886586562124	-3.77467065794684
-20%	-3.36347555856579	-3.58527524158040
-19%	-3.17304251359429	-3.33670266559028
-18%	-2.98888456434467	-3.08384184466606
-17%	-2.84254431248445	-2.88158169349733
-16%	-2.74670000000000	-2.78340000000000
-15%	-2.50710000000000	-2.60400000000000
-14%	-2.29241881974953	-2.41538296086888
-13%	-2.07370617221822	-2.19355983097749
-12%	-1.88068411481215	-1.98168522065166
-11%	-1.74307470493738	-1.82291374021722
-10%	-1.69060000000000	-1.76040000000000
-9%	-1.53000000000000	-1.62000000000000
-8%	-1.36000000000000	-1.44000000000000
-7%	-1.10000000000000	-1.26000000000000
-6%	-0.920000000000000	-1.08000000000000
-5%	-0.850000000000000	-0.900000000000000
0%	-0.850000000000000	-0.900000000000000
0%	-0.430000000000000	-0.720000000000000
0%	-0.250000000000000	-0.540000000000000
0%	-0.100000000000000	-0.360000000000000
0%	0.120000000000000	-0.180000000000000
0%	0.300000000000000	-0.200000000000000
0%	0	0
0%	0.207758734380475	0.145709612732192
0%	0.403232625014601	0.296035633952512
0%	0.583440203141426	0.451018838196575
0%	0.745400000000000	0.610700000000000
4%	0.745400000000000	0.610700000000000
5%	0.886547663434025	0.778294508515028
6%	1.01195169033712	0.945980148908241
7%	1.13212933128968	1.08348961207881
8%	1.30859707703217	1.22086605016626
9%	1.47825000000000	1.36140000000000
10%	1.64673240873941	1.50745707235048
11%	1.82078754192701	1.65808275291719
12%	1.99717751049978	1.81158739092318
13%	2.17266442539468	1.96628133559148
14%	2.34401039754872	2.12047493614513
15%	2.50797753789885	2.27247854180717
16%	2.66132795738207	2.42060250180064
17%	2.80082376693534	2.56315716534858
18%	2.92322707749566	2.69845288167402
19%	3.02530000000000	2.82480000000000
20%	3.10650194840733	2.94441317598470
21%	3.17254614948278	3.06023972604339

22%	3.23790073858168	3.17168899053473
23%	3.34263518256265	3.27817030981739
24%	3.44583183164349	3.34116057018019
25%	3.54943774954800	3.40800208339045
26%	3.65540000000000	3.49040000000000
27%	3.77179391368344	3.60104239129568
28%	3.89369433185551	3.68909718099596
29%	4.00354758409982	3.78200000000000
30%	4.08380000000000	4.01920000000000
31%	4.12980000000000	4.07660000000000
32%	4.21128734475512	4.17849494926252
33%	4.34831745731033	4.31706337522832
34%	4.53350623670298	4.43886323771283
35%	4.68150000000000	4.55340000000000
36%	4.77556005150824	4.64170829090897
37%	4.86670000000000	4.73440000000000
38%	4.97367707601092	4.84329610297843
39%	5.08461079612768	4.95989160418827
40%	5.20261175868461	5.08560764409499
41%	5.33079056201601	5.22186536316406
42%	5.47225780445616	5.37008590186093
43%	5.63012408433939	5.53169040065109
44%	5.80750000000000	5.70810000000000
45%	6.05510000000000	5.93700000000000
46%	6.29380000000000	6.09440000000000
47%	6.47560696300625	6.20803327336216
48%	6.63033343605570	6.30482054510440
49%	6.75200000000000	6.40970000000000
50%	6.83743557272138	6.53104162389811
51%	6.90201805758141	6.66080605566489
52%	6.96759151365074	6.79931745959923
53%	7.05600000000000	6.94690000000000
54%	7.19757340697475	7.10821060641932
55%	7.38686339741988	7.28339378705125
56%	7.59204668915507	7.46600507415756
57%	7.78130000000000	7.64960000000000
58%	7.95402712181644	7.84171938737194
59%	8.12098942607309	8.03790193743425
60%	8.26740000000000	8.21370000000000
61%	8.38647407350068	8.36303722896108
62%	8.49769227875206	8.47701283377085
63%	8.62000000000000	8.58370000000000
64%	8.72790264269818	8.69219915481950
65%	8.82470304000972	8.79768441569937
66%	8.93540656028151	8.90517746872956
67%	9.05759000000000	9.01970000000000

68%	9.18921821335155	9.14178454573767
69%	9.32878089716048	9.26845705104118
70%	9.47442660714789	9.39947931530795
71%	9.62430389903485	9.53461313793539
72%	9.77656132854246	9.67362031832090
73%	9.92934745139179	9.81626265586190
74%	10.0808108233039	9.96230194995579
75%	10.2291000000000	10.1024000000000
76%	10.3774849523168	10.2312842834063
77%	10.5208000000000	10.3597000000000
78%	10.6605289547923	10.5011135315213
79%	10.7754000000000	10.6224000000000
80%	10.8562874708593	10.7008268561572
81%	10.9216887400126	10.7632314284020
82%	10.9644000000000	10.8194952864457
83%	10.9853000000000	10.8795000000000
84%	11.0352102589363	10.9463281242765
85%	11.1156231432212	11.0150135753734
86%	11.2102808980379	11.0833649643321
87%	11.3029257685697	11.1491909021938
88%	11.3773000000000	11.2103000000000
89%	11.4332009279642	11.2659458985584
90%	11.4825781554664	11.3176797228691
91%	11.5274023287427	11.3667563102620
92%	11.5696440940292	11.4144304980667
93%	11.6112740975620	11.4619571236131
94%	11.6542629855773	11.5105910242309
95%	11.7005814043113	11.5615870372500
96%	11.7522000000000	11.6162000000000
97%	11.8255335849977	11.6630638842565
98%	11.9041834591655	11.7136124018072
99%	11.9409000000000	11.8033000000000
100%	11.9337000000000	11.8716000000000

Table C.2: The Medium Frequency Filter characteristic's values.

C.3 LF characteristic

Percentage value of Modification	First Interval Value dB	Second Interval Value dB
-62%	-19.0703000000000	-20.2243000000000
-61%	-18.5833914328882	-19.5794681729037
-60%	-18.0537937551358	-18.9529195938297
-59%	-17.4704949476754	-18.3391779396152
-58%	-16.8927924028412	-17.7302600227745
-57%	-16.3209311601172	-17.1263593719025
-56%	-15.7551562589872	-16.5276695155941
-55%	-15.1957127389351	-15.9343839824442
-54%	-14.6428456394447	-15.3466963010479
-53%	-14.0968000000000	-14.7648000000000
-52%	-13.5539709754097	-14.1883868203363
-51%	-13.0116996392401	-13.6168800578861
-50%	-12.4716499091116	-13.0500705734345
-49%	-11.9354857026445	-12.4877492277669
-48%	-11.4048709374590	-11.9297068816683
-47%	-10.8814695311756	-11.3757343959241
-46%	-10.3669454014146	-10.8256226313193
-45%	-9.86296246579615	-10.2791624486394
-44%	-9.37118464194071	-9.73614470866934
-43%	-8.89327584746855	-9.19636027219448
-42%	-8.40450000000000	-8.67080000000000
-41%	-7.91377067744577	-8.13297288010112
-40%	-7.50230000000000	-7.68100000000000
-39%	-7.18880211914402	-7.35528055743475
-38%	-6.90150636990404	-7.05914719862558
-37%	-6.63725053614395	-6.78904970755982
-36%	-6.39287240172769	-6.54143786822479
-35%	-6.16520975051918	-6.33304199478783
-34%	-5.95110036638232	-6.14404762915826
-33%	-5.74738203318105	-5.96654577917391
-32%	-5.55089253477927	-5.79591363219519
-31%	-5.35846965504091	-5.62752837558249
-30%	-5.16695117782989	-5.45676719669622
-29%	-4.97317488701011	-5.27900728289678
-28%	-4.77397856644551	-5.08962582154457
-27%	-4.56620000000000	-4.88400000000000
-26%	-4.35297329106177	-4.66472019406329
-25%	-4.14050409878127	-4.43967600217921
-24%	-3.93023754066579	-4.21219344567496
-23%	-3.72361873422263	-3.98559854587771
-22%	-3.52209279695908	-3.76321732411467
-21%	-3.32710484638244	-3.54837580171304

-20%	-3.14010000000000	-3.34440000000000
-19%	-2.95868595779679	-3.15285430525850
-18%	-2.77966252156600	-2.96684318608078
-17%	-2.60326296152719	-2.78247852057058
-16%	-2.42972054789991	-2.59987218683162
-15%	-2.25926855090373	-2.41913606296760
-14%	-2.09214024075818	-2.24038202708225
-13%	-1.92856888768284	-2.06372195727929
-12%	-1.76878776189725	-1.88926773166244
-11%	-1.61303013362096	-1.71713122833542
-10%	-1.46152927307354	-1.54742432540194
-9%	-1.31451845047454	-1.38025890096574
-8%	-1.17223093604351	-1.21986165434049
-7%	-1.03490000000000	-1.06420000000000
-6%	-0.895130279677909	-0.919311744787470
-5%	-0.739249550267964	-0.781284950188059
-4%	-0.586469689873881	-0.654979364390194
-3%	-0.436902576599381	-0.539807444092343
-2%	-0.290660088548184	-0.430758453279794
0%	-0.147854103824009	-0.328065662172099
0%	-0.166635522173432	-0.463134762944904
0%	-0.00437422832543355	-0.264801091241378
0%	0.161309576767322	-0.0668045578036116
0%	0.325994700627076	0.130385288453359
4%	0.489559997498663	0.326298898614497
5%	0.651884321626915	0.520466723764767
6%	0.812846527256666	0.712419214989131
7%	0.972325468632750	0.901686823372555
8%	1.13020000000000	1.08630000000000
9%	1.28634897560325	1.23815669076639
10%	1.45839032090901	1.38871817786573
11%	1.62743601837106	1.53791325016490
12%	1.79148307797958	1.68567069653083
13%	1.95461519477129	1.83191930583040
14%	2.11194125364710	1.97658786693051
15%	2.26282603836169	2.11960516869808
16%	2.40680000000000	2.26090000000000
17%	2.54494292753403	2.40183807589910
18%	2.67877293271539	2.54331738431280
19%	2.80847795079849	2.68456512338985
20%	2.93424591703776	2.82480849127904
21%	3.05626476668761	2.96327468612913
22%	3.17472243500244	3.09919090608889
23%	3.28980685723668	3.22145865252957
24%	3.40170596864475	3.33140179780430
25%	3.51625893256221	3.44187117144573

26%	3.63250000000000	3.55380000000000
27%	3.75539888540421	3.66452003001996
28%	3.88933881476209	3.76771366130086
29%	4.02340395604915	3.86317326264311
30%	4.14667847724095	3.96027213895356
31%	4.24824654631300	4.06089720603349
32%	4.31719233124084	4.17650000000000
33%	4.34260000000000	4.31060000000000
34%	4.52933125379170	4.29133788140388
35%	4.70774663905590	4.43282253440275
36%	4.87313668203111	4.55160000000000
37%	5.02685738924707	4.61071393272720
38%	5.17026476723355	4.65445088096328
39%	5.30471482252029	4.69891238871772
40%	5.43156356163704	4.76020000000000
41%	5.55216699111354	4.85902834829521
42%	5.66788111747956	4.99953890959289
43%	5.78006194726484	5.17297349165917
44%	5.89006548699912	5.37057390226020
45%	5.99924774321216	5.58358194916212
46%	6.10896472243371	5.80323944013106
47%	6.22057243119352	6.02078818293319
48%	6.33542687602134	6.22746998533462
49%	6.46207133545334	6.40696409477377
50%	6.59880000000000	6.54470000000000
51%	6.68547167578184	6.63864479953683
52%	6.77469955465403	6.71919411557644
53%	6.96190000000000	6.89320000000000
54%	7.28108189021176	7.20757430400975
55%	7.53463636744206	7.41987544099706
56%	7.75555138952917	7.60455269501826
57%	7.94718585127895	7.76521751287934
58%	8.11289864749728	7.90548134138629
59%	8.25604867299002	8.02895562734508
60%	8.37999482256304	8.13925181756170
61%	8.48809599102221	8.23998135884214
62%	8.58371107317340	8.32977643519896
63%	8.67019896382248	8.41205516408312
64%	8.75091855777532	8.49839417946240
65%	8.82922874983779	8.59026244817563
66%	8.90848843481576	8.69342102209239
67%	8.99205650751509	8.81220000000000
68%	9.08329186274167	8.93959657025981
69%	9.18555339530134	9.06064292104567
70%	9.30220000000000	9.17890000000000
71%	9.43064033092018	9.31209954352934

72%	9.56437761338474	9.44614649760691
73%	9.70091259905995	9.56883211469371
74%	9.83774603961210	9.69150223003241
75%	9.97237868670749	9.81563903317024
76%	10.1023112920124	9.92979313868025
77%	10.2250446071931	10.0364875114353
78%	10.3380793839160	10.1516931630670
79%	10.4389163738472	10.2787255178353
80%	10.5250563286531	10.4209000000000
81%	10.6900000000000	10.5694000000000
82%	10.7680334027125	10.6229229152721
83%	10.8416851915392	10.6670071630684
84%	10.9109740675357	10.7048313608200
85%	10.9759187317577	10.7383853493304
86%	11.0365378852608	10.7613197050707
87%	11.0928502291007	10.7879409714197
88%	11.1448744643331	10.8225000000000
89%	11.1926292920137	10.8639340065855
90%	11.2361334131981	10.9083137922394
91%	11.2754055289420	10.9288843249226
92%	11.3104643403011	10.9463676889275
93%	11.3413285483310	10.9643715841673
94%	11.3680168540874	10.9849017526738
95%	11.3905479586260	11.0099639364785
96%	11.4359601160137	11.0415638776134
97%	11.4700839076289	11.0817073181099
98%	11.4824000000000	11.1324000000000
99%	11.4615441666667	11.2747710541493
100%	11.5034000000000	11.4165000000000

Table C.3: The Low Frequency Filter characteristic's values.

Bibliography

[1] L. Yang et al., *Music in the Air: Stairway to Heaven.*, The Goldman Sachs Group Inc., October 2016.

[2] B. Swinburne, *Revival — The Investment Case for Music.*, Morgan Stanley, April 2018.

[3] S.R. Vishwanath, *Corporate finance: theory and practice.*, Sage, London, 2007.

[4] Insight and Analysis, *Music Consumer Insight Report.*, IFPI, 2018.

[5] *Asia Pacific Dance Music Study.*, NIELSEN ENTERTAINMENT, 2017.

[6] *52 Places to Go in 2016.*, The New York Times, 2016.

[7] K. Watson, *IMS Business Report 2018: an annual study on the electronic music industry.*, International Music Summit, 2018.

[8] K. Watson, *IMS Business Report 2019: an annual study on the electronic music industry.*, International Music Summit, 2019.

[9] S. Ciocca, *How Does Spotify Know You So Well?*, Medium, October 2017.

[10] S. Bird et al., *Natural Language Processing with Python.*, O'REILLY, USA, 2009.

[11] ISO IEC TR 21000-11 (2004), Multimedia framework (MPEG-21) – Part 11: Evaluation Tools for Persistent Association Technologies.

[12] C. Johnson, *From idea to execution: Spotify's Discover Weekly.*, DataEngConf, New York City, 2015.

[13] G. James et al., *An Introduction to Statistical Learning with Applications in R.*, Springer, USA, 2013.

[14] S. Gupta *Sentiment Analysis: Concept, Analysis and Applications.*, Towards Data Science, 2018.

[15] S. Dieleman, *Recommending music on Spotify with deep learning.*, GitHub, 2014.

[16] *Convolutional Neural Network for Visual Recognition.*, CS231n of Stanford University, http://cs231n.github.io/

[17] S. Bouguezal et al., *An Efficient Algorithm for the Computation of the Multidimensional Discrete Fourier Transform.*, Multidimensional Systems and Signal Processing 10, 275–304 (1999).

[18] K.R. Rao et al., *Fast Fourier Transform - Algorithms and Applications.*, Springer, Dordrecht, 2010.

[19] S.S. Stevens et al., *A Scale for the Measurement of the Psychological Magnitude Pitch.*, Journal of the Acoustical Society of America Volume 8, pp. 185–190, 1937.

[20] I.V. McLoughlin, *Speech and Audio Processing: A MATLAB-based approach.*, Cambridge University Press, Cambridge, 2016.

[21] R. Sobot, *Wireless Communication Electronics: Introduction to RF Circuits and Design*, Springer, New York, 2012.

[22] E. Sejdić et. al, *Time-Frequency Feature Representation Using Energy Concentration: An Overview of Recent Advances.*, Digital Signal Processing Volume 19, pp. 153-183, 2009.

[23] S.G. Mallat, *A Wavelet Tour of Signal Processing.*, San Diego Academic Press, San Diego, 1999.

[24] I. Daubechies, *The Wavelet Transform, Time-Frequency Localization and Signal Analysis.*, IEEE Transactions On Information Theory Volume 36, pp. 961 − 1005,1990.

[25] H. Li and Y. Kiang, *Radar and Inverse Scattering.*, The Electrical Engineering Handbook ch. 10, pp 671–690, Academic Press, 2004.

[26] L. Cohen, *Time-Frequency Analysis.*, N.J: Prentice Hall PTR, Englewood Cliffs, 1995.

[27] S. Qian and D. Chen, *Decomposition of the Wigner-Ville Distribution and Time-Frequency Distribution Series.*, IEEE Transaction On Signal Processing Volume 42, pp. 2386–2842, 1994.

[28] M.J. Levin, *Instantaneous spectra and ambiguity functions.* IEEE Transaction On Information Theory Volume 10, pp. 95–97, 1964.

[29] R.D. Hippenstiel and P.M. De Oliveira, *Time-Varying Spectral Estimation Using the Instantaneous Power Spectrum (IPS).*, IEEE Transactions On Acoustic Speech and Signal Processing Volume 38, pp. 1752–1759, 1990.

[30] J.C. Wood and D.T. Barry, *Randon Transformation of Time-Frequency Distributions for Analysis of Multicomponent Signals.*, IEEE Transaction On Signal Processing Volume 41, pp 3166–3177, 1994.

[31] G.T. Herman, *Image Reconstruction from Projections: The Fundamentals of Computerized Tomography.*, New York Academic Press, New York, 1980.

[32] R.A. Carmona et al., *Multiridge Detection and Time-Frequency Reconstruction.*, IEEE Transaction On Signal Processing Volume 47, pp. 480–492, 1999.

[33] D. Gamerman et al., *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference.*, Chapman & Hall/CRC, Boca Raton, 2006.

[34] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques.*, MIT Press Ltd, USA, 2009.

[35] N. Baydar and A. Ball, *A Comparative Study of Acoustic and Vibration Signals in Detection of Gear Failures using Wigner-Ville Distribution.*, Mechanical Systems and Signal Processing, pp. 1091–1107, Academic Press, 2001.

[36] R. Graf, *Modern Dictionary of ELECTRONICS Seventh Edition.*, Newens, Burlington, 1999.

[37] A.V. Oppenheim and R.W. Schafer, *Digital Signal Processing.*, Prentice Hall, Englewood Cliffs, 1975.

[38] S. Haskel and D. Sygoda, *Biology, A contemporary Approach.*, Amsco, New York, 1996.

[39] D. Folgado et al., *Time Alignment Measurement for Time Series.*, Pattern Recognition 81, pp. 268–279, 2018.

[40] D. Rothmann, *Human-Like Machine Hearing With AI (1/3).*, Towards Data Science, 2018.

[41] Q. Jun, *Auditory Features Based on Gammatone Filters for Robust Speech Recognition.*, IEEE International Symposium on Circuits and Systems (ISCAS), 2013.

[42] D. Lavry, *Understanding FIR (Finite Impulse Response) Filters - An Intuitive Approach.*, Lavry Engineering, 1997.

[43] M. Slaney, *An Efficient Implementation of the Patterson-Holdsworth Auditory Filter Bank.*, Apple Computer Technical Report 35, Advanced Technology Group, 1993.

[44] S. Strahl and A. Mertins, *Analysis and design of gammatone signal models.*, The Journal of the Acoustical Society of America 126, pp. 2379–2389, 2009.

[45] B. Moore et al., *Auditory filter shapes at low center frequencies.*, The Journal of the Acoustical Society of America 88, pp. 132–140, 1990.

[46] L. Chaparro and A. Akan, *Discrete-Time Signals and Systems.*, Signals and Systems Using MATLAB (Third edition), pp. 487–557, 2019.

[47] A. Abraham, *Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification.*, International Journal of Advanced Research in Artificial Intelligence 2, 2013.

[48] P. Berkhin, *A Survey of Clustering Data Mining Techniques.*, Kogan J., Nicholas C., Teboulle M. (eds) Grouping Multidimensional Data, pp. 25–71, Springer, 2006.

[49] D. Soni, *Supervised vs. Unsupervised Learning.*, Towards Data Science, 2018.

[50] P.W. Holland & R.E. Welsch, *Robust regression using iteratively reweighted least-squares.*, Communications in Statistics - Theory and Methods, 1977.

[51] J. Clark et. al, *Adaptive Threshold for Outlier Detection on Data Streams.*, IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA), 2018.

[52] R. Killick et al., *Optimal Detection of Changepoints With a Linear Computational Cost.*, Journal of the American Statistical Association, pp. 1590–1590, 2012.

[53] S. Kullback and R.A. Leibler, *On Information and Sufficiency.*, Annals of Mathematical Statistics Volume 22, pp. 79-–86,1951.

[54] K.P. Burnham and D.R. Anderson, *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach.*, Springer-Verlag, New-York, 2002.

[55] C.E. Shannon, *A Mathematical Theory of Communication.*, The Bell System Technical Journal Volume 27, pp. 379–Not Modelled3, 623—656, 1948.

[56] C.G. Chakrabarti and K. De, *Boltzmann Entropy: Generalization and Applications.*, Journal of Biological Physics volume 23, pp. 163—170, 1997.

[57] J.A. Cherry, *Distortion Analysis of Weakly Nonlinear Filters Using Volterra Series*, Carleton University, Canada, 1994.

[58] S.K Remillard et al., *Three-Tone Intermodulation Distortion Generated by Superconducting Bandpass Filters*, IEEE Transaction On Applied Superconductivity Volume 13, pp. 3797–3802, 2003.

[59] E. Coates, *Negative Feedback and Distortions*, in https://learnabout-electronics.org, Amplifiers Module 3.4, 2012.

[60] S.P. Mead, *Phase Distortion and Phase Distortion Correction.*, Bell System Technical Journal Volume 7, pp. 195–224, 1928.

[61] F.E. Toole, *The Acoustics and Psychoacoustics of Loudspeakers and Rooms - The Stereo Past and the Multichannel Future.*, 109th AES Conv., Los Angeles, 2000.

[62] W.M. Hartmann, *Signals, Sound, and Sensation.*, AIP Press, U.S.A., 1998.

[63] C. Herff and D.J. Krusienski, *Extracting Features from Time Series.*, Fundamentals of Clinical Data Science,pp. 85–100, Springer, Cham, 2018.

[64] L.C. Marsh and D.C. Cormier, *Spline Regression Models.*, Sage Publications, U.S.A., 2001.

[65] S. Garavaglia and A. Sharma, *A SMART GUIDE TO DUMMY VARIABLES: Four Applications and a Macro.*, Dun & Bradstreet, Murray Hill, New Jersey, 2003.

[66] C.R. de Boor, *A Practical Guide to Splines.*, Springer, New York, 2001.
[67] S. McKinley and M. Levine, *Cubic Spline Interpolation*, College of the Redwoods, 1998.
[68] *csaps: Cubic smoothing spline*, Matlab Documentation.

Acknowledgements

First of all, I would like to thank my parents, without whom I could never have achieved this goal: their tenacity and their sacrifices are really the backbone of this work and of future ones.
I will always carry with me a silent teaching that they transmitted to me: the very strong ability to adapt and react to adversity.

Marta: without you who kept telling me that I would have made it, every time I was faced with a new obstacle, the work would not have come to a conclusion. Obviously I have you to thank for many more things, but this is not the place.

A special thanks, in this case, goes to Professor Taragna who has shown an unexpectedly high interest in the work, assisted by a constant and genuine availability: qualities that, incidentally, one could have guessed by following his class and which, however, exceeded expectations.

I cannot miss a thought for those who have been close to me, during these years of study, and who have shared with me the joys and difficulties (especially difficulties) that the Politecnico entails. In particular the coach, Matteo Giovannetti, since there is a part of him in this thesis: his enthusiasm and his skills allowed to make this dream come alive.